Langages et protocoles dans un centre de calcul reparti

Yves Tosan

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT ETIENNE

INSTITUT NATIONAL POLYTECHNIQUE
DE GRENOBLE

N° d'ordre : 9 IS

THESE

présentée par

Yves TOSAN

pour obtenir

le grade de Docteur de 3° cycle

"SYSTEMES ET RESEAUX INFORMATIQUES"

LANGAGES ET PROTOCOLES DANS UN CENTRE DE CALCUL REPARTI

Soutenue à Saint Etienne le 15 Mars 1978 devant la commission d'examen:

MM.	L. BOLLIET	Président
	JF.CHAMBON	
	S. GUIBOUD-RIBAUD	Examinateurs
	J. HEBENSTREIT	

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT ETIENNE

INSTITUT NATIONAL POLYTECHNIQUE
DE GRENOBLE

N° d'ordre : 9 IS

THESE

présentée par

Yves TOSAN

pour obtenir

le grade de Docteur de 3° cycle

"SYSTEMES ET RESEAUX INFORMATIQUES"

LANGAGES ET PROTOCOLES DANS UN CENTRE DE CALCUL REPARTI

Soutenue à Saint Etienne le 15 Mars 1978 devant la commission d'examen:

MM.	L. BOLLIET	Président
	JF.CHAMBON	
	S. GUIBOUD-RIBAUD	Examinateurs
	J. HEBENSTREIT	

ECOLE NATIONALE SUPERIEURE DES MINES DE SAINT.ETIENNE

Directeur : M. G. ARNOUIL
Direction des Recherches : M. P. COUEIGNOUX
Direction des Etudes : M. A. COINDE

PROFESSEURS DE 1ère CATEGORIE

MM. COINDE Alexandre	Gestion
GOUX Claude	Métallurgie
LEVY Jacques	Métallurgie
RIEU Jean	Mécanique-Résistance des Matériaux
SOUSTELLE Michel	Chimie
FORMERY Philippe	Mathématiques Appliquées

PROFESSEURS DE 2ème CATEGORIE

MM. GUIBOUD-RIBAUD Serge	Informatique
LOWYS Jean-Pierre	Physique
TOUCHARD Bernard	Physique Industrielle

PROFESSEURS ASSOCIES DE 2ème CATEGORIE

MM. DAVOINE Philippe	Géologie
PERRIN Michel	Géologie

DIRECTEUR DE RECHERCHE

M. LESBATS Pierre	Métallurgie

MAITRES DE RECHERCHES

MM. BISCONDI Michel	Métallurgie
BOOS Jean-Yves	Métallurgie
COUEIGNOUX Philippe	Informatique
Mlle FOURDEUX Angéline	Métallurgie
MM. LALAUZE René	Chimie
LANCELOT Francis	Chimie
LE COZE Jean	Métallurgie
THEVENOT François	Chimie
TRAN MINH Canh	Chimie

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Président : M. P. TRAYNARD

Vice-Présidents : M. R. PAUTHENET
M. G. LESPINARD

PROFESSEURS TITULAIRES

MM.	BENOIT Jean	Electronique - Automatique
	BESSON Jean	Chimie Minérale
	BLOCH Daniel	Physique du solide - Cristallographie
	BONNETAIN Lucien	Génie Chimique
	BONNIER Etienne	Métallurgie
	BOUDOURIS Georges	Electronique - Automatique
	BRISSONNEAU Pierre	Physique du solide - Cristallographie
	BUYLE-BODIN Maurice	Electronique - Automatique
	COUMES André	Electronique - Automatique
	DURAND Francis	Métallurgie
	FELICI Noël	Electronique - Automatique
	FOULARD Claude	Electronique - Automatique
	LANCIA Roland	Electronique - Automatique
	LONGEQUEUE Jean Pierre	Physique Nucléaire Corpusculaire
	LESPINARD Georges	Mécanique
	MOREAU René	Mécanique
	PARIAUD Jean Charles	Chimie - Physique
	PAUTHENET René	Electronique - Automatique
	PERRET René	Electronique - Automatique
	POLOUJADOFF Michel	Electronique - Automatique
	TRAYNARD Philippe	Chimie - Physique
	VEILLON Gérard	Informatique Fondamentale et Appliquée

PROFESSEURS SANS CHAIRE

MM.	BLIMAN Samuël	Electronique - Automatique
	BOUVARD Maurice	Génie Mécanique
	COHEN Joseph	Electronique - Automatique
	GUYOT Pierre	Métallurgie Physique
	LACOUME Jean Louis	Electronique - Automatique
	JOUBERT Jean Claude	Physique du solide - Cristallographie
	ROBERT André	Chimie Appliquée et des Matériaux
	ROBERT François	Analyse numérique
	ZADWORNY François	Electronique - Automatique

MAITRES DE CONFERENCES

MM.	ANCEAU François	Informatique Fondamentale et Appliquée
	CHARTIER Germain	Electronique - Automatique
	CHIAVERINA Jean	Biologie, biochimie, agronomie
	IVANES Marcel	Electronique - Automatique
	LESIEUR Marcel	Mécanique
	MORET Roger	Physique Nucléaire Corpusculaire
	PIAU Jean Michel	Mécanique
	PIERRARD Jean Marie	Mécanique
	SABONNADIERE Jean Claude	Informatique Fondamentale et Appliquée
Mme	SAUCIER Gabrielle	Informatique Fondamentale et Appliquée
	SOHM Jean Claude	Chimie Physique

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

M.	FRUCHART Robert	Directeur de Recherche
MM.	ANSARA Ibrahim	Maître de Recherche
	BRONOEL Guy	Maître de Recherche
	CARRE René	Maître de Recherche
	DAVID René	Maître de Recherche
	DRIOLE Jean	Maître de Recherche
	KLEITZ Michel	Maître de Recherche
	LANDAU Ioan Doré	Maître de Recherche
	MATHIEU Jean Claude	Maître de Recherche
	MERMET Jean	Maître de Recherche
	MUNIER Jacques	Maître de Recherche

SOMMAIRE

Je tiens à remercier :

Monsieur Louis BOLLIET, Professeur à l'Université Scientifique et Médicale de Grenoble, qui m'a fait l'honneur de présider le jury de cette thèse.

Monsieur Jacques HEBENSTREIT, Professeur à l'Ecole Supérieure d'Electricité, qui a bien voulu accepter de juger mon travail.

Monsieur Serge GUIBOUD-RIBAUD, Directeur du département informatique de l'Ecole des Mines de Saint Etienne, grâce auquel ce projet a pu aboutir.

Monsieur Jean François CHAMBON, Chercher à l'Ecole des Mines de Saint Etienne, qui est l'initiateur de cette étude et qui ne m'a jamais ménagé son aide ni ses conseils.

Les membres de l'équipe du Centre de Calcul Réparti qui ont contribué directement au projet : Mademoiselle Michèle CART et Monsieur Yannis ANGELIDES, ainsi que Monsieur Michel TOURNOIS, Chercheur à l'INSA de Rennes, lui aussi concerné par cette recherche.

Madame LALLICH qui m'a aidé à établir la bibliographie de cette étude, Madame BONNEFOY, Messieurs LOUBET et DARLE qui ont assuré avec beaucoup de soin et de gentillesse la frappe et le tirage de cet ouvrage.

PREMIERE PARTIE :

LE CENTRE DE CALCUL REPARTI

CHAPITRE 1

OBJECTIFS

1.1. - APPROCHE PHYSIQUE DE LA REALISATION DE SERVICES INFORMATIQUES POINT DE VUE DE L'UTILISATEUR

- 1.1.1. - Services informatiques
- 1.1.2. - Environnement limité à un seul ordinateur
- 1.1.3. - Environnement composé de plusieurs ordinateurs
- 1.1.4. - Point de vue de l'utilisateur

1.2. - APPROCHE FONCTIONNELLE

- 1.2.1. - Les différentes fonctions
- 1.2.2. - Cas particulier de la fonction de communication
- 1.2.3. - Exemples de réalisations

1.1. - APPROCHE PHYSIQUE DE LA REALISATION DE SERVICES INFORMATIQUES

POINT DE VUE DE L'UTILISATEUR

1.1.1. - Services informatiques :

Le développement de l'informatique pose de façon nouvelle le problème de l'adaptation des outils fournis par les ordinateurs aux désirs des utilisateurs. A la notion d'outil d'emploi souvent trop technique doit se substituer la notion de service ou application qui permet aux utilisateurs d'ignorer le mode de fonctionnement interne du système auquel ils accèdent.

Les calculateurs offrent un certain nombre de possibilités physiques. Le logiciel développé par programmation donne à ces possibilités une forme plus complète et plus attrayante. A partir d'un ensemble de primitives de bas niveau, les capacités du matériel, on construit un ensemble de primitives de haut niveau. Une application est une structuration d'un sous-ensemble de ces primitives de haut niveau de telle sorte qu'elle réalise une fonction bien précise présentant un interface de communication avec l'extérieur bien déterminé et compréhensible.

L'utilisateur n'écrit pas d'application mais désire soumettre des travaux à des services qu'il connait par leurs spécifications externes. L'apparence (pour l'utilisateur) d'un service ne correspond pas forcément à sa structure réelle ni à la façon dont il s'exécute.

Suivant l'environnement d'ordinateurs qui lui est offert l'utilisateur peut disposer de différents moyens d'accès aux applications.

1.1.2. - Environnement limité à un seul ordinateur :

L'utilisateur accède à cet ordinateur par un terminal (pour simplifier nous groupons sous ce vocable tout moyen d'entrer sur le calculateur: télétype, lecteur de cartes, lecteur de ruban, etc ...).

Sur un ordinateur existe un système d'exploitation qui:

. gère les terminaux
. gère l'accès aux services et contrôle leur bonne exécution.

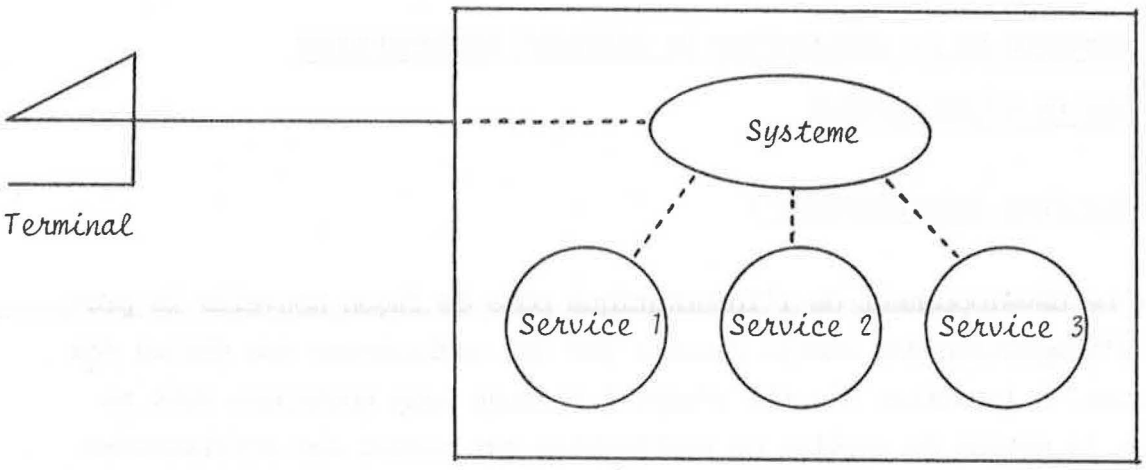


Figure 1

L'usager adresse des requêtes au système pour lui dire qu'il veut soumettre tel travail à tel service donné. Il dispose donc d'un langage de commande (JCL: Job Command Language) pour parler au système.

Le travail qu'il veut faire exécuter peut en fait nécessiter plusieurs applications. Par exemple compiler un fichier source Fortran et faire sortir le listage sur une imprimante revient à utiliser trois services: la gestion des fichiers, le compilateur Fortran et la gestion de l'imprimante. C'est le langage de commande qui permet à l'utilisateur d'indiquer au système cette suite d'appels et dans quel but. Suivant la forme du langage de commande les ordres de l'utilisateur prendront un aspect plus ou moins ésotérique.

L'utilisateur veut: compiler le fichier source Fortran TOTO dans le fichier objet TITI et sortir le listage sur l'imprimante. Avec un langage aux ordres primaires cela donne:

. Entrée = TOTO
. Sortie = TITI
. Listage = IMP
. Compile FORTRAN

Un langage plus évolué, utilisant des structures de contrôle mieux

appropriées, permet de l'écrire en une seule instruction:

. Imprime(Compile(FORTRAN,TOTO,TITI))

1.1.3. - Environnement composé de plusieurs ordinateurs :

Malgré tous ses perfectionnements un ordinateur ne peut offrir tous les services intéressant les usagers. Il est alors possible de fournir plusieurs calculateurs, chacun supportant un certain nombre de services dont l'union donne un ensemble à peu près complet.

Si ces ordinateurs sont totalement indépendants (il n'y a pas d'interconnexion et chacun gère ses propres terminaux) nous sommes ramenés à autant de configurations semblables à celle décrite précédemment qu'il y a de calculateurs.

L'utilisateur doit se déplacer d'un ordinateur à l'autre sur un même site suivant la localisation des services avec lesquels il veut travailler.

Une meilleure solution est que l'utilisateur accède à tous les ordinateurs à partir d'un même terminal [Pouzin, GEN-1].Le terminal est relié à une "boite noire" qui peut accéder à tous les calculateurs. On appelle cette "boite noire" connecteur. Du terminal on peut envoyer à ce connecteur un ordre de connexion avec l'un quelconque des ordinateurs.

Nous allons présenter des réalisations possibles de ce connecteur.

1.1.3.1. - Commutateur physique :

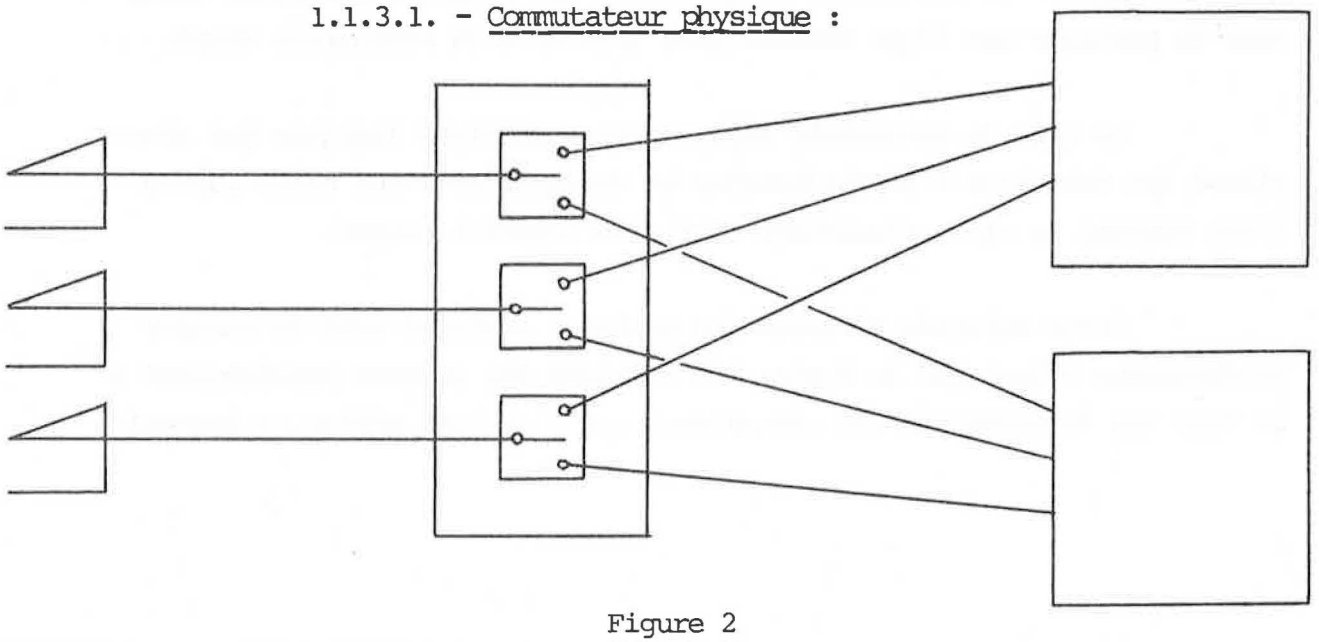


Figure 2

L'idée la plus simple est celle d'un commutateur attaché à chaque terminal. Il réalise la connexion physique avec l'ordinateur intéressant l'utilisateur en fermant un circuit entre la ligne joignant le terminal et celle reliant l'ordinateur.

L'inconvénient majeur de cette solution est que le nombre de connexions croit proportionnellement avec le nombre de terminaux et de calculateurs.

1.1.3.2. - Concentrateur - diffuseur de terminaux :

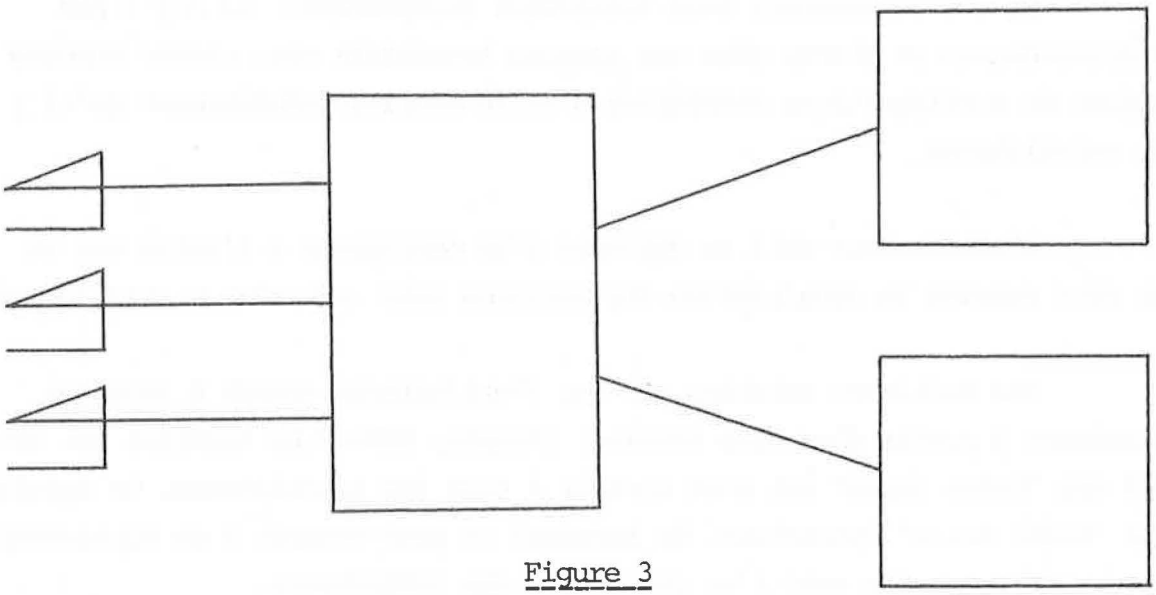


Figure 3

Plutôt que de réaliser des commutations physiques entre lignes on peut utiliser un connecteur programmable qui permet à plusieurs terminaux de partager une ligne commune pour accéder à un ordinateur donné.

Ce type de connecteur réalise des connexions logiques qui nécessitent des fonctions d'identification et de contrôle. Pour cette raison c'est souvent un mini-calculateur doté d'un logiciel adapté.

Cette solution présente une certaine analogie avec le service téléphonique offert par la Poste. Les combinés des abonnés représentant à la fois les terminaux et les ordinateurs que l'on veut mettre en communi-

cation. Il n'existe pas de ligne particulière entre chaque couple de combinés mais chacun d'eux est relié à un ensemble de centraux téléphoniques maillé qui joue le rôle du connecteur. En effet sur un ordre venant du combiné (numéro d'appel formé sur le cadran) cet ensemble de centraux est capable d'établir une liaison temporaire pour la durée de la communication entre l'appelant et l'appelé. Une fois la communication terminée cette liaison est détruite.

1.1.3.3. - Réseau de communication :

Jusqu'ici nous avons supposé qu'un service était entièrement réalisé sur un seul ordinateur. Si l'on dispose de plusieurs ordinateurs il peut être intéressant de répartir une application sur plusieurs d'entre eux. De même des services situés sur des calculateurs distincts peuvent avoir la nécessité de coopérer, donc de communiquer entre eux, pour faire un travail utile. C'est le cas pour une application particulière capable d'utiliser des données stockées sur un autre calculateur que celui où elle s'exécute ou d'un programme pouvant faire des entrées/sorties sur des appareils spéciaux (graphiques par exemple) gérés par un ordinateur différent de celui où se fait le traitement et qui peut ne pas posséder de tels appareils.

Les réseaux généraux [Pouzin, GEN-2] sont conçus pour permettre à un grand nombre d'utilisateurs d'accéder à de multiples ressources, mais aussi pour fournir un outil qui puisse servir de support à des applications diverses, en leur permettant éventuellement de dialoguer entre elles.

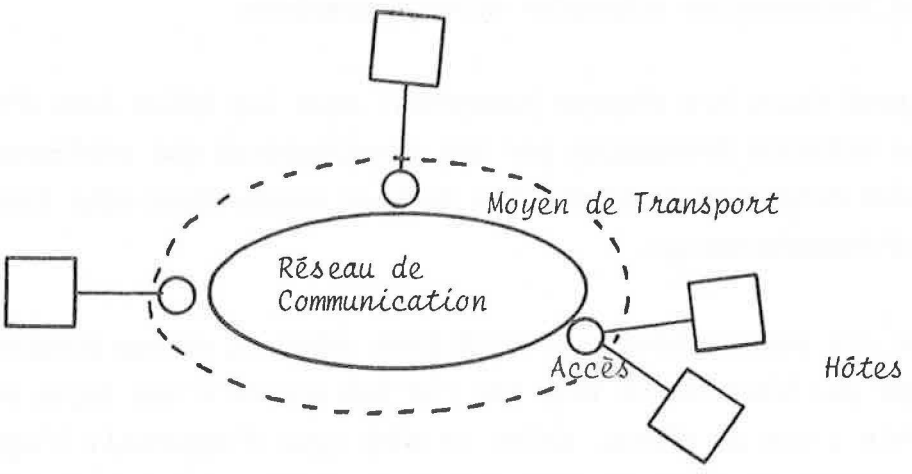


Figure 4

La structure du réseau est à deux niveaux:

. Le réseau ou système de communication qui offre aux usagers un service de transmission.

. Les hôtes qui sont les ordinateurs et les terminaux connectés.

Chaque hôte accède donc au système de communication et l'utilise comme une boite noire pour y déposer et en retirer des messages. Tous les accès des hôtes doivent présenter le même interface de communication vis à vis du réseau, ceci dans le but de ne pas compliquer ce dernier en l'obligeant à connaitre et accepter des accès différents particuliers à chaque hôte et de permettre l'introduction facile d'un hôte: c'est au nouvel arrivant de se conformer à l'interface standard. Ces accès des hôtes forment une nouvelle couche sur le système de communication que l'on appelle moyen de transport.

Du côté des hôtes il faut gérer cet accès au réseau. Les calculateurs ainsi que les terminaux intelligents ont les moyens de le faire, ce qui n'est pas le cas des terminaux simples, d'où la nécessité de créer des concentrateurs de terminaux (généralement de petits ordinateurs) auxquels sont reliés plusieurs terminaux et dont le rôle est de gérer ces derniers et l'accès au réseau.

Le moyen de transport, comme son nom l'indique, ne fait que le transport. Il fournit un adressage pour permettre aux hôtes de se désigner les uns les autres et assure des connexions logiques entre eux avec différents contrôles (flux, erreur) pour éviter au maximum les pertes et les erreurs, mais il n'intervient pas sur les informations transportées. Celles-ci dépendent des hôtes qui doivent se connaitre et si les types d'hôtes sont nombreux la gestion des informations échangées devient complexe.

On peut faire des réseaux homogènes: tous les hôtes sont d'un même type. C'est la solution développée par les constructeurs qui réalisent des réseaux avec des ordinateurs compatibles de leur gamme. Mais elle limite les possibilités d'interconnexion.

Pour les réseaux hétérogènes il faut créer un niveau appareil virtuel qui permet aux hôtes de se voir les uns les autres d'une façon standard. Chaque hôte, vis à vis du réseau, offre le même type d'appareil: l'appareil virtuel. A sa charge de faire la correspondance entre cet appareil virtuel et

un (ou plusieurs) appareil(s) réel(s) existant chez lui, ce qui est plus aisé à faire que connaitre chaque appareil réel connecté au réseau et permet de fonctionner avec des appareils différents en des instants différents.

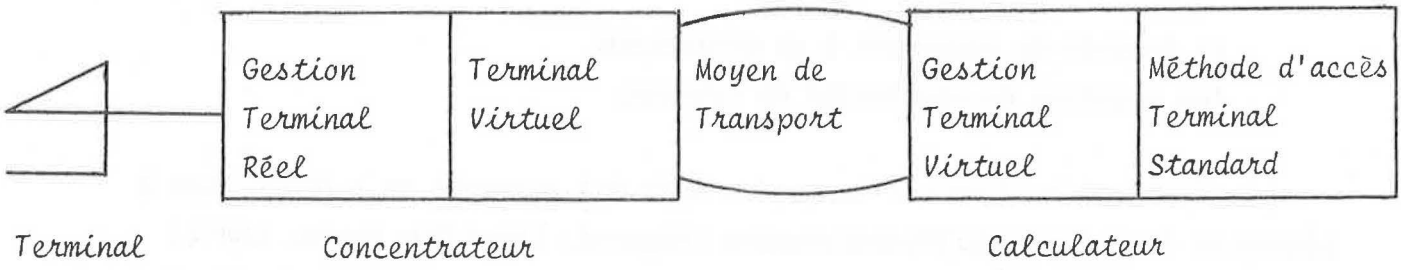


Figure 5

Ainsi entre un terminal et un ordinateur le concentrateur va simuler le terminal en appareil virtuel pour le calculateur, ce dernier présentant sa gestion de terminaux comme acceptant le terminal virtuel.

Dans ce cas le connecteur qui permet au terminal d'accéder à n'importe quel ordinateur est l'ensemble: concentrateur de terminaux - moyen de transport - appareil virtuel.

1.1.4. - Point de vue de l'utilisateur :

Qu'il y ait une "petite boite" ou tout un réseau pour faire la connexion l'utilisateur ne fait pas la différence. Il doit connaitre les ordres à envoyer au connecteur pour lui dire: je veux accéder à tel ordinateur. Une fois la liaison établie, et gérée de façon transparente pour lui, il se retrouve sous le système d'exploitation du calculateur auquel il doit adresser des requêtes dans le langage de commande correspondant. C'est à dire qu'on revient au mécanisme étudié au paragraphe 1.1.2. L'utilisateur doit donc être capable de s'adresser à chaque système de chaque ordinateur auquel il peut se connecter, ce qui l'oblige à connaitre de multiples langages de commande.

Il apparait intéressant de fournir à l'usager un moyen unique, c'est à dire un langage unique, pour accéder aux différentes applications.

Lorsqu'un seul ordinateur supporte tous les services le langage est

effectivement unique: c'est le langage de commande du système d'exploitation.

Dans le cas de multiple calculateurs il devient nécessaire de développer un langage réseau regroupant:

. la demande de connexion à un ordinateur

. les requêtes de soumission de travaux.

Le besoin d'un tel langage a déjà été ressenti et a donné lieu à plusieurs études sur différents réseaux [Raymond, LAN-1][Du Masle, LAN-2] [Sergeant, LAN-3]. Ce langage est conduit à remplacer, du point de vue de l'utilisateur, les langages de commande des systèmes accessibles.

Nous voyons donc que quelle que soit la configuration offerte, un ordinateur supportant m applications ou n ordinateurs supportant m applications (avec $m = m_1 + m_2 + \ldots + m_n$, m_i étant le nombre d'applications supportées par l'ordinateur i), l'utilisateur n'est pas concerné par l'aspect physique (le nombre de calculateurs) de cette configuration. Ce qui l'intéresse ce sont les m applications et un langage unique, que ce soit le langage de commande d'un système donné ou un langage de commande réseau, permettant de les utiliser.

La structure logique apparente à l'usager doit être:

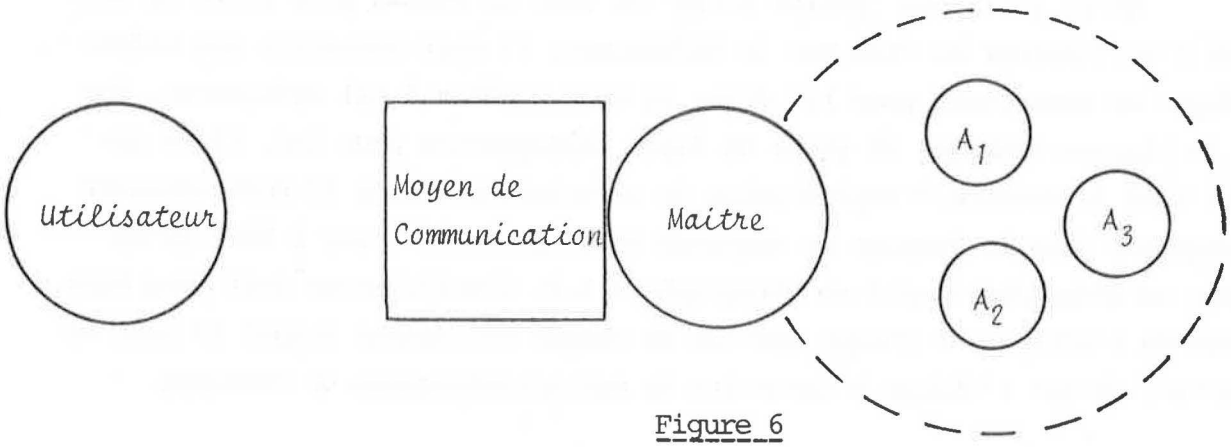


Figure 6

. un ensemble d'applications bien définies réalisant chacune une fonction bien précise

. un maitre qui gère ces applications et à qui on peut adresser des requêtes dans un langage de commande unique

. un moyen de communication avec le maitre: souvent par l'intermédiaire d'un terminal.

Les détails de réalisation n'intéressent pas l'utilisateur. Peu lui importe si cette structure logique est virtuelle et simulée pour lui cacher les vrais mécanismes qui autorisent l'exécution des travaux. Tout ce qu'il demande c'est qu'ayant cette vision du centre de calcul toute la machinerie nécessaire au bon fonctionnement des ordinateurs respecte cette vision et lui permette de travailler.

Objectifs à atteindre :

Un centre de calcul, c'est à dire la structure administrative et technique mise en place sur un site donné pour offrir des services aux utilisateurs à l'aide de moyens informatiques, doit être organisé en vue des objectifs suivants:

. accès facile aux différents services du centre, que ceux-ci soient sur:

- des ordinateurs locaux
- un réseau externe. Celui-ci peut alors être considéré comme un seul ordinateur réalisant diverses applications. Par exemple le réseau Cyclades accessible par un concentrateur de terminaux est considéré au même titre que les calculateurs présents sur le site.

. transparence de la réalisation des services vis à vis des utilisateurs.

1.2. - APPROCHE FONCTIONNELLE

Notre approche va se faire en deux étapes:

. La première consiste à distinguer des entités fonctionnelles disjointes, c'est à dire autant d'aspects que l'organisation du centre de calcul doit présenter ou offrir [Rodriguez, GEN-3].

. La seconde vise à établir des règles de communication entre les entités concidérées.

1.2.1 - Les différentes fonctions :

Nous distinguons quatre types de fonctions:

1) Fonction d'accès de l'utilisateur au centre de calcul

Celle-ci comprend les terminaux et leur gestion. Cette fonction se divise en autant de sous-fonctions qu'il y a de terminaux.

2) Fonction d'interprétation du langage de commande

Elle permet d'adresser des requêtes d'utilisation des services offerts par le centre de calcul.

3) Fonction réalisée par un service

C'est la fonction qui intéresse l'utilisateur.

4) Fonction de communication

On dit qu'une fonction est réalisée par une entité, cette fonction pouvant être complexe l'entité se décompose en objets qui réalisent:

. soit la fonction

. soit des outils pour la réalisation de la fonction.

Il apparait donc nécessaire d'avoir une fonction de communication entre les objets ainsi qu'entre les entités.

Exemple 1 : dans le cas d'un centre de calcul basé sur un seul ordinateur

- La fonction d'accès est réalisée par l'ensemble (terminaux , gestion des terminaux).
- La fonction d'interprétation du langage de commande est comprise dans le système d'exploitation.
- La fonction de communication dépend des caractéristiques du matériel et du logiciel: (appel de routines systèmes ou utilisateurs, coopération entre tâches du système ou exécutant les programmes d'application).

Exemple 2 : l'entité réalisant la fonction d'un service réparti sur deux ordinateurs se compose de trois objets:

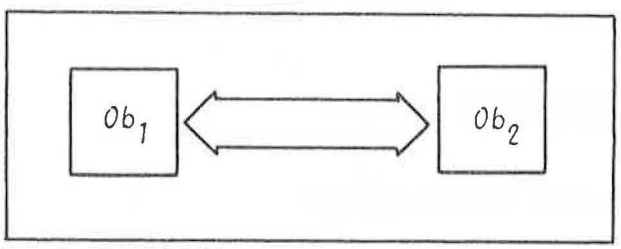


Figure 7

- Deux objets Ob_1 et Ob_2 correspondant respectivement à chacune des parties du service localisées sur l'un ou l'autre des ordinateurs.
- Un objet qui permet de faire communiquer Ob_1 et Ob_2 et qui recouvre les moyens de connexion et de coopération des deux ordinateurs concernant les deux parties du service.

Les différentes fonctions mises en évidence peuvent être schématisées de la façon suivante:

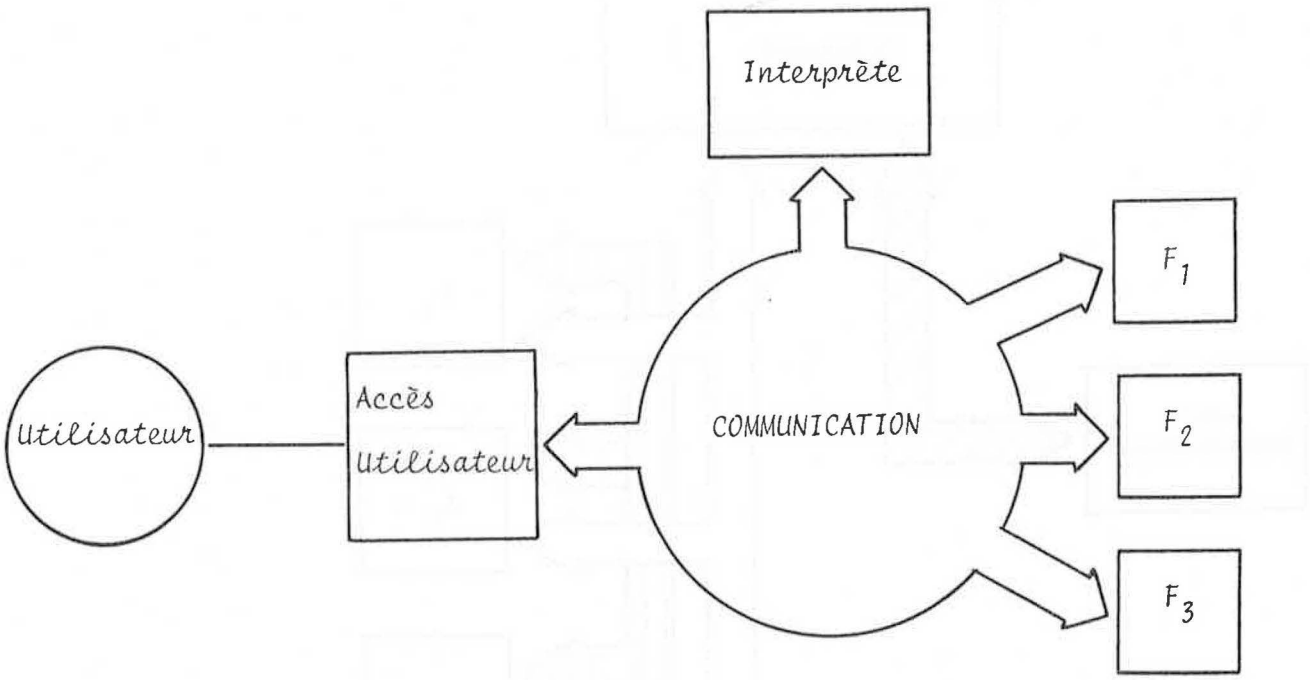


Figure 8

1.2.2. - Cas particulier de la fonction de communication :

Dans ce paragraphe nous allons expliciter la nature des échanges réalisés par la fonction de communication.

