Traitement distribué d'informations réparties dans les réseaux d'ordinateurs

Jean Seguin

▶ **To cite this version:**

Jean Seguin. Traitement distribué d'informations réparties dans les réseaux d'ordinateurs. Modélisation et simulation. Institut National Polytechnique de Grenoble - INPG; Université Joseph-Fourier - Grenoble I, 1978. tel-00288182

HAL Id: tel-00288182
https://theses.hal.science/tel-00288182

Submitted on 16 Jun 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THESE

présentée à

Université Scientifique et Médicale de Grenoble

Institut National Polytechnique de Grenoble

pour obtenir le grade de
Docteur d'Etat ès- Sciences Mathématiques

par

Jean SEGUIN

TRAITEMENT DISTRIBUE D'INFORMATIONS REPARTIES DANS LES RESEAUX D'ORDINATEURS

Thèse soutenue le 9 mars 1978 devant la Commission d'Examen

Président : L. BOLLIET

Examinateurs : C. DELOBEL
M. GRIFFITHS
C. KAISER
J. LE BIHAN
J.P. VERJUS

UNIVERSITE SCIENTIFIQUE
ET MEDICALE DE GRENOBLE

Monsieur Gabriel CAU : Président
Monsieur Pierre JULLIEN : Vice Président

MEMBRES DU CORPS ENSEIGNant : DE L'U.S.M.G.

PROFESSEURS TITULAIRES

MM	AMBLARD Pierre	Clinique de dermatologie
	ARNAUD Paul	Chimie
	ARVIEU Robert	I.S.N
	AUBERT Guy	Physique
	AYANT Yves	Physique approfondie
Mme	BARBIER Marie-Jeanne	Electrochimie
MM.	BARBIER Jean-Claude	Physique Expérimentale
	BARBIER Reynold	Géologie appliquée
	BARJON Robert	Physique nucléaire
	BARNOUD Fernand	Biosynthèse de la cellulose
	BARRA Jean-René	Statistiques
	BARRIE Joseph	Clinique chirurgicale
	BEAUDOING André	Clinique de Pédiatrie et Puériculture
	BELORIZKY Elie	Physique
	BERNARD Alain	Mathématiques Pures
Mme	BERTRANDIAS Françoise	Mathématiques Pures
MM.	BERTRANDIAS Jean-Paul	Mathématiques Pures
	BEZEZ Henri	Pathologie chirurgicale
	BLAMBERT Maurice	Mathématiques Pures
	BOLLIET Louis	Informatique (IUT B)
	BONNET Jean-Louis	Clinique ophtalmologique
	BONNET-EYMARD Joseph	Clinique gastro-entérologique
Mme	BONNIER Marie-Jeanne	Chimie générale
MM.	BOUCHERLE André	Chimie et toxicologie
	BOUCHEZ Robert	Physique nucléaire
	BOUSSARD Jean-Claude	Mathématiques appliquées
	BOUTET DE MONTVEL Louis	Mathématiques Pures
	BRAVARD Yves	Géographie
	CABANEL Guy	Clinique rhumatologique et hydrologique
	CALAS François	Anatomie
	CARLIER Georges	Biologie végétale
	CARRAZ Gilbert	Biologie animale et pharmacodynamie
	CAU Gabriel	Médecine légale et toxicologie
	CAUQUIS Georges	Chimie organique
	CHABAUTY Claude	Mathématiques Pures
	CHARACHON Robert	Clinique Oto-rhino-laryngologique
	CHATEAU Robert	Clinique de neurologie
	CHIBON Pierre	Biologie animale
	COEUR André	Pharmacie chimique et chimie analytique
	CONTAMIN Robert	Clinique gynécologique
	COUDERC Pierre	Anatomie pathologique
Mme	DEBELMAS Anne-Marie	Matière médicale
MM.	DEBELMAS Jacques	Géologie générale
	DEGRANGE Charles	Zoologie
	DELORMAS Pierre	Pneumophtisiologie

MM.	DEPORTES Charles	Chimie minérale
	DESRE Pierre	Métallurgie
	DESSAUX Georges	Physiologie animale
	DODU Jacques	Mécanique appliquée (IUT I)
	DOLIQUE Jean-Michel	Physique des plasmas
	DREYFUS Bernard	Thermodynamique
	DUCROS Pierre	Cristallographie
	GAGNAIRE Didier	Chimie Physique
	GALVANI Octave	Mathématiques Pures
	GASTINEL Noël	Analyse numérique
	GAVEND Michel	Pharmacologie
	GEINDRE Michel	Electroradiologie
	GERBER Robert	Mathématiques Pures
	GERMAIN Jean-Pierre	Mécanique
	GIRAUD Pierre	Géologie
	JANIN Bernard	Géographie
	KAHANE André	Physique générale
	KLEIN Joseph	Mathématiques Pures
	KOSZUL Jean-Louis	Mathématiques Pures
	KRAVTCHENKO Julien	Mécanique
	KUNTZMANN Jean	Mathématiques Appliquées
	LACAZE Albert	Thermodynamique
	LACHARME Jean	Biologie végétale
Mme	LAJZEROWICZ Janine	Physique
MM.	LAJZEROWICZ Joseph	Physique
	LATREILLE René	Chirurgie générale
	LATURAZE Jean	Biochimie Pharmaceutique
	LAURENT Pierre	Mathématiques Appliquées
	LEDRU Jean	Clinique médicale B
	LE ROY Philippe	Mécanique (IUT I)
	LLIBOUTRY Louis	Géophysique
	LOISEAUX Pierre	Sciences Nucléaires
	LONGEQUEUE Jean-Pierre	Physique Nucléaire
	LOUP Jean	Géographie
Melle	LUTZ Elisabeth	Mathématiques Pures
MM.	MALINAS Yves	Clinique Obstétricale
	MARTIN-NOEL Pierre	Clinique Cardiologique
	MAZARE Yves	Clinique Médicale A
	MICHEL Robert	Minéralogie et Pétrographie
	MICOUD Max	Clinique Maladies infectieuses
	MOURIQUAND Claude	Histologie
	MOUSSA André	Chimie Nucléaire
	NOZIERES Philippe	Spectrometrie Physique
	OZENDA Paul	Botanique
	PAYAN Jean-Jacques	Mathématiques Pures
	PEBAY-PEYROULA Jean-Claude	Physique
	PERRET Jean	Semeiologie Médicale (Neurologie)
	RASSAT André	Chimie systématique
	RENARD Michel	Thermodynamique
	REVOL Michel	Urologie
	RINALDI Renaud	Physique
	DE ROUGEMONT Jacques	Neuro-Chirurgie
	SEIGNEURIN Raymond	Microbiologie et Hygiène
	SENGEL Philippe	Zoologie
	SIBILLE Robert	Construction mécanique (IUT I)
	SOUTIF Michel	Physique générale
	TANCHE Maurice	Physiologie
	TRAYNARD Philippe	Chimie générale

MM.	VAILLANT François	Zoologie
	VALENTIN Jacques	Physique Nucléaire
	VAUQUOIS Bernard	Calcul électronique
Mme	VERAIN Alice	Pharmacie galénique
MM.	VERAIN André	Physique
	VEYRET Paul	Géographie
	VIGNAIS Pierre	Biochimie médicale

PROFESSEURS ASSOCIES

MM.	CRABBE Pierre	CERMO
	DEMBICKI Eugéniuz	Mécanique
	JOHNSON Thomas	Mathématiques appliquées
	PENNEY Thomas	Physique

PROFESSEURS SANS CHAIRE

Mle	AGNIUS-DELORD Claudine	Physique pharmaceutique
	ALARY Josette	Chimie analytique
MM.	AMBROISE-THOMAS Pierre	Parasitologie
	ARMAND Gilbert	Géographie
	BENZAKEN Claude	Mathématiques appliquées
	BIAREZ Jean-Pierre	Mécanique
	BILLET Jean	Géographie
	BOUCHET Yves	Anatomie
	BRUGEL Lucien	Energétique (IUT I)
	BUISSON René	Physique (IUT I)
	BUTEL Jean	Orthopédie
	COHEN ADDAD Pierre	Spectrométrie physique
	COLOMB Maurice	Biochimie
	CONTE René	Physique (IUT I)
	DELOBEL Claude	M.I.A.G.
	DEPASSEL Roger	Mécanique des fluides
	FONTAINE Jean-Marc	Mathématiques Pures
	GAUTRON René	Chimie
	GIDON Paul	Géologie et Minéralogie
	GLENAT René	Chimie organique
	GROULADE Joseph	Biochimie médicale
	HACQUES Gérard	Calcul numérique
	HOLLARD Daniel	Hématologie
	HUGONOT Robert	Hygiène et Médecine préventive
	IDELMAN Simon	Physiologie animale
	JOLY Jean-René	Mathématiques Pures
	JULLIEN Pierre	Mathématiques Appliquées
Mme	KAHANE Josette	Physique
MM.	KRAKOWIACK Sacha	Mathématiques Appliquées
	KUHN Gérard	Physique (IUT I)
	LUU DUC Cuong	Chimie organique
	MAYNARD Roger	Physique du solide
Mme	MINIER Colette	Physique (IUT I)
MM.	PELMONT Jean	Biochimie
	PERRIAUX Jean-Jacques	Géologie et Minéralogie
	PFISTER Jean-Claude	Physique du solide
Mle	PIERY Yvette	Physiologie animale

MM.	RAYNAUD Hervé	M.I.A.G.
	REBECQ Jacques	Biologie (CUS)
	REYMOND Jean-Charles	Chirurgie générale
	RICHARD Lucien	Biologie végétale
Mme	RINAUDO Marguerite	Chimie macromoléculaire
MM.	ROBERT André	Chimie papetière
	SARRAZIN Roger	Anatomie et chirurgie
	SARROT-REYNAULD Jean	Géologie
	SIROT Louis	Chirurgie générale
Mme	SOUTIF Jeanne	Physique générale
MM.	STIEGLITZ Paul	Anesthésiologie
	VIALON Pierre	Géologie
	VAN CUTSEM Bernard	Mathématiques Appliquées

MAITRES DE CONFERENCES ET MAITRES DE CONFERENCES AGREGES

MM.	ARMAND Yves	Chimie (IUT I)
	BACHELOT Yvan	Endocrinologie
	BARGE Michel	Neuro chirurgie
	BEGUIN Claude	Chimie organique
Mme	BERIEL Hélène	Pharmacodynamie
MM.	BOST Michel	Pédiatrie
	BOUCHARLAT Jacques	Psychiatrie adultes
Mme	BOUCHE Liane	Mathématiques (CUS)
MM.	BRODEAU François	Mathématiques (IUT B) (Personne étrangère habilitée à être directeur de thèse)
	CHAMBAZ Edmond	Biochimie médicale
	CHAMPETIER Jean	Anatomie et organogénèse
	CHARDON Michel	Géographie
	CHERADAME Hervé	Chimie papetière
	CHIAVERINA Jean	Biologie appliquée (EFP)
	CONTAMIN Charles	Chirurgie thoracique et cardio-vasculaire
	CORDONNIER Daniel	Néphrologie
	COULOMB Max	Radiologie
	CROUZET Guy	Radiologie
	CYROT Michel	Physique du solide
	DENIS Bernard	Cardiologie
	DOUCE Roland	Physiologie végétale
	DUSSAUD René	Mathématiques (CUS)
Mme	ETERRADOSSI Jacqueline	Physiologie
MM.	FAURE Jacques	Médecine légale
	FAURE Gilbert	Urologie
	GAUTIER Robert	Chirurgie générale
	GIDON Maurice	Géologie
	GROS Yves	Physique (IUT I)
	GUIGNIER Michel	Thérapeutique
	GUITTON Jacques	Chimie
	HICTER Pierre	Chimie
	JALBERT Pierre	Histologie
	JUNIEN-LAVILLAVROY Claude	O.R.L.
	KOLODIE Lucien	Hématologie
	LE NOC Pierre	Bactériologie-virologie
	MACHE Régis	Physiologie végétale
	MAGNIN Robert	Hygiène et médecine préventive
	MALLION Jean-Michel	Médecine du travail

MM.	MARECHAL Jean	Mécanique (IUT I)
	MARTIN-BOUYER Michel	Chimie (CUS)
	MICHOULIER Jean	Physique (IUT I)
	NEGRE Robert	Mécanique (IUT I)
	NEMOZ Alain	Thermodynamique
	NOUGARET Marcel	Automatique (IUT I)
	PARAMELLE Bernard	Pneumologie
	PECCOUD François	Analyse (IUT B) (Personnalité étrangère habilitée à être directeur de thèse)
	PEFFEN René	Métallurgie (IUT I)
	PERRIER Guy	Géophysique-Glaciologie
	PHELIP Xavier	Rhumatologie
	RACHAIL Michel	Médecine Interne
	RACINET Claude	Gynécologie et Obstétrique
	RAMBAUD André	Hygiène et Hydrologie (Pharmacie)
	RAMBAUD Pierre	Pédiatrie
	RAPHAEL Bernard	Stomatologie
Mme	RENAUDET JAcqueline	Bactériologie (Pharmacie)
MM	ROBERT Jean-Bernard	Chimie Physique
	ROMIER Guy	Mathématiques (IUT B) (Personnalité étrangère habilité à être directeur de thèse)
	SCHAERER René	Cancérologie
	SHOM Jean-Claude	Chimie Générale
	STOEBNER Pierre	Anatomie Pathologie
	VROUSOS Constantin	Radiologie

MAITRESDE CONFERENCESASSOCIES

MM.	DEVINE Roderick	Spectro Physique
	HODGES Christopher	Transition de Phases

Fait à SAINT MARTIN D'HERES, NOVEMBRE 1976

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE — ANNEE 1976-1977

Président : M. Philippe TRAYNARD

Vice-Présidents : M. Pierre-Jean LAURENT
M. René PAUTHENET

PROFESSEURS TITULAIRES

MM.	BENOIT Jean	Radioélectricité
	BESSON Jean	Electrochimie
	BLOCH Daniel	Physique du solide
	BONNETAIN Lucien	Chimie minérale
	BONNIER Etienne	Electrochimie et Electrométallurgie
	BRISSONNEAU Pierre	Physique du solide
	BUYLE-BODIN Maurice	Electronique
	COUMES André	Radioélectricité
	DURAND Francis	Métallurgie
	FELICI Noël	Electrostastique
	FOULARD Claude	Automatique
	LESPINARD Georges	Mécanique
	MOREAU René	Mécanique
	PARIAUD Jean-Charles	Chimie-Physique
	PAUTHENET René	Physique du solide
	PERRET René	Servomécanismes
	POLOUJADOFF Michel	Electrotechnique
	VEILLON Gérard	Informatique fondamentale et appliquée

PROFESSEURS SANS CHAIRE

MM.	BLIMAN Samuel	Electronique
	BOUVARD Maurice	Génie Mécanique
	COHEN Joseph	Electrotechnique
	LACOUME Jean-Louis	Géophysique
	LANCIA Roland	Electronique
*	ROBERT François	Analyse numérique
	ZADWORNY François	Electronique
*	ROBERT André	Chimie - papetière

MAITRES DE CONFERENCES

MM.	ANCEAU François	Mathématiques Appliquées
	CHARTIER Germain	Electronique
	GUYOT Pierre	Chimie Minérale
	IVANES Marcel	Electrotechnique
	JOUBERT Jean-Claude	Physique du solide
	LESIEUR Marcel	Mécanique
	MORET Roger	Electrotechnique Nucléaire
	PIAU Jean-Michel	Mécanique
	PIERRARD Jean-Marie	Mécanique
	SABONNADIERE Jean-Claude	Informatique Fondamentale et Appliquée
MMe.	SAUCIER Gabrièle	Informatique Fondamentale et Appliquée

CHERCHEURS DU C.N.R.S. (Directeur et Maîtres de Recherche).

M.	FRUCHART Robert	Directeur de Recherche
MM.	ANSARA Ibrahim	Maître de Recherche
	CARRE René	Maître de Recherche
	DRIOLE Jean	Maître de Recherche
	LANDAU Ioan Doré	Maître de Recherche
	MATHIEU Jean-Claude	Maître de Recherche
	MUNIER Jacques	Maître de Recherche

Depuis 1972, date à laquelle j'ai commencé les travaux qui constituent la matière de cette thèse, j'ai été successivement ingénieur de recherches au Centre Scientifique CII (puis CII-Honeywell Bull), puis ingénieur C.N.R.S. au Laboratoire d'Informatique de Grenoble (I.M.A.G.), en fonction au Centre Interuniversitaire de Calcul de Grenoble (C.I.C.G.). Je tiens à remercier toutes celles et tous ceux qui ont participé avec moi à ces travaux de recherche, qui m'ont aidé à mener à bien cette thèse et qui ont fait partie de mon jury.

Au Centre Scientifique CII de Grenoble, j'ai apprécié la confiance que m'a toujours témoignée Monsieur L. Bolliet ; le travail d'équipe que nous avons réalisé avec E. André, J.C. Chupin, P. Decitre, G. Bogo, J.P. Metzger et P.A. Pays a été pour moi d'une grande richesse aussi bien sur le plan technique qu'amical.

Au Laboratoire I.M.A.G., j'ai fait un bon bout de chemin avec G. Sergeant (ainsi que quelques tours de roues !). Ng. X. Dang, V. Quint, H. Richy et P. Wilms ont, à plusieurs titres, contribué aux succès de projets développés ou mentionnés dans cette thèse. Je n'oublierai pas l'enthousiasme de J.P. Verjus qui m'a encouragé dans la rédaction de ce manuscrit... et qui a accepté de le critiquer et de le juger sans complaisance. C. Delobel a su contribuer à l'enrichissement de ce travail aussi bien par son soutien permanent, même dans des circonstances difficiles, que par ses critiques constructives.

Dans le cadre du projet pilote de l'I.R.I.A. : CYCLADES, j'ai apprécié les nombreuses discussions que nous avons eues avec les centres participants du réseau comme avec M. Gien, L. Pouzin et H. Zimmermann. Le sens du travail d'équipe et l'amitié de J. Le Bihan m'ont été précieux aussi bien dans Cyclades que dans la Commission Scientifique qui a préparé (avec H. Gallaire, P. Kalfon et J.P. Villard) le lancement du projet pilote SIRIUS.

Longue vie au groupe CORNAFION dont les 14 de cordée espèrent faire toute la lumière sur les systèmes répartis. D'ores et déjà notre travail en commun constitue pour moi un capital inestimable, lentement

accumulé par les Rennais de l'I.R.I.S.A. : D. Hermann, F. Kerangueven, G. Le Lann et J.P. Verjus, les Parisiens de l'I.R.I.A. : J.S. Banino, C. Kaiser, D. Lanciaux, le Montpelliérain J. Ferrié, le Toulousain C. Bétourne, les Grenoblois de l'I.M.A.G. : S. Krakowiak, G. Mazaré, J. Mossière et X. Rousset de Pina.
Je remercie C. Kaiser pour sa contribution positive à l'élaboration de ce document.

Grâce à la compréhension des responsables et des ingénieurs du C.I.C.G., j'ai pu terminer ce travail de thèse laissant parfois à d'autres certaines tâches .

J'exprime mon amical remerciement à M. Griffiths pour sa participation à mon jury de thèse.

J'exprime toute ma gratitude à F. Blanc, C. Chaland et au service tirage qui ont eu le mérite de la préparation matérielle de ce document.

S O M M A I R E

T A B L E D E S M A T I E R E S

TABLE DES FIGURES

1. INTRODUCTION

1.1. Environnement de travail

Le travail dont il est question dans cette thèse trouve ses racines dans celui développé depuis 1972 par l'équipe "Réseaux d'Ordinateurs et Bases de Données Réparties" de l'ENSIMAG et du Centre Scientifique CII (puis CII-HB) à laquelle j'ai appartenu pendant plus de quatre ans.

En 1971-1972, les premiers réseaux d'ordinateurs démarraient ; les seules études complètes et les seules expérimentations significatives étaient menées outre Atlantique dans le cadre de quelques grands projets dont celui de l'ARPA.

Pour développer des études sur ce thème, cette équipe a bénéficié d'une triple chance :

. d'une part, elle a toujours possédé ou acquis les moyens de mener à bien ses études et ses réalisations de façon suivie ;

. d'autre part, étant aux confluents de la recherche universitaire (Centre Interuniversitaire de Calcul, ENSIMAG, Université Scientifique et Médicale) et de la recherche industrielle (CII), elle a été au fait des réalités scientifiques et industrielles ;

. enfin, elle a été partie prenante de projets de recherche et de réalisations dans le domaine des réseaux et de leur utilisation, et d'abord de deux projets pilotes de l'IRIA : CYCLADES [IRI73], bases de données réparties SIRIUS [IRI76].

Dans ce cadre général, plusieurs thèmes de recherche ont été dégagés et retenus :

Thème 1 :
Le logiciel de base pour la téléinformatique, la communication et le transport d'informations ;

Thème 2 :
Les applications utilisant un réseau d'ordinateurs comme support matériel ;

Thème 3 :

Les systèmes de gestion de bases de données réparties sur un réseau général d'ordinateurs.

1.2. Objectifs et motivations de la recherche

Ces trois thèmes ne sont pas indépendants et une hiérarchie fonctionnelle existe entre eux. Les bases de données sont des applications qui utilisent les logiciels de communication pour réaliser leurs échanges au travers des réseaux d'ordinateurs.

Au départ, le problème qui nous était posé était de faire fonctionner un système de gestion de bases de données (SGBD) sur un réseau d'ordinateurs. Ce problème est très important et se révèle sous de multiples facettes :

- adaptation de systèmes existants aux réseaux,
- conception de bases de données réparties,
- architecture de systèmes de gestion distribués sur plusieurs sites,
- nature des échanges,
- nature des services rendus aux utilisateurs du réseau,
- indépendance par rapport aux réseaux de communication utilisés comme supports.

Ce problème reste toujours d'actualité et de nombreuses équipes de recherche poursuivent leurs investigations... dans l'attente d'une confrontation fructueuse avec les utilisateurs.

Pour notre part, ce thème général a fait l'objet d'un grand nombre de découpages permettant d'offrir des solutions partielles, en particulier pour les fichiers, c'est-à-dire en mettant de côté pour études ultérieures, les problèmes posés par la conjonction de la structure des données manipulées et d'un réseau de communication.

Ces découpages successifs ont souvent été guidés par des considérations extérieures. Comme on pourra le voir dans la suite, certaines "conditions initiales" ont justifié les priorités données :

(a) l'état d'avancement du réseau général d'ordinateurs accessible où la priorité des moyens était donnée au réseau de commutation de paquets et aux stations de transport ;

(b) les logiciels et matériels téléinformatiques offerts par les constructeurs, très variables et disponibles sur des systèmes très refermés sur eux-mêmes ;

(c) les systèmes de gestion de fichiers et de bases de données disponibles, caractérisés par une grande disparité dans les spécifications et les services offerts.

1.3. Contribution personnelle : étude, réalisations et publications

Depuis cinq ans l'effectif de l'équipe "Réseaux d'Ordinateurs et Bases de Données Réparties" a été en moyenne de quatre personnes. Ma contribution personnelle concerne plus particulièrement :

- les fichiers dans les réseaux : méthodes d'accès, dénomination, adressage, allocation,...
- les méthodes d'accès réseau : fichiers, bases de données,...
- la duplication de l'information,
- les outils de réalisation pour les applications réparties et la téléinformatique et en particulier :
  - l'écriture de langages portables
  - les sous-systèmes de traitement de processus téléinformatiques.
- les modèles de coopération de bases de données.

Sur ces cinq points, différents travaux ont été menés à bien, notamment à l'occasion de contrats de recherche dans le cadre du projet CYCLADES (DRME, SESORI) et du projet pilote bases de données réparties : MADRE, MARS, SOCRATE CYCLADES, SYNCOP, FANNY, POLYPHEME, SYNDIC. Chacun de ces projets représente un morceau de la matière de cette thèse.

Nous indiquons à la fin du présent document les différentes publications auxquelles ces projets ont donné naissance.

Les publications y sont présentées dans un ordre chronologique avec une mention :

a : papier d'ordre général orienté vers une formalisation de concepts sur un thème donné ;

b : papier d'ordre général axé sur la présentation d'un travail particulier ;

c : rapport technique.

Dans cette thèse, nous mettrons l'accent sur quelques grands problèmes existant dans le traitement distribué de l'information dans les réseaux, et présenterons quelques solutions illustrées par nos travaux.

Si le lecteur voit dans ce document une introduction attrayante à l'informatique répartie, notre objectif sera atteint : une contribution aux logiciels des réseaux informatiques.

1.4. Terminologie

Dans le texte qui suit, plusieurs termes techniques seront maintes fois utilisés. Ils méritent ici une première définition et apparaîtront pour ce travail comme de véritables mots-clés.

Téléinformatique :

technique d'utilisation à distance d'un ordinateur grâce à des circuits téléphoniques ou télégraphiques.

Réseau d'ordinateurs :

ensemble d'ordinateurs interconnectés mettant en commun un ensemble de ressources.

Informatique répartie :

technique de traitement automatisé de l'information mettant en oeuvre plusieurs processeurs d'un système multiprocesseur ou plusieurs ordinateurs d'un réseau d'ordinateurs.

Réseau informatique :

ensemble des traitements de l'information rendus possible par une configuration téléinformatique ou un réseau d'ordinateurs.

Protocole :

ensemble de règles régissant les échanges entre deux correspondants (en particulier utilisant le réseau comme support d'échanges).

Site :

ordinateur connecté à un réseau de communication et constituant avec d'autres un réseau d'ordinateurs.

Méthode d'accès réseau :

service réseau mettant à la disposition de l'utilisateur les moyens d'accès à des données résidantes sur des sites quelconques.

Fichier :

collection d'informations cataloguées sur un support quelconque et susceptible d'être utilisées ultérieurement.

Fichier à accès réparti :

fichier résidant sur un site et dont l'accès est possible depuis un site quelconque du réseau.

Fichier réseau :

fichier n'ayant pas un site de résidence unique, décomposable en sous-fichiers, chacun étant de type "fichier à accès réparti".

Base de données :

collection d'informations structurées.

Base de données réparties :

base de données n'ayant pas un site de résidence unique.

Système de gestion de bases de données réparties :

application mettant en oeuvre des bases de données réparties.

M.A.D.R.E. :

méthode d'accès direct réseau (pour les fichiers) [A6], [B8], [A9]

M.A.R.S. :

méthode d'accès réseau SOCRATE (pour les bases de données SOCRATE) [B7], [A11], [A16], [CHP77].

C.R.I.C. :

contribution aux réseaux interactifs de calculateurs [C1].

FANNY :

langage d'écriture de programmes portables [C3], [QUI74], [C10].

SOCRATE :

système de gestion de bases de données [ABR72], [B4], [CII08].

SYNCOP :

sous-système normalisé de traitement de processus téléinformatiques et réseaux [C10], [C17].

IGOR :

interpréteur général orienté réseau [SER74], [DAN77], [DUM74].

SYNDIC :

système de gestion de fichiers à copies multiples dans les réseaux [A14], [B19], [WIL77].

POLYPHEME :

modèle de coopération de bases de données hétérogènes dans les réseaux d'ordinateurs [B12], [ADI77].

SOCYCRATE :

SOCRATE CYCLADES : extension de SOCRATE au support de terminaux CYCLADES [SER75].

ST :

station de transport CYCLADES [ELI74], [DAN76].

CT :

concentrateur de terminaux CYCLADES.

1.5. Plan de la thèse

Nous commencerons (chapitre 2) par mieux situer le cadre du travail : celui des multiprocesseurs et des réseaux d'ordinateurs. Ce sont les outils de base de l'informatique répartie pour lesquels on décomposera les mécanismes élémentaires du traitement de l'information.

Les méthodes d'accès réseau sont développées au chapitre 3 : la gestion de fichiers (avec une réalisation : MADRE), les bases de données réparties et les systèmes documentaires.

Le problème de la duplication de l'information est présenté dans le chapitre 4 ; un algorithme conçu sur des bases originales y est détaillé.

Les outils des applications réparties et télé-informatique sont développés au chapitre 5 : l'expérience du réseau CYCLADES et des logiciels développés dans le cadre de ce projet permet d'offrir quelques outils expérimentés et de tirer quelques enseignements.

Les architectures des réseaux informatiques et les modèles de bases de données réparties sont dévelopés au chapitre 6.

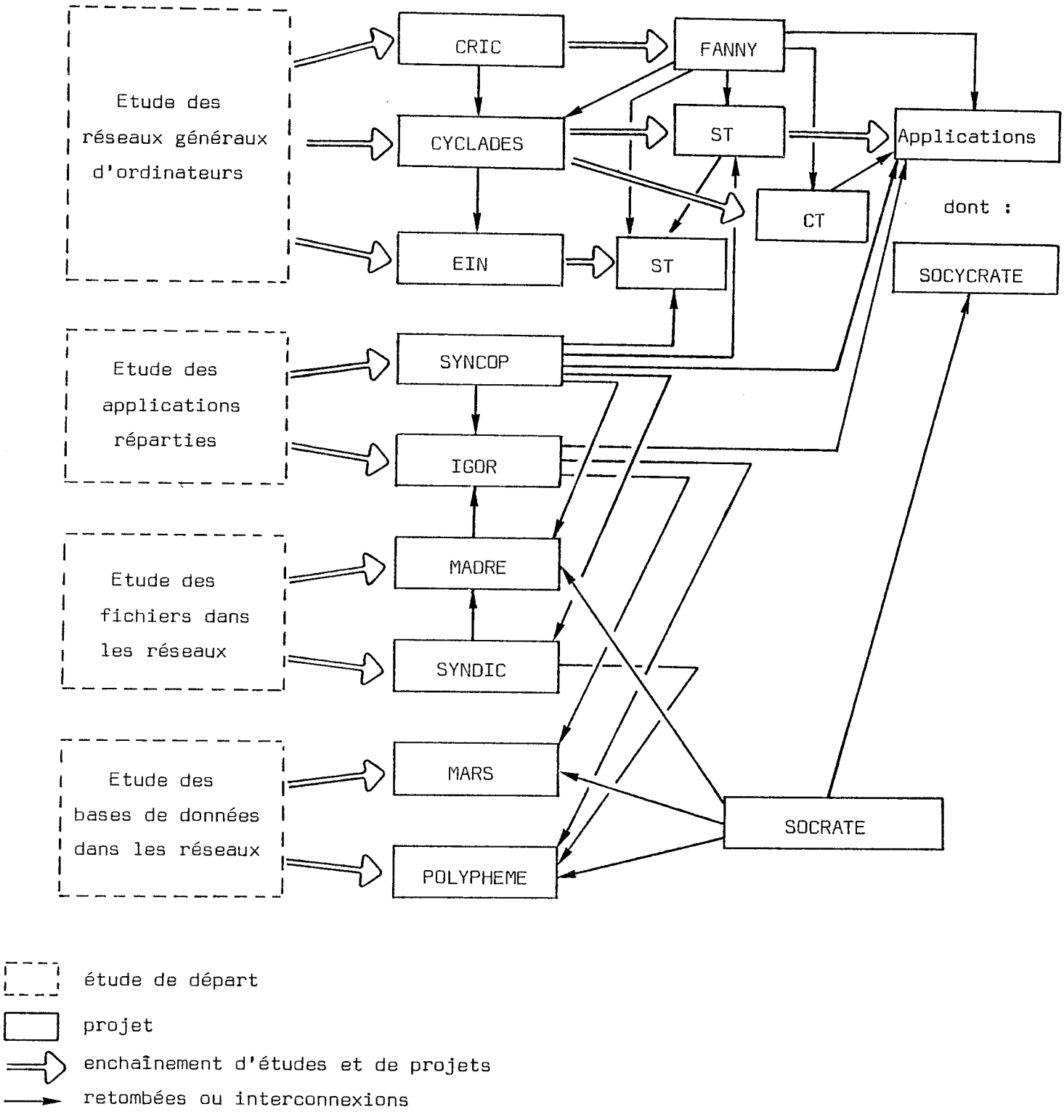


Figure 1.1

Présentation schématique des différents projets auxquels ce document fait directement référence et liens existant entre eux

2. APPROCHE À L'INFORMATIQUE RÉPARTIE

L'informatique répartie correspond à un nouveau stade de l'évolution informatique qui intègre plusieurs facteurs. Nous dénombrons cinq facteurs importants :

a) les centres de calcul élargissent la gamme des services rendus et les possibilités de modes d'accès autour d'unités de traitement centralisées : c'est la *télé-informatique* avec de gros système temps partagés et traitements par lots [GUI76].

b) la miniaturisation des équipements et la fabrication de processeurs spécialisés à de faibles coûts permettent des constructions modulaires de *système multi-processeurs* plus fiables [SER76], [MAZ77], [MAZ78].

c) les réseaux de commutation de données élargissent les modes de connexion d'ordinateurs : ce sont les *réseaux d'ordinateurs* bâtis autour de réseaux de commutation de messages, de paquets et de circuits [POU76], [BAC76], [BAR76], [CLI76], [DAV76].

d) le stockage de l'information sur support magnétique est marqué par l'introduction de disques à grande capacité et à faible temps d'accès, et les mémoires secondaires dites à hiérarchie [BAH72].

e) le traitement de l'information inclut les logiciels de gestion de bases de données ainsi que les systèmes documentaires.

L'informatique répartie se présente comme une combinaison de ces technologies nouvelles. Nous les présentons maintenant en nous efforçant de dégager les problèmes qu'elles poseront dans un environnement réparti.

2.1. Les éléments de l'informatique répartie :

2.1.1. La télé-informatique [GUI76]

Les applications de télé-informatiques sont destinées à satisfaire les besoins d'utilisateurs distants d'un centre de calcul dont les unités de traitement et de stockage restent locales. Trois types d'applications peuvent être dégagés :

- les applications *conversationnelles* comme les systèmes de gestion de bases de données ;
- les applications *transactionnelles* où l'usager a préparé les opérations à effectuer et stocké les programmes destinés à gérer ces "transactions" : les transactions opèrent sur des données telles que fichiers classiques, bases de données bâties suivant des structures connues des usagers locaux comme distants ; les temps de restitution de chaque transaction sont courts (quelques secondes à quelques minutes) et permettent une interaction entre le programme et l'utilisateur ;
- les applications dites *"traitements par lots"* pour lesquelles les organes d'entrée (tels que lecteur de cartes, lecteur de floopies-disques ou de rubans papier), et les organes de sortie (tels que imprimante) comme les organes de commande (tels que pupitre opérateur) peuvent être locaux ou distants.

Pour cette gamme d'applications, la composante topographique s'insère dans les systèmes par l'adjonction de matériels de télé-transmissions, de procédures de transmissions (tel que BSC, HDLC, TMM-RB,...) et d'adresses de périphériques. La topographie n'introduit pas de bouleversement dans la conception des systèmes : elle s'introduit par des modifications de catalogue et des bibliothèques de procédures, par des améliorations dans la définition et la multiplicité des postes de travail opérateur comme dans le traitement des fichiers d'entrée-sortie. La télé-informatique permet un élargissement de la clientèle d'un centre de calcul, mais n'accroit pas les ressources mises à la disposition des utilisateurs locaux.

La mise en oeuvre de ces applications de télé-informatiques en environnement réparti nécessitera une définition précise de la notion d'*utilisateur* - terminal et de *serveur*-programme ainsi que des protocoles de communication et d'échanges entre ces entités [ZI276].

2.1.2. Les systèmes multiprocesseurs :

Les multiprocesseurs ont deux caractéristiques générales :

- l'existence de plusieurs processeurs,
- le partage d'une mémoire commune.

A l'heure actuelle, plusieurs expérimentations sont menées avec des processeurs de la taille des minis ou des micros ordinateurs :

. le calculateur *C.mmp* de l'Université de Carnegie-Mellon et le système HYDRA qui y est développé ; les processeurs sont des PDP 11 (jusqu'à 16) qui se partagent une mémoire commune de 4096 pages de 8 K octets. Le système *HYDRA* est construit de façon modulaire autour d'un noyau qui définit un jeu d'opérations permettant de réaliser le contrôle de l'accès aux ressources et d'en créer de nouvelles [WUL72], [WUL75]. L'architecture ainsi constituée représente une grosse puissance de calcul, comparable aux gros systèmes PDP10. HYDRA reste un système centralisé ;

. le système PLURIBUS [HEA73] orienté vers les applications temps réel est réalisé pour un IMP (noeud) du réseau ARPA ; le concepteur du système définit des configurations matérielles sous forme d'éléments à assembler avec des règles d'assemblage.

. le projet Computer Module (CM*) [SFS77] est un multi-microprocesseur où les micro-processeurs sont des LSI 11, processeurs de 16 bits exécutant du code DEC ; il est composé d'un seul espace mémoire visible de tous les processeurs, formé d'une réunion de modules mémoire accessibles de façon différente par les différents processeurs. L'utilisateur, comme pour PLURIBUS, a une structure de base à sa disposition sur laquelle il répartit les processeurs et les modules mémoire.

D'une façon générale, ces systèmes ont pour objectif une meilleure disponibilité en permettant la substitution d'un processeur par un autre, une meilleure *rentabilité* en utilisant des processeurs de série, permettant l'extensibilité et la reconfiguration de systèmes.

Par rapport aux réseaux d'ordinateurs qui seront vus ultérieurement, la caractéristique essentielle est le partage d'une mémoire commune avec des mécanismes d'adressage très caractéristiques. Les ressources mises en commun sont :

. la *mémoire* parfois divisée en une mémoire commune et des mémoires locales réservées à des classes de processeurs [MAZ76], [REC76],

. les *interruptions* (parfois),
. les *bus* [KAI77]
. les *coupleurs*...

Ces systèmes font apparaître plusieurs problèmes importants de l'informatique répartie : l'allocation de ressources sans arbitre unique, l'exclusion mutuelle, la duplication d'informations, l'adressage.

2.1.3. Les réseaux d'ordinateurs :

Nous appellerons réseau d'ordinateurs un ensemble constitué d'ordinateurs reliés entre eux par des moyens de télé-communication en vue d'une mise en commun de ressources. Les *moyens de télé-communication* peuvent être des lignes directes, des réseaux de commutation de paquets ou de circuits, des liaisons par satellite,... Nous ne les détaillerons pas, pas plus que les topologies existantes (étoilées, maillées, centralisées), pas plus que la place des réseaux de commutation par rapport aux ordinateurs du réseau (réseaux à deux niveaux...).

Les *ressources* mises en commun peuvent être des processeurs de stockage de données, des unités de contrôle, des terminaux,...

Du point de vue de l'utilisateur, certains modes de classification sont intéressants :

a. Les réseaux *généraux* ne sont pas orientés vers des applications particulières : c'est le cas de ARPA [ROB71], de CYCLADES [IRI73], de EIN [BAR76] pour lesquels les ressources mises en commun sont celles de centres de calcul différents et pour lesquels le réseau n'est pas, a priori, source d'uniformisation des besoins et des programmes ;

b. les réseaux *spécialisés* sont l'apanage des grandes administrations, des banques, des agences aérospatiales, des constructeurs automobiles, des armées,... avec pour objectif :

. une disponibilité accrue par une banalisation des ressources et des moyens de reconfiguration,
. une décentralisation des points d'accès,
. une meilleure intégration des services interactifs avec des services centraux tels que statistiques, comptabilité, gestion de stocks, précédemment effectués off-line.

c. Les réseaux *homogènes* sont tels que les ordinateurs interconnectés (les sites) ont des systèmes d'exploitation aux mêmes caractéristiques pour l'utilisateurs (sinon ils sont dits *hétérogènes*) ;

d. Les réseaux sont *distribués* si tous les sites ont le même rôle et *centralisés* lorsqu'un site privilégié dicte certaines règles de conduite aux autres sites.

Les premiers problèmes nouveaux que soulèvent les réseaux d'ordinateurs seront la maîtrise de l'outil de communication, la banalisation des ressources, l'hétérogénéité. La construction de nouvelles application (les *applications réparties*) sera ensuite envisagée.

2.1.4. Les techniques de stockage de l'information :

Dans les applications de *gestion*, la fiabilité, la sécurité, l'accessibilité sont fondamentaux. L'évolution des techniques de stockage de l'information les rendent possibles.

La situation de la technologie des supports magnétiques peut se résumer comme ceci :

- . abaissement du coût des mémoires,
- . abaissement des temps d'accès aux mémoires de masse telles que les disques,
- . accroissement des capacités des disques, de la densité des bandes magnétiques,
- . compatibilité accrue entre les constructeurs.

De nouvelles mémoires de masse permettent la fabrication de systèmes spécialisés dans le stockage de l'information qui offriront d'ici peu un coût par bit très inférieur à celui des disques.

Une notion de *service "stockage de l'information"* tend à apparaître : la clientèle de ces services aura besoin d'opérer des transferts de fichiers (ou bases de données) pour les exploiter sur des unités de traitement externes et refaire des transferts en fin de traitement.

Dans les réseaux d'ordinateurs où les ressources seront banalisées, de tels services devront être disponibles ; ils feront apparaître des spécialisations de ressources avec des problèmes de localisation, d'accès, de partage et de désignation.

2.1.5. Les techniques de gestion de l'information :

La situation traditionnelle est schématiquement celle-ci : à chaque application ses fichiers ; à chaque fichier rarement plus d'une application.

Cela a plusieurs inconvénients :

- redondance d'informations souvent incohérentes,
- applications peu évolutives étant donné les structures rigides des fichiers,
- programmation complexe,
- peu de sécurité des fichiers en cas d'incident.

Les logiciels de bases de données sont construits autour des idées suivantes :

- la non redondance des informations élémentaires,
- la structuration des données dans les fichiers,
- l'indépendance entre les programmes et les bases,
- l'accès sur critères,
- la sécurité,
- les langages d'interrogation,
- le multi-accès.

A l'heure actuelle les concepteurs de réseaux parlent des applications réseaux en termes de bases de données réparties ; les problèmes à résoudre seront : la répartition des bases, la localisation, la duplication, l'exécution de programmes sur des bases réparties, etc.

2.2. Les réseaux informatiques :

2.2.1. Introduction :

Dans ce document nous nous intéresserons essentiellement aux logiciels des réseaux informatiques : nous pourrons cependant dégager plusieurs conclusions valables dans le cadre plus général de l'informatique répartie. Nous commencerons par mieux situer la place de l'outil de communication et les services qu'il rend.

Nous dirons qu'un utilisateur d'un réseau d'ordinateurs est une entité capable d'émettre et (ou) de recevoir des informations par le canal d'un réseau de communication. Cet utilisateur est souvent appelé abonné du réseau : programmes d'application, terminaux, services système, sont autant d'utilisateurs potentiels.

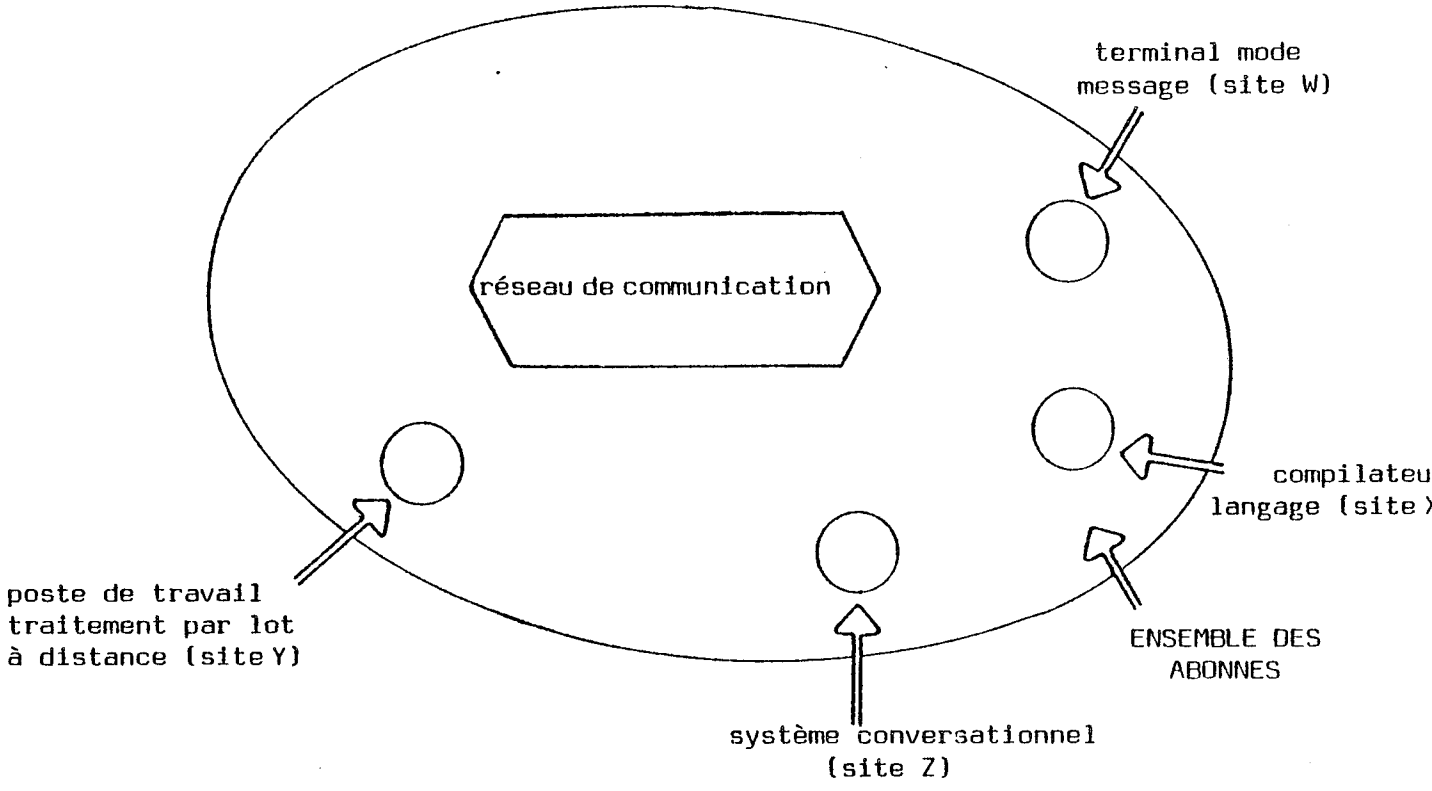


Figure 2.1

Abonnés d'un réseau

Pour dialoguer, les utilisateurs (abonnés) s'adressent à des stations de transport (unités logiques de transport) : celles-ci attribuent aux informations échangées un format compatible avec le réseau de communication, délivrent les informations aux abonnés dans une forme ad-hoc. De plus, pour les abonnés qui souhaitent s'échanger des informations pendant des sessions avec des garanties (contrôle d'erreur, contrôle de flux de messages), il y a la possibilité d'ouvrir et de fermer des voies.

Les voies sont des canaux logiques de communication parfois appelées flots, voies logiques, voies virtuelles. La matérialisation des voies est variable suivant les réseaux : dans les réseaux à commutation de paquets les chemins physiques empruntés par deux informations sur une même voie peuvent être différents ; dans les réseaux à commutation de circuits pour une même voie ces chemins sont identiques, mais entre deux abonnés deux voies peuvent emprunter des chemins différents.

Vis à vis de la communication l'abonné est représenté par un ensemble de portes, chaque porte pouvant être l'aboutissement d'une ou plusieurs voies. Il existe alors un catalogue général des noms des portes du réseau, la mise en correspondance d'un nom de porte avec un abonné est assurée localement par le site auquel appartient la porte (exemple : les portes Cyclades [ELI74]).

Deux opérations élémentaires assurent la transmission de messages :

1) l'*émission* : délivrance d'un message au réseau de communication à destination d'une porte réceptrice,

2) la *réception* : retrait d'un message par une porte destinatrice.

Pour exécuter ces opérations de base, le réseau de communication doit remplir les fonctions suivantes :

- *fonction 1), l'adressage :* identification et désignation des portes communicantes et mécanismes d'accès,
- *fonction 2), la synchronisation :* mécanisme permettant de mettre les portes communicantes dans un état où la communication soit possible.