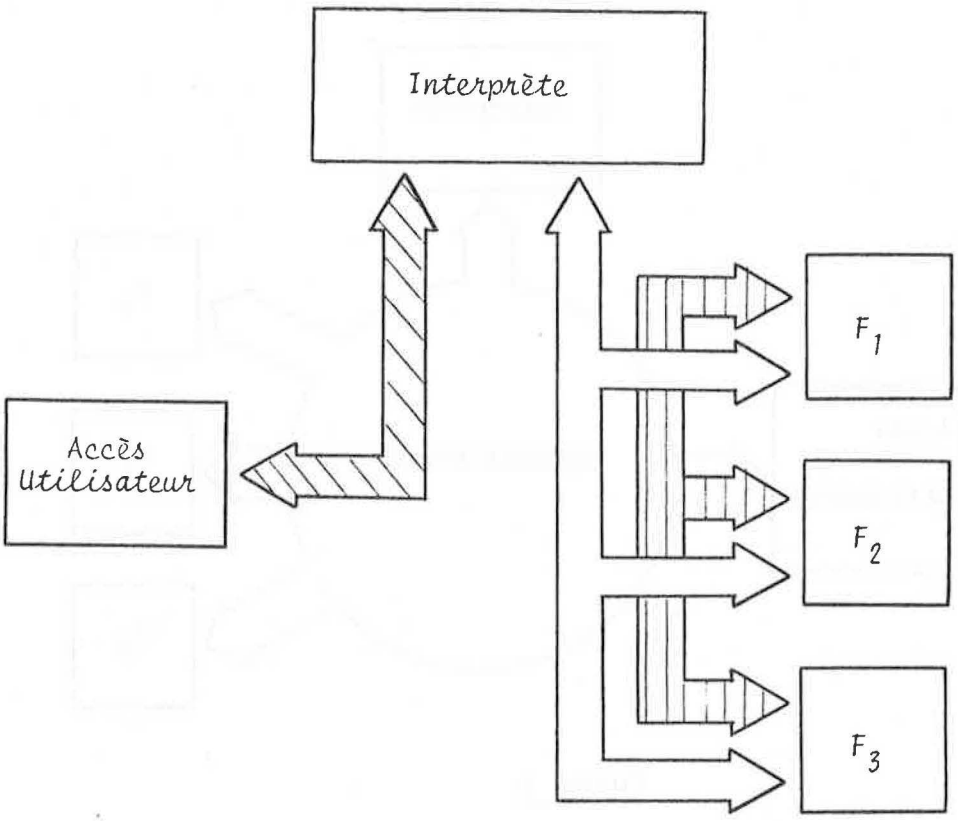
Celle-ci se divise en trois sous-fonctions:

1) Echanges utilisateur - interprète : ils permettent à l'utilisateur de communiquer avec l'interprète pour lui transmettre ses requêtes de soumission de travaux.

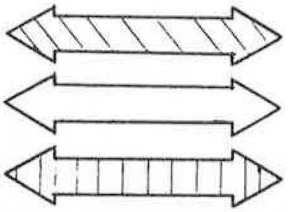
2) Echanges interprète - service : ils permettent l'exécution des travaux demandés par l'utilisateur.

Remarque: il est important de noter que l'utilisateur n'accède à un service que via l'interprète.

3) Echanges service - service : ils sont nécessaires dans le cas de coopération de services.



Nature des échanges:



Accès utilisateur - Interprète

Interprète - Service

Service - Service

Figure 9

1.2.3. - Exemples de réalisation :

SOC ..... [RES-1]

Dans le projet SOC (Système d'Ordinateurs Connectés) apparait la notion de système réseau qui est au réseau ce qu'est un système d'exploitation à l'ordinateur [Kimbleton, GEN-4]. Il est physiquement réparti sous forme d'une tâche utilisateur sur chacun des calculateurs connectés. La réalisation est simplifiée par le fait qu'il s'agit d'un réseau homogène (composé d'IBM 360 - 370). L'utilisateur dispose d'un langage de commande unique: le NCL (Network Control Language) ou LE (Langage Externe) qui lui permet de transférer des données d'un ordinateur à l'autre ou de soumettre des travaux (NJob = Network Job) à n'importe quel calculateur du réseau. Le LE est traduit en NIL (Network Internal Language) langage interne qui circule dans le réseau. Chaque commande NIL correspond à une fonction réseau exécutable soit entièrement sur un des ordinateurs connectés soit en synchronisation avec un autre ordinateur. La traduction du LE et l'aiguillage des commandes NIL vers l'ordinateur concerné sont faits par la partie du système réseau résidente sur le calculateur auquel est connecté le terminal d'accès de l'utilisateur [De Wouters, RES-2].

IGOR ..... [Sergeant, LAN-3]

Une autre étude a été faite pour doter le réseau Cyclades d'un système réseau. Comme pour SOC deux langages ont été développés:un langage externe (LE) pour les utilisateurs et un langage interne (LI) compréhensible par chaque ordinateur connecté (qui doit donc disposer d'un interprète LI : un interprète général orienté réseau, IGOR, a été défini).

Globalement ces deux études réalisent les fonctions décrites au paragraphe 1.2.1 de la même façon.

- La fonction d'interprétation est physiquement répartie sur chacun des ordinateurs auxquels sont connectés les terminaux par la traduction de phrases du langage externe en phrases du langage interne.

- La communication terminal - interprète est, pour chaque calculateur

celle offerte par le système d'exploitation local entre un terminal et une tâche utilisateur.

- La communication interprète - service passe par le moyen de transport fourni par le réseau et l'interprétation locale du LI pour le système d'exploitation de l'ordinateur où est localisé le service concerné. L'interprète traduisant en langage de commande local les ordres LI qui viennent du réseau, tout se passe pour le système comme si les requêtes de l'utilisateur arrivaient d'un terminal local.

UNIX ..... [Manning, RES-3]

Prenant une orientation différente ce projet développe un accès réseau basé sur le système UNIX (réalisé sur PDP.11). L'idée est de concevoir un langage de commande réseau en ajoutant des requêtes réseau au langage de commande du système d'exploitation de l'ordinateur connecté supportant les terminaux. Le système remplit ainsi deux rôles: processeur local et machine d'accès au réseau. Les commandes réseau respectent la forme standard de toutes les commandes du langage, ce qui simplifie la vision de l'utilisateur, et sont réalisées chacune par un processus. On récupère ainsi toutes les fonctions réalisées par le système d'exploitation. Mais comme dans les deux exemples précédents les fonctions de communication avec les services distants sont celles fournies par le réseau et l'ordinateur local doit s'y soumettre.

L'intérêt de cette solution est surtout de ne pas vouloir résoudre le problème du langage utilisateur globalement pour tout le réseau. Disposer d'un langage de commande réseau externe qui soit le même dans tous les centres de calcul ayant un accès au réseau est bien évidemment séduisant mais difficile à réaliser à cause de l'hétérogénéité des ordinateurs. Bien entendu un langage local pose le problème d'obliger un utilisateur à réapprendre un autre langage s'il change de site. Mais on peut considérer qu'il change d'environnement et est donc forcé de s'adapter à son nouveau milieu.

Nous nous sommes placés dans l'optique de réaliser un centre de calcul présentant aux utilisateurs l'aspect externe défini au paragraphe 1.1.4. Notre but n'est donc pas de nous attaquer aux problèmes généraux de l'utili-

sation d'un réseau important du type Cyclades. Notre ambition est plus modeste et se limite à la définition du centre de calcul de notre site. Pour cette raison nous nous sommes orientés vers une solution du type langage local.

CHAPITRE 2

PRESENTATION DU CENTRE DE CALCUL REPARTI

2.1. - LOGIQUE DE LA REALISATION PROPOSEE

Notre but est de réaliser les fonctions définies au paragraphe 1.2 en utilisant un certain nombre d'ordinateurs et d'accès à des réseaux externes (Cyclades). Cet ensemble sera appelé Centre de Calcul Réparti et noté CCR par la suite.

2.1.1. - Contraintes fixées :

Notre approche tient compte de certaines contraintes résultant soit d'un choix délibéré de notre part, soit du matériel (logiciel et physique) à notre disposition et auquel il faut s'adapter [CCR-1].

1) Utilisation de mini-ordinateurs

L'idée d'affecter à un ordinateur une (ou n, si n reste petit) fonction simple n'est pas nouvelle. Un calculateur "universel" essayant de réaliser le maximum de fonctions ne peut être optimisé pour chacune d'elle, il faut trouver un compromis entre chaque fonction. A partir du moment où on limite le nombre de fonctionsil devient plus facile d'optimiser leur réalisation. De plus une solution basée sur une idée de ce style est plus simple dans la mesure où l'écriture d'un logiciel spécifique est plus aisée à développer et maintenir qu'un logiciel général. Une telle solution a été rarement mise en oeuvre à cause de son coût sur un gros ordinateur qui, s'il ne travaille que pour deux ou trois applications, est nettement sous-employé. Avec des mini-ordinateurs, dont le prix n'a cessé de baisser ces dernières années, l'idée devient économiquement viable.

Remplacer un gros ordinateur par un réseau de mini-calculateurs est financièrement intéressant d'abord parce que la somme des coûts des minis n'atteint généralement pas le prix d'un gros. Ensuite parce que l'investissement peut être progressif: on achète les ordinateurs un par un et on les ajoute au réseau. Un autre avantage est politique: le matériel peut être hétérogène et provenir d'origines diverses, on n'est donc plus sous la dépendance d'un seul constructeur pour son équipement.

Un centre de calcul basé sur un ensemble de mini-calculateurs interconnectés est donc une solution séduisante. C'est vrai dans le cadre des objectifs que l'on se fixe, mais un réseau de minis n'a pas toutes les performances d'un gros ordinateur pour certaines applications ; le temps de réponse est généralement plus long à cause des transferts, ce qui fait qu'il ne faut pas croire que les gros ordinateurs sont condamnés.

2) Spécialisation des ordinateurs

Une approche d'un réseau de mini-ordinateurs est de redéfinir une nouvelle structure matérielle: c'est le cas du projet M2 [Hebenstreit, RES-4].

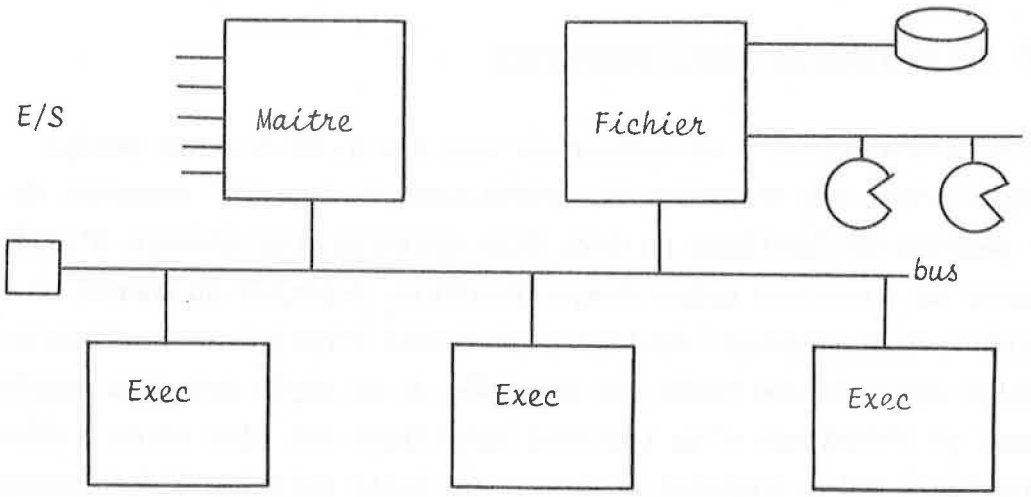


Figure 10

On distingue:
- un mini-fichier qui gère une mémoire disque pour les autres minis.
- un mini-maître auquel sont connectés tous les périphériques d'entrées / sorties et qui assume le rôle de répartiteur de tâche entre les minis.
- des minis "exécutants" qui sont banalisés. Une exécution peut s'effectuer indifféremment sur l'un d'eux, notamment en fonction de la charge.

Ils sont tous reliés entre eux par un bus unique. Cette solution permet facilement l'ajout d'un nouvel ordinateur du point de vue matériel. De plus le maître peut jouer le rôle d'interlocuteur privilégié pour les utilisateurs. Mais les minis "exécutants" doivent avoir des systèmes capables

de comprendre le même langage de commande puisqu'ils sont équivalents. C'est à dire qu'il faut réécrire le logiciel de chaque calculateur, celui fourni par le constructeur n'étant pas adapté au réseau.

Nous préférons maintenir la spécialisation de chaque mini. Nous conservons le système d'exploitation fourni par le constructeur ainsi que les services développés sous ce système. Nous nous limitons à l'adjonction, là où c'est nécessaire, du logiciel d'accès qui permet au calculateur d'être connecté au réseau.

Notre rôle n'est pas d'écrire les applications mais de mettre à la disposition des utilisateurs celles fournies par le constructeur ou d'autres programmeurs sans les modifier. En ce sens il ne s'agit pas d'un véritable système réparti, mais plutôt de la réunion d'un ensemble de services indépendants "chapeautés" par un même langage.

3) Limitation des techniques développées [Mac Quillan, GEN-5]

Toute architecture de réseau entraine des problèmes de connexion et de communication entre ordinateurs. Les minis venant de fabriquants différents respectent très souvent, heureusement, des normes internationales bien définies. Pour les communications il faut bien voir qu'un réseau de minis diffère d'un réseau universel style ARPA ou Cyclades. Les ordinateurs sont sur le même site, moins nombreux et spécialisés. Le problème du routage de messages peut être particularisé. On peut réduire la quantité d'information qui circule sur le réseau par la connaissance et la définition de ces informations alors qu'un réseau universel nécessite toutes sortes de dialogues à tous les niveaux.

Notre projet ne tend pas à définir de nouvelles techniques réseau, mais une logique d'accès pour satisfaire aux objectifs définis au paragraphe 1.1.4 en utilisant des techniques existantes et connues. Dans cette optique notre étude diffère du projet ARAMIS [Beaufils, RES-5] qui se propose de réaliser un réseau de mini-ordinateurs homogènes. L'effort a été porté sur l'aspect transport d'information (commutation de paquets) et sûreté de fonctionnement. Cela a débouché sur la conception d'un matériel multiprocesseur adapté.

4) Développement du centre de calcul

Un des objectifs essentiels est de permettre une extension facile du centre de calcul. Cela comprend:

- l'addition aisée de nouveaux services. Il peut s'agir d'ajouter des services inexistants ou de réaliser localement un service jusqu'alors sous-traité à l'extérieur (réseau externe) parce que peu fréquent. Sa fréquence d'utilisation augmentant, on peut le traiter sur le site en ajoutant de nouvelles ressources (nouveau calculateur, extension d'un calculateur existant, etc ...).

- le redistribution des fonctions en cas de panne d'un des ordinateurs. Une nouvelle configuration du CCR permet alors d'offrir, sinon tous les services, du moins un fonctionnement en mode dégradé.

Il est nécessaire que le projet soit développé progressivement, les services existants devant être exploitables par les utilisateurs du centre. Il n'est pas question en effet de supprimer des services, même temporairement, mais d'en simplifier l'accès et d'en ajouter d'autres. L'extension pourra être faite au coup par coup avec mise en service effective des produits sitôt qu'ils sont déclarés opérationnels, sans pour autant stopper l'exploitation pendant de longues périodes, solution qui serait jugée inacceptable par les utilisateurs.

2.1.2. - Choix et motivations [CCR-2]

L'utilisateur dispose d'un langage de commande unique que nous appelons LCE (Langage de Commande Externe). Mais les ordinateurs du CCR, nous l'avons vu, conservent leur langage de commande propre. Il est nécessaire d'effectuer la traduction à un instant donné.

Une première idée est de localiser l'interprète sur un seul ordinateur. Il doit donc connaitre tous les calculateurs du centre et leur langage de commande respectif. Ainsi il peut traduire les requêtes de l'utilisateur dans le langage de commande de l'ordinateur concerné. On voit immédiatement l'inconvénient majeur d'une telle solution: l'interprète risque d'être gros et toute nouvelle connexion d'ordinateur entrainera des modifications importantes avec l'introduction du nouveau langage dans le traducteur.

Nous préférons que soit réalisée sur chaque calculateur la traduction dans le langage de commande local.

Néanmoins la fonction d'aiguillage qui assure la transparence de la localisation des services vis à vis des utilisateurs doit être réalisée avant la traduction en langage local puisqu'il faut bien orienter les requêtes à traduire vers les ordinateurs qu'elles concernent [Chupin, LAN-4]. Donc une partie de l'interprétation du langage utilisateur doit être faite avant l'accès à un ordinateur.

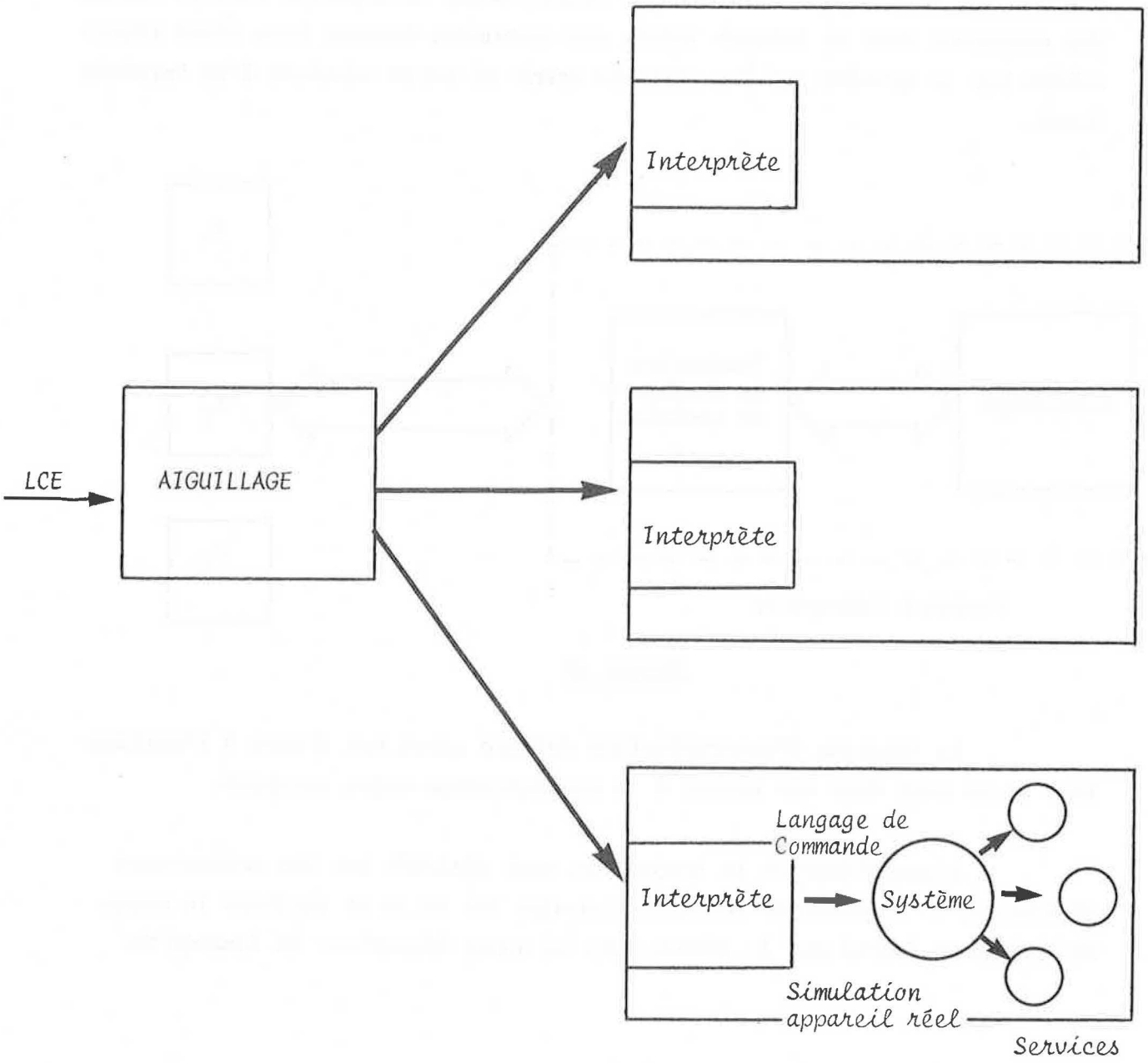


Figure 11

La fonction d'interprétation est ainsi effectuée par deux sous-fonctions:

- aiguillage vers les calculateurs des requêtes qui les concernent.
- traduction en langage local réalisée par l'interprète local qui simule pour le système d'exploitation un appareil réel et lui envoie des commandes standards. Le système peut alors assurer normalement la gestion des services.

Une fonction de communication est nécessaire entre les deux sous-fonctions de l'interprète. La communication entre interprète et services est assurée par le fait que l'interprète local simule un appareil réel et envoie des commandes dans le langage local, ces commandes peuvent être ainsi répercutées par le système sur les services comme si elles venaient d'un terminal local.

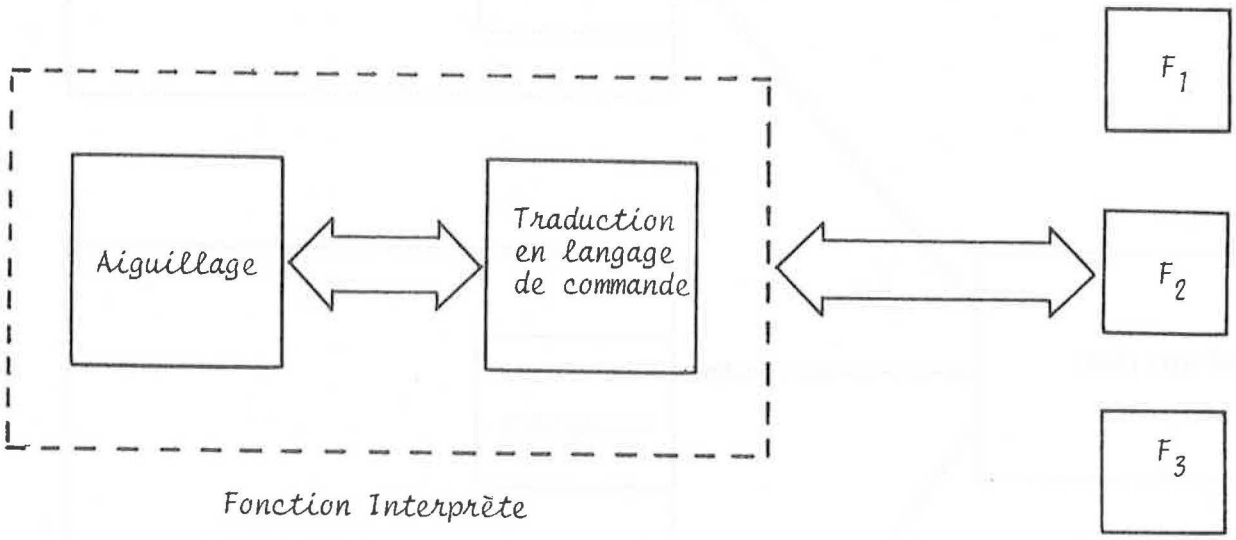


Figure 12

La fonction d'interprétation définie ainsi est dédiée à l'utilisateur et ne peut donc pas servir à la communication entre services.

L'aiguillage et la traduction sont réalisés sur des ordinateurs distincts, la fonction de communication qui les relie va utiliser le moyen de transport fourni par le réseau mais va aussi nécessiter un langage de

communication. Ce langage peut être le LCE, mais celui-ci est un langage externe dont la forme est adaptée aux utilisateurs. Il faut tenir compte de l'hétérogénéité du réseau et considérer que le langage à traduire par les interprètes locaux ne doit pas obliger à une analyse trop complexe. Pour cette raison, et suivant l'exemple de SOC, nous avons défini un langage interne (le LCI) à la forme et aux fonctions plus primaires que le LCE, mais dont l'analyse et la traduction sont plus faciles à faire. L'aiguillage ayant comme rôle supplémentaire de traduire le LCE en LCI.

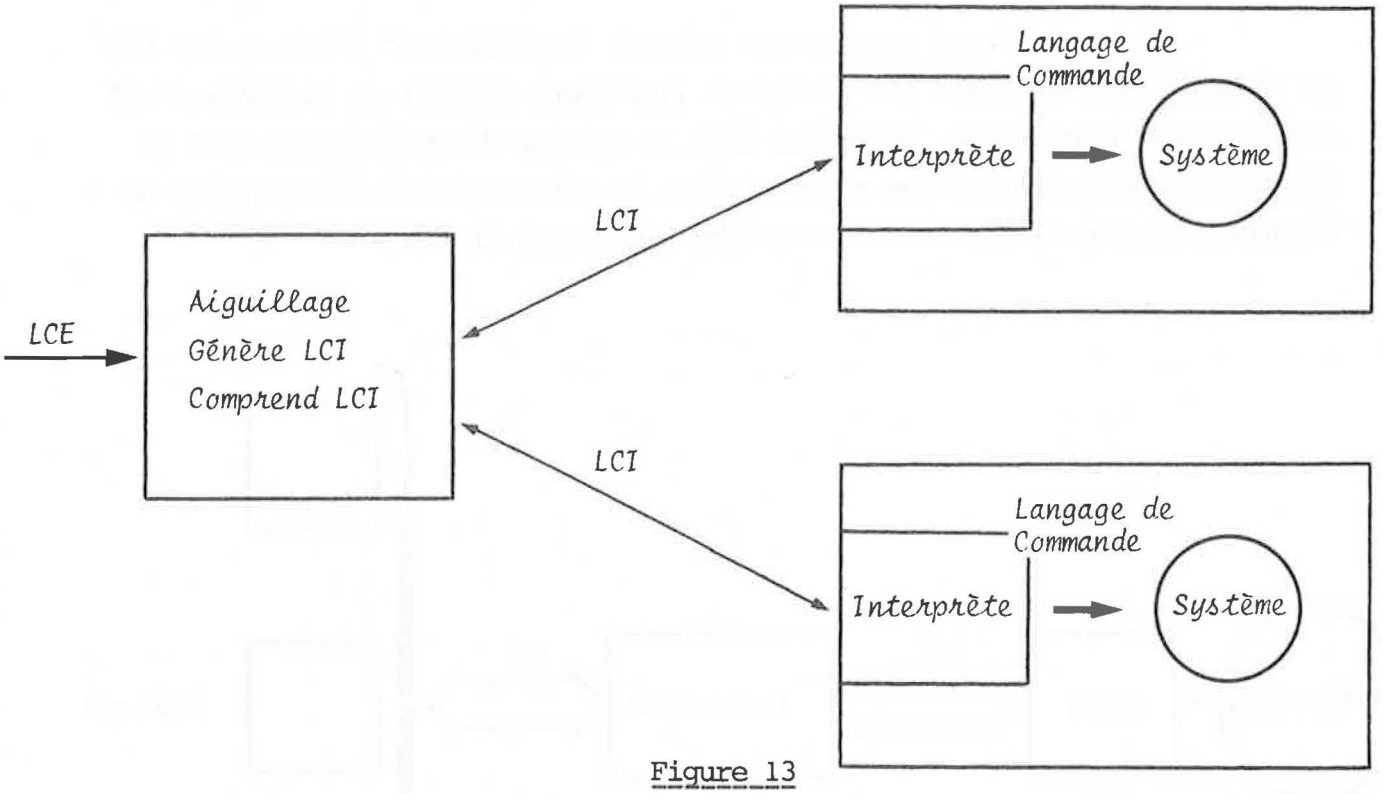


Figure 13

Il apparait intéressant de donner à ce LCI un rôle plus important que celui de simple "code intermédiaire" dans l'interprétation du LCE. Jusqu'à présent nous avons considéré que l'entité qui réalise la fonction service est le processus (ou programme) existant sous le système d'exploitation. Définissons plutôt le service comme une entité plus complète [Franchi, RES-6] contenant:

- la traduction LCI - langage de commande

- le système (ou une partie du système)
- l'application locale.

Cette entité communique avec l'aiguillage en LCI (c'est nécessaire pour pouvoir envoyer des réponses aux utilisateurs). La fonction aiguillage réalisée pour aller des usagers vers les services peut aussi être utilisée pour communiquer entre services puisqu'elle dispose des données nécessaires (localisation des services).

Cette structure nous permet donc de "recompacter" l'interprète LCE et résoud au niveau local les problèmes (justement locaux) de traduction LCI vers langage de commande, problèmes liés au système d'exploitation et à la réalisation particulière des applications. Au niveau global l'interprète ne s'adresse plus qu'à des entités services comprenant le LCI.

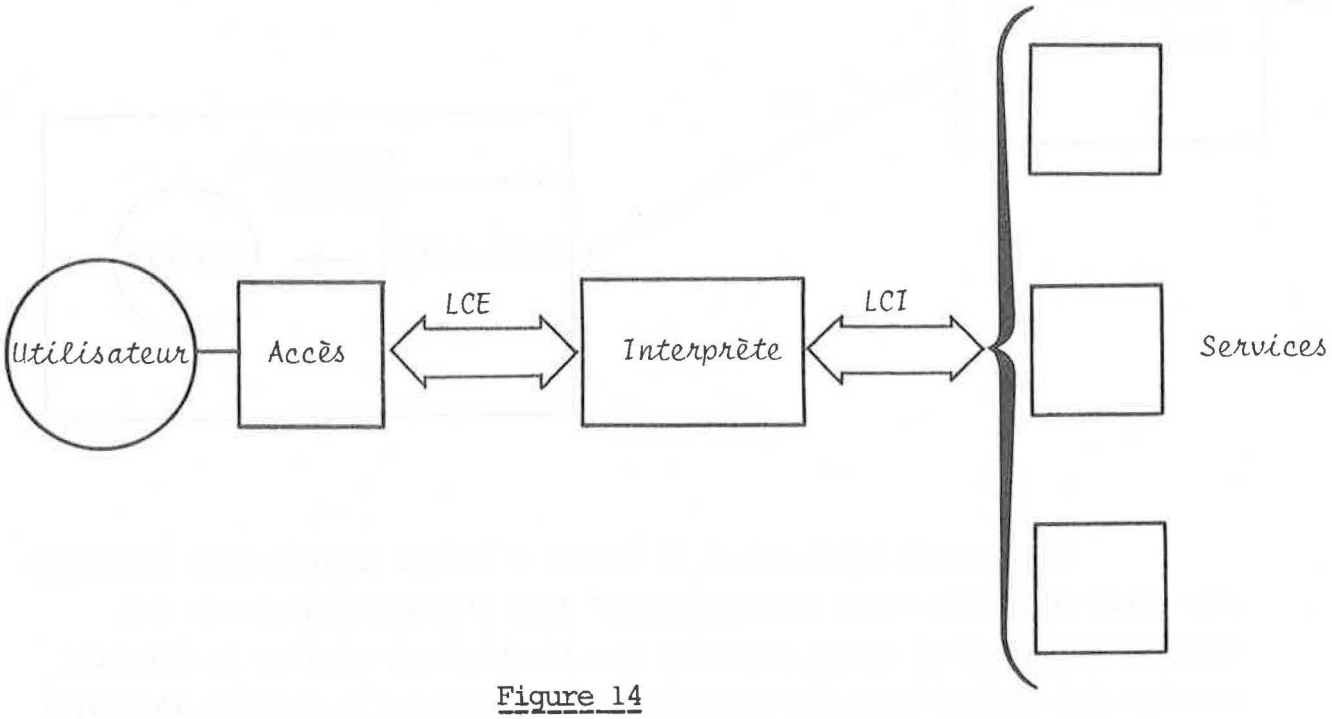


Figure 14

Le LCE ne permet pas de distinguer les points d'accès utilisateur (les terminaux) car c'est un langage de requêtes. Il faut donc que chaque terminal ait un point d'entrée chez l'interprète pour que ce dernier puisse le distinguer. On appelle voie logique de communication (notée VLC) le chemin logique entre le terminal et son point d'entrée. C'est l'ensemble

des voies logiques de communication qui représente la fonction de communication entre accès utilisateur et interprète.

La fonction de communication entre interprète et services peut être incluse dans le LCI. La distinction entre applications est alors faite au niveau LCI. Cette solution oblige l'interprète à connaitre la localisation des services sur les différents ordinateurs et à s'adresser à l'interprète LCI local. Nous avons dit plus haut qu'il est préférable que l'interprète s'adresse aux services qui, pour lui, comprennent le LCI sans se préoccuper de savoir si, en fait, ils se servent tous du même interprète LCI local (sous forme de routines systèmes partageables par exemple). C'est à dire qu'une solution semblable à celle qui résoud la communication utilisateur-interprète est meilleure. Chaque service possède un point d'entrée chez l'interprète. Le chemin logique interprète-service est aussi appelé voie logique de communication.

Maintenant que nous avons exposé les raisons et motivations qui ont guidé nos choix dans l'élaboration du CCR, nous allons décrire la structure logique en résultant.

2.2. - LES ENTITES EN PRESENCE ET LEURS RAPPORTS [CCR-2][CCR-3]

2.2.1. - Présentation des entités :

Elles sont au nombre de trois :

- Les utilisateurs.

- Le Correspondant qui se divise en deux: le Correspondant utilisateur (CU) à qui les utilisateurs s'adressent et le Correspondant service (CS) à qui les services s'adressent. C'est l'entité qui réalise la fonction d'interprétation, on lui a donné le nom de Correspondant car il peut correspondre à la fois avec les utilisateurs et les services.

- Les services réseaux.

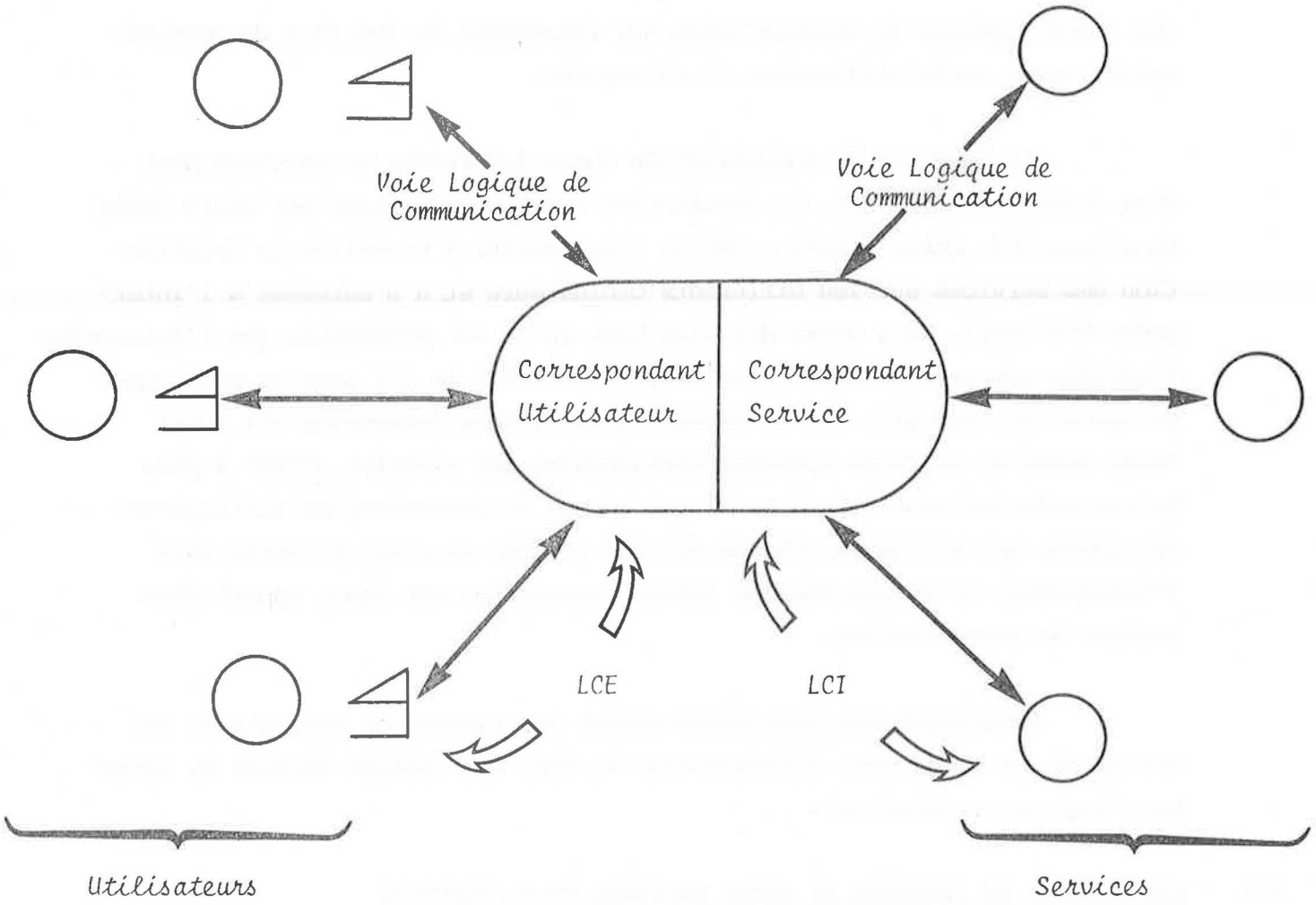


Figure 15

Le Correspondant utilisateur et les utilisateurs sont des entités de même niveau. L'utilisateur envoie ses requêtes au Correspondant utilisateur, il le voit comme le système d'exploitation du CCR, et ce dernier lui répond. Il y a donc dialogue entre eux par l'intermédiaire du LCE (Langage de Commande Externe). Rappelons que le qualificatif d'externe vient du fait qu'il permet la communication entre des entités extérieures au CCR (les utilisateurs) et le CCR.