Le service de communication doit tenir compte de caractéristiques particulières données aux voies par le réseau :

1) la capacité (nombre maximum de messages qu'elle peut contenir),
2) la bande passante (débit d'informations qu'elle peut transmettre),
3) la fiabilité (probabilité de transmissions correctes de messages, sans perte ni erreur).

La prise en compte de ces caractéristiques introduit trois nouvelles fonctions :

- *fonction 3), la régulation de flux :* pour éviter la saturation des ressources associées aux portes (côté abonné),
- *fonction 4), la régulation de charge :* pour éviter la surcharge de la voie, c'est-à-dire la saturation des ressources du réseau de communication,
- *fonction 5), le contrôle d'erreur* : pour assurer une fiabilité supérieure à celle des composants de la voie.

Pour les logiciels des réseaux informatiques, ces cinq fonctions vont apparaître comme autant de fonctions de base. On doit noter ici certaines similitudes avec d'autres modes de communications par messages entre entités d'un même ordinateur de traitement ; la communication par boites aux lettres par exemple.
Dans l'optique d'une uniformisation de ces services pour l'utilisateur, des propositions de primitives ont été faites, disponibles depuis les langages évolués (comme PL/1) [DEC77], [AND78].

2.2.2. Un exemple : le réseau CYCLADES :

Les premiers réseaux informatiques expérimentaux sont développés outre-Atlantique. Le réseau ARPA a introduit en 1969 la commutation de paquets. Il couvre le territoire des Etats-Unis et se prolonge, en utilisant des liaisons par satelitte, à Hawaï et à Londres. Les noeuds du réseau sont des ordinateurs Honeywell DDP516 et DPP316. Son objectif, du côté des utilisateurs, est la consultation de fichiers, la mise à disposition de services de télé-traitement et de temps partagé.
En France, le réseau CADUCEE sert à raccorder des terminaux lourds. Les liaisons entre ordinateurs par CADUCEE ne permettant que les transferts d'informations pour des traitements par lots différés. Le projet pilote CYCLADES démarre en 1972 sous l'égide de la délégation à l'informatique. Il a pour objectif de montrer la faisabilité d'un réseau d'ordinateurs et de mieux analyser les besoins des utilisateurs potentiels. Pour ce faire, il accorde plusieurs priorités :

1) l'*accès aux bases de données* : mise sur le réseau des bases de données de l'administration, études sur les bases de données réparties et sur la confidentialité...
2) l'*échange des données* entre programmes et terminaux,
3) le *partage des programmes* : activation d'un programme à distance, transferts de fichiers...
4) le *partage d'équipement* : banalisation des ressources, partage de charge...

Le développement du réseau Cyclades a nécessité l'écriture d'un grand nombre de logiciels réseau : le réseau de commutation Cigale [GRA73],

les stations de transport [ANS76], [DAN76], [AND76], les applications fichier réseau [B8] , bases de données réparties [IRI76], [GAR76],[B12], etc.

Un grand nombre d'expérimentations ont été menées à bien dans l'environnement de machines hétérogènes et de réseau général constitué par Cyclades : pour les logiciels de base et pour les applications.

2.3. Les logiciels réseau :

Les trois premières caractéristiques des logiciels réseau sont :

- l'utilisation, pour la communication entre entités,de réseaux de commutation : nous avons déjà vu les cinq fonctions de base (adressage, synchronisation, régulation de flux, régulation de charge, contrôle d'erreur) dont la mise en oeuvre est rendue indispensable par ce type de communication ;

- le parallélisme d'exécution rendu possible par la mise en oeuvre des différentes entités sur des ordinateurs ayant chacun leur unité de traitement ;

- l'autonomie de fonctionnement des ordinateurs interconnectés qui nécessite de pouvoir faire fonctionner les logiciels réseau même en mode dégradé de façon durable en assurant un minimum de services.

Avec les deux exemples qui suivent, nous présentons des logiciels d'applications des réseaux informatiques .

2.3.1. Quelques exemples :

2.3.1.1. Télex avec archivage :

Soit un service permettant aux différents abonnés du réseau de s'échanger des messages sur les principes suivants :

- l'expéditeur peut envoyer un message à son correspondant sans devoir s'assurer de la présence de celui-ci sur le réseau (envoi de courrier),
- le destinataire peut prendre connaissance de son courrier quand il le souhaite.

Le télex avec archivage permet aux abonnés d'utiliser le réseau dans un mode de communication classique. Il constitue pour les administrateurs du réseau un moyen de diffusion de leurs notes de service.

Pour sa mise en oeuvre, il convient de définir et de réaliser :

- une fonction de *production* et d'*archivage* des messages, effectuée sur le site : ceci pour s'assurer que cette fonction peut être mise en oeuvre indépendamment de la disponibilité des autres sites du réseau ;

- une fonction de *relevé* du courrier effectuée sur le site de celui qui en est destinataire (consommation de messages) ;

- une fonction de *transfert des messages* qui tienne compte de la disponibilité des machines sur le réseau pour réduire le délai entre la production et la consommation, ce qui donne des critères de localisation pour le choix du site d'archivage.

Ces fonctions ne sont pas indépendantes. La fonction de transfert fait intervenir un fichier source (archivage des messages envoyés) et un fichier destination (archivage des messages reçus). Ces deux fichiers apparaissent comme autant de ressources à gérer dans le cadre du réseau utilisées par d'autres fonctions (production : fichier source ; consommation : fichier destination). (La fonction de transfert est réalisable à la demande, en fonction de la disponibilité du réseau, à l'initiative soit du producteur, soit du consommateur, soit d'une tierce personne faisant fonction d'opératrice réseau).

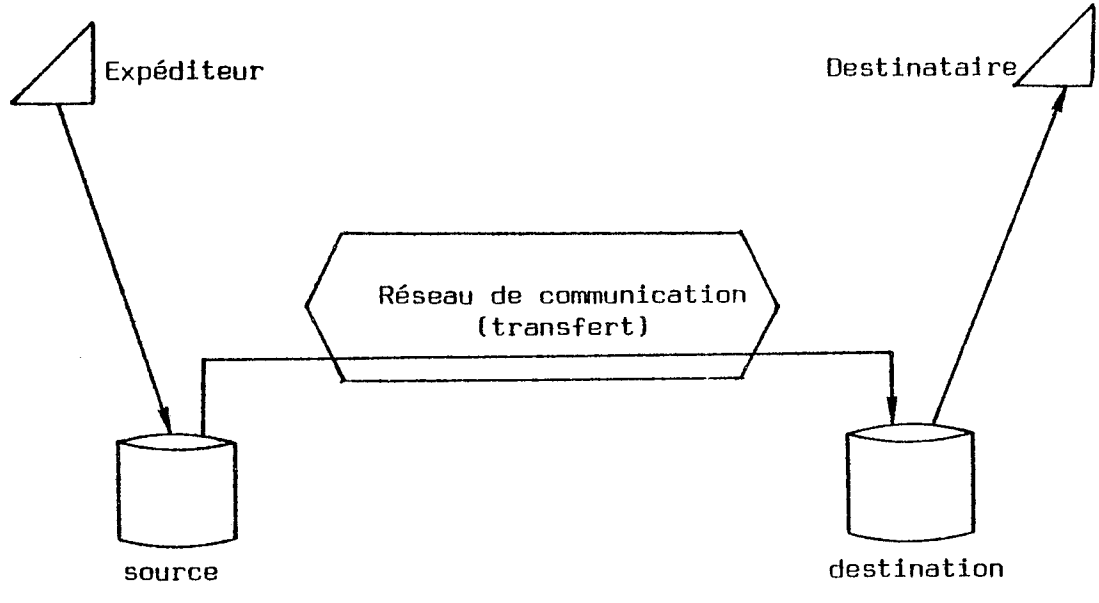


Figure 2.2

Télex réseau

L'allocation de réseau est un problème déjà largement abordé dans la littérature [MAH72], [NEG76]. On peut noter que pour les réseaux informatiques où les ordinateurs interconnectés ont leur propre système d'exploitation, les modèles d'allocation de ressources applicables font apparaître une hiérarchie à deux niveaux : niveau ordinateur site avec un allocateur local ; niveau réseau avec un allocateur réseau. Alors que pour l'informatique centralisée une demande d'allocation de ressources non satisfaite provoque le plus souvent l'attente de la tâche logicielle demandeur, en informatique répartie le logiciel devra pouvoir garder l'initiative des actions à effectuer en cas de ressources indisponibles: les actions possibles peuvent être variables et en particulier le logiciel en cours peut poursuivre son déroulement en mode dégradé, ce qui est une notion inusité pour les logiciels classiques.

2.3.1.2. Documentation réseau :

Soit un système de documentation pour les abonnés du réseau (informations sur les services mis à la disposition, par exemple) :

- le créateur de la documentation doit pouvoir fabriquer des documents sans dépendre de la disponibilité des autres sites et du réseau lui-même ;
- les sites doivent pouvoir obtenir des copies de documents à jour ;
- le créateur - ou ses mandataires - doit pouvoir mettre à jour les informations, tout en assurant la cohérence des copies diffusées.

Un tel service met en oeuvre plusieurs fonctions :

- une fonction "production",
- une fonction "transfert",
- une fonction "consommation",
- une fonction "mise à jour",
- une fonction "fabrication de copies".

La fonction "consommation" est réalisée dans un cadre différent de celui décrit au paragraphe précédent (§ 2.3.1.1.). En effet, elle peut

être mise en oeuvre localement si le fichier document en question a préalablement fait l'objet d'une recopie, mais aussi on peut concevoir une solution plus souple et plus économique où on va récupérer à la demande de l'abonné *tout ou partie* du document. La fonction "consommation" suppose une fonction "mise à jour" de copies multiples de fichiers (cf. figure 2.3. consommation type 2).

Cet exemple permet de montrer un logiciel pour lequel :

- les fichiers-documents sont d'une taille suffisamment importante pour que l'on évite des transferts globaux : d'où la nécessité de définir des ressources plus fines (partition de fichier, sous-ensemble de fichiers,...) manipulables dans le réseau ;
- les fichiers-documents sont des fichiers à accès réparti pour lesquels il faudra effectuer des mises à jour et assurer la cohérence des documents lus ;
- les fichiers-documents doivent être répertoriés dans des *catalogues* accessibles aux abonnés du réseau ;
- les fichiers-documents peuvent utilement faire l'objet de duplication (cf. chapitre 4).

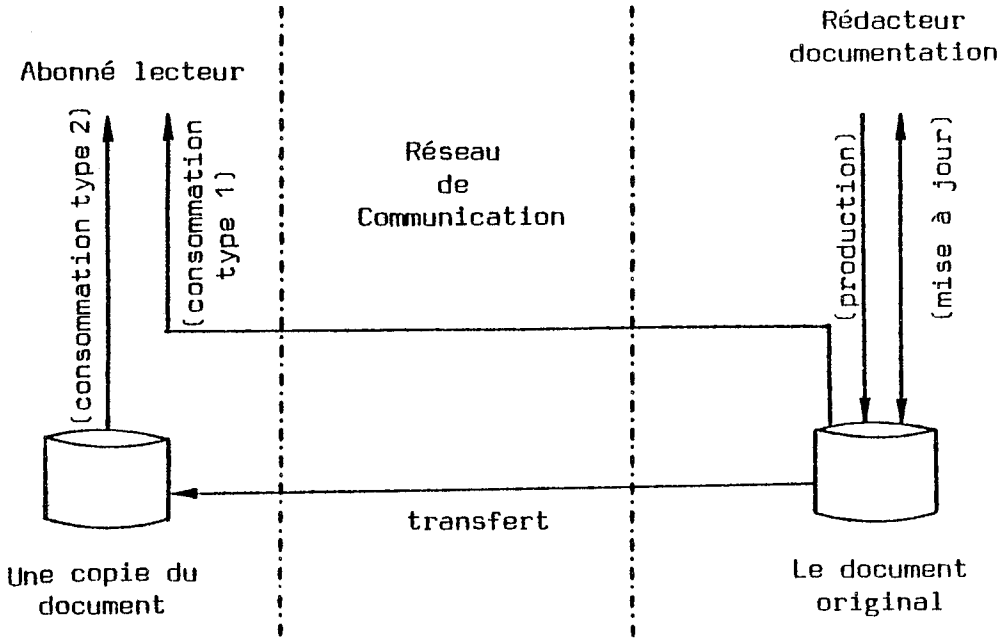


Figure 2.3

Documentation réseau

2.3.2. Nature des contrôles :

Les réseaux informatiques sont composés le plus souvent de systèmes distincts et différents qui mettent en commun un certain nombre de ressources en vue d'une coopération. C'est dire que la démarche conceptuelle est différente de celle qui existe pour les systèmes classiques ou multi-processeurs dans lesquels le concepteur répartit les fonctions et les responsabilités en fonction d'un ensemble de critères cohérents et homogènes.

Nous développons au chapitre 6 cette démarche "coopérative" avec les modèles de bases de données.

Le conception de réseaux généraux comme Cyclades et Arpa s'est faite en considérant que les ordinateurs à interconnecter étaient munis de leurs propres systèmes d'exploitation. Les premières applications réseau apparaissent alors comme des ensembles de tâches coopérants de ces systèmes : ces tâches étant des entités communiquant par le réseau et reproduisant des applications de télé-informatiques classiques : le traitement par lots, le conversationnel, le temps partagé. Pour ces applications, on peut parler de *contrôle centralisé* avec un serveur et des clients : le serveur conversationnel ou traitement par lots est l'élément de l'application qui en a le contrôle. Pour ces applications, les réseaux d'ordinateurs apportent un nouveau support à des applications de conception classique.

Les réseaux informatiques peuvent donner naissance à d'autres applications organisées de façon nouvelle : les entités composantes peuvent s'entendre entre elles, sur un pied d'égalité pour se partager le travail, les ressources et ceci, en s'engageant à respecter un ensemble de règles préalablement définies dénommées *protocoles* : on parle alors de *contrôle réparti*.

Dans l'état actuel des systèmes d'exploitation, les applications à contrôle réparti sont des logiciels tels que IGOR [DAN77], MADRE [B8], POLYPHEME [B12]. Ils donnent des éléments d'architecture de système répartis et des exemples de protocoles mis en oeuvre ; MADRE est développé au chapitre 3, POLYPHEME au chapitre 6.

3. LES MÉTHODES D'ACCÈS RÉSEAU

3.1. Définition :

Méthode d'accès réseau : (définition cf. 1.4)
Service réseau mettant à la disposition de l'utilisateur les moyens d'accéder à des données résidant sur des sites quelconques.

Notre propos dans ce chapitre sera donc d'envisager le point de vue de l'utilisateur et les moyens de lui fournir un ensemble de services faisant appel au réseau sans effort supplémentaire. Les appels à ces services se présenteront comme des primitives intégrées à différents langages de programmation.

Nous aborderons successivement : les fichiers et les bases de données (respectivement 3.2, 3.3). Ces domaines ont chacun leurs caractéristiques propres, mais l'idée directrice reste de favoriser la communication entre la donnée (quelle que soit sa forme, son support...) et l'utilisateur.

Avant d'aborder plus à fond l'étude des méthodes d'accès réseau, nous devons mentionner la démarche généralement suivie dans l'expérimentation d'un réseau d'ordinateurs : elle consiste à définir des services réseau spécifiques pour chaque application. Parmi ces services, nous pouvons citer :

3.1.1. Les langages de commandes réseau :

avec lesquels deux objectifs principaux devaient être atteints :

- la désignation et l'utilisation des fichiers distants,
- l'exécution de programmes sur des sites distants où ils étaient préalablement rangés en bibliothèques.

Cette démarche repose sur une restriction importante : un programme s'exécute sur un seul site ; elle a permis des réalisations rapides sans modifications de systèmes d'exploitation existants.

a) Le système RSEXEC :

RSEXEC [THO73]est un système distribué organisé avec les sites TENEX du réseau ARPA (RSEXEC = Ressources Sharing Executive). TENEX est

système d'exploitation partagé implanté sur des ordinateurs DEC PDP10 : RSEXEC en est un sous-système. Recopié sur chaque hôte TENEX, il permet la mise en commun des ressources, leur partage entre hôte TENEX.

L'utilisateur accède au réseau par un hôte TENEX ou par un TIP (Terminal Interface Processor ou mini-hôte) ; il dispose d'un langage de commandes analogue à celui de TENEX qui lui permet de désigner tous les hôtes TENEX du réseau. Par un mode d'utilisation et de désignation unique, il manipule un ensemble de ressources du réseau gérées dans un environnement TENEX.

RSEXEC dispose d'un système de fichiers organisé en un arbre pour l'ensemble des TENEX. Tout hôte TENEX, tout utilisateur, tout fichier ou périphérique (dans TENEX on accède à un périphérique à travers le système de fichiers) est désigné par un nom d'arbre complet. Cet arbre de fichiers (regroupement des arbres locaux) constitue un référentiel unique.

b) Le système UNIX :

UNIX [RIT74], [CHE75] est un système d'exploitation implanté sur DEC PDP 11/40 et 11/45 ; par adjonction d'un programme d'interface (NIP), le système UNIX devient le système "UNIX réseau" connecté au réseau ARPA. Les autres hôtes du réseau sont vus alors par UNIX comme des extensions de ses propres objets.

Pour l'utilisateur, les entrées/sorties sont uniformisées : mêmes opérations sur les fichiers, les périphériques, les processus divers.

UNIX possède un système de gestion de fichiers hiérarchisé. Chaque fichier est repéré par un nom d'arbre complet ou incomplet s'il fait référence au répertoire de l'utilisateur. Des fichiers spéciaux sont associés aux périphériques qui sont désignés par les noms de ces fichiers.

Le système de fichiers et la structure de répertoire sont réalisés par un simple système de pointeurs ; une entrée de répertoire contient seulement un nom pour le fichier associé et un pointeur (nombre entier). Cet entier est utilisé comme index dans une table contenant les descriptions physiques et logiques des fichiers.

Dans UNIX réseau, les hôtes sont désignables par le même mécanisme que les périphériques : un fichier spécial "réseau" est associé à chaque hôte à distance.

c) Le réseau SOC :

SOC [SOM 75] (système d'ordinateurs interconnectés) est un réseau d'ordinateurs IBM homogène (IBM 360/370). Sur chaque système participant est implanté un interpréteur de langage de commande réseau (le LE/1 [DCA73]) : ce langage comprend un certain nombre de variable (CPU, fichier, volume,...) et d'opérations permettant de construire des programmes réseau comme les transferts de fichiers, la demande d'exécution depuis le site A d'un programme sur le site B avec fourniture des résultats sur le site C, etc. Le système réseau est constitué d'un ensemble de sous-systèmes de contrôle locaux coopérants, chaque sous-système étant une tâche des systèmes d'exploitation des ordinateurs participants.

3.1.2. Les transferts de fichiers :

[GIE77], [DAY73], [NEI73], [BHU72], [BAE72], [EPS74].

Les services transferts de fichiers ont donné lieu à un grand nombre d'études sur les modalités de désignation et d'accès de fichiers quelconque dans le cadre de réseaux hétérogènes d'ordinateurs.

Les études faites par [GIA74] et qui complètent très largement celles de [RAY73] pour les systèmes 360 d'IBM, permettent de dégager un mode de description de fichiers grâce à une liste de paramètres qui trouve son équivalent dans chaque système local. Cette description s'organise dans deux catégories de paramètres :

a) Les paramètres de localisation :

- le site sur lequel le fichier est implanté,
- le nom du fichier,
- le nom ou numéro de volume d'implantation,
- le nom de l'unité associée (type ou adresse du périphérique),
- des informations sur le type et le degré de partage,
- des informations sur le nombre de volumes (éventuellement),
- l'état du fichier (existant, à créer,...),
- la taille principale et la taille additionnable lorsque le fichier est à créer.

b) Les paramètres de description interne du fichier et de l'accès utilisé :

- le type de l'organisation (séquentielle, directe, partitionnée),
- le format de l'article (fixe, variable, indéfini),
- la longueur de l'article,
- la longueur de l'en-tête d'article dans le cas du format variable ou indéfini,
- la longueur de bloc ou enregistrement physique (cf. facteur de blocage),
- le mot de passe (s'il y a protection d'accès),
- le type de l'accès (assisté ou non assisté),
- le nombre maximum d'entrées/sorties simultanées,
- le type des erreurs ou anomalies à traiter,
- l'adresse de la séquence de traitement des erreurs.

Cette description générale peut se projeter sur les différents systèmes considérés. Les choses sont plus complexes lorsqu'on s'intéresse à la comparaison des descripteurs et des modes de traitement possibles sur les fichiers : cette étude est importante si on veut autoriser une transparence totale pour l'utilisateur.

Les résultats sont très diversifiés suivant l'organisation retenue.

1) Les fichiers séquentiels :

Leur *désignation* est faire par une valeur de paramètre (ORG : SIRIS 7, DSORG : OS 360) ou un indicateur dans une table partagée par le système et l'usager (FET : SCOPE) ou du DCB (SFD : SIRIS 2).

Leur *gestion* est possible :

- au niveau de l'article : tous les formats d'article sont envisageables, les positionnements se font au niveau du bloc,
- au niveau du bloc : les longueurs de blocs sont fixes sauf sous OS360, le positionnement se fait comme précédemment, la longueur de transfert peut être inférieure à la longueur du bloc en SIRIS 2 et SIRIS 7 (options dans les instructions d'entrées/sortie).

La *création* se fait en séquentiel, toutes les méthodes d'accès sont autorisées en lecture.

2) Les fichiers partitionnés :

Les possibilités sont totalement fonction du système et du traitement effectué. L'adressage est machine-dépendant, de même que les méthodes d'accès autorisées, les conditions d'allongement d'une partition, la récupération d'espace, la création... L'écriture, par exemple, se fait tantôt au niveau de l'article, tantôt au niveau du bloc.

3) Les fichiers séquentiels indexés :

La création se fait de façon séquentielle, avec tous les systèmes, dans l'ordre croissant des clés. A partir d'une clé donnée, la mise à jour peut être directe ou séquentielle. En écriture, les différentes méthodes d'accès sont autorisées.

4) Les fichiers à accès direct :

Les longueurs de blocs sont fixes (SIRIS 7/8), variables ou indéfinies. Le système SCOPE gère une table d'index dont le nombre d'entrées est égal à la taille maximum du fichier ; le rang d'une entrée correspond à un numéro d'article (la clé plus l'adresse sur le support). Le système SIRIS 2 a deux types d'adressage possibles : le mode sélectif avec adressage par clé d'article et numéro de case, le mode indéfini avec traitement au niveau du bloc et adresse relative.

Ce tour d'horizon fait apparaître des moyens très diversifiés dans l'adressage et les méthodes d'accès pour chaque catégorie de fichier. D'une façon générale, il sera difficile de parler d'un fichier de type quelconque en laissant à l'utilisateur ses facilités d'accès et en laissant à un administrateur la responsabilité du site d'implantation dans un réseau hétéràgène. Dans certains cas, on peut faire apparaître des convergences suffisantes : c'est le cas des fichiers séquentiels avec des limitations sur les facteurs de blocage, ou des fichiers à accès direct avec des longueurs de blocs maximales, par exemple.

3.1.3. Méthodes d'accès réseau : application répartie + primitives utilisate

Les réseaux informatiques font apparaître une grande diversité des systèmes d'exploitation, une multiplicité de ressources et de moyens de

communication. Le développement anarchique d'applications réseau risque de poser à terme des problèmes d'incompatibilité lorsqu'on cherchera à décloisonner les applications, à les faire coopérer, à les mettre en oeuvre sur d'autres réseaux ou d'autres systèmes...

C'est pourquoi nous pensons qu'il est nécessaire de définir une méthodologie pour écrire des logiciels réseau qui respecte un certain nombre de règles :

a) donner un mode de description et d'accès unifié aux données manipulées ; ceci est imposé par les contraintes d'hétérogénéité. Cette démarche est adoptée par les concepteurs de modèles de bases de données dans les réseaux [IRI76] ; elle nous guidera dans les spécifications de la méthode d'accès aux fichiers réseau : MADRE, en tenant compte des observations faites au paragraphe précédent [B8],

b) construire les applications réseau suivant des architectures permettant :

- le contrôle réparti des applications pour exploiter les possibilités nouvelles offertes par les réseaux d'ordinateurs,
- la coopération entre les entités composant une même application ou entre des entités d'applications différentes.

Les applications réparties seront décomposées en deux types d'entités :

- les *serveurs* : un serveur est un abonné du réseau, ayant un site de résidence unique et fournissant un certain service (accès à un fichier, à une base de données...),
- les *clients* : un client est un abonné du réseau qui utilise les services mis à disposition sur le réseau grâce aux serveurs.

Une application répartie sera constituée d'un ensemble de clients et de serveurs. L'utilisateur accèdera à l'application au moyen d'un client auquel il se connectera en respectant les protocoles en vigueur dans le réseau.

Un même client pourra traiter les demandes de plusieurs utilisateurs. Ceux-ci pourront être des terminaux, des programmes externes ou des serveurs d'autres applications répartis. Des applications réparties complexes offrant plusieurs niveaux de services seront ainsi décomposés dans un graphe de clients et de serveurs. J.C. Chupin dans [CHP77]

développe quelques propriétés de cette décomposition architecturale des applications réparties.

c) Pour l'utilisateur, les services fournis par une application répartie seront accessible au moyen de primitives munies de paramètres dans le cadre de langages de programmation : on peut alors parler des *méthodes d'accès réseau* comme étant des applications réparties dont les services sont accessibles à l'utilisateur par un jeu de primitives intégrées dans un langage de programmation de logiciel réseau (figure 3.1).

Pour illustrer notre propos, nous proposons au paragraphe 3.2 un exemple de méthode d'accès fichier MADRE que nous avons réalisé dans Cyclades et au paragraphe 3.3 un modèle de structuration des systèmes de gestion de bases de données permettant de définir différents niveaux de méthodes d'accès réseau.

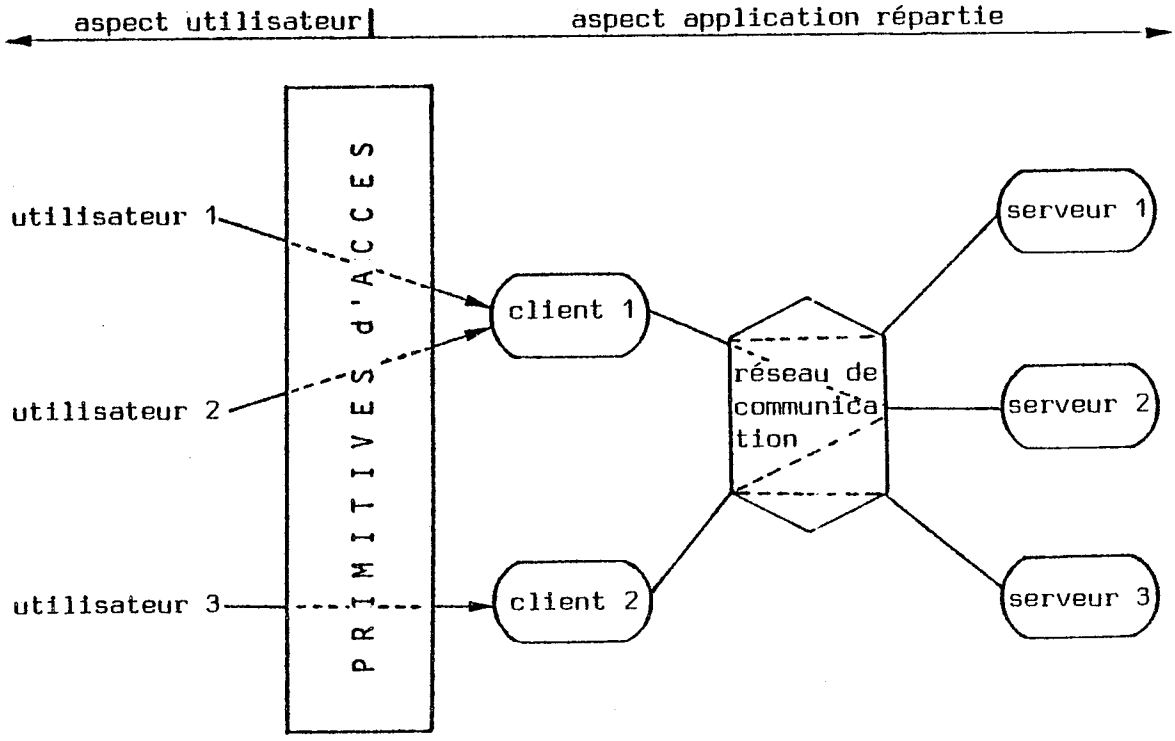


Figure 3.1
Méthode d'accès réseau

3.2. Les méthodes d'accès fichier :

3.2.1. Introduction :

Avec les études sur les transferts de fichiers, nous avons vu les difficultés qu'il y a à définir un modèle général de description et d'accès aux fichiers dans un réseau hétérogène.

La méthode d'accès fichier MADRE qui est développée dans ce paragraphe s'intéresse aux fichiers séquentiels et à accès direct. La justification est la suivante :

- les fichiers séquentiels sont les plus utilisés dans les applications classiques,
- les fichiers à accès direct sont les plus utilisés par les systèmes de gestion de bases de données : ceux-ci ont des modes d'organisation des données sur les supports physiques qui leur sont propres et font appel au mode d'accès aux fichiers le plus basique et le moins coûteux.

Nous rappelons ici trois définitions importantes pour l'exposé qui suit :

fichier : collection d'informations cataloguées sur un support accessible à un ordinateur de façon automatique, et susceptible d'être utilisée ultérieurement.

fichier à accès réparti : fichier résidant sur un site unique du réseau et dont l'accès est possible depuis un site quelconque.

fichier réseau : fichier n'ayant pas un site de résidence unique, partitionné en sous-fichiers, chacun étant un fichier à accès réparti.

3.2.2. Objectifs de MADRE : [A6], [DUB76]

Les objectifs de MADRE sont :

a) permettre aux abonnés du réseau d'accéder à des fichiers à accès réparti ou à des fichiers réseau,

b) supporter des fichiers à accès réparti de types "direct" ou "séquentiel",

c) supporter des fichiers réseau,

d) permettre, dans certaines conditions, la création et la suppression de fichiers,

e) favoriser le passage application locale - application réseau, en offrant à l'utilisateur des primitives d'accès utilisables dans le langage de programmation,

f) être réalisable sur des machines hétérogènes.

Définitions :

FAR : fichier à accès réparti, standard au sens du système de gestion de fichiers de son site de résidence et accessible au cours d'une session MADRE,

FR : fichier réseau ; fichier partitionné, chaque partition étant un FAR (le nom de ce fichier doit être unique dans le réseau) (voir remarque page suivante),

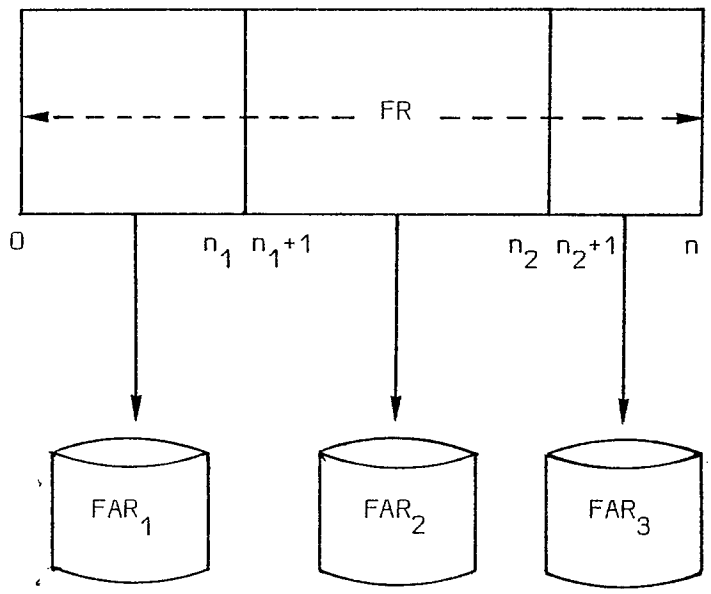
ADM : nom administratif ou nom d'un FAR dans le réseau : il est unique.

RER : catalogue (ou répertoire) des fichiers FAR et FR utilisables grâce à MADRE.

La méthode d'accès MADRE est de fait une application réseau pour laquelle on distingue une partie *client* et une partie *serveur*.

Le serveur est un processus sur le même site que le fichier ; le client est soit un processus sur le même site que l'utilisateur, soit un processus dialoguant au travers du réseau avec l'utilisateur selon un protocole programme - terminal. Les fonctions de chacune des deux parties ainsi que le protocole client - serveur seront développés au paragraphe 3.2.6.

Remarque sur les fichiers réseaux : un fichier réseau peut être considéré comme un fichier virtuel dont l'implantation sur les différents sites peut être représentée comme suit :



numéro virtuel 0 à n_1 ≡ numéro 0 à n_1 sur FAR_1

numéro virtuel n_1+1 à n_2 ≡ numéro 0 à $n_2 - n_1 - 1$ sur FAR_2

numéro virtuel n_2+1 à n ≡ numéro $n - n_2 - 1$ à n sur FAR_3

- la projection se fait en exploitant le RER,
- pour l'utilisateur de la méthode d'accès MADRE, il y a transparence totale (fichier réseau ou fichier accessible du réseau)

(FR) (FAR)

Figure 3.2

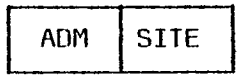
MADRE : le fichier réseau est un fichier virtuel

Projection du fichier virtuel sur les fichiers réels : les FAR

3.2.3. Le catalogue de fichiers :

Dans MADRE, le catalogue (RER) est une liste des fichiers accessibles du réseau FAR's ou FR's. Pour ces deux types de fichiers, l'entrée dans le catalogue est le nom administratif (ADM) dont l'unicité spatiale est nécessaire (celle-ci sera garantie par la mise en oeuvre d'un algorithme de mise à jour de fichiers dans le réseau : cf. chapitre 4).

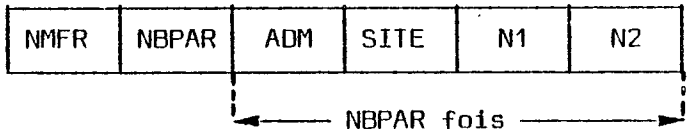
a) Entrée type FAR :



ADM nom administratif,

SITE site de résidence : cette information est équivalente au nom d'abonné du serveur du fichier.

b) Entrée type FR :



NMFR nom fichier réseau

NBPAR nombre de partitions (chaque partition étant un fichier local aux caractéristiques communes avec les FAR).

Pour chaque partition :

ADM nom administratif de la partition

SITE nom abonné serveur de la partition

N1 numéro du bloc correspondant au bloc 0 de la partition

N2 numéro du bloc correspondant au dernier bloc de la partition.

Ceci constitue la partie réseau du catalogue. En fait, sur chaque site serveur, il y a lieu de compléter les informations concernant les entrées gérées par le site.

NMSGF	nom fichier localement au site
OPLVL	étiquette logique du volume de résidence
BLKSZ	taille du bloc
MODE	mode de propriété
PROP	nom du propriétaire
ORG	organisation du fichier
SUP	type du support
LGART	longueur article
ARTF	format article
CODE	code interne des données
ETAT	état du fichier
TYPE	type des données

Le catalogue a plusieurs fonctions pour la méthode d'accès :

- permettre aux clients de connaître le site de résidence des fichiers dont la connexion (= assignation) est demandée par les utilisateurs,
- permettre aux serveurs d'apporter les modifications rendues nécessaires par des créations ou des suppressions de fichiers FAR,
- permettre aux clients d'apporter les modifications rendues nécessaires par des créations ou des suppressions de fichiers FAR,

Le catalogue regroupe l'ensemble des informations fondamentales d'une méthode d'accès réseau : on peut parler d'une *zone commune* [COM75] partagée par les différents composants de l'application : processus clients et serveurs. Le choix d'implantation de cette zone se pose dans les termes suivants :

- une zone sur un site unique, en fonction des coûts de stockage et des coûts de transmission,
- une zone qui circule entre les sites en fonction des accès,
- une zone dupliquée en fonction du taux de mise à jour par rapport aux accès, des coûts de stockage et de transmission,
- une zone dont les copies sont localisées en fonction de différents coûts, de disponibilité des sites, de sécurité.

Ce choix déborde le cadre des spécifications de méthode d'accès : il doit être envisagé pour chaque implantation particulière. Par commodité de présentation, nous supposerons maintenant que ce catalogue est dupliqué avec des copies sur chaque site.

3.2.4. Les primitives mises à la disposition de l'utilisateur :

Pour l'utilisateur, le service MADRE se présente comme une méthode d'accès qu'il manipule en formulant des primitives qui lancent des commandes pour l'application répartie MADRE : le résultat de ces commandes sont les données que l'utilisateur obtient en sortie de ces primitives.

Le *client* peut être un concentrateur d'usagers et il assure une fonction de connexion aux serveurs MADRE et des fonctions de multiplexage/démultiplexage des requêtes ; il interprète les commandes MADRE et dialogue avec les *serveurs* de fichiers au moyen d'un protocole interne dit "protocole fichier MADRE".

L'assignation d'un fichier FAR par un utilisateur mettra en oeuvre un client (côté utilisateur) et un serveur (site du FAR) ; le protocole interne entre le client et le serveur comprendra deux phases : un traitement de commandes pour effectuer l'assignation et l'ouverture du fichier, un traitement de données pour fournir à l'utilisateur les éléments des fichiers qui l'intéresse (cf. figure 3.3).

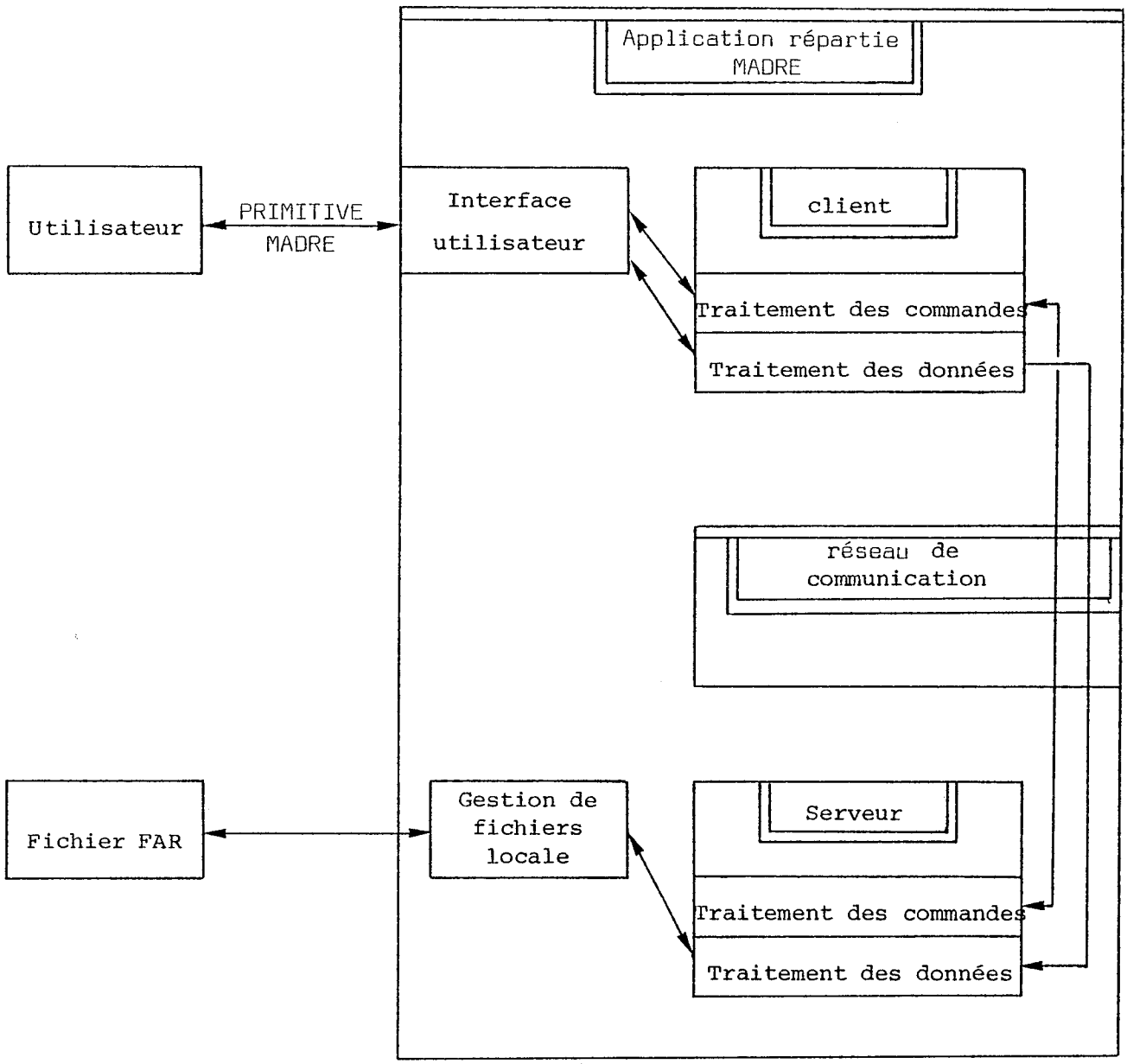


Figure 3.3 -Utilisateur - Machine MADRE - Fichier FAR

Assignation par l'utilisateur d'un fichier FAR

Lorsque l'utilisateur travaille avec un fichier réseau faisant intervenir plusieurs sites de résidence, l'application MADRE mettra en oeuvre un client et autant de serveurs qu'il y a de sites de résidence pour le fichier réseau (cf. figure 3.4).

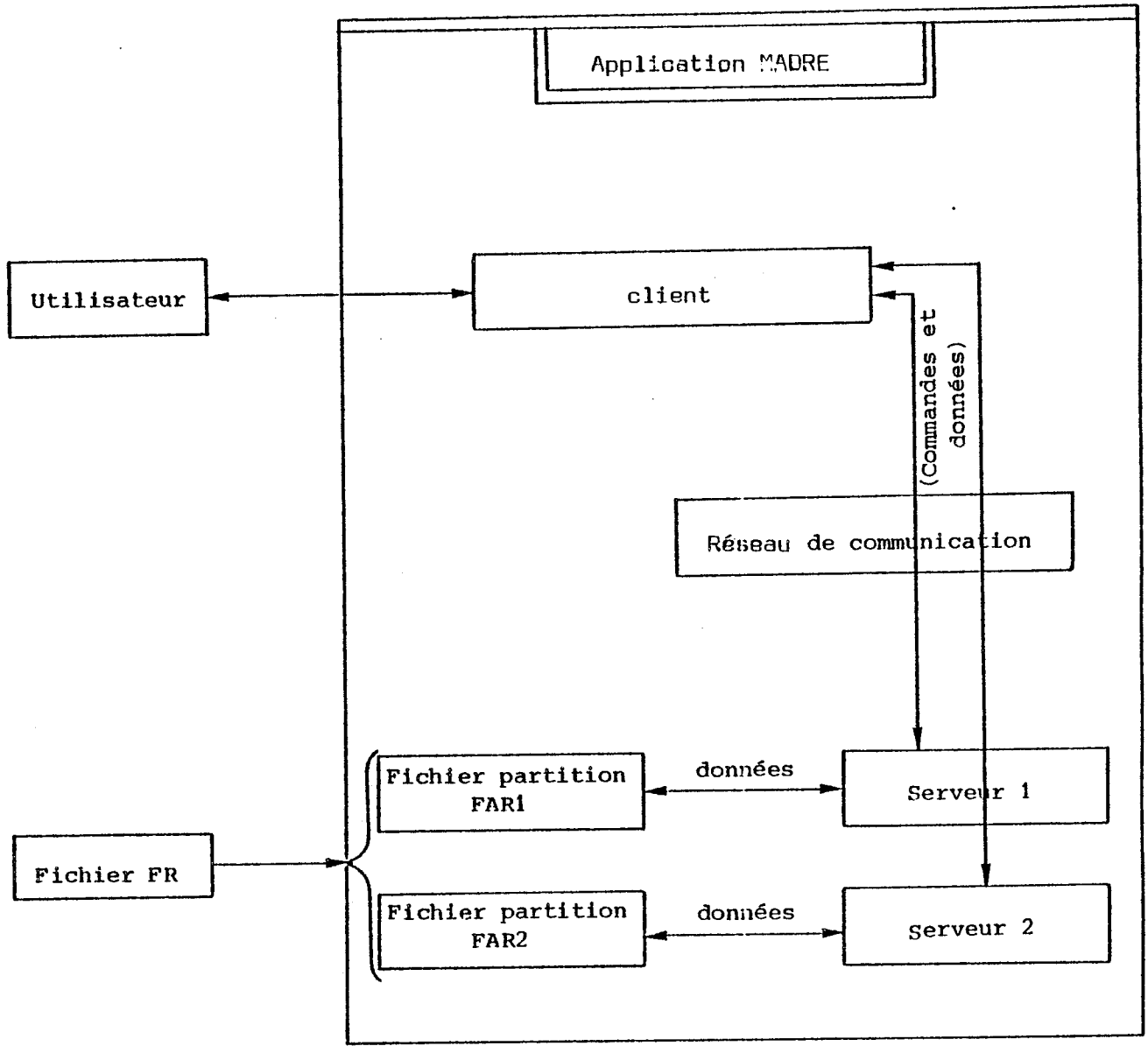


Figure 3.4

Assignation par l'utilisateur d'un fichier réseau FR implanté sur deux sites

3.2.4.1. Les primitives mises à la disposition de l'utilisateur [B8]

Primitives	Fonction	L = locale au client R sinon	Remarques
F:LOGIN	Connexion de l'utilisateur à la méthode d'accès	L	bloquant (1)
F:LOGOUT	Déconnexion	L ou R	bloquant
F:INIT	Contact de l'utilisateur avec un ou des fichiers	L	bloquant
F:END	Libération contexte utilisateur fichier dans client MADRE	L	bloquant
F:CONNECT	Assignation et fermeture d'un fichier	R	non bloquant
F:DISCNT	Libération et fermeture d'un fichier	R	non bloquant
F:READ	Lecture sur fichier	R	non bloquant
F:WRITE	Ecriture sur fichier	R	non bloquant
F:CHECK	Acquittement des primitives non bloquantes	R	

(1) Une primitive est dite *bloquante* lorsque le client de la méthode d'accès garde le contrôle de l'unité centrale tant que l'opération en cours n'est pas terminée.

D'une façon générale, seules les primitives qui ne mettent pas en oeuvre les serveurs (donc locales au client) sont bloquantes : la déconnexion (F:LOGOUT) peut faire exception lorsque l'utilisateur ne s'est pas déconnecté explicitement de ses fichiers (F:CLOSE reste(nt) à faire).

3.2.4.2. Exemple d'utilisation :

lecture des 1024 premiers octets du troisième bloc du fichier ADM1

F:LOGIN nom — nom = nom abonné de l'utilisateur, permet à l'utilisateur de se faire (re)connaître du client MADRE

F:INIT(FIL=opl1,adm1),(ACK=adr1)
contact avec un fichier de nom administratif adm1 : dans le programme il sera référencé avec l'étiquette logique opl1 ; l'adresse adr1 est celle d'une séquence de déroutement où seront traitées les anomalies en cours et fin de traitement (fichier inaccessible, ennuis dans la transmission, fin de fichier)

F:CONNECT opl1,(PTR=adr2),(MOD='L')
la connexion à un fichier est demandée pour une exploitation en lecture (mode = 'L') : à cette opération est associé un événement dont MADRE délivre l'adresse dans adr2 et sur lequel l'utilisateur pourra se mettre en attente

F:CHECK opl1,(PTR=adr2)
c'est l'attente sur fin de connexion ; si l'événement était déjà arrivé, l'opération serait non bloquante. A partir de ce moment, entre l'utilisateur et le fichier tout est en place chez le client, dans le réseau de communication, chez le serveur, pour échanger des données.

F:READ opl1,(PTR=adr3),(BLK=3),(BUF=adr4),(TRL=1024)

une lecture du troisième bloc (BLK) est demandée ; l'utilisateur indique (BUF) où il veut récupérer le résultat et quelle longueur l'intéresse (TRL=1024 octets) ; l'opération est non bloquante (PTR).

F:CHECK opl1,(PTR=adr3)

attente sur fin d'opération.

F:DISCNT opl1,(PTR=adr4)

déconnexion et libération de toutes les ressources immobilisées au niveau du réseau et du serveur.

F:CHECK opl1,(PTR=adr4)

F:END opl1 libération des ressources immobilisées chez le client.

F:LOGOUT fin de session.

Mentionnons ici la connexion multi-fichiers FAR's nécessaire lorsqu'on veut travailler avec plusieurs fichiers simultanément.

La figure 3.5 illustre une session MADRE où les primitives utilisateur font appel aux services MADRE dont l'ensemble constitue le clien localement à l'utilisateur.

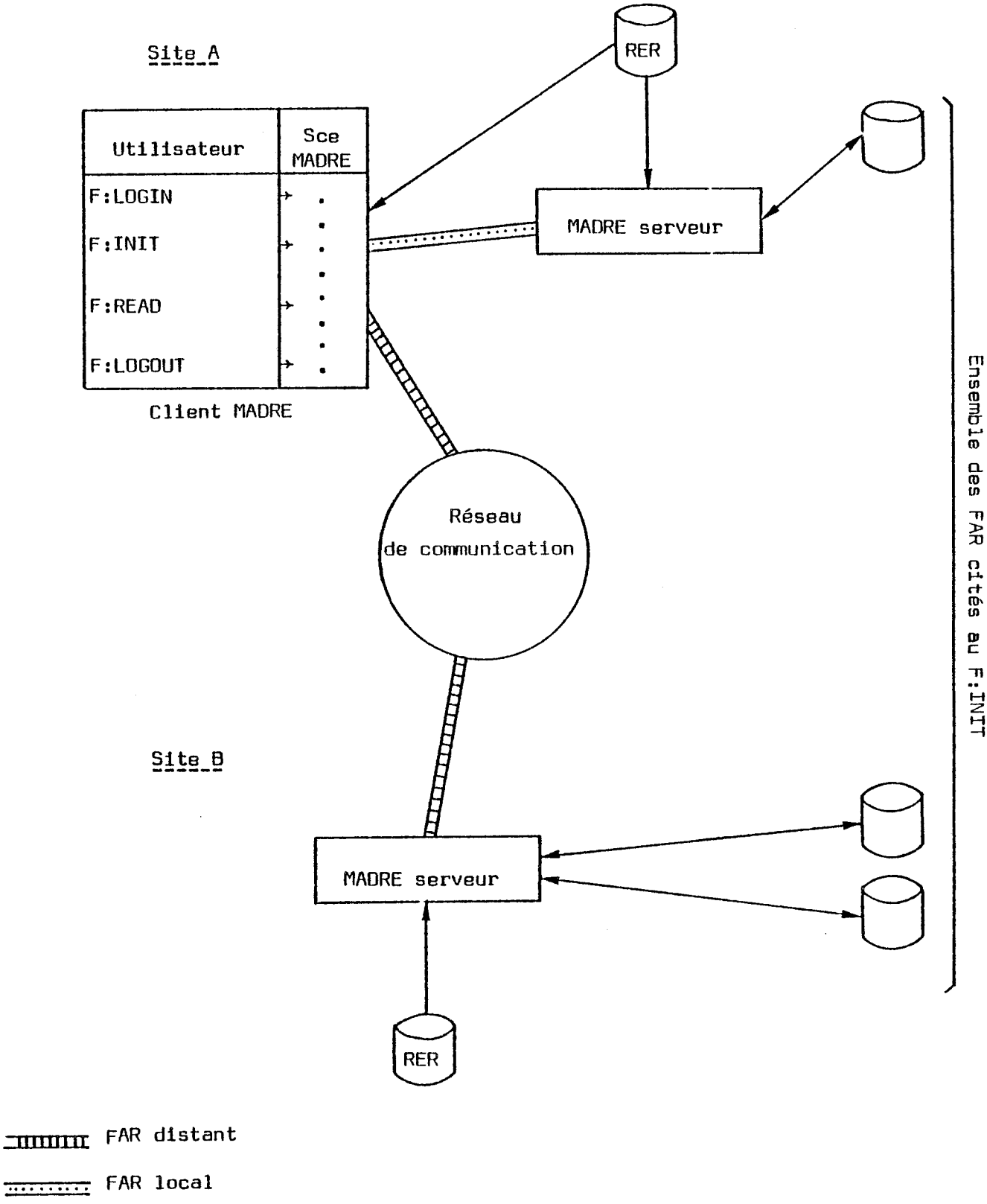


Figure 3.5

MADRE : aspects utilisateurs

Vue macroscopique d'une session MADRE

3.2.5. Particularités des fichiers réseaux (FR) :

Ces fichiers ont comme particularité, par rapport aux FAR's, qu'ils ont plusieurs sites de résidence et donc que leur nom n'a de signification que dans le réseau.

La mise en oeuvre de ces fichiers pose des problèmes nouveaux : lors de la création, il faut leur attribuer un nom unique et de l'espace de stockage pour les différentes partitions. Pour faire cette attribution, le demandeur doit mobiliser un certain nombre de ressources réparties et acquérir le droit de modifier le catalogue pour y prendre une entité pour son fichier ; dans une application répartie comme MADRE, il n'y a pas d'arbitre sur un site privilégié capable de réaliser une allocation de ressources réseau. Les demandes d'allocation peuvent être formulées par un client MADRE quelconque.

On peut définir une séquence critique en associant deux opérations : d'une part la mise à jour du fichier catalogue, d'autre part l'allocation des ressources réseau espace de stockage. On est alors ramené à un problème d'exclusion mutuelle pour lequel divers algorithmes sont proposés pour les systèmes répartis (cf. chapitre 4).

3.2.6. L'architecture de MADRE :

L'architecture de MADRE est décomposée en niveaux fonctionnels : les relations entre deux niveaux, ou *interfaces*, sont définies ainsi que les règles de communication et d'échanges entre fonctions de même niveau ou *protocoles* [ZIM75], [B18], [DAT75].

L'application MADRE est décomposable comme suit : un aspect client (service de l'utilisateur) et un aspect serveur, sur le site du fichier servi. En regroupant sous le terme "sous-système MADRE" les différents composants de l'application locaux à un site, la figure 3.7 représente l'environnement de MADRE.

Comme pour toutes les applications réseau, les niveaux inférieurs de l'architecture sont ceux constitués par le réseau de communication :

[ZIM74] fournit une architecture d'un tel réseau que l'on peut compléter pour MADRE comme suit :

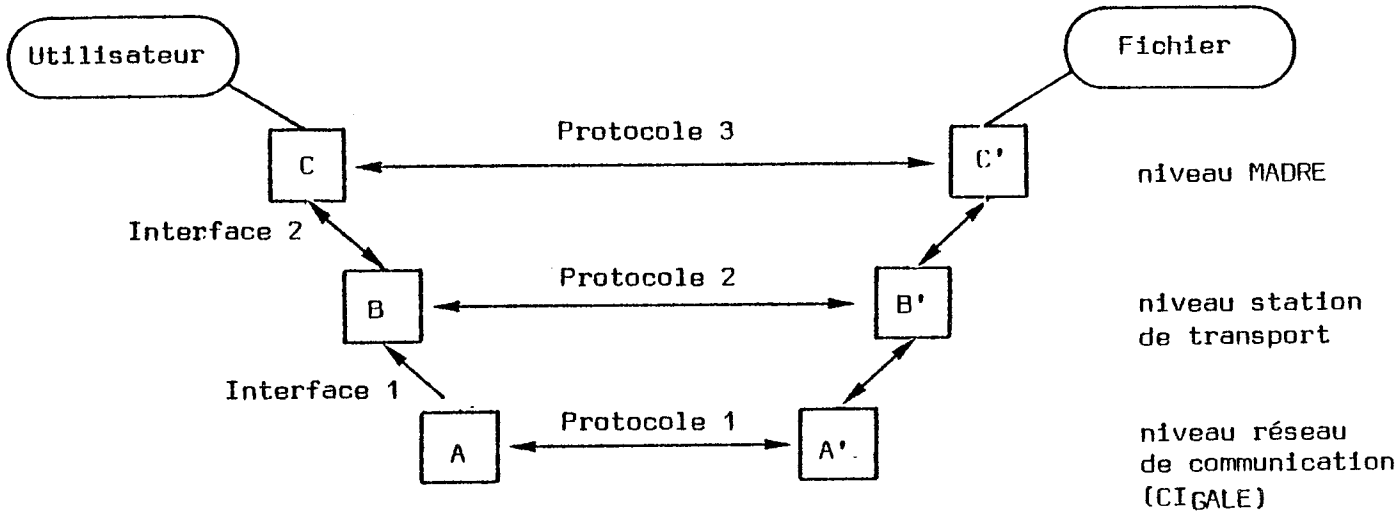


Figure 3.6

Niveaux, protocoles et interfaces dans MADRE

Le niveau réseau de communication (type CIGALE pour CYCLADES) est constitué d'une machine répartie dont les noeuds (les processus) A et A'... communiquent, se synchronisent, etc..., grâce à un protocole de communication (le protocole 1).

Le niveau station de transport, constitué de processus stations de transport B et B', a un interface avec le niveau réseau de communication : l'interface 1 est la connexion site - noeud, le protocole 2 est le protocole de bout en bout [ZIM76].

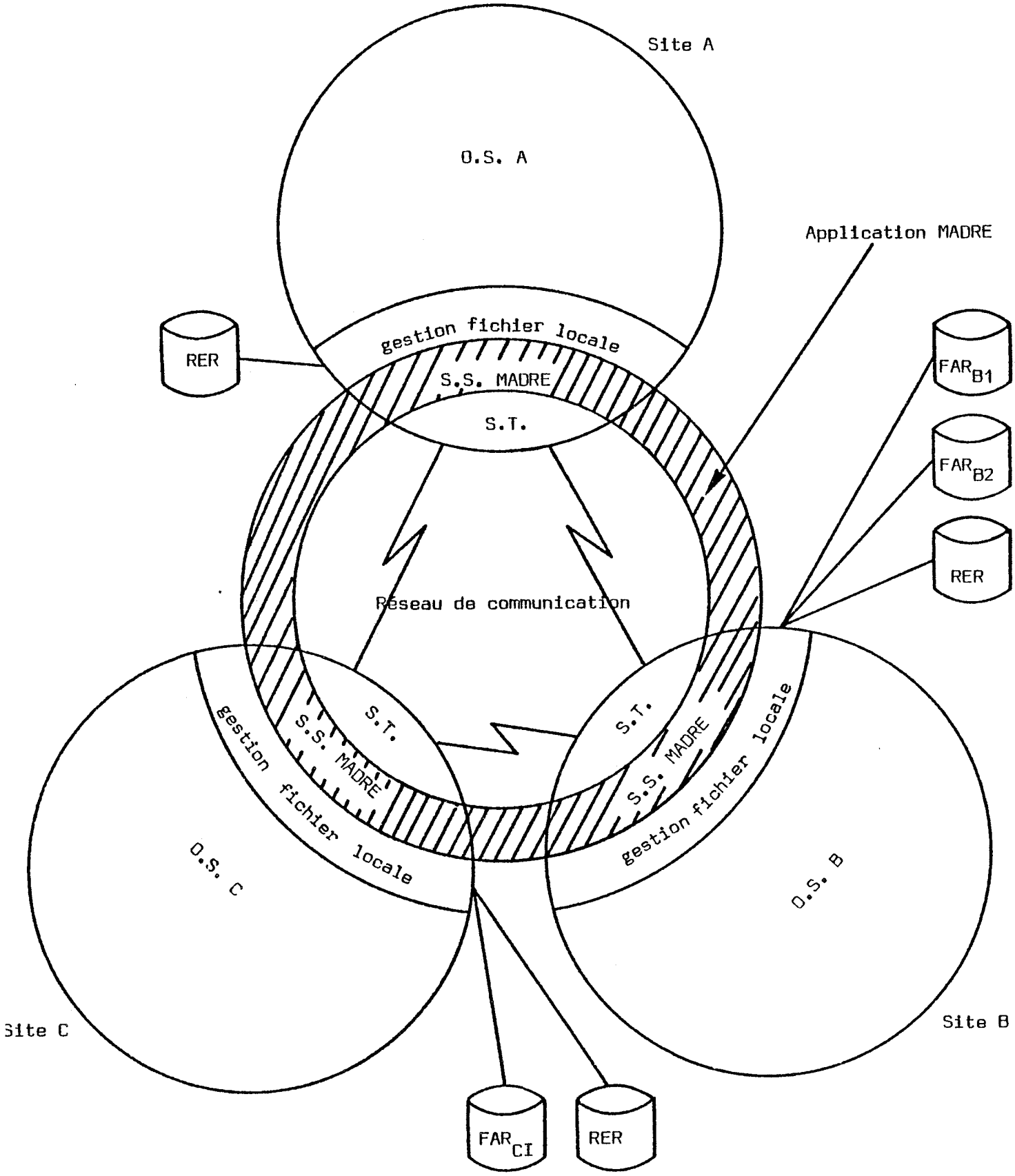
Le niveau MADRE est constitué des processus C et C' qui ont un interface avec le niveau inférieur ; l'interface 2 est à construire, le protocole 3 est le protocole fichier MADRE.

La symétrie qui existe aux niveaux 1 et 2 n'est pas respectée au niveau 3 ; en effet, les deux correspondants mis en oeuvre par MADRE sont un utilisateur et un fichier. En fait, la façon dont sont réparties les fonctions entre les deux processus représentant l'utilisateur et le fichier, c'est-à-dire un processus client et un processus serveur, permettra d'obtenir une symétrie au moins pour un service de base (cf. protocole d'exploitation).

La figure 3.7 donne l'environnement d'exécution de l'application MADRE : cette application apparaît dans les parties hachurées comme un ensemble de sous-systèmes à raison d'un sous-système par site concerné par l'application.

Dans le paragraphe suivant, on décompose le niveau MADRE en trois étapes, chacune correspondant à une phase dans le traitement de fichiers par l'utilisateur :

1) la prise de contact entre l'utilisateur et la méthode d'accès,
2) la connexion de l'utilisateur avec son (ses) fichier(s) de travail,
3) l'exploitation, c'est-à-dire l'échange de données entre l'utilisateur et le fichier.



RER = répertoire des fichiers
FAR = fichier à accès réparti
ST = station de transport
SS MADRE = sous-système MADRE-
OS = système d'exploitation

Figure 3.7 - Environnement de MADRE

3.2.6.1. Les phases d'une session MADRE :

Une session MADRE correspond à une tranche de temps où l'utilisateur souhaite se servir de la méthode d'accès fichier.

a) La prise de contact utilisateur - méthode d'accès MADRE (figure 3.8) :

Elle est envisageable de plusieurs manières :

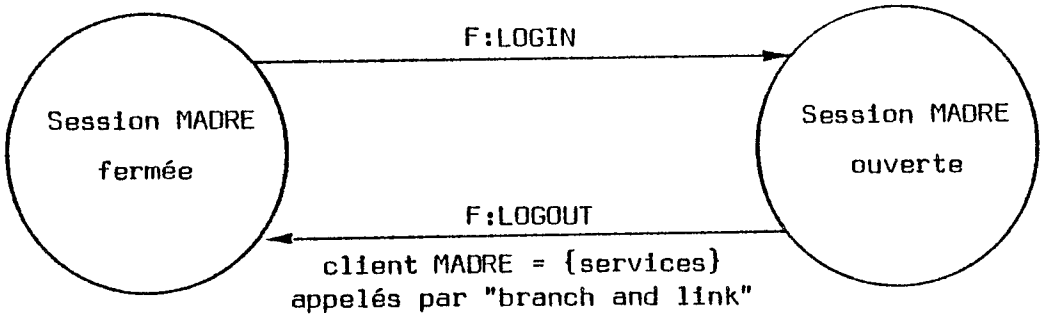
- MADRE apparaît à l'utilisateur comme un ensemble de services appelables par appel et retour (que ces services soient incorporés au système appel par CAL1/CII ou SVC/IBM, ou non) : ceci est envisageable lorsque MADRE est un produit au point, incorporé au système d'exploitation (dessin A, figure 3.8).
- MADRE se présente avec le label "service réseau" et une procédure de connexion à ce service régie par un protocole réseau ad hoc. Cette deuxième manière permet son utilisation depuis un point d'accès quelconque au réseau (en particulier les mini-ordinateurs concentrateurs de terminaux) (dessin B, figure 3.8).

b) La connexion utilisateur - fichier(s) MADRE (figure 3.9) :

Sur la requête F:CONNECT, elle se fait à l'initiative du client en trois temps :

- création d'un contexte de travail chez le client ;
- établissement d'une voie de communication entre le client et le serveur ;
- négociations d'options pour l'exploitation du fichier, par échange de lettres sur la voie (D.SET / D.RESET). Ces options concernent :
  - le contrôle d'accès pour les mises à jour,
  - le type de blocage (cf. 3.2.7),
  - le facteur d'anticipation, c'est-à-dire lorsqu'il y a accès séquentiel le nombre de blocs dont la lecture est demandée au serveur par le client en anticipant sur la demande de l'utilisateur.

A) Utilisateur et client MADRE sur un même site



B) Utilisateur et client MADRE sur deux sites différents

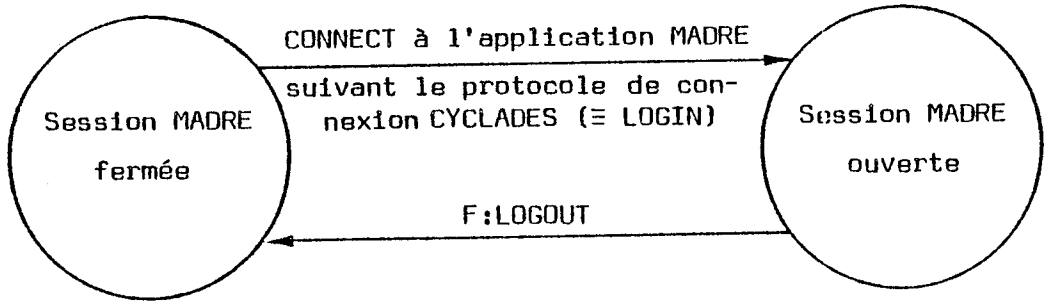


Figure 3.8

MADRE : le protocole de connexion
utilisateur - méthode d'accès MADRE

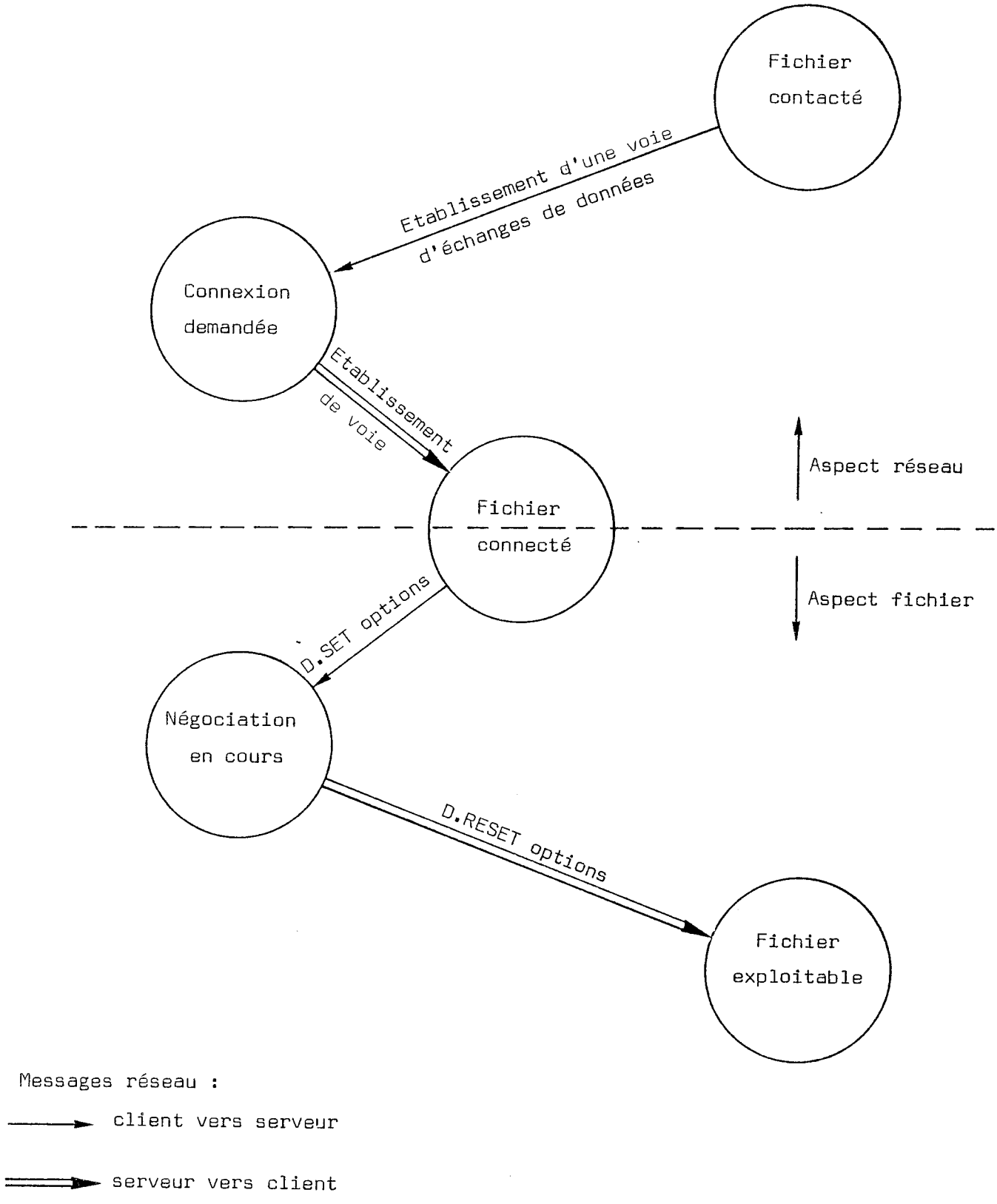


Figure 3.9

MADRE : le protocole de connexion

utilisateur - fichier MADRE

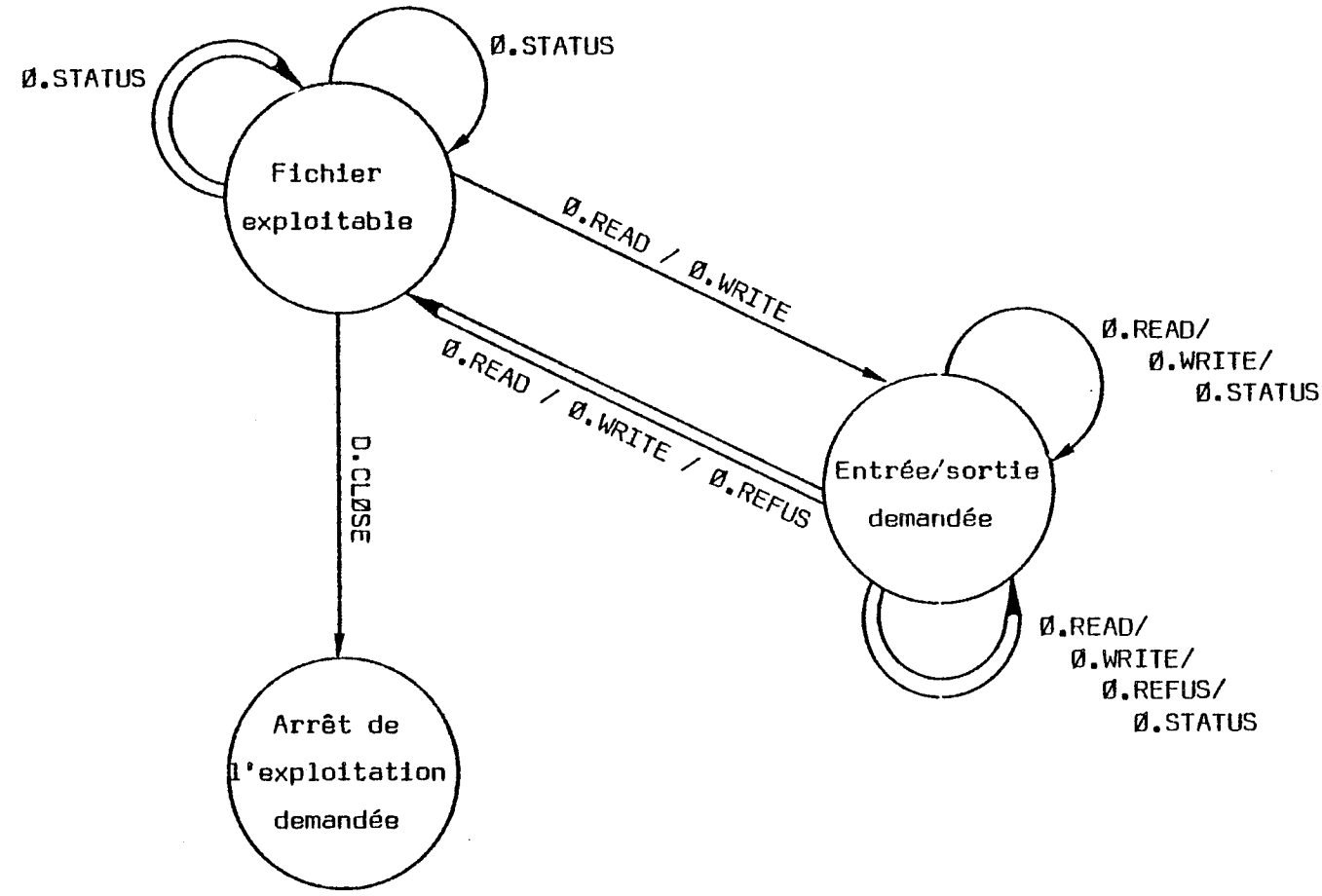


Figure 3.10

MADRE : le protocole d'exploitation

utilisateur - fichier MADRE

- la longueur du bloc,
- le facteur de blocage des articles.

c) L'exploitation : échanges de données utilisateur - fichier MADRE (figure 3.10)

Des demandes d'entrées/sorties (READ/WRITE) peuvent être envoyées sur la voie d'exploitation. Une symétrie existe dans les demandes d'écriture/demandes de lecture et réponses à l'écriture/réponses à la lecture.

Le REFUS correspond à l'incapacité du serveur à satisfaire la demande.

Le STATUS correspond à des échanges d'informations utiles pour les deux correspondants sur leur environnement (symétrie dans la question et la réponse).

Le CLOSE entraîne la suppression de la voie (à la charge du serveur).

On distingue dans ce protocole les ordres Ø et les demandes D. Les ordres ne provoquent pas de mises en attente de celui qui les donne et qui peut envoyer ordre sur ordre.

Lorsqu'il y a un jeu de questions/réponses (toujours à l'initiative du client), la question contient un indicatif que le destinataire recopie dans sa réponse pour permettre de faire la correspondance ; à charge au client, et pour chaque contexte de voie de délivrer des indicatifs uniques.

3.2.6.2. Les mécanismes client-serveur :

L'architecture de la méthode d'accès fichier s'articule autour de deux pôles ; le client et le serveur, avec un protocole d'échanges de commandes et de données entre les deux. Cette conception appelle quelques remarques :

a) MADRE regroupe plusieurs sous-systèmes, chacun d'entre eux pouvant comprendre une fonction client et une fonction serveur. La règle est qu'il y a autant de clients que de sites où des utilisateurs souhaitent utiliser la méthode d'accès, autant de serveurs que de fichiers où sont implantés des fichiers FAR.

b) Pour le compte d'un utilisateur donné, une session MADRE met en oeuvre un client (celui auquel l'utilisateur est connecté) et un ou plusieurs serveurs, suivant le nombre de fichiers exploités pendant la session. Lorsque plusieurs sessions sont ouvertes simultanément, on a des échanges client-serveur représentables comme suit :

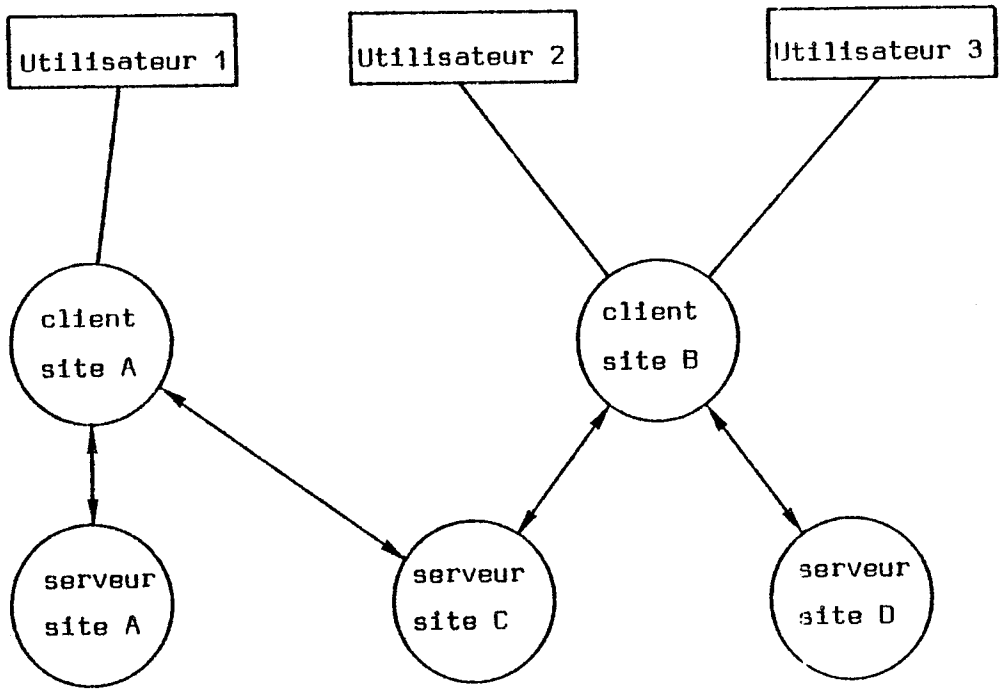


Figure 3.11
MADRE client - MADRE serveur

Ceci implique des verrous d'accès à un FAR locaux au serveur.

Pour diminuer les échanges sur la voie de communication et simplifier le protocole d'échanges de commandes et de données entre le client et le serveur, le serveur assure le minimum de fonctions compatibles avec la méthode d'accès, en laissant au client la gestion des services spécifiques. Les buts poursuivis sont :

- permettre des réalisations diversifiées des serveurs dans un environnement hétérogène et laissant une possibilité de négociation à la connexion utilisateur-fichier, entre un client qui souhaire exploiter un certain fichier et un serveur qui n'est pas toujours capable de le supporter,
- permettre au serveur de ne connaître que la notion de bloc : le découpage des blocs en articles étant géré par le client.

3.2.7. Verrouillage et protection des fichiers :

On doit envisager certains problèmes liés au multi-accès aux fichiers lorsqu'on veut réaliser les opérations suivantes :

- éditer un fichier sans qu'il soit modifié pendant la lecture,
- modifier un fichier et en interdire l'accès pendant la mise à jour,
- interdire l'accès d'un fichier pendant sa construction,
- interdire la lecture d'un fichier considéré comme secret,
- limiter la mise à jour à l'auteur du fichier,
- autoriser l'accès d'un fichier à un ensemble déterminé d'utilisateurs.

Ces six exemples posent un problème de verrouillage et de protection ; ces restrictions d'accès peuvent être modulées :

- *dans le temps* : la durée de la protection peut être limitée à une session MADRE, par exemple, ou permanente,
- *suivant le type d'accès* : lecture, écriture,
- *suivant l'utilisateur*.

Le blocage permanent introduit une notion de *fichier privé* et donc un classement des fichiers en *public* et *privé*.

Le blocage temporaire est envisagé au niveau d'une session (ou partie de session).

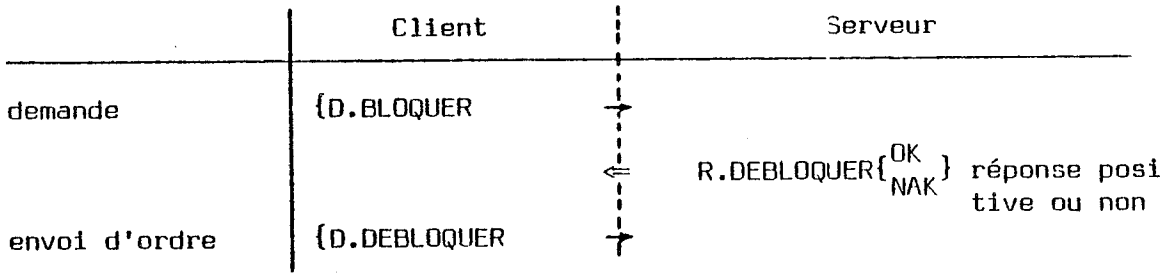
a) Le blocage temporaire :

S'il est valable pendant toute la durée de la connexion utilisateur-fichier, il est *statique* :

On choisit alors comme mode de connexion la mode de blocage souhaité : pour une session de lecture, on demande le blocage des écritures, pour une session avec des accès quelconques, on demande le blocate "tous accès".

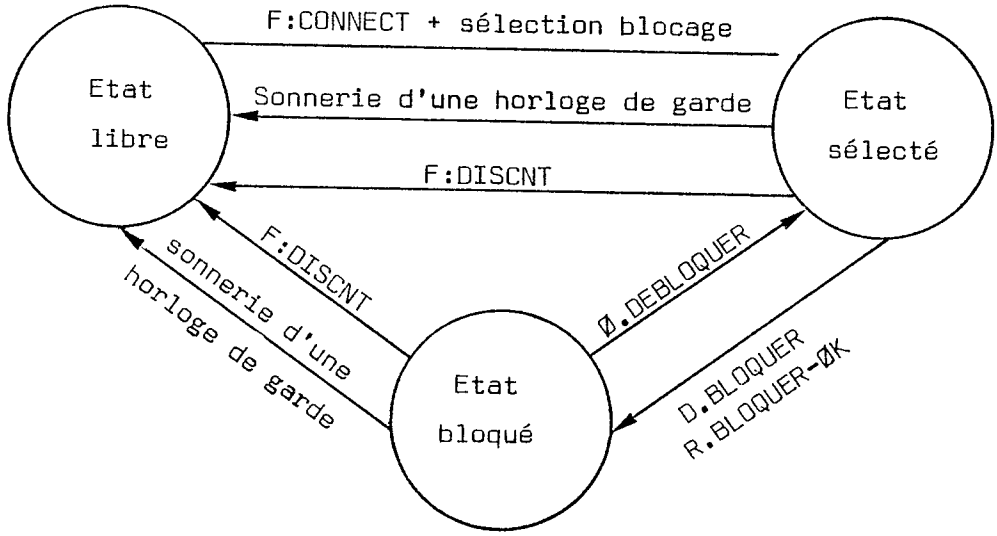
Si la solution a l'avantage de la simplicité, de la sécurité ils pénalisent les autres utilisateurs. D'autres solutions sont envisageables.

Le blocage à la demande peut se réaliser par l'échange d'ordre entre le client et le serveur en cours d'exploitation du fichier :



C'est le serveur, responsable du fichier, qui gère le blocage. Cette solution à la demande peut être complétée pour le service de l'utilisateur d'une possibilité de réservation du droit de blocage à la connexion (option dans la primitive F:CONNECT).

Ceci est acceptable lorsque la connexion n'intéresse l'utilisateur *que si* le blocage est possible au bout d'un temps fini pendant l'exploitation. Pour ce faire, le serveur doit gérer un état du fichier par rapport au blocage.



La sélection du mode de blocage est une des options négociées à la connexion.

Le diagramme d'état fait apparaître deux transitions d'états sur sonnerie d'horloge de garde, seul moyen offert au serveur de se protéger du client qui tombe en panne ou abuse de son droit temporairement accordé :

b) Le blocage permanent :

L'interdiction permanente de certains modes d'accès à certains utilisateurs fait apparaître une notion de propriété des fichiers avec des droits associés.

On dira qu'un *fichier privé* est un fichier pour lequel tout ou partie des accès sont réservés à certains utilisateurs : les propriétaires. Cette information doit être contenue dans le catalogue sous forme de titre de propriété, une preuve de la possession du titre est exigible à la connexion de l'utilisateur avec le fichier : ce peut être une option négociable, le traitement des preuves de possession n'est valablement assuré que par les serveurs.

Le titre de propriété contient un mode de protection selon lequel le serveur doit régler les conflits pour un demandeur donné, propriétaire ou non du fichier : ces vérifications sont utilisées aussi bien à la connexion qu'à chaque entrée/sortie demandée.

Cette proposition appelle quelques réflexions d'une portée plus générale : le serveur peut-il déléguer, après une connexion, certains contrôles sur le fichier au client, dans un cadre précis résultat de négociations ? Cette question peut être posée en d'autres termes : *jusqu'où peut-on favoriser l'accès aux données sans nuire à la confidentialité et à la sécurité des informations ?*

La décentralisation des contrôles d'accès favorise l'efficacité des accès mais la délégation de contrôles d'accès pose un problème de *protection* de l'information.

La chaîne de logiciels et de matériels entre l'utilisateur et le fichier est suffisamment longue pour trouver une multitude de niveaux et de moyens pour déjouer une mécanique aussi bien conçue soit-elle. Le choix d'une concentration des contrôles localement aux données peut être guidée par un souci de réduire cette chaîne d'incertitude, surtout si le propriétaire du fichier n'est pas maître de la constitution de toute la chaîne. Ce sera le cas (par exemple) dès lors que le client ne sera pas réalisé par la même personne que le serveur.

c) Le blocage temporaire d'un fichier privé :

Un fichier privé doit pouvoir être bloqué temporairement pour permettre au propriétaire de le modifier ou à un utilisateur quelconque de le lire sans qu'il soit altéré.

Un état du fichier va devoir être tenu à jour par le serveur en maintenant trois variables :

- le mode de sélection
  - aucun
  - écriture
  - tout accès
- le mode de propriété (ou de blocage permanent)
  - sans
  - écriture
  - tout accès
- le mode de blocage temporaire
  - sans
  - écriture
  - tout accès

Donc 27 états au maximum.

Ce nombre élevé souligne la difficulté qu'il y a à fournir des services variés. On notera que le mode sélecté n'apporte rien de plus pour le problème qui nous intéresse ici : il se justifie pour d'autres considérations. En effet, si l'utilisateur fait une exploitation de plusieurs fichiers, la méthode d'accès répartie doit tenir compte des interblocages que peuvent occasionner des requêtes insatisfaites simultanément.

Ce problème est un problème d'allocation de ressources pour lequel nous n'allons pas développer d'algorithmes;[MAH72], [NEG76] : des solutions ont été proposées récemment à ce problème.

Pour ce qui concerne une méthode d'accès telle que MADRE, cette allocation de ressources dans le réseau doit être envisagée en :

. respectant l'autonomie et l'indépendance de chaque serveur,

. mettant chaque site et chaque fichier sur un pied d'égalité.

3.2.8. L'adressage :

Une des nouveauté introduites par le réseau est que le moyen de désigner (de repérer un objet) n'est pas accessible sur une mémoire commune à tous les processeurs, mais diffusée par message entre chaque processeur.

L'adressage réseau est le moyen donné à l'utilisateur d'accéder à un objet, quels que soient les sites respectifs de l'utilisateur et de l'objet.

Le rôle de la mémoire commune est tenu, dans le réseau, d'une part par le réseau de communication qui prend en charge les messages porteurs d'adresses réseau, d'autre part par les catalogues d'information consultables par des processeurs différents et distants (catalogues dupliqués ou non).

L'exemple de MADRE montre que l'adressage se fait par étapes et compléments successifs, les compléments étant apportés par les processeurs sur la foi de fichiers réseau (accessibles par plusieurs dans la même chaîne) ou des fichiers locaux.

Dans un réseau hétérogène, on ne peut pas utiliser directement les termes de l'adressage local : une limitation importante est imposée par la nécessité d'interpréter ce qui est contenu dans les messages, c'est-à-dire :

. des constantes de type caractère,

. des constantes de type numérique.

La chaîne d'adressage peut être représentée comme suit : l'utilisateur manipule des étiquettes logiques (OPL) et des numéros de blocs virtuels (cf. définition des fichiers réseau) : le client retrouve dans le catalogue MADRE le fichier correspondant sur le site d'implantation ou le site d'implantation du bloc considéré donc le numéro de bloc réel ; le serveur, grâce à un catalogue, est capable d'interpréter cette adresse réseau en une adresse locale et de faire l'accès associé.

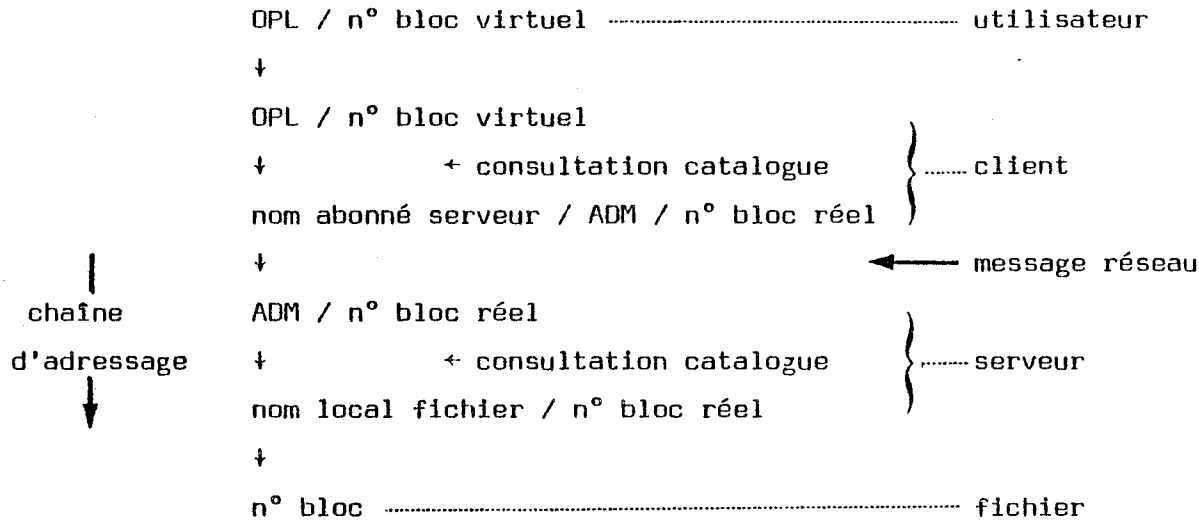


Figure 3.12
Adressage dans MADRE

3.2.9. Les fichiers bases de données :

Nous envisageons ici une application de la méthode d'accès fichier MADRE pour le système de gestion de bases de données SOCRATE.

Les systèmes de gestion de bases de données (SGBD) gèrent des fichiers bases implantés sur des supports mémoire secondaire. Sans nous intéresser, pour l'instant (cf. 3.3) à la méthode d'accès logique propre à chaque SGBD, l'accès physique aux fichiers bases se fait par une méthode d'accès direct travaillant sur des blocs physique d'informations (des pages). Dans un SGBD mono-ordinateur, cette méthode d'accès est locale. Sa substitution par une méthode fonctionnellement équivalente permet d'accéder indifféremment à des fichiers locaux et distants : on définit de la sorte un *SGBD à mémoire de masse répartie* sans qu'aucune fonction propre à ce SGBD soit concernée par cette répartition.

Dans le LADDER de [MOS77] c'est l'optique adaptée avec le FAM (File Access Manager). La figure 3.13 qui suit permet d'illustrer cette substitution avec SOCRATE et MADRE. Pour cette dernière, et au niveau réalisation, la situation initiale était un SGBD et une méthode d'accès physique travaillant avec une zone de mémoire commune : le passage en environnement réseau nécessite un remplacement de cette zone commune par un protocole d'échanges d'information entre le SGBD et la méthode d'accès réseau [B4], [C5].

Ce qui vient d'être présenté s'apparente au déport de bases. Compte-tenu qu'un serveur MADRE est obligé vis-à-vis d'un nombre quelconque de clients, le partage *en lecture seule* d'une base par plusieurs SOCRATE peut se faire sans soulever ni problèmes logiques, ni problèmes d'interface nouveaux.

Dans cet esprit, on fait apparaître les notions de *bases publiques*, *bases privées*, comme on peut parler de fichiers publics, fichiers privés (cf. figure 3.14).

La coopération, lecture seulement, peut être élargie à :

- un seul modifie,
- les autres lisent,

en utilisant les ordres de blocage/libération pour encadrer un ensemble logique d'entrées/sorties physiques. Ceci n'est possible que si tous les fichiers composant une même base sont sur un même site [C5].

Dans le cas contraire, un certain nombre de problèmes doivent être étudiés [CHP74].