De la même façon Correspondant service et services sont des entités de même niveau et dialoguent à l'aide du LCI (Langage de Commande Interne). Ce langage est qualifié d'interne puisque cette fois il s'agit de communication entre entités internes au CCR.

Le LCE et le LCI diffèrent par leur forme (le LCE doit être lisible par un être humain ce qui n'est pas le cas du LCI) mais aussi par leur fonction. Ils sont utilisés par des entités dont les buts et les besoins ne sont pas les mêmes. Si certaines de leurs commandes ont des fonctions similaires (n'oublions pas que le LCI joue en partie le rôle de "code intermédiaire" pour le LCE) beaucoup ne recouvrent pas le même champ d'activité .

La communication entre Correspondant et utilisateurs d'une part, Correspondant et services d'autre part se fait à travers les voies logiques de communication, entités logiques qui recouvrent l'ensemble des moyens de connexion et de transport permettant les liaisons entre les entités, toutes n'étant pas localisées sur le même site.

2.2.2. - Utilisateur :

C'est un être humain qui désire se servir des possibilités offertes par les ordinateurs. Il utilise (d'où son nom) les services.

Pour le CCR il existe un certain nombre d'accès utilisateur: les terminaux se trouvant chacun au bout d'une voie logique de communication dont l'autre extrèmité est reliée à un point d'entrée du Correspondant utilisateur. Quand il veut travailler avec le CCR l'utilisateur doit connaître le LCE et s'approprier un de ces accès. S'approprier veut dire qu'il s'est fait connaitre du système et que jusqu'à ce qu'il le quitte il est associé à son accès. Donc le Correspondant utilisateur doit connaitre les utilisateurs et cette connaissance doit être rémanente d'une session de travail à l'autre.

Pour cela tout utilisateur possède un contexte qui le décrit: son environnement contenant toute une série d'informations qui permettent de le qualifier:

- Un identificateur pour distinguer les usagers les uns des autres.
- Un mot de passe qui protège l'environnement.
- L'accès utilisé au cours d'une session.
- Le service appelé à un instant donné.
- Les ressources associées à l'utilisateur (fichiers, procédures, etc ...).

- Une priorité définissant les prérogatives de l'utilisateur. Tous les utilisateurs ne peuvent et ne veulent pas faire la même chose, notamment cette priorité contient la classe de l'utilisateur. A l'heure actuelle cinq classes sont définies:

  . Opérateur - il peut voir et modifier l'état du CCR
  . Manager Frontal - voir Chapitre 3
  . Système - il peut accéder aux systèmes d'exploitation des ordinateurs du CCR
  . Programmeur - il écrit des applications
  . Utilisateur d'application - accède simplement aux services communs

On peut aussi ranger certaines indications statistiques: nombre de connexions, temps de connexion, etc ... Nous verrons par la suite les autres informations que l'on met dans l'environnement utilisateur.

Créer un utilisateur c'est lui créer un environnement et le ranger dans un catalogue du Correspondant. Détruire un utilisateur c'est supprimer son environnement du catalogue, il n'existe alors plus pour le CCR.

2.2.3. - Service réseau :

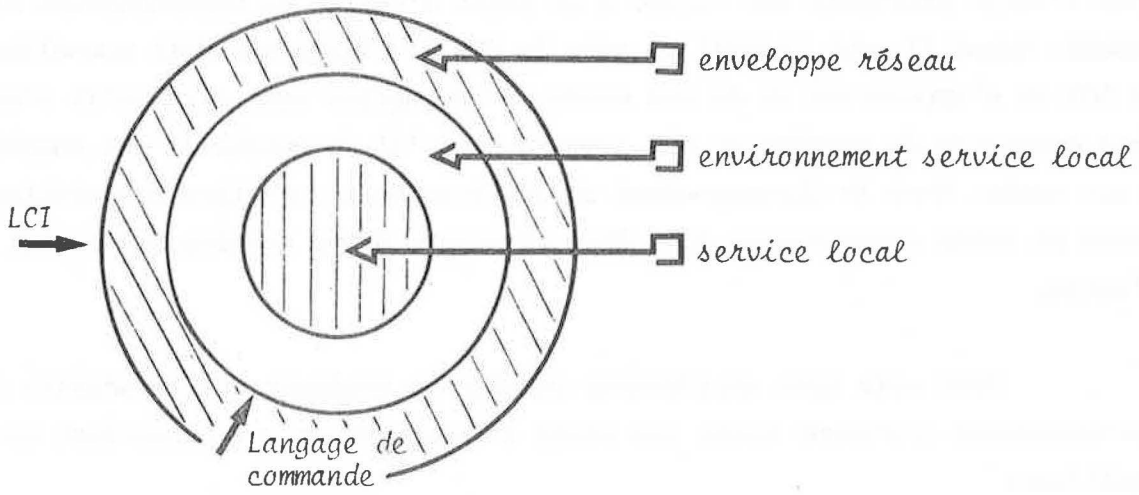


Figure 16

Un service réseau est constitué de trois niveaux :

- Service local: c'est l'application offerte par le calculateur hôte (compilateur, gestion de fichier, éditeur de texte, programme

de gestion, etc ...). Il est écrit par un programmeur ou fourni par le constructeur.

- Environnement service local: ce sont les parties du système d'exploitation du calculateur hôte nécessaires à l'exécution du service local. Ce niveau reçoit des commandes dans le langage local. Dans certains cas il peut ne pas exister. Certains ordinateurs sont livrés sans système mais avec des primitives et un macro-générateur permettant de créer son propre système. Dans ce cas on a tout intérêt à créer un système compatible avec le CCR et notamment dont le langage de commande (ou une partie) est le LCI.

- Enveloppe réseau: il s'agit du niveau qui reçoit les commandes LCI, les interprète et effectue la correspondance avec le langage de commande local. C'est ce niveau qui connait le Correspondant service et dialogue avec lui. Il connait aussi la voie logique de communication qui le relie au Correspondant et sait l'utiliser.

Un service réseau peut être réparti sur plusieurs sites (on entend par site tout ordinateur connecté ou réseau externe accessible) mais il n'est connu du CCR que sur un site donné. C'est le composant du service sur ce site qui est appelé service réseau et est maître des actions des différentes parties du service. Les éléments répartis du service sont reliés entre eux soit par des lignes spécialisées, soit par l'intermédiaire de points d'entrée du Correspondant service. Le maitre présente la même structure que celle décrite plus haut. Ce n'est pas le cas des autres parties s'il y a des lignes spécialisées, mais si on passe par des points d'entrée du Correspondant il faut qu'elles présentent le même aspect qu'un service réseau même si en fait elles ne sont pas accessibles en tant que tel.

Un exemple d'un tel service réparti est le service gestion de fichiers Chaque partie du service contient la gestion de fichiers fournie par le système d'exploitation du site. Le maitre tient à jour des tables de localisation des fichiers. Dans le cas d'un transfert de fichier d'un site à un autre le maitre prévient la (ou les) partie(s) concernée(s) et donne au site émetteur

le nom réseau du site récepteur. S'il n'existe pas de ligne spécialisée entre les deux sites en jeu le maitre fournit en plus au site émetteur le point d'entrée du Correspondant identifiant le destinataire.

Dans la suite nous ne considèrerons plus que les services réseaux que nous appellerons simplement services.

2.2.4. - Le Correspondant :

Nous avons vu qu'il se divise en deux.

. Correspondant utilisateur

Il connait les terminaux par les voies logiques de communication.

Il est capable de reconnaitre un utilisateur par son identificateur et de l'associer au terminal par lequel il accède. Ensuite il connait l'utilisateur par son environnement.

Il dialogue en LCE avec l'utilisateur. Il interprète le LCE en faisant évoluer l'environnement au fur et à mesure que des commandes lui arrivent et en déclenchant les actions voulues pour la bonne exécution de ces commandes. Parmi ces actions certaines peuvent être répercutées chez le Correspondant service.

Il envoie des réponses aux utilisateurs provenant d'actions déclenchées par le Correspondant service.

. Correspondant service

Il connait les services par les voies logiques de communication.

Il dialogue avec les services en LCI. Il interprète le LCI en effectuant un routage entre services ou en répercutant chez le Correspondant utilisateur des actions déclenchées par les commandes venant des services.

Il envoie des commandes aux services par suite des actions engendrées par le Correspondant utilisateur.

Le Correspondant a donc à la fois un rôle d'interprète double pour le LCE et le LCI, de concentrateur de terminaux et de carrefour central pour tous les échanges internes au CCR. C'est lui qui fait la liaison entre les

utilisateurs et les services, entre l'extérieur et l'intérieur. Il connait la logique du CCR et fournit à l'opérateur les moyens de le contrôler. Par contre il n'a pas la vision physique du CCR.

2.2.5. - Voies Logiques de Communication :

C'est par elles que passent les communications entre entités du CCR. Les informations qu'elles transportent leurs sont transparentes. Leur rôle est de donner aux entités une vision particulière de leurs liaisons et de garantir que les informations véhiculées arrivent à destination avec un taux d'erreur minimal et un contrôle du flux des données échangées.

2.2.6. - Rapports entre les entités :

Il y a deux types de rapport bien précis: les dialogues et les échanges de données.

- Les dialoques : Comme nous l'avons vu ils se font entre utilisateur et Correspondant utilisateur en LCE, et service et Correspondant service en LCI. Les rapports sont interactifs.

- Les échanges de données : Ils peuvent se faire entre

. utilisateur - utilisateur
. utilisateur - service
. service - service.

Dans ce cas les entités sont l'émetteur (celui qui envoie) et le destinataire (celui qui reçoit). Ces échanges se font par l'intermédiaire du dialogue puisqu'utilisateurset services ne connaissent et ne s'adressent qu'au Correspondant. C'est à dire que le LCE et le LCI doivent offrir des commandes du style: transmettez ces données à telle entité. C'est au Correspondant qu'il est demandé de faire la transmission.

- Utilisateur - utilisateur :

Messages échangés entre utilisateurs sous forme de chaine de caractères.

- Utilisateur - service :

L'utilisateur appelle le service (une commande d'appel est nécessaire en LCE). Le Correspondant utilisateur vérifie la syntaxe de la commande, que l'utilisateur peut accéder au service et répercute l'appel chez le Correspondant service. Celui-ci se met alors en relation avec le service concerné et lui envoie une requête LCI. Le service la prend en compte et effectue les actions nécessaires. Tant que l'appel n'est pas fini (il faut une commande de fin d'appel en LCE) utilisateur et service s'envoient mutuellement des données grâce aux commandes de transfert LCE et LCI.

- Service - service :

Comme tous deux parlent en LCI le Correspondant service n'a qu'à faire un routage. Mais les deux services ne s'adressent pas l'un à l'autre en LCI. Il se trouve que l'un reçoit le LCI de l'autre parce que tous deux dialoguent avec le Correspondant dans le même langage. Ce qu'ils échangent ce n'est pas le LCI mais le texte transporté dans les commandes. Ces échanges se font dans le cas ou un service s'adresse à la gestion de fichier pour obtenir un fichier qui lui est nécessaire ou à la gestion d'appareils du type lecteur ou imprimante pour effectuer des entrées/sorties, suite à une demande utilisateur.

La différence entre dialogue et échange de données est fondamentale comme le montre une analogie avec la Poste.

.Le dialogue: c'est la conversation téléphonique, le téléphone étant la voie logique de communication. Les deux interlocuteurs se parlent directement dans la même langue.

.L'échange de données: c'est le courrier. Le Correspondant est la poste où l'expéditeur porte sa lettre dans une enveloppe sur laquelle est écrite l'adresse du destinataire. Le texte de la lettre ne concerne que le destinataire et la poste ne s'en occupe pas, mais elle est capable de comprendre l'adresse et elle seule a les moyens (tri, transport par train, avion, bateau, distribution par le facteur) de faire parvenir la lettre à destination.

CHAPITRE 3

ARCHITECTURE DU CENTRE DE CALCUL REPARTI

3.1. - HISTORIQUE DE L'ETUDE

Nous avons vu la structure logique du Centre de calcul réparti. Nous disposons d'un certains nombre de matériels: terminaux, ordinateurs connectés (ou connectables) entre eux et à un réseau externe (Cyclades). Il faut donc maintenant expliciter l'application du graphe logique sur le graphe physique en tenant compte de la localisation des entités et du fait que nous ne voulons pas réécrire tout le logiciel de chaque calculateur. Nous l'avons déjà dit pour les services: pour ceux fournis par le constructeur ou existants déjà sous un système donné nous nous contentons d'écrire leur enveloppe réseau.

Pour bien comprendre l'architecture choisie un petit historique de la recherche dans le domaine de la télé-informatique à l'Ecole des Mines de Saint Etienne est nécessaire [Chambon, GEN-6].

Disposant de terminaux et d'un calculateur Phillips P.1175 l'objectif était l'accès au réseau Cyclades par ces matériels et de permettre :

1 - Aux terminaux locaux d'accéder aux services distants disponibles sur le réseau.
2 - Aux terminaux distants connectés au réseau d'accéder aux services locaux disponibles sur l'ordinateur du site.
3 - Aux terminaux locaux d'accéder indifféremment aux divers services locaux.

Ce fut réalisé grâce à un Frontal, c'est à dire en interposant un mini-calculateur (un Télémécanique T.1600 en l'occurence) entre les terminaux, le P.1175 et Cyclades. La gestion de tous les accès étant déportée sur ce Frontal [Chambon, GEN-7].

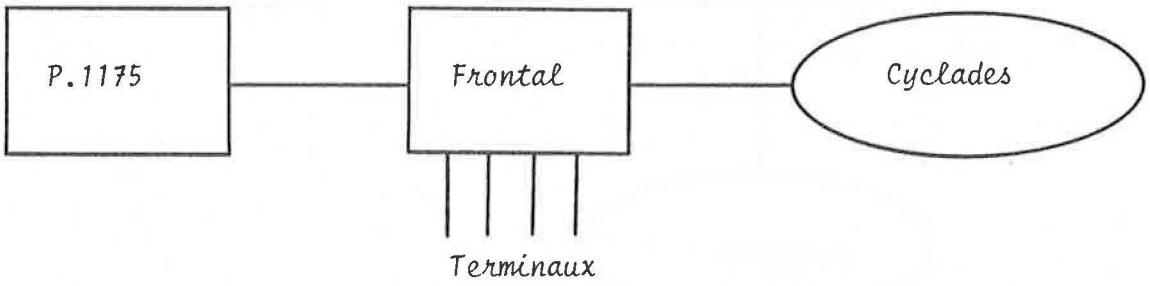


Figure 17

D'autres ordinateurs ayant été acquis par l'Ecole il apparut intéressant de les connecter aussi au Frontal pour profiter des accès déjà réalisés. Le T.1600 concentrerait ainsi les logiciels de communication avec chaque ordinateur local et le logiciel de communication avec le réseau externe et serait utilisé:

- par chaque ordinateur local en frontal du réseau,
- par le réseau en frontal de chaque ordinateur local,
- par chaque ordinateur local en frontal des autres ordinateurs du site,
- par les terminaux pour accéder le réseau ou les ordinateurs locaux.

Ainsi naquit l'idée du CCR puisqu'on aboutissait à un ensemble de services offerts par les ordinateurs locaux et Cyclades accessibles les uns par les autres et par les terminaux.

L'architecture du CCR est donc basée sur un réseau de mini-ordinateurs dont un, le Frontal, a une fonction particulière de concentrateur-diffuseur de terminaux (locaux ou distants) et de plaque tournante. Tous les autres ordinateurs et le réseau Cyclades sont connectés au Frontal (ce qui n'exclut pas des lignes particulières entre certains des calculateurs).

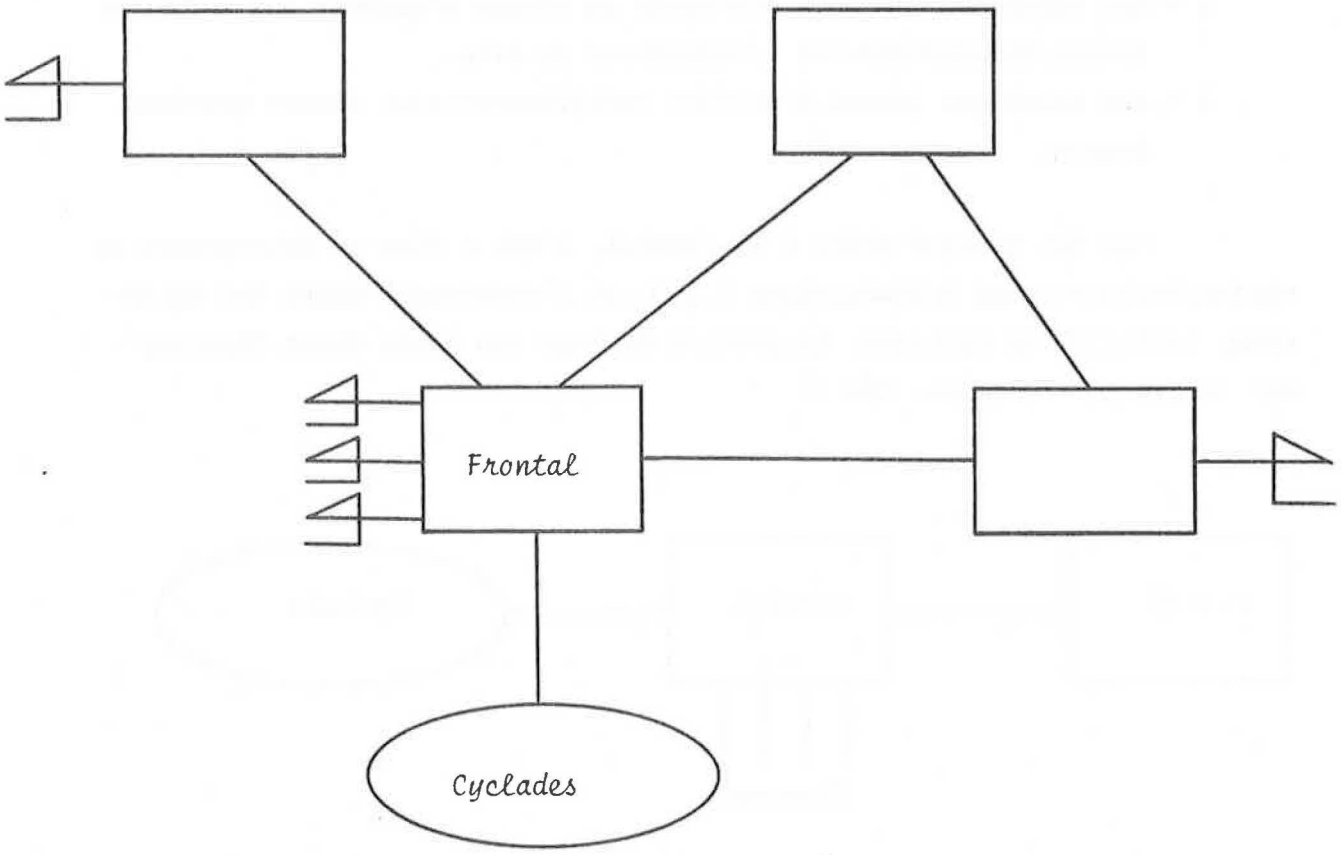


Figure 18

3.2. - LE FRONTAL

3.2.1. - Techniques de base et concepts :

Pour satisfaire aux fonctions du Frontal initial un système spécialement adapté avait été entièrement réalisé sur le T.1600: NOS (Network Operating System).

Nous partons donc d'une base solide pour réaliser le Frontal du CCR puisqu'une grande partie du logiciel existe ainsi que la définition de certains concepts.

- NOS assure en effet la gestion des Tâches et des interruptions.

- Sur le calculateur de base a été réalisé un "calculateur étendu" qui fournit un sur-ensemble des possibilités du T.1600. Ce "calculateur étendu" offre les fonctions du calculateur de base et, par logiciel, des fonctions plus évoluées:

  . gestion dynamique de la mémoire principale
  . gestion d'une mémoire auxiliaire sur disque
  . gestion des files d'attente
  . gestion des réveils
  . fonctions élémentaires diverses (décalage, transfert, positionnement de bits dans une file, etc ...)

Ces possibilités facilitent la programmation du Frontal.

- Les logiciels de gestion de différents coupleurs et plusieurs procédures de transmission gérant les lignes sont écrits.

Il faut maintenant définir le concept de machine [Chambon, GEN-6] qui est un élément de base dans la description du Frontal.

Une machine est un ensemble fonctionnel spécialisé dans l'exécution d'une fonction (plus ou moins complexe). Toute machine se situe à un niveau hiérarchique donné. Elle reçoit des commandes des machines de niveau supérieur et leurs signale des évènements. Pour exécuter ces commandes elle peut elle même adresser des commandes aux machines de niveau inférieur dont elle

est également succeptible de recevoir des évènements. Enfin, elle est de nature séquentielle et possède un état courant qui évolue depuis un état initial.

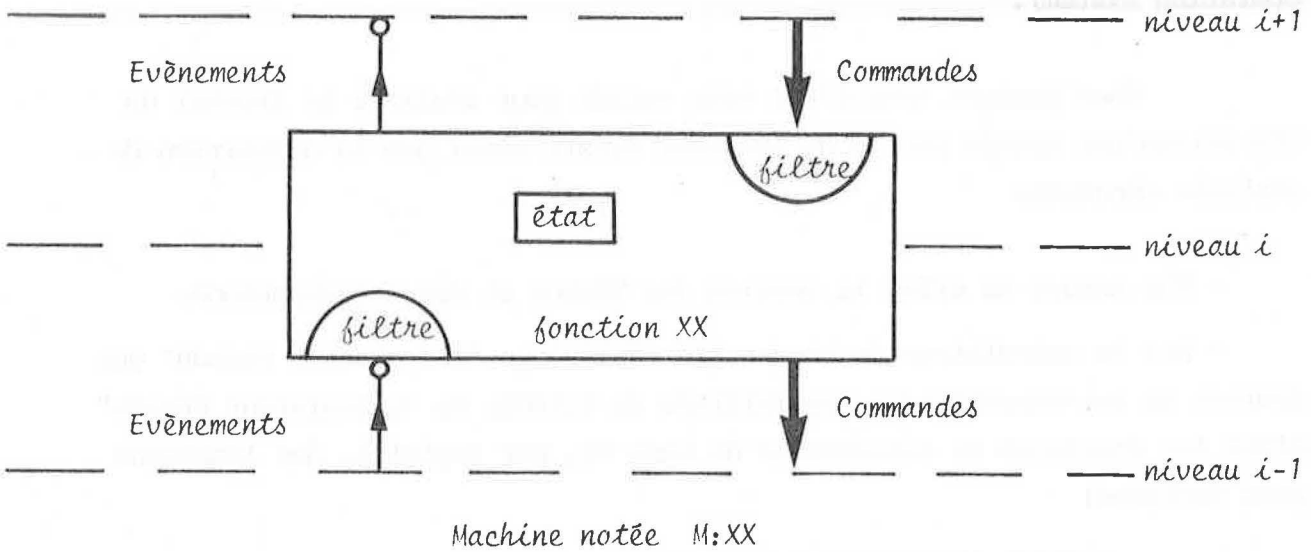


Figure 19

Lorsqu'une machine réalise une fonction complexe, elle peut elle même renfermer des machines plus élémentaires réalisant chacune un sous-ensemble de la fonction. Ces machines élémentaires s'ignorent et évoluent parallèlement au même niveau, sous le contrôle d'un aiguilleur interne à la machine englobante, lorsque la fonction globale est l'union de fonctions indépendantes de même niveau. Elles coopèrent, en propageant une action à des niveaux hiérarchiques internes successifs, lorsque la fonction globale se décompose en une série de fonctions élémentaires successives. Ces machines élémentaires sont inconnues hors de la machine englobante.

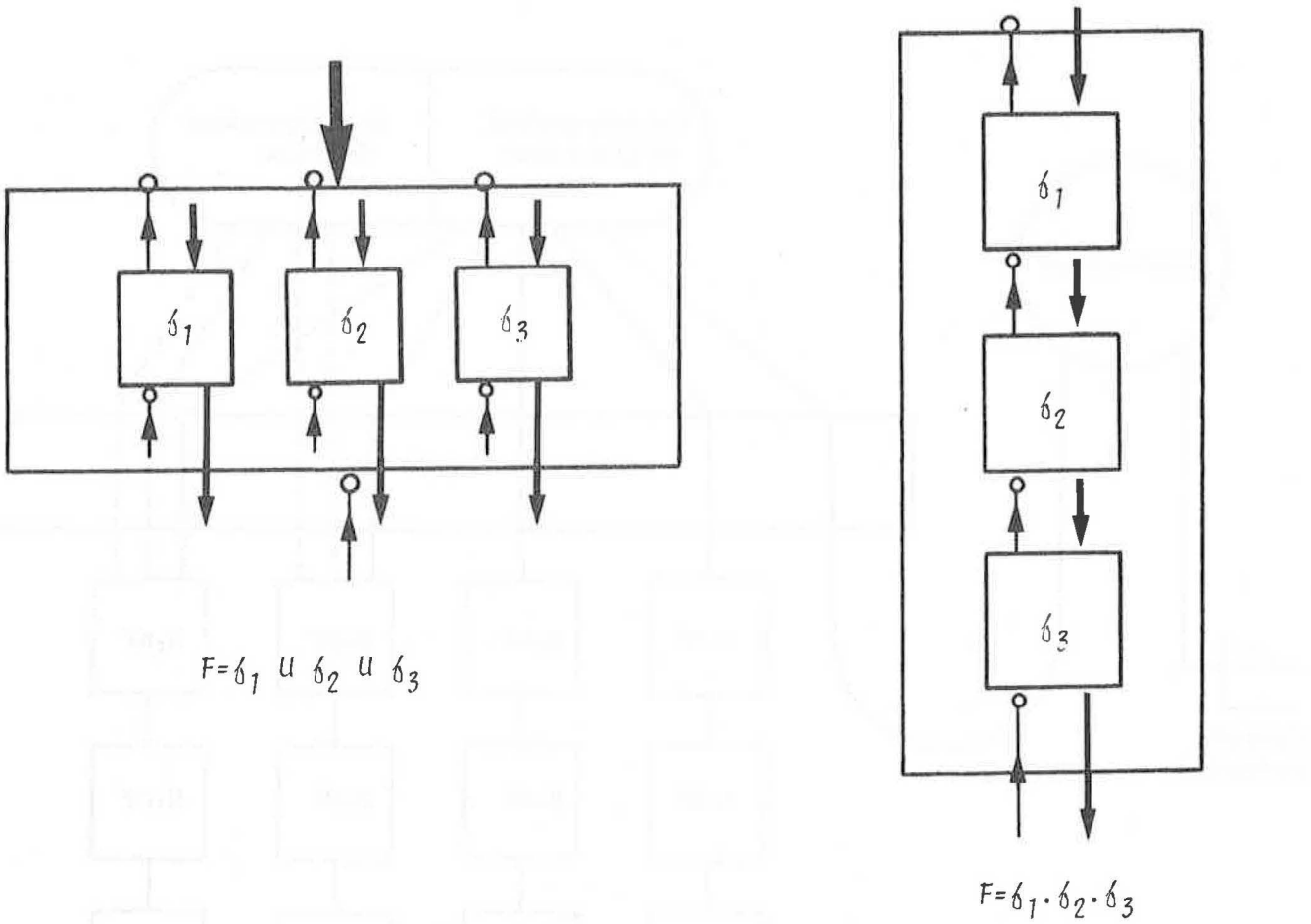


Figure 20

3.2.2. - Architecture du Frontal : [CCR-3]

Tout naturellement le Correspondant se trouve sur le Frontal. Pour communiquer avec les utilisateurs et les services il émet et reçoit sur des Unités Symboliques (SU) qui représentent la vision qu'il a des voies logiques de communication.

Ces SU sont banalisées: il existe un moyen standard de faire des entrées/sorties sur elles, qu'il s'agisse de SU.Terminal (utilisateur) ou de SU.Service. Ceci pour respecter la logique puisque le Correspondant ne désire que dialoguer avec les utilisateurs et les services et n'a pas à se préoccuper des accès qui sont à la charge des voies logiques de communication (cf. Chapitre 2). Ces entrées/sorties sont gérées par une machine appelée M:DM (Machine Data Management).

Cette machine se compose de cinq niveaux hiérarchiques qui sont:

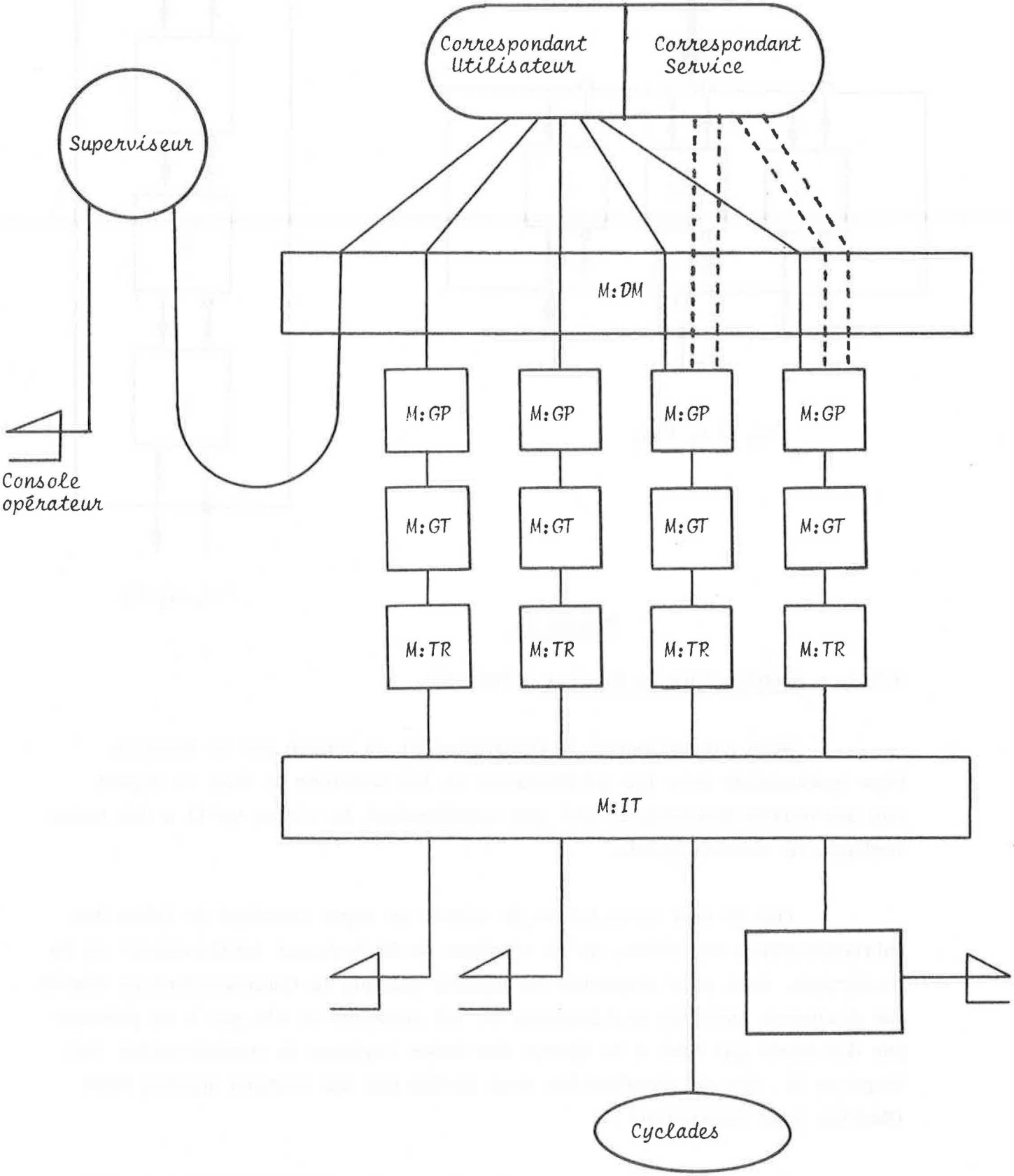

Correspondant Utilisateur
Correspondant Service
Superviseur
M:DM
M:GP
M:GP
M:GP
M:GP
M:GT
M:GT
M:GT
M:GT
M:TR
M:TR
M:TR
M:TR
Console opérateur
M:IT
Cyclades

Figure 21

1 - Niveau gestion du bus d'entrée et des interruptions (M:IT).

2 - Niveau ligne: ensemble de machines (Machine de Transmission M:TR) qui gèrent chacune un type de coupleur d'entrée/sortie.

3 - Niveau procédure: ensemble de machines (M:GT) qui sont chargées des échanges de donnée avec un appareil connecté en respectant une procédure de transmission. Elles fixent les conventions d'échange à travers le médium de transmission.

4 - Niveau protocole: ensemble de machines (M:GP) gérant chacune un protocole de communication.

5 - Niveau multiplexage / démultiplexage: fait correspondre une ou plusieurs SU à une "Unité Fonctionnelle" (FU).

Une Unité Fonctionnelle est liée à une ligne, c'est un processus chargé de gérer cette ligne. Elle englobe un ensemble de trois machines (M:GP , M:GT , M:TR). Elle gère une (ou deux) file(s) d'attente (suivant qu'elle est half ou full duplex). Elle permet de la sorte une désynchronisation dans l'émission ou la réception de données sur une ligne, entre la cadence propre de cette ligne et les demandes d'entrées/sorties du Correspondant.

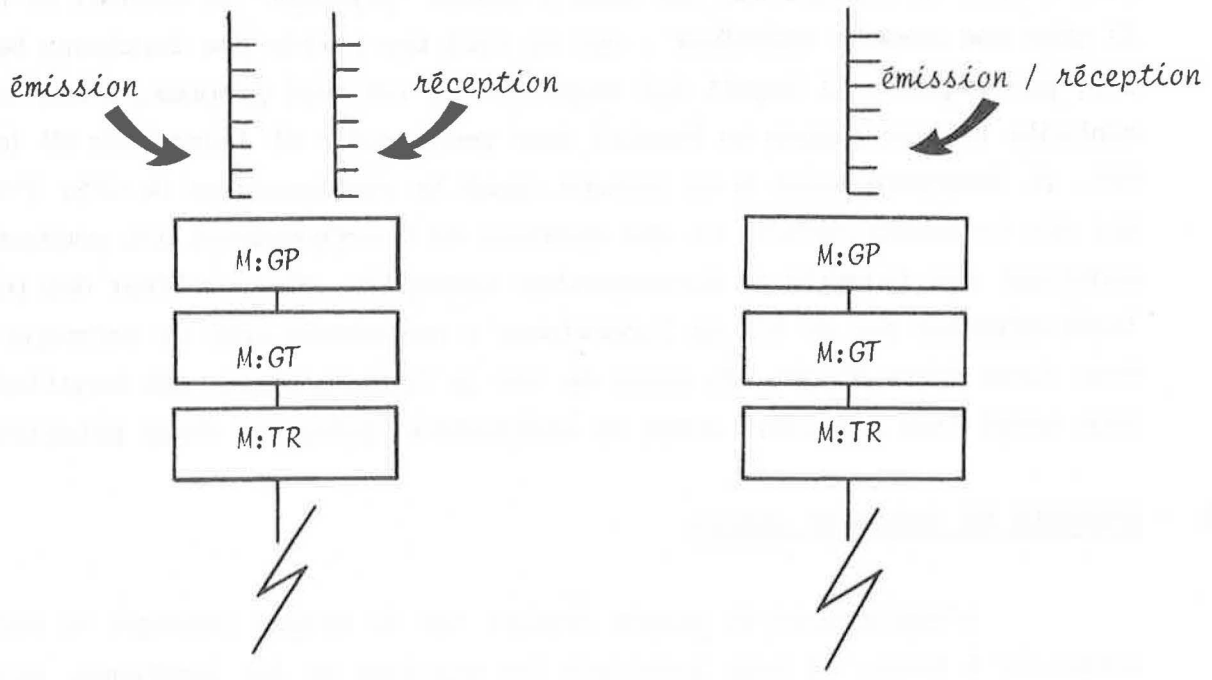


Figure 22

Pour le Frontal l'ensemble des FU représente le moyen de communication avec les appareils physiques du CCR. On a donc :

- Une FU par terminal connecté au Frontal.
- Une FU par ordinateur local.
- Une FU par réseau externe accessible.