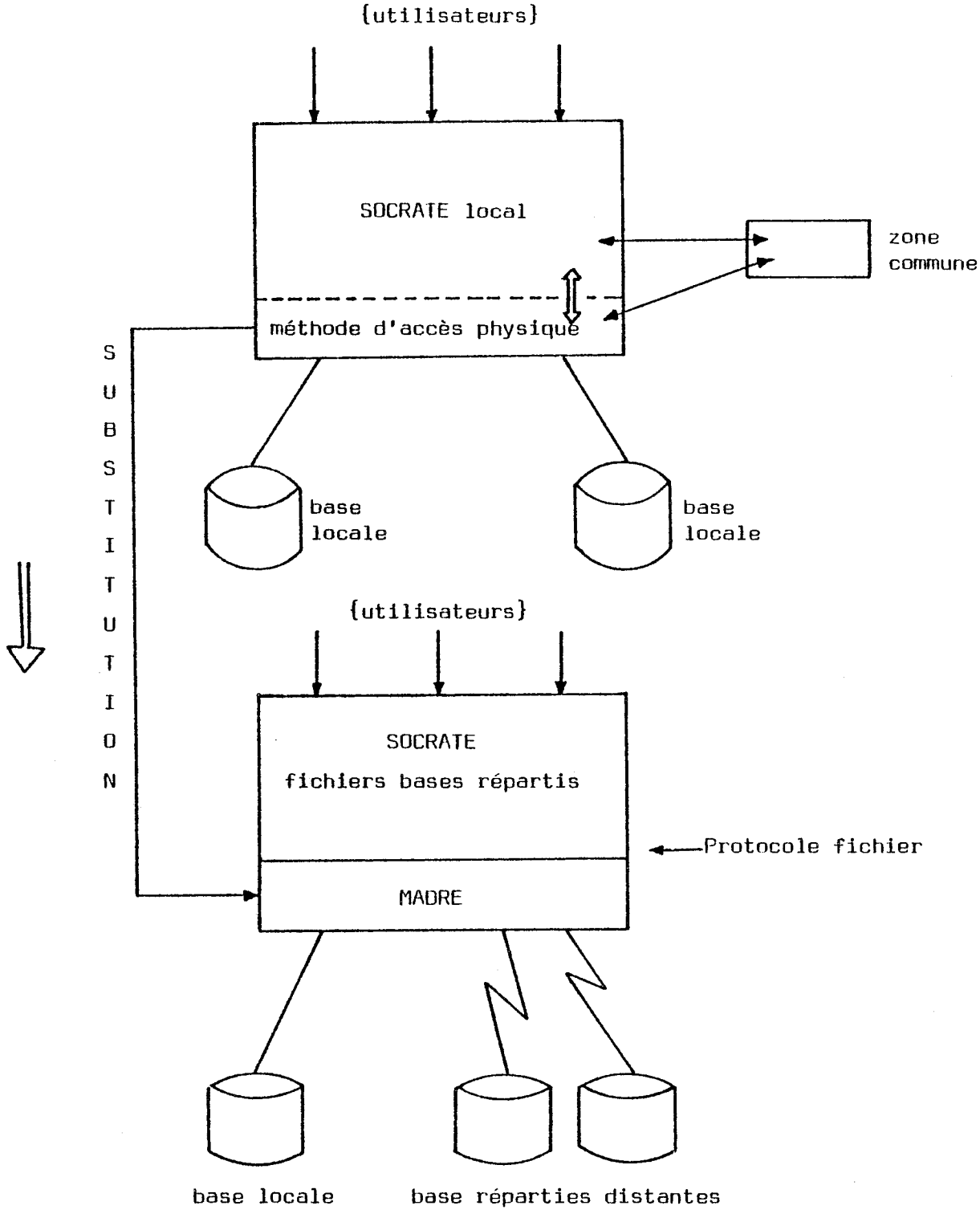


Figure 3.13

SOCRATE : la répartition des fichiers bases physiques

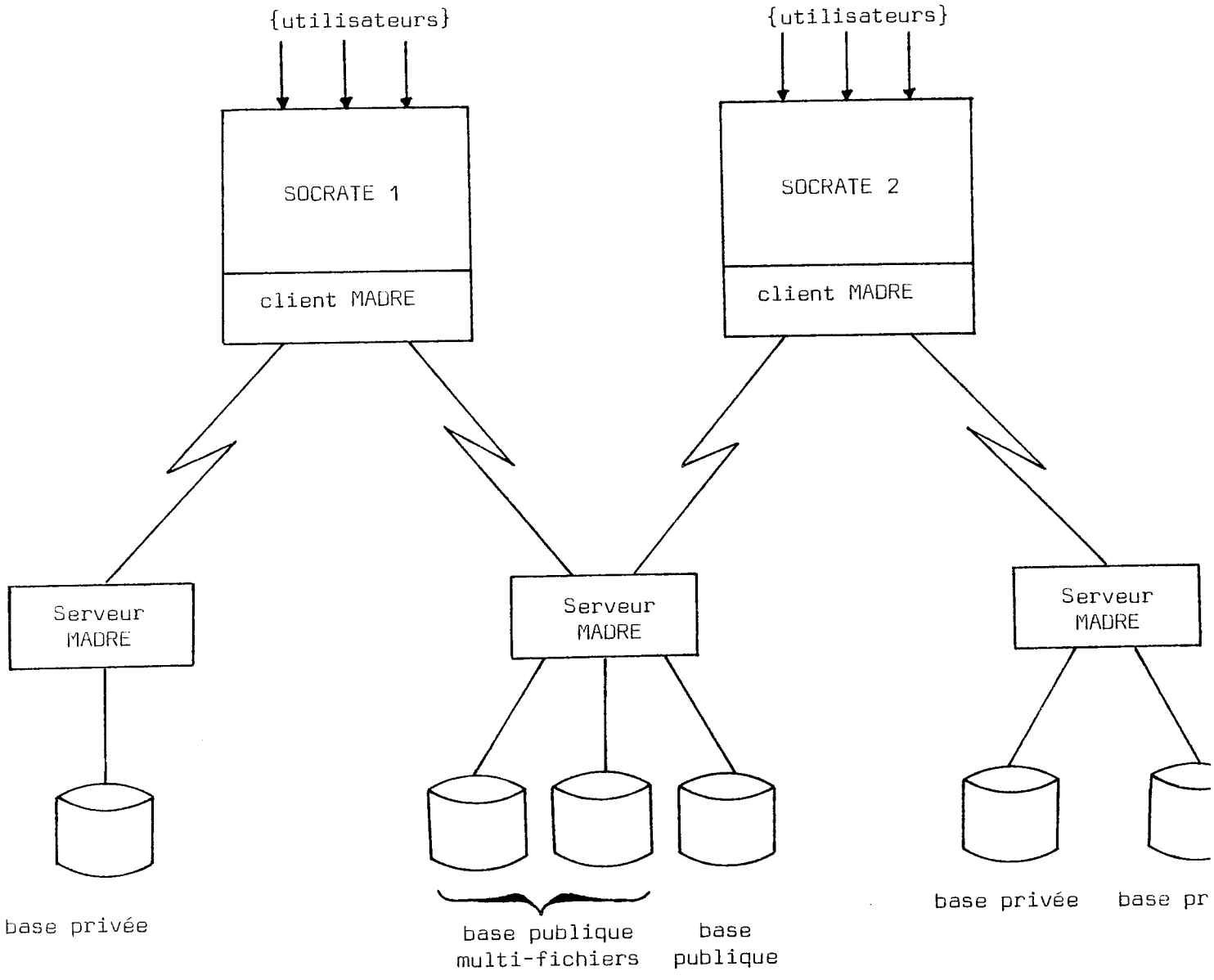


Figure 3.14

SOCRATE : la dispersion des fichiers bases physiques accessibles en LECTURE seulement par plusieurs SOCRATE (bases publiques)

3.3. Les méthodes d'accès bases de données

Au paragraphe 3.2.9. nous avons envisagé une configuration bases de données réseau dans laquelle l'interface réseau utilisé est celui mis en oeuvre par une méthode d'accès fichier. C'était la configuration "mémoire de masse répartie" dans laquelle aucun problème de logique de description et d'accès de bases n'était posé.

Les SGBD sont des systèmes qui assurent un certain nombre de fonctions :

- compilation de langages utilisateur,
- édition des résultats,
- multi-accès en conversationnel,
- cohérence des données,
- sécurité des accès et des bases,
- accès physique aux bases, etc.

3.3.1 Structuration des SGBD

Envisager de construire un SGBD comme un système distribué, c'est répartir tout ou partie de ces fonctions. Un certain nombre d'objectifs peuvent être atteints suivant la démarche suivie [CHA77][LEB76].

On peut, par exemple, situer le point de départ de cette répartition dans la proposition d'architecture de SGBD proposée par [ANX75]en vue d'une standardisation.

La figure 3.15 décrit cette architecture qui fait apparaître quatre niveaux fonctionnels. A chaque niveau fonctionnel est associé un langage de description de données (LDD) ; chaque interface entre niveaux est décrite par un langage de manipulation de données.

Chaque niveau peut être transformé en une application répartie afin de répartir les fonctions correspondantes. Ces niveaux sont respectivement la vue externe, la vue conceptuelle, la vue interne, la description des fichiers. Etudions chaque niveau en commençant par le bas :

3.3.2 Choix des interfaces réseau

a) La description des fichiers : c'est le niveau de la méthode d'accès fichier (cf. 3.2.9).

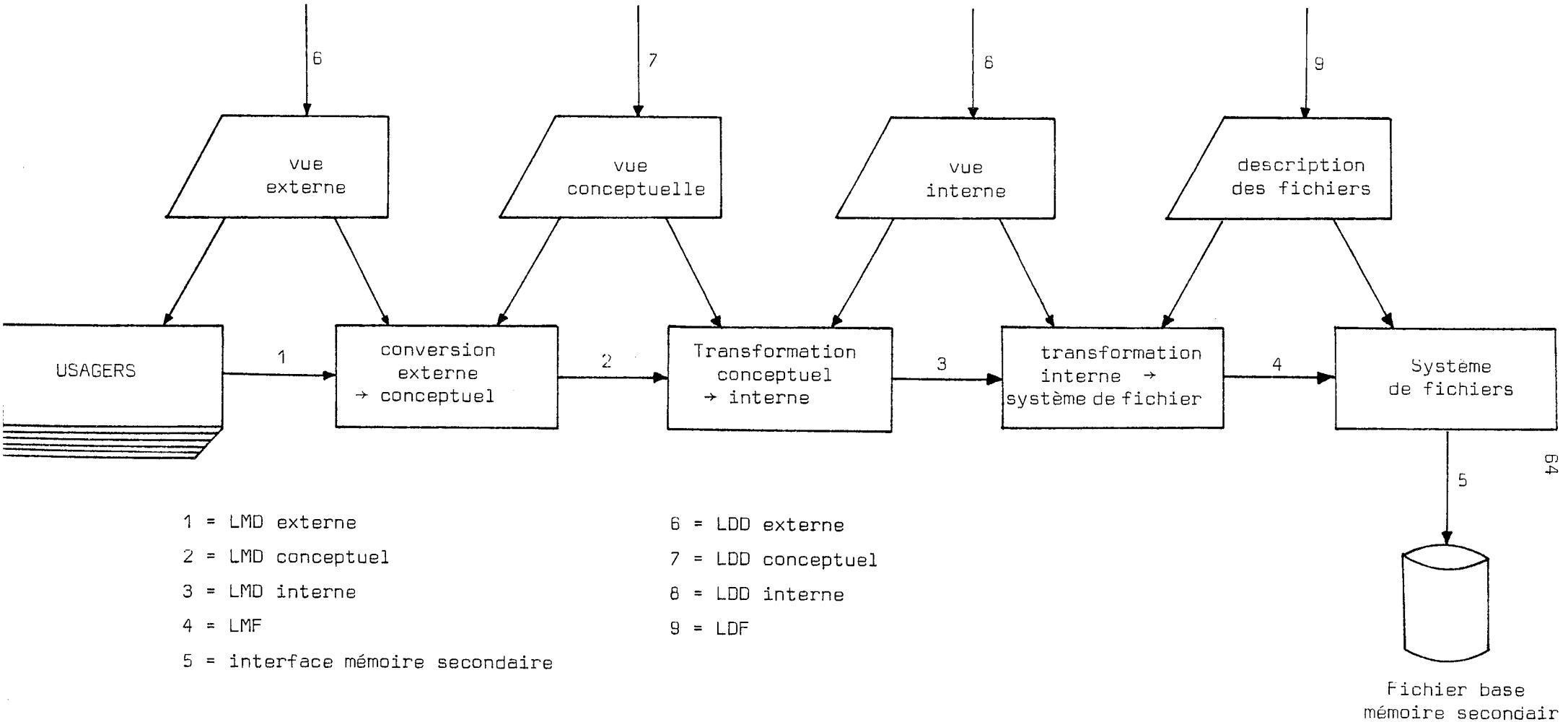


Figure 3.15

Architecture proposée par [ANX75]

b) La vue interne : C'est le niveau dans lequel le SGBD traite l'ensemble des problèmes logiques d'accès aux bases : multi-accès, recherche de données dans les bases (transformation espace virtuel-espace réel), tenue à jour des journaux de sécurité... Ce niveau correspond à celui de la méthode d'accès aux bases de données. Ce niveau de distribution a été développé dans plusieurs études [MET73], [B4], [C5] sur Socrate pour le réseau Cyclades. La thèse de J.C. Chupin [CHP77] présente de façon détaillée le projet MARS (Méthode d'Accès Réseau Socrate).

c) La vue conceptuelle : C'est le niveau de description de la structure des données dans la base par le concepteur du système d'informations.

d) La vue externe : C'est le niveau de l'utilisateur qui interroge et modifie les bases avec son langage de programmation. Placer l'interface réseau à ce niveau, c'est résoudre le problème de l'accès distant à un système de gestion de base de données. L'utilisateur est vu du SGBD comme un terminal : le dialogue, dans le réseau, sera celui d'un terminal et d'un serveur base de données suivant le protocole client - serveur et appareil virtuel en vigueur sur ce réseau.
Ce sont les objectifs de SOCYCRATE [SER75] pour Socrate dans Cyclades.

Il n'est pas dans notre propos de détailler ici les différentes vues et les différentes configurations de SGBD que l'on peut obtenir dans les réseaux. Nous renvoyons ici le lecteur à d'autres documents et en particulier ceux du projet SIRIUS [IRI76].

3.3.3. Espace virtuel et répartition

Nous détaillons seulement un aspect pour le niveau description des fichiers. La méthode de répartition développée en 3.2.9,dite de la mémoire de masse répartie, repose sur la notion de fichier, chaque fichier est sur un site, les bases sont des ensembles de fichiers. Nous détaillons ici l'aspect découpage en fichiers-sites sur l'exemple de Socrate [CII08], [ABR72].

Une base Socrate est constituée d'un espace virtuel et d'un espace réel. Le placement des objets de la base dans l'espace virtuel résulte d'un choix conceptuel, c'est-à-dire que chaque objet potentiel possible a une définition dans cet espace. La projection de l'espace virtuel sur l'espace réel est réalisée avec une méthode de repliement et de gestion de squatters (cf. fig. 3.16).

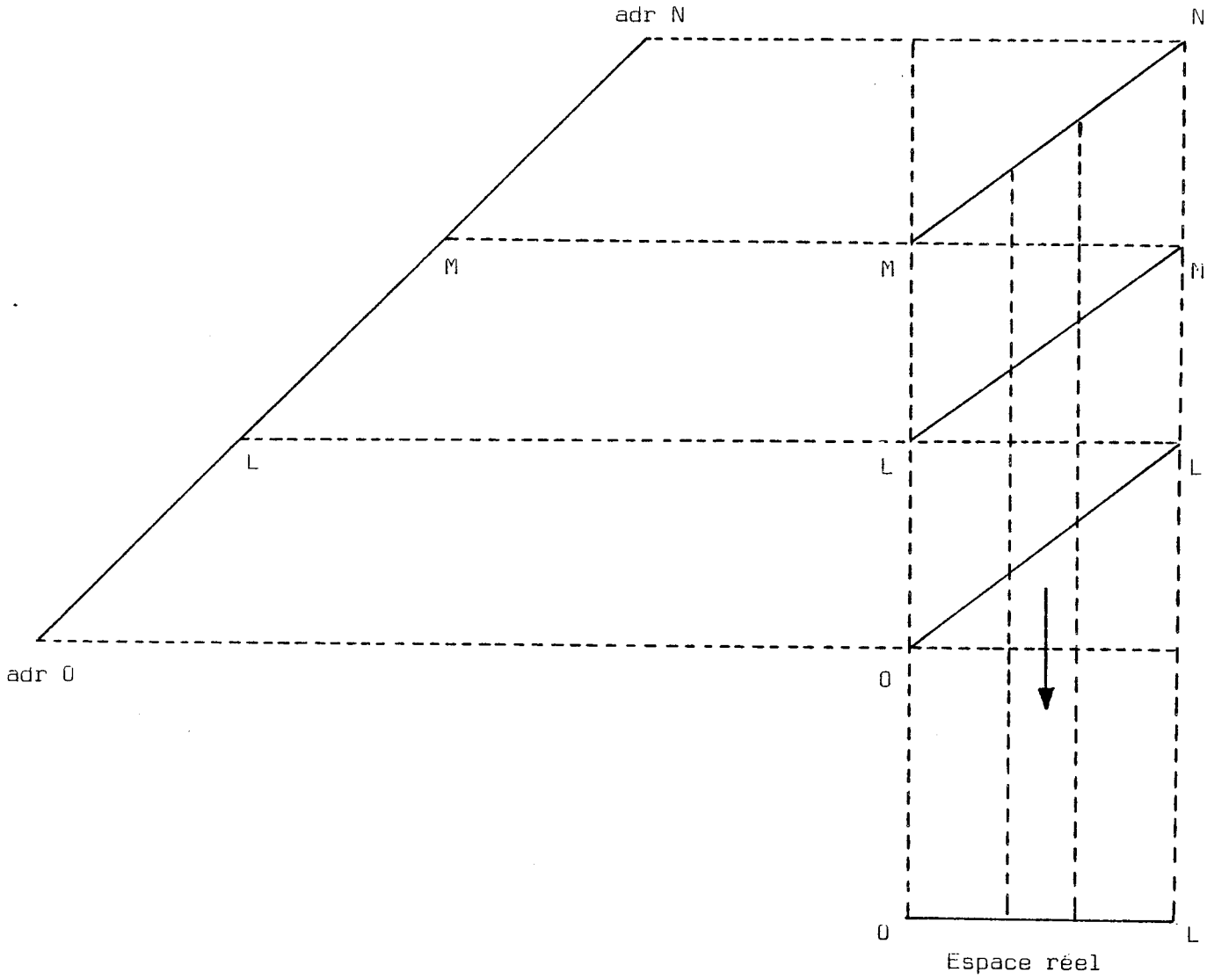


Fig. 3.16

Mécanisme de projection, espace virtuel → espace réel

Si l'on découpe l'espace réel en un ensemble de sous-espace ayant chacun un site d'implantation, il faut disposer de partition dans l'espace virtuel : chaque partition de l'espace virtuel ayant son propre mécanisme de projection (figure 3.17).
Ceci suppose :

- qu'il est facile de déterminer le fichier réel à partir d'une adresse virtuelle donnée,
- qu'il n'y est pas possible de faire déborder un fichier sur l'autre, lorsque l'un est saturé.

Cette partition trouve tout son intérêt lorsque la base peut faire l'objet de découpages logiques significatifs (telles réalisations d'entité sur un site, telles autres sur un autre), ou lorsque le site du SGBD n'a pas une capacité suffisante de stockage en ligne.

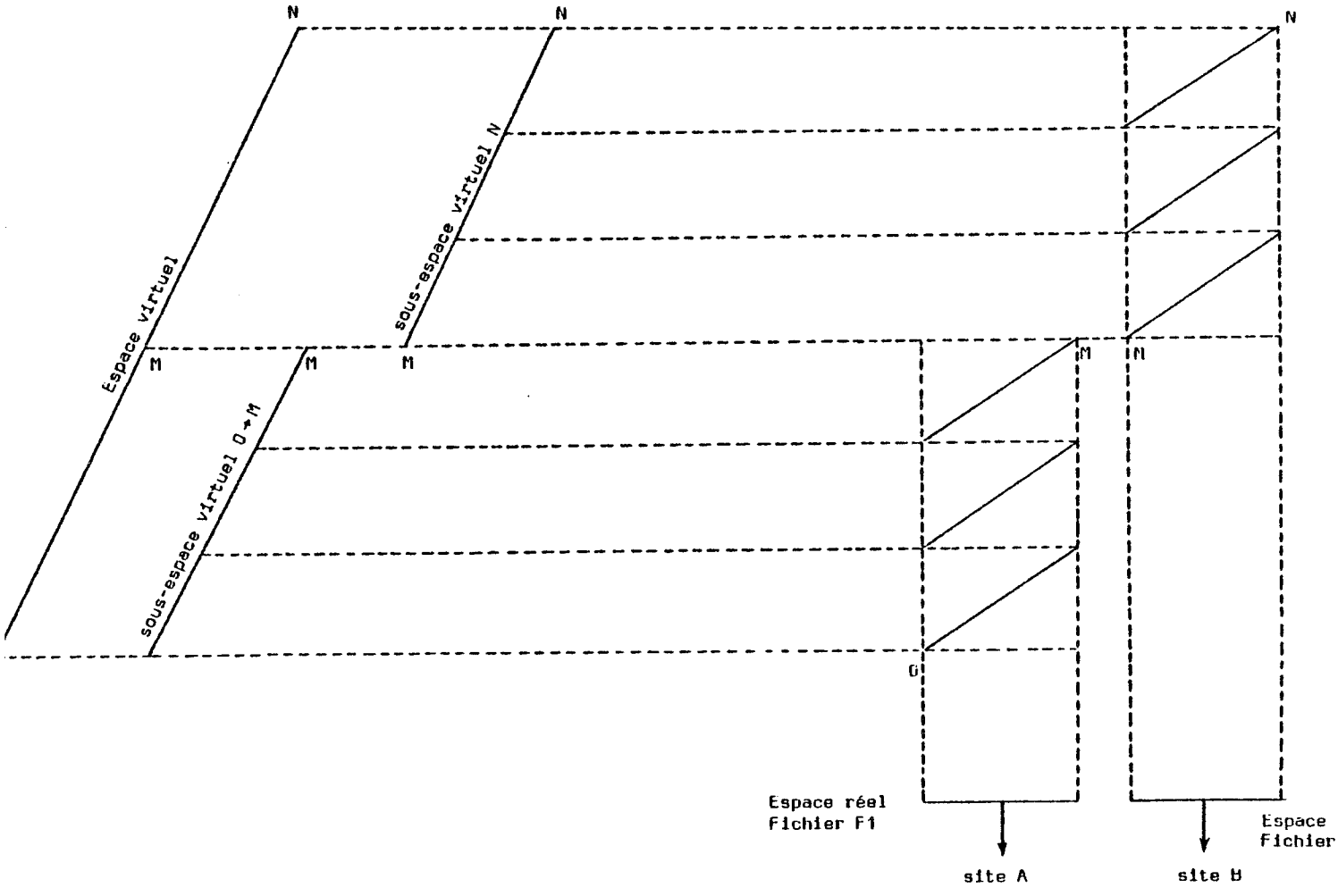


Figure 3.17

Mécanisme de projection sur deux fichiers réels disjoints

Exemple de structure SOCRATE :

```
entité personne (350000)
     nom mot
     âge (1 à 120)
     logement réfère locataire de un appartement
     entité disque (50)
        titre mot
        auteur mot
        date 1 à 1974
entité appartement (80000)
     adresse texte
     type (F1/F2/F3/F4/F5)
     superficie (1 à 130)
     locataire anneau
```

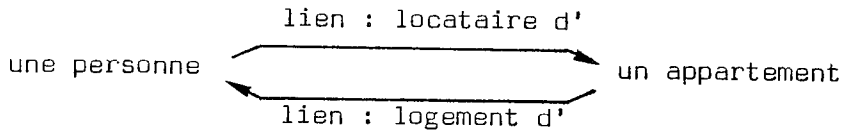
Commentaires sur cette structure :

On s'intéresse aux personnes habitant l'agglomération grenobloise (il y en a 350 000) et aux appartements existant dans cette agglomération.

Pour les personnes, on s'intéresse à un état civil succinct (nom - âge) et aux disques qu'ils possèdent (titre - auteur - date).

Pour les appartements, on s'intéresse à l'adresse - type - superficie.

Ces deux ensembles d'informations ont des liens entre eux : à un instant donné il existe un lien entre



Ce lien est destructible, modifiable, par opposition à celui existant entre une personne et un disque (marché d'occasions mis à part).

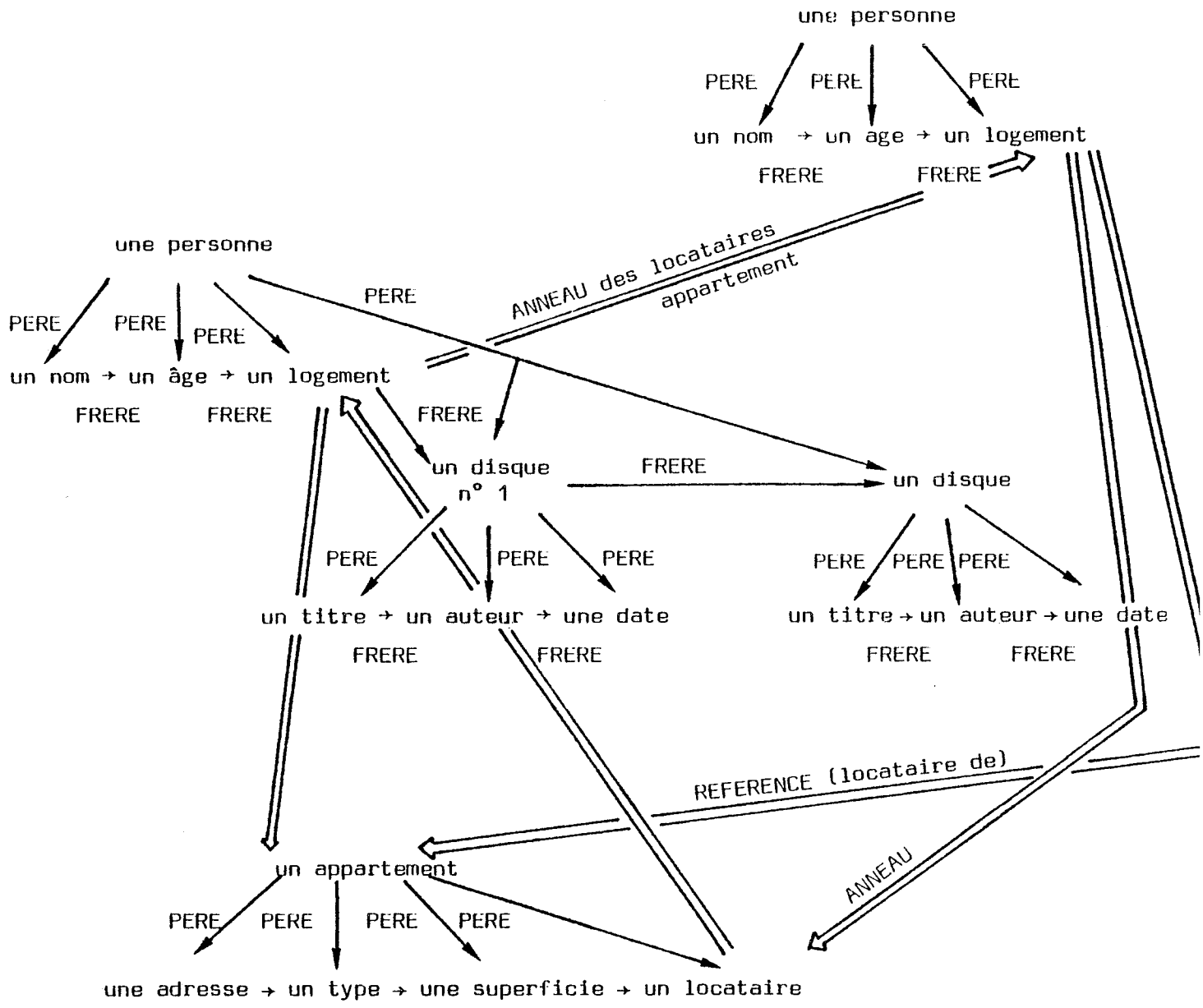


Figure 3.18

Exemple de représentation de la structure avec les liens :
deux personnes : l'une a deux disques, l'autre aucun et
elles sont locataires du même appartement

Arborescer

relation

Espace virtuel correspondant à la structure précédente :

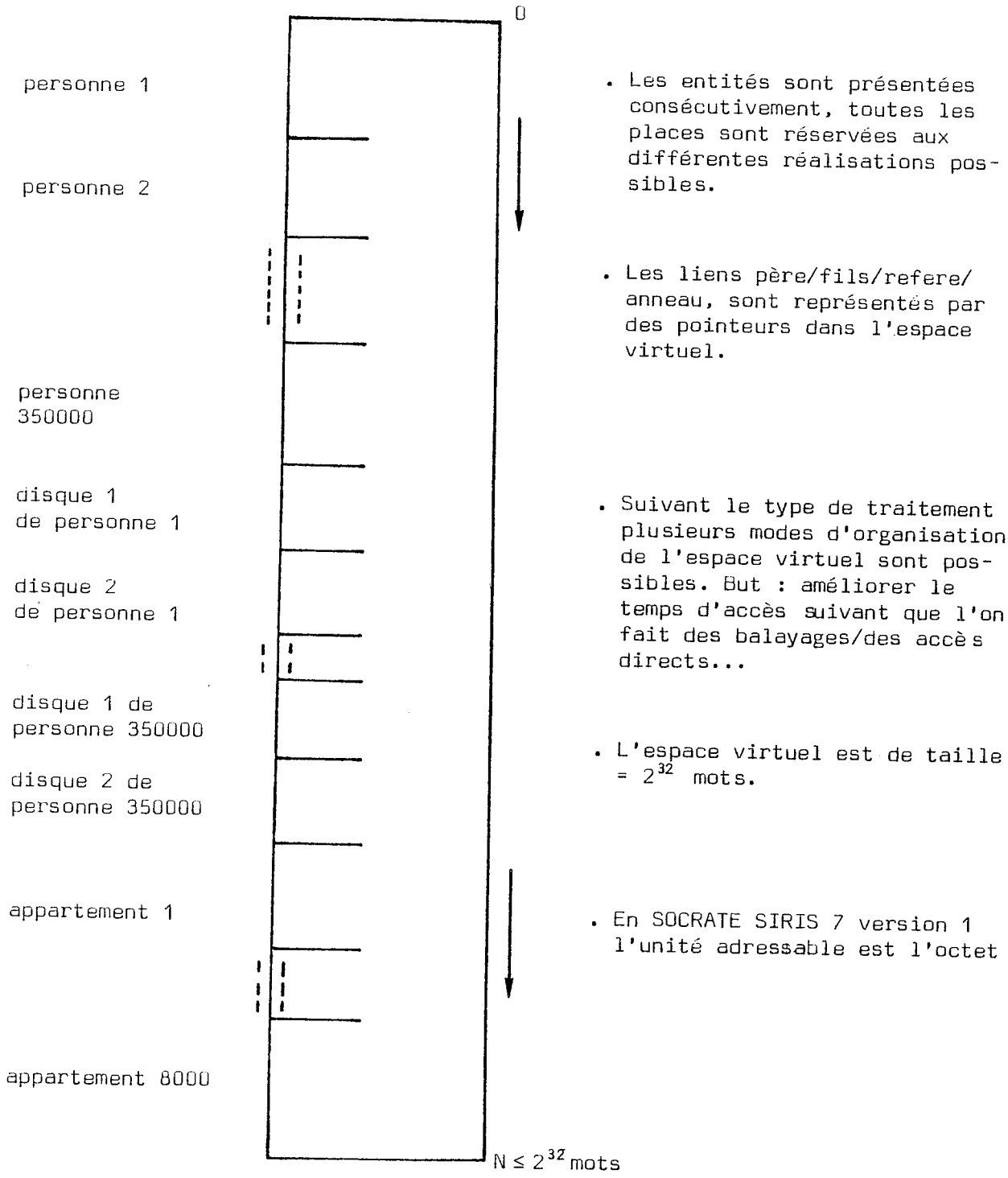


Figure 3.19 - Espace virtuel

Incidences possibles du découpage de l'espace réel sur l'espace virtuel :

L'espace réel peut se présenter sous des formes variées :

- . un fichier mono/multi-volumes
- . un ensemble de fichiers mono/multi-volumes.

Suivant la forme adoptée, on peut avoir :

- . soit, espace réel un et indivisible
- . soit, espace réel union d'espaces réels.

On peut avoir les tranches suivantes :

. en 2 tranches : une pour toutes les personnes (disques compris)
une pour tous les appartements

. en 3 tranches : une pour toutes les personnes (disques compris)
une pour les 40 000 premiers appartements
une pour les 40 000 derniers appartements

Dans la version 1 de SOCRATE SIRIS 7, seuls les découpages d'entités non imbriquées sont possibles, toutes les réalisations d'une même entité étant sur le même fichier.

Conclusion :

Les avantages de ces découpages sont multiples :

- . gérer simplement de grosses bases partitionnables,
- . permettre des montages partiels de bases dans la mesure où il n'existe pas de liens entre les sous-bases (tranches),
- . faire fonctionner un SGBD sur un site qui trouve sur le réseau une extension mémoire.

3.3.4. Les bases de données documentaires :

3.3.4.1. Position du Problème

Les systèmes documentaires sont une technique informatique en plein essor. De nombreux organismes ont déjà créé d'importants fonds documentaires dont certains sont déjà accessibles sur des réseaux d'ordinateurs : SDC Tymnet, ASE Frascati, etc. Ainsi, le centre de l'Agence Spatiale Européenne de Frascati (Italie) mettent à disposition (via Cyclades, par exemple) 12 fichiers représentant quelques 6.738.500 références de documents ; ces fichiers sont

sur des mémoires secondaires (disques de gros volume et fort débit) d'une taille de 26 x 200 Mégabytes en ligne pilotés par un IBM 360/65. En France de gros fonds documentaires sont constitués et le BNIST consacre de gros moyens à la constitution de nouveaux fonds et à leur mise à disposition sur Cyclades [IRI77].

A l'heure actuelle les systèmes documentaires (Mistral de CII-HB [CII02], Recon d'IBM) ont chacun leurs conventions : l'utilisateur réseau doit donc connaître chaque système sur chaque site, c'est-à-dire :

- le type de données que contient le fonds,
- le site d'implantation de chaque fonds,
- les règles de connexion à chaque fonds (identificateur, mot de passe, prix,...),
- le système d'interrogation.

Chaque fonds étant conçu comme un serveur réseau, l'utilisateur Cyclades, par exemple, ne peut se connecter qu'à un seul fonds à la fois. Cette restriction l'oblige s'il n'a pas trouvé les références des documents recherchés sur un site, à se deconnecter, se reconnecter ailleurs, poursuivre sa recherche, etc.

3.3.4.2 Langage Pivot et Interface Réseau

L'exploitation actuelle des systèmes documentaires dans les réseaux est donc soumis à deux contraintes fortes [KAM77] :

- une seule connexion, à un instant donné, à un fonds documentaire d'un site,
- diversité des systèmes.

Pour lever ces deux contraintes, on peut avancer les propositions suivantes :

1) La connexion simultanée à plusieurs fonds documentaires n'est pas un problème spécifique aux systèmes documentaires : c'est le problème de la connexion d'un même terminal réel à plusieurs serveurs. Le concentrateur multi-connexions [RIC76] simule des terminaux virtuels. Chaque connexion attache un terminal virtuel sur le poste du serveur correspondant. Toutes les connexions établies pour un terminal réel sont multiplexées sur ce terminal. Moyennant un certain nombre de dispositifs de partage du support terminal, l'utilisateur peut exploiter simultanément ses connexions, comparer les résultats, faire des rapprochements sémantiques entre

les données qu'il reçoit de sources différentes.

2) La proposition précédente ne concerne pas la sémantique des données. [KAM77] propose, dans le cadre de Mistral version 3 un système SYDRE qui donne à l'utilisateur l'impression de travailler sur un fonds unique et réparti ; le système se chargeant d'interpréter les commandes, faire l'interface avec chaque fonds local, synchroniser les différentes requêtes et synthétiser les résultats.
La diversité des systèmes sera surmontée en étendant un système, comme SYDRE, par la définition d'un *langage pivot* d'interrogation de fonds, ce langage étant interprétable par chaque système local.

La simplicité des systèmes documentaires, qui fait leur succès auprès des utilisateurs, rend aisée la définition d'un tel langage. Mais les utilisateurs tiennent trop à un outil stable et simple pour le laisser modifier par des informaticiens qui tiennent trop aux choses complexes : c'est peut être la raison pour laquelle la définition du langage pivot n'est pas plus avancée. Les conditions d'exploitation des bases documentaires sur les réseaux vont peut être rendre indispensable un rapprochement des points de vue entre utilisateurs jaloux et informaticiens hautains : économie oblige !... ou nationalisme ! lorsque les français verront que c'est un bon moyen de ramener en France ceux qui, à bon marché, accèdent aux bases américaines !

4. LA DUPLICATION DE L'INFORMATION

Nous abordons dans ce chapitre la duplication de l'information. Nous ne faisons pas d'hypothèse sur la nature de cette information, son rôle dans le réseau, sa structure interne : nous la considérons comme regroupée en des ensembles de fichiers ; chaque fichier sera considéré comme l'unité de base dupliquée et sera parfois appelé *objet* dans la suite.

La duplication trouve, d'abord, sa justification dans la prise en compte de critères d'allocation de fichiers ; ces modèles d'allocation ont été développés ; cette allocation étant réalisée, il conviendra d'être capable de localiser chaque copie lorsque le choix de la duplication aura été fait (paragraphe 4.1).

L'information étant vivante, il faudra l'actualiser ; si elle est dupliquée, on aura besoin d'un mode de gestion de copies pour des mises à jour afin d'assurer la meilleure cohérence possible de l'ensemble des copies (paragraphe 4.2).

Nous proposons au paragraphe 4.3 un mode de gestion d'objets dupliqués dans un système réparti:SYNDIC, avec un certain nombre d'objectifs de cohérence et de fonctionnement en mode dégradé.

Les applications de ces mécanismes sont envisagés au paragraphe 4.4 ainsi que les enseignements à en tirer pour les systèmes répartis en général.

4.1. Les problèmes d'allocation et de localisation [ESW74], [MOR77]

Dans les systèmes répartis, la duplication d'objets apparaît pour :

- une augmentation de la disponibilité des objets : lorsque le système réparti peut fonctionner en mode dégradé avec les copies restant en ligne,
- une amélioration de la sécurité et de la fiabilité du système global lorsqu'il s'agit d'objets-système,
- une plus grande indépendance des processus du système en associant une copie à chaque processeur ou sous-ensemble de processeurs,

- une diminution du trafic réseau pour les accès locaux aux objets.

Plusieurs études théoriques ont été menées sur la question de savoir :

a) quels objets faut-il dupliquer ?

b) combien de copies faut-il créer et gérer ?

c) sur quels supports de quels sites faut-il allouer ces copies ?

d) comment localiser des copies existantes ?

Les trois premières questions doivent être envisagées en parallèle. Plusieurs facteurs sont pris en compte, on peut citer [MIR76] :

- le nombre d'accès aux fichiers depuis les différents sites,
- le trafic induit par ces accès,
- le rapport entre les mises à jour et les lectures d'information,
- la taille du fichier,
- la capacité et le coût des lignes de transmission et des différents support de communication,
- la capacité et le coût de stockage des mémoires support,
- la disponibilité.

D'autres facteurs plus complexes dans leur définition peuvent intervenir comme l'évolution dans le temps de chaque facteur, la propriété, la disponibilité "physique" des données...

Les études de modélisation se décomposent en deux types : les modèles statiques dans lesquels les variations au cours du temps ne sont pas prises en compte, les modèles dynamiques qui prennent en compte ces variations.

Les modèles statiques :

[CHU68] cherche à minimiser les coûts de stockage et de communication avec par hypothèse des temps d'accès et des capacités de stockage bornés. Ses résultats sont applicables pour un petit nombre de fichiers sur un réseau de quelques noeuds.

Pour déterminer l'allocation optimale de fichiers, [CAS72] introduit le facteur P = "taux de mise à jour / taux d'interrogation". Il démontre l'existence d'une valeur P_0 telle que lorsque $P > P_0$ le choix de fichiers centralisés est la meilleure. Il modèle, par ailleurs, pour un petit nombre de sites, le coût d'allocation des fichiers en intégrant les facteurs coûts de communication, de stockage ainsi que les trafics dus aux accès fichiers. Ces résultats ont trouvé des illustrations dans le réseau ARPA.

Les modèles dynamiques :
On peut citer les études et modèles de [LEW75] dans lesquels il est tenu compte de la dépendance existante entre données et programmes d'accès à des données. Ces modèles intègrent des taux d'accès aux fichiers variable dans le temps.

Ces modèles représentent un grand intérêt pour les concepteurs de réseaux informatiques en leur fournissant des outils d'évaluation économique des différents composants des architectures ; ces modèles peuvent aussi aider à certaines décisions dans le choix d'implantation des fichiers, des programmes et des compléments logiciels et matériels à apporter au réseau en fonction de certains coûts de congestion.

Les *problèmes de localisation* (question d) précédente) ont été illustrés au chapitre 3 avec les fichiers et l'utilisation ou non du catalogue de fichiers.

Les *modalités de désignation :* si on donne un *nom unique* à toutes les copies, c'est que l'on garantit à l'utilisateur que quelle que soit la copie à laquelle le système lui donnera accès (par exemple, celle pour laquelle les accès seront les moins coûteux), il disposera d'une version dont le rafraîchissement est assuré par le système.

Si on *distingue* dans leur désignation chaque copie, chaque copie devient un fichier classique. On sépare alors désignation et cohérence, ce qui revient à laisser à l'utilisateur la responsabilité du choix de la copie et de son actualisation périodique.

Dans MADRE, le catalogue de fichiers qui est dupliqué a un nom unique pour tous les utilisateurs ; la méthode d'accès traite les accès au catalogue, pour un site donné, sur la copie de ce site.

4.2. Cohérence et gestion d'objets dupliqués

Le choix, duplication ou pas des objets dans un système distribué, étant fait, il importe de voir comment assurer la cohérence d'objets dupliqués. Nous dirons :

4.2.1. Définition de la cohérence

Deux copies d'un même objet sont cohérentes lorsqu'elles correspondent à un même niveau de connaissance (mêmes informations, ayant éventuellement des structures différentes).

Cette cohérence peut être maintenue de façons différentes : nous en distinguerons deux :

La *cohérence stricte* correspond à deux copies cohérentes au même instant sur deux sites différents. Elle permet à tout utilisateur d'accéder indifféremment à une copie quelconque tout en étant assuré de disposer d'une copie à jour. Dans un système réparti le maintien de la cohérence stricte est une contrainte *forte* qui ne peut être assurée dans de nombreux cas : défaillance d'un processeur quelconque, isolement d'un site du réseau... D'autre part, cette contrainte n'est pas toujours indispensable : objet non vital dans un système, remise à jour différée. On peut, par exemple, se contenter de savoir que telle copie est ou n'est pas à jour en sachant qu'au bout d'un temps fini il y aura une opération de réactualisation de l'information.

Pour tenir compte de ces contraintes plus faibles, nous définirons une *cohérence lâche*.

Définition : deux copies d'un même objet sont en cohérence lâche lorsqu'on peut les ramener au bout d'un temps fini en cohérence stricte.

La cohérence lâche se justifie par exemple lorsque le système réparti opère des ordonnancements de mise à jour sur chaque site avec exécution de ces mises à jour à des fréquences variables, ou encore lorsque chaque site souhaite effectuer ses mises à jour dès leurs formulation, les autres sites pouvant se contentrer pendant une certaine période d'informations non rafraîchies. Dans ce dernier cas, on fera apparaître deux types de transactions pour les lectures d'informations : une lecture *directe*

où l'information est délivrée en l'état sans précautions particulières et une lecture *cohérente* où l'information est délivrée qu'après s'être assurée qu'elle correspond au dernier niveau de version.

En cohérence lâche, une remise à niveau d'une copie sera toujours effectuée à la demande du site responsable de la copie dès qu'une mise à jour est effectuée ou de façon différée à la suite d'une panne du réseau de communication ou à la fin d'une période d'isolement d'un site du réseau ou encore à des fréquences fixes [MEY76], [HOR73], [WIL77], [ELI77].

Comme nous allons le voir dans la suite, les modalités de gestion des objets dupliqués seront dépendants du type de cohérence choisi.

Quel que soit ce type, pour contribuer au maintien de la cohérence, on considère que chaque copie est gérée par un moniteur dit *"moniteur de gestion de copies"*. Chaque moniteur est commandé soit par un utilisateur local (demande de lecture ou de mise à jour d'informations) soit par un moniteur distant (ordonnancement des mises à jour, synchronisation pour mise à jour, communications de versions à jour...). En sortie, il effectue les opérations sur les copies proprement dites et participe aux opérations inter-moniteurs (un exemple de moniteur sera développé au paragraphe 4.3).

4.2.2. Etats de copie :

Chaque copie est caractérisée par un état. La liste des états est fonction du type de cohérence. Pour la *cohérence stricte* [HOR73] et [MUL75] distinguent trois états dans leurs stratégies de gestion de copies :

- un état *libre* où la copie est à jour et dans lequel les demandes de lecture locale sont traitées,

- un état *intermédiaire* (ou *réservé)* dans lequel chaque copie transite pour permettre à un moniteur d'obtenir le droit de faire une mise à jour locale (les demandes de lecture locale étant toujours acceptées),

- un état *bloqué* pendant lequel le demandeur de la mise à jour peut lancer ses opérations d'actualisation de l'information et diffuser les modifications correspondants sur l'ensemble des copies.

Pour l'utilisateur qui fait des accès en lecture sur la copie "la plus proche de lui", il y a deux états.

- un état *stable* et *ouvert* aux accès,

- un état *instable* et *fermé* aux accès.

Pour la cohérence lâche [MEY76], [B19] font apparaître les états suivants :

- un état *à jour* où la copie a pris en compte toutes les modifications apportées aux informations ; elle est active sur le réseau et le système réparti est opérationnel (cf. chapitre 4.3),

- un état *instable* où la copie est en cours de modification dans le cadre d'un système réparti opérationnel,

- un état *incertain* où la copie n'est peut être pas à jour dans le cadre d'un système réparti non opérationnel ou d'un site déconnecté du réseau.

Pour l'utilisateur il y a trois états :

- un état *instable* et *fermé* aux accès,

- deux états *stables* et *ouvert* aux accès : dans l'un de ces deux états, il y a incertitude sur le niveau de version de la copie accédée.

4.2.3. Propriétés du réseau de communication :

Les différentes stratégies de gestion d'objets dupliqués sont mises en oeuvre sur les réseaux comme des applications réparties dont les entités, ayant des échanges d'informations à travers le réseau, sont les moniteurs de gestion de copies. Nous nous intéressons ici aux propriétés minimales du réseau de communication, sans lesquelles les objectifs de cohérence ne sauraient être atteints.

Les moniteurs de gestion de copies (ou moniteurs) sont des automates d'états finis pour lesquels les transitions d'états s'opèrent à la suite de décisions du moniteur prises au bout d'un temps fini. Dans un système ayant plusieurs échelles de temps comme les réseaux informatiques, cette notion de temps est locale au site du moniteur.

Nous nous plaçons ici dans le seul cas où le réseau est conçu de façon répartie : dans le cas contraire il y a toujours un site qui fait office de maître et de responsable de l'ensemble des copies et les échanges sont du type question-réponse entre le site maître et les autres sites esclaves ; tous les problèmes d'ordonnancement des requêtes sont réglés au niveau du seul site maître qui a évidemment une seule échelle de temps [MAR76], [HOR73], [ELL77].

Pour les systèmes répartis, nous reprenons la classification que propose [LEL77] :

Les moniteurs peuvent avoir l'une des trois propriétés suivante :

P1 : les délais de propagation de l'information entre deux moniteurs sont connus avec exactitudes a priori, finis mais peut être différents (*systèmes référentiels parfaits*).

P2 : les mêmes délais sont variables, finis mais connus avec exactitudes (*système multi-référentiels)*

P3 : les mêmes délais sont variables, finis mais connus a postériori avec exactitude (*système multi-référentiels presque parfaits*).

Aux deux questions :

Q1 : les moniteurs ont-ils connaissance d'un état global du système ?

Q2 : les moniteurs ont-ils connaissance d'une même séquence d'événements ?

les réponses apportées sont différentes suivant les propriétés :

	P1	P2	P3
Q1	oui	non	oui
Q2	oui	non	non

L'ordonnancement des mises à jour dans une gestion décentralisée n'est pas possible lorsque les propriétés P2 et P3 sont vérifiées. D'autre part les échanges de message entre moniteurs à l'aide d'un réseau à commutation des données doivent assurer un contrôle d'erreur et un contrôle de flux pour éviter les attentes infinies.

4.2.4. Gestion centralisée ou décentralisée : [MEY76], [HOL73], [MAR76], [ELL77]

La *gestion centralisée* est particulièrement bien adaptée pour les réseaux en étoile. Dans les autres cas, elle ne se justifie que lorsqu'il existe une autorité privilégiée sur le réseau (moniteur site maître) dont la fiabilité est suffisante pour limiter les situations de blocage à ce qui est compatible avec l'application.

La figure 4.1 illustre le mécanisme qu'une telle gestion met en oeuvre. Les moniteurs traitent les demandes des utilisateurs, les messages inter-moniteurs, les opérations sur les fichiers. La présence d'une version de référence sur le site du moniteur maître n'est pas indispensable (cf. chapitre 4.2.7).

Sur une demande de mise à jour (opération ①) formulée par un utilisateur du moniteur esclave site 1 ; celui-ci la reconnait et la transmet (②) au moniteur site maître qui l'accepte, la refuse ou la met en file d'attente (③). S'il l'accepte (ou lorsqu'il l'accepte), il effectue les mises à jour demandées (④) ainsi que la diffusion de ces mises à jour sur les copies existantes (⑤ en diffusion).

Il signale la fin de la mise à jour au moniteur demandeur (⑥) qui retransmet à l'utilisateur (⑦).

On peut mettre en oeuvre avec une gestion centralisée, aussi bien une cohérence stricte que lâche.