Les M:TR et M:GT sont récupérées du Frontal initial. Mais il faudra en écrire de nouvelles lors de l'introduction d'un coupleur d'un type nouveau ou d'une nouvelle procédure de transmission.

La grosse diffèrence se situe au niveau des protocoles. Dans le Frontal CCR leurs gestions sont toutes descendues sous M:DM (dans les M:GP), alors que dans le Frontal initial certaines se trouvaient au dessus de la gestion des entrées/sorties sous forme de serveur comme par exemple le serveur Cyclades (gérant le protocole d'appareil virtuel et le protocole ST de Cyclades) ou le serveur "traitement par lot" du P.1175. Au dessus de M:DM ne reste plus à présent que le Correspondant qui s'occupe des dialogues et des langages de commande. Tous les protocoles d'accès, de transmission, d'appareil, etc ... sont gérés dans les voies logiques de communication.

Le Correspondant n'a qu'une vue et un contrôle logique du CCR. Il existe donc un Superviseur qui gère l'aspect "physique" du Frontal et du CCR. Il gère une console opérateur , qui ne fait pas partie des terminaux banalisés, par laquelle il reçoit des requêtes qui lui sont propres. C'est lui qui contrôle les connexions au Frontal donc peut ouvrir et fermer les SU (et les FU). Il détermine ainsi à un instant donné la configuration du CCR: l'ensemble des terminaux ouverts et des services en fonctionnement (en pouvant la modifier) sur laquelle le Correspondant travaille. Pour profiter des possibilités offertes par le LCE ce Superviseur a une entrée chez le Correspondant sous forme d'une SU qui, du point de vue du Correspondant, est banalisée. Il peut ainsi être considéré comme un utilisateur (avec une forte priorité).

3.3. - ENSEMBLE DU CENTRE DE CALCUL

L'application du graphe logique sur le graphe physique va surtout consister à savoir où sont localisés les services et les terminaux, et que

recouvrent comme moyens physiques et logiques de liaison les voies logiques de communication.

Un service (dans le cas d'un service réparti nous ne considérons que le maître) peut être:

- Localisé entièrement sur le Frontal.
- Localisé en partie sur le Frontal et en partie sur un autre ordinateur (y compris un réseau externe). Dans ce cas c'est en général l'enveloppe réseau qui "déborde" sur le Frontal. La ligne fait alors partie du service et est inconnue de la voie logique.
- Entièrement localisé sur un autre ordinateur ou sur un réseau externe.

Suivant les cas les voies logiques de communication seront plus ou moins "longues" et plus ou moins complexes.

Pour la voie de communication (qui correspond à la FU du Frontal) on retrouve sur les ordinateurs locaux les mêmes niveaux que sur le Frontal, à ceci près que la gestion du coupleur de transmission, ainsi que la gestion de procédure de ligne utilisée sont en général groupées sous le vocable de "méthode d'accès".

Le protocole de communication est (suivant le degré d'ouverture, vis à vis du monde extérieur, de l'ordinateur connecté) soit partie intégrante du système d'exploitation (auquel cas la machine M:GP du Frontal s'y conforme) soit construit de toutes pièces sous forme d'un programme utilisateur (ce qui ne modifie pas le système d'exploitation).

Le cas d'un réseau externe comme Cyclades est un peu particulier. Disons que globalement il est considéré par le Frontal comme un ordinateur offrant un protocole de communication avec l'extérieur. Nous l'étudierons plus précisement par la suite.

Pour le Frontal initial un des objectifs était de lui faire gérer tous les terminaux locaux, ce qui impliquait qu'ils lui soient tous connectés. Ceci afin que la gestion des terminaux soit concentrée et que les autres calculateurs en soient complètement déchargés. Nous en sommes revenus à une

connexion purement logique au Correspondant utilisateur, la gestion physique des appareils étant laissée à leurs ordinateurs respectifs. A cela il y a plusieurs raisons:

- Avec Cyclades il existe de toute façon des terminaux distants qu'il faut apprendre à gérer en tant que tels. Cette gestion étant obligatoire il est possible de l'utiliser pour les terminaux connectés aux autres ordinateurs locaux. Il faut remarquer que les utilisateurs qui accèdent le CCR par Cyclades le voient comme un ordinateur dont le langage de commande est le LCE. C'est à dire qu'avant d'être considérés comme des utilisateurs-CCR ils doivent se connecter à lui par les moyens qu'offrent leurs sites respectifs. Ainsi il faut bien être conscient que les objectifs définis au Chapitre 2 ne sont valables que pour les utilisateurs locaux du CCR. Les autres sont soumis aux contraintes imposées par Cyclades et le centre de calcul dont ils sont clients.

- Si le Frontal tombe en panne il ne faut pas que toute exploitation cesse. Avec leurs terminaux les ordinateurs peuvent se remettre à travailler en local. On peut aussi faire fonctionner le CCR en mode dégradé en attribuant le rôle de Frontal à un autre ordinateur (il est nécessaire de prévoir un logiciel de secours) sans être obligé de lui reconnecter physiquement tous les terminaux du centre (ce qui n'est pas facile). Les terminaux du Frontal en panne ne sont plus utilisables mais les autres le restent.

- Certains calculateurs sont livrés avec leur terminaux (parfois spécialisé comme la console opérateur) et le système de gestion correspondant. C'est faire un travail inutile que de réécrire toute cette gestion sur le Frontal si ce terminal n'est pas d'un type déjà connu de lui.

3.4. - SUITE DE L'ETUDE

Actuellement le CCR comprend : le Frontal (un Télémécanique T.1600), un Philipps P.1175, un Digital Equipement PDP.11, un SEMS Solar 16 et l'accès au réseau Cyclades. Mais les solutions définies et retenues peuvent s'appliquer à tout nouvel ordinateur ou réseau externe que l'on voudrait connecter. Notamment,bien que nous nous soyons orientés vers les mini-ordinateurs pour

les raisons exposées au Chapitre 2, rien n'empêche l'introduction d'un gros calculateur dans le CCR.

Dans les chapitres suivants nous allons étudier plus en détail tout ce qui se rapporte aux communications :

. Les langages de commande LCE et LCI ainsi que leur interprétation et les dialogues qu'ils permettent.

. Les voies logiques de communication et les différents protocoles qu'elles gèrent.

DEUXIEME PARTIE :

LES LANGAGES DE COMMANDE

CHAPITRE 4

LANGAGE DE COMMANDE EXTERNE

4.1. - DEFINITION DU LCE

4.1.1. - Rappel des objectifs :

Le LCE est le langage dans lequel un utilisateur s'adresse au Correspondant utilisateur. C'est à travers lui, par sa connaissance, qu'est vu le CCR de l'extérieur. Ainsi, par une bonne définition du langage, on ne fournit à l'utilisateur que les services et les fonctions qui lui sont indispensables. L'ensemble des systèmes sous-jacents nécessaires à la bonne gestion des services lui est alors transparent [Watson, GEN-8].

Que veut l'utilisateur ? S'installer à un terminal, moyen physique d'accès au CCR, et faire effectuer un certain nombre de travaux à des services, en ayant éventuellement recours à des ressources supplémentaires (imprimante, lecteur, fichier, etc ...).

Nous devons donc lui fournir le moyen d'entrer logiquement sous le système en se faisant reconnaitre du Correspondant utilisateur (Login avec numéro de compte ou identificateur) et de sortir du système (Logout).

Il doit pouvoir appeler les services offerts par le CCR, leur envoyer des données et en recevoir. C'est ce que l'on peut appeler les requêtes de soumission de travaux. Ce sont elles qui vont entrainer des répercutions d'action entre Correspondant utilisateur et Correspondant service avec traduction du LCE vers le LCI. Mais le CCR est un système conversationnel. Le Correspondant utilisateur ne se contente pas d'un simple rôle de "passeur" entre l'extérieur et le Correspondant service. Il peut mettre à la disposition des utilisateurs, suivant leur classe, des fonctions qu'il est capable de traiter entièrement sans avoir besoin du Correspondant service.

Nous l'avons vu au paragraphe 2.2.6 il est intéressant qu'un usager puisse échanger des messages avec un autre utilisateur. Il peut aussi désirer avoir des informations sur sa connexion au système: nombre de connexions, temps de connexion (et la facture qui en découle), les services qui lui sont accessibles.

Les utilisateurs de classes supérieures doivent disposer de toute une

série de fonctions particulières. L'opérateur, par exemple, désire contrôler tout ou partie de la logique du CCR: qui est connecté, quels services sont en marche, quels sont les terminaux ouverts, ... et modifier cette logique: tuer un autre utilisateur, débloquer une SU. Le Manager du Frontal a besoin de fonctions de maintenance pour visualiser les états des machines, faire tracer certaines informations et même pour décharger une partie de la mémoire ou modifier le contenu de certaines adresses.

Certaines sessions sont toujours les mêmes, il est donc inutile et fastidieux d'obliger l'utilisateur à retaper plusieurs fois la même suite de lignes. Il est intéressant de fournir la possibilité de définir et utiliser des procédures cataloguées.

4.1.2. - Choix du langage :

Notre problème n'est pas de définir de nouvelles formes syntaxiques et sémantiques. La structure du LCE a finalement peu d'importance. Nous ne lui imposons comme contrainte que d'être facilement lisible et, le CCR étant succeptible de développement, d'être aisément extensible: par l'ajout de nouveaux services sitôt que ceux-ci deviennent opérationnels et même de nouvelles fonctions.

Quand on veut définir un langage le plus simple est souvent d'essayer d'en trouver un dans la littérature, chez un constructeur ou une autre équipe de recherche, qui réponde en grande partie aux objectifs et contraintes que l'on s'est fixé. On l'adapte ensuite à son cas particulier.

Le LCE est du type langage de commande réseau. Or nous avons déjà vu que plusieurs études ont été faites sur ce sujet. Le NJCL (Network Job Control Language) [Raymond, LAN-1] proposé pour un réseau hétérogène est dérivé de Pascal. Il a donc une forme de langage de haut niveau et peut être facilement appri et manipulé par des usagers courants, souvent désorientés par les langages de commande. Le NCL de SOC [Du Masle, LAN-2] et le LE développé pour Cyclades [Sergeant, LAN-3] présentent tous deux la similitude d'être traduit en un langage interne au réseau pour être exécutés. Notre solution nous a conduit à deux langages de commande, l'un externe et l'autre

interne. Il était donc tentant de prendre le NCL ou le LE Cyclades comme LCE, surtout le LE puisque nous désirions aussi accéder à Cyclades.

En fait ces langages de commande sont des langages très puissants, permettant une programmation complexe, dont les fonctions débordent largement le simple accès à des services. Leur objectif est beaucoup plus l'écriture d'application réseau en fournissant les moyens de faire coopérer les services existant sur les ordinateurs du réseau que de les mettre à la disposition des utilisateurs [Goyer, LAN-5].

On voit donc, outre le fait que leur interprétation sur un mini-calculateur comme le Frontal risquait de ne pas être simple, qu'ils ne répondent pas exactement aux buts que nous nous sommes fixés.

Nous avons préféré créer un LCE de toutespièces selon une approche descendante, c'est à dire qu'il a été défini en fonction des objectifs à atteindre et non pas en partant des langages de commande existants sur les ordinateurs du site (nous reverrons plus tard ce problème à propos du LCI). Ce travail est simplifié par le fait que notre CCR recouvre un "petit" réseau de mini-ordinateurs et que le LCE est orienté dans une optique particulière, alors que les autres solutions vues dans ce paragraphe tentent de résoudre de façon globale le problème du langage de commande pour un réseau général important, tâche qui se révèle fort ardue [Nagel, LAN-6].

4.1.3. - Forme du langage :

Le LCE est un langage interprétatif et interactif [Laforgue, GEN-9].

- Interprétatif parce que toutes les commandes sont interprétées.
- Interactif parce qu'il permet à l'utilisateur de dialoguer avec le Correspondant utilisateur.

Nous lui donnons une forme simple: une suite de requêtes encadrée par un début de session et une fin de session.

Parmi ces requêtes les commandes doivent avoir une définition syntaxique très générale: un nom de commande suivi d'une liste de paramètres.

Nous avons vu que les éléments de base utilisables par l'usager sont: les fonctions, les services et les procédures. Ils sont donc désignés chacun par un nom externe unique connu de l'utilisateur. Pour les procédures c'est évident puisque c'est l'utilisateur lui même qui leur attribue un nom.

Les services, indépendamment de leur localisation, apparaissent ainsi de façon standard aux usagers. Pour faciliter cette transparence et la lecture, les demandes de services pourront se faire sous forme d'appels procéduraux en ne tapant que le nom du service. C'est une aide à l'écriture car la fonction appel de service, avec le nom du service comme paramètre, existe.

On peut dire que l'ensemble de ces noms de commande et les paramètres possibles définissent l'espace d'exécution accessible à l'utilisateur [Guiboud Ribaud, GEN-10].

Enfin il ne faut pas oublier que l'utilisateur s'adresse toujours au Correspondant utilisateur et parle constamment en LCE. C'est à dire que lorsqu'il appelle un service il reste sous le contrôle du Correspondant qui se contente de marquer dans l'environnement utilisateur que ce dernier a demandé un service. Le LCE offre une requête qui donne le moyen à l'utilisateur de dire au Correspondant utilisateur: "Transmettez au service que j'ai appelé cette suite ce caractères", cette suite étant transparente au Correspondant. Nous n'avons donc pas comme pour le concentrateur CT.2 de Cyclades [Weber, RES-7] des requêtes locales ou distantes, avec la possibilté pour l'usager d'avoir deux interlocuteurs potentiels: le concentrateur ou une application. Dans notre CCR l'utilisateur n'a qu'un seul interlocuteur: le Correspondant utilisateur. Comme pour l'utilisateur du CT.2 il doit savoir qui va interpréter ce qu'il tape sur son terminal et c'est le LCE, par sa syntaxe, qui le lui permet. Mais cette interprétation est fondamentalement différente suivant les cas: il dialogue avec le Correspondant utilisateur; il fait exécuter un travail par le service (même si cela se réduit à une instruction avec un service conversationnel) et attend le résultat de ce travail.

4.2. - STRUCTURE DU LCE

4.2.1. - Syntaxe :

La notation employée est la forme normale de Backus. De plus :

Une unité syntaxique entre crochets [] est facultative.

Une unité syntaxique encadrée par des accolades suivie d'une astérisque {}* peut apparaitre zéro ou plusieurs fois.

. Le vocabulaire :

<caractère>::= tout caractère ASCII. Il faut faire attention au blanc qui est un caractère séparateur (seul ou répété on le notera ␣).

<lettre> ::= A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z

Il est à noter que les lettres peuvent être minuscules ou majuscules,cela n'a pas d'importance. Les minuscules sont converties en majuscules avant l'interprétation.

<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Il existe des caractères spéciaux appelés caractères de contrôle:

(AC) Appel Correspondant

(E1) évènement 1

(E2) évènement 2

(SC) Suppression de caractère

(SL) Suppression de ligne

(FL) Fin de ligne

Leur codage dépend de la gestion du terminal. Le Correspondant utilisateur ne les reçoit pas en tant que caractère. (SC) , (SL) et (FL) ne sont même pas interprétés à son niveau mais par le gestionnaire du terminal. Les cinq premiers

sont modifiables par l'utilisateur (quand c'est possible). Pour les terminaux connectés au Frontal les codes standards de ces caractères sont:

(AC) Escape

(E1) Contrôle A

(E2) Contrôle Y

(SC) Back Space

(SL) ←

(FL) Carriage Return

. Chaines et nombres :

<nombre> ::= <digit>{<digit>}*

<chaine> ::= <caractère>{<caractère>}*

<letdig> ::= <lettre> | <digit>

. On peut maintenant définir une session de travail :

<Session> ::= <Début session> <Corps session> <Fin session>

<Début session> ::= (AC) L[OGIN] ␣ <identificateur> (FL) <Mot de passe> (FL)

<identificateur> ::= <nombre>

<Mot de passe> ::= <chaine>

<Corps session> ::= {<Ligne>}*

<Ligne> ::= (AC) <Ligne AC> | <Ligne E> | (E1) | (E2)

Nous voyons apparaitre la distinction entre commande et requête de transmission dont nous avons souligné au paragraphe 4.1.3 qu'elle peut être faite par la syntaxe.

. Requête de transmission :

(E1) et (E2) sont des évènements à envoyer au service appelé et dont la

signification dépend du service.

<Ligne E> ::= <data> (FL)

<data> ::= {<caractère>}*

C'est la suite de caractères que l'utilisateur demande au Correspondant utilisateur d'envoyer au service et qui n'a de signification que pour ce dernier.

. Commande :

<Ligne AC> ::= <Commande> (AC) <Ligne AC> | <Commande> (FL)

<Commande> ::= <fonction> | <appel service> | <appel procédure>

On retrouve ainsi les trois éléments de base fournis à l'usager: fonction, service, procédure.

<fonction> ::= <nom fonction>{␣ <param>}*

<nom fonction> ::= <lettre> <lettre> [{<lettre>}*]

Pour faciliter l'écriture et l'interprétation les deux premières lettres du nom d'une fonction sont suffisantes pour la reconnaitre.

<appel service> ::= <nom service>{␣<param>}*

<nom service> ::= <lettre>{<letdig>}*

<appel procédure> ::= <nom procédure>{␣<PP>}*

<nom procédure> ::= <lettre>{<letdig>}*

. Paramètres :

<param> ::= <PP> | ([␣]{<PMC>␣}* <PMC> [␣])

<PP> ::= <paramètre>

<PMC> ::= <mot clé> [␣] = [␣] <paramètre>

<mot clé> ::= <lettre>{<letdig>}*

<paramètre> ::= <chaine> | <nombre>

. Et pour terminer la session :

<Fin session> ::= (AC) L[OGOUT][: (FL) <Mot de passe>] (FL)

En tapant deux points après LOGOUT l'utilisateur peut changer son mot de passe.

4.2.2. - Cas des procédures :

Il existe sur le Frontal un service dit d'entrée/sortie qui s'occupe de la gestion des fichiers-appareils. Ce service a autant d'entrées que de types d'appareils qu'il peut gérer, c'est à dire qu'il est vu des utilisateurs et du Correspondant comme plusieurs services. Chaque entrée a une fonction bien définie et toutes les entrées partagent la gestion des fichiers du Frontal. On a ainsi une entrée éditeur de texte qui permet aux utilisateurs de créer ou détruire des fichiers. Les entrées appareils (imprimante, lecteur de cartes, perforateur) utilisent soit ces fichiers permanents créés par les usagers, soit des fichiers temporaires qui sont alimentés ou vidés à l'aide des appareils physiques connectés au Frontal.

Les utilisateurs se définissent des procédures cataloguées avec l'éditeur de texte. Chaque procédure est stockée sur un fichier. Lors de l'appel d'une procédure le Correspondant utilisateur va aller lire le fichier qui la contient, ligne par ligne, par l'entrée télétype (notée TTY) comme s'il lisait les lignes venant d'un terminal.

Les procédures, même si elles ne sont que le stockage permanent d'une liste de requêtes LCE, nécessitent une syntaxe un peu particulière.

<Corps procédure> ::= [<Déclaration>] <Ligne proc> *

<Déclaration> ::= <PP>{␣<PP>}*

<Ligne proc> ::= ?<fonction> | ?<appel service> |␣ <Ligne E> | /E1 | /E2

Les paramètres sont passés par nom et il n'est pas permis d'appeler

une procédure de l'intérieur d'une autre procédure. Un (AC) venant de l'utilisateur fait sortir de l'exécution de la procédure. Le Correspondant utilisateur sait que la procédure est finie quand il a lu tout le fichier.

4.2.3. - Fonctions et mots clé :

La liste des fonctions et des mots clé se trouve en annexe, nous ne regarderons ici que quelques exemples.

Les mots clé qualifient les paramètres qui forment l'environnement de la commande.

Ainsi dans la fonction MODIF avec laquelle un utilisateur peut changer le codage des caractères de contrôle de son terminal il existe un mot clé par caractère de contrôle. Par exemple si l'usager désire que (AC) ne soit plus "escape" mais BEL il tape la commande:

MODIF (BK=87) (87 étant le code ASCII de BEL)

Lors de l'appel d'un service chaque appareil supplémentaire dont l'utilisateur a besoin pour faire effectuer un travail est identifié par un mot clé:

LP pour la sortie imprimante
LC pour l'entrée lecteur
FI pour la sortie fichier

Signalons que la fonction d'appel service est CSERV, celle de fin d'appel DSERV. Le LCE offre aussi des fonctions de création et mise à jour des utilisateurs, services et même fonctions elles-mêmes. Différents mots clé définissent les caractéristiques de ces entités.

4.3. - SEMANTIQUE DU LCE

Nous avons dit que l'utilisateur "voit" le CCR à travers le LCE, c'est à dire que sa connaissance du LCE luidit ce qu'il peut faire et, implicitement, lui masque ce qu'il ne peut pas faire. Or des utilisateurs différents n'ont pas forcément les mêmes droits ni les mêmes désirs.Ce qui intéresse un utilisateur d'application c'est soumettre un travail à un service courant.

Un programmeur, chargé d'écrire une application, a besoin de services d'assemblage et de stockage. Pour le Manager du Frontal des fonctions de maintenance sont nécessaires mais les applications simples ne lui servent pas.

Le terminal par lequel l'usager accède au CCR influe aussi sur ses possibilités. Considérons le cas de la fonction MODIF avec laquelle on peut modifier les caractères de contrôle. Ceci n'est possible que si le Correspondant utilisateur a les moyens d'envoyer au gestionnaire du terminal des ordres pour lui indiquer les caractères de contrôle à employer. C'est le cas pour les terminaux physiquement connectés au Frontal (appelés terminaux locaux) mais pas pour les terminaux distants connectés à d'autres ordinateurs locaux ou aux concentrateurs d'un réseau externe. Par exemple le CT.2 de Cyclades gère ses terminaux de façon purement local et détermine les caractères de contrôle. Plus tard on peut envisager de créer des commandes spéciales permettant au Correspondant de s'adresser aux gestionnaires de terminaux situés ailleurs que sur le Frontal. Mais deux catégories de terminaux existent : ceux dont le Correspondant peut modifier les caractères de contrôle et ceux qui sont hors de son influence. Pour les usagers travaillant avec ces derniers il est bien évident que la fonction MODIF n'a aucun sens.

Comme pour l'utilisateur le LCE est la description du système accessible et utilisable par lui,il n'y a donc pas un LCE unique dans tout le CCR mais des LCE,particuliers pour chaque utilisateur. D'ailleurs cette notion de LCE différents apparait dès l'instant où les procédures cataloguées sont intégrées au langage; en effet des usagers distincts ne se créent pas les mêmes procédures.

Les droits (la classe) et l'accès de l'utilisateur forment la priorité qui est inscrite dans son environnement. C'est donc ce dernier qui définit le LCE particulier de l'usager.

A la base nous avons une syntaxe, qui est la même pour tous, et un ensemble de commandes primitives contenant toutes les fonctions implémentées et les services existants dans le CCR. Appelons ECB (ensemble des commandes de base) cet ensemble. C'est à partir de lui, de la syntaxe et des procédures cataloguées qu'est bâti le LCE de l'utilisateur.

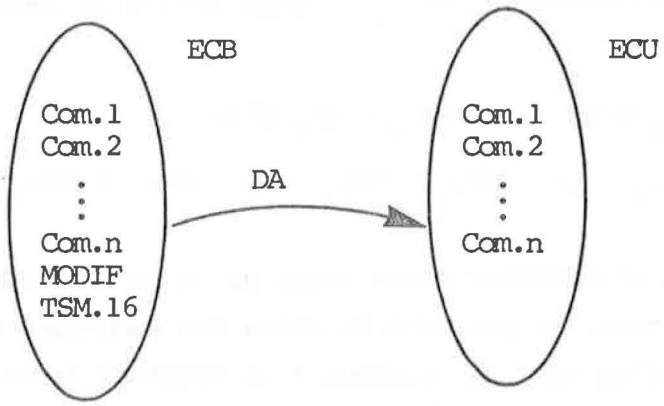
Soit : EPU l'ensemble des procédures de l'utilisateur.

DA l'application de ECB dans lui-même qui donne les commandes accessibles à l'utilisateur en fonction de sa priorité.

On a un ensemble des commandes de l'utilisateur (ECU) défini par:

$$ECU = DA(ECB)$$

Par exemple:



Un utilisateur d'application s'installe à un terminal relié à un CT.2 de Cyclades. Il n'a pas accès à TSM.16, système d'exploitation du Solar 16, et, n'étant pas sur un terminal local, ne peut pas se servir de MODIF. TSM.16 et MODIF ne se trouvent donc pas dans son ECU.

On peut dire que le langage de commande utilisateur (LCE.U) est l'union de ECU et de EPU (avec la syntaxe qui,elle,ne change pas).

En fait ce n'est pas tout à fait vrai car toutes les procédures ne sont pas exécutables. Seules peuvent l'être celles qui ne font appel qu'aux commandes appartenant à ECU.

Définissons l'application $CP : EPU \rightarrow P(ECB)$

$P \in EPU \qquad CP(P) = \{$ensemble des commandes utilisées dans P$\}$

On a: $LCE.U = ECU \cup \{P \ ; \ (P \in EPU) \text{ et } (CP(P) \subseteq ECU)\}$

On peut le présenter diffèrement en se définissant une application PE de $P(ECU) \times EPU$ dans EPU :

$X \in P(ECU)$, $P \in EPU$ $\qquad PE(X,P) = P$ si $CP(P) \subseteq X$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad = \xi$ sinon

Ce qui donne: $\qquad LCE.U = ECU \cup PE(P(ECU)\ ,\ EPU)$

Deux utilisateurs n'ont pas le même langage de commande mais peuvent aussi ne pas avoir le même ECU.

Soient deux utilisateurs U_1 et U_2 , leurs droits et accès déterminent DA_1 et DA_2.

On a: $\qquad ECU_1 = DA_1(ECB) \qquad ECU_2 = DA_2(ECB)$

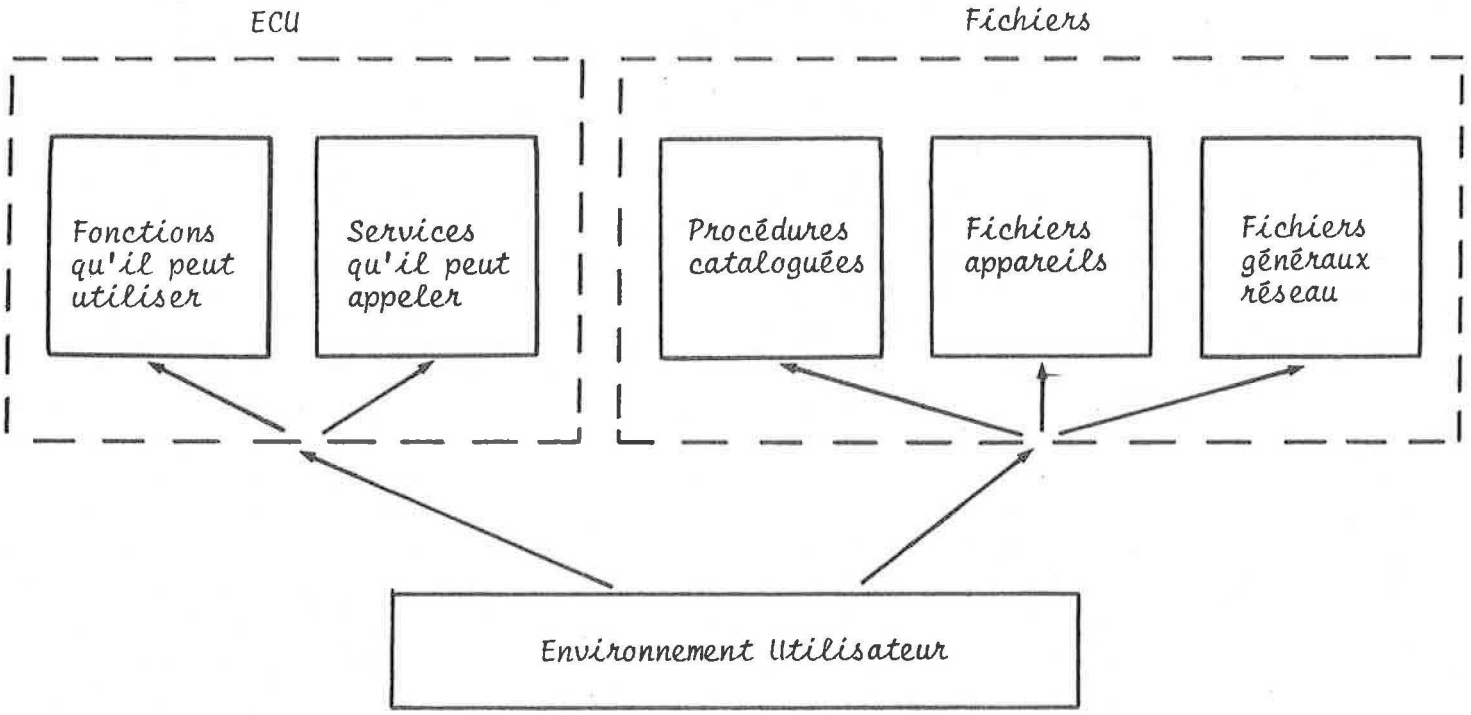
On peut avoir: $ECU_1 = ECU_2$ ou $ECU_1 \subset ECU_2$ ou $ECU_1 \oplus ECU_2 \neq \emptyset$

Le langage d'un utilisateur varie aussi au cours du temps (d'une session de travail à l'autre). Sa priorité ou celle des commandes peut être modifiée. Il peut passer d'un terminal distant à un terminal local.

Nous aurions très bien pu considérer que le LCE était le même pour tous tout le temps, mais que certaines commandes étaient interdites à un utilisateur suivant sa priorité du moment. Il ne faut d'ailleurs pas se leurrer, la réalisation est pratiquement identique dans les deux cas. Un indicateur donne la priorité de l'usager, un autre celle de la commande. Si ces indicateurs ne coïncident pas, dans un cas on dit que la commande est interdite, dans l'autre qu'elle n'est pas accessible. Mais il est intellectuellement malhonnête de dire que le LCE est le même pour tout le monde avec des restrictions particulières. Quand un utilisateur dispose d'un langage il doit avoir l'usage de toutes les commandes. Si on ne veut pas qu'il puisse se servir d'une commande bien précise il est plus logique de considérer qu'elle n'appartient pas à l'ensemble des ressources qui lui sont fournies. Il ne la voit pas et ne peut donc pas l'exécuter.

Nous voyons ainsi nettement apparaitre l'espace d'exécution défini au paragraphe 4.1.3 et qui est décrit par l'environnement de l'utilisateur.

Nous l'avons dit l'utilisateur et le Correspondant utilisateur dialoguent, ils sont interlocuteurs. Il est fondamental qu'au cours d'une session ils se comprennent, c'est à dire qu'ils parlent le même langage. Il



Les fichiers sont ceux ,privés ou partageables, qui peuvent être passés en paramètre

Figure 13

importe peu que des utilisateurs différents n'emploient pas le même langage, ce qui compte c'est que le Correspondant utilisateur le sache. Or le Correspondant utilisateur connait l'usager par son environnement et celui-ci définit le langage particulier de l'utilisateur, nous pouvons même dire qu'il lui appartient. Il n'y a donc pas de problème, pour un utilisateur donné le Correspondant utilisateur connait et peut comprendre le LCE.U correspondant. Si l'utilisateur modifie son LCE ou si ce dernier est changé par une intervention externe (de l'opérateur par exemple), cette modification est immédiatement inscrite dans l'environnement de l'usager. Le Correspondant utilisateur peut ainsi parfaitement suivre l'évolution du LCE.U et continuer à être capable de l'interpréter.

CHAPITRE 5

LANGAGE DE COMMANDE INTERNE

5.1. - DEFINITION DU LCI

5.1.1. - Fonctions à remplir :

Le LCI est le langage employé par les entités internes (Correspondant, services) pour dialoguer. Il présente deux aspects: un aspect commande et un aspect contrôle de dialogue. En effet entre le Correspondant utilisateur et un utilisateur le Correspondant utilisateur est maître du dialogue, les règles du contrôle étant définies implicitement dans le LCE. Vis à vis de l'usager le Correspondant utilisateur, quand il a la main, la garde tant qu'il veut, tandis que dès que l'utilisateur envoie une ligne il perd la main. L'utilisateur a la main quand son terminal peut émettre (le clavier est débloqué), c'est à dire lorsque le Correspondant a fait une demande de réception sur la voie logique le reliant au terminal en question. Entre le Correspondant service et un service le mécanisme est plus complexe. Donner la main à l'autre implique qu'on le prévienne, car rendre la voie logique passante dans un sens ne suffit pas. Celle-ci n'est qu'un moyen de transport, elle garantit que les messages émis sont bien arrivés, mais elle ne s'occupe pas de savoir qui a le droit d'émettre à un instant donné. Le LCI doit donc offrir un mécanisme de "à vous" permettant aux deux interlocuteurs de toujours se mettre d'accord pour savoir qui est l'émetteur et qui est le récepteur. Cette fonction est indispensable car les applications peuvent être localisées ailleurs que sur le Frontal, la voie logique de communication qui les relie au Correspondant service devient alors complexe. Alors que le correspondant utilisateur gère presque entièrement les utilisateurs, le correspondant service n'a pas lesmoyens de le faire avec les services.

Il est aussi nécessaire de pouvoir transmettre des évènements asynchrones. Lorsqu'un utilisateur veut signaler un fait au service qu'il a appelé,il tape (E1) ou (E2). Ces évènements ne peuvent pas passer par le mécanisme normal d'échange car sinon le Correspondant service serait obligé d'attendre d'avoir la main vis à vis du service pour les envoyer; or ceux-ci peuvent justement vouloir dire: "J'attends depuis longtemps, renvoyez moi la main". Ces évènements doivent donc être transmis par des commandes non soumises au mécanisme du "à vous".

Le Correspondant interprète les requêtes des utilisateurs et c'est en fonction de ces requêtes qu'il va s'adresser à telle ou telle application. Il parle à un service pour le compte d'un utilisateur. Or deux utilisateurs peuvent appeler le même service au même moment:

U_1 et U_2 , accèdant au CCR par SU_1 et SU_2 , appellent tous deux le service S.

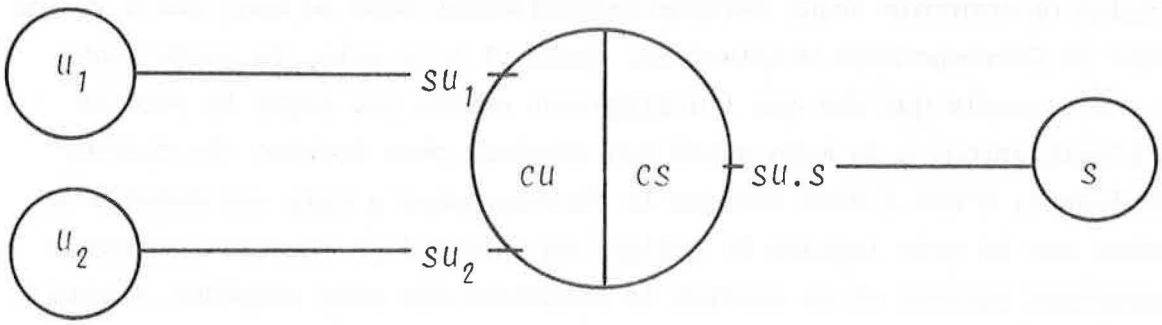


Figure 24

Le Correspondant va ouvrir deux flots d'échange ou liaisons avec le service S respectivement pour chaque usager. Ces liaisons passent par la même voie logique reliant S au Correspondant service. Il faut indiquer en LCI, seule structure d'information délivrée à chaque extrémité de la voie logique, de quelle liaison il s'agit lors d'un échange donné pour que le service ou le Correspondant fasse la différence entre U_1 et U_2.