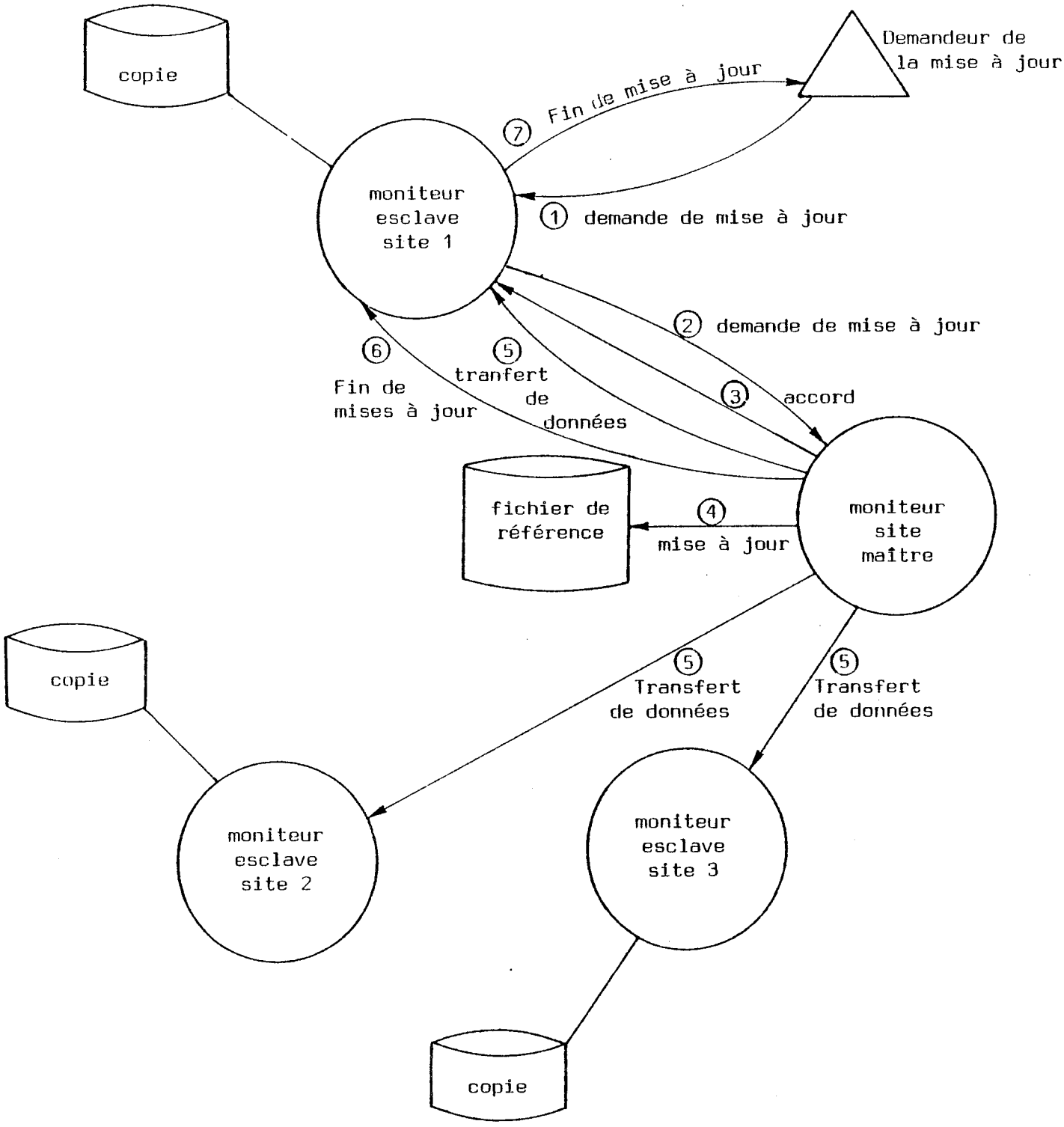


Figure 4.1

Mise à jour d'un fichier à copies multiples

avec un fichier de référence, un site maître (solution centralisée)

La gestion *décentralisée* permet d'éviter que la défaillance d'un seul site (celui du moniteur maître) ne paralyse tout le système. Deux types de solutions sont proposées :

- la première est illustrée par [HOL73] et [MUL75]. Le moniteur qui cherche à faire une mise à jour tente d'obtenir le droit unique de faire une mise à jour. Ce droit étant acquis, toutes les copies se trouvent *synchronisées* (état bloqué) ; l'ensemble des mises à jour sont faites et diffusées à toutes les copies pendant cette phase de synchronisation. La solution se trouve dans la définition de cette phase de synchronisation décomposable en deux temps :

  . une *séquence critique* : obtenir des autres moniteurs le droit de faire la mise à jour,

  . une *séquence de mise à jour* pour laquelle des procédures de reprise en cas de panne sont envisagées par [LAM75], [ELL77] pour forcer la fin de la synchronisation et le déblocage des moniteurs.

- la seconde est illustrée par [B19]. A un instant donné, un moniteur est considéré comme étant celui ayant la dernière version de l'objet dupliqué : il a un statut particulier (administrateur). Lorsqu'un moniteur veut faire une mise à jour, il cherche à devenir administrateur en adressant sa demande à l'administrateur en titre ; après avoir obtenu son nouveau statut, il effectue sa mise à jour et diffuse ensuite sa copie modifiée ; la nouvelle version ne sera validée que dans certaines circonstances (cf. notion de quorum et de système réparti opérationnel, chapitre 4.3).

Lorsque l'administrateur est absent, et sous certaines conditions de fonctionnement, un mécanisme de sondage à l'initiative d'un moniteur - dit "formateur" - permet d'élire un moniteur administrateur parmi ceux qui sont actifs.

La gestion décentralisée suppose l'existence, en environnement réparti, de mécanismes d'entrée en séquence critique ou de passation d'un privilège (celui de faire la mise à jour).

4.2.5. L'exclusion mutuelle :

Dans [DIJ74], Dijkstra définit un système distribué stable. Il considère un système constitué d'un ensemble de moniteurs disposés en anneau, chaque moniteur ayant connaissance uniquement de ses deux voisins immédiats. Sur cet anneau circule un témoin (de gauche à droite par exemple), le moniteur ayant le témoin a la faculté de rentrer en séquence critique ; s'il ne le désire pas,ou à la fin de la section critique, il transmet le témoin à son voisin de droite. Dijkstra démontre que moyennant la définition d'un nombre fini d'états pour chaque moniteur (trois au minimum), l'anneau pourra se stabiliser au bout d'un temps fini, rendant possible la circulation du témoin.

Après une démonstration de l'algorithme de Dijkstra, [MOV77] propose des solutions aux problèmes provoqués par des défaillances du système (perte du témoin, unicité du témoin...).

Ce mécanisme, comme celui développé par Le Lann dans [LEM77], est applicable à la gestion décentralisée de copies de fichiers. Elle est cependant peu économique mettant en oeuvre des procédures complexes et nécessitant un accès exclusif au fichier même pour les lectures.

On peut noter également pour ces systèmes organisés en anneau que les mécanismes de reconstitution de l'anneau ne font pas d'hypothèse sur le nombre de moniteurs présents sur l'anneau et, de ce fait, plusieurs sous-anneaux peuvent se constituer sur lesquels un moniteur pourra rentrer en section critique. Pour les fichiers à copies multiples, ceci n'est acceptable que si on s'autorise à avoir, à un instant donné, deux versions dites à jour mais indépendantes, donc incompatibles. Dans le cas contraire, où on souhaiterait garantir l'unicité sur tout le réseau d'une seule version à jour, le mécanisme n'est pas applicable (cf. 4.3). C'est le cas pour tout fichier ou base de données pour lesquels l'ordre des mises à jour se fait par consultation d'un journal de modifications et non par écrasement de la dernière version du fichier par une nouvelle version.

4.2.6. La priorité des sites :

Dans les algorithmes de gestion décentralisés, on retrouve partout des hypothèses discriminatoires sur les moniteurs du système répartis :

- dans certains systèmes répartis en anneau, les moniteurs sont rangés de gauche à droite suivant une règle d'ordonnancement fixée a priori :

le moniteur "le plus à gauche" P_0 a un rôle particulier : en effet la panne de P_0 est plus que la panne d'un moniteur quelconque qu'on isole de l'anneau puisqu'il faut retrouver un nouveau P_0. Ce privilège peut amener à définir un moniteur P_0 particulièrement fiable en le privilégiant matériellement dans la construction du système réparti.

- dans l'algorithme de Mullery [MUL75], à chaque moniteur est associé une priorité donnée a priori ; toutes les priorités sont différentes pour éviter les interblocages. Elles interviennent comme suit : une copie dans l'état "réservé par moniteur de priorité i" ou "état R_i" peut passer dans l'état "réservé par moniteur de priorité j" ou "état R_j" si et seulement si $j > i$.

- dans l'algorithme Seguin-Sergeant-Wilms [B19], le même mécanisme de priorité (appelé numéro d'identification) intervient dans la procédure de sondage pour éviter les conflits entre deux formateurs concurrents.

Ces signes distinctifs attachés à des moniteurs tendent à accorder, dans un système réparti, des privilèges à certains moniteurs par rapport à d'autres.

4.2.7. Les mécanismes de reprise :

Parmi les algorithmes de gestion d'objets dupliqués que nous avons étudié, seuls [MUL75] et [HOL73] n'abordent pas les problèmes de défaillance, de blocage et de reprise du système. [ELL77], [THO76], [B19] formulent un certain nombre de propositions pour le redémarrage d'un moniteur.

Dans les procédures de reprise, nous distinguerons trois types de mécanismes :

a) le premier problème est global et concerne l'*opérationalité du système*. L'absence d'état global nécessite la définition d'un certain nombre de conditions complémentaires remplies par chaque moniteur ; lorsqu'elles sont remplies le système est opérationnel (par exemple l'anneau est constitué, ou le quorum est atteint cf. 4.3.2). Lorsqu'elles ne sont pas remplies, toute mise à jour est rendue impossible et une procédure de reprise globale est toujours disponible pour les moniteurs (reconstitution de l'anneau, quorum de nouveau réuni, etc.).

b) le deuxième problème se rapporte au *blocage d'un moniteur ayant temporairement un certain privilège* (avoir le témoin, être formateur lors d'un sondage, modifier des copies...). Pour éviter le blocage général du système, le privilège étant monopolisé, une procédure d'isolement du moniteur coupable et de récréation d'un nouveau privilège doit être définie.
Cette procédure, dans tous les cas, repose sur l'armement d'horloges de garde chez tous les moniteurs. Lorsque ces horloges sonnent, ils cherchent à isoler le coupable (reformer l'anneau, remettre à jour la table des moniteurs collaborateurs) et à recréer un nouveau privilège (remettre un témoin unique, pratiquer un sondage...).

c) le troisième problème a trait au redémarrage d'un moniteur et à la mise à jour de la copie locale à ce moniteur. Pour [B19], il s'agit pour le moniteur défaillant de retrouver le moniteur administrateur ; pour [LAM75], il s'agit de modifier la priorité des sites;dans l'état stable chaque moniteur actif a une copie à jour ; pour réactiver le moniteur défaillant J, on lui accorde une priorité incrémentée de N ($P_j = P_J + N$, N > nb total de processeurs) de telle sorte que l'action qu'il va mettre en oeuvre soit prioritaire sur les actions normales de mises à jour (on favorise la réinsertion).

En cas de gros fichiers, des transferts globaux étant trop longs et coûteux, la tenue de journaux de mise à jour sera nécessaire.

4.3. Motivations et Présentation de SYNDIC

Nous développons ici les concepts de bases et les mécanismes généraux de SYNDIC pour lequel on pourra se rapporter pour sa genèse et sa présentation complète aux références [A14], [B19], [WIL77].

4.3.1. Objectifs de cohérence :

Les objectifs qui ont été retenus dans SYNDIC sont les suivants :

- la cohérence maintenue est une cohérence lâche (cf. 4.2.1.),
- chaque moniteur peut fonctionner de façon autonome quand il est isolé des autres,

- l'ensemble des moniteurs garantissent à tout instant l'existence d'une copie originale, ou à défaut, le moyen de choisir un original parmi un ensemble de copies.

Comme le système est réparti, l'original du fichier n'est pas lié statiquement et définitivement à un site particulier, mais la notion d'original revêt un caractère dynamique. Dans cette approche, l'original caractérise la copie attachée au site qui a le plus récemment écrit dans le fichier.

4.3.2. Notion de système réparti opérationnel :

Nous nous intéressons à des systèmes répartis où les moniteurs peuvent communiquer entre eux à l'aide d'un réseau permettant le "broadcast" et le contrôle de flux sur les voies de communication. Pour assurer l'unicité de l'original, les mises à jour ne seront autorisées que lorsque le système sera opérationnel.

Définition : le système réparti sera dit opérationnel lorsqu'un nombre de moniteurs au moins égal à la moitié plus un du nombre total (quorum) seront interconnectés.

4.3.3. Etats de copie et statuts de moniteur :

a) toute copie est caractérisée par l'un des trois états suivants :

. à jour,
. instable,
. incertain.

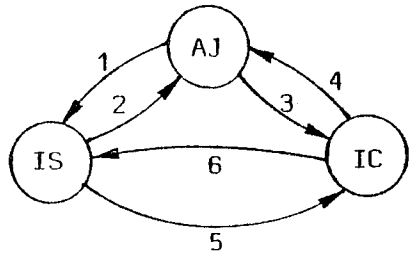
Une copie est *A JOUR* (AJ) si et seulement si cette copie a pris en compte toutes les modifications apportées au fichier et n'est pas en cours de modification ; il faut de plus que le moniteur de la copie soit en liaison avec le moniteur détenteur de la copie originale (voie de communication ouverte).

Une copie est dans un état *INSTABLE* (IS) si et seulement si une modification sur cette copie est en cours de réalisation. Dans ce cas aussi, la voie avec le moniteur détenteur de la copie originale est établie.

Une copie est dans un état *INCERTAIN* (IC) dans les autres cas ;

en particulier si la connexion entre le moniteur de cette copie et le moniteur détenteur de la copie originale est rompue ; cette rupture de connexion isole ce moniteur et l'empêche d'être tenu au courant des mises à jour susceptibles d'intervenir en son absence. Cet état permet d'envisager les pannes survenues à un site quelconque.

L'évolution de l'état d'une copie peut être schématisé par la figure suivante :



1 : AJ ⇢ IS demande de mise à jour de la copie (processus de mise à jour),

2 : IS → AJ fin du processus de mise à jour validant la copie,

3 : AJ → IC rupture de la voie avec le moniteur détenteur de la copie originale,

4 : IC → AJ rétablissement de la voie avec le moniteur détenteur de la copie originale et remise à niveau de la copie,

5 : IS → IC rupture de la voie ouverte avec le moniteur détenteur de la copie à jour (processus de mise à jour non terminé),

6 : IC → IS transition impossible ; une copie ne peut être instable que pendant un processus de validation de mise à jour ; le moniteur devant pour participer à ce processus avoir une copie à jour.

b) statuts de moniteur (MGF) :

Le MGF assume la gestion d'une copie sur le site d'implantation de celle-ci ; cette gestion est une fonction de l'état de la copie sur le site et de la connaissance de l'évolution du fichier. Suivant les droits dont dispose un moniteur sur une copie à l'instant t, nous allons associer une série de statuts à ce moniteur.

Statut administrateur A :

Ce statut est réservé au moniteur *détenteur de la copie originale*, qui désire, réalise ou termine une modification sur le fichier.

Ce statut, qui offre à son détenteur un privilège pour un temps limité, est transmissible, sous certaines conditions, à tout moniteur qui en fait la demande.

Ce statut ne peut être octroyé à plus d'un moniteur simultanément sous peine de violer le principe de cohérence ; en effet, si une telle situation se produisait, plusieurs moniteurs pourraient effectuer simultanément des modifications, ce qui conduirait à la disparité des copies originales.

Ce statut doit être obtenu par un moniteur avant d'effectuer toute modification du fichier.

Ainsi, ce statut peut être assimilé à une sémaphore permettant à son détenteur d'entrer dans la section critique que représente la modification du fichier.

Il y a, à tout moment, au plus un moniteur qui possède ce statut.

Ceci a pour conséquence qu'il faut en cas de panne, une procédure de reprise permettant de désigner un moniteur qui remplisse les conditions requises pour se voir attribuer le statut d'administrateur.

L'administrateur perd son privilège lorsqu'il lègue son privilège ou lorsqu'il n'est plus relié au quorum des moniteurs MGF.

A chaque nouvel administrateur est associé un numéro d'ordre de copie (valeur croissante).

Statut postulant P :

Ce statut transitoire caractérise le moniteur qui a reçu une demande d'accès en écriture au fichier et ne possède pas le statut d'administrateur. Ce statut subsiste tant que le passage du privilège n'est pas terminé entre l'administrateur en titre et le postulant.

Statut formateur F :

Le moniteur qui a le statut d'administrateur pouvant tomber en panne, un mécanisme doit permettre de désigner un nouvel administrateur. Ce mécanisme de désignation se matérialise par un sondage (cf. 4.3.5.) effectué par n'importe quel moniteur qui se déclare alors formateur (par analogie au rôle joué par un formateur de gouvernement).

Statut collaborateur C :

Le statut de collaborateur est attribué à tout moniteur qui ne dispose ni ne demande de privilège, mais dont la voie de communication avec l'administrateur est ouverte ; il accepte donc les mises à jour que lui communique ce dernier.

Statut reclus R :

Un moniteur est dit reclus lorsque sa voie de communication avec l'administrateur est rompue ; il est alors isolé et n'a plus connaissance des mises à jour. La réclusion le prive du droit de mise à jour. Grâce à un mécanisme de reprise, la réclusion n'est pas perpétuelle.

4.3.4. Le degré d'actualité :

Ce concept permet en effet de répondre à deux questions très importantes :

- comment un moniteur reclus peut-il déterminer les modifications qu'il n'a pas prises en compte ?
- comment retrouver les moniteurs qui ont leur copie à jour, lorsque l'administrateur est mort ?

Le degré d'actualité, DA, d'une copie est un nombre entier positif assigné à chaque copie, et incrémenté lors de chaque modification apportée au fichier. Ce nombre prend des valeurs sans cesse croissantes (cf. EO dans [ELL77]).

La comparaison des DA des différentes copies permet de distinguer celles qui ont intégré le plus grand nombre de modifications : pour ces copies, le DA est le plus élevé.

Le DA n'a pas de signification dans l'absolu : il prend tout son sens lorsque les DA sont comparés entre eux : c'est donc une notion relative.

La *comparaison* du DA des différentes copies permet de :

- reconnaître les copies qui n'ont pas été tenues au courant de toutes les modifications ;
- remettre ces copies à jour ;

- retrouver un moniteur dont la copie est à jour lorsque l'administrateur est inexistant.

Le *mécanisme de variation* du DA se résume en quatre points :
- tout administrateur effectuant une mise à jour du fichier incrémente d'une unité le DA attaché à sa copie ;
- les collaborateurs incrémentent également d'une unité le DA de leur copie lorsque cette modification leur est communiquée ;
- tout reclus qui se reconnecte avec l'administrateur reçoit les mises à jour nécessaires pour avoir une copie identique à celle de l'administrateur et son DA est aligné sur celui de l'administrateur ;
- à l'occasion d'un sondage, les DA des copies sont réajustés sur celui du nouvel administrateur.

Lorsqu'un processus de mise à jour n'est pas en cours, il y a un nombre suffisant (> quorum) de moniteurs ayant des copies avec un DA maximal (copies à jour).

Le fait que la majorité des moniteurs ait un DA identique permet de garantir la cohérence : en effet, lorsqu'une panne provoque la disparition de l'administrateur, un sondage ne peut avoir lieu que si le formateur est connecté à une majorité de moniteurs ; il trouve donc nécessairement une copie dont le DA est maximal à cet instant, et la cohérence est conservée.

Le processus normal de mise à jour se déroule de la façon suivante :
- l'administrateur modifie sa copie qui passe dans l'état instable,
- l'administrateur communique la mise à jour aux collaborateurs qui la prennent en compte, incrémentent leur degré d'actualité, mettent leurs copies en état instable et envoient un acquittement à l'administrateur,
- lorsque l'administrateur a reçu un quorum d'acquittements, la mise à jour est définitivement acquise, sa copie repasse à l'état à jour,
- il signale la fin du processus de mise à jour aux autres collaborateurs qui mettent leurs copies à l'état à jour.

Pendant la durée du processus de mise à jour, chaque moniteur garde une trace de la dernière version de copie à jour avec son DA associé. Pour tout collaborateur une panne qui se déroule pendant ce processus (système non opérationnel, voie rompue avec l'administrateur) provoque le retour à la dernière version à jour avant l'engagement d'une quelconque opération (sondage...).

Ainsi hors des processus de mise à jour soit toutes les copies sont à jour avec des DA identiques, soit un quorum de copies sont à jour avec des DA identiques et maximum.

Ainsi, lors d'un sondage, le système réparti étant opérationnel, la copie considérée comme originale aura obligatoirement un DA maximal.

On doit noter que le DA est un entier qui prend des valeurs croissantes : on fait DA = DA+1 à chaque modification de copie. Si le DA est représenté en mémoire par un mot de 32 bits (valeur maximum 2.10^9 environ), avec 20 moniteurs, 200 mises à jour par moniteur et par jour, ce n'est qu'au bout de plusieurs centaines d'années qu'il y aura débordement de capacité mémoire !...

4.3.5. Le sondage :

Le sondage est destiné à retrouver parmi un ensemble de moniteurs celui (un parmi ceux) qui possède (nt) une copie à jour. Le sondage est indispensable lorsque l'administrateur est mort. Il se déroule de la façon suivante :

- un moniteur reclus se déclare formateur (copie dans l'état incertain IC) : ce moniteur souhaite effectuer une mise à jour ou une lecture cohérente et veut mettre sa copie dans l'état à jour AJ (ou s'assurer qu'elle est dans cet état),

- le formateur lance son sondage en envoyant des messages aux autres moniteurs avec lesquels il a ouvert des voies de communication ; ces messages contiennent l'identification du formateur,

- Tous les moniteurs reclus renvoient au formateur leur DA si et seulement s'il ne l'ont pas déjà fait pour un autre formateur d'identification supérieure.
(On associe au sondage un temps maximum).

- le formateur dépouille les messages qu'il a reçu et parmi les DA maximum qu'il dénombre, il trouve un moniteur qui lui permet de mettre à jour sa copie (si nécessaire) et de prendre le statut d'administrateur (la communication aux autres moniteurs de ce changement de statut constitue la fin du sondage). Parmi les DA maximum, il peut se trouver que des moniteurs ayant acquis leur version à jour lors d'un processus de mise à jour non terminé normalement (copie restée instable) : le formateur choisit alors celui de numéro d'ordre maximum.

- le nouvel administrateur peut alors communiquer aux autres moniteurs reclus son DA et ses mises à jour : les reclus deviennent collaborateurs et leur copie passe à l'état AJ.

- l'administrateur peut lancer le processus de mise à jour.

4.3.6. Les primitives d'accès aux objets :

Il existe trois primitives :

- l'*écriture* E qui n'est possible que si le moniteur local acquière le statut d'administrateur et lance le processus de mise à jour,
- la *lecture directe* LD qui fournit le contenu du fichier tel qu'il se trouve au moment de l'accès, à jour ou pas,
- la *lecture cohérente* LC qui fournit le contenu à jour du fichier tel qu'il se trouve au moment de l'accès, à jour ou pas.

On peut établir le tableau des primitives possibles suivant les valeurs des deux variables critiques : le statut du moniteur et l'état de la copie.

état / statut	à jour : AJ	instable : IS	incertain : IC
A : Administrateur	LD, LC, E	LD	
C : Collaborateur	LD, LC	LD	
R : Reclus			LD
P : Postulant	LD, LC	LD	LD
F : Formateur			LD

Figure 4.2
Primitives d'accès possibles suivant l'état
de la copie et le statut du moniteur

4.3.7. La mise à jour :

Elle est possible pour un moniteur lorsqu'il a le statut d'administrateur ; ce statut s'acquière soit en devenant postulant (du statut d'administrateur possédé par un autre moniteur), soit en étant formateur avec un degré d'actualité DA maximum. Le deuxième cas est le sondage (cf. 4.3.5.). Le premier cas est traité de la façon suivante :

- le collaborateur (B) prend le statut postulant ; il demande le statut d'administrateur à celui qui le possède (A) ; (A) avertit les autres collaborateurs de l'opération en cours : pour ces collaborateurs et (A), les copies passent à l'état instable IS,

- (A) n'a plus de privilège et donne son privilège à (B),

- (B) communique son DA à tous les collaborateurs ; lorsqu'il a obtenu un quorum d'acquittement à son message, l'ensemble des copies repasse à l'état AJ sur avis de (B),

- (B) est alors administrateur capable de lancer un processus de mise à jour.

Le processus de mise à jour est décrit en 4.3.4. On doit noter la difficulté de changement de version de référence pour chaque moniteur qui nécessite la constitution d'une séquence indivisible : changer de version et incrémenter le degré d'actualité. L'existence d'une telle sequnece est une condition nécessaire de bon fonctionnement de l'algorithme.

4.3.8. La remise à niveau d'un moniteur reclus :

Cette remise à niveau ne peut avoir lieu qu'après rétablissement des voies de communication entre le reclus (B) et les collaborateurs, l'un de ces derniers se présentant au reclus comme étant un administrateur (A). (B) communique à (A) son DA permettant à (A) de fournir à (B) l'ensemble des mises à jour qui ne sont pas encore effectuées sur son site. (B) remet sa copie à jour et devient collaborateur en le signalant aux autres collaborateurs (le nombre de moniteursdu système réparti opérationnel est incrémenté de un).

4.3.9. Nombre de transmissions inter-moniteurs :

La validité de cet algorithme n'est pas un critère suffisant d'appréciation ; en effet, cet algorithme étant destiné à une implantation sur un réseau d'ordinateurs, il est indispensable que le nombre de transmissions inter-moniteurs et le délai nécessaire à l'exécution d'une mise à jour ne deviennent pas rédhibitoires.

C'est pourquoi, nous avons étudié le nombre de transmissions pour les différentes primitives d'accès aux objets suivant les différents cas de figures.

a) Lecture directe :

Il n'y a pas de transmissions inter-sites. L'utilisateur lit directement le contenu de la copie, quel que soit l'état de cette copie.

b) Lecture cohérente :

Si la copie est dans l'état AJ, la lecture est effectuée directement et il n'y a aucune transmission intersites nécessaire.

Si la copie est dans l'état IC, le moniteur R gérant la copie est reclus : le mécanisme utilisé pour remettre ce moniteur à niveau (collaborateur) détermine le nombre de transmissions inter-sites, soit :

- 1 transmission A→R : avertissement de connexion avec l'administrateur A ;
- 1 transmission R→A : envoi du DA du reclus à A ;
- 1 transmission A→R : envoi du fichier de mises à jour et du DA.

Au total, une lecture cohérente exige au minimum 3 transmissions inter-sites.

c) Mise à jour :

Le nombre de transmissions exigé par ce type d'accès au fichier dépend essentiellement des conditions initiales du statut du MGF et de l'état de la copie gérée par ce MGF.

Cas idéal :

La mise à jour est demandée par un utilisateur du site dont le moniteur a le statut d'administrateur. Le nombre de transmissions est minimal dans un tel cas et il s'établit comme suit :

. T_1 transmissions A→C, avec $T_1 \geq Q$ (quorum) ;
ces transmissions correspondent à l'envoi de la mise à jour à tous les collaborateurs ;
. T_2 transmissions C→A ; avec $T_2 \geq Q$ et $T_1 \geq T_2$;
envoi d'un acquittement à l'administrateur.

Au minimum, une mise à jour exige donc 2x(N/2) transmissions ; cependant, le cas idéal correspond à la connexion de tous les moniteurs à l'administrateur, auquel cas le réseau présente la configuration suivante :

1 administrateur A
N-1 collaborateurs C
0 reclus R
donc T1 = T2 = N-1

Nous pouvons donc conclure qu'une mise à jour idéale, aucune panne sur le réseau, comporte 2(N-1) transmissions inter-sites.

Il faut remarquer,par rapport au temps de réalisation,que toutes ces transmissions peuvent s'effectuer en parallèle.

Cas intermédiaire

Le moniteur qui doit effectuer la mise à jour n'a pas le statut d'administrateur, mais est connecté à ce dernier.

. 1 transmission P→A : demande du privilège ;
. T_1 transmissions A→C : communication aux collaborateurs d'un changement d'administrateur ;
. 1 transmission A→P : passation du privilège ;
. T_3 transmissions A→moniteurs connectés.

NB : T_1 et T_3 valent au maximum (N-2).

Le nombre de transmissions nécessitées par la phase de préparation vaut donc 2(N-1).

Dans ce cas, on a deux fois plus de transmissions inter-moniteurs que dans le cas précédent. Il paraît dès lors intéressant d'accorder à l'administrateur la possibilité d'effectuer toutes les mises à jour qui lui sont adressées par des utilisateurs locaux avant de transmettre ses pouvoirs à un autre moniteur.

Cette possibilité doit être limitée dans le temps pour éviter une attente infinie d'autres moniteurs.

Cas défavorable :

La désignation d'un nouvel administrateur nécessite les transmissions suivantes :

- S_1 transmissions F → R : envoi de bulletin de sondage ;
- S_2 transmissions R → F : renvoi du bulletin de sondage complété ;
- 1 transmission F → moniteur à jour : demande de la copie et du DA ;
- 1 transmission moniteur à jour → F : renvoi de la copie et du DA ;
  F := A ;
- S_3 transmissions A → R : envoi des mises à jour nécessaires et du DA aux moniteurs connectés ;
  R := C ;
- S_4 transmissions C → A : acquittements ;

où : $S_2 \leq S_1 \leq N-1$

$S_4 \leq S_3 \leq N-2$

Le nombre total de transmissions s'élève donc au maximum à 4(N-1).

Nous constatons donc que le nombre de transmissions inter-sites est très élevé ; la charge indue est importante, mais elle résulte d'une dégradation elle-même importante du reseau (perte de l'administrateur).

4.3.10. Conclusions :

L'algorithme SYNDIC que nous venons d'analyser répond, en conclusion, aux objectifs suivants :

- gestion décentralisée,
- maintien d'une cohérence lâche,
- tolérance aux pannes (résistance à la défaillance d'un ou plusieurs moniteurs, réinsertion d'un moniteur...),
- respect des règles de l'exclusion mutuelle même en cas de coupure du réseau.

Dans certains cas de figure, l'algorithme peut apparaître très lourd en transmissions réseau : comme pour d'autres algorithmes équivalents [THO76], [ELL77] c'est le prix à payer pour assurer un fonctionnement satisfaisant en mode dégradé.

Le respect des règles de l'exclusion mutuelle fait apparaître une contrainte forte : aucune mise à jour possible avec moins de la moitié des moniteurs présents sur le réseau. Cet algorithme peut changer d'objectifs en modifiant le mode de fixation du quorum :

- quorum = 2 : on accepte les coupures du réseau et l'existence de copies incohérentes : une procédure manuelle peut être envisagée pour rétablir la cohérence des copies ou une procédure automatique, une copie particulière prévalant en cas d'incohérence de différentes copies supposées à jour.

- quorum fixé en privilégiant un sous-ensemble de moniteurs, dont la présence sur le réseau est estimé indispensable pour faire une mise à jour. Ce choix est justifié lorsqu'il existe une hiérarchie entre les moniteurs correspondant, par exemple, à l'existence de centres principaux de traitement de fichiers et de centres secondaires de traitement de fichiers.

4.4. Domaines d'Application et Conclusions :

Dans les réseaux d'ordinateurs un certain nombre d'algorithmes ont déjà été développé. Leur validité théorique et pratique reste à démontrer en raison de leur complexité et en l'absence d'expérimentations véritables [THO76], [B19], [MUL73], [HOL73], [ELL77], [LAM75].

Dans les systèmes multi-caches [MAZ77], un certain nombre d'algorithmes ont été développés : dès qu'un système comporte plusieurs caches, plusieurs copies d'un même bloc peuvent se trouver dans différents caches. On doit alors informer chaque processeur des écritures effectuées par d'autres, de façon à ce qu'il n'utilise pas une copie périmée de bloc figurant dans son cache.

En dehors des systèmes multi-caches, un certain nombre d'applications sont intéressées par ces modes de gestion d'objets dupliqués :

- les systèmes de gestion de fichiers réseau construits à l'aide de catalogue réseau (type MADRE). Pour ces systèmes les mises à jour du catalogue sont peu nombreuses par rapport aux lectures et sa duplication ne peut qu'accroître la fiabilité du système et diminuer les coûts de transmission.

- le système INPOL [KAL74] de la police allemande qui travaillent avec des copies de fichiers entre deux ou plusieurs capitale régionales et entre une capitale régionale et le central. La structure du réseau impose un mécanisme centralisé pour assurer la synchronisation au cours des mises à jour,

- les tendances actuelles dans le développement des bases de données dans les réseaux indiquent qu'il y aura nécessité de dupliquer les structures des bases, ainsi que les modèles de coopération développés pour les nouveaux utilisateurs réseaux [B12]. Si la duplication des bases elles-mêmes ne semble pas être d'actualité, les bases documentaires (cf. 5.3) ayant des taux de mises à jour faibles par rapport aux interrogations pourront utilement être dupliqués : des études économiques sur les coûts de stockage et les coûts de transmissions, lorsque des réseaux comme Euronet et Transpac seront opérationnels, permettront de faire ces choix.

En conclusion, la duplication de l'information dans les réseaux a trois motivations principales :

a) augmenter la *disponibilité* ; si une information est confinée dans un site, elle est indisponible dès que ce site est en panne.

b) augmenter l'*efficacité* : en diminuant les transferts sur les réseaux et les temps de réponse.

c) favoriser le traitement en *parallèle* des lecteurs de fichiers.

L'efficacité et l'adéquation des différents algorithmes seront mieux établies lorsque les études sur la nature et la distribution des pannes dans les réseaux auront aboutis à des résultats significatifs, lorsque le coût des journaux de bord nécessaires sera mieux évalué dans les applications, lorsque le degré de parallélisme possible sera mieux apprécié (pourcentage de lectures cohérentes par rapport aux lectures directes en cohérence lâche).

5. LES OUTILS DE RÉALISATION D'APPLICATIONS RÉPARTIES ET TÉLÉ-INFORMATIQUES

On s'intéresse dans ce chapitre aux outils développés pour l'écriture de logiciel télé-informatique et réseau. Ces logiciels sont, en général, organisés selon des architectures voisines qui ont donné lieu à l'écriture d'un grand nombre de noyaux de systèmes ayant beaucoup de points communs : une proposition de noyau de système "type" est développée avec SYNCOP (paragraphe 5.1).

Dans le cadre du projet CYCLADES, une expérience importante a été acquise dans le domaine de la portabilité des programmes et de l'hétérogénéité des machines d'implantation. Un bilan est fait dans le paragraphe 5.2 de l'utilisation de FANNY et des possibilités offertes par les programmes portables pour cette gamme d'applications.

Après les systèmes d'exploitation "mono-ordinateur" qui ont nécessité une large gamme d'outils spécifiques (spécifications, systèmes, langages), il paraît utile de faire le point de ceux qui existent pour les systèmes répartis.

5.1 Les sous-systèmes pour la télé-informatique et les réseaux d'ordinateurs.

5.1.1 Introduction sur les sous-systèmes :

Les systèmes informatiques sont,de façon désormais classique, organisé en couche. Les couches sont disposées de façon concentrique avec des règles de définition fonctionnelle des couches ainsi que des interfaces entre couches.

Si l'on considère que chaque site, sur lequel vont être implantés des fonctions de télé-informatique et de réseaux d'ordinateurs, est doté d'un système d'exploitation, on part de la connaissance d'une certaine couche - service système - offrant une gamme de services accessibles à l'utilisateur. Nous dirons que ces systèmes d'exploitation sont des *systèmes hôtes* au sein desquels l'utilisateur s'introduit par l'intermédiaire d'une *tâche*.

Les tâches, dont il est question ici, ont une vocation fonctionnelle particulière. Au niveau architecture (structure interne), elles ont de nombreux points communs avec les systèmes d'exploitation eux-mêmes : en particulier elles sont multiprogrammées. Nous dirons qu'elles sont organisées autour de *sous-systèmes*.

La nécessité de superposer deux niveaux de système de contrôle sera justifiée en détail plus loin : retenons pour l'instant les insuffisances des systèmes hôtes, les inconvénients et les difficultés qu'il y a à les modifier.

En dehors du contexte télé-informatique, on peut citer plusieurs catégories de tâches multi-programmées, elles-mêmes pilotées par un système d'exploitation :

- travaux de simulation
- applications temps réel supportées par des systèmes pilotant d'autres applications (traitement par lots, temps partagé, autre application temps réel)
- systèmes (ou compléments de systèmes d'exploitation) dont la mise au point ne doit pas perturber l'exploitation courante.

A titre d'exemple, on peut retenir :

- *MULTICS :* un des objectifs de l'architecture du système est de mettre à la disposition de concepteurs de systèmes des outils leur permettant de développer des sous-systèmes spécifiques. Ces sous-systèmes sont réalisés au moyen de un ou plusieurs processus, chacun ayant une fonction particulière, des droits spécifiques... Ils constituent des ensembles de facilités s'adressant à une famille d'utilisateurs conçue pour faire un ensemble cohérent indépendant des autres sous-systèmes avec lesquels ils garderont la possibilité de coopérer et de dialoguer.

- *SIRIS 8 et SOCRATE :* le système de gestion de bases de données SOCRA est une tâche au sens du système d'exploitation Siris 8 de l'ordinateur IRIS 8(CII-HB. Cette tâche assure une double fonction : gérer des bases de données d'une part, offrir un service conversationnel à une gamme d'utilisateurs d'autre part. Elle réalise donc une fonction de multi-programmation entre les différents processus utilisateurs qui mettent en oeuvre des séquences de compilation de programme Socrate, d'éditeur de texte ou d'accès aux bases. Les spécifications du noyau comme les services disponibles pour les processus sont adaptés aux processus particuliers de l'application Socrate.

- *VM370 et CMS* : VM370 est un superviseur de contrôle des ressources d'un ordinateur IBM 370 qui simule des machines fictives sur une machine réelle. CMS est un sous-système de VM, utilisant les services de VM et mettant à disposition une gamme de services tels que gestion de fichiers, tests et exécutions de processeurs. La philosophie générale de ce sous-système est une indépendance totale vis à vis des autres sous-systèmes équivalents avec les garanties de sécurité y afférant.

[SOM75] précise la notion de sous-système, le sous-système étant un processus du système de contrôle. Les sous-systèmes peuvent être empilés en faisant apparaître une arborescence de processus sous-systèmes.

Pour les applications télé-informatiques et réseaux, cette arborescence est limitée, les fonctions mises en oeuvre pouvant être considérées comme proches de celles du système hôte. En particulier dans le cas de stations de transport gérant une connexion avec un noeud du réseau de commutation, le processus gestion de ligne fournira un service plus efficace s'il est à un niveau d'interruption le plus bas possible.

Dans le cas de certains ordinateurs spécialisés ou certaines machines virtuelles, les systèmes hôtes seront absents faisant apparaître les sous-systèmes comme des systèmes d'exploitation particuliers [BLA75], [SFE01], [TEC01].

Avant d'approfondir les sous-systèmes télé-informatiques et réseaux et pour mieux cerner leur rôle, on peut noter quelques propriétés générales :

a) ils gèrent des sous-ensembles de ressources des systèmes hôtes,
b) ils sont indépendants des autres sous-systèmes,
c) ils utilisent une partie des fonctions du système hôte,
d) ils communiquent avec les systèmes hôtes,
e) ils sont un intermédiaire obligatoire pour des processus du sous-système qui souhaiteraient faire appel à des services du système hôte.

5.1.2 Multiplicité des sous-systèmes télé-informatiques et réseaux

Le développement de la télé-informatique, comme celui des réseaux a donné naissance à une multitude de sous-systèmes,en général spécialisés, pour une gamme d'applications donnée.

Les méthodes d'accès en télé-communications sont bâties en général sur des "moniteurs de communications de données" où les processus mis en oeuvre ont des vocations spécialisées. A titre d'exemple, on peut citer [GEP76] un ensemble de produits-programmes commercialisés à l'heure actuelle pouvant être implantés sur des machines différentes, qui supportent certaines facilités de télé-communications et de support de terminaux, qui ont des procédures de recouvrement d'erreur et des interfaces langages évolués.

L'étude de la liste mentionnée en figure 5.1 permet de dégager quelques caractéristiques :

- grande dépendance vis à vis du système hôte,
- nombre de processus très variable (de 4 à 200),
- plusieurs facilités voisines dans la gestion des télé-communications, des fichiers, des terminaux (type télétype ou 3270 d'IBM),
- possibilités de reprises très variables (point de reprise, redémarrage à chaud) sauf pour le recouvrement automatique des erreurs de télé-communications,
- grande similitude dans les langages évolués supportés, mais les interfaces avec des systèmes de gestion de bases de données sont très variables (Total, PL/1, IMS ...).

sous-système télé-informatique	système hôte	protocoles supportés	nombre d'utilisateurs (processus)	langages évolués supportés
CICS	DOS DOS/VS OS/VS1,2	BSC Start Stop SDLC	?	Cobol PL/1
Betacomm	DOS DOS/VS	BSC Start Stop	4	Cobol PL/1
Datacom/DC	DOS DOS/VS OS,/VS1,/VS2	BSC Start Stop	30	Cobol PL/1 Fortran
Environ 1	DOS OS DOS/VS OS/VS1,VS2	BSC Start Stop	160	Cobol Fortran PL/1
Gbaswift	DOS DOS/VS	BSC Start Stop	55	Cobol Fortran PL/1
Intercomm	OS OS/VS	BSC Start Stop	150	Cobol Fortran PL/1
Minicomm	DOS DOS/VS	BSC Start Stop	100	Cobol PL/1
Task/master	DOS DOS/VS OS,/VS1,/VS2	BSC Start Stop	200	Cobol Fortran PL/1
Téléprocessing Interface System WESTI	DOS DOS/VS	BSC Start Stop	200	Cobol PL/1
TP 200	OS OS/VS1/VS2	BSC Start Stop	5	Cobol Fortran

Figure 5.1

Produits programmes : sous-systèmes télé-informatiques

D'une façon générale, ces outils sont spécifiques, ce qui explique que les programmeurs de développement d'applications réseaux ont créé leurs propres sous-systèmes réseaux ; la liste qui suit fournit quelques exemples typiques pour les réseaux généraux type ARPA, CYCLADES, EIN.

ordinateur hôte	système hôte	sous-système	réseau d'ordinateurs
IBM 360	OS	HASP	ARPANET
IBM 360	OS	ASP	SOC
IBM 360	OS	TELCOM [PAP74] CRIC, SYNCOP	CYCLADES
machine virtuelle CP 67 / VM 370	machine nue	TELCOM, SYNCOP	CYCLADES, EIN
CII-HB IRIS 45 / 50 / 60	SIRIS 2 / 3	CRIC - SGT[CII01]	CYCLADES
CII-HB IRIS 45 / 50 / 60	SIRIS 2 / 3	CRIC, SYNCOP	CRIC
CII-HB 10070 / IRIS 80	SIRIS 7 / 8	CRIC, SYNCOP	CRIC, CYCLADES, EIN
SIEMENS 4004	DOS 16 / BS 1000	CRIC	CYCLADES
SEMS Mitra 15	MTRD, machine nue	SYNCOP	CYCLADES
SFENA ordo 300/400	machine nue	SYNCOP	CYCLADES

Figure 5.2

Sous-systèmes réseau

5.1.3 Caractéristiques fondamentales des sous-systèmes télé-informatiques et réseaux.

Les différentes caractéristiques que nous retenons ici permettent de mieux préciser l'originalité de ces sous-systèmes, en particulier leur confrontation avec celles des systèmes hôtes permet de justifier la hiérarchie système hôte - sous système : la prise en compte de ces caractéristiques faisant apparaître tantôt une inéfficacité des systèmes hôtes, tantôt une insuffisance.

a) les *contextes de tâche* des processus télé-informatiques sont à la fois peu volumineux et spécifiques (quelques mots, quelques dizaines au maximum) ; la recherche d'informations se fait par indirection (note : dans les systèmes hôtes, les contextes de tâche, sauvegardés à chaque commutation, sont volumineux et généraux).

b) les *blocs de mémoire* manipulés par les processus sont en grand nombre, de taille très variable (bloc contexte, buffer paquet ou message...); il y a multi appartenance dans le temps des tampons entre des processus différents et nécessité de contrôler les échanges d'informations interprocessus.

c) une des caractéristiques importantes des actions mettant en oeuvre des échanges de lettres sur un réseau est la nécessité d'associer à ces échanges des *réveils* avec la possibilité de dérouler des séquences ad-hoc lorsqu'ils sonnent. Une réalisation par création de tâche à chaque sonnerie d'horloge est trop lourde, trop coûteuse en temps machine et déformatrice du temps réel. Un outil spécifique sera indispensable.

d) la réalisation du logiciel de base pour la télé-informatique et les réseaux reste du domaine des spécialistes : les *contrôles* d'exécution, d'accès mémoire, d'immobilisation de l'unité centrale assurés, par exemple, par un système hôte pour ses tâches banales doivent être éliminés pour des logiciels mis au point.

e) les processus se *synchronisent* très fréquemment et s'échangent des tampons : la commutation de processus doit être efficace, la gestion de tampons souples et bien intégrés à la gestion de la synchronisation.

Remarque : Nous venons d'introduire le terme de processus ; cette entité est appelée à être dans le sous-système ce que la tâche est dans le système hôte.

f) le *mode d'élection* des processus-candidats à l'activation (prise de contrôle de l'unité centrale) doit être souple : les critères sont variables suivant les applications ; avec les priorités ce sont des données réservées et protégées.

g) la nature des *événements* est variable suivant les processus : la programmation doit pouvoir en définir de nouveaux types suivant son application (E/S périphériques, E/S réseau, réveil, files...).

h) dans les applications télé-informatiques, les processus pratiquent des échanges fréquents et importants d'informations : la gestion de mémoire commune et de mémoire transférable d'un processus à un autre est fondamentale.

i) l'existence d'un grand nombre de processus (en dehors des processus ayant un caractère permanent, il apparaît utile de définir un processus par porte ouverte sur le réseau ou abonné actif ou voie ouverte).

5.1.4 Définition d'un sous-système normalisé télé-informatique et réseau : SYNCOP [C17]

Les différents sous-systèmes mentionnés au chapitre 5.1.2 ont certaines des caractéristiques typiques qu'une étude exhaustive a pu dégager. D'une façon générale, ils sont un outil intégré dans une (ou deux) applications sans que l'on puisse en dégager une vue globale et permettre son utilisation dans un contexte différent, pour une autre application réalisée sur un autre ordinateur.

Pour faire progresser l'ingénérie des applications télé-informatiques et réseau, il importe d'offrir un outil application indépendante, machine indépendante, programmeur indépendant, réseau indépendant.
C'est l'idée de base de SYNCOP (Sous-Système Normalisé de Commutation de processus télé-informatique et réseau). Nous en reprenons l'idée des principes de base en renvoyant le lecteur intéressé aux brochures de définition et de réalisation [C17][C10][DAN75].

Ses principaux objectifs sont :

- une adaptabilité maximale aux applications télé-informatiques,
- un interface utilisateur standardisé et propre,
- un ensemble de réalisations sur plusieurs machines,
- des versions extensibles suivant les besoins.

a) Définition des objets SYNCOP :

Objet défini	Pour servir de variable à :
le processus	la gestion multi-tâches
la file	la communication inter-processus
le segment	la gestion de la mémoire utilisateur
la ressource	la gestion des ressources de l'utilisateur et du sous système (exclusion manuelle)
l'interruption	la gestion des entrées/sorties, des lignes de transmission, ...
le temps	la gestion des réveils
l'événement	la synchronisation inter-processus

b) Principes de base de SYNCOP :

- réalise la commutation d'un grand nombre de processus, d'où une optimisation de la représentation de l'objet processus (cette représentation est appelée contexte de processus) ;
- prévoit une forte interaction entre les processus, d'où une optimisation de la représentation et de la manipulation des files ;
- tient compte du fait fondamental suivant : le faible taux d'occupation de l'unité centrale par chaque processus activé séparément ; le processus dont il est question ici est l'archétype du processus téléinformatique ;
- gère des processus temps réel : actions déclenchées à l'aide des sonneries d'horloge (optimisation des blocs de réveil et des mécanismes de leur gestion, minimisation de la charge indue produite par le traitement des horloges, ...) ;

- a la notion d'attente de P événements sur une liste de N, la mémorisation de l'attente de Q processus sur un événement, la synchronisation sur acquisition de ressources ;
- a une méthode d'accès normalisée aux différents périphériques ;
- sait rendre optionnel la protection inter-processus rendue inutile pour les applications mises au point.

5.1.5 Les services de base de SYNCOP.

Les services de base proposés sont les suivants :

- la gestion des processus,
- la synchronisation entre les processus,
- la gestion de la mémoire,
- la gestion du temps,
- la gestion des files,
- la gestion des ressources,
- la gestion des entrées/sorties, des lignes de transmission, (cas où les systèmes hôtes sont absents),
- l'aide à la mise au point.

a) Introduction :

Globalement, le sous-système SYNCOP peut être considéré comme un commutateur de processus créé par un processus d'initialisation. Ce processus préexistant et s'intercalant entre le système hôte et le sous-système, a pour fonction de créer l'environnement nécessaire à l'existence de SYNCOP. Ce processus disparait sans laisser de trace (récupération du code, de l'espace mémoire), après avoir créé (au sens de SYNCOP) les processus commutateur et initiateur d'application (cf. figure 5.3).

Le commutateur de processus de SYNCOP, gérant le contrôle de l'unité centrale, est en effet lui-même considéré comme un processus,. De même que tout autre processus, il possède, vu de l'utilisateur, deux états fondamentaux :

- actif,
- non actif (cf. figure 5.4).

Néanmoins, il y a dualité entre les états du commutateur et ceux des autres processus dits "banals". A un instant donné, un seul processus est actif, tous les autres étant inactifs. Les transitions d'état se déroulent

toujours de la manière suivante :

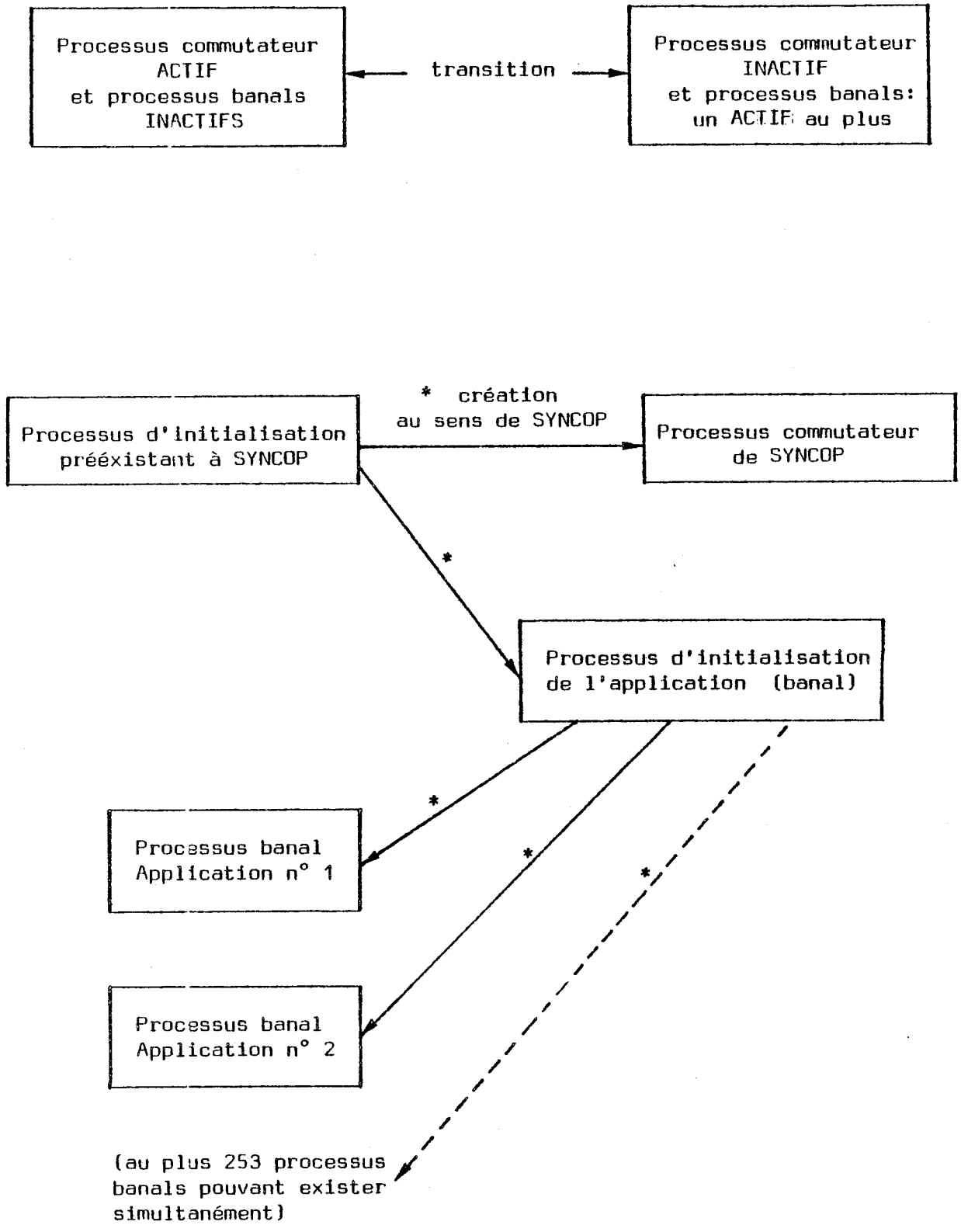
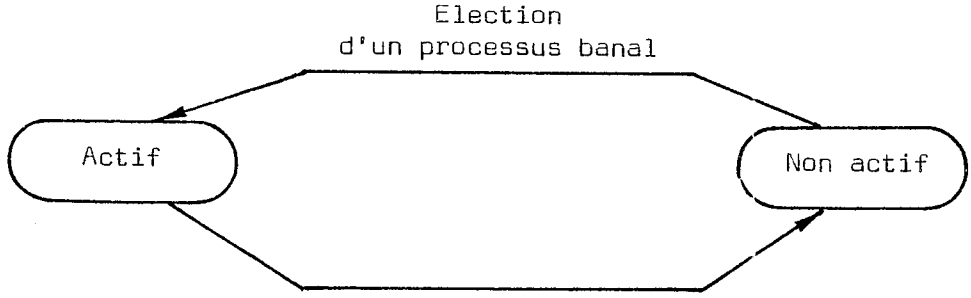


Figure 5.3

Initialisation et types de processus dans SYNCOP



. soit une action A_i d'un processus banal

. soit une action temps réel (sonnerie d'horloge)

A_i : A_i = se mettre en attente simple ou multiple

A_i = attente sur ressource occupée

A_i = se suicider sur erreur irrécupérable

A_i = mort sur fin normale de processus

Figure 5.4

Etats fonctionnels d'un processus SYNCOP

b) La gestion des processus :

Les applications télé-informatiques se caractérisent par un ensemble de propriétés :

- nombreux algorithmes de fonctionnement à exécuter en parallèle dans le temps et de durée indéterminée,
- algorithmes d'importances différentes pour la bonne marche de l'application,
- faible temps de contrôle de l'unité centrale pour chaque algorithme à tour de rôle (ces algorithmes se caractérisent par de fréquentes entrées/sorties périphériques et surtout réseau et par des traitements de durée très limitée).

On appellera *programme* la séquence d'instructions permettant la réalisation de l'algorithme et *processus* l'entité dynamique permettant l'exécution du programme dans le cadre de SYNCOP. Ce processus se caractérise par le programme à exécuter, par les données associées et par un *contexte* qui comprend :

- des informations d'identification du processus par le commutateur,

- l'état du programme,
- la sauvegarde des registres (dont un contient l'adresse du contexte),
- une zone de travail organisée en pile.

La troisième propriété précédemment exposée permet de répondre à l'une des questions fondamentales : *sous-système avec ou sans préemption ?* L'interêt d'un système à preemption est de garantir, quel que soit l'etat du système, l'exécution privilegiée de certains types d'actions (par exemple horloge de garde). Les différents processus associés aux applications télé-informatiques "passent leur temps à se mettre en attente" ; on peut donc garantir que les commutations de processus se suivent à des durees très brèves. Un système à préemption, très coûteux à gérer, devient donc inutile dès que l'on peut faire executer les actions privilégiées au moment de la commutation.

La première propriété conduisant à envisager de gérer de nombreux processus, il est nécessaire de minimiser au maximum la taille du contexte de processus et d'assurer un temps de commutation le plus faible possible. De même, il est utile d'assurer une gestion dynamique des processus en offrant la possibilité de créer, de tuer et de se suicider.

On définira une relation de filiation entre un *processus créateur* et le *processus créé*. Cette relation donne pouvoir au père de tuer ses fils sans pour cela introduire d'arbre de processus dont le comportement serait déterminé par une racine.

L'existence de processus réalisant les algorithmes d'importance inégale, conduit à doter chaque processus d'une *priorité d'activation*. Le commutateur élit (donne le contrôle de l'unité centrale),parmi les processus activables,celui de plus forte priorité en attente depuis le plus longtemps.

c) Le contrôle de la synchronisation

Le type des applications réseau dont on envisage la réalisation à l'aide de SYNCOP se caractérise par de nombreuses relations entre les processus et le système hôte et entre les processus eux-mêmes ; ces relations apparaissent sous deux aspects différents :

. *relations d'égalité* : les processus s'échangent des informations sans que cet échange agisse nécessairement sur l'algorithme d'élection des

processus par le commutateur. Les informations échangées changent de propriétaires (cf. gestion de la mémoire).

• *relations de dépendance :* les processus interagissent et se rendent dépendants les uns des autres. Dans ce cas, les informations utilisées agissent directement sur l'algorithme d'élection du commutateur et peuvent avoir une action sur l'état des processus impliqués. La mono-appartenance de ces informations est une propriété remarquable.

Nous désignons l'ensemble de ces relations de ce deuxième type sous le terme de *synchronisation.* Elle consiste, dans notre cas, à définir des mécanismes permettant à un processus :

- de se bloquer en attente d'un ou plusieurs signaux émanant du système hôte ou d'autres processus, avant de poursuivre son exécution,
- d'indiquer que tel signal attendu par tel autre processus est réalisé.

Nous utilisons un mécanisme d'action indirecte, c'est-à-dire que la synchronisation s'effectue par l'intermédiaire d'un objet connu des deux processus à synchroniser. Nous attachons le terme *d'événement* à cet objet intermédiaire.

Nous distinguons deux types d'evénements suivant la provenance du signal associé :

- les événements système (dits externes) sont ceux que le système hôte délivre à l'occasion d'appels de services système (entrees/sorties des périphériques, boîtes à lettres,...) ;
- les événements SYNCOP (dits internes) sont ceux que le sous-système met à la disposition des processus pour se synchroniser entre eux.

Nous traitons à part les "sémaphores" que SYNCOP associe à la gestion des ressources.

L'événement que nous considérons est une définition étendue de la notion classique d'événement. En effet, le système SYNCOP offre à l'utilisateur la notion "d'attente multiple", c'est-à-dire la possibilité de se bloquer en attendant la réalisation de P événements sur une liste de N données ($1 \leq P \leq N$). De plus, chaque évenement est à "mémorisation multiple",

le processus pouvant attendre la réalisation de plusieurs occurences du même événement avant son déblocage.

A cette notion générale est associée une restriction dans le cadre de SYNCOP : un événement est nécessairement *propriété d'un processus* qui seul a droit de se mettre en attente dessus, mais peut être réalisé par n'importe quel autre processus. L'arrivée d'un événement ne permet donc de débloquer que le processus en attente, propriétaire de cet evénement. Il faut donc remarquer que le sous-système SYNCOP ne pourra être réalisé que sur des systèmes hôtes possédant eux-mêmes la notion *"d'attente multiple"*.

La généralisation de la notion d'événement permet de résoudre la plupart des problèmes attachés à la synchronisation des applications envisagées. Ce mécanisme est néanmoins insuffisant pour résoudre les problèmes d'exclusion mutuelle ou d'accès à des ressources partagées. On verra dans la gestion des ressources ce qui a été prévu pour ce type de problème.

d) La gestion de la mémoire centrale :

Trois propriétés essentielles ont été retenues :

1. il existe de nombreuses demandes d'allocation de mémoire de taille variable : petits blocs de contexte (de 4 à 16 mots), tampons d'entrées/sorties (de 64 à 1024 mots) ;
2. de nombreux processus à durée de vie très variable peuvent se terminer de manière anormale ;
3. des processus de priorités différentes dont ceux de priorité la plus élevée ne peuvent être bloqués sur demande de mémoire, sous peine interblocage.

La *première propriété* conduit à laisser les processus effectuer eux-mêmes les demandes et les libérations de mémoire dans une zone commune à tous les processus. En effet, l'algorithme de fonctionnement de ce type de processus ne permet pas de déterminer de manière précise la taille de la zone nécessaire à son exécution. Néanmoins, et pour éviter une trop grande fragmentation de la zone commune, nous avons défini deux stratégies d'allocation suivant la taille des blocs demandés.

La *deuxième propriété* nous a conduits à mettre l'accent sur la réallocation dynamique de la mémoire et la possibilité de libérer de façon simple toute la mémoire occupée par un processus en cas de mort, quelle qu'en soit la cause.

D'où l'existence d'un mécanisme qui conserve processus par processus le repérage des zones allouées. De plus, nous avons associé à l'algorithme d'allocation des petits blocs, la priorité pour répondre à la *troisième propriété*.

Deux *stratégies d'allocation* sont prévues suivant la taille des blocs de mémoire demancés :

- une allocation par zone, dans une mémoire entièrement gérée par le système SYNCOP pour les blocs inférieurs à 64 mots ;
- une allocation "système", demandée au système hôte pour les blocs de plus de 64 mots.

Le sous-système SYNCOP assure dans les deux cas un parfait contrôle d'appartenance. Cette notion de propriété qui associe un bloc au processus demandeur, permet à la fois un contrôle à la libération (la demande et la libération doivent être faites par le même processus) et une vérification à la mort du processus (libération éventuelle de mémoire non libérée).

Après deux années d'utilisation, il semble que ces mécanismes d'allocation mémoire soient très satisfaisants dans le cadre de la mise au point des applications. Cependant, les nombreux contrôles relatifs à la propriété et à la structure des zones d'allocation introduisent une surcharge importante. Celle-ci ne se justifie pas pour des applications entièrement mises au point. Ce fait conduit à disposer de deux versions différentes de la gestion mémoire selon le type ou l'état des applications :

- une avec contrôle et vérification,
- l'autre sans cette surcharge de l'algorithme.

e) La gestion des files :

Les primitives de synchronisation définissent un moyen de communication inter-processus réduit à la forme élémentaire d'un signal. Ce moyen ne suffit pas à tous les besoins. Le mécanisme des files constitue dans SYNCOP le moyen privilégié d'échanges d'informations entre les processus.

Ce dispositif permet l'échange de messages ou blocs d'informations entre plusieurs processus. Un message sera constitué par un bloc de mémoire de longueur variable. Une file est composée d'un descripteur et d'un ensemble de messages. L'arrivée d'un message dans la file impliquant le déblocage du destinataire eventuellement en attente, il est nécessaire d'associer à la file un mécanisme de synchronisation. Aussi le descripteur de la file sera-t-il constitué par :

- un événement interne permettant la synchronisation,
- un repérage permettant de retrouver les messages,
- des informations diverses facilitant le contrôle de la manipulation de la file (nom du propriétaire, nombre d'élements présents,...).

La file fonctionne globalement suivant le principe "premier entré, premier sorti" (voir cas particulier de la recherche sur critère).

Pour éviter les conflits possibles, on associe à la file une notion de *propriété*. Une file appartient au processus qui a demandé sa création ; lui seul est autorisé à *consommer* dans la file, tous les autres processus pouvant produire dedans. Pour résoudre les problèmes de gestion mémoire, un message produit dans une file deviendra automatiquement propriété du créateur de la file qui aura à charge sa libération après utilisation. Cette méthode permet de désynchroniser totalement producteur et consommateur, le producteur n'ayant plus à tenir compte des blocs enfilés.

L'évènement associé à la file permet au consommateur de se mettre en attente sur la production de un ou plusieurs messages.

Dans certains cas de mise en oeuvre de SYNCOP, il a été réalisé un mécanisme de gestion de files légèrement différent. Ce mécanisme est fondé sur des files de taille fixe (c'est-à-dire nombre maximum d'éléments présents fixé à l'avance), les éléments ayant soit une longueur fixe soit des longueurs variables (propriétés mutuellement exclusives).

Le descripteur de ce genre de file contient en plus un événement associé à chaque producteur permettant l'attente de production d'un processus sur file pleine. Il est sous-entendu que le nombre de processus producteurs dans une file de ce type est connu à la création de la file. Dans ce cas, le consommateur peut être protégé contre une saturation par les producteurs. De plus, la mémoire n'est pas encombrée par des messages que le consommateur ne peut traiter.

Un risque de mauvaise utilisation des files se présente lorsque le propriétaire de la file disparaît sans avertir les producteurs. Pour minimiser ce risque, l'utilisation de la file se fera non pas par communication de l'adresse du descripteur, mais par une variable intermédiaire (utilisation par adresse indirecte) dite *ancre* de la file. L'ancre de toute file est une variable n'appartenant ni au processus producteur , ni aux processus consommateurs. Cette variable est propriété du sous-système SYNCOP. L'ancre de la file étant connue du consommateur, à sa mort une des actions de la primitive chargée de le tuer est de mettre l'ancre à l'état "file morte" avant de libérer le descripteur de file.

Le mécanisme des files est un dispositif simple et souple permettant l'échange d'informations entre processus. Il offre aussi une possibilité indirecte de se synchroniser entre processus (cf. files ayant un nombre de messages fixe).

f) La gestion des ressources :

Une ressource est un objet partageable utilisable par un ensemble de processus, mais ne pouvant être acquise que par un nombre limité N de processus à la fois ($N \geq 1$) ; Les processus occupant la ressource sont chargés de sa libération. Pour son fonctionnement, SYNCOP gère ses propres ressources de manière interne (exemple la mémoire centrale). Il offre de plus aux programmeurs le moyen de déclarer une ressource sans restriction sur son type.

La gestion des ressources se ramène au problème de réalisation de l'exclusion mutuelle. Un processus qui désire une ressource doit demander son obtention :

- si la ressource est libre, elle lui est affectée,
- sinon, le sous-système lui fournit une possibilité d'attendre sa libération par le processus occupant la ressource. Lorsque le processus en attente est débloqué, s'il veut toujours utiliser la ressource, il doit réitérer sa demande.

Remarque : La notion d'événement telle qu'elle a été définie (§ c) ne permet pas aux processus de mémoriser l'attente d'un événement ; ici, "la ressource est libre". En associant un événement à une ressource, la fonction de mémorisation sur attente de ressource est réalisable comme suit :

Processus :

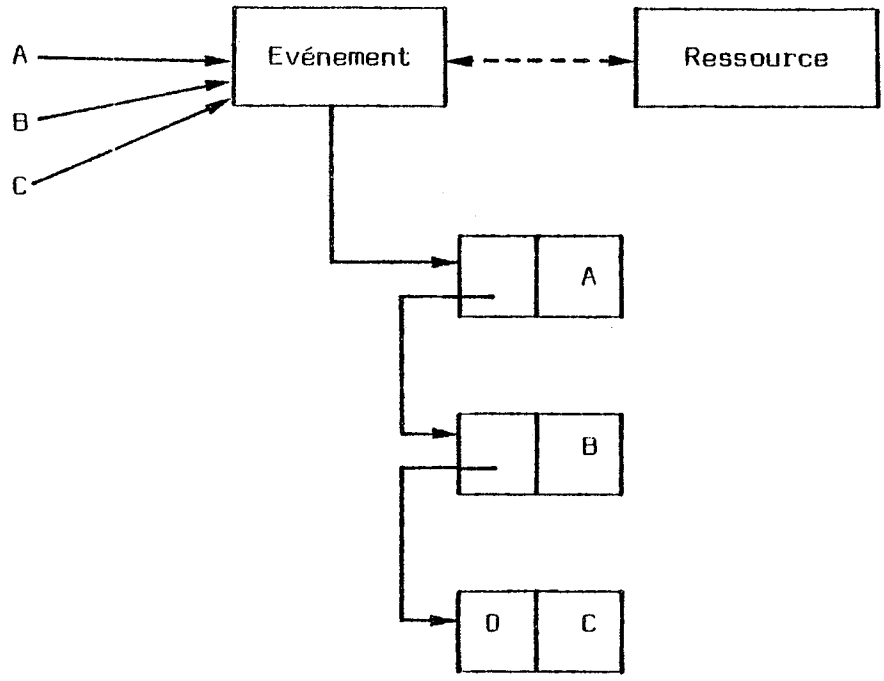


Figure 5.5

Mémorisation sur attente de ressource (1)

Cet événement-ci n'est pas l'événement SYNCOP. On dissocie donc la mémorisation de l'attente (point de vue de la ressource) de l'attente elle-même (point de vue du processus) pour avoir des notions d'événements cohérents et permettre en toutes circonstances au processus de choisir la liste des événements sur laquelle il souhaite se mettre en attente du contrôle de l'unité centrale.

Vue du processus, l'utilisation se passera en deux temps :

. demande de ressource = mémorisation de l'attente,

. attente de P événements parmi N = dans la liste des N événements il y en a obligatoirement un associé à l'opération précédente (il y a au plus une demande de ressource avant l'attente).

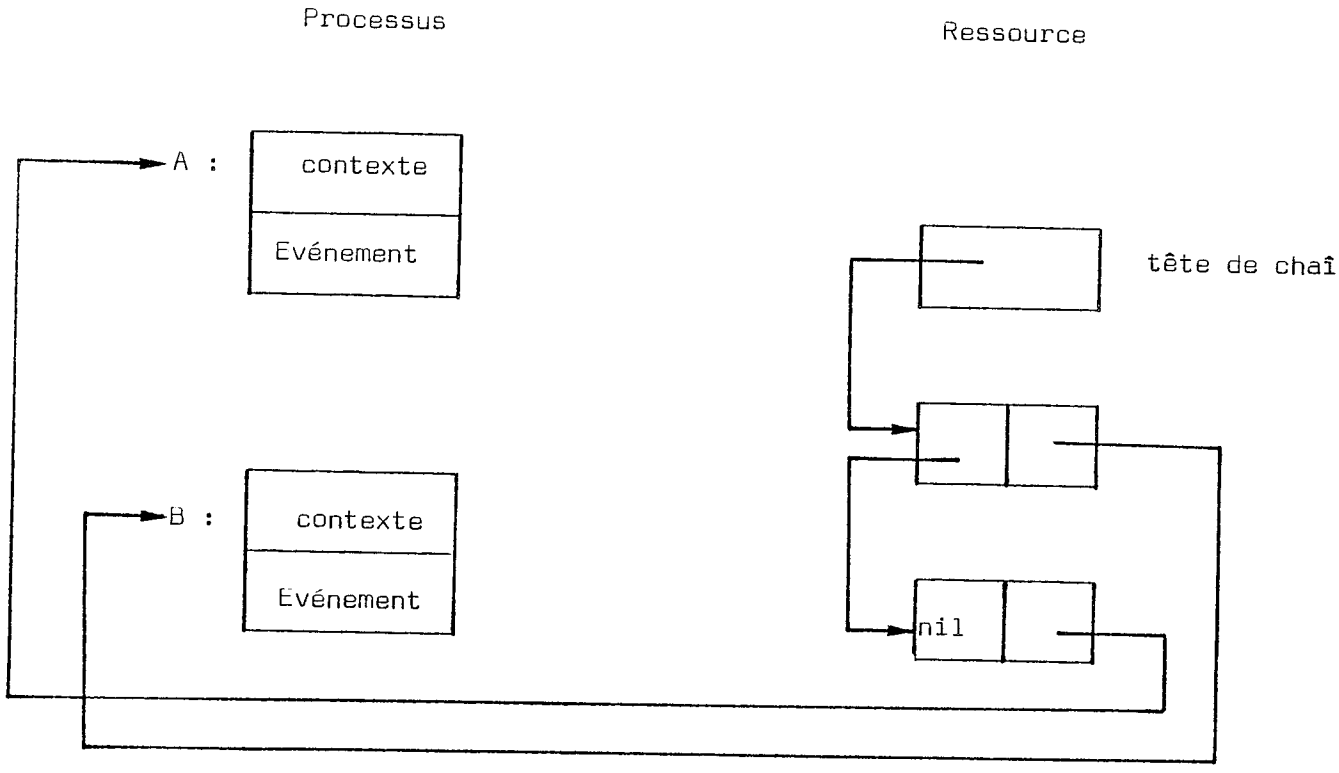


Figure 5.6
Mémorisation sur attente de ressource (2)

Vue de la ressource à chaque demande, on enfile un élément dans la chaîne associée, à chaque libération de ressource on défile tout ou partie de la chaîne en réalisant pour chaque processus en attente de la ressource l'événement associé à son contexte. Quand un processus est tué ou se meurt, les ressources qu'il occupe éventuellement sont libérées ; s'il était en attente d'une ressource, cette attente sera supprimée par la primitive traitant la mort du processus.

g) La gestion du temps :

Le temps est un élément fondamental des applications téléinformatiques que l'on envisage de gérer avec le système SYNCOP. Son utilisation se présente dans les processus sous deux formes différentes :

- attente d'une durée déterminée avant de reprendre l'exécution,
- exécution d'une séquence *en parallèle* avec le programme principal au bout d'un temps déterminé.

Ceci nous conduit à définir deux types de réveils :

1) le réveil *simple* avec attente : permet de se mettre en attente (WAIT SYNCOP) pour une durée déterminée ;

2) le réveil avec séquence associée : lorsque le réveil sonne, quel que soit l'état des processus (actifs ou en attente), la séquence associée au réveil est déroulée.

g.1) Réveil avec attente :

Ce type de réveil est associé à un événement interne permettant de mettre un processus en attente pour une durée déterminée. Lorsque la durée prévue s'est écoulée, l'événement est posté et le processus est rendu activable. Ce type de réveil peut être placé dans une liste pour attente multiple.

g.2) Réveils avec séquence associée :

Ils permettent de mettre en correspondance un réveil et une séquence particulière définie par le programmeur. Cette facilité est destinée à alléger la programmation : l'association entre le réveil et l'action à effectuer étant gérée par le commutateur.

g.3) Désarmement d'un réveil :

A un type de réveil donné, on associé un bloc de réveil implanté dans une pile particulière de traitement : le réveil qui sonne est toujours associé à la tête de pile. Par contre, si l'utilisateur veut désarmer un réveil précédemment lancé, SYNCOP enlève sur critère le bloc réveil associé dans la pile.

Dans les applications télé-informatiques, l'utilisation essentielle des réveils est faite par les processus pour se constituer des horloges de garde (time-out). Ils veulent se prémunir contre les pannes ou les défaillances momentanées d'un dispositif quelconque. Comme les dispositifs utilisés sont d'une bonne fiabilité, les processus qui ont lancé des horloges de garde les désarment fréquemment dès qu'ils sont avertis du bon fonctionnement des opérations lancées auparavant.

h) Le traitement des interruptions :

Dans une implantation sur machine nue, SYNCOP doit traiter les interruptions :

. intérruptions d'entrées/sorties,

. interruptions d'horloge,

. interruptions de programmes.

Dans une implémentation sous-système hôte, l'arrivée des événements liés à ces interruptions doit être connue de SYNCOP. Ceci peut être obtenu par deux moyens :

. un processus utilise un événement système qui est réalisé par le système hôte. C'est le cas, par exemple, pour les interruptions d'entrée/sortie : l'opération est lancée directement par le processus qui fait appel aux méthodes d'accès du système hôte et se met en attente sur un événement système associé. Le rôle de SYNCOP est réduit à la réactivation du processus quand l'événement est arrivé.

. SYNCOP demande un déroulement en cas d'interruption et traite lui-même l'interruption. C'est ainsi que sont traitées les interruptions programmes.

i) L'aide à la mise au point :

Note : nous ne nous intéressons pas au problème d'écriture des processus, mais uniquement à la mise au point en cours d'exécution.

Un des problèmes importants souvent négligé dans les sous-systèmes est l'aide apportée au programmeur pour la mise au point de ses applications Nous avons volontairement mis l'accent sur ce point pour fournir un ensemble des services le plus complet possible. Deux types d'aide à la mise au point nous ont paru particulièrement intéressants :

. informer l'utilisateur sur le déroulement normal de son application,

. fournir le maximum de renseignements lors d'un avortement de son application.

De plus, une application étant généralement composée d'un ensemble de processus, nous avons associé l'aide à la mise au point au processus lui-même. En outre, les options de mise au point sont définies à la création des processus permettant, par paramétrage, de les supprimer lorsque l'application est parfaitement mise au point.

5.1.6 SYNCOP et les Réalisations Télé-informatiques et réseaux

a) Applications en cours ou en projet :

sur la base des spécifications décrites aux chapitres précédents, plusieurs réalisations ont déjà été menées à bien. Les caractéristiques des systèmes hôtes sont variables, mais tous sont nantis d'un WAIT simple et d'une horloge en tant que primitives utilisateurs ; l'existence de ces deux services est la seule hypothèse à faire sur les système hôtes.

ordinateur	système hôte	machine nue
IBM 360/370	machine virtuelle CP 67 / VM 370	
IBM 360/370	O.S.	oui
CII 10070/IRIS 80	SIRIS 7/8	non
CII MITRA 15	MTRD	oui
SFENA ordo 300/400	non	oui

Figure 5.7
Réalisations de SYNCOP

Sur machine nue ou machine virtuelle CP/VM, le sous-système est un système d'exploitation particulier, propre gérant de ses interruptions : ce cas se produit lorsqu'il y a spécialisation de l'ordinateur ou de la machine virtuelle pour une fonction télé-informatique qui peut être :

- ordinateur frontal,
- support terminaux lourds,
- serveur réseau.

Plusieurs applications ont été développées autour de SYNCOP : elles constituent une validation de cette idée de sous-système normalisé télé-informatique :

- station de transport ST2 CYCLADES (SIRIS 7/8, OS 360, CP 67), [DAN76],

- station de transport ST3 : protocoles ST2 Cyclades, interface X25 Transpac,
- concentrateursde terminaux programmés (CP 67, SIRIS 7/8) [ANS76],
- concentrateur multi-connexions (SIRIS 8) [RIC76],
- méthode d'accès fichier réseau MADRE (SIRIS 7/8) [B8],
- interpréteur général reseau IGOR (SIRIS 7/8, OS 360) [DAN77],
- utilitaire réseau (transferts de fichiers,...) [GIE76],
- serveur réseau (batch OS 360 et SIRIS 8, temps partagé CP 67 et SIRIS 8) [AND76], [FOU76],[QUI76],
- applications fichiers répartis (FRERES) [BOS77],
- applications de simulation réseau (gestion de fichiers à copies multiples) [WIL77].

Sur la base de ces expérimentations, plusieurs remarques méritent d'être présentées.

b) Pédagogies, maquette et produits :

Même spécialisés dans leur définition, les choix télé-informatiques qui ont été faits ne sont pas suffisants pour prétendre s'adapter au mieux avec les applications concrètes. Dans toute sa généralité, SYNCOP est ouvert à plusieurs types d'utilisation :

- *produits télé-informatiques*, commercialisés par des constructeurs ou des sociétés de services ; les considérations de performances, de coût et de fiabilité sont importantes ;
- *maquettes* (ou prototypes) montées par des équipes de recherche universitaires ou industrielles ; les impératifs de la mise au point, de la communication et de la lisibilité des programmes doivent être pris en compte ;
- *outils de pédagogie* pris en charge par des équipes d'enseignants comme support pratique de cours système et réseau ; la facilité d'utilisation, la clarté des manuels d'utilisation et la rigueur des concepts utilisés permettent de rendre l'outil adapté aux besoins des enseignés.

Ces trois types différents suffisent à montrer qu'à vouloir retenir chaque considération mise en valeur, on ne peut qu'aboutir à un monstre à cinq pattes. A l'heure actuelle, SYNCOP ne prétend pas être bien adapté à chaque utilisation particulière : en le considérant comme une version de base, il semblera intéressant de donner à l'tuilisateur des moyens simples pour adapter le sous-système à ses besoins précis.

c) Coût et modularité de SYNCOP :

SYNCOP est construit comme un système modulaire composé de :

. *un noyau* : gestion des processus, de la synchronisation, des ressources, de la mémoire,

. *des services* : gestion du temps, des files, aide à la mise au point.

Le tableau suivant permet de préciser de quelle façon on peut disposer des différents modules ; la mention "au choix" indique que l'utilisateur a la possibilité de substituer un service par un autre suivant les mêmes spécifications externes.

Notation : G = cas général,
M = maquettes,
P = produits télé-informatiques,
E = enseignement et pédagogie.

Services	Observations
processus	G : obligatoire
synchronisation	G : obligatoire
ressources	G : obligatoire P : inutile si les ressources sont en nombre suffisant
mémoire	G : obligatoire service "au choix"
temps	G : facultatif P, M : le traitement des séquences associees aux réveils peut être isolé.
files	G : facultatif service "au choix"
aide à la mise au point	G : facultatif P : à retirer pour un produit "au point" E : fondamental

d) Conclusion :

SYNCOP constitue le premier investissement important fait dans le domaine de l'ingénierie de la télé-informatique et des applications réparties. L'expérience déjà acquise permet de conclure sur les actions à développer pour poursuivre l'effort entrepris :

- modularité des sous-systèmes et adaptation aux applications existantes ;
- primitives utilisateurs et langages hôtes assurant un interface SYNCOP avec pour objectif un *langage* de programmation des applications réseau ;
- sous-systèmes locaux et machines réseau d'exécution répartie avec pour objectif des *systèmes* ayant des fonctions réseau intégrées et des modes de communication standardisés entre tâches quel que soit leur site de résidence.

Deux projets en cours nous montrent que cette voie loin d'être abandonnée, connaît un approfondissement :

- la mise à disposition de primitives SYNCOP [LOT77] depuis le langage PL/1 permet l'écriture d'applications réseau lisibles et portables,
- le développement du projet SIGOR [DAN77] tend à mettre à disposition un outil d'écriture d'applications réparties, c'est-à-dire un outil, par rapport à SYNCOP, ayant une dimension supplémentaire

5.2 L'expérience de la portabilité sur le réseau CYCLADES

5.2.1 Extensibilité, hétérogénéité, portabilité :

La portabilité des programmes est ni un problème nouveau, ni un problème typiquement réseau. Nous l'abordons ici car il trouve une acuité particulière dans les réseaux hétérogènes, tels que Cyclades : des solutions particulières ont été trouvées pour lesquelles un bilan peut être tiré.

La portabilité d'un programme [MAL71] est généralement définie par la propriété d'adaptation de ce programme à des modifications de son environnement matériel et logiciel de base. Les techniques employées peuvent être classées en deux catégories :

- les niveaux de machines abstraites en couches de langages qui permettent d'isoler les modifications entre deux couches et en particulier lorsque le changement de matériel n'affecte que la couche de plus bas niveau,
- les générateurs de programmes tels que générateurs de compilateurs ou macro-générateurs.

Il y a un problème de portabilité seulement lorsqu'on ne programme pas dans des langages de haut niveau comme Fortran ou Cobol. Ceux-ci sont très mal adaptés aux besoins en cause ici.

Un certain nombre de langages ont été développés pour l'écriture de systèmes d'exploitation [WIR71],[WIR72],[CII05],[CII06],[ICH74]. Ils allient pour la plupart une facilité d'utilisation à une rapidité de compilation. Mais leurs implémentations restent le fait de quelques machines et faute d'une procédure rapide pour implémenter des versions nouvelles de compilateur, ces langages répondent mal aux buts poursuivis pour les applications réparties en milieu hétérogène.

Le réseau Cyclades avait dès ses origines besoin d'un outil répondant à quelques impératifs importants de la programmation système dans un environnement réseau :

(a) l'*extensibilité* : l'inventaire des besoins en matière d'applications réparties n'est pas du tout achevé et dans la phase actuelle plus qu'à toute autre période, les outils sont appelés à évoluer,

(b) l'*hétérogénéité* : les programmes écrits doivent être exécutables sur des machines ayant des instructions de base et des systèmes d'exploitation différents.

(c) la *portabilité* : les programmes doivent pouvoir être mis au point sur un site quelconque et les codes exécutables transférés sans autre artifice sur d'autres sites pour leur mise en oeuvre.

Répondant à ces trois objectifs majeurs, un langage de macros instructions FANNY[C10] a été défini qui permet :

- la *structuration des données* : les données sont des registres, des zones mémoire (initialisees ou simplement réservées), de taille variable :

octet, demi-mot, mot, double mot, des champs de tables de tailles analogues. Les tables sont des sections fictives que l'on ouvre et que l'on ferme en ouvrant une autre section fictive.

- La *manipulation des données* : les données sont manipulables avec ou sans indirection, avec ou sans index, comme une valeur immédiate. Elles peuvent être chargées, additionnées, soustraites, quelle que soit leur catégorie, soit entre elles, soit avec des valeurs immédiates, soit avec des variables de type adresse (mot ou octet). On peut charger des bits, tester des données et travailler sur des chaînes d'octets (déplacement et initialisation avec une valeur).

- La *structuration des programmes* : les programmes sont décomposables en modules, chaque module ayant ses points d'entrée et ses références externes. Le code des modules est découpable en routines : les routines sont l'objet de définitions disjointes. Les appels de routines se font avec retour. Les appels peuvent être empilés, récursifs et conditionnels. Les variables registre peuvent faire l'objet de définitions locales aux routines.

- La *structuration des routines* : les routines sont structurées en niveaux : boucles, avec ou sans test d'arrêt, actions conditionnées par un test. Les niveaux peuvent être imbriqués avec possibilité de revenir au début de n'importe quel niveau englobant. On a une instruction d'aiguillage sur test d'égalité d'une variable avec une liste de valeurs.

Pour produire des codes objets à partir de texte source FANNY, deux techniques complémentaires ont été utilisées :

- sur un type de machine (celle qui peut être considérée comme une machine de référence, la plus utilisée), une technique de macro-assemblage est pratiquée.
- sur les autres, on fait appel à la technique de bootstrapping et de macro-génération telle qu'elle est utilisée par le macro-générateur STAGE2 [POO70], [GIE72].

5.2.2 Bootstrapping, Macrogénération et Portabilité :

Le macro-générateur STAGE2 est tel que :
- on définit une machine abstraite, adaptée au problème à résoudre,
- on formule le programme dans le langage de cette machine,
- on établit la correspondance entre la machine abstraite et la machine cible,

- on génère le programme dans la machine cible.

La mise en oeuvre de STAGE2 sur une machine quelconque est une opération d'une durée d'une semaine ou deux;STAGE2, en effet, permet de définir non pas une machine abstraite mais une hiérarchie de machines permettant de réduire à un seul niveau de machines le travail d'adaptation machine abstraite-machine cible.

L'utilisation de STAGE2 reste très souple et n'apporte que peu de contrainte aussi bien sur le langage source que l'on souhaite utiliser (jeu de caractères, nombre de registres...) que sur les langages objets :

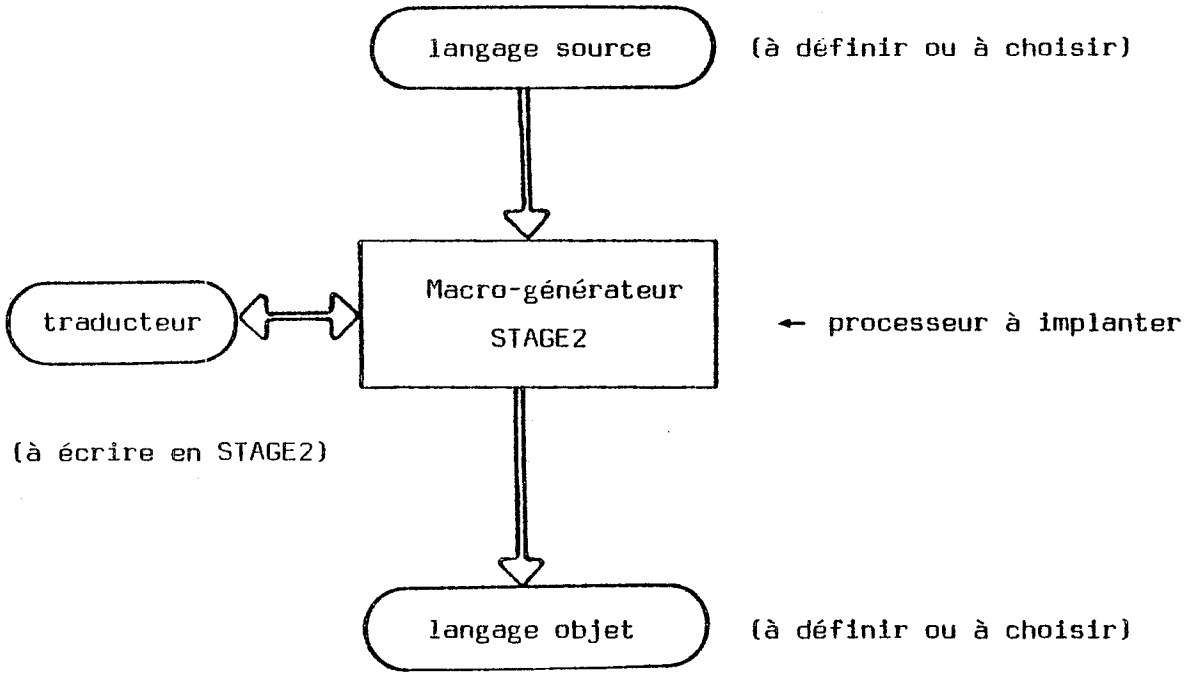


Figure 5.8 - Utilisation de STAGE2 (1)

Certaines mises en oeuvre de STAGE2 permettent, à partir d'un langage source unique et d'un traducteur ad hoc, d'obtenir tous les codes objets souhaitables, ce qui est une bonne approche de la transportabilité automatique :

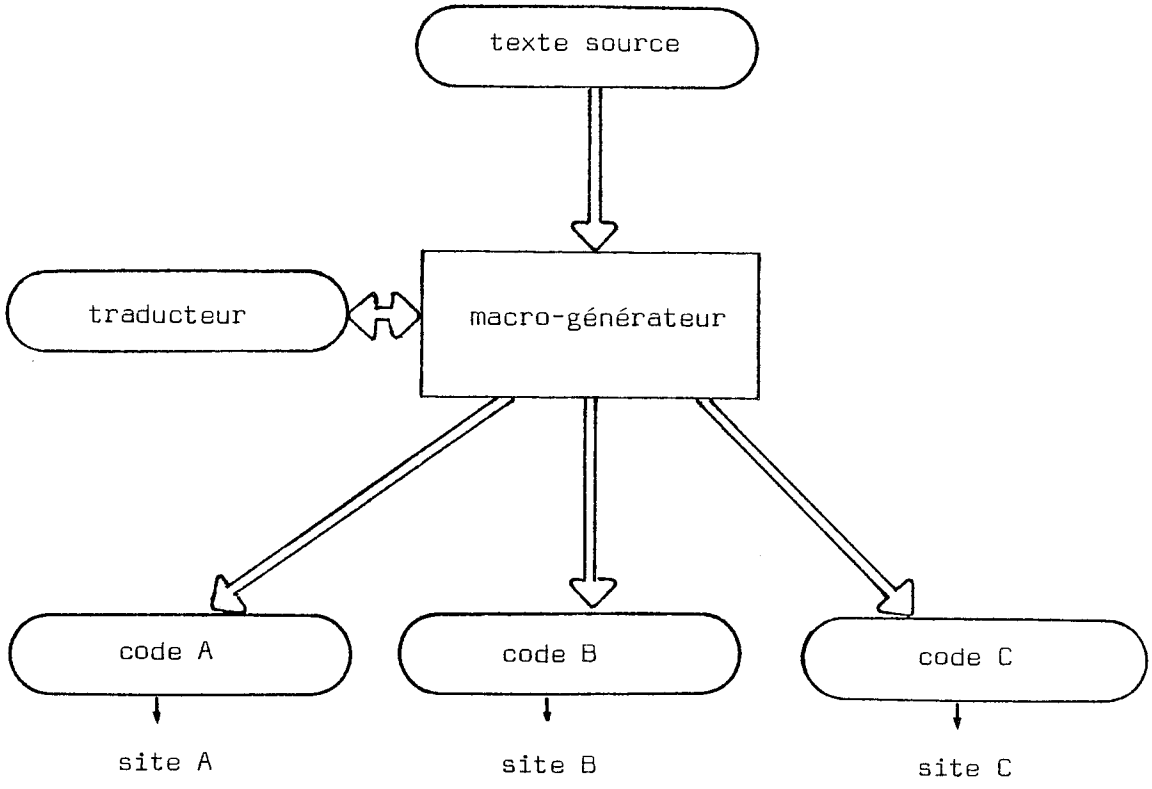


Figure 5.9
Utilisation de STAGE2 (2)

5.2.3 L'expérience de la portabilité :

Cette expérience a été menée sur des machines ayant de nombreux points communs (IRIS 80/10070, SIEMENS 4004, OS 360, IRIS 45/50/60) :

- mots de 4 octets, octet de 8 bits,
- jeu d'instructions de base comparable,

et certaines différences :

- instruction de longueur différente,
- adressage mot ou octet,
- taille des valeurs immédiates,
- manipulation de piles,
- registres de base et d'index.

Cette situation de départ conduit à définir certaines restrictions (ou conseils) d'utilisation pour que le code écrit soit portable.

La portabilité ne concerne pas les appels aux services des systèmes d'exploitation locaux ou réseaux pour lesquels les sous-systèmes réseaux normalisés - tel SYNCOP - doivent apporter une réponse.

Les bases de l'expérimentation à l'heure actuelle concerne des traducteurs pour trois machines cibles :

- Siemens 4004 et OS360
- CII-HB 10070 et IRIS 80
- CII-HB IRIS 45/50/60.

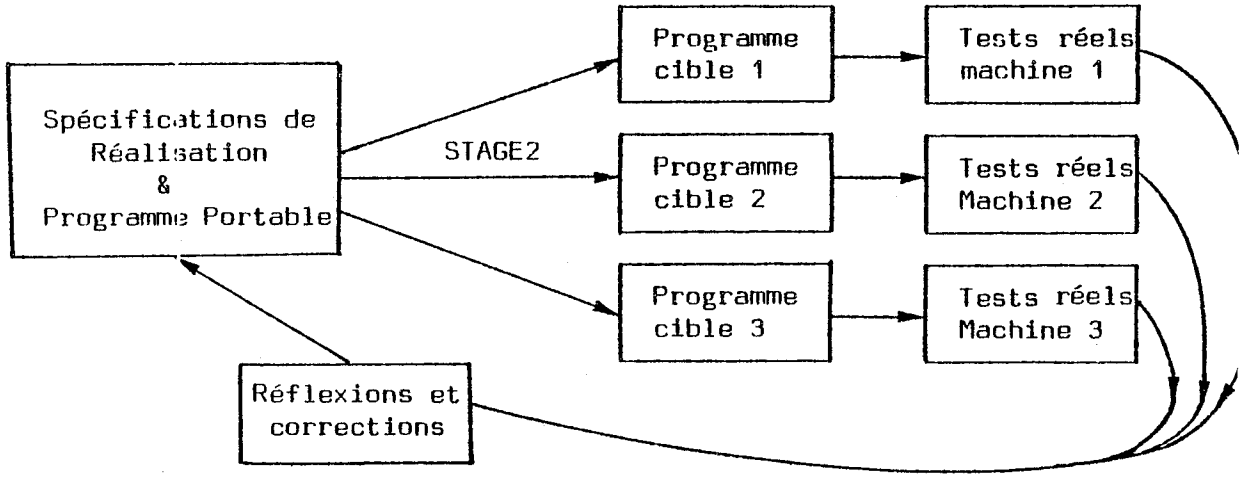
Ces traducteurs ont été utilisés dans le cadre de plusieurs projets "portables" :

- le projet CRIC [C1]
- le projet ST1 Cyclades [ZIM75],[KER74]
- le projet ST2 Cyclades [ELI74]
- le projet IGOR [SER74].