De même l'utilisateur a la possibilité de se servir d'appareils supplémentaires lors d'un appel de service. Par exemple l'usager U soumet un travail au service S et demande que le résultat de ce travail sorte sur l'imprimante I.

Le Correspondant établit les liaisons SU.U - SU.S et SU.I - SU.S. Les liaisons sont bien distinctes mais les commandes circulant sur elles concernent le même utilisateur. Quand S envoie à la gestion de l'imprimante les lignes à imprimer, cette dernière (qui a pu être demandée par plusieurs utilisateurs) doit savoir à qui elles sont destinées. Ceci afin de regrouper

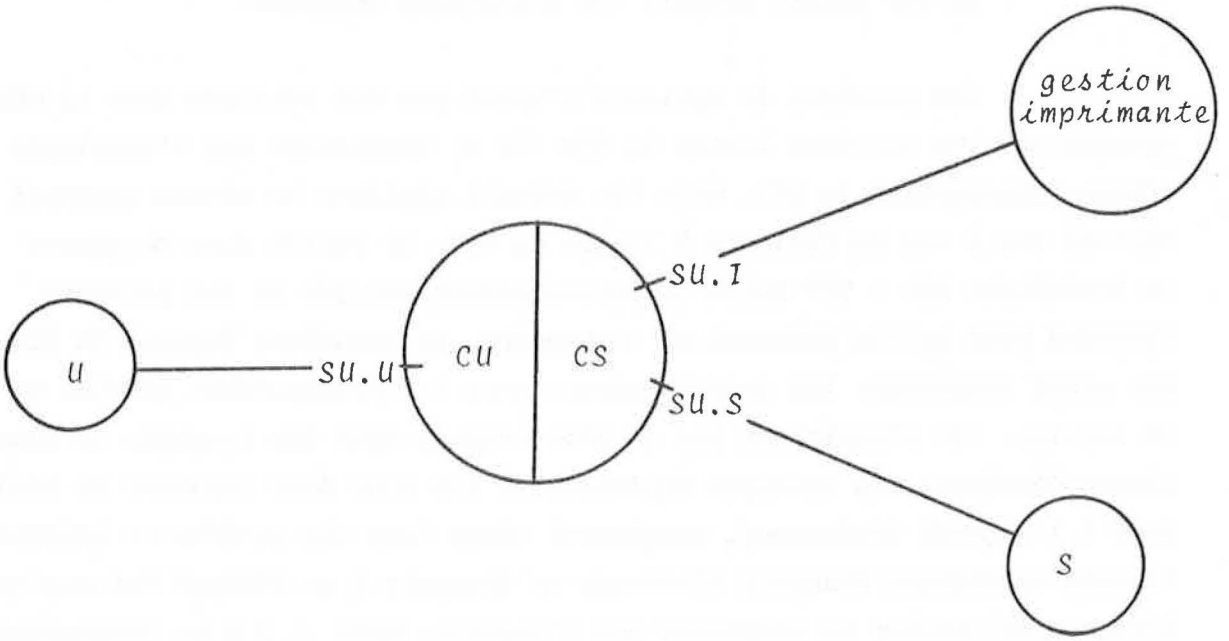


Figure 25

les lignes de chaque usager et les sortir d'un seul bloc sans les mélanger.

Le LCI offre donc un adressage qui se divise en deux:

- L'indication de l'émetteur et du destinataire. C'est l'identification de la liaison sur laquelle sont échangées les informations.

- Un indicateur permettant de reconnaitre l'utilisateur concerné par ces informations.

5.1.2. - LCI basé sur le VTP de Cyclades :

Nous l'avons déjà dit pour le LCE (cf. paragraphe 4.1.2), pour créer un langage le plus simple est de prendre comme base un langage existant qui semble répondre à nos désirs et le modifier pour que toutes les fonctions souhaitées soient réalisées.

Nous sommes donc partis du VTP (Virtual Terminal Protocol) du réseau Cyclades [Zimmermann, RES-8] pour définir le LCI. Ceci pour deux raisons:

. Le VTP parait remplir les conditions requises.

. Les services de Cyclades n'ayant pas été réalisés dans la même optique que les services locaux du CCR ils ne comportent pas d'enveloppe réseau interprètant le LCI. Mais ils doivent réaliser le niveau appareil virtuel vis à vis de Cyclades à l'aide du VTP. Il suffit donc de placer un traducteur LCI - VTP entre le Correspondant service et les services Cyclades pour qu'ils puissent se comprendre. La structure logique du CCR est ainsi respectée: les interlocuteurs sont le Correspondant service et le service. Ils n'emploient pas le même langage mais des langages de même niveau décrivant des concepts équivalents. Ils sont donc capables de dialoguer à l'aide du traducteur, exactement comme dans une conférence internationale un délégué Français s'adresse en français à un délégué Britannique bien que ce dernier ne comprenne que l'anglais, mais il y a un interprète entre les deux. Les délégués se parlent l'un à l'autre et non pas à l'interprète qui n'est là que pour faire la traduction. Le traducteur LCI - VTP est d'autant plus facile à écrire que le LCI est semblable au VTP.

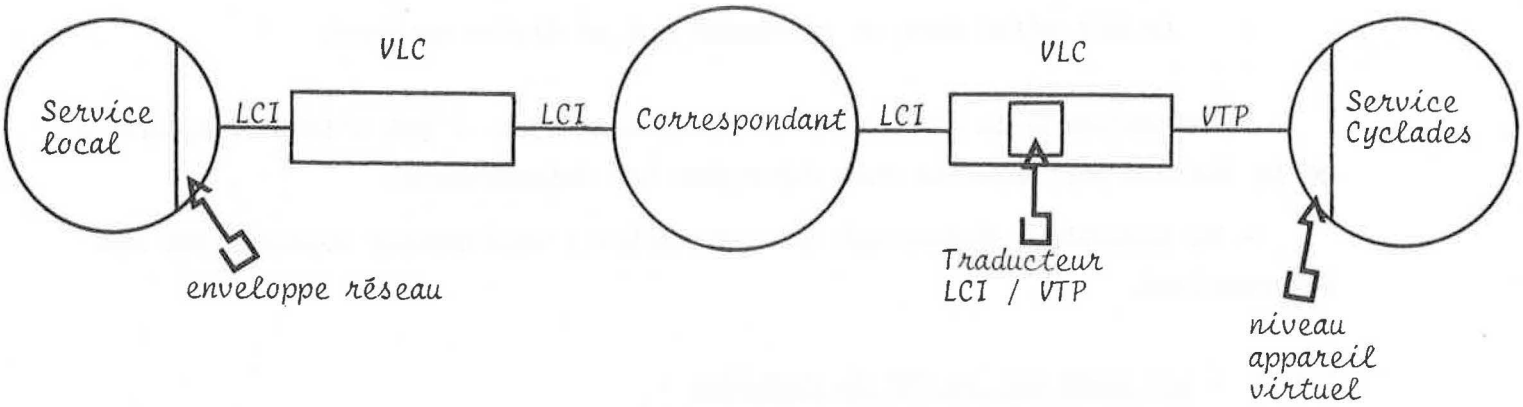


Figure 26

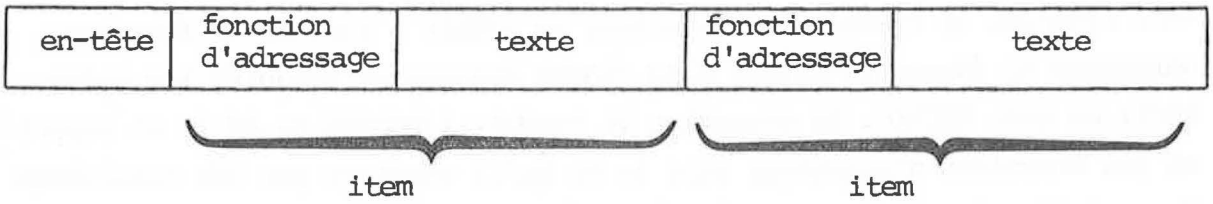
Le VTP est un langage de protocole qui permet de définir un modèle d'appareil virtuel auquel tout terminal doit se conformer. Il comprend donc un contrôle de dialogue convenant parfaitement au LCI. Les terminaux virtuels échangent des messages formés d'un en-tête et éventuellement de texte. Ce texte a une structure de Livre défini par:

. Livre : suite de pages.

. Page : suite de lignes.

. Ligne : suite de caractères.

Le VTP devant décrire entièrement un terminal offre des fonctions d'adressage du type: caractère courant, caractère différent, nouvelle ligne, nouvelle page, etc ... Les caractères du texte des messages, transparent au VTP, sont codés avec les 95 codes EBCDIC de l'ensemble ISO/ASCII dans le cas du VTP minimum.



Message VTP

Le VTP minimum étant trop restrictif deux modes supplémentaires ont été définis:

- Le mode transparent caractère: on peut se servir des 255 valeurs de l'octet comme code caractère. Ce qui permet l'utilisation des 128 codes ASCII ou des 255 codes EBCDIC.

- Le mode transparent appareil: la structure du texte ainsi que les fonctions d'adressage ont disparu.

Le VTP ne fait pas de relation entre fin de message et fin de ligne, c'est à dire que plusieurs lignes peuvent être mises dans un message et un message n'est pas forcément composé d'un nombre entier de lignes [Quint, RES-9].

Nous avons voulu simplifier le LCI en lui donnant l'aspect d'un cas particulier du VTP.

Les terminaux de notre CCR ainsi que la plupard des ordinateurs locaux utilisent l'ASCII, nous avons donc adopté le codage du texte en ASCII.

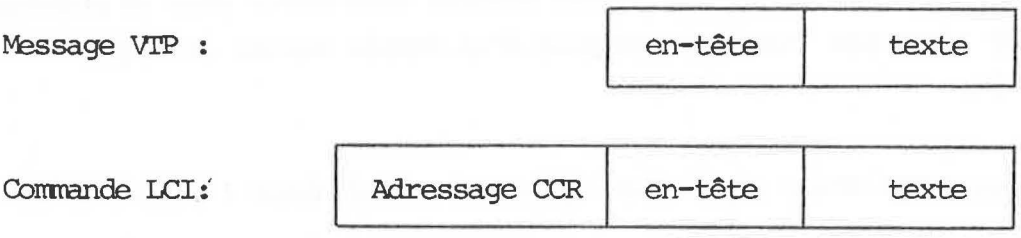
Cela nous a conduit à abandonner la structuration du texte et les fonctions d'adressage du VTP, laissant les caractères de contrôle de l'ASCII remplir ces dernières. Ainsi le LCI correspond au mode transparent appareil du VTP.

Pour l'extérieur il a fallu tenir compte du fait que la plupart des concentrateurs de terminaux et serveurs de Cyclades n'acceptent pas pour l'instant le mode transparent appareil. Le CCR se présente donc, vis à vis du réseau, en mode transparent caractère, en prenant comme convention de n'envoyer qu'une ligne par message. Ce qui fait que dans le sens émission vers Cyclades le travail du traducteur se réduit à ajouter la fonction adressage NL (Nouvelle Ligne) dans chaque message et traduire les codes ASCII en code EBCDIC. En réception il traduit l'EBCDIC en ASCII et supprime les fonctions d'adressage sauf le NL qu'il remplace par les caractères de contrôle: "carriage return" - "line fead" .

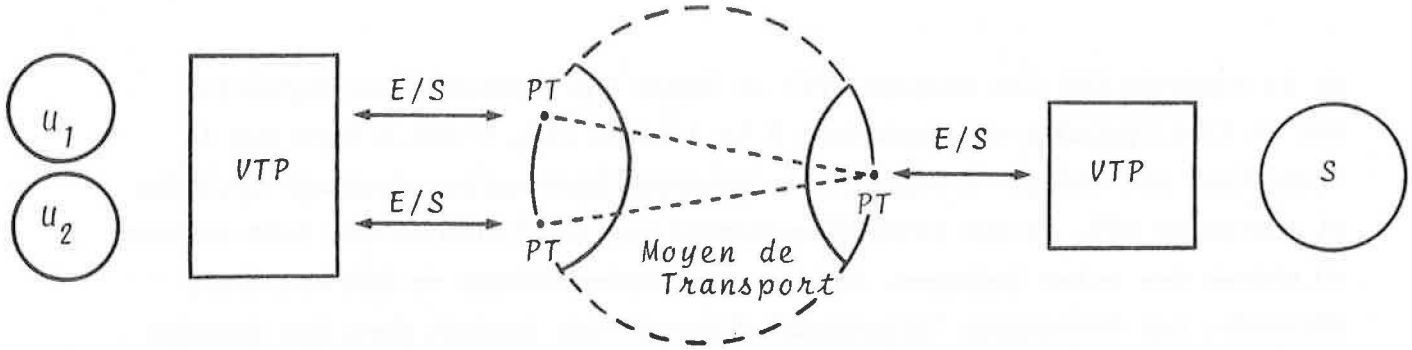
5.1.3. - Différences entre le LCI et le VTP :

Les difficultés commencent à apparaitre avec l'adressage. L'adressage n'existe pas au niveau VTP dans Cyclades, il est déterminé par le moyen de transport, le protocole ST - ST [Zimmermann, RES-10] situé sous le niveau appareil virtuel, qui offre aux hôtes une série de points d'entrée/sortie: les portes (PT) et la possibilité de relier deux portes distinctes par un flot.

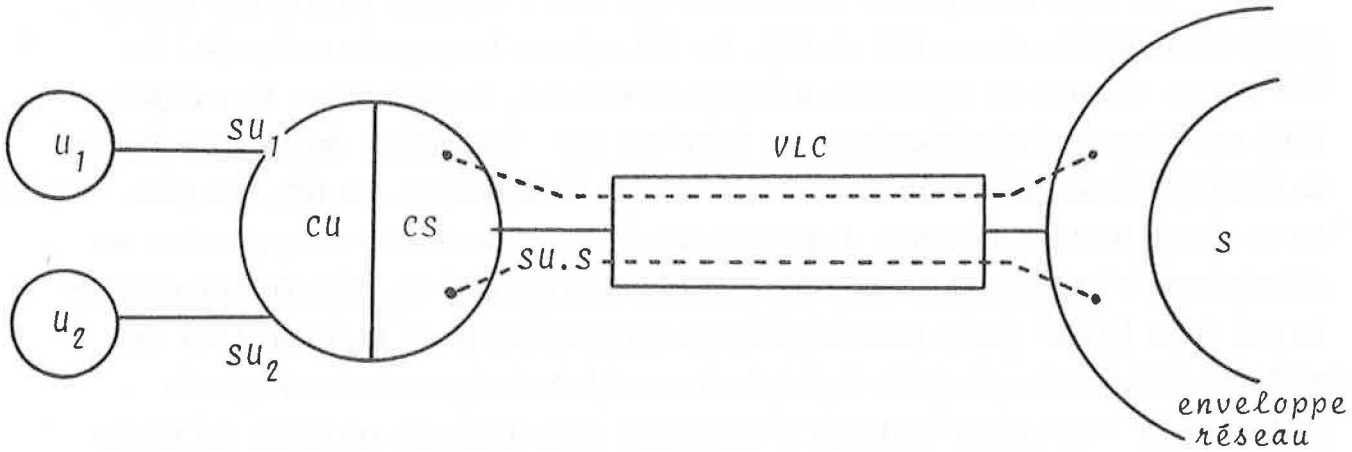
Pour le LCI il faut rajouter l'adressage CCR aux messages VTP.



De plus le protocole ST - ST permet deux types de structure d'échange: les télégrammes (informations courtes: 16 bits) et les lettres



Cyclades: *Les deux flots sont connus au niveau Moyen de transport*



CCR: *Les deux liaisons sont connues au niveau du dialogue CS-Service et inconnues de la VLC*

Figure 27

(de taille variable). Dans le CCR les voies logiques n'acceptent qu'une seule structure d'échange: des plis de longueur variable analogues aux lettres. Certains évènements asynchrones, les "attentions", sont codés sous forme de télégramme en VTP, à l'intérieur du CCR leur codage doit être différent.

La traduction devient donc complexe.

Le traducteur doit transformer l'adressage CCR en adressage Cyclades (et inversement) en supprimant (respectivement en ajoutant) l'adressage CCR

de la commande LCI (au message VTP) et faire une demande d'entrée/sortie sur le flot Cyclades correspondant à la liaison CCR. C'est à dire que le traducteur est obligé d'établir la correspondance entre adressage Cyclades et adressage CCR, ce qui signifie notamment que les liaisons CCR sont connues au niveau des voies logiques. La structure hiérarchique du CCR est alors détruite. Les évènements "attention" doivent être traduit dans des formats différents.

Ce dernier point est sans doute un détail, mais il montre que la structure du LCI et celle du VTP s'éloignent de plus en plus.

En fait nous devons considérer que dès l'origine il y a une différence fondamentale entre LCI et VTP. Le LCI est un langage de commande. Le VTP est un langage de protocole d'appareil virtuel. Le VTP offre la possibilité aux hôtes qui dialoguent de se voir les uns les autres de la même façon, en gros il permet de formater et de contrôler le dialogue. Le LCI, en plus de ce rôle, fournit le moyen d'envoyer des ordres: connexion à un service ou attachement d'un appareil, donc de désigner les entités du CCR. Ce qui n'existe pas dans le VTP (cela pose d'ailleurs un problème pour la définition de poste de travail à appareils multiples) car l'adressage est du niveau du protocole ST - ST. Dans Cyclades l'ouverture d'un flot est un ordre au moyen de transport, c'est une notion connue à ce niveau. Dans le CCR la notion de liaison est inconnue des voies logiques de communication.

Le LCI et le VTP ne sont pas des langages de même niveau, contrairement à ce que nous avions pensé. Nous avons donc construit le LCI en partant des fonctions à remplir et non plus en se basant sur le VTP. Du VTP nous avons concervé une partie du formatage et certains codes opération, ainsi que le codage du "à vous". Mais nous avons été obligés de définir des commandes et des paramètres n'existant pas dans le VTP.

Pour résoudre le problème posé par l'accès à Cyclades, il ne peut plus être question de simple traduction entre le LCI et le VTP, il nous faut considérer l'ex-traducteur comme l'enveloppe réseau des services offerts par Cyclades (cf. Chapitre 9). Il ne traduit plus les commandes LCI mais les in-

terprète en déclenchant les actions nécessaires: émission de messages VTP (dont la génération est simplifiée par le fait que le LCI conserve en partie le formatage du VTP) ou ordres divers au moyen de transport.

5.2. - DESCRIPTION DU LCI

5.2.1. - Syntaxe :

L'ouverture et la fermeture de la voie logique de communication reliant le Correspondant service à un service ne dépend ni de l'un ni de l'autre mais du Superviseur. Cette voie logique étant ouverte le dialogue entre le Correspondant et le service va être constitué par l'échange d'une suite de commandes LCI.

<Commande LCI> ::= <nom commande> <adressage> [(<à vous>,<niveau>)]
[(<param>{,<param>}*)]
[(<information>)]

<nom commande> ::= CNEX | DCNX | SUSP | RACT | ATCH | DTCH | ATTN |
PRTY | EXCH

<adressage> ::= (<origine>,<destination>,<Id.ut>)

C'est l'adressage dont la nécessité a été montrée dans le paragraphe 5.1.1. <origine> et <destination> forment l'identification de la liaison sur laquelle passe la commande. <Id.ut> permet de savoir quel est l'utilisateur concerné par cette commande.

<origine> ::= numéro de SU origine

<destination> ::= numéro de SU destination

<Id.ut> ::= identificateur de l'utilisateur concerné (celui se trouvant dans son environnement)

<à vous> ::= 0 je ne passe pas la main
1 je passe la main mais il faut me la rendre dès le prochain échange
7 je passe la main sans condition

<niveau> ::= 1 fin de pli d'échange
             4 fin de fichier

Cette notion de niveau est directement inspirée du VTP. Elle permet de qualifier l'échange fait et sert pour le mécanisme de "à vous". Le <à vous> peut prendre toutes les valeurs comprises entre 1 et 6 et doit être renvoyé si le niveau atteint ou dépasse cette valeur. C'est pour rester cohérent avec le VTP que nous avons conservé cette notion.

<param> ::= <APP> {n° SU}* | suite de caractères

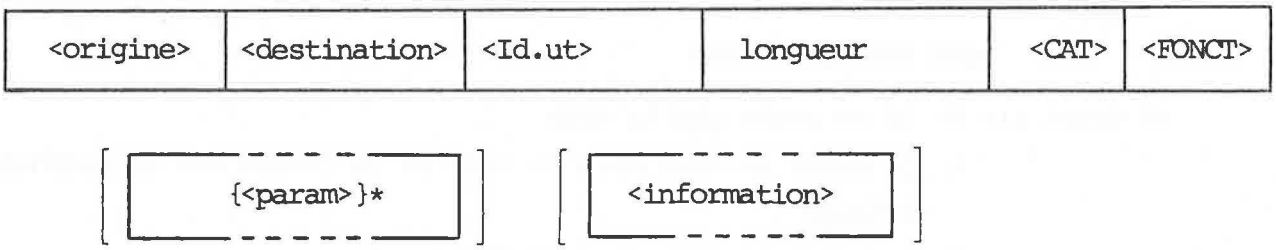
Ce sont les paramètres que l'on peut passer dans une commande. <APP> sert lors d'une connexion, il indique au service les appareils (repérés par un bit) que l'utilisateur s'est attaché. Les numéros de SU suivants donnent l'adressage des services gérant ces appareils. Un paramètre peut aussi être un nom de fichier, un numéro d'attention, etc ...

<information> ::= suite d'octets transparente au LCI

C'est le texte que l'utilisateur ou le service demande au Correspondant de passer de l'un à l'autre.

5.2.2. - Format :

Cette syntaxe est purementdescriptive. En effet, contrairement au LCE, le LCI n'est pas vu de l'extérieur, donc des utilisateurs, et n'a pas besoin d'adopter une forme lisible. Son format, inspiré de celui du VTP, doit concentrer le maximum d'information dans le minimum de place et se présente sous forme d'une suite de codes.



<origine> : 1 octet
<destination> : 1 octet } <adressage>
<Id.ut> : 2 octets

longueur : 2 octets , longueur totale de la commande

<CAT> : 1 octet
<FONCT> : 1 octet } déterminent le code opération de la commande

le <à vous> et le <niveau> sont codés dans <FONCT>.

<param> : <APP> tient sur un octet, chaque n° SU aussi
les autres paramètres sont de longueur variable et séparés par des blancs.

<information> : variable

5.2.3. - Détail des commandes :

- Contrôle de connexion :

Commandes portant sur l'établissement et la destruction des liaisons entre le Correspondant service (pour le compte d'un utilisateur) et un service ou entre services.

CAT : '02

. CNEX : FONCT = '00 éventuellement des paramètres

Etablissement d'une liaison entre un utilisateur et un service (donc cette commande vient toujours du Correspondant). La SU origine est celle de l'utilisateur. APP indique les appareils attachés, donc les liaisons ouvertes en même temps entre le service récepteur de la commande et les services gérant les appareils.

APP : | | | | | | | | les bits à 1 indiquent que l'appareil est attaché

→ FI : fichier (perforateur)
→ LP : imprimante
→ LC : lecteur de cartes

Les n° SU désignent, dans l'ordre des bits positionnés dans APP, les services gérant les appareils.
Il peut y avoir d'autres paramètres.

. DCNX : FONCT = '01 pas de paramètre

Romptune liaison. Cette commande peut venir du Correspondant service ou du service.

. SUSP : FONCT = '02 pas de paramètre

Suspend les échanges sur une liaison, mais celle-ci continue d'exister.

. RACT : FONCT = '03 pas de paramètre

Réactive une liaison suspendue.

. ATCH : FONCT = '04 un paramètre = nom de fichier

Attachement d'un appareil. Cette commande vient du Correspondant service et est destinée au service gérant l'appareil. La SU origine indiquée n'est pas celle de l'utilisateur mais celle du service qu'il a appelé. Une liaison est établie entre le service appelé et le gérant de l'appareil.

. DTCH : FONCT = '05 pas de paramètre

Détachement d'un appareil.

- Contrôle du dialogue :

CAT : '01

. ATTN : Attention FONCT = '0A paramètre = numéro d'attention (1 ou 2) sur un octet

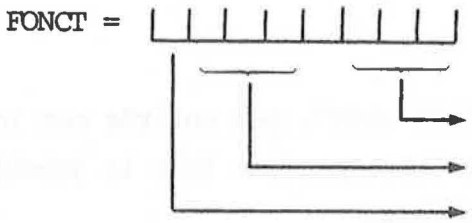
. PRTY : Priority (demande de renvoi de la main)
FONCT = '0B pas de paramètre

Le "à vous" fait parti du contrôle de dialogue mais il est passé dans le FONCT des commandes d'échange.

- Commande d'échange :

CAT : '00

. EXCH : passage d'information (soit à l'utilisateur, soit au service) transparente au Correspondant



5.3. - UTILISATION DU LCI

5.3.1. - Les liaisons :

On s'aperçoit que la plupart des commandes ne sont utilisables que par le Correspondant service, les services ne pouvant envoyer que des DCNX, des EXCH et parfois des DTCH. Ils ne prennent jamais l'initiative d'une connexion avec le Correspondant pour un utilisateur ou d'attacher un appareil. Ils n'envoient pas non plus d'évènement.

Le protocole client-serveur est simple. Le Correspondant service envoie un CNEX puis considère que la liaison est ouverte et la main chez le service. Si le service ne veut pas ouvrir la liaison il répond par DCNX, sinon il peut immédiatement envoyer des EXCH. Si le service ne répond pas l'initiative est laissée à l'utilisateur.

Il n'y a pas de protocole pour l'attachement d'un appareil. Le Correspondant service dit au gérant que l'appareil est attaché et signale le fait au service appelé. Si le service n'accepte pas ce genre d'appareil il peut soit refuser la connexion, soit détacher l'appareil. Pour le gérant de l'appareil si le fichier dont le nom a été passé comme paramètre dans ATCH existe il l'ouvre et s'en sert comme entrée ou comme sortie suivant le type de l'appareil et le ferme au détachement. Si le fichier n'existe pas il en crée un temporaire de ce nom et, suivant son type, le remplit ou le vide à partir ou sur l'appareil physique concerné. Vis à vis du service appelé le

fichier est utilisé comme un fichier permanent. Au détachement il est détruit.

5.3.2. - Le mécanisme du "à vous" :

Il faut bien voir ce que signifie le "à vous". Les entités ont la main les unes par rapport aux autres. Comme le Correspondant fait la jonction entre les entités c'est à lui qu'on signale le fait.

Quand sur une liaison, ouverte à la suite de l'appel d'un service par un utilisateur, le service envoie le "à vous" cela veut dire: "Vous pouvez me transmettre des informations venant de l'utilisateur". Il ne passe pas la main à l'usager au sens où on l'entend habituellement puisqu'il n'est pas en contact direct avec lui. Réciproquement le Correspondant service en envoyant le "à vous" au service lui dit: "Vous pouvez envoyer des informations pour l'utilisateur". Il en va de même lorsque la liaison est ouverte entre deux services.

C'est le Correspondant service qui établit une liaison. Il le fait pour permettre des échanges de données entre deux entités, pas pour son propre compte. Le mécanisme du "à vous" permet de savoir laquelle des deux entités a le droit d'émettre, c'est à dire de demander au Correspondant de transmettre, sur la liaison, des informations à l'autre entité.

CHAPITRE 6

INTERPRETATION DES LANGAGES

6.1. - PLACE DES INTERPRETES

L'utilisateur interprète le LCE, c'est sa capacité intellectuelle qui lui permet de le faire plus ou moins bien.

Le Correspondant interprète le LCE et le LCI. On ne peut pas dire qu'il possède un interprète de chaque langage auquel il fait appel pour analyser les commandes qu'il reçoit. Il est l'interprète lui-même puisque son rôle est de dialoguer avec les autres entités du CCR et de comprendre leurs requêtes, bien que pour le faire il utilise des procédures d'interprétation mais ceci se place au niveau de la réalisation.

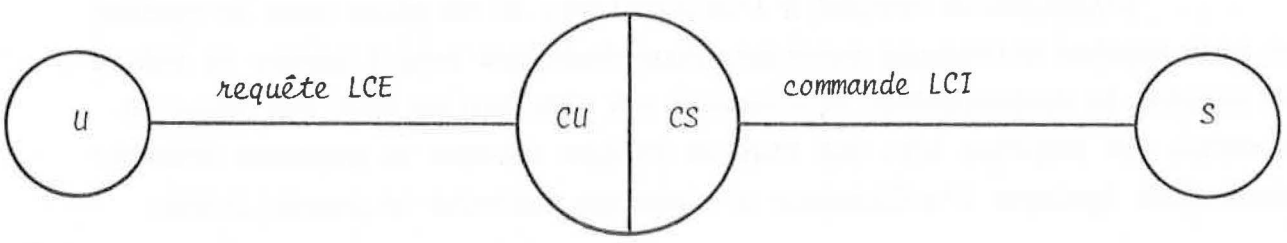
Avec son enveloppe réseau un service interprète le LCI.

Le problème est de savoir à quel niveau l'interprétation complète d'une requête est faite. Il y a deux sortes de requêtes:

- Les requêtes entièrement exécutables par le Correspondant utilisateur ou le Correspondant service.
- Les requêtes qui déclenchent des actions chez l'autre partie du Correspondant.

Pour les premières la question est résolue. Soit le Correspondant utilisateur, soit le Correspondant service les interprète complètement.

Pour les autres le problème se pose dans le sens utilisateur vers service. En effet l'initiative ne vient jamais du service et il ne connait l'usager que lorsque ce dernier l'a appelé.



Analyse de la requête
Génération d'une commande LCI

Exécution de la commande

Quand une requête LCE concernant un service parvient au Correspondant utilisateur celui-ci analyse sa syntaxe. Mais ensuite va-t'il :

. entièrement contrôler la sémantique de la requête; à cette fin il doit posséder toutes les informations utiles relatives au service pour savoir ce qui est correct pour lui et ce qui ne l'est pas. Ainsi le service recevant la (ou les) commande(s) LCI concéquence de cette requête LCE n'a qu'à l'(les)exécuter sans s'occuper de la sémantique. Il déclenche des actions en étant sûr qu'il n'y aura pas d'erreur d'exécution.

. ou laisser passer des informations sémantiques non contrôlées, suivant le principe que nul ne connait mieux le service que le service lui même et que c'est son rôle de s'assurer de la bonne signification des commandes qu'il reçoit.

Dans un cas les enveloppes réseau sont réduites mais le Correspondant est obligé de bien connaitre les services. Dans l'autre le Correspondant est moins important, mais ce sont les enveloppes réseau qui croissent, ce qui est ennuyeux car, alors que le Correspondant est unique, ces dernières sont multiples.

Nous avons choisi la seconde solution car elle plus souple pour l'adjonction d'une nouvelle application. L'introduction d'un service se traduit en général par l'écriture de son enveloppe réseau, qu'elle soit importante ou non c'est un travail qu'il faut faire de toute façon. Avec notre solution les modifications du Correspondant sont minimes, tandis que dans l'autre il faudrait lui ajouter toutes les données relatives au nouveau service.

6.2. - DONNEES DU CORRESPONDANT

L'initiative revient à l'utilisateur, il se place sous le système et peut appeler différents services. Pour dialoguer avec l'usager et suivre sa session le Correspondant utilisateur lui attribue un état qui varie en fonction des requêtes LCE. Cet état va évoluer suivant un automate déterministe fini (puisque l'utilisateur n'a pas une infinité de possibilités).

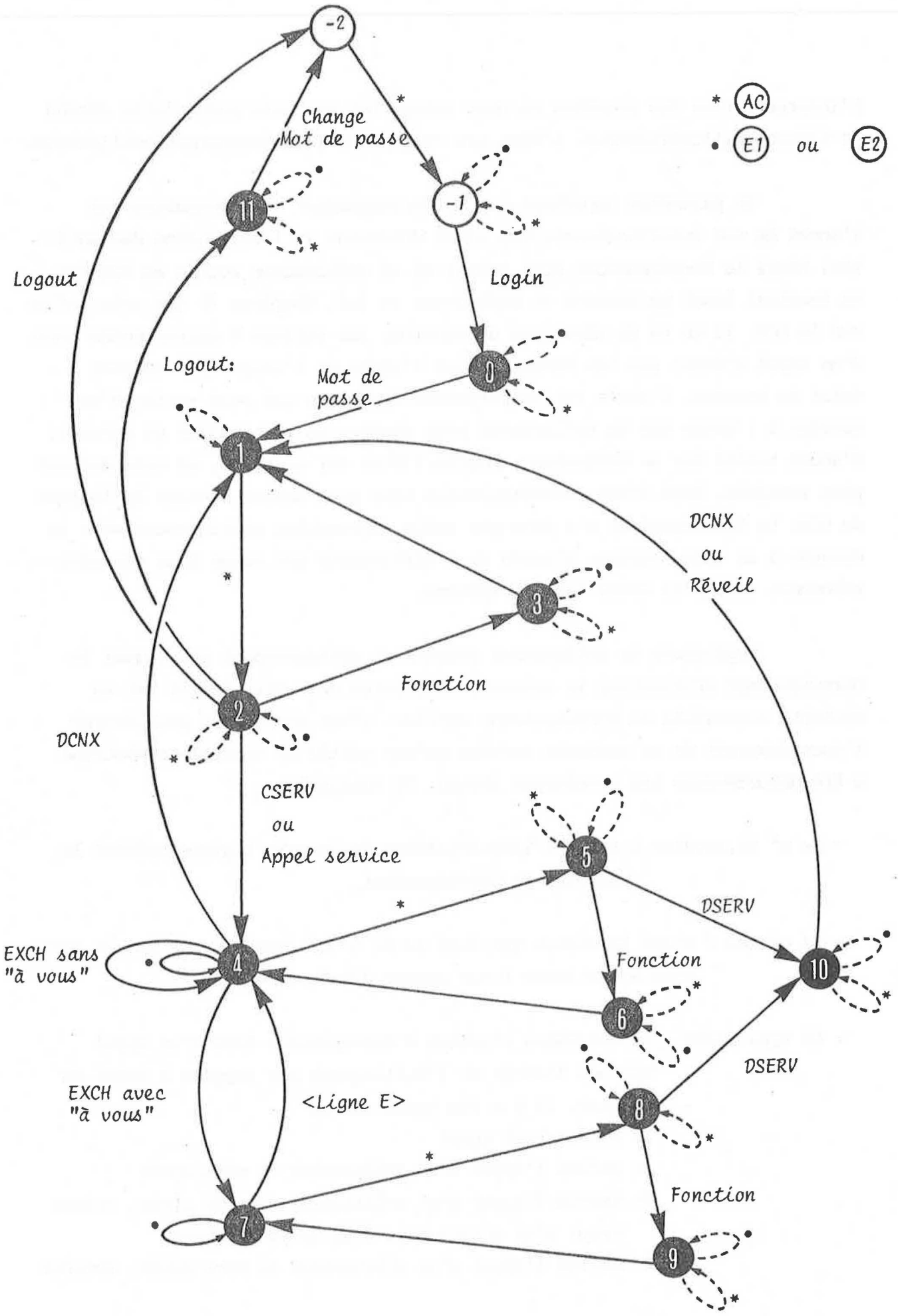

-2
-1
11
0
1
2
3
4
5
6
7
8
9
10
* AC
• E1 ou E2
Change
Mot de passe
Logout
Login
Logout:
Mot de
passe
DCNX
ou
Réveil
Fonction
DCNX
CSERV
ou
Appel service
DSERV
EXCH sans
"à vous"
Fonction
DSERV
EXCH avec
"à vous"
<Ligne E>
Fonction

Figure 29

L'interprétation des requêtes va donc suivre cet automate puisqu'elle dépend de l'état de l'utilisateur. L'état est rangé dans l'environnement utilisateur.