De ces expérimentations on peut dégager quelques conclusions :

- bien que les règles d'emploi de STAGE2 soient précises et bien définies, le transport d'un programme nécessite de nombreux tests à tous les niveaux (texte source, machine abstraite, texte d'objet). A cet égard on peut mentionner la technique utilisée par l'équipe du projet ST2 qui menait de front les spécifications d'implémentation de la ST2 et l'écriture de trois prototypes. Le mouvement de va et vient entre le texte source et les tests en réel sur les machines cibles permettaient de conserver une cohérence entre plusieurs réalisations en ayant un même texte source et de corriger les spécifications en fonction d'observations faites à propos de tests réels :

Figure 5.10 : Mise en oeuvre d'un programme portable



- La prise en compte de machines différentes (par exemple différence sur l'adressage ou la manipulation de piles) ne permettent pas avec cette technique d'obtenir des codes générés optimisés mais seulement des prototypes constituant une base solide pour des optimisations nécessairement spécifiques à chaque machine. C'est ce qui a été mené à bien pour le développement dans un contexte industriel de la ST2 IRIS80.

5.3 L'ingénérie des applications réseaux

En abordant les deux problèmes des sous-systèmes télé-informatiques et réseaux d'une part, de la portabilité des programmes d'autre part, nous avons envisagé deux domaines où les réseaux et le télé-traitement faisaient apparaître des caractéristiques spécifiques. Ces domaines ne sont pas isolés : nous en retiendrons cinq particulièrement importants pour lesquels il reste beaucoup à faire pour faciliter le travail des concepteurs et des réalisations d'applications et de systèmes réseau.

5.3.1 La définition des protocoles et des interfaces

Définitions :

Protocoles : ensemble de règles régissant les échanges entre deux correspondan (entités).

Interfaces : ensemble de moyens (instructions et structures de données) utilisables par deux entités pour agir l'une sans l'autre.

Les différents systèmes de communication tels que réseaux de commutation, bus définissent un environnement spécifique pour le mode de communication et d'échange entre les entités composantes d'une application ou d'un système. Comme pour la spécification d'un système ou d'une application classique, il y a nécessité d'abstraction dans la spécification et de décomposition des éléments fonctionnels. Une technique largement développée dans le réseau consiste dans la définition de niveaux : les relations entre niveaux étant consignées dans des interfaces, les règles d'échanges entre entités de même niveau faisant, elles, l'objet d'un protocole.

Dans le cadre des réseaux généraux, tels que Cyclades, ARPA ou EIN, ces outils de spécification ne sont pas que la préoccupation d'une équipe qui, menant à bien un projet, éprouve le besoin pour le spécifier tout au long de son avancement d'employer des outils, un langage profitable à l'équipe elle-même et compréhensible à l'extérieur.

[ZIM74] souligne que protocole et interface constituent les deux points forts dans la spécification des réseaux informatiques en particulier ceux qui sont généraux et hétérogènes :

- l'hétérogénéité est surmontée par la définition d'interface réalisable sur n'importe quel site et dont la seule connaissance est suffisante pour la réalisation d'une entité particulière,

- la généralité est acceptée dès lors que les équipes participants au réseau (les entités) se comprennent dans l'acceptation d'un protocole commun pour chaque niveau de service que l'on veut rendre.

Pour atteindre ces objectifs, le support de définition des protocoles et des interfaces doit être rigoureux, clair et compréhensible de façon non ambigü par quiconque qui y a affaire. Plusieurs démarches sont proposées :

[ZIM74], [ELI73] où on emploie un modèle de machine abstraite manipulée par des commandes et échangeant des informations avec d'autres machines abstraites. La machine abstraite correspondant à un certain niveau de protocole (ex. le protocole ST1 Cyclades [ZIM75]) est décomposé en un ensemble de machines élémentaires. Pour chaque machine élémentaire on fournit un ensemble de spécifications en pseudo-Algol qui recouvre :

- le rôle joué par la machine
- l'adressage
- les interfaces (instructions et structures de données)
- les mécanismes internes
- les traitements des commandes.

[BOC75] où on propose, à partir d'un même type de décomposition en niveaux d'abstraction, une formalisation des protocoles élémentaires en termes de machines d'états finis, avec la démonstration de certaines propriétés. Des spécifications de réalisation, en concurrent Pascal, sont ensuite énoncées.

Le formalisme en termes de machines et de niveaux de machines constitue un niveau de spécification abstrait et d'une certaine rigueur. D'une part certaines ambiguïté subsistent dans l'interprétation des commandes ; D'autre part, ce formalisme ne rend pas aisé des opérations de comparaison entre des protocoles équivalents faute d'un langage commun dans la qualification des objets manipulés : une voie est traçée avec le formalisme de

Data Semantics [ABR73] dans lequel la structure générale de l'architecture du système POLYPHEME [B12] ainsi qu'une étude composée de plusieurs protocoles de transports [AND77] ont été présentées.

5.3.2 Le logiciel de base réseau

Pour les réseaux informatiques, le logiciel de base sera l'équivalent du système d'exploitation pour les centres informatiques mono-ordinateur. Ce logiciel sera composé d'un ensemble de protocoles et d'interfaces.

- le logiciel de plus bas niveau est constitué d'une *machine de commutation* avec un protocole inter-noeuds de commutation et un interface de connexion à la machine de commutation sur un noeud,

- la *machine de transport* d'information s'interface sur la machine de commutation. Elle est définie à l'aide d'un protocole de transport d'informations inter-sites et d'un interface de connexion à la machine de transport. Pour le réseau Cyclades, cet interface est constitué de portes, de flots, de télégrammes, de lettres avec un certain nombre de relations entre ces objets [B12].

A ce niveau de logiciel, les entités qui utiliseront l'interface de connexion à la machine de transport sont quelconques. Les différents outils décrits aux chapitres 5.1 et 5.2 aident à la spécification et la réalisation de ces entités conçues comme des *entités locales* à un site de réseau et utilisant le service de transport réseau comme un service système comme la gestion de fichiers ou la gestion du temps. La sémantique des informations véhiculées par le réseau échappe au domaine d'application de ces outils.

Sans outil supplémentaire, deux nouveaux protocoles ont été définis sur la majorité des réseaux généraux d'ordinateurs : le *protocole appareil virtuel* qui comprend un protocole d'échange entre une entité appareil et une entité service non répartie et un interface de connexion appareil réel - appareil virtuel. Le *protocole transfert de fichiers* qui comprend un protocole d'échanges entre une entité fichier virtuel origine et une entité fichier virtuel destination et un interface de connexion fichier virtuel - fichier réel.

Ces protocoles constituent le logiciel de base pour l'utilisation des réseaux d'ordinateurs. Sans interprétation de la sémantique des données véhiculées, il reste insuffisant pour l'écriture d'applications réparties.

Les échanges entre les entités dans lesquelles seront décomposées les applications réparties font appel à des mécanismes de synchronisation réseau dont l'expression simple passe par la définition de nouveaux objets.

Dans le réseau SOC [SOM75] les processus sont des jobs au sens des sites du réseau. Ces processus s'activent entre eux, s'échangent des fichiers au sens des sites du réseau ; ils se synchronisent sur ces échanges et ces activations.

Dans le système S-IGOR [DAN77] on associe aux processus un processeur : le processeur est une procédure au sens du langage intermédiaire LI de SIGOR (cf. notion de procédure d'Algol 60). L'unité d'information qui transite sur le réseau est la procédure rendant possible le transfert d'algorithme. Les processus sont structurés suivant une arborescence avec des mécanismes d'échanges père-fils et fils-fils via le père. Les outils de synchronication constituent une extension réseau de ceux qui ont pu être défini dans SYNCOP.

Le système S-IGOR constitue l'archétype du logiciel de base d'un réseau informatique hétérogène où la définition des processus mis en jeu n'est lié ni aux sites, ni à l'application. L'exemple d'un système de gestion de base de données SOCRATE réparti et utilisant S-IGOR le montre bien [A11], [DAN77]. Dans POLYPHEME les primitives réseau mises en oeuvre par l'exécution d'une requête en termes de la vue globale sont fonctionnellement équivalents à celles que met à disposition le système S-IGOR.

5.3.3 L'utilisation des structures distribuées

Pour chaque réseau informatique (ou système multi-processeur), nous nous intéressons ici au système muni d'un certain logiciel de base dont la fonctionnalité est variable suivant les objectifs du système informatique en cause.

Lorsque le système informatique réparti ou multi-processeur a pour objectif l'amélioration d'un service existant ou l'extension d'un service sans modification des habitudes de l'utilisateur, la composante "répartition" est invisible à l'utilisateur.

C'est le cas de DCS [FAR72] où les processus ont des noms globaux, la dénomination des sites est implicite ; lorsque l'utilisateur fait appel à un service, c'est le système qui choisit suivant des critères de coût ou de disponibilité le processus le plus approprié.

C'est le cas des systèmes de gestion de fichiers où l'utilisateur dispose d'une méthode d'accès unique pour les fichiers locaux ou distants. Le site d'implantation du fichier est contenu dans un catalogue réseau connu du système comme dans MADRE[B8], DECNET (architecture réseau DNA de DEC) [PAS77]. Dans RSEXEC le site est une des composantes du profil de l'utilisateur.

C'est le cas des systèmes de gestion de bases de données où l'espace mémoire secondaire est gérée comme une ressource banalisée du réseau. L'utilisateur manipule des fichiers sur des volumes eux-mêmes gérés par le système dans le cadre d'un espace virtuel réparti. Dans DCN [MIL75], [MIL76] cet espace virtuel est composé de segments de 64 à 8192 octets ; l'adressage réseau de 32 bits est décomposé en une adresse de volume (16 bits) et une adresse relative dans le volume. Dans LADDER (de Stanford Research Institute) [MOS77], la méthode d'accès aux bases ne connaît pas le site des bases, c'est une méthode d'accès fichier (FAM) qui, comme pour MADRE, gère cette composante de l'adressage.

Pour d'autres systèmes la désignation et la manipulation explicite des sites ou des processus est une possibilité offerte à l'utilisateur [DCA73]. Dans le système Illiac IV [KUC68], on met à sa disposition un mécanisme destiné à l'exécution d'opérations sur des vecteurs ou des tableaux, charge à l'utilisateur d'en tirer parti en rangeant correctement les éléments des vecteurs ou des tableaux dans les différents bancs mémoire.

Pour le système SOC, la variable CPU (unité de controle d'ordinateur IBM) est une des caractéristiques des entités jobs et fichiers sur lesquels portent les opérations réseau. Dans S-IGOR, le site est associé à la procédure et au fichier. Pour le système de gestion de bases de données réparties POLYPHEME [B12], le site est associé à la structure locale (en fait l'association porte Cyclades, base locale). Les hôtes d'UNIX sont manipulés comme des périphériques avec une méthode d'accès unifiée pour les fichiers, les périphériques et les processeurs quelconques.

Il n'apparaît pas de règle générale dans l'utilisation de l'entité site, de la composante répartition pour les différents systèmes. On peut distinguer deux grandes catégories : ceux pour lesquels il y a transparence, et les autres. Pour les autres, au site est associé un processus mais les opérations autorisées dessus sont très variables suivant la finalité du réseau : gestion de fichiers, gestion de bases de données, gestion de processus quelconques.

5.3.4 La validation et la mise au point

La production de programmes dans une informatique répartie est soumise à des contraintes spatiales, organisationnelles, topographiques, humaines qui sont variables suivant les sites et les objectifs des réseaux.

Quelle que soit la finalité du réseau informatique, il y a la nécessité de définir de façon centralisée et unique le système global. A partir de cette base conceptuelle unique, la décomposition modulaire peut se faire et pour chaque site on aura un module de description des structures de données, des spécifications locales, des spécifications des commandes réseau.

Ce sont sur ces bases que l'on a mené à bien la réalisation d'un service de transport. La production des logiciels station de transport était poursuivie par des opérations de tests de proche en proche [ANS76], [QUI76], [AND76].

- tests locaux : ils sont possibles dès lors qu'il y a symétrie dans les protocoles et indépendance des niveaux ; le réseau est simulé par un miroir, les utilisateurs par des générateurs de trafic.

- tests d'interface : ils sont menés à bien en opérant par rebouclages successifs à chaque niveau de machine pour isoler les causes d'erreur et accélérer la mise au point.

- tests en vraie grandeur : ils sont réalisés avec chaque site concerné par l'application en cours de test. Chaque site (ou chaque centre d'exploitation) doit conserver la maîtrise de ces opérations, en particulier en définissant les conditions de test, les cahiers de recettes ...

Ces opérations sont rendues plus aisées avec le développement des mécanismes de télé-chargement, télé-démarrage, télé-voisinage... Quatre types d'expérience ont été menés à bien dans le cadre du réseau Cyclades :

- les tests des stations de transport mettant en oeuvre une conception centralisée, des productions décentralisées, des tests indépendants puis concertés, une exploitation décentralisée. Pour ce faire, les nécessités premières sont une clarté des protocoles, des concertations entre personnes responsables de son secteur (production ou exploitation), des outils de télé-visionnage,

- les tests des procédures de ligne rendues plus faciles avec des outils tels que TIPAC : ce système microprocesseur permet de visionner en réel ce qui se passe sur la ligne et sous une forme exploitable rapidement par le testeur,

- les tests de concentrateur de terminaux mettant en oeuvre une conception et une production centralisée, des tests concentrés avec le centre de contrôle et une exploitation concentrée avec télé-chargement. Pour concilier exploitation réseau en démarrage et exploitation locale en cours, les conditions de tests et de recettes des produits livrés doivent être très rigoureuses pour assurer aux utilisateurs un service continu dans le cadre d'une exploitation stable. Ces conditions sont nouvelles et doivent faire l'objet dans l'avenir de définitions précises indispensables pour assurer une promotion convenable des services réseau pour des utilisateurs habitués aux services mono-ordinateurs,

- les tests de maquettes (MADRE, FRERES,...) qui n'intervient pas dans la fourniture du logiciel de base réseau et à ce titre leur test ne peuvent perturber en aucune manière l'exploitation du réseau. Faute d'outils

de télé-chargement et télé-visionnage de ce type de logiciels, les conditions de test sont extrêmement difficiles et encouragent peu à l'innovation :

- obligation de se déplacer de site en site,
- nécessité d'une coordination des personnes responsables du logiciel sur ces différents sites,
- problèmes d'horaire, de disponibilité du service de transport...

Ce dernier domaine est celui dans lequel il reste le plus à faire pour favoriser des applications spécifiques, la définition de nouvelles méthodes d'accès, en bref la construction de véritables réseaux informatiques apportant une dimension nouvelle aux services offerts.

6. ARCHITECTURE DES SYSTÈMES RÉPARTIS

Nous analysons donc dans ce chapitre quelques caractéristiques des systèmes répartis. Au paragraphe 6.1, nous présentons un impact de la répartition sur les architectures de systèmes. Les systèmes de gestion de bases de données réparties sont développés au paragraphe 6.2 où nous présentons une architecture pour un système de coopération de bases de données dans un réseau. Le point est fait sur les nouvelles architectures réseau disponibles (paragraphe 6.3). Ensuite nous présentons un exemple de nouveau service réseau : la machine de données, qui peut constituer un des éléments des nouvelles architectures de systèmes répartis (paragraphe 6.4).

6.1. Impact de la répartition sur les architectures de systèmes :

Un système informatique classique se présente à l'utilisateur avec un seul processeur de traitement (les ordinateurs ayant plusieurs unités centrales n'offrent pas à l'utilisateur les outils pour programmer du traitement en parallèle, par exemple). L'utilisateur n'a à connaître que trois aspects du système :

a) le sous-ensemble d'*entrée* qui constitue tout ce qu'il pourra soumettre au système et sous quelles formes (structure de données ou interface d'entrées).

b) le sous-ensemble de *sortie* qui constitue tout ce qu'il obtiendra du système et sous quelles formes.

c) une fonction (*processeur abstrait*) qui, à tout élément d'entrée, associe un élément de sortie [BOU77].

Dans les systèmes répartis, il n'y a pas un seul processeur abstrait pour tous les utilisateurs du réseau. Au contraire, les réseaux informatiques qui réalisent un partage des ressources de différents systèmes à l'aide d'un réseau de communication sont faits d'un grand nombre de processeurs ; ceux-ci, pour l'utilisateur et en fonction des applications, sont regroupés de façon variables au sein d'applications réparties.

La spécification de ces applications réparties est rendue possible par la disposition des spécifications des processus associés aux ressources partagées du réseau et des spécifications des protocoles de communication et d'échange entre les différents processus composants des applications.

Dans le cadre de la méthode d'accès réseau pour les fichiers MADRE, nous avons eu l'occasion de développer une telle application, les processus composants étant les clients et serveurs MADRE, le protocole de communication étant le protocole fichier MADRE (cf. chapitre 3).

Dans le cadre de l'étude des méthode d'accès réseau pour les bases de données, nous avons présenté un mode de structuration des systèmes de gestion de bases de données permettant de définir plusieurs niveaux de méthode d'accès réseau. Cet exemple de système permet de dégager un impact de la répartition sur les architectures de système. Le découpage en niveaux fonctionnels autorise différents modes de partage de ressources dans le cadre d'un réseau ; à chaque niveau est associé un protocole de communication entre entités de même niveau ; entre deux niveaux un interface (cf. figure 6.1).

Suivant le niveau retenu, on met en oeuvre différents services bâtis autour de ressources partageables dans le cadre du réseau :

- *niveau terminal virtuel* : les ressources sont les terminaux définis par un protocole terminal virtuel (qui comprend les caractéristiques des terminaux supportés et les règles d'utilisation des services disponibles sur le réseau depuis ces terminaux) [ZI276].

- *niveau concentrateur interpréteur* : les ressources sont les concentrateurs de terminaux (type Cyclades ou satellite multifonctions CII-HB [CHB77]) ou frontaux.

- *niveau méthode d'accès bases de données* : les ressources sont les processeurs bases de données. Plusieurs exemples ont été développés dans le cadre d'UNIX (bases de données médicale distribuée [CHG76]) ou de XDMS [CAN74].

Dans ce dernier projet, chaque base est locale à un site définissant autant de "back-end" ou "dorsal". Cette notion de dorsal est étendue par [CHP77] qui définit un dorsal réparti où la composante site est transparente à l'utilisateur.

- *niveau méthode d'accès fichier* : la ressource est le processeur fichier accessible et partageable par l'ensemble des utilisateurs des sites du réseau. Plusieurs projets développent ce processeur particulier [CHG76], [CHA76].

En conclusion, une telle structuration des systèmes de gestion de bases de données favorise la répartition de ses fonctions et la définition de nouvelles méthodes d'accès réseau. Nous retiendrons plusieurs impacts de la répartition sur l'architecture des systèmes informatiques : modularité, structuration des fonctions et définition de processeurs coopérants suivant des protocoles et des interfaces, définition de nouvelles ressources telles que frontaux, dorsaux, processeurs de données.

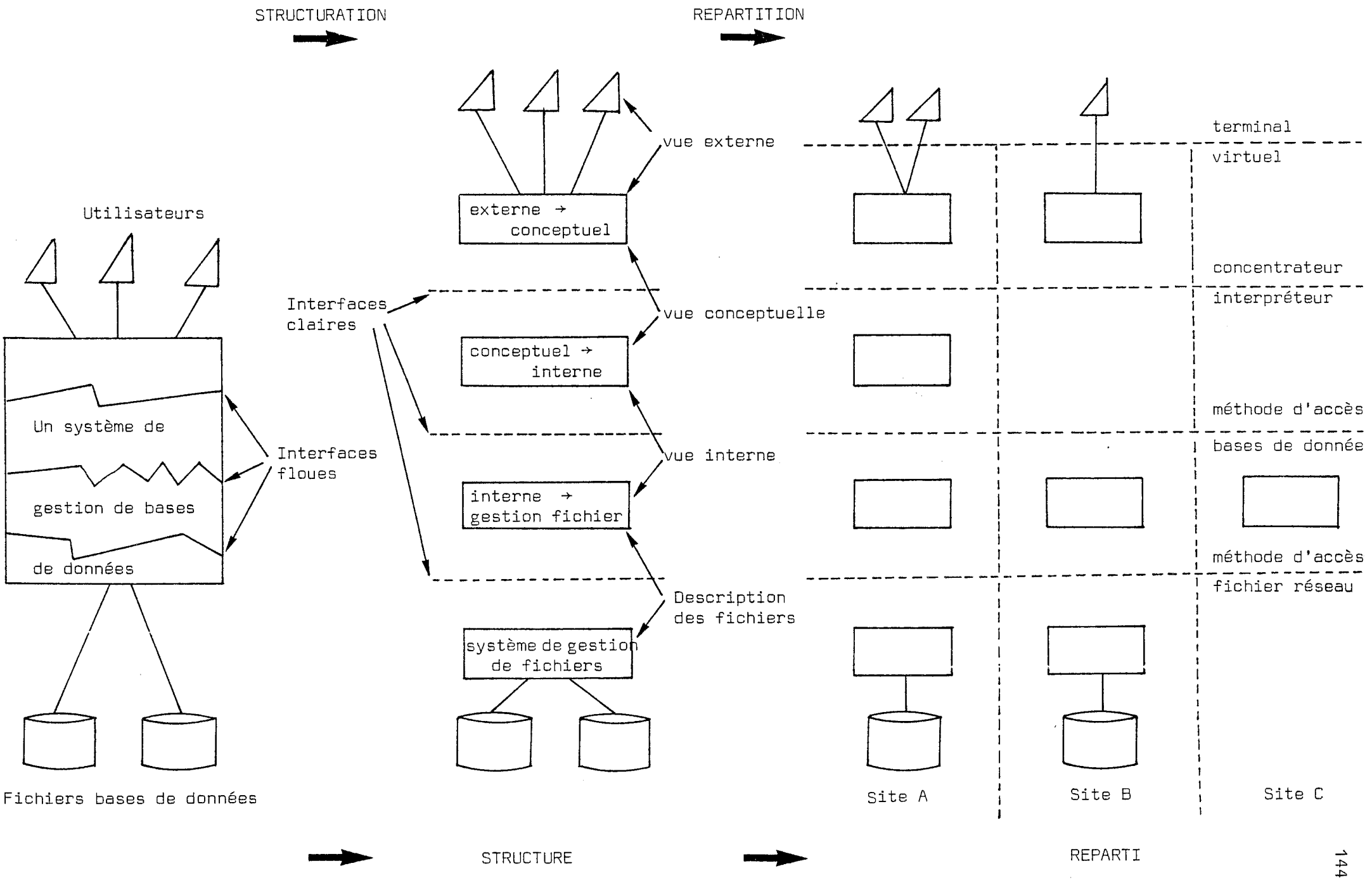

STRUCTURATION
REPARTITION
Utilisateurs
Un système de
gestion de bases
de données
Interfaces
floues
Fichiers bases de données
vue externe
externe → conceptuel
Interfaces
claires
vue conceptuelle
conceptuel → interne
vue interne
interne → gestion fichier
Description
des fichiers
système de gestion de fichiers
terminal
virtuel
concentrateur
interpréteur
méthode d'accès
bases de donnée
méthode d'accès
fichier réseau
Site A
Site B
Site C
STRUCTURE
REPARTI

Figure 6.1

6.2. Modèles de répartition et de coopération de bases de données

Nous avons eu l'occasion au chapitre 5.2 d'aborder le problème du passage d'une application locale en application réseau sous le double aspect des méthodes d'accès (compatibilité des primitives...) et de la portabilité des programmes en milieu hétérogène. Il n'est plus question ici des programmes, mais des données. Le point de départ est constitué de bases de données implantées sur des sites (bases locales). Ces bases sont définies à l'aide de modèles de structuration de données appelées schéma ou structure ou modèle suivant les origines que nous quelifierons de *vues locales* de façon uniforme.

6.2.1. Les modèles de répartition de bases de données :

Pour définir la notion de base de données réseau, c'est-à-dire la collection d'informations structurées gérée dans le cadre d'un réseau, deux démarches apparaissent possibles ; dans les deux la prise en compte, au départ, de bases avec leurs vues associées écarte la conception globale d'une application réseau dont aucun élement n'existerait au préalable sur des sites particuliers [GAR76].

a) La première consiste à définir une *base de données réparties* comme formée de sous-ensembles, chacun ayant un site de résidence unique et ayant été extrait d'une base locale.
Cette définition n'est possible que si les bases sont homogènes. En cas de bases hétérogènes, des correspondances de vues devront être définies, ainsi que des conversions de bases.

Une fois la base répartie constituée, on peut l'exploiter comme une base classique : le réseau n'apportant qu'un support nouveau pour cette base. Cette axe de travail correspond à celui développé avec des bases ayant un espace virtuel réparti (cf. chapitre 3).

[ROC73] propose la notion de banque de données virtuelle qui, basée sur un espace virtuel réparti, propose au niveau des vues (schémas) et des langages utilisateurs plusieurs modalités d'expression de la répartition.

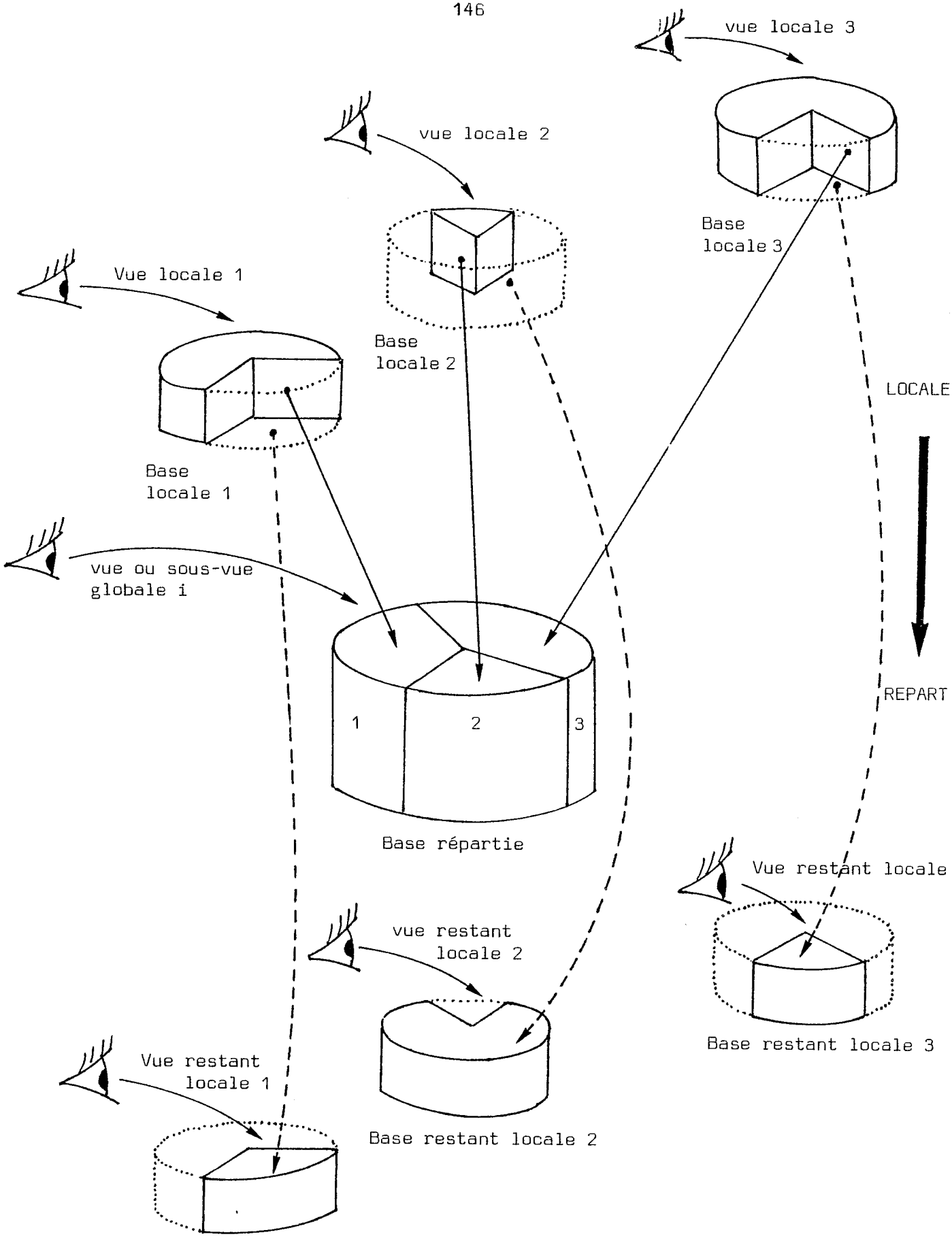


Figure 6.2

Répartition de bases de données

La constitution de telles bases réparties reposent sur une forte intégration dans le réseau d'informations d'origines diverses. Une fois la base constituée avec une propre vue associée, dite vue réseau ou vue globale, l'accès à cette base ne peut se faire que par le biais de cette nouvelle vue (cf. figure 6.2).

b) Une autre démarche consiste à bâtir des bases de données réseau sans modifier ni l'état des bases locales, ni leurs conditions d'exploitation en local. Dans ce cadre, l'aspect réseau est envisagé en plus. L'utilisateur réseau connaît une base réseau définie comme étant composée de bases locales ayant leurs vues propres, visibles du réseau à l'aide d'une vue globale dont les projections sur les bases locales se font à l'aide de vues locales.

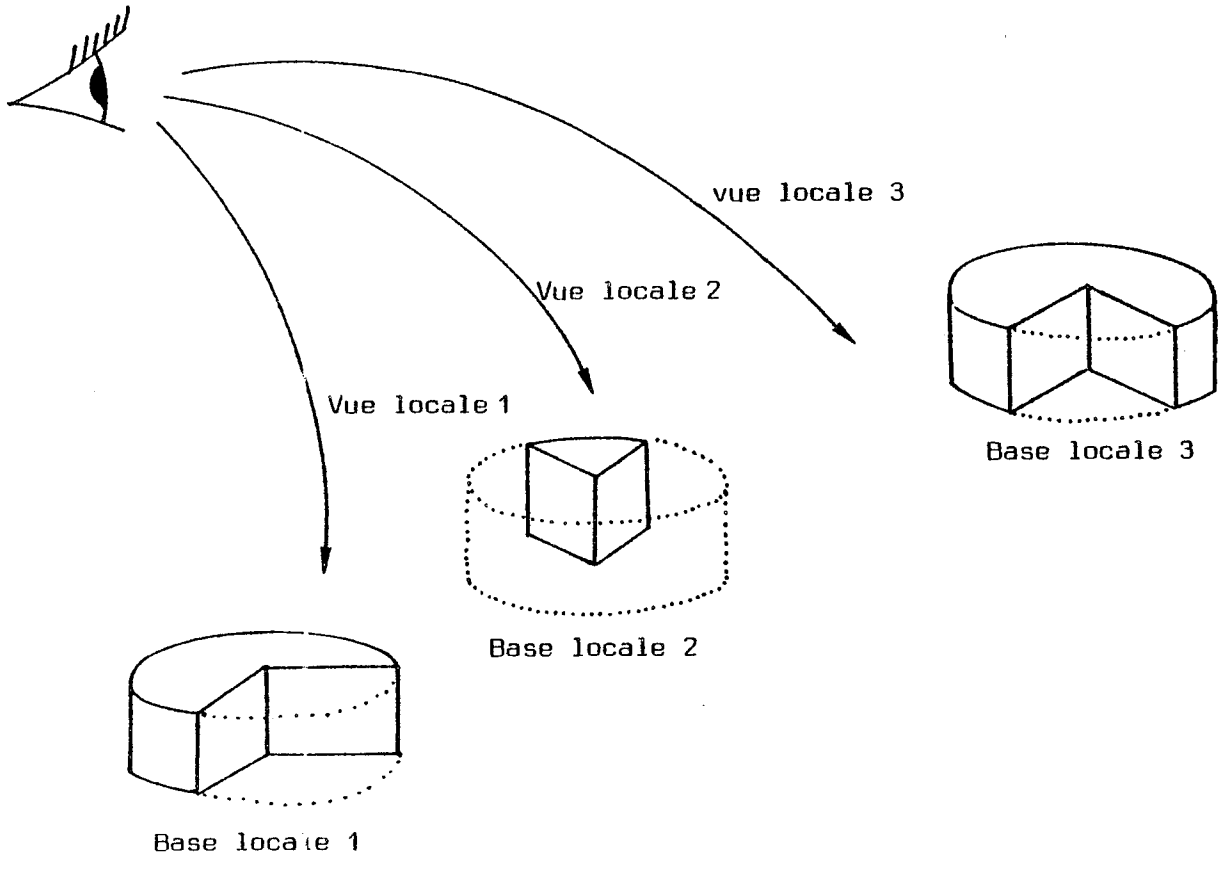


Figure 6.3

Bases de données indépendantes

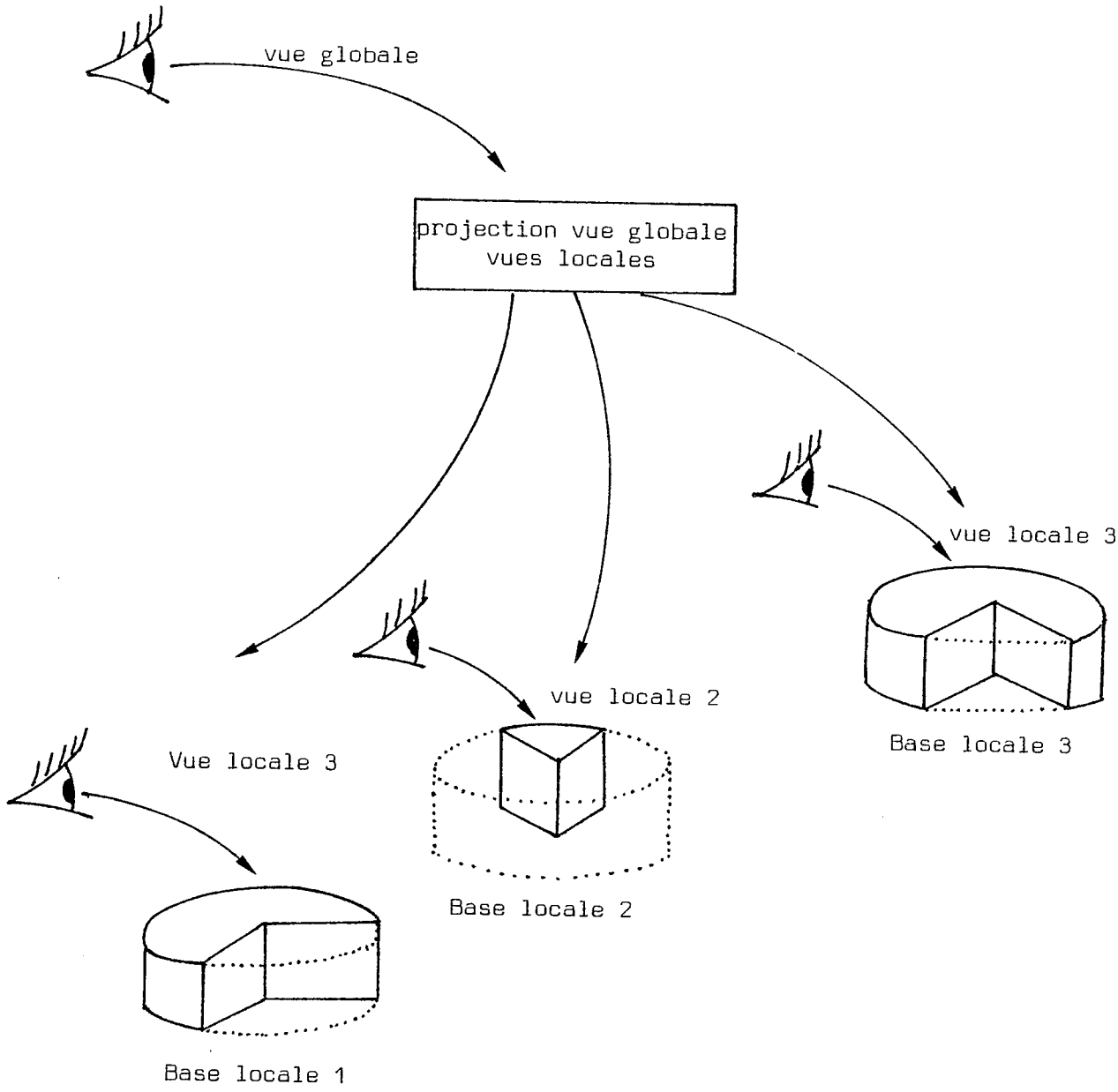


Figure 6.4
Bases de données coopérantes

Cette démarche est celle qui retient le plus l'attention des chercheurs en bases de données à l'heure actuelle : le projet POLYPHEME la développe [B12] comme plusieurs équipes participantes au projet pilote SIRIUS [IRI76], [BOS77], [LEB76]. Les nouvelles vues réseau qui sont offertes ou *vues globales* sont construites dans l'optique d'une *coopération* de plusieurs bases de données (cf. figures 6.3 et 6.4).

6.2.2. Les modèles de coopération de bases de données :

Pour l'utilisateur réseau (ce nouvel utilisateur), les systèmes de coopération de bases de données fournissent un nouveau service : lui permettre de connaître de façon transparente un ensemble de bases nées de façon autonome ayant leur organisation, leur système de gestion, leurs utilisateurs (cf. 6.2.2.1.) ; cette connaissance est complétée par la fourniture d'outils permettant des rapprochements sémantiques entre des données aux formats variables, mais à la signification voisine pour l'utilisateur ; c'est ce que recouvre la notion de vue globale (cf. 6.2.2.2.).

Le traitement des requêtes de l'utilisateur, au niveau global, suppose un certain type d'architecture du système de coopération : contrôle de l'exécution des transactions dans le réseau, cohérence, élaboration des réponses et synthèse des résultats (cf. 6.2.2.3.). Les systèmes de coopération posent plusieurs problèmes de localisation de fichiers et de processeurs qui sont abordés au paragraphe 6.2.2.4.

6.2.2.1. Les bases locales : hétérogénéité et standardisation :

Les bases locales ayant leurs origines spécifiques, il est naturel de trouver dans des réseaux hétérogènes des systèmes différents de gestion de bases de données. Cette hypothèse impose de trouver un mode de description général qui puisse rendre compte des différents systèmes. Dans le projet RESEDA [GAR76] le modèle entité-relation est choisi comme outil de description. Dans le projet POLYPHEME, c'est le modèle "Data Semantics" [ABR73] qui a été retenu. Pour ce dernier projet, les raisons de ce choix sont développées dans [B12] ; nous retiendrons ici deux critères importants :

- le premier est la démonstration faite de la possibilité de traduire n'importe quel système (relationnel, DBTG, hiérarchique type IMS) en termes de Data Semantics (la traduction inverse est possible dans certaines conditions),
- le deuxième est la capacité d'extension du modèle, donc sa stabilité : ceci permet de choisir un modèle unique pour les bases locales et la vue globale (cf. 6.2.2.2.).

Les vues locales contiennent, dans une forme standardisée, la description des données de la vue globale présentes dans la base locale et un mécanisme d'accès à ces données. Pour la vue globale, les bases globales ne sont vues qu'avec ces vues locales : les systèmes locaux peuvent être assimilés à des *automates* manipulant des objets à l'aide d'un ensemble de fonctions d'accès à ces objets.

6.2.2.2. La vue globale :

Dans la vue globale, il faut décrire les objets locaux et leurs fonctions d'accès tels que les automates locaux les mettent à disposition. Cette description ne peut être une simple transposition au niveau du système de coopération d'informations extraites du niveau local. Elle doit traduire la façon dont on voit ces objets et dont on veut relier, rapprocher sémantiquement des objets de bases locales différentes [ADI77].

Décrire au niveau global des objets locaux, c'est définir un ensemble d'objets globaux et des fonctions d'accès globales. Cette description peut nécessiter la fabrication d'un objet unique à partir de plusieurs objets locaux. Par exemple, on peut dire qu'il y a un objet "animal" au niveau global à partir des objets locaux "chien" sur le SGBD1, "chat" sur le SGBD2 et "souris" sur le SGBD3.

La duplication d'objets sur des bases locales différentes peut être rendue de façon différente dans le vue globale. Par exemple ces objets peuvent être désignés au niveau global en les différençiant suivant leur origine, ou bien en prenant une base locale pour base de référence : lorsqu'un accès à cet objet est lancé, on recherche d'abord dans cette base et, si cet accès est infructueux, on recherche dans une deuxième base, etc. On met ainsi en évidence les fonctions "Avez-vous ? et "Donnez-moi" développés dans [ROC73].

D'autres objets peuvent être définis dans la vue globale, en particulier pour aider l'utilisateur réseau à faire des synthèses de résultats. D'une façon générale, ces objets seront construits en appliquant un certain programme aux objets locaux.

6.2.2.3. L'architecture du système de coopération :

Un système de coopération dans un réseau d'ordinateurs, c'est d'abord une application répartie. Les spécifications de son architecture doivent donc respecter les règles de spécification des applications réparties, c'est-à-dire définir des fonctions-processeurs de base de l'application ; l'architecture du système sera faite de niveaux fonctionnels avec des interfaces, les processeurs de niveau équivalent communiquent en respectant des protocoles.

Les différentes architectures [BOT77], [GAR76], [LEB76], [B12] respectent la hiérarchie qu'impose la définition des vues globales par rapport aux vues locales et la définition des nouveaux utilisateurs réseaux (cf. figure 6.5). On considère que les utilisateurs (programmes ou terminaux) ont à leur disposition leurs propres outils (langage de manipulation de données, langages de description de données ou *vues externes*) correspondant au mieux aux types d'application qu'ils mettent en oeuvre. La projection vue externe - vue globale permet de construire des programmes dans les termes de la vue globale (analyse syntaxique et compilation).

La projection de la vue globale sur les vues locales consiste à décomposer le programme dans les termes de la vue globale en un ensemble de sous-programmes exécutables sur chaque automate SGBD local avec des règles d'ordonnancement de ces sous-programmes. L'exécution de ce programme se traduira par un aiguillage, une gestion du parallélisme et de la synchronisation et le traitement de résultats intermédiaires qui peuvent conditionner la poursuite du programme et la délivrance des résultats.

Les sous-programmes exécutables par les automates locaux sont exprimés dans les termes d'un SGBD standard. Après une traduction, l'automate déroule l'algorithme ad hoc et délivre des résultats à l'image d'un serveur SGBD classique. Ces résultats sont mis en forme en vue de leur communication à la vue globale : pour celle-ci leur traitement correspond à la fabrication soit d'un résultat intermédiaire nécessaire à la poursuite du programme global, soit d'un résultat qui sera communiqué, après édition et mise en forme, à l'utilisateur réseau.

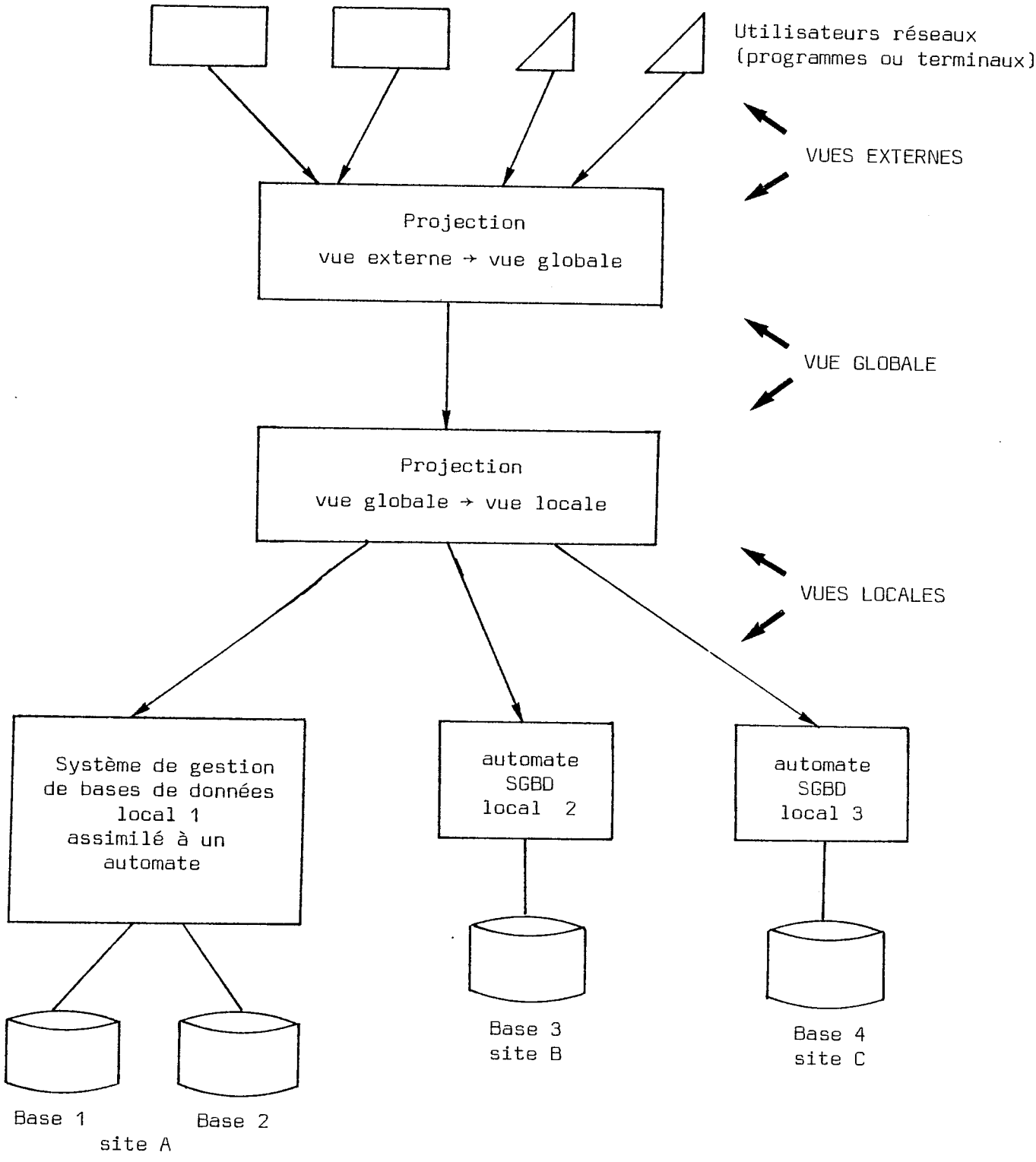


Figure 6.5

Architecture d'un système de coopération
et utilisation des vues

Cette architecture suppose la résolution de quelques problèmes nouveaux :

- l'exécution du programme global suppose l'existance d'une machine réseau capable de faire des transferts de sous-programmes (d'algorithmes) d'un site à un autre, capable de synchroniser au travers du réseau plusieurs processeurs. Une machine réseau telle que S-IGOR répond à ces besoins [DAN77],

- les échanges de résultats entre processeurs hétérogènes posent des problèmes de conversions et de structures standards de données dont les définitions ne sont pas encore acquises dans les réseaux hétérogènes d'ordinateurs,

- la gestion du parallélisme et la résolution des problèmes d'optimisation que posent le travail possible en parallèle des automates locaux n'ont que peu avancé dans les systèmes multi-processeurs,

- une mention particulière doit être faite sur la cohérence des données manipulées : l'hypothèse de base est que chaque SGBD local assure la cohérence des données qu'il gère ; d'autre part, au niveau global, on ne peut assurer une cohérence globale des données appartenant aux différents SGBD locaux. L'utilisateur peut donc avoir à connaître des incohérences, au niveau global, résultant de modifications faites de façon non synchronisée sur plusieurs SGBD locaux. Ceci fixe une limite du système de coopération si on maintient des accès autonomes aux SGBD locaux. Une cohérence au niveau global n'est possible que si tout accès aux SGBD est fait dans les termes de la vue globale ; les ressources SGBD locaux ne pouvant être requises qu'en respectant une stratégie globale d'allocation de ressources [MAH72].