Un paramètre important que le Correspondant doit connaitre est l'accès de ses interlocuteurs. Cet accès détermine la localisation des entités. Ainsi le Correspondant peut savoir si un utilisateur accède au CCR par un terminal local ou distant et nous avons vu (cf. Chapitre 4) que cela influe sur le LCE. Il en va de même pour un service, les actions à entreprendre lors d'un appel n'étant pas les mêmes suivant l'accès de l'usager par rapport à celui du service. L'accès est un paramètre dynamique qui peut varier d'une session à l'autre car un utilisateur peut changer de terminal et un service, d'abord traité sur un ordinateur, pourra l'être sur un autre, si cela devient plus rentable, lors d'une reconfiguration sans pour autant changer la logique du CCR. Le Correspondant n'a donc pas cette information statiquement mais la demande à la voie logique. L'accès de l'utilisateur est rangé dans son environnement lorsqu'il entre sous le système.

Tout comme un utilisateur possède un environnement par lequel le Correspondant utilisateur le connait, un service doit être décrit par un contexte accessible au Correspondant service. C'est d'ailleurs pour éviter l'accroissement de ce contexte service qu'une partie du contrôle sémantique a été déportéedans les enveloppes réseau. Il contient:

- Le n° SU.service : c'est l'identificateur de la voie logique reliant le service au Correspondant.

- La classe : c'est la classe que doit avoir l'utilisateur pour que le service appartienne à son espace d'exécution.

- Le type accès : il détermine l'action à entreprendre lors d'un appel suivant l'accès de l'utilisateur par rapport à celui du service. Il y a six types accès:
  1- refuse tout appel
  2- refuse l'appel d'un utilisateur de même accès
  3- SWITCH l'appel d'un utilisateur de même accès, refuse celui d'un utilisateur d'un accès différent
  4- SWITCH l'appel d'un utilisateur de même accès, accepte

celui d'un utilisateur d'un accès différent
5- refuse l'appel d'un utilisateur d'un accès différent
6- accepte tout appel

Un SWITCH tient compte du fait que pour les échanges de données il est inutile de passer physiquement par le Frontal si le terminal et le service sont sur le même site. Une telle opération n'est pas encore réalisée car elle implique un mécanisme complexe sur le site en question pour savoir qu'une <Ligne AC> doit être transmise au Correspondant utilisateur alors qu'une <Ligne E> est remise directement au service et que les EXCH venant du service pour l'utilisateur n'ont pas besoin de transiter par le Correspondant service. Néanmoins sa réalisation est prévue et il faut en tenir compte dès maintenant.

- La catégorie : elle définit les évènements que le service accepte. Il y a deux catégories:
  1- accepte PRTY
  2- accepte ATTN

- Les appareils supplémentaires autorisés lors d'un appel.

- La longueur des lignes en entrée et en sortie.

Chaque fonction LCE a aussi un bloc de données propre, mais ceux-ci n'ont pas une forme standard, les fonctions étant fort différentes les unes des autres. Ils contiennent généralement des informations relatives aux paramètres de la fonction. Toutes les fonctions possèdent quand même deux données semblables: une classe, qui est la même que pour les services, et un ensemble d'accès qui donne tous les accès possibles aux utilisateurs pour que la fonction appartienne à leur LCE.

Nous avons déjà dit que l'environnement de l'utilisateur contient sa classe. Il comprend aussi une pseudo-table de contrôle modifiable (par MODIF) et qui décrit "l'appareil utilisateur": le codage des caractères de contrôle dans le LCE.U. Cette pseudo-table de contrôle est rémanente d'une session à l'autre de l'utilisateur.

6.3. - INTERPRETATION

Nous allons suivre la session d'un utilisateur et voir comment chaque commande est interprétée. L'évolution de l'état suit l'automate décrit figure 29. Pour le LCE nous n'écrirons que ce que le Correspondant utilisateur reçoit, le caractère (FL) étant supprimé par la voie logique.

6.3.1. - Début de session :

Au départ l'utilisateur est à l'état -2, il n'est pas connecté au système. Pour le faire il tape:

(AC) LOGIN numéro utilisateur

Le (AC) fait passer à l'état -1, le Correspondant sait qu'un utilisateur l'appelle sur une voie mais il ignore lequel, il fait une demande de réception sur la voie et attend un LOGIN dont le numéro utilisateur lui permettra de reconnaitre l'usager. Le Correspondant utilisateur vérifie que ce numéro existe dans son catalogue et charge l'environnement de l'utilisateur. Par charger nous entendons qu'il met en place un mécanisme simple pour retrouver l'environnement à partir du n° SU.terminal, c'est en ce sens que l'on dit que l'utilisateur est confondu avec son terminal. Dans cet environnement il range le n° SU.terminal et l'accès qu'il demande à la voie logique. S'il s'agit d'un terminal local il remet à la gestion du terminal la pseudo table de contrôle trouvée dans l'environnement. Ainsi l'utilisateur retrouve bien "son" LCE avec les caractères de contrôle qu'il s'était défini. L'état passe à 0 et le Correspondant utilisateur fait une demande de réception pour lire le mot de passe.

L'utilisateur doit alors taper son mot de passe, si ce dernier est mauvais l'entrée sous le système avorte, sinon l'état passe à 1. L'usager est maintenant reconnu par le CCR et peut commencer à travailler. Il tape:

(AC) <Commande>

(AC) fait passer à l'état 2 et provoque une demande de réception de la part du Correspondant utilisateur pour lire la commande. Plusieurs cas se présentent:

- Demande d'une fonction (différente de CSERV) :

L'état passe à 3 et la fonction est exécutée si la priorité de l'utilisateur le permet. Nous n'entrerons pas dans le détail de toutes les fonctions. Certaines sont des utilitaires faisant appel à des programmes de services, d'autres permettent de visualiser des états ou contrôler le CCR, etc ... A la fin de l'exécution l'état revient à 1.

- Appel d'un service (ou fonction CSERV) :

Une analyse du nom du service permet d'accéder à son contexte. Le Correspondant peut vérifier si ce service appartient à l'espace d'exécution de l'utilisateur et le type accès lui indique l'action à entreprendre. Voyons le cas le plus intéressant, celui de la connexion. Le Correspondant utilisateur range dans l'environnement usager le n° SU.service, mais aussi sa catégorie et ses longueurs de ligne, ceci afin d'éviter d'aller chercher ces informations dans le contexte service au cours des échanges.

6.3.2. - Appel de service sans appareil supplémentaire :

Soit un utilisateur d'identificateur IU sur le terminal SU.U qui appelle le service SERVI d'accès SU.S. L'usager tape la commande:

CSERV SERVI ou plus simplement SERVI

Le Correspondant utilisateur répercute cet appel chez le Correspondant service qui, pour ouvrir une liaison, envoie:

CNEX (SU.U,SU.S,IU)

Si l'utilisateur a mis des paramètres dans sa commande le Correspondant se contente de les transférer de la commande LCE qu'il a reçu à la commande LCI qu'il envoie sans les analyser. C'est en ce sens que dans le paragraphe 6.1 nous parlions d'une partie du contrôle sémantique reportée dans les enveloppes réseau. Dans le cas d'un appel c'est au service de s'assurer que les paramè-

tres sont corrects, ils sont transparents au Correspondant.

Si l'utilisateur tape: SERVI X Y Z

Le Correspondant service envoie: CNEX (SU.U,SU.S,IU)(X,Y,Z)

L'état passe à 4 : on considère que le service est connecté et qu'il a la main, c'est à dire que le Correspondant service fait une demande de réception du service.

Le service envoie des données à l'usager par une commande d'échange:

EXCH (SU.S,SU.U,IU)(n,1)(données)

Les données sont passées au Correspondant utilisateur qui les transmet telles quelles à l'usager.
Si n = 0 le service garde la main. L'état reste à 4 et le Correspondant service se remet en réception du service.
Si n = 1 le service donne la main à l'utilisateur. L'état passe à 7 et le Correspondant service le signale au Correspondant utilisateur qui se met en réception d'une <Ligne E> de l'utilisateur.

A son tour l'utilisateur peut envoyer des données au service en tapant une <Ligne E>. Le Correspondant utilisateur donne cette ligne au Correspondant service qui l'expédie au service:

EXCH (SU.U,SU.S,IU)(7,1)(données = <Ligne E>)

L'état passe à 4 car l'usager ne peut pas garder la main. Et le Correspondant service se met en réception du service.

L'interprétation de (E1) et (E2) va entraîner des actions différentes suivant la catégorie du service:

- catégorie 1 : envoi de PRTY (SU.U,SU.S,IU)
- catégorie 2 : envoi de ATTN (SU.U,SU.S,IU)(1 ou 2)

La fin des échanges entre utilisateur et service se fait à l'initiative de l'un ou l'autre. Pour indiquer la fin de son appel l'usager tape: (AC) DSERV . L'état passe à 10 que l'on vienne de l'état 4 ou de l'état 7. Le Correspondant service envoie: DCNX (SU.U,SU.S,IU) et attend la réponse du service. Pour éviter une attente indéfinie un réveil protège cet état. Quand le service répond (par un DCNX) ou que le réveil arrive à terme le Correspondant considère la liaison rompue. Il efface de l'environnement utilisateur les informations concernant le service et remet l'état à 1.

Pour fermer la liaison (ou refuser de l'ouvrir) le service doit avoir la main et expédie: DCNX (SU.S,SU.U,IU) . Le Correspondant service lui répond par un DCNX et le signale au Correspondant utilisateur qui annonce la fin des échanges à l'utilisateur. Comme précédemment l'état repasse à 1.

Exemple 1 : Appel d'APL, service conversationnel
L'utilisateur tape: APL ou CS APL
Le Correspondant service envoie: CNEX (SU.U,SU.APL,IU)
L'enveloppe réseau d'APL simule l'attachement d'un terminal au système APL qui envoie un message d'en-tête avec passage de la main et masquage (bit secret) pour recevoir le mot de passe particulier de l'utilisateur pour entrer sous le système APL.
Service APL envoie: EXCH (SU.APL,SU.U,IU) ('1,1) ("APL - MINES")
La main est à l'utilisateur et les échanges peuvent commencer.

Exemple 2 : Appel d'un service de compilation Fortran
L'utilisateur tape: FORT X Y c'est à dire compiler le fichier source Fortran X et ranger le code généré dans le fichier Y.
Le Correspondant service envoie: CNEX (SU.U,SU.FORT,IU)(X,Y)
L'enveloppe réseau du compilateur Fortran va créer les cartes de contrôles d'un travail (Job) effectuant la compilation demandée en extrayant des noms fichiers réseau X et Y les noms fichiers locaux XLOC et YLOC. Eventuellement, si ces fichiers ne sont pas présents sur le site du compilateur, elle s'adresse au service gestion de fichiers réseau pour qu'il les transfère sur le site. Elle doit en outre vérifier que X est bien du type source et Y du type objet (dans certains cas Y peut même être créé). Elle passe à l'envi-

ronnement service local les ordres:

```
% JOB ...
% DCB INPUT = XLOC
% DCB OUTPUT = YLOC
% COMPILE FORTRAN
% JS
```

Le service envoie à l'utilisateur, dans un EXCH, le résultat de la compilation (tout s'est bien passé ou non). Des paramètres supplémentaires peuvent indiquer ce que l'utilisateur veut comme listage et où le sortir.

6.3.3. - Appel de service avec appareils supplémentaires :

Les appareils forment l'environnement de l'appel. Ils sont passés sous forme de paramètres par mot clé (<PMC>) et, contrairement aux paramètres positionnels, sont analysés par le Correspondant qui vérifie que le service appelé accepte ces appareils.

Prenons le cas d'un utilisateur appelant un service de traitement par lot (RB) avec une entrée pour des cartes et une sortie imprimante.

L'utilisateur tape: RB (LC = X , LP = Y) X et Y étant des noms de fichiers

Le Correspondant va attacher les appareils "lecteur de cartes" et "imprimante" à l'appel et connecter le service en le prévenant de la présence de ces appareils. On a donc établissement de trois liaisons:

Utilisateur - Service RB
Service LC - Service RB
Service LP - Service RB

Le Correspondant utilisateur marque dans l'environnement utilisateur les appareils attachés.

Le Correspondant service envoie:

```
ATCH (SU.RB,SU.LC,IU) (X)
ATCH (SU.RB,SU.LP,IU) (Y)
CNEX (SU.U,SU.RB,IU) (<APP>,SU.LC,SU.LP)
```

respectivement aux services concernés.

Si X et Y n'existent pas les services LC et LP créent des fichiers temporaires de ce nom. X sera alimenté par les cartes du lecteur (appareil physique) et Y sera envoyé au symbion imprimante.

Sur les trois liaisons le service RB a la main, mais sur la liaison RB - LC il la donne immédiatement au service LC par un EXCH. Sur la liaison utilisateur - RB tout se passe comme dans le cas où il n'y a pas d'appareil, à l'appel l'état est donc passé à 4. Cet état n'est influencé que par les échanges sur la liaison utilisateur - RB, car la main peut changer de côté. Pour les liaisons avec les appareils la main est fixée une fois pour toute, le sens des échanges étant unidirectionnel.

- Liaison utilisateur - RB :

Voir paragraphe 6.3.2.

- Liaison LC - RB :

Le service LC envoie EXCH (SU.LC,SU.RB,IU)(0,n)(carte) que le Correspondant service renvoie aussitôt au service RB. Le Correspondant service se remet en réception du service LC.

- Liaison LP - RB :

Le service RB envoie EXCH (SU.RB,SU.LP,IU)(0,n)(ligne) que le Correspondant service renvoie aussitôt au service LP. Le Correspondant service se remet en réception du service RB.

n = 1 indique fin de carte ou fin de ligne, n = 4 fin de fichier. IU est très important car il permet aux services de distinguer les différentes liaisons qu'ils peuvent avoir entre eux en désignant l'utilisateur concerné par leurs échanges.

La fin des échanges peut être faite à l'initiative de l'utilisateur ou du service RB, mais pas des appareils. Pour ces derniers le Correspondant service envoie: DTCH (SU.RB,SU.LC,IU)

et DTCH (SU.RB,SU.LP,IU)

Les liaisons RB - LC et RB - LP sont ainsi rompues. Les services LC et LP n'ont pas besoin de répondre et ferment (ou détruisent s'ils étaient temporaires) leurs fichiers respectifs.

6.3.4. - Fonction lors d'un appel de service :

A tout instant, au cours d'un appel de service, l'utilisateur peut adresser une fonction au Correspondant utilisateur (nous l'avons vu dans le cas du DSERV). Il tape alors: (AC) <Commande>.

Le (AC) provoque le passage de l'état 4 à l'état 5 ou le passage de l'état 7 à l'état 8, dans ce dernier cas le Correspondant utilisateur doit annuler la réception de <Ligne E> en cours. Le Correspondant utilisateur se met en réception de la commande. L'exécution de cette dernière (si ce n'est pas un DSERV) fait passer dans les états respectifs 6 et 9. La fin de l'exécution fait retourner à l'état d'avant le (AC) (le retour à l'état 7 provoque aussi la remise en réception d'une <Ligne E>).

6.3.5. - Appel de procédure :

Les procédures sont formées d'une liste de requêtes que le Correspondant va lire et exécuter les unes après les autres. Pour cela, lors d'un appel de procédure, il marque dans l'environnement que l'on passe en mode exécution de procédure. L'interprétation continue de suivre l'automate mais au lieu d'aller chercher les requêtes utilisateur sur la SU.terminal le Correspondant utilisateur les reçoit maintenant de la SU.TTY (cf. paragraphe 4.2.2), en tenant compte du fait que les évènements (AC), (E1) et (E2) ne peuvent plus passer par le mécanisme normal.

(E1) et (E2) existent comme instructions de procédure /E1 et /E2.

Une <Ligne AC> est reconnue par son premier caractère qui doit être "?" et une <Ligne E> commence par un blanc.

Les sorties continuent à se faire normalement sur la SU.terminal.

Le Correspondant utilisateur établit une correspondance entre paramètres réels et paramètres formels par leur position. Les paramètres réels sont substitués aux paramètres formels dans les commandes où ces derniers apparaissent.

Par exemple un utilisateur s'est défini la procédure:

```
PROC
SERV1   SERV2
?CS  SERV1
  :
  :
?DS
?CS  SERV2
  :
  :
?DS
```

et l'appelle au cours d'une session: PROC APL CP67

Le Correspondant utilisateur établit la correspondance par une table:

SERV1	APL
SERV2	CP67

CS SERV1 après substitution du paramètre est interprété comme CS APL, et CS SERV2 comme CS CP67.

Quand toutes les commandes de la procédure ont été exécutées on repasse dans le mode courant d'exécution.

6.3.6. - Fin de session :

Lorsque l'utilisateur a fini son travail il peut sortir du système et terminer sa session. Il tape alors:

(AC) LOGOUT

Le (AC) fait passer de l'état 1 à 2 et le LOGOUT à l'état -2, avec rupture de la correspondance n° SU.terminal - environnement, on dit que l'on décharge l'environnement. Si le terminal est local on lui redonne comme table de contrôle la table standard (cf. paragraphe 4.2.1).

L'usager peut changer son mot de passe en tapant ":" après sa commande de sortie:

(AC) LOGOUT:

L'état passe alors de 2 à 11 et le Correspondant utilisateur se met en réception du nouveau mot de passe qu'il range dans l'environnement dès que l'utilisateur l'a envoyé. Puis l'état passe à -2.

Remarque : Pour simplifier le travail de l'utilisateur on l'autorise à appeler un autre service ou à terminer sa session alors qu'il n'a pas fini ses échanges avec un service (état 4 ou 7). Dans ce cas (AC) CSERV SERV1 est en fait interprété comme si l'utilisateur avait tapé:

(AC) DS
(AC) CSERV SERV1

et (AC) LOGOUT comme:

(AC) DS
(AC) LOGOUT

6.4. - FORME DES INTERPRETES

Il n'est pas question de détailler la réalisation des interprètes, mais leur forme logique.

6.4.1. - Interprète LCI :

Il est très simple puisqu'il ne peut avoir que trois entrées:

- Soit l'entrée ne correspond pas à l'état de la liaison (ou la liaison n'existe pas), il laisse la liaison dans le même état.
- Soit c'est un DCNX.
- Soit c'est un EXCH.

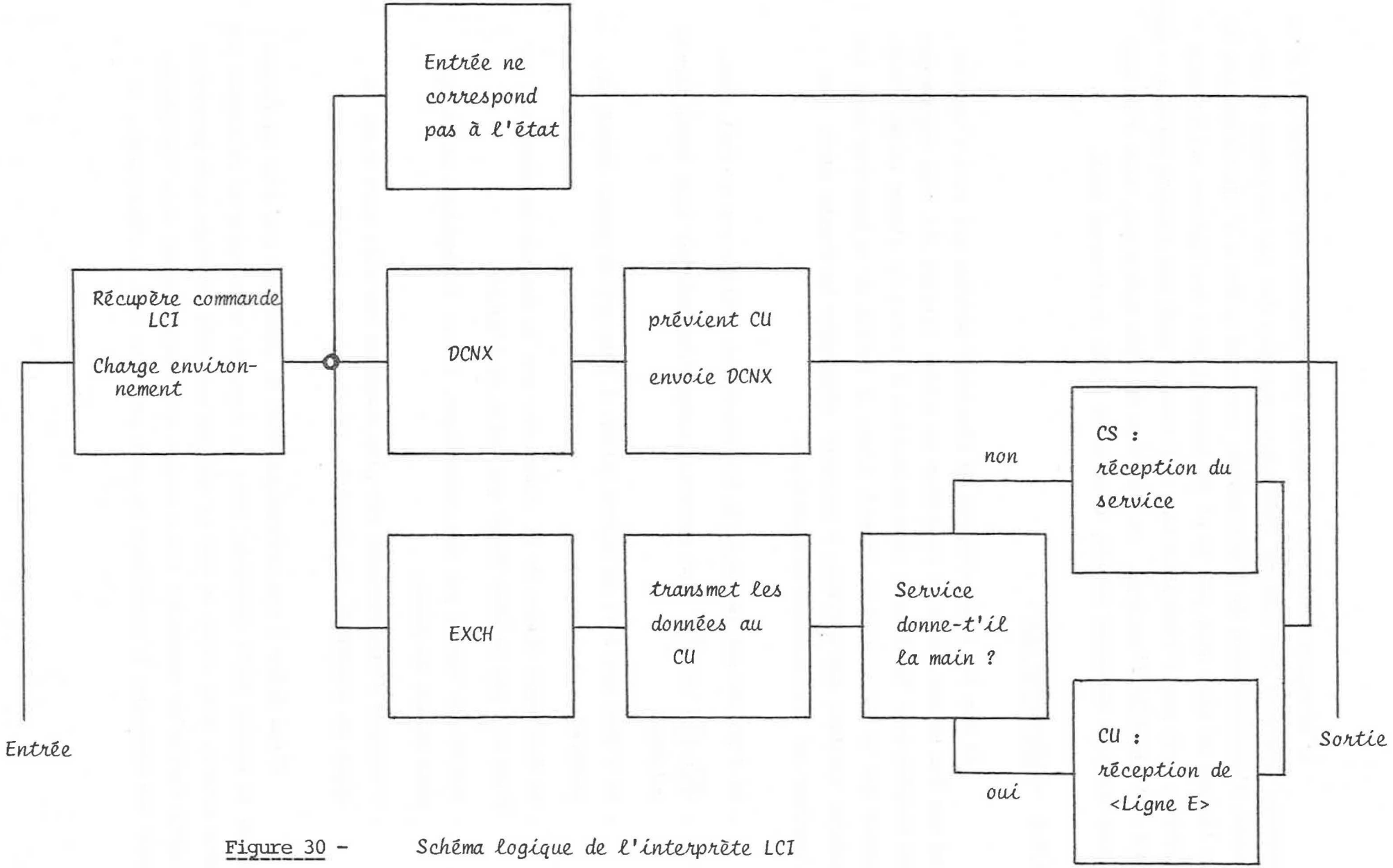


Figure 30 - Schéma logique de l'interprète LCI

L'interprète comprend un module qui récupère une commande LCI d'un service (récupère les fins de réceptionsdemandées sur les services) et retrouve l'environnement de l'utilisateur concerné grâce à l'identification de la liaison en LCI: soit par le n° SU.terminal pour une liaison utilisateur - service, soit par l'identificateur utilisateur pour une liaison service - appareil. Il aiguille l'analyse, en fonction du code opération, vers l'un des trois modules: mauvaise entrée, traitement DCNX, traitement EXCH.

6.4.2. - Interprète LCE :

Il est lui aussi composé de plusieurs modules qui vont s'appeler les uns les autres. Si nous regardons le schéma (figure 31) nous voyons que les entrées sont facilement reconnaissables à partir de l'état et de l'évènement que le Correspondant attend, ainsi il suffit de se brancher dans les modules traitant ces entrées. A partpour <Commande> le chemin suivi par l'analyse est parfaitement déterministe.

- Si l'entrée est mauvaise le Correspondant utilisateur ne fait rien.
- (AC), (E1) et (E2) sont reconnus comme évènement par leur forme particulière.
- Si l'état est à -1 on attend LOGIN, à 0 le mot de passe (début de session).
- On distingue <Ligne E> et <Commande> car la demande de réception de l'une n'a pas la même forme que celle de l'autre.
- LOGOUT est repéré par son mnémonique, l'état 11 implique un changement de mot de passe.
- L'analyse d'une commande est plus complexe car elle peut être: un appel de procédure, un appel de service ou une fonction.

C'est grâce à son mnémonique (nom de commande) que l'on va reconnaitre la nature de la commande. Mais la forme du mnémonique ne détermine pas cette nature, nous avons vu que la syntaxe est très générale pour permettre l'ajout facile de commande: toute chaîne de caractèrespeut être considérée comme une commande. L'aiguillage ne peut pas être fait immédiatement. Le

Correspondant utilisateur va donc exécuter successivement les modules capables de reconnaître la nature de la commande, une fois celle-ci déterminée il se branche dans les modules de traitement. L'ordre de reconnaissance est : procédure, service et fonction. Il est à noter que si l'utilisateur s'est créé une procédure de même nom qu'un service ou une fonction ces derniers ne lui sont plus accessibles jusqu'à ce qu'il détruise cette procédure.

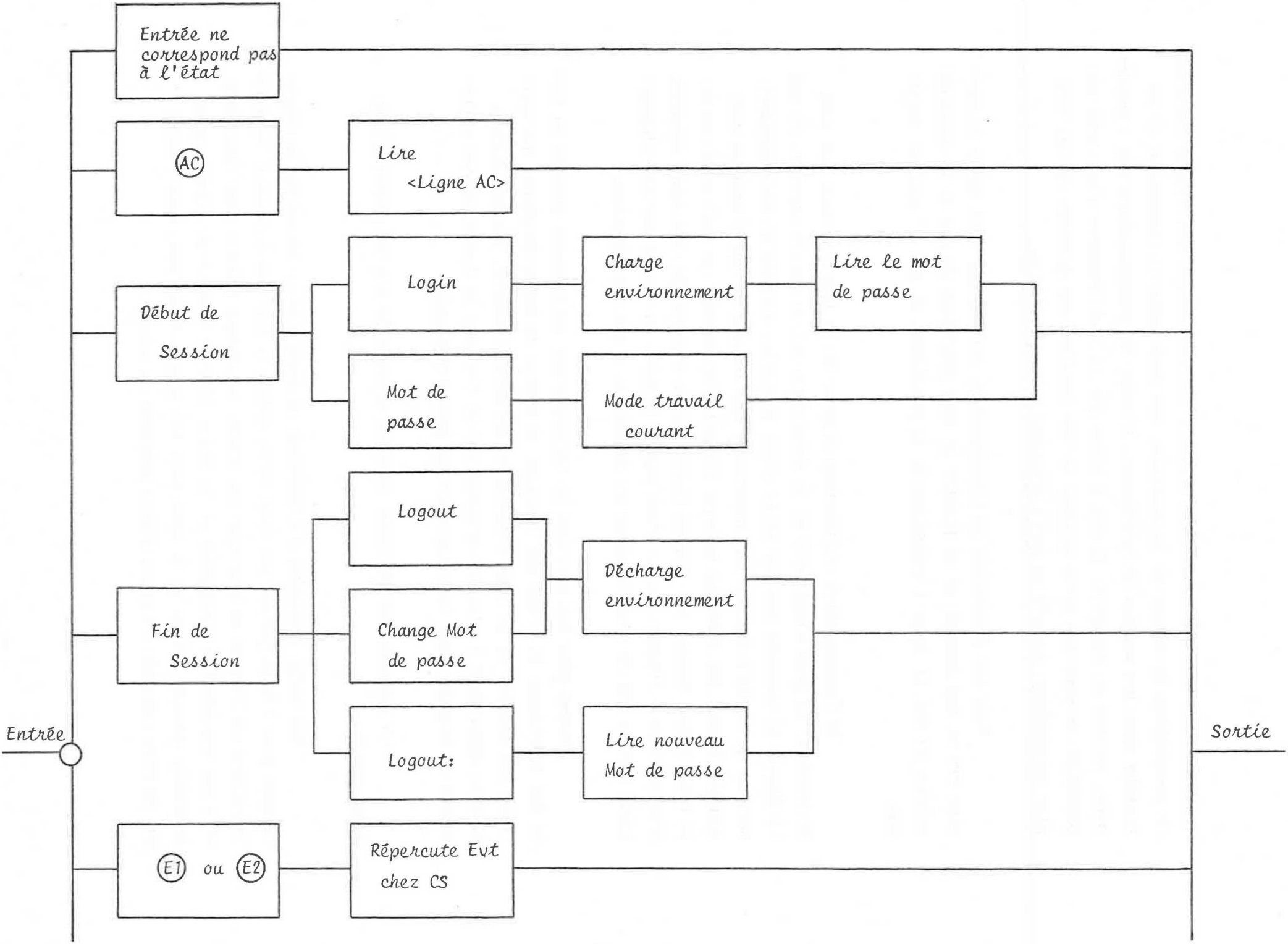
Pour les procédures le Correspondant utilisateur fait appel au service TTY et lui demande si un fichier de type procédure du nom de la commande existe. Si oui il lance l'exécution de la procédure. Si non l' analyse continue.

Le Correspondant utilisateur dispose de la liste de tous les noms de service, il peut ainsi voir si le mnémonique est un nom de service. Si oui il charge le contexte service et,si c'est possible, signale au Correspondant service qu'il doit établir une connexion. Les modules de cette analyse sont communs à tous les services puisque l'appel d'un service se fait toujours de la même façon. Seules les données relatives au service (le contexte) changent d'un service à l'autre. Si ce n'est pas un service ou que le service n'appartient pas au LCE de l'utilisateur on va voir si c'est une fonction.

Comme pour les services le Correspondant utilisateur possède la liste des fonctions. Si c'est une fonction il charge la boite-fonction, qui contient les données et le code d'exécution de cette fonction, correspondante. Si elle appartient à l'espace d'exécution de l'usager le Correspondant utilisateur se branche dans la boite-fonction pour l'exécuter.

La commande est déclarée inconnue quand elle n'a été reconnue par aucun module.

Une telle structure a l'avantage de permettre un mécanisme de recouvrement pour l'interprète: une fois qu'un module a fini son exécution le module suivant où l'on va se brancher est connu. Le module exécuté peut donc être vidé sur une mémoire auxiliaire et la place libérée devient utilisable pour le module suivant. Il en va de même pour les boites-fonctions, seule celle qui va être exécutée a besoin d'être présente en mémoire.


Entrée ne correspond pas à l'état
AC
Lire <Ligne AC>
Début de Session
Login
Charge environnement
Lire le mot de passe
Mot de passe
Mode travail courant
Fin de Session
Logout
Change Mot de passe
Logout:
Décharge environnement
Lire nouveau Mot de passe
Entrée
Sortie
E1 ou E2
Répercute Evt chez CS

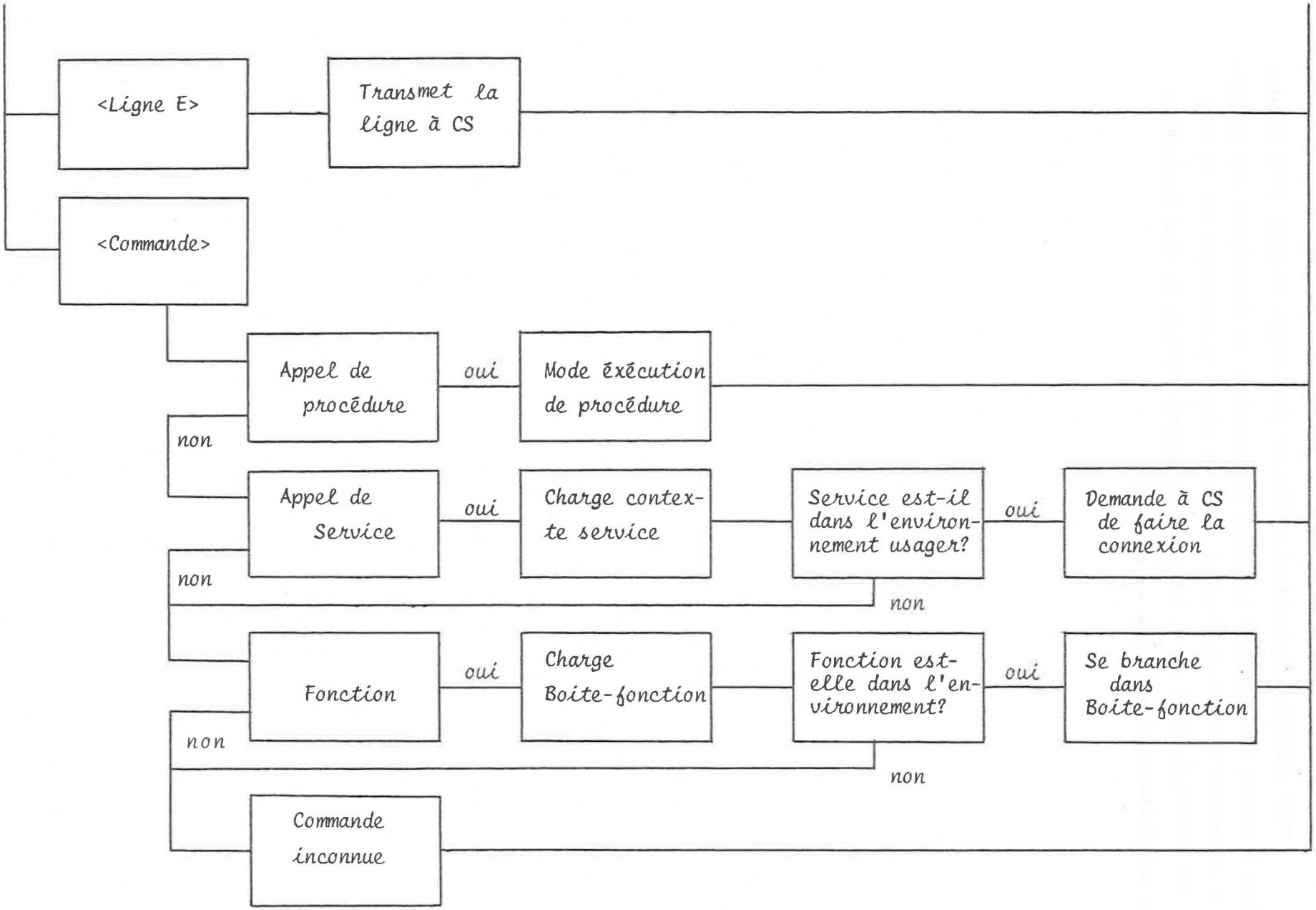


Figure 31

Schéma logique de l'interprète LCE

Mais surtout l'extension du LCE est facilitée. L'ajout d'un service revient à mettre son nom dans la liste des services et introduire son contexte. Le mécanisme d'accès et d'appel reste le même. Le principe est le même pour une fonction, sauf qu'au lieu d'un contexte service il faut introduire toute une boite-fonction (code et données).

TROISIEME PARTIE :

LES PROTOCOLES

CHAPITRE 7

LES VOIES LOGIQUES DE COMMUNICATION

7.1. - DEFINITION

Une voie logique de communication est un ensemble qui permet à deux entités de même niveau de communiquer en garantissant un taux d'erreur minimal et un contrôle du flux des informations qui transitent.

Une voie logique permet donc de relier:

- Le Correspondant service à un service

- Le Correspondant utilisateur à un utilisateur; en fait au bout de la voie logique il y a un terminal qui est l'accès de l'utilisateur, une fois que le terminal est pris par l'utilisateur et que le Correspondant utilisateur l'a reconnu on confond l'usager et son accès.

En général une voie logique de communication se décompose en un support de transmission (ligne, bus, ensemble modem-ligne-modem, etc ...) c'est à dire tout moyen physique de transmission, une gestion des transmissions: gestion physique du support, gestion de la procédure de transmission et une gestion des protocoles de communication.

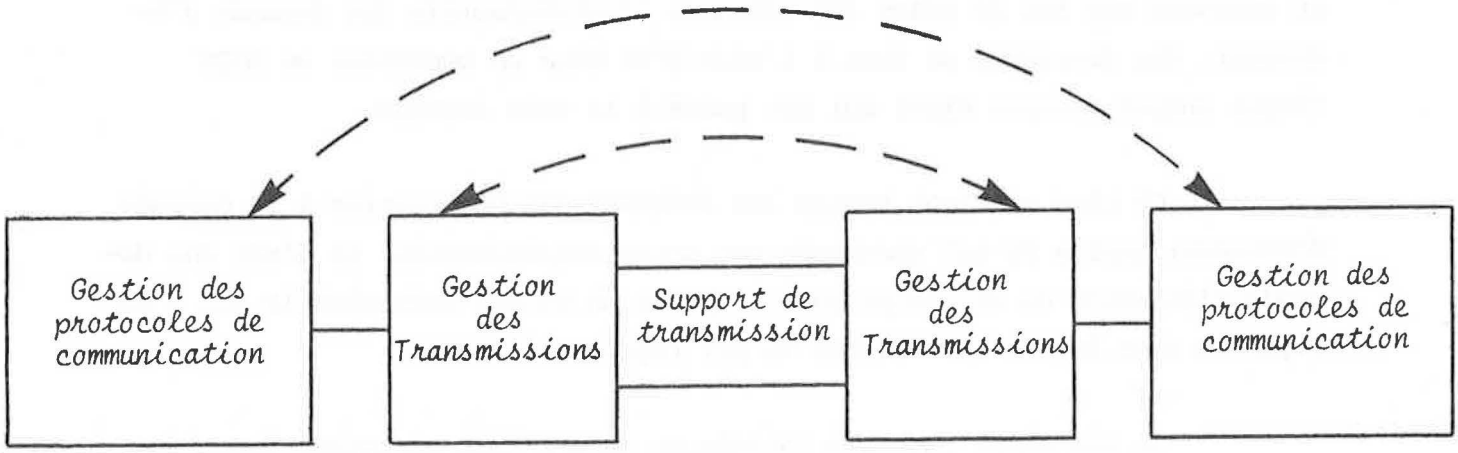


Figure 32

Ce schéma est général car parfois tout ou partie d'un des niveaux peut être réduit à rien.

Nous allons étudier plus en détail la place de ces voies logiques dans le CCR, comment le Correspondant les voit et les utilise ainsi que la forme qu'elles prennent sur le Frontal.

7.2. - INTERFACE ENTRE LE CORRESPONDANT ET LES VOIES LOGIQUES DE COMMUNICATION

Le Correspondant voit les voies logiques de communication sous forme de SU (cf. paragraphe 3.2.2) identifiées chacune par un numéro et sait les utiliser car il connait l'ensemble des actions qu'il peut déclencher sur ces SU pour communiquer avec les terminaux ou les services.

L'interface se constitue d'un ensemble d'ordres que le Correspondant peut donner à une voie logique et d'un ensemble d'évènements que cette dernière peut lui signaler.

7.2.1. - Ordres du Correspondant à une voie logique :

- Demande d'entée / sortie [CCR-4]

Le Correspondant communique avec ses interlocuteurs en émettant et recevant sur les SU grâce aux demandes d'entrée/sortie (ou demande d'échange). Ces dernières se font à l'aide d'un bloc de contrôle: un IOCB (Input Output Control Bloc) qui est passé à la voie logique.

Ce bloc contient toutes les informations nécessaires à la demande d'échange: quelle SU est concernée par cette entrée/sortie, si c'est une demande d'émission ou de réception, l'adresse du buffer contenant le pli à envoyer ou dans lequel sera stocké un pli reçu, etc ...

Il est aussi possible d'indiquer dans l'IOCB certaines fonctions qui permettent de piloter le terminal situé au bout de la voie logique (ces fonctions ne servent évidemment pas dans le cas d'une SU.service):

- Secret de l'entrée: supprime l'écho du terminal, il est utilisé afin que les usagers puissent taper leur mot de passe en toute sécurité.
- Filtrage des blancs.

- Passage à la ligne avant l'entrée/sortie: sert au cadrage du "spot" du terminal.

Le Correspondant peut faire toute une série de demandes d'échange en une seule fois en chainant les IOCB entre eux.

- Test de la voie

Cet ordre permet au Correspondant de savoir si une voie logique est ouverte ou fermée. N'oublions pas, en effet, que c'est le Superviseur qui détermine la configuration du CCR, le Correspondant se contentant de travailler sur cette dernière qu'il doit donc être capable de connaitre. C'est pour cela qu'il a parfois besoin de savoir si une voie logique est ouverte ou non, surtout dans le cas d'une SU.service car il peut ainsi savoir si le service est en fonctionnement.

- Relance de la voie

Une SU.terminal se bloque sur certains évènements, ceci afin de stopper les échanges et de laisser la voie dans un état cohérent jusqu'à ce que le Correspondant, ayant fini son travail provoqué par l'évènement, la débloque par un ordre de relance.

- Annulation d'une réception

Cet ordre sert aussi bien pour les SU.terminal que pour les SU.service. Il permet de récupérer le buffer fourni lors d'une demande d'entrée pour recevoir un pli d'un utilisateur ou d'un service.

Par exemple quand un usager fait un appel au Correspondant (il tape (AC)) alors qu'il a la main vis à vis d'un service appelé, donc que le Correspondant utilisateur a passé un buffer de réception à la SU.terminal pour recevoir une <Ligne E>, le Correspondant utilisateur récupère ce buffer (il annule la réception de <Ligne E>) avant de se mettre en réception d'une <Ligne AC>.

Lorsque le Correspondant service veut rompre une liaison il envoie

un DCNX et se met en réception pour que le service lui réponde. Si celui-ci ne le fait pas le Correspondant décide, au bout d'un certain temps, que la liaison est rompue, mais il a besoin de récupérer le buffer de réception qu'il avait fourni.

- Modification de la table de contrôle

Cet ordre n'est adressable qu'aux SU des terminaux locaux. C'est lui qui permet au Correspondant de s'adresser au gestionnaire du terminal pour lui dire d'utiliser les caractères de contrôle déterminés par l'usager (cf. paragraphe 6.3.1).

- Demande de l'accès de l'entité située à l'autre bout de la voie

Nous avons vu au paragraphe 6.2 que le Correspondant a parfois besoin de connaitre la localisation de ses interlocuteurs (leur accès) et qu'il ne peut avoir cette information qu'en la demandant à la voie logique.

7.2.2. - Evènements signalés par la voie logique au Correspondant :

- Fin d'entrée / sortie

Quand une demande d'échange a été complètement traitée par la voie logique elle le signale au Correspondant en lui rendant l'IOCB.

- Evènements LCE

Ce sont les évènements appelés en LCE (AC), (E1) et (E2) qui font partie du dialogue entre Correspondant utilisateur et utilisateur mais, étant asynchrones, doivent passer par un canal particulier. La voie logique ne les interprète pas, elle les répercute chez le Correspondant. Ce genre d'évènement ne peut venir que d'une SU.terminal puisqu'il s'agit de "codage" du LCE. La voie est bloquée par la montée d'un tel évènement: aucun échange n'est plus possible, ni la montée d'un autre évènement. La voie doit être relancée par le Correspondant.

7.3. - REALISATION DES VOIES LOGIQUES DE COMMUNICATION

7.3.1. - Rappel sur l'architecture du CCR et du Frontal :

Si le Correspondant voit toutes les voies logiques de communication sous une même forme, elles ne sont pas toutes réalisées de la même façon puisque dépendant étroitement de leur support sur le Frontal : les FU (Unités Fonctionnelles). Nous avons vu au paragraphe 3.2.2 qu'une FU est le moyen de connexion entre le Frontal et les entités physiques du CCR où sont réalisées les applications ou gérés les terminaux distants: autres ordinateurs et réseaux externes.

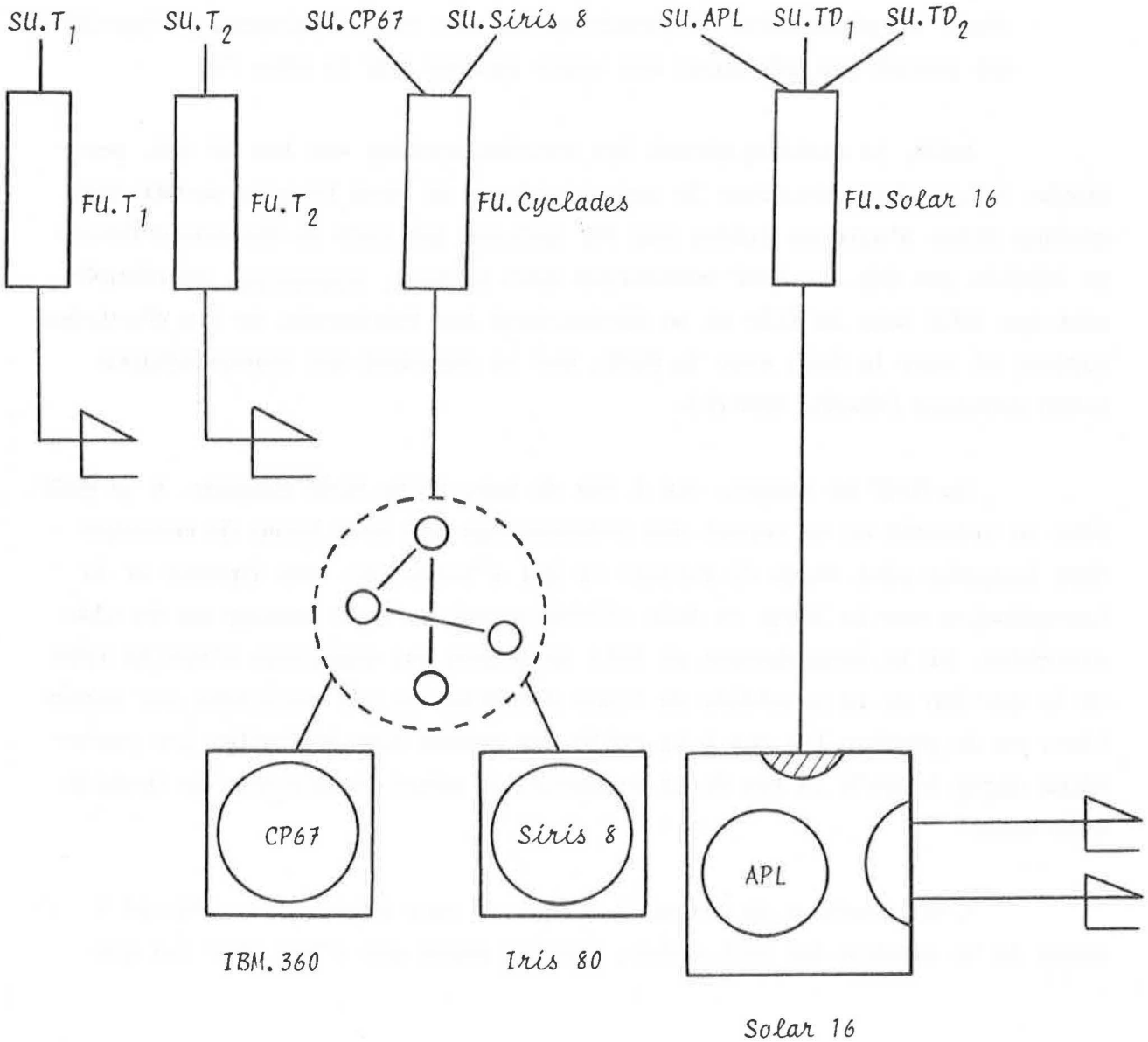


Figure 33

Plusieurs voies logiques peuvent donc passer par la même FU. On a donc un multiplexage / démultiplexage des SU sur les FU au niveau du Frontal. Sur les autres entités physiques du CCR il est aussi nécessaire d'avoir un multiplexage / démultiplexage des voies logiques de communication (telles qu'elles sont vues localement) sur la ligne qui vient du Frontal.

Une FU est composée de trois machines: M:GP, M:GT et M:TR et d'une (ou deux) file(s) d'attente qui sert(servent) à l'alimenter. Aux niveaux M:GT et M:TR on ne connait plus que la ligne, donc la FU. Tandis que la M:GP connait encore les SU. On voit donc que les M:GP ont deux rôles:

- assurer le parallélisme des voies logiques sur la FU
- gérer les protocoles de communication des voies logiques, protocoles qui seront les mêmespour des voies passant par la même FU.

M:DM, la machine gérant les entrées/sorties sur les SU (cf. paragraphe 3.2.2) doit connaitre la correspondance SU vers FU pour savoir dans quelles files d'attente (liées aux FU) enfiler les IOCB de demande d'échange (dédiés aux SU). La M:GP communique avec le haut, notamment en récupérant ces IOCB dans la file et en déclenchant les évènements de fin d'entrée/sortie, et vers le bas, avec la M:GT, par le standard des communications entre machines [Gouda, GEN-11].

La M:GP ne traite, vis à vis du bas, qu'en mode message. A la M:GT elle ne transmet et ne reçoit des informations que sous forme de messages dans lesquels elle range ou extrait ce qui l'intéresse. Peu importe si la transmission sur la ligne se fait effectivement en mode message ou en mode caractère. Si la transmission se fait caractère par caractère c'est le rôle de la gestion de la procédure de ligne d'émettre le pli caractère par caractère, ou de stocker les uns à la suite des autres dans un buffer les caractères reçus jusqu'à la fin de la transmission avant de délivrer le message à la M:GP.

C'est surtout ce niveau M:GP qui va nous intéresser. D'abord à cause de la gestion des protocoles, ensuite parce que c'est chez lui que

sont répercutés les ordres du Correspondant, il les répercute parfois à son tour aux niveaux inférieurs.

Il ne faut pas oublier que certaines commandes peuvent venir du Superviseur, comme l'ouverture et la fermeture de FU ou de SU. Il y a au moins un processus attaché à une FU (celui qui s'alimente à la file d'attente), comme celle-ci n'est pas permanente ce processus ne l'est pas non plus. Ouvrir la FU c'est créer ce processus, à charge pour lui d'initialiser le reste de la FU: création d'autres processus si c'est nécessaire, initialisation des données, ouverture de la ligne. Ouvrir ou fermer une SU c'est prévenir la M:GP pour qu'elle mette à jour ses données concernant cette SU et entreprenne certaines actions nécessaires. Pour fermer la FU on prévient la M:GP qui va fermer les SU encore existantes, fermer la ligne et déclencher la destruction des processus attachés à la FU.

Il n'y a que deux types de voie logique: les voies logiques des terminaux et les voies logiques des services. Mais on compte quatre types de FU sur le Frontal: les FU.terminal, les FU.Frontal, les FU.ordinateur connecté et les FU.réseau externe.

7.3.2. - FU.terminal :

C'est une FU liée à un terminal connecté au Frontal. Il n'y a donc qu'une SU (la voie logique joignant le Correspondant utilisateur au terminal) sur cette FU. Cette dernière comprend les trois machines habituelles mais la M:GP est réduite à rien puisqu'il n'y a pas de protocole à gérer ni de multiplexage / démultiplexage à faire. Du côté du terminal la gestion de la procédure et de la ligne est faite entièrement par matériel. Les évènements LCE sont déclenchés par reconnaissance des caractères de contrôle (définis par la table de contrôle du terminal, modifiable par ordre du Correspondant utilisateur). Un seul processus est attaché à la FU (les terminaux sont half duplex) et le blocage de la voie logique se fait par blocage du processus.

7.3.3. - FU.Frontal :

Lorsqu'un service est localisé sur l'ordinateur Frontal la voie logique de communication le reliant au Correspondant service se trouve réduite à sa plus simple expression. La FU par laquelle elle passe se réduit à sa (ou ses) file(s) d'attente et à un niveau de multiplexage / démultiplexage, car on peut quand même avoir plusieurs SU sur une telle FU. C'est le cas du service d'entrée/sortie gérant les fichiers appareils (cf. paragraphe 4.2.2) qui est vu comme plusieurs services: éditeur de texte, imprimante, lecteur, perforateur, etc ... par le Correspondant.

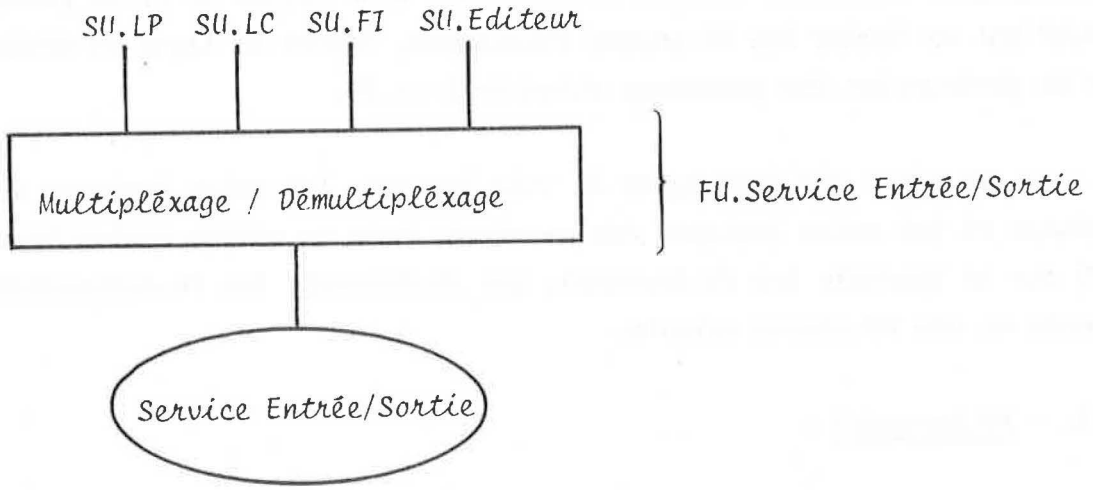


Figure 34

La FU doit aussi être capable de recevoir les ordres aux voies logiques que peut envoyer le Correspondant afin que ses SU soient bien vues comme des SU standards.

7.3.4. - FU.ordinateur connecté et FU.réseau externe :

Ce sont à elles, à leur M:GP, que nous allons plus particulièrement nous intéresser dans les deux chapitres suivants.

Elles possèdent tous les niveaux hiérarchiques et la gestion des protocoles y est importante. En effet ce sont par elles que passent les voies

logiques de communication qui relient le Correspondant aux entités distantes: services localisés sur d'autres ordinateurs et terminaux non gérés par le Frontal. Ainsi leur rôle de moyen de transport est fondamental. La voie logique doit garantir le transport de l'information aux entités situées à ses deux bouts. Donc les protocoles gérés doivent assurer [Pouzin, GEN-12]:

- le séquencement des informations
- le contrôle d'erreur
- le contrôle de flux
- le passage d'évènements asynchrones.

Ce dernier point est important car une voie logique d'un terminal distant doit se comporter comme celle d'un terminal local, notamment par le passage d'évènement LCE au Correspondant utilisateur. Des problèmes vont apparaitre à cause de cette nécessité de simuler les terminaux distants comme des terminaux locaux et des différences vont voir le jour entre les voies logiques des terminaux et celles des services, bien que passant par la même FU. C'est au niveau M:GP que l'on sera obligé de faire cette distinction.

Les protocoles gérés sont soit imposés de l'extérieur, soit créés de toutes pièces pour le CCR. Nous allons d'abord voir ce dernier cas avec le Protocole Ordinateur Connecté défini pour gérer les communications avec n'importe quel ordinateur dont le système n'a pas de protocole de communication avec l'extérieur. Puis nous verrons les protocoles de Cyclades auxquels nous avons dû nous soumettre puisqu'étant ceux d'un réseau externe existant.

CHAPITRE 8

PROTOCOLE ORDINATEUR CONNECTE

8.1. - NECESSITE

Introduire un nouvel ordinateur dans le CCR nécessite de développer sur le Frontal la FU liée à la ligne qui réalise la connexion physique, c'est à dire développer un ensemble (M:GP , M:GT , M:TR).

Généralement les M:TR et M:GT peuvent être prises parmi celles existant déjà sur le Frontal. La M:TR dépend du coupleur et seule l'installation d'un nouveau type de coupleur oblige d'en réaliser une nouvelle. Nous ne devons écrire une M:GT que dans le cas où le nouvel ordinateur ne sait gérer qu'une procédure de transmission qui lui est particulière et d'un type non standard.

Si ce calculateur offre un protocole de communication avec l'extérieur nous avons tout intérêt à développer une M:GP pour ce protocole afin d'éviter de toucher au système du nouveau venu. Dans le cas contraire nous avons défini un Protocole Ordinateur Connecté (POC) suffisamment complet pour gérer toutes les communications entre deux ordinateurs, et simple pour être facilement introduit sous un système d'exploitation donné.

Ainsi le travail se trouve toujours réduit à l'installation d'une gestion de protocole sur un seul ordinateur: soit d'une M:GP sur le Frontal, soit de la gestion du Protocole Ordinateur Connecté sur l'ordinateur à connecter.

Dans la configuration actuelle du CCR le Protocole Ordinateur Connecté est utilisé pour la FU.Solar 16 et la FU.PDP 11.

8.2. - DEFINITION DU PROTOCOLE ORDINATEUR CONNECTE

Le Protocole Ordinateur Connecté doit assurer les possibilités décrites au paragraphe 7.3.4. Mais en fait il recouvre deux protocoles de communication:

- Le Protocole Appareil Service (PAS), géré dans les voies logiques des services.

- Le Protocole Appareil Terminal (PAT), géré dans les voies logiques des terminaux distants.

Le Protocole Appareil Terminal fournit le moyen de passer des évènements asynchrones et un contrôle du dialogue pour que le terminal distant puisse se comporter comme un terminal local. Le Protocole Appareil Service quant à lui est plus simple : dans le dialogue entre Correspondant service et un service il existe des évènements et un contrôle, mais ils sont passés au niveau du LCI et la voie logique n'a pas à s'en préoccuper.

8.2.1. - Protocole Appareil Service :

Les échanges se font sur une voie logique de communication. L'adressage fourni par le Protocole Appareil Service utilise le numéro de SU attribué à la voie logique, chaque pli le contient donc pour indiquer sur quelle voie se fait l'échange.

Les plis échangés ont une longueur maximale fixée actuellement à 256 octets et ne transportent qu'une ligne LCI à la fois.

Il y a deux types de plis: les lettres (LT) et les acquittements (ACK).

LT :

n° SU	CRD	0	MY.REF	YR.REF	texte

ACK :

n° SU	CRD	1		YR.REF

Seules les lettres, formées d'un en-tête Protocole Appareil Service et d'un texte transparent au protocole, permettent le transport de données.

Toutes les lettres sont référencées (MY.REF) séquentiellement, ce qui permet:

- Le bon séquencement: la lettre n n'est reçu que si la lettre n-1 a été reçue.

- Le contrôle d'erreur: lorqu'une lettre arrive on l'acquitte en envoyant sa référence (YR.REF) à l'émetteur. Un même acquittement peut ainsi acquitter plusieurs lettres. Un réveil est armé par l'émetteur à l'envoi d'une lettre pour éviter une attente indéfiniede l'acquittement. Si ce réveil arrive à terme, donc que l'acquittement n'est pas venu, la lettre est renvoyée. Au bout d'un certain nombre de renvois non réussis on déclare l'envoi impossible.

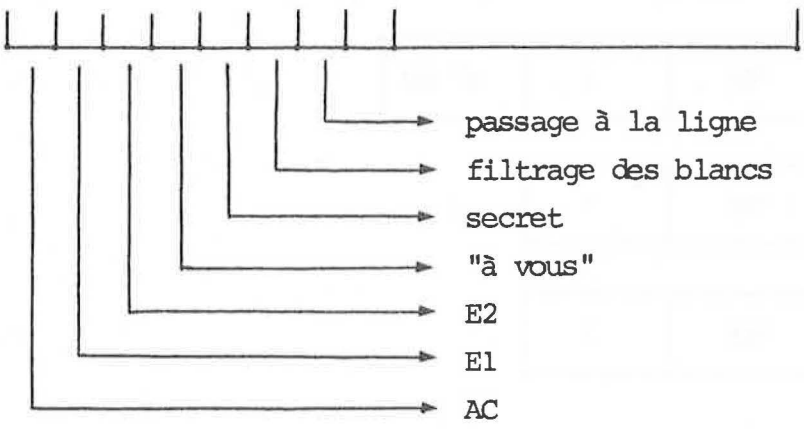
Le contrôle de flux se fait par échange mutuel de crédits (CRD) qui représentent le nombre de lettres que le récepteur autorise l'émetteur à envoyer jusqu'à attribution de nouveaux crédits.

Les crédits et acquittements sont passés dans les plis ACK, mais aussi dans les en-tête de lettre.

8.2.2. - Protocole Appareil Terminal :

Le Protocole Appareil Terminal contient toutes les possibilités du Protocole Appareil Service et offre en plus un contrôle du dialogue qui est codé dans un mot de l'en-tête d'une lettre.

Ce mot se présente sous la forme:



Les bits passage à la ligne, filtrage des blancs et secret servent à transmettre à la gestion du terminal distant les fonctions correspondantes indiquées dans l'IOCB (cf. paragraphe 7.2.1) qui, dans le cas du terminal local, sont exécutées par la M:GT.

Le passage de la main se fait par le bit "à vous". Pour respecter le contrôle de dialogue implicitement défini par le LCE la main est donnée au terminal distant lorsque le Correspondant utilisateur se met en réception sur la voie, le terminal perdant la main dès qu'il envoie une ligne.

Les bits AC, E1 et E2 permettent la transmission des évènements LCE correspondants.

On peut utiliser les lettres sans leur texte, uniquement pour transmettre le contrôle de dialogue (les évènements notamment) mais les crédits ne servent que pour les lettres avec texte.

8.2.3. - Commandes d'ouverture et de fermeture d'une voie :

On remarque que le Protocole Ordinateur Connecté est un protocole de communication géré sur une voie ouverte. Or il faut bien ouvrir la SU avant de commencer les échanges. Ceci est fait par le Superviseur, mais l'ordre doit être connu aux deux bouts de la voie. C'est pour cela qu'ont été définies des commandes spéciales, qui ne font pas partie du Protocole Ordinateur Connecté même si elles passent sur la même ligne.

Ouverture SU	'FF	0	n° SU
Fermeture SU	'FF	1	n° SU
OK	'FF	2	
non OK	'FF	3	

L'extrémité Frontal de la voie logique, qui reçoit l'ordre du Superviseur, envoie la commande correspondante à l'autre extrémité qui doit lui répondre par OK ou non OK. Il ne s'agit pas d'un protocole d'établissement de flot ou de liaison, mais de l'ouverture (ou la fermeture) "physique" de la voie logique de communication.

8.3. - LA MACHINE GERANT LE PROTOCOLE ORDINATEUR CONNECTE (M:GPOC)

La M:GPOC est la machine qui gère le Protocole Ordinateur Connecté dans la FU d'accès à l'ordinateur connecté. Elle recouvre deux sous-machines parallèles:

- M:GPAS qui gère le Protocole Appareil Service
- M:GPAT qui gère le Protocole Appareil Terminal.

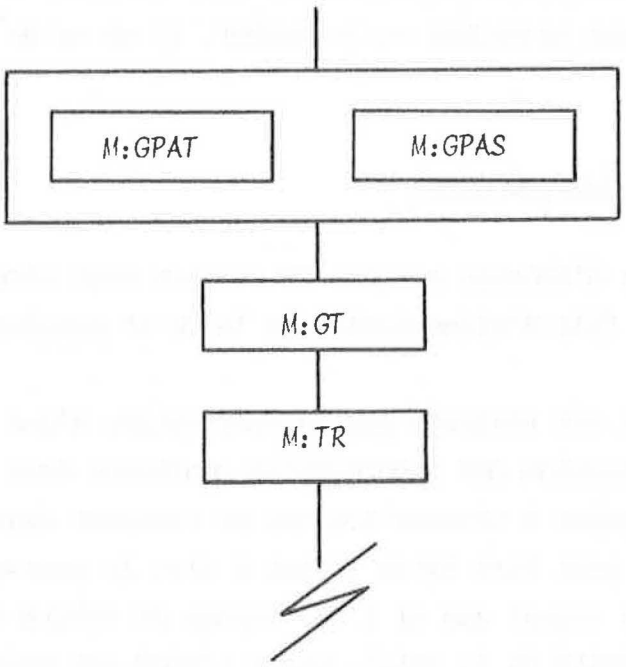


Figure 35

8.3.1. - Structure de données :

Pour que la M:GPOC soit utilisable dans n'importe quelle FU d'accès à un ordinateur il faut que ses données soient essentiellement dynamiques pour

assurer la réentrance. Notamment les SU qui passent par elle ne sont pas fixées.

Un contexte est associé à chaque SU, il est créé dynamiquement à l'ouverture de la SU (ordre du Superviseur) et accessible par la mise en place d'un mécanisme standard d'accès en fonction du numéro de SU. Il est détruit à la fermeture de la voie logique ainsi que le moyen de l'accéder.

Ce contexte contient les informations nécessaires à l'identification de la voie: numéro de SU et type (terminal ou service) tous deux fournis à l'ouverture, et à la gestion du protocole: références des lettres, compteur d'envoi, crédits, etc ...

8.3.2. - Machine gérant le Protocole Appareil Service :

Les ordres venant du Correspondant sont dirigés, suivant le type de la voie logique, vers la M:GPAS ou la M:GPAT. Il en va de même des plis arrivant de la ligne.

- Ordres du Correspondant :

Les demandes d'échange sur une SU.service sont récupérées, sous forme d'IOCB, dans la file d'alimentation de la FU et passées à la M:GPAS.

Une émission est réalisée par la fabrication d'une lettre: l'entête est formaté en fonction des informations contenues dans le contexte SU, le texte étant les données à transmettre qui se trouvent dans le buffer de l'IOCB. L'émission ne peut être faite (c'est à dire le passage du pli à la M:GT pour envoi sur la ligne) que si l'on dispose de crédits d'émission (attribués par l'autre extrémité de la voie). Sinon l'IOCB est stocké dans le contexte jusqu'à l'arrivée de nouveaux crédits. On introduit ainsi un niveau de désynchronisation entre l'alimentation de la FU et l'émission sur la ligne, ceci afin de pouvoir contrôler le flux.

L'évènement fin d'entrée/sortie relatif à l'IOCB n'est déclenché que lorsqu'on est sûr que la lettre correspondante est bien arrivée (elle a été acquittée) ou qu'elle s'est perdue (plusieurs renvois successifs infructueux).

Un IOCB de réception est stocké dans le contexte SU. C'est le nombre de ces IOCB qui détermine les crédits que la M:GPAS attribue à son interlocuteur. Une annulation de réception se traduit donc par la récupération d'un IOCB stocké et diminution des crédits de réception.

- Arrivées de pli :

Une lettre envoyée par l'ordinateur connecté est contrôlée à son arrivée: sa référence doit être celle de la lettre précédente plus un (modulo 256) et il y a au moins un IOCB de réception (donc un crédit) pour la recevoir. Elle est alors acquittée et son texte est rangé dans le buffer de l'IOCB de réception qui est rendu au Correspondant service (fin de réception).

L'arrivée de l'acquittement d'une lettre, par pli ACK ou dans l'entête d'une lettre, permet d'acquitter le (ou les) IOCB d'émission correspondant(s) (n'oublions pas que plusieurs lettres peuvent être acquittées à la fois) et l'attribution de nouveaux crédits autorise de faire les émissions en attente (s'il y en a).

8.3.3. - Machine gérant le Protocole Appareil Terminal :

La M:GPAT s'occupe de la partie du Protocole Appareil Terminal semblable au Protocole Appareil Service comme la M:GPAS, mais elle doit en plus gérer le contrôle de dialogue et les évènements.

Remarquons que dans le mot de contrôle de dialogue situé dans l'entête d'une lettre les trois premiers bits ne sont utilisés que dans le sens terminal vers Correspondant utilisateur et les bits 4 à 6 dans le sens Correspondant utilisateur vers terminal. Seul le "à vous" sert dans les deux sens. Ce contrôle de dialogue peut passer aussi bien dans l'en-tête d'une lettre avec texte que dans celui d'une lettre sans texte suivant qu'il peut être associé ou non à une demande d'émission du Correspondant ou du terminal.

Les évènements LCE (AC), (E1) et (E2), représentés respectivement par les bits 0, 1 et 2 du mot de contrôle de dialogue, sont reconnus par la

M:GPAT et répercutés chez le Correspondant utilisateur de la même manière que s'ils venaient d'un terminal local. La voie logique est alors bloquée, c'est à dire que la M:GPAT n'accepte plus ni les demandes d'échange du Correspondant utilisateur ni les lettres avec texte venant du terminal sur cette voie jusqu'à ce que le Correspondant donne l'ordre de déblocage. Si l'évènement arrive dans l'en-tête d'une lettre avec texte ce dernier est d'abord transmis au Correspondant avant la prise en compte de l'évènement.

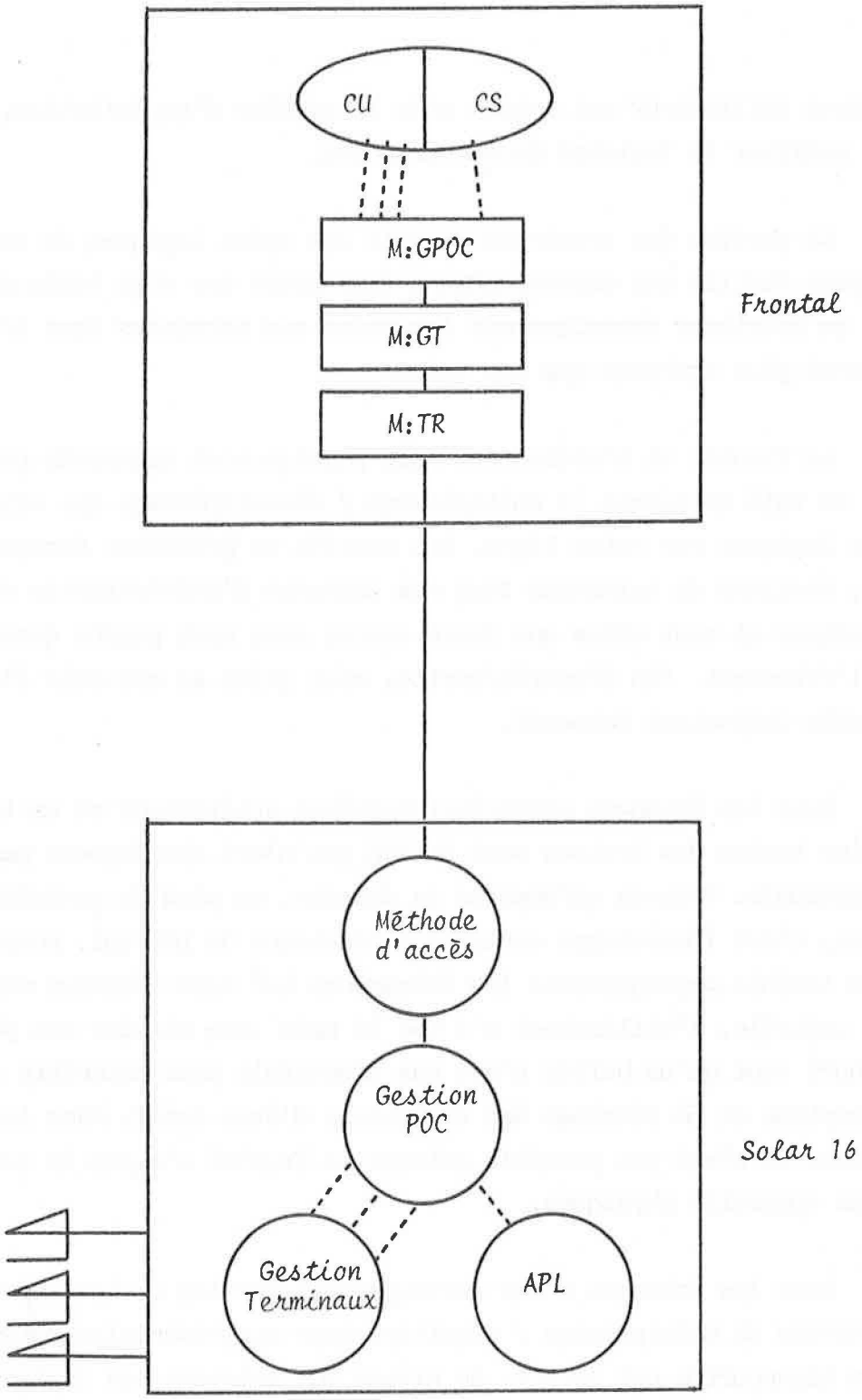
Pour les fonctions de l'IOCB passage à la ligne, filtrage des blancs et secret la M:GPAT se contente de positionner les bits correspondants dans l'en-tête de la lettre qu'elle émet, c'est à son interlocuteur de traduire ces fonctions d'une façon ou d'une autre pour qu'elles soient exécutées par la gestion des terminaux de l'ordinateur connecté.

Ainsi l'interface entre Correspondant utilisateur et voie logique d'un terminal est parfaitement respecté par la SU.terminal distant.

Pour être conforme aux règles implicites de passage de la main du LCE toute lettre avec texte (donc contenant une ligne) venant du terminal doit comporter le bit "à vous", car l'utilisateur perd la main dès qu'il envoie une ligne. Dans le cas contraire la lettre est déclarée erronée et n'est pas reçue: son texte n'est pas transmis au Correspondant utilisateur et elle n'est pas acquittée. Dans le sens Correspondant utilisateur vers le terminal la M:GPAT envoie le "à vous" dès que lui parvient un IOCB d'entrée, c'est à dire que le Correspondant utilisateur passe la main en se mettant en réception sur la voie. Il faut noter que dans la réalisation actuelle le Correspondant utilisateur donne souvent des IOCB chainés: un IOCB d'émission suivi d'un IOCB de réception. La M:GPAT a alors intérêt à émettre le "à vous" dans l'en-tête de la lettre qui correspond à l'émission, afin d'éviter d'encombrer la ligne avec plusieurs plis.

8.4. - VOIE LOGIQUE GERANT LE PROTOCOLE ORDINATEUR CONNECTE

Sur l'ordinateur connecté on va aussi avoir une gestion du protocole dont les mécanismes sont semblables à ceux de la M:GPOC. C'est généralement



Exemple de connexion avec le Solar 16 qui offre le service APL et possède des terminaux

Figure 36

un programme utilisateur qui tourne sous le système d'exploitation, ce qui évite de modifier le logiciel du constructeur.