6.2.2.4. Localisation des fichiers et des fonctions :

Le processeur vue globale (cf. figure 6.6) gère une base propre à la vue globale qui comprend la vue globale elle-même et les objets globaux. Elle est assimilable à une base locale. Cette base peut être soit unique, soit à copies multiples ; le choix est dicté par des critères classiques de localisation des utilisateurs réseaux, des coûts de transmissions...

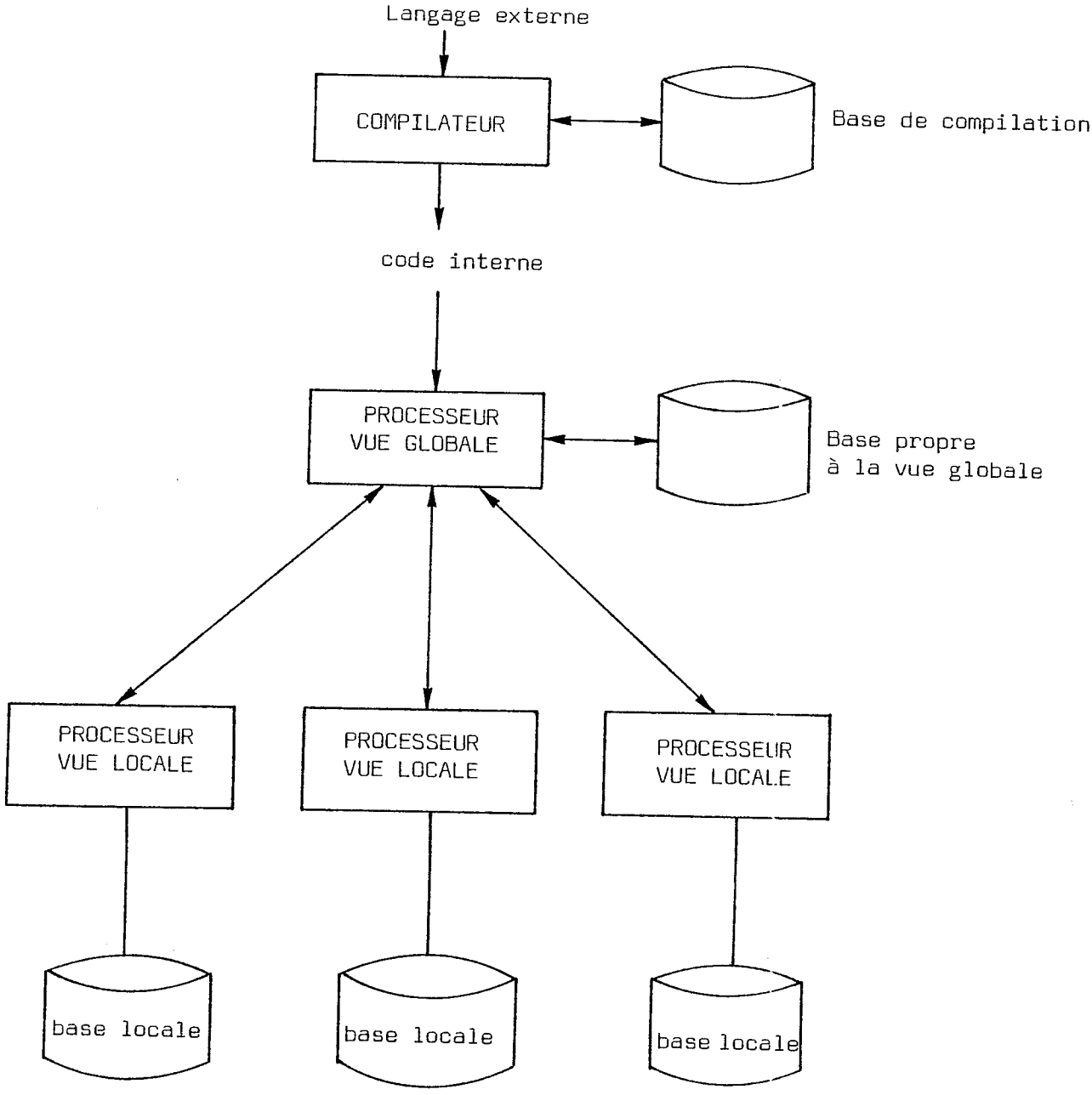


Figure 6.6

Localisation des fichiers et des fonctions

Le maintien de la cohérence de cette base, lorsqu'elle est à copies multiples, permet de fournir un service complémentaire à l'ensemble des utilisateurs réseau de cette vue globale : il s'agit de la cohérence des données traitées grâce au système de coopération.

On peut noter ici que plusieurs vues globales, aux finalités différentes pour les utilisateurs, peuvent coexister dans le réseau : cela peut constituer un moyen d'offrir des services spécifiques aux utilisateurs réseau en prenant en compte des critères de confidentialité et de pouvoir de modification des données, par exemple.

La base de compilation contient les informations nécessaires à la compilation d'une requête du langage externe. Elle doit être sur le site du compilateur et sera à copies multiples dès qu'il y aura plus d'un site d'interrogation.

Pour das raisons de cohérence des informations concernant les bases locales, on peut envisager de répartir la base de compilation sur les sites de bases locales.

Cette répartition laisse à la charge d'une base locale la gestion de la partie de la base de compilation qui l'intéresse. Il reste sur le site de compilation une base racine, qui peut, de la même façon que précédemment, être à copies multiples.

6.3. Les architectures réseau :

Les études de cas que nous avons eu l'occasion de développer dans les chapitres précédents sont le plus souvent des systèmes informatiques répartis expérimentaux, ayant tiré parti des outils de communication mis à disposition par des réseaux généraux d'ordinateurs tels que ARPA, CYCLADES. Pour conserver la généralité de ces réseaux (une de leur caractéristique essentielle), ces applications réparties ont tenu compte des matériels et, le plus souvent, des systèmes d'exploitation existant.

Cette démarche est propice au démarrage rapide d'applications sur les réseaux. Dans Cyclades, la réalisation des stations de transport a ainsi été possible en limitant les modifications de système à l'incorporation d'outils de communication entre tâches abonnées du réseau. Le bilan de réalisations menées à bien sur des matériels différents [ANS76], [AND76], [FOU76], [QUI76] font apparaître que, faute de systèmes orientés réseau, l'insertion de stations de transport dans les systèmes hôtes ne permet pas une pleine utilisation des possibilités de communication réseau. Par exemple, accès dégradé avec plus de cinq abonnés actifs [ANS76], performance globale du système abaissés [FOU76], accès au réseau réservé à des abonnés privilégiés [AND76].

Ces inadéquations ont favorisé d'une part l'écriture de systèmes organisés autour d'architectures nouvelles, d'autre part la commercialisation par les constructeurs d'architecture réseau devant couvrir une large gamme d'application réparties.
Parmi les systèmes organisés autour d'architectures nouvelles, nous pouvons citer :

- le système RSEXEC [THO73], système distribué attaché aux hôtes TENEX (DEC PDP10) du réseau ARPA. RSEXEC est un système d'exploitation partagé ; dupliqué sur chaque hôte TENEX, il permet la mise en commun des ressources, leur partage entre hôte TENEX, le réseau étant transparent à l'utilisateur.

- le projet ARAMIS, de l'Université Paul Sabatier à Toulouse, bâti autour d'un réseau de mini-ordinateurs homogène (Mitra 15 CII) ; sa vocation est la sûreté de fonctionnement et la mise en service d'un

système de gestion de bases de données SYMBAD (de CII) dans une version répartie.

- le projet ESE, de l'Ecole Supérieure d'Electricité, composé de mini-ordinateurs hétérogènes ; les ordinateurs sont classés en deux catégories : traitement et systèmes. Les processeurs de traitement sont banalisés et le système assure un partage de charge invisable à l'utilisateur.

- le projet de centre de calcul de l'Ecole des Mines de Saint-Etienne [CHA76] basé sur un frontal et un ensemble de mini-ordinateurs ayant des vocations spécifiques. Un interpréteur réparti de langage de commande est développé avec contrôle centralisé dans le frontal.

- le projet de Multicalculateur de l'Université de Rennes, basé sur un bus-boîte à lettres (boîte à lettres cablée) permettant la communication entre des processeurs spécialisés (calculateur associatif, processeur fichier, mini-machine langage...). Ce projet développe plusieurs aspects des multi-mini(micro)processeurs sans mémoire commune (mémoire paginée, processeur fichier, aspects langage) [BOU77].

- le projet de système transactionnel réparti de manière fonctionnelle est organisé en grappes de processeurs communicants : différents niveau de liaisons sont prévus (rapides dans une grappe, plus lentes entre deux grappes locales, de type télé-informatique entre processeurs reliés dans un cadre régional ou national) [ANH77]. Un processeur base de données y est plus particulièrement développé.

- le réseau homogène du CERN [AFG77] constitue une architecture en étoile, l'ordinateur central assurant une fonction de communication par message entre les ordinateurs de traitement sur lesquels des tâches s'exécutent en parallèle avec possibilité de synchronisation, de communication. Des outils sont fournis au programmeur pour faciliter l'exécution des parties de son programme dans différents ordinateurs.

Les différents systèmes répartis que nous venons de mentionner ont des objectifs différents, mais sont tous organisés autour d'un outil de communication (réseau de commutation, bus...) qui est exploité par différents processeurs (en général des minis ou micros ordinateurs).

Pour tirer parti de ces nouveaux outils de communication, les constructeurs offrent des architectures de systèmes avec des méthodes d'accès permettant la communication entre processeurs distants (terminal, programme).

Il s'agit pour la CII-HB de son architecture DSE (Distributed System Equipement) et de sa méthode d'accès VCAM (Virtual Communication Access Method) [CHB77]. IBM propose SNA (System Network Architecture) et une méthode d'accès VTAM (Virtual Terminal Access Method) [GRB75], [FAY76], [MOU77], [ALB76]. Pour Digital Equipment : DNA (Digital Network Architecture) et une méthode d'accès aux données DAP (Data Access Protocol) [CON76], [PAS77], [TEI75], [WEC76].

DAP correspond à un niveau de méthodes d'accès fichier, offrant à l'utilisateur un ensemble de primitives équivalent à celui de MADRE (cf. chapitre 3.2).

Ces différentes architectures sont bâties autour du concept de *frontal* qui sépare, au niveau du matériel, les processeurs de traitement d'autres processeurs : gestion de terminaux, gestion d'ordinateurs satellites concentrateurs ou de pré-traitement, liaisons avec d'autres frontaux, connexion avec des commutateurs de paquets (cas de la CII-HB).

Le frontal apparaît comme la première ressource mise en commun entre plusieurs centres de traitement. Il apporte à l'utilisateur une plus grande souplesse d'accès aux services et aux données, une indépendance accrue entre l'application et les terminaux ou les réseaux, une souplesse de configuration.

Un des objectifs des réseaux informatiques est le partage de ressources : le frontal apparaît comme la première ressource logique et physique partagée ; pour l'utilisateur, elle permet un accès banalisé à l'ensemble des ressources locales et distantes disponibles.
Au paragraphe suivant, nous présentons un deuxième type de ressources qui apparaît dans les architectures réparties : la ressource mémoire de stockage de données ou machine de données.

6.4. Les machines de données :

Nous appelons machines de données (ou machines de stockage de données), un processeur de stockage et de gestion de données sur mémoire secondaire utilisable dans le cadre d'un réseau d'ordinateurs comme une ressource commune et partageable.

La définition d'un tel service est rendu possible par des progrès technologiques qui, par exemple, permettent l'adressage cablée dans une mémoire virtuelle de taille illimitée. Le dispositif développé par la SEMS [TEL01] permet un adressage sur 2^{32} mots de 16 bits en utilisant une technique de repliement d'un espace virtuel sur l'espace réel sans perte de place en mémoire réelle (série Solar). Dans [BAL75], [ANH77] des processeurs bases de données sont développés en considérant SOCRATE comme une méthode d'accès.

D'autres produits, comme la librairie IBM 3850, offrent une puissance de gestion et de stockage de données qui ne peut être rentabilisée qu'avec des processeurs de traitement de données en grand nombre. Ces évolutions tendent à la définition de machines de données, entités autonomes conçues comme des services réseau. Ces machines de données se présentent comme des serveurs pour plusieurs processeurs de traitement soit en local, soit sur un réseau via un frontal qui assure le rôle d'un processeur réseau pour la machine de données (cf. figure 6.7).

Si la machine de données ne connaît pas la structure de données qu'elle manipule, c'est une *machine fichier* dont l'interface d'entrée est fait d'un jeu de primitives : par exemple [BOU77] ; la machine permet à plusieurs utilisateurs de créer et de stocker des collections de données et des procédures d'accès à ces données et de demander l'exécution de ces procédures par la machine fichier, d'où les primitives :

- *créer* (nom fichier, procédure 1,... procédure n)
- *modifier* (nom fichier, procédures à modifier)
- *exécuter* (nom fichier, procédures d'accès)
- *tuer* (nom fichier).

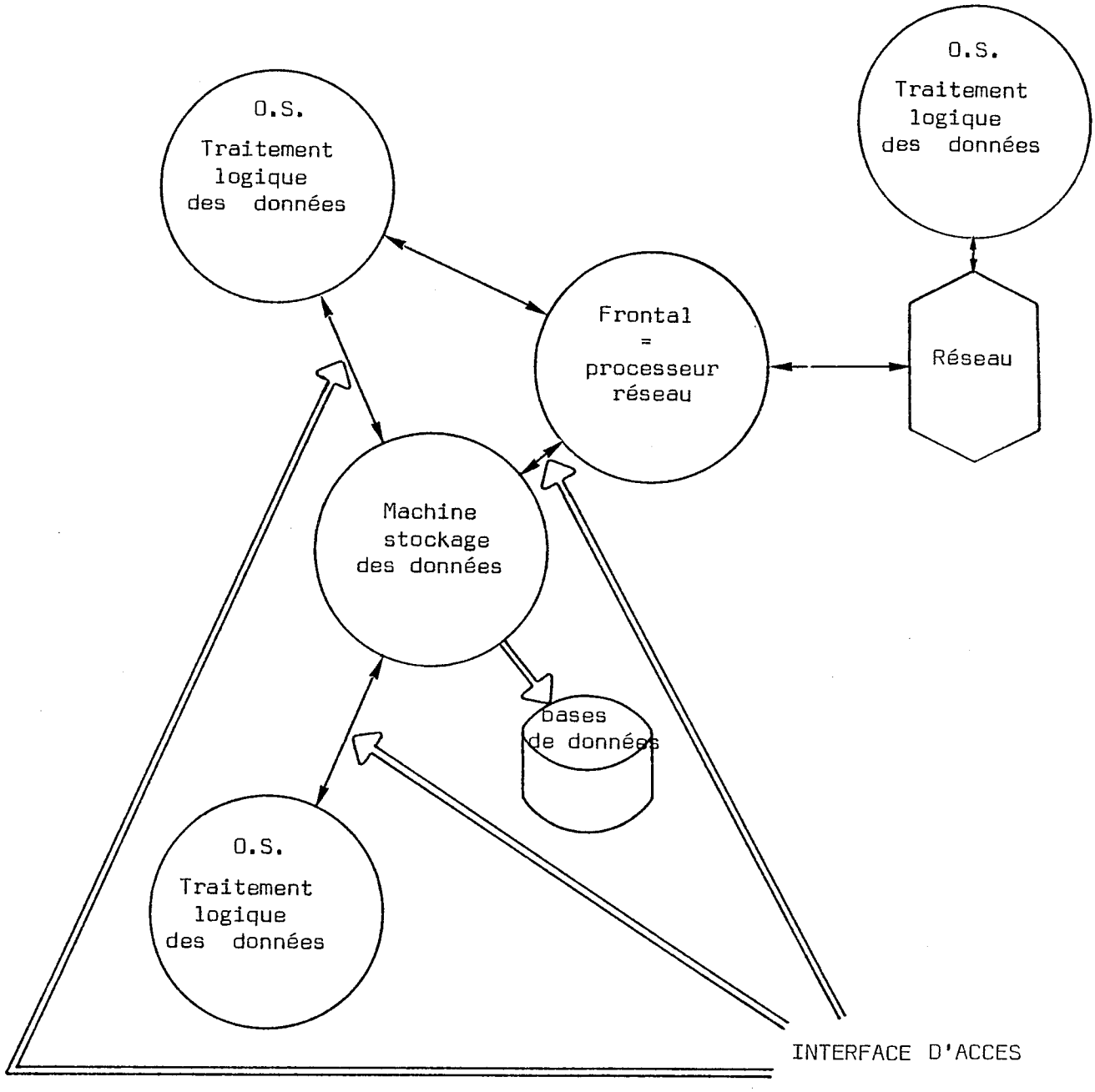


Figure 6.7

Interface d'accès à une machine stockage de données

Si la machine de données connaît la structure de données qu'elle manipule, c'est une *machine base de données* dont l'interface d'entrée est du type méthode d'accès bases de données (cf. chapitre 3.3).

Les machines de données apparaissent comme un des développements importants, aussi bien pour les services des réseaux généraux d'ordinateurs que pour les centres de calcul répartis construits comme des centres multi-processeurs de traitement.

7. CONCLUSION

Nous avons essayé de présenter nos recherches sur "le traitement distribué d'informations réparties dans les réseaux d'ordinateurs" comme une contribution aux logiciels des réseaux informatiques.

Les éléments de base de l'informatique répartie connaissent une transformation profonde qui modifie la conception de l'informatique autour de son outil de travail : l'ordinateur. La télé-informatique, les systèmes multiprocesseurs, la micro-informatique, les réseaux de commutation de données, les systèmes de gestion de bases de données sont les éléments de base de l'informatique répartie qui posent de nouveaux problèmes pour les logiciels : répartition du contrôle, méthodes d'accès réseau, duplication de l'information, parallélisme d'exécution, coopération de systèmes, hétérogénéité.

Nous avons pu dégager quelques solutions en conclusion de nos travaux et d'autres projets réalisés pour les logiciels des réseaux informatiques, les architectures de systèmes multiprocesseurs. La répartition du contrôle est une notion nouvelle pour les applications réparties qui doivent fonctionner même en mode dégradé sans être dans un état d'erreur ; les méthodes d'accès réseau sont les outils de base pour les programmeurs de logiciels réseau ; la duplication de l'information apparaît comme indispensable pour le traitement des fichiers catalogues dans les réseaux ; dans les réseaux généraux d'ordinateurs, les systèmes de coopération sont une voie possible pour fournir des services aux utilisateurs des réseaux sans modifier les habitudes de travail et les prérogatives des utilisateurs de chaque site ; prendre en compte des logiciels et des matériels hétérogènes dans un réseau, c'est à l'évidence le seul moyen pour ne pas renoncer à la recherche dans ce domaine... en évitant de s'en remettre aux annonces commerciales des plus gros constructeurs.

8. BIBLIOGRAPHIE

ABO75 C. ABONNEAU
L'utilisation des bases de données en France.
Journée d'étude sur les systèmes de gestion de bases de données. Montpellier, mai 1975.

ABR72 J.R. ABRIAL and C°
Projet SOCRATE.
Spécifications version 3. Université de Grenoble. Septembre 1972.

AB272 J.R. ABRIAL
Projet SOCRATE.
Cours avancé CEE sur l'architecture des systèmes informatiques. L'Alpe d'Huez, décembre 1972.

ABR73 J.R. ABRIAL
Data Semantics.
IFIP Conference. Cargèse, Corse, 1973.

ADI77 M. ADIBA, C. DELOBEL
The cooperation problem between different data base management systems.
IFIP-TC2, Working Conference, Nice, janvier 1977.

ALB76 H.R. ALBRECHT, K.D. RYDER
The Virtual Telecommunication access method.
A system network architecture perspective. IBM Systems Journal,15.1.197

AND76 E. ANDRE, G. BOGO, P. DECITRE, R. GARDIEN, P.A. PAYS
Specifications de définition de Cycliris.
Rapport du C.S. CII-HB Grenoble, avril 1976.

AND77 E. ANDRE
ST3 Cyclades
Note de travail. C.S. CII-HB Grenoble 1977.

AND78 E. ANDRE, P. DECITRE
On providing distributed application programmers with control over synchronisation.
CII-HB Grenoble, Synposium Protocoles de Réseaux d'Ordinateurs.

ANS 76 J.P. ANSART
Système Interactif dans un environnement réseau.
Connexion IBM-CP67 au réseau Cyclades.
Thèse Docteur Ingénieur Grenoble, février 1976.

ANX75 ANSI/X3/SPARC
Interim report on data base management systems.
150/TC 97/SC5
Washington, mai 1975.

BAC76 A. BACHE, L. GUILLOU, H. LAYEC, B. LORIG, Y. MATRAS
RCP, The Experimental Packet-Switched Data Transmission Service of the French PTT, History, Connections, Control.
3rd Conference Computer Communication, Toronto, Août 1976.

BAH72 C.W. BACHMANN
The evolution of storage structure.
Honeywell Informations Systems Communications, ACM, juillet 1972.

BAE72 W. BAECHI, A. DUENKI, P. SCHICKER
File Transfers.
ETH Computer Science, Zurich, mars 1974.

BAL75 R. BALTER, J. CHARLET, S. GUIBOUD-RIBAUD, H. SMIT
Processeur Socrate
Note informelle, équipe architecture, C.S. CII, 1975, Grenoble.

BAR76 D.L.A. BARBER
An European Informatics Network Achievement and Prospects
3rd Conference Computer Communications, Toronto, Août 1976.

BHU72 A.K. BHUSHAN
The file transfer protocol
Nic note 10596, 1972.

BLA75 G. BLAIN, G. DELPUECH, G. GARDARIN, D. KANDEL, G. SCHLATTER
ADP, manuel de présentation du système
Institut de Programmation, Paris, décembre 1975.

BOC75 G.V. BOCHMANN
Logical Verification and Implementation Protocols
Pub#190, Université de Montréal, Département d'Informatique, 1975.

BOS77 P. BOSC, A. CHAUFFAUT, J. LE PALMEC, J.M. VILLARD
Projet FRERES, interrogation de fichiers répartis sur un réseau de calculateurs hétérogènes.
Publication IRISA n° 60, janvier 1977.

BOT77 P. BOSC, A. CHAUFFAUT, J. LE PALMEC, R. TREPOS, J.M. VILLARD
Projet SIRIUS, fonctions et architecture d'un SGBDR
Rennes, juillet 1977.

BOU77 G. BOULAYE, B. CANET & all
MURENE (Multicalculateur de l'Université de Rennes)
RAIRO Informatique, Vol11, n° 3, 1977.

CAN74 R.H. CANADAY, R.D. HARRISON, E.L. IVIE, J.L. RYDER, L.A. WEHR
A Back-end Computer for data-base management
Communications ACM, octobre 1974.

CAS72 R.G. CASEY
Allocation of copies of a file in information networks
Proceedings AFIPS, SJCC, vol 40, mai 1972.

CHA76 J.F. CHAMBON, S. GUIBOUD-RIBAUD, B. LE BIHAN, Y. TOSAN
Intérêt et faisabilité d'un centre de calcul fondé sur un réseau de mini-ordinateurs.
Note technique, GENE01.
Département d'informatique, Ecole des Mines, Saint Etienne, Octobre 1976.

CHA77 G.A. CHAMPINE
Six approach to distributed data bases
Datamation, mai 1977.

CHB77 CII-HB
Annonce des produits réseaux IRIS80-C10 et de l'architecture DSE,
Documents CII, 1er semestre 1977.

DAT75 A. DANTHINE
Host-Host Protocoles and Hierarchy
IFIP-IIASA, Interconnecting computer networks, Luxembourg, Austria, septembre 1975

DAV76 G.M.P. DAVIES
Euronet Projects
Communication Networks 1976

DAY73 J. DAY
A proposed file access protocol specification
University of Illinois, juillet 1973

DCA73 R. DE CALUWE
Etude du langage de commande et de contrôle pour le réseau d'ordinateurs SOC
Thèse de Troisième Cycle, Grenoble, septembre 1973

DEC77 P. DECITRE
Accès depuis PL/1 à la station de transport ST2 sous SIRIS8
Note technique n° 9, Projet Polyphème, Grenoble, août 1977

DEP76 M.E. DEPPY, J.P. FRY
Distributed data bases
A summary of research, Computer Network 1 (1976), 130-138

DIJ74 E.W. DIJKSTRA
Self-stabilizing Systems in spite of Distributed Control
C. ACM, novembre 1974, volume 17

DMS77 M. DEMUYNCK, P. MOULIN, S. VINSON
La portabilité des programmes utilisant un système de gestion de bases de données
Service informatique EDF, HI-2431/04, mai 1977

DUB76 DUBOIS, LEVANTINH
Madre ST2 Cyclades
Projet de Troisième Année ENSIMAG, Grenoble, juin 1976

CHG76 E. CHANG
A distributed medical data base network software design
Université de Waterloo, Computer Networks n° 1, 1976.

CHE75 G. CHESSON
The network Unix system
Proceedings of the 5th symposium on operating systems principles, University of Texas, Austin, novembre 1975.

CHU68 W.W. CHU
Optimal file allocation in a multicomputer information system
Proceeding IFIP, Edinburg, novembre 1968.

CHU77 W.W. CHU
Performance of file directory systems for data bases in star and distributed networks
2nd workshop Berkeley, distributed data management and computer networks, mai 1977.

CHP74 J.C. CHUPIN
Control concepts of a logical network machine for data banks,
IFIP Stockholm, août 1974.

CHP77 J.C. CHUPIN
Répartition d'applications et de bases de données sur un réseau général d'ordinateurs, Thèse, Grenoble, octobre 1977.

CII01 CII SGT Siris 2/3
Système de gestion de transmissions

CII02 CII Mistral
Recherche documentaire
4017 P1/F2, 1975

CII03 CII SGF Siris 7/8
Système de gestion de fichiers
3704 E2/F2

CII04 CII SGF Siris 2/3
Système de gestion de fichiers
3073 E5/F2

CII05 CII LPS, MacroLPS
Langage de programmation système
4042 E2/F2, 1973

CII06 CII LP70
Langage de programmation Siris 7/8
4069 E1/F2, 1973

CII07 CII
Manuel Socrate
4338 E/Fr, juillet 1973.

CLI76 W.W. CLIPSHAM
Data network overview
3rd ICCC Toronto, août 1976

COM75 D. COMTE
Techniques de communication et de synchronisation entre programmes répartis sur le réseau Cyclades
Thèse Docteur Ingénieur, Toulouse, janvier 1975.

CON66 CONTROL DATA
CDC6600 Computer System
Scope reference manual

COW76 G. CONANT, S. WECKER
DNA : An architecture for heterogeneous computer network
Proceedings of the 3rd ICCC, Toronto, mai 1976

CRO75 CROCUS, ouvrage collectif
Système d'exploitation des ordinateurs
Dunod 1975.

DAN75 Ng.X. DANG, R. FOURNIER, V. QUINT
SYNCOP implementation sous CP67
Rapport ENSIMAG, septembre 1975

DAN76 Ng. X. DANG
Implementation de la station de transport ST2 sous le système CP67.
Rapport ENSIMAG, janvier 1976.

DAN77 Ng.X. DANG, G. SERGEANT
System and portable language intended for distributed and heterogeneous network applications
Rapport de recherche ENSIMAG n° 78, juin 1977.

DUM74 J. DU MASLE, M.N. FARZA, G. SERGEANT
Proposed organization of an interpreter intended for the implementation of high level procedures on a computer language
IFIP Working Conference on Command Language, Sweden, juillet 1974.

ELI73 M. ELIE, H. ZIMMERMANN
Vers une approche systématique des protocoles sur un réseau d'ordinateurs, application au réseau Cyclades
Congrès AFCET, Rennes
Novembre 1973

ELI74 M. ELIE, H. ZIMMERMANN
Transport Protocol, Standard host-host protocol for heterogeneous networks
Réseau Cyclades SCH519.1, avril 1974

ELL77 C.A. ELLIS
A robust algorithm for updating duplicate data bases
2nd Workshop Berkeley "Distributed Data Management and Computer Networks", mai 1977.

ELS77 C.A. ELLIS
Consistency and Correctness of duplacate database systems
6th ACM Symposium on operating systems principles, novembre 1977, pages 67 à 84.

EPS74 EPSS liaison group
Report of the higher level protocol working group
HLP/CP, août 1974

ESW74 K.P. ESWARAN
Placement of records in a file and file allocation in a computer network
IFIP Congress, Stockholm, août 1974

FAR72 D.J. FARBER, F. HEINRICH
The structure of a distributed computer system
The distributed file system, 1st ICCC, Washington, octobre 1972.

FAY76 J.H. Mc FAYDEN
Systems network architecture
An overview, IBM's Systems Journal n° 1, 1976.

FER77 J. FERRIE
Désignation dans les systèmes répartis et les réseaux
Document de travail, Groupe Systèmes Répartis, avril 1977.

FOU76 R. FOURNIER
Le traitement par lots dans un réseau hétérogène. Implémentation du serveur OS/MVT sur IBM 360/67 pour le réseau Cyclades.
Thèse Docteur-Ingénieur, Grenoble, décembre 1976.

GAR76 G. GARDARIN, R. GOMEZ, M. JOUVE, C. PARENT, S. SPACCAPIETRA
Architecture des systèmes de gestion de bases de données réparties
Institut de Programmation, séminaire IRIA, Lans-en-Vercors, mars 1976.

GEP76 H.L. GEPNER
Comparing Data Communications Monitors
Datamation vol. 21, avril 1976.

GIA74 Ng GIA TOAN
Fichiers réseau, protocoles fichiers
DEA ENSIMAG, Grenoble, juin 1974.

GIE76 M. GIEN
Functional specifications of a reduced file management protocol
Note Technique Cyclades IRIA, février 1976.

GIE72 M. GIEN
Implantation du macro-processeur STAGE2 sur CII 10070
Centre Scientifique CII, Grenoble, mars 1972.

GIE77 M. GIEN
Transfert de fichiers sur le réseau Cyclades
Rapport Cyclades DAT520, juillet 1977.

GRA73 J.L. GRANGE, L. POUZIN
Cigale, la machine de commutation de paquets du réseau Cyclades
MIT536, Congrès AFCET, novembre 1973.

GRB75 J.P. GRAY, C.R. BLAIR
IBM's system network architecture
Datamotion, vol. 21, issue 4, avril 1975.

GUI76 GUILBERT, MACCHI éditeurs (ouvrage collectif)
Séminaire de télé-informatique
IRIA - AFCET - CNET - IP, Bonas, Gers, septembre 1976.

HAB72 A.N. HABERMANN
Synchronization of Communicating Process
C. ACM 15.3, mars 1972

HEA73 F.E. HEART, S.M. ORNSTEIN, W.R. CROWTER, W.B. BARKER
A new mini-computer/multiprocessor for the ARPA network (Pluribus)
AFIPS Conference 1973, vol. 42.

HOA74 HOARE
Monitors : on operating system structuring concept
C. ACM 1974, 17,10.

HOL75 C.R. HOLLANDER
Construction of primitives for a multiprocess environment
Note Technique 63, Digital Systems Laboratory, Stanford University, août 1975.

HOR73 E. HOLLER (Kenforchungozentrum Karlhuruhe)
Files in computer networks
European workshop on computer networks, Arles, avril-mai 1973.

HOT72 R. HOLT
Some deadlock properties of computer systems
Computing surveys ACM, vol. 4, n° 3, décembre 1972.

IBM01 IBM
PL/1 Multitasking

IBM02 IBM
Supervisor and data management services OS360

ICH74 J.D. ICHBIAH, J.P. RISSEN, J.C. HELIARD, P. COUSOT
The system implementation language LIS
Reference manual, CII-Louveciennes, décembre 1974.

IRI73 IRIA
Présentation du réseau Cyclades
Novembre 1973

IRI76 IRIA
Bases de données réparties
Séminaire avant projet pilote SIRIUS, Lans-en-Vercors, mars 1976.

IRI77 IRIA
Le réseau Cyclades, annuaire des ressources disponibles
Version préliminaire 1977.

KAI77 C. KAISER
Les Bus
Document de travail, Groupe Systèmes répartis, avril 1977.

KAL74 H. KARL
The distributed data bases of the information system of the German police (INPOL)
European computer workshop, Darmstadt, octobre 1974

KAM77 A. KARMOUCH
Définition du projet SYDRE et choix du système documentaire à répartir
Document interne 77921, Université Toulouse (USST), juin 1977.

KER74 de KERGOMMOUX, M. NOEL
Station de transport ST1 Cyclades Siris 2/3
Projet 3ième année ENSIMAG, Grenoble, juin 1974.

KRA74 H. KRAYL, C. UNGER, T. WELLER
Portability of JCL Programs
IFIP Working Conference on command languages, Lund, Sweden, juillet 1974.

KUC68 D.J. KUCH
ILLIAC4 Software and Application Programming
IEEE TC, C17.8, août 1968.

LAM75 C. et O. LAMBERT
Verrouillage et partage d'informations au sein d'une méthode d'accès réseau
Projet de 3ième année ENSIMAG, Grenoble, juin 1975.

LEB76 J. LE BIHAN
La répartition des données dans les réseaux d'ordinateurs
Congrès AFCET, Gif sur Yvette, novembre 1976.

LEL77 G. LE LANN
Distributed systems. Towards a formal approach
Congrès IFIP77, Toronto, août 1977

LEW75 LEWIN et MORGAN
Optimizing distributed data base. A framework for research
ACP volume 14, janvier 1975.

LOT77 C. LOTHORE
Interface PL/1-SYNCOP
Manuel d'utilisation, ENSIMAG, juin 1977.

LUC75 M. LUCAS
Primitives de synchronisation pour langages de haut niveau
Séminaire de programmation, Grenoble, décembre 1975.

MAH72 R. MAHL, S. KAMRAR
Allocation de ressources dans un réseau d'ordinateurs
Workshop IRIA/ACM, Réseaux d'ordinateurs, Rocquencourt, mars 1972.

MAN75 E. MANNING, R. PEEBLES
A computer architecture for large (distributed) data bases
Proc. VLDB, Fromingham, Massachussetts, USA, septembre 1975.

MAR76 S. MAHMOUD, J.S. RIORDON
Protocol considerations for software controlled access methods in distributed data bases
Synposium on computer performance, Harward University, ACM/IFIP, Mars 1976.

MAU76 C. MAUGER
Une approche à la transparence d'un réseau hétérogène de calculateurs. Conception et réalisation d'une station de transport Cyclades
Rapport de Recherche, Lyon, Février 1976.

MAL71 P. MAURICE, M. MALAGARDIS
Portabilité, implémentation de processus portables
Séminaire IRIA, Structure et Programmation des Calculateurs, Novembre 1971.

MAZ76 G. MAZARE
MCS : a symetric multi-microprocessor system
Euromicro symposium, Venise, octobre 1976.

MAZ77 G. MAZARE
A few examples of how to use a symmetrical multi-micro processor
4th annual symposium on computer architecture, mars 1977.

MAZ78 G. MAZARE
Les microprocesseurs, introduction à l'usage des informaticiens
Document de travail, Groupe Système Répartis, avril 1977.

MET73 J.P. METZGER
Socrate et Réseaux d'ordinateurs
Note Technique Socrate, ENSIMAG, mars 1973.

MEY76 D. MEYER
Objets dupliqués sur un réseau d'ordinateurs
Séminaire IRIA, Lans-en-Vercors, mars 1976.

MIL75 D.L. MILLS
Dynamic file access in a distributed computer network
octobre 1975, TR415, University of Maryland, USA.

MIL76 D.L. MILLS
An overview of the Distributed Computer Network
Octobre 1975, TR413, University of Maryland, USA.

MIR76 S. MIRANDA
Systèmes d'accès interactif à des fichiers répartis sur un réseau d'ordinateurs
Thèse de Troisième Cycle , Toulouse, 1976

MOR77 H.L. MORGAN, K.D. LEVIN
Optimal program and data locations in computer networks
C. ACM, mai 1977, vol. 20, n° 5

MOS77 P. MORRIS, D. SASALOWICZ
Managing network access to a distributed data base
Stanford University, mai 1977.

MOS74 D.J. MOSS
The control data computer communication network
European computer workshop, darmstadt, octobre 1974.

MOU77 P.D. MOULTON, R.C. SANDER
Another look at SNA
Datamation, mars 1977.

MOV77 J. MOSSIERE, M. TCHUENTE, J.P. VERJUS
Sur l'exclusion mutuelle dans les réseaux informatiques
IRISA Publication interne n° 75, 1977, ENSIMAG rapport de recherche.

MUL75 ALVIN P. MULLERY
The distributed control of multiple copies of data
Computer sciences department IBM, T.J. Watson R.C., août 1975.

NEG76 R. NEGARET
Etude de l'allocation de ressources dans les systèmes informatiques répartis
Thèse de Troisième Cycle , Rennes, décembre 1976.

NEI73 N. NEIGUS
File transfer protocol
Network working group, BBN net, juillet 1973.

PAP74 Z. PAPACHRISTODOULOU
Le système Telcom
Thèse de Docteur-Ingénieur, Grenoble, 1974.

PAS77 J.J. PASSAFIUME, S. WECKER
Distributed file access in Decnet
Digital Equipment Corporation, Maynard, Massachussetts, mai 1977.

POO70 P.C. POOLE, W.M. WHITE
The STAGE2 macro-processor
User reference manual, University of Colorado, juillet 1970.

POU75 L. POUZIN
The Cyclades network, present state and development trends
Res 505.1 Cyclades, IEEE Computer networks symposium washington, juin 1975.

POU77 L. POUZIN
Les Réseaux Informatiques
Séminaire d'informatique SI n° 01, Laboratoire d'Informatique, Grenoble, mars 1977.

QUI74 V. QUINT
Traducteur Fanny en assembleur Siemens 4004, ST1 Cyclades
Rapport IRIA/SOGETI, juin 1974.

QUI76 V. QUINT
Protection logicielle contre les erreurs dans un réseau d'ordinateur hétérogènes. Application au 360/67 du réseau cyclades.
Thèse de Docteur-Ingénieur
Décembre 1976.

RAY73 J. RAYMOND
NJCL, un langage de commandes pour réseau d'ordinateurs
Contrat CRI, Grenoble, juillet 1973.

REC76 A. RECOQUE
Architecture à processeurs composants
Congrès AFCET, novembre 1976.

RIC76 H. RICHY, D. PORTAL
Réalisation d'un concentrateur multi-connexions dit "intelligent"
Rapport IRIA, séminaire IRIA, Lans-en-Vercors, mars 1976.

RIT74 D. RITCHIE, K. THOMPSON
The UNIX time-sharing System
C. ACM, juillet 1974.

ROB71 L.G. ROBERTS, B.D. WESSLER
The ARPA network
ARPA, Washington D.C., nic 7750, mai 1971.

ROC73 A. ROCHFELD
Réalisation d'une banque de données virtuelle Socrate
Rapport de la direction scientifique de la SEMA, Metra International, mai 1973.

SEN76 M.E. SENKO
DIAM as a detailed example of the ANSISPARC architecture
IFIP TC2 working Conference, Freudenstadt, janvier 1976.

SER76 D. SERAIN
Mécanismes de description et d'évolution dans une structure de réseaux de processus s'exécutant sur une machine multiprocesseur
Thèse de docteur-ingénieur, Grenoble, juin 1976.

SER75 G. SERGEANT
SOCYCRATE
Note Technique, Centre Scientifique CII, Grenoble, mai 1975.

SER74 G. SERGEANT, M.N. FARZA
Machine interprétative pour la mise en oeuvre d'un langage de commande sur le réseau Cyclades
Thèse de Troisième Cycle, Toulouse, octobre 1974.

SFE01 SFENA-DSI
Système d'exploitation METEOR. Moniteur d'exploitation en temps partagé de l'ordoprocesseur.

SFS77 R.J. SWAN, S.H. FULLER, D.P. SIEWOREK
CM* : a modular multimicroprocessor
National Computer Conference 77, Université Carnegie Mellon

SOM75 M. SOMIA BRECHET
Etude du système de contrôle d'un réseau d'ordinateurs distribué et de ses relations avec les systèmes de contrôle locaux
Thèse de Docteur ès Sciences, Paris VI, juin 1975.

TEC01 TECSI SOFTWARE
Task Master, un moniteur de télé-communications IBM 360/67
DOS/OS/VS

TEI75 N.A. TEICHOLTZ
Digital network architecture
Eurocomp, Brunnel University 13.24, septembre 1975.

TEL01 TELEMECANIQUE INFORMATIQUE
Série Solar 16 : Polybus 16 9/16 DD8818F
VSS16 9/75 DD8868E

THO73 R.H. THOMAS
A ressource sharing executive for the ARPANET
AFIPS Conference proceedings, vol. 42, pages 155 à 163, juin 1973.

THO76 R.H. THOMAS
A solution to the upate problem for multiple copy data bases which uses distributed control
Report n° 3340 BBN, juillet 1976.

WAD74 E. WADA
On the possibility of the unification of command and programming languages
University of Tokyo, IFIP Working Conference on Command Languages, Lund, Sweden, juillet 1974.

WEC76 S. WECKER
The design of DEC net. A general purpose network base.
IEEE electro 76, Boston, Massachussets, mai 1976.

WIL77 P. WILMS
Cohérence d'objets dupliqués dans un réseau général d'ordinateurs
DEA Génie Informatique, Grenoble, juin 1977.

WIR71 N. WIRTH
PL360 A programming language for the 360 computers
Stanford University

WIR72 N. WIRTH
The programming language Pascal
Eth Zurich, 1972.

WIR76 N. WIRTH
MODULA : a language for modular multiprogramming
ETH Zurich, mars 1976.

WUL72 W.A. WULF, C.G. BELL
C. MMP a multi-micro-processor
Proceedings AFIPS, 1972.

WUL75 W.A. WULF, R. LEVIN
The Hydra Operating System Manuel
Carnegie Mellon University, Pittsburgh, juin 1975.

ZIM76 H. ZIMMERMANN
High level protocols standardization : technical and political issues
SCH583 Cyclades ICCC 76, Toronto, août 1976.

ZI276 H. ZIMMERMANN
Proposal for a virtual terminal protocol TER533
Cyclades/IRIA, janvier 1976.

ZIM75 H. ZIMMERMANN
Protocoles de communications interordinateurs. Une expérience de réalisation portable
Convention Informatique, Paris, septembre 1975.

ZIM74 H. ZIMMERMANN
Protocoles de communication
Journée "Procédure, Protocoles et langages dans les télécommunications", ACM/AFCET, juin 1974.

9. PUBLICATIONS

a = papier sélectionné à un Congrès
b = papier invité à un Congrès, un Séminaire ou présenté à un jury ad hoc
c = publication technique ou rapport de recherche

[c1] E. ANDRE, M. GIEN, P. GUILLIER, J.P. METZGER, C. de MONTETY, P.A. FAYS, J. SEGUIN,
Projet CRIC, Etude d'un prototype de réseau d'ordinateurs,
Contrat DRME, Janvier 1973.

[c2] J. SEGUIN,
Note sur la méthode d'accès fichier utilisée par SOCRATE SIRIS 7/8 pour ses bases,
Note technique, CS CII Grenoble, Contrat DRME, 30 Juillet 1973.

[c3] M. GIEN & J. SEGUIN,
FANNY, un langage d'écriture de systèmes portables,
Note technique, CS CII Grenoble, Contrat DRME, Novembre 1973.

[b4] P.A. FAYS & J. SEGUIN,
Travail effectué sur SOCRATE SIRIS 7 pour un SOCRATE réseau,
Note technique, CS CII Grenoble, Séminaire Université de Rennes, Juin 1973.

[c5] E. ANDRE, J.C. CHUPIN, J. SEGUIN,
Présentation des recherches 1974 : SOCRATE CYCLADES,
Equipe Réseaux CS CII Grenoble.

[a6] J.C. CHUPIN & J. SEGUIN,
A network direct access method,
Network Conference, Darmstadt, Octobre 1974.

[b7] E. ANDRE, J.C. CHUPIN, J. SEGUIN,
Distributed data base management systems,
EEC Advanced Course, Serre Chevalier, Décembre 1974.

[b8] J.C. CHUPIN & J. SEGUIN,
MADRE, Spécifications de définition et de réalisation,
Publication CS CII Grenoble, Séminaire Université de Lyon, Avril 19

[a9] J.C. CHUPIN & J. SEGUIN,
NDAM : A network direct access method,
Symposium on Computer Architecture, University of Houston,
Janvier 1975.

[c10] J. SEGUIN & G. SERGEANT,
FANNY/SYNCOP : Manuel d'utilisation et spécifications SIRIS 7/8,
Publication CS CII / ENSIMAG, Grenoble, Septembre 1975.

[a11] J.C. CHUPIN, J. SEGUIN, G. SERGEANT,
Distributed applications on heterogeneous networks,
IFIP IAASA Inter-connecting Computer Networks, Lasenburg, Austria,
Septembre 1975.

[b12] M. ADIBA, E. ANDRE, J.C. CHUPIN, P. DECITRE, C. DELOBEL, M. LEONARD
Nguyen GIA TOAN, D. PORTAL, F. REYNAUD, H. RICHY, J. SEGUIN,
G. SERGEANT, A. STIERS,
POLYPHEME, Propositions pour un modèle de répartition et de coopération de bases de données dans un réseau d'ordinateurs,
Rapport de Recherche ENSIMAG, n° 29, Décembre 1975, Séminaire IRIA,
Lans en Vercors, 1-3 Mars 1976.

[a13] J.C. CHUPIN, J. SEGUIN, G. SERGEANT
Distributed applications on heterogeneous networks,
NTG, University of Aachen, 31 mars - 2 avril 1976.

[a14] J. SEGUIN & G. SERGEANT
Interrogation, mise à jour et cohérence d'un fichier à copies multiples réparties sur un réseau d'ordinateurs,
Congrès AFCET 1976, Gif sur Yvette, Novembre 1976.

[b15] J. SEGUIN
Les modèles relationnels de répartition et de coopération de bases de données,
Séminaire Informatique Répartie de Santiago, Université Saint Jacques de Compostelle, Espagne, 13-17 Septembre 1976.

[a16] J.C. CHUPIN, H. RICHY, J. SEGUIN
Data sharing and cooperation between DBMS in heterogeneous computer networks,
AICA 77 Congress, Pise, octobre 1977.

[c17] Ng.X. DANG, V. QUINT, J. SEGUIN, G. SERGEANT
SYNCOP : un sous-système de traitement de processus pour la télé-informatique et les réseaux d'ordinateurs,
Rapport de Recherche ENSIMAG, N° 64, novembre 1976.

[b19] J. SEGUIN, G. SERGEANT, P. WILMS
Cohérence et Gestion d'Objets Dupliqués dans les Systèmes Distribués,
Rapport de recherche ENSIMAG n° 77, mai 1977.

[c20] R. FOURNIER, J. SEGUIN
Le Télé-traitement et les Réseaux d'Ordinateurs au Centre Inter-universitaire de Calcul de Grenoble,
Rapport C.I.C.G., novembre 1977.

dernière page de la thèse

AUTORISATION DE SOUTENANCE

VU les dispositions de l'article 5 de l'arrêté du 16 Avril 1974,

VU les rapports de Messieurs :

- L. BOLLIET, Professeur U.S.M.G.
- Cl. DELOBEL, Professeur U.S.M.G.
- J.P. VERJUS, Professeur à l'Université de RENNES.

Monsieur Jean SEGUIN

est autorisé à présenter une thèse en soutenance pour l'obtention du grade de DOCTEUR d'ETAT ES-SCIENCES.

Le 24 Février 1978

Le Président de l'U.S.M.G.

Le Président de l'I.N.P.G.

Ph. TRAYNARD
Président
de l'Institut National Polytechnique
P.O. le Vice-Président,