La gestion des terminaux connait les voies logiques de communication et peut établir une correspondance fixe entre une voie logique et un terminal ou attribuer dynamiquement les voies aux terminaux dans le cas où ceux-ci sont plus nombreux que les voies.

Le Frontal et l'ordinateur sont physiquement connectés par une ligne et on voit nettement le multiplexage / démultiplexage qui existe entre les voies logiques sur cette ligne. Les entités en présence: Correspondant, services, gestions de terminaux font des demandes d'entrée/sortie sur les voies logiques et sont sûres que leurs envois sont bien passés quand leur parvient l'évènement fin d'entrée/sortie, ceci grâce au contrôle d'erreur du Protocole Ordinateur Connecté.

Dans les échanges entre Correspondant utilisateur et un terminal distant les textes des lettres sont du LCE qui n'est absoluement pas touché par le protocole. Tout ce qu'apporte ce dernier, en plus du contrôle d'erreur et de flux, c'est l'enveloppe contrôle de dialogue du LCE qui, pour les terminaux locaux se traduit physiquement: les évènements LCE sont reconnus comme caractères de contrôle, l'utilisateur n'a pas la main (son clavier est physiquement bloqué) tant qu'un buffer n'est pas disponible pour permettre au mécanisme de réception et de stockage des caractères d'être actif. Avec les terminaux distant ce n'est pas possible puisque le Frontal n'a pas le contrôle direct des appareils physiques.

Avec les échanges entre Correspondant service et les services apparait un niveau de multiplexage / démultiplexage supplémentaire puisque l'information transportée est du LCI. Au niveau LCI existent les liaisons utilisateur - service ou service - service. Le Correspondant service et les enveloppes réseau des applications font la correspondance liaison vers voie logique, les gestion du Protocole Ordinateur Connecté font la correspondance voie logique vers une ligne. L'information qui circule sur la voie logique d'un service est donc découpée en plusieurs zones bien précises dont chacune n'est compréhensible et utilisable qu'à un niveau hiérarchique donné.

compréhensible par les gestions du Protocole Ordinateur Connecté

compréhensible par le Correspondant service et les enveloppes réseau des services

compréhensible par les utilisateurs et les services locaux

CHAPITRE 9

PROTOCOLES CYCLADES

9.1. - RAPPELS SUR CYCLADES [Pouzin, RES-11]

9.1.1. - Généralités :

Cyclades est un réseau d'ordinateurs hétérogène et général. N'importe quel ordinateur peut lui être connecté et il n'a pas été conçu en fonction d'une application particulière. Il s'agit d'un outil servant de support à diverses applications, en leur permettant éventuellement de dialoguer entre elles. Il se décompose en trois niveaux principaux:

- Le réseau de commutation de paquets Cigale.
- Les Stations de Transport (ST)
- Les abonnés (AB) sur les sites (hôtes).

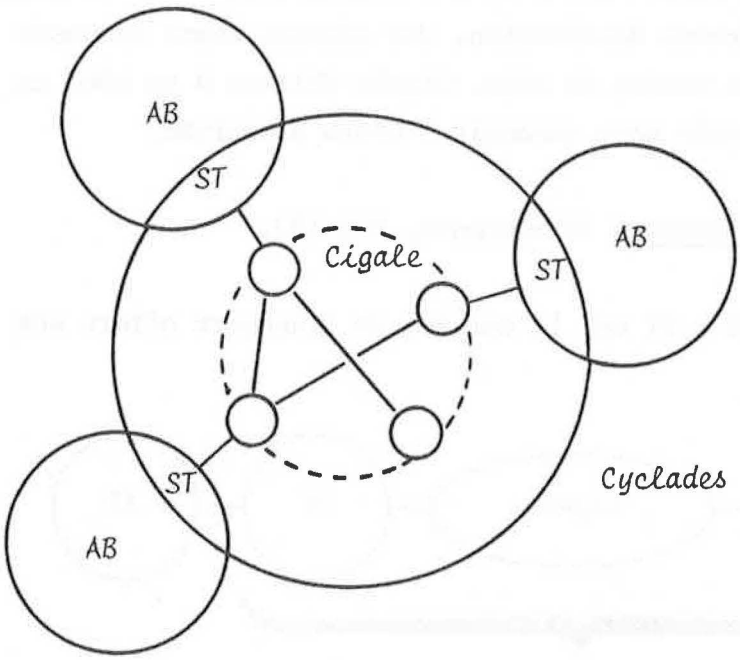


Figure 37

Ces différents niveaux sont régis par des protocoles particuliers:

- Protocole Cigale
- Protocole ST - ST
- Protocole Abonné.

9.1.2. - Cigale [Grangé, RES-12]:

C'est un réseau de commutation de paquets formé de mini-calculateurs (Mitra): les noeuds, connectés entre eux par des lignes téléphoniques formant un réseau maillé. Il assure une connexion permanente entre tous les utilisateurs du réseau (ordinateurs ou terminaux). Les noeuds sont reliés en mode synchrone selon la procédure de transmission TMM-UC full duplex. Les hôtes communiquent par paquets de taille maximale de 255 octets qu'ils délivrent à Cigale. Les paquets sont transmis de noeuds en noeuds suivant un itinéraire évolutif déterminé à chaque noeud traversé selon un algorithme de routage tenant compte de la densité du traffic et de l'occupation des diverses lignes.

Chaque hôte est connecté à un noeud particulier par une ligne. Il voit Cigale comme une boite aux lettres à laquelle il donne un paquet avec adresse origine et adresse destination, une adresse étant composée d'un numéro de région et d'un numéro de site. Cigale délivre à un hôte un paquet qui lui est destiné, mais sans garantir l'ordre d'arrivée.

9.1.3. - Station de Transport [Zimmermann, RES-10]:

Le couple ST - ST est le service de transport offert aux abonnés.

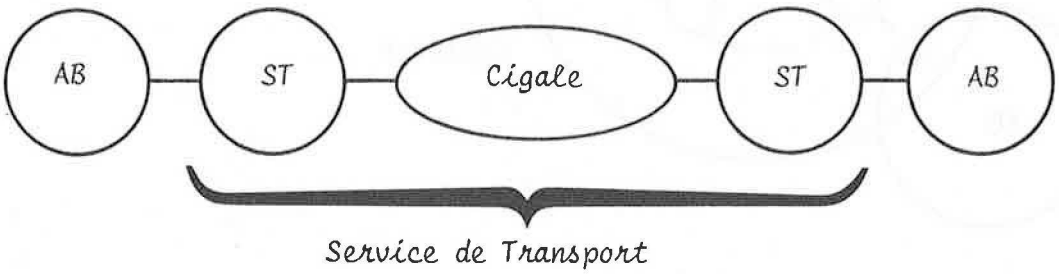


Figure 38

Le protocole ST - ST assure la restauration du séquencement et un certain nombre de contrôle. A l'heure actuelle Cyclades fonctionne avec le protocole ST.2.

Une ST est connue du réseau par son numéro (n° région - n° site). Elle offre un adressage composé de portes (PT), son (ou ses) abonné(s) utilisant ces portes comme point d'entrée sur le réseau. Tout échange d'information se fait sur un flot (FL) qui est l'association de deux portes. Sur les flots les ST échangent des commandes de flot. Aux abonnés deux types d'envoi sont permis:

- les télégrammes (TG) courts et de longueur fixe (16 bits)
- les lettres (LT) de longueur variable.

Comme la taille des paquets circulant dans Cigale est limitée, pour les lettres trop longues les ST fournissent un mécanisme de fragmentation - réassemblage. La ST émettrice découpe la lettre en fragments (FR) de taille fixe, sauf le dernier dont la taille peut être variable, et les envoie les uns après les autres. La ST réceptrice reconstitue la lettre en remettant les fragments dans l'ordre (à cette fin ils sont numérotés) et ne la délivre à l'abonné que lorsqu'elle est entière.

Il existe deux modes de fonctionnement pour un flot:

- Le service de base: La création et la destruction du flot sont décidées unilatéralement. La ST n'assure que la fragmentation et le réassemblage des lettres.

- Les services additionnels: L'ouverture d'un flot avec services additionnels requiert une négociation de session entre les deux ST afin de se mettre d'accord. Sur un tel flot on a:

. Un contrôle d'erreur sur la lettre entière: Chaque lettre est référencée et quand elle est parvenue à la ST destinatrice celle-ci l'acquitte en renvoyant sa référence à la ST émettrice. Comme les ST numérotent les lettres séquentiellement et ne les acquittent que dans l'ordre de leur numéro un acquittement peut acquitter plusieurs lettres à la fois. De plus si au bout d'un certain temps une lettre envoyée n'a toujours pas été acquittée la ST la réexpédie. Elle réémet la lettre un certain nombre de fois avant de déclarer l'émission impossible.

. Un contrôle de flux: A l'initialisation du flot les ST s'accordent sur une longueur maximale des lettres particulière au flot et, en cours d'échan-

ge, s'allouent des crédits qui représentent le nombre de lettres que l'émetteur est autorisé à envoyer jusqu'à attribution de nouveaux crédits.

. Un adressage réduit permettant de donner un diminutif au flot (au lieu de l'identification n° PT - n° PT) avait été prévu mais il n'est pas réalisé à l'heure actuelle.

Les commandes de flot sont:

FL-INIT : initialisation du flot
FL-TERM : fermeture du flot
FL-TG : télégramme
FL-LT : fragment de lettre (qui peut être la lettre entière)
FL-ACK : acquittement et crédits (sont aussi passés dans FL-LT)
FL-ERROR: signale une erreur sur une commande de flot reçue.

9.1.4. - Abonnés :

La notion d'abonné peut recouvrir aussi bien un programme, une application, un terminal (ou un ensemble de terminaux). Il peut y avoir un ou plusieurs abonnés par ST. Le seul protocole abonné défini est le VTP (Virtual Terminal Protocol) déjà vu au Chapitre 5 :

- Le protocole de connexion est réduit à sa plus simple expression par demande d'ouverture d'un flot entre une porte locale et une porte particulière d'un abonné d'un autre site.

- Le VTP permet à un serveur de voir tout terminal du réseau sous une forme unique et ainsi de n'avoir besoin que d'une seule gestion de dialogue avec les terminaux. Le client, généralement un concentrateur de terminaux ou un terminal intelligent, simule ce terminal virtuel et le protocole lui donne le moyen de dialoguer d'une façon standard avec tous les serveurs.

9.2. - LA FU.CYCLADES

C'est elle qui permet au Frontal l'accès à Cyclades. Nous avons dû nous adapter aux spécifications définies par le réseau externe qui est réalisé

indépendamment du CCR.

La FU.Cyclades comprend les trois niveaux hiérarchiques: M:GP , M:GT et M:TR.

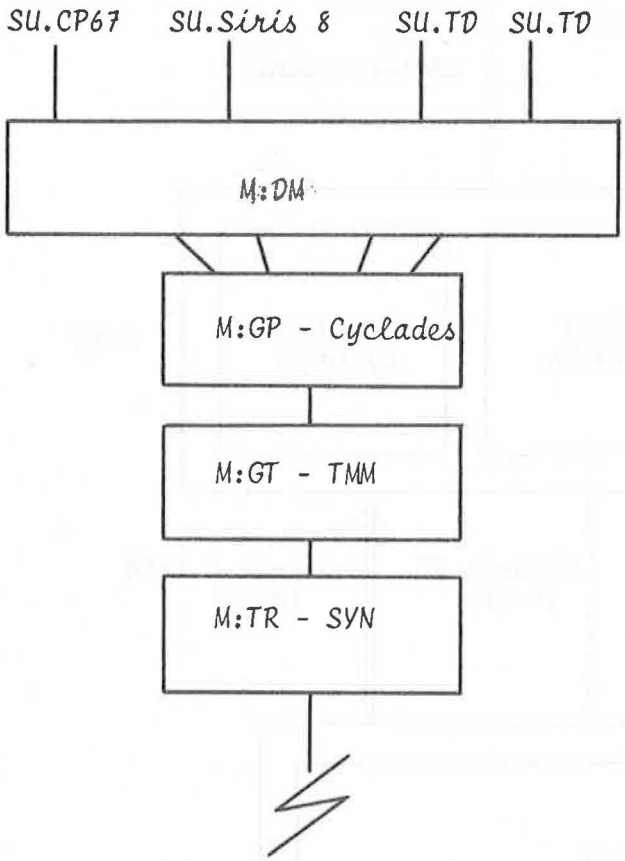


Figure 39

- La M:TR-SYN : gère le modem interfaçant la ligne reliée au noeud de Cigale en mode synchrone full duplex et mode caractère.

- La M:GT-TMM : gère la procédure TMM-UC adoptée sur toutes les liaisons de Cyclades.

- La M:GP-Cyclades : elle gère tous les protocoles de Cyclades. On a vu que ceux-ci se répartissent en trois couches bien définies. On doit donc retrouver cette décomposition au niveau de la M:GP qui est formée de trois sous-machines travaillant en série:

. Machine gérant le protocole Cigale
. Machine gérant le protocole ST - ST
. Machine gérant le protocole abonné.

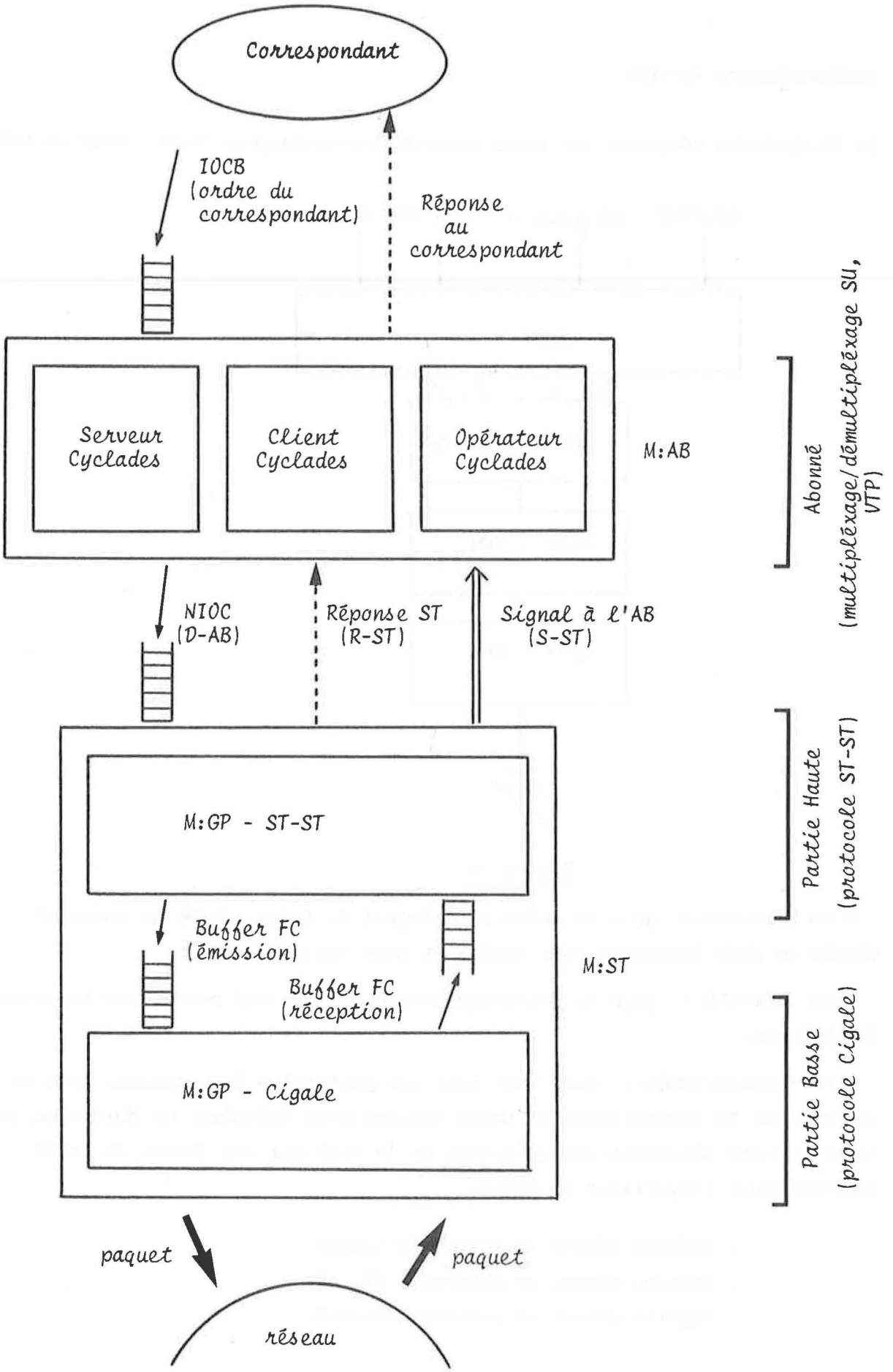


Figure 40 - Stucture de la M:GP - Cyclades

C'est la Station de transport qui gère le protocole ST - ST et l'accès à Cigale dans Cyclades, donc les deux sous-machines s'occupant de ces protocoles vont être regroupées dans une même machine, la Machine Station de Transport (M:ST).

L'abonné Cyclades ou Machine Abonné (M:AB) a plusieurs fonctions. Il gère le VTP dans les deux sens: en tant que client pour les terminaux locaux et en tant que serveur pour les terminaux distants. C'est à lui qu'incombe de faire le multiplexage / démultiplexage des SU sur la FU. Il doit montrer les voies logiques des entités de Cyclades comme des voies logiques standards.

Nous allons étudier plus en détail la M:ST et la M:AB, mais il faut remarquer que dès le niveau ST on ne connait plus que Cyclades. Le parallélisme des SU est entièrement laissé à la responsabilité de l'abonné, elles passent toutes par la ST de la même façon, exactement comme dans un réseau routier plusieurs routes aboutissent à un carrefour qui les rassemble en une même voie. A partir de la ST la route est unique, c'est la route Cyclades qui se subdivise elle-même en plusieurs chemins mais d'une manière interne au réseau, c'est à dire transparente aux voies logiques qui ont l'impression d'emprunter la même voie.

9.3. - LA MACHINE STATION DE TRANSPORT (M:ST) [CCR-6]

9.3.1. - Présentation :

La ST ne connait qu'un seul abonné: l'abonné Cyclades, elle reçoit ses demandes, si nécessaire elle les met en forme du protocole ST - ST et les envoie sur le réseau.

Elle reçoit des paquets du réseau, si nécessaire elle prévient l'abonné et les met en forme compréhensible pour lui.

Pour les échanges courants l'abonné dispose d'un adressage formé des portes 0 à 'FFFF sauf les portes du type 'FFFX réservées pour des utilisations particulières (pour l'instant seule la porte "écho" : 'FFFE est im-

plémentée, elle renvoie systématiquement tout ce qu'elle reçoit).

La ST connait deux types de flot:

- Les flots en service de base (SB): cette notion recouvre à la fois l'ouverture d'une porte en réception et celle, sans négociation de session, d'un flot pour pouvoir émettre. A l'ouverture d'un tel flot (on peut aussi dire ouverture d'une porte) seule la porte locale est connue, la porte distante n'est précisée qu'en cours d'utilisation du flot (et peut varier), donc:

  . A l'émission l'abonné devra indiquer la destination.
  . En réception la porte locale se comporte comme un entonnoir recueillant tous les plis lui étant destinés quelle que soit leur origine.

- Les flots avec services additionnels (SA): l'identification du flot (ST-distante , PT-distante , PT-locale) est fixée pour toute la durée de vie du flot qui dispose obligatoirement du contrôle d'erreur et du contrôle de flux.

Une porte n'a pas besoin d'être ouverte pour qu'on puisse ouvrir un flot sur elle, en fait les portes n'ont pas d'existence réelle, elles servent d'identification pour distinguer les flots.

La ST attribue à un flot un numéro qu'elle communique à l'abonné, ainsi la désignation du flot dans leurs échanges peut se faire simplement.

La ST permet d'échanger des messages avec les STI (Stations Internes: les noeuds de Cigale).

9.3.2. - Interface avec l'abonné :

Les échanges entre l'abonné et la ST sont de trois types:

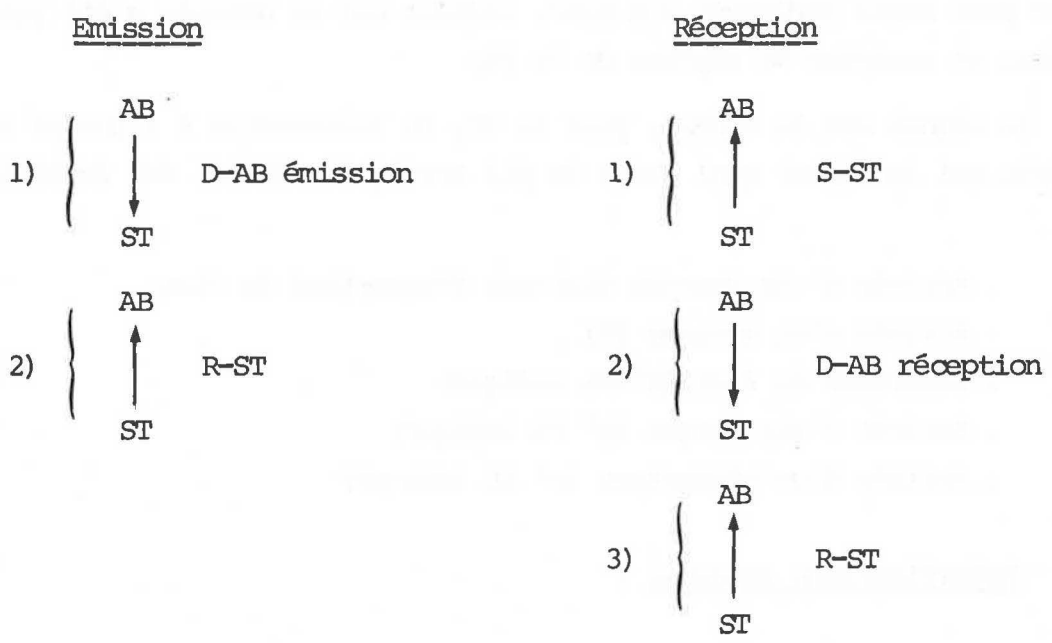
- demande de l'abonné à la ST : D-AB
- réponsede la ST à l'abonné : R-ST
- signal de la ST à l'abonné : S-ST

L'abonné peut:

- Ouvrir et fermer la ST
- Ouvrir et fermer un flot (en service de base ou avec les services additionnels)

- Accepter une ouverture ou une fermeture de flot (avec les services additionnels) venant d'un abonné d'un autre site (appelé abonné distant)
- Envoyer et recevoir des lettres et des télégrammes sur flot
- Echanger des messages avec les STI;

Le mécanisme général d'émission / réception a cette forme:



Les échanges, lorsque la ST est active, se font par l'intermédiaire d'un bloc de contrôle standard appelé NIOC (Network Input Output Control bloc) analogue à un IOCB mais dont la forme est plus adaptée à l'interface entre l'abonné et la ST.

D-AB : L'abonné remplit un NIOC et le passe à la M:ST. Une demande peut-être:

. Envoi d'un message STI
. Réception d'un message STI
. Ouverture d'un flot
. Réception d'une demande distante d'ouverture de flot (uniquement avec les services additionnels)
. Fermeture d'un flot
. Envoi d'un télégramme
. Réception d'un télégramme
. Envoi d'une lettre

. Réception d'une lettre

L'abonné peut faire d'autres demandes sans utiliser de NIOC comme l'ordre d'ouverture ou de fermeture de la ST et certains ordres de maintenance.

R-ST : Quand tout le traitement concernant ce NIOC est terminé, la ST positionne un compte rendu et rend le NIOC: c'est l'évènement fin d'entrée/sortie. L'abonné peut alors récupérer son bloc, sachant que sa demande a été prise en compte, et analyser le réponse de la ST;

S-ST : Le signal est le moyen , pour la ST, de transmettre à l'abonné des évènements qui indiquent quel genre de pli arrive du réseau. Ces évènements sont:

. Arrivée d'une demande distante d'ouverture de flot
. Arrivée d'un message STI
. Fermeture de flot (n° FL indiqué)
. Arrivée d'une lettre (n° FL indiqué)
. Arrivée d'un télégramme (n° FL indiqué)

9.3.3. - Spécifications internes :

La machine M:ST est divisée en deux partie (voir figure 40):

. La partie haute qui traite le protocole ST - ST (commande de flot).
. La partie basse qui traite le protocole Cigale (paquet).

Ces deux parties communiquent par deux files: l'une d'émission, l'autre d'arrivée en échangeant des buffers FC (buffers flow command) formés des commandes de flot auxquelles on ajoute un en-tête qui contient la longueur et le numéro de la ST distante. Dans le protocole ST - ST le flot est identifié par les numéros des deux portes qu'il joint, ce qui est insuffisant pour le distinguer des autres flots, il faut aussi connaitre la ST distante.

Les trois niveaux de la M:GP-Cyclades sont donc désynchronisés les uns par rapport aux autres. L'abonné émet en fonction des demandes du Correspondant (alimentation de la FU). La partie haute ne peut envoyer des lettres que si elle a les crédits nécessaires sur flot, comme pour la M:GPOC c'est ce

qui permet le contrôle de flux. Les émissions et réceptions de la partie basse dépendent de la cadence de la ligne.

Il faut se rappeler que le Frontal est un mini-calculateur. Les buffers utilisés dans les échanges sont souvent pris dans une mémoire libre et il faut les restituer le plus vite possible. Pour les arrivées la partie haute possède ses propres zones de stockage (sur disque), où elle peut notamment faire le réassemblage des lettres. Elle ne prévient l'abonné que lorsque la lettre est entière et contrôlée, toutes les lettres mal référencées ou incomplètes sont éliminées. Les échanges effectués par NIOC sont ainsi réduits à ceux absoluement nécessaires. Cela évite à l'abonné, après qu'il ait donné la main, de se mettre en réception et d'attendre un envoi de son interlocuteur, il ne fait sa demande, donc immobilise un buffer, que lorsqu'un pli correct est effectivement arrivé.

- Partie basse :

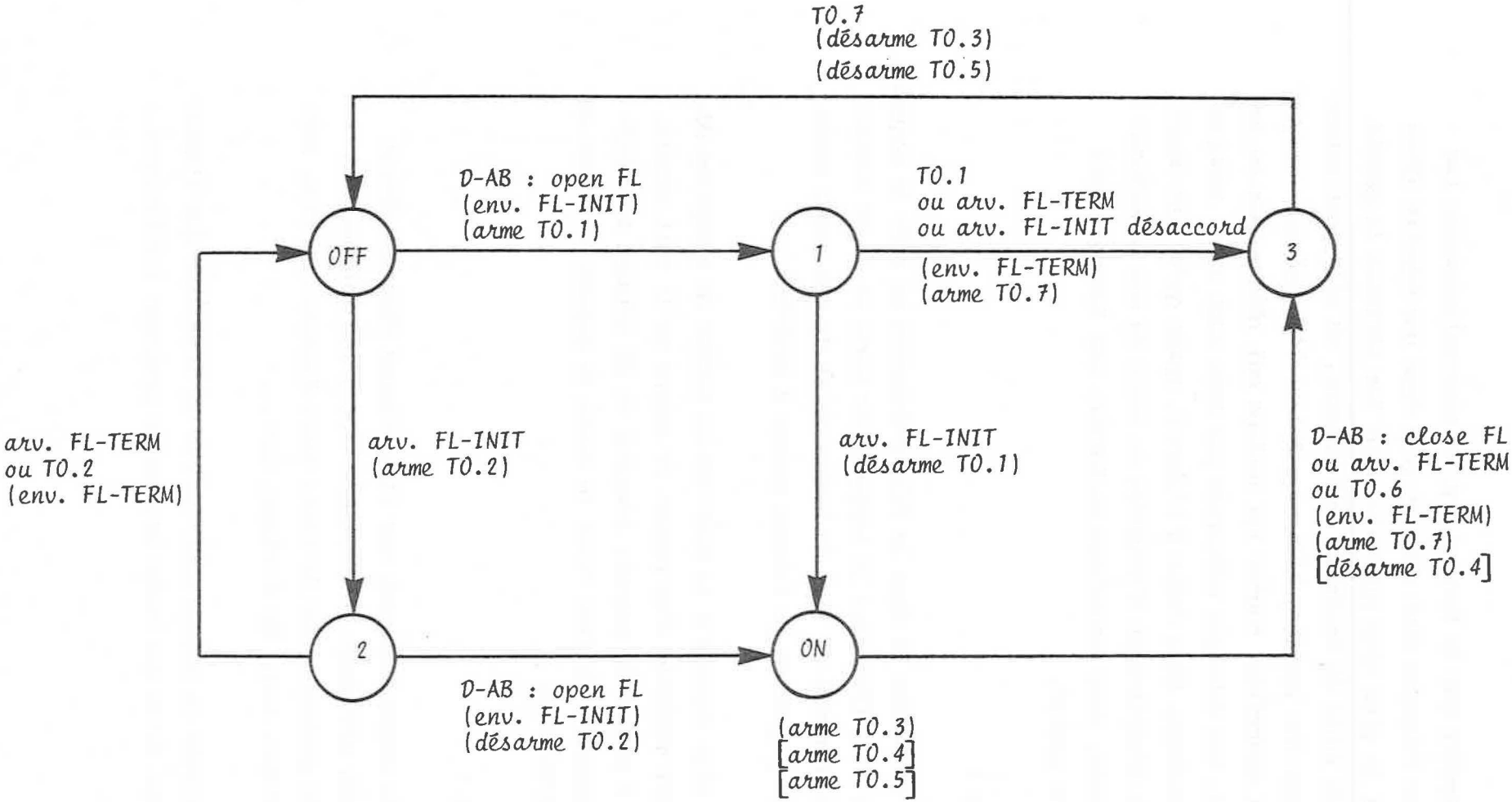
Elle récupère un buffer FC dans la file d'émission et crée le paquet Cigale: mise en forme de l'en-tête avec la longueur du texte et la ST destinatrice, elle extrait la commande de flot du buffer FC et la met comme texte du paquet. Elle envoie le paquet en le faisant passer à M:GT-TMM.

Pour recevoir elle fournit à la M:GT-TMM un buffer de réception de taille égale à la longueur maximale d'un paquet et attend qu'il soit rempli. Elle forme un buffer FC à partir du paquet: longueur et ST origine sont prises dans l'en-tête, la commande de flot étant le texte du paquet. Elle met ce buffer FC dans la file d'arrivée.

- Partie haute :

La ST travaille essentiellement sur flot. Chaque flot est décrit par un contexte créé à son ouverture. Il contient toutes les informations nécessaires pour gérer le protocole sur le flot: identification du flot, crédits, références des lettres, zones de stockage, etc ...

Les FL-INIT arrivés du réseau sont stockés en attendant que l'abonné leur réponde. Un réveil évite que cette attente se prolonge indéfiniment.



T0.1 attente réponse distante (état=1)
T0.2 attente réponse locale (état=2)
T0.3 émission cyclique d'ACK (état=on/3) pour entretenir les échanges sur le flot
T0.4 attente d'ACK (état=on)
T0.5 attente FR (état=on/3)
T0.6 attente pli sur FL (état=on)
T0.7 fermeture (état=3)

() : action à faire
[] : action éventuellement à faire

Figure 41 - AUTOMATE DU FLOT

Le flot fonctionne suivant l'automate de la figure 41.

Comme la M:GPOC la ST ne rend une réponse positive à une demande d'émission de lettre sur flot avec services additionnels que lorsque la lettre a été acquittée par la ST destinatrice.

9.4. - LA MACHINE ABONNE (M:AB) [CCR-7]

Elle s'occupe de toutes les demandes du Correspondant concernant le réseau Cyclades. Elle se compose de trois machines de même niveau, donc travaillant en parallèle, qui gèrent chacune un aspect particulier de l'accès à Cyclades et utilisent en commun la M:ST pour faire leurs entrées/sorties sur le réseau :

. La machine client Cyclades
. La machine serveur Cyclades
. Le service opérateur Cyclades

L'aiguillage vers une de ces trois machines va se faire pour les ordres du Correspondant suivant la SU et pour les réponses et signaux ST selon le flot.

9.4.1. - La machine client Cyclades :

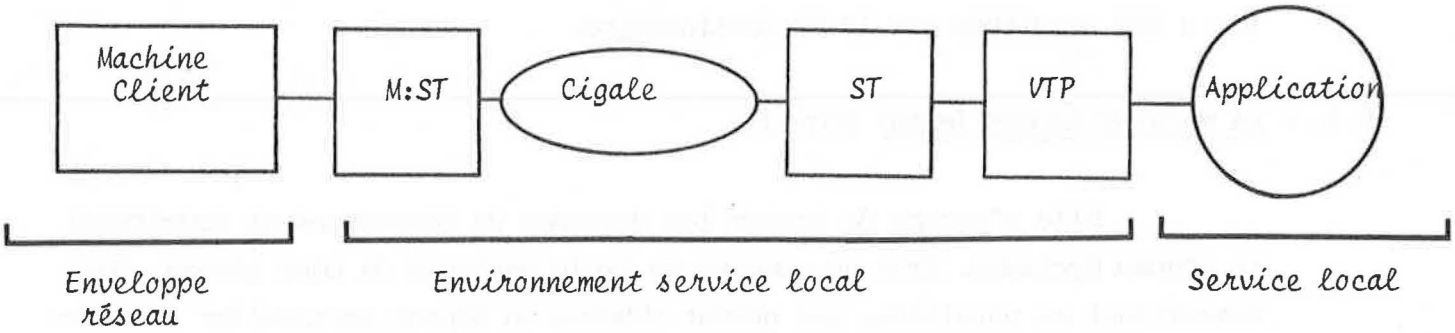
Elle permet l'accès aux services offerts par le réseau. Nous avons vu au Chapitre 5 que dans un premier temps nous avions espéré pouvoir en faire un simple traducteur entre LCI et VTP mais que cela s'était révélé impossible. Pour rester cohérent avec la logique du CCR nous considérons que les services Cyclades sont localisés en partie sur le réseau et en partie sur le Frontal. En reprenant la décomposition par niveau d'un service nous avons:

. Le service local qui est l'application offerte par un des hôtes de Cyclades.

. L'environnement service local comprend non seulement les parties du système d'exploitation de l'ordinateur hôte nécessaires à l'application, mais aussi toute la partie de Cyclades utilisée pour connecter le Frontal

à l'hôte. Cyclades est vu comme le "système d'exploitation" d'un "ordinateur" supportant toutes les applications offertes par le réseau.

. L'enveloppe réseau est la machine client.



Service Cyclades

Figure 42

Comme pour les services écrits sur le Frontal la voie logique se réduit donc à la file d'attente de la FU, bien que dans ce cas la FU existe. Mais il n'y a pas forcément de relation entre voie logique et FU. La FU est le moyen "physique" d'accèder à une entité matérielle extérieure au Frontal. La voie logique joint des entités purement logiques.

La machine client est l'interlocuteur du Correspondant service pour les services Cyclades, elle va donc comprendre et parler le LCI.

Il faut d'abord établir la correspondance entre adressage CCR et adressage Cyclades. L'adresse réseau d'un service est un n° ST et n° PT, pour le CCR c'est un numéro de SU. Les coordonnées Cyclades sont compliquées par le fait que le service peut accepter des appareils supplémentaires. La description Cyclades du service donne sa nature (les appareils qu'il accepte) et les n° PT correspondantes aux sorties (ou entrées) sur ces appareils. Toutes ces informations sont contenues dans un contexte statique de la SU du service dans lequel on marque aussi si la voie logique est ouverte ou non.

Pour chaque liaison demandée par le Correspondant service la machine client crée un contexte dynamique qui sert à la fois à décrire cette liaison côté CCR et faire la correspondance avec Cyclades.

Pour plus de compréhension regardons plus en détail l'interprétation d'un CNEX venant du Correspondant service. La machine client crée le contexte liaison dans lequel elle range l'identification LCI de la liaison: SU.utilisateur, SU.service, Identificateur de l'utilisateur. Grâce au contexte SU elle trouve l'adresse Cyclades du service appelé et demande à la ST d'ouvrir un flot entre la porte distante du service et une porte locale qu'elle attribue à la liaison (choisie de façon à distinguer les flots ouverts avec le même service). Si elle essuie un refus après plusieurs tentatives elle envoie un DCNX au Correspondant service par l'intermédiaire du buffer de réception que celui-ci a fourni puisqu'il considère que c'est le service qui a la main. Si le flot est ouvert elle range dans le contexte liaison le numéro de flot rendu par la ST. La correspondance liaison - flot est établi et les échanges peuvent commencer. Les commandes EXCH sont facilement interprétables par des envois de lettre.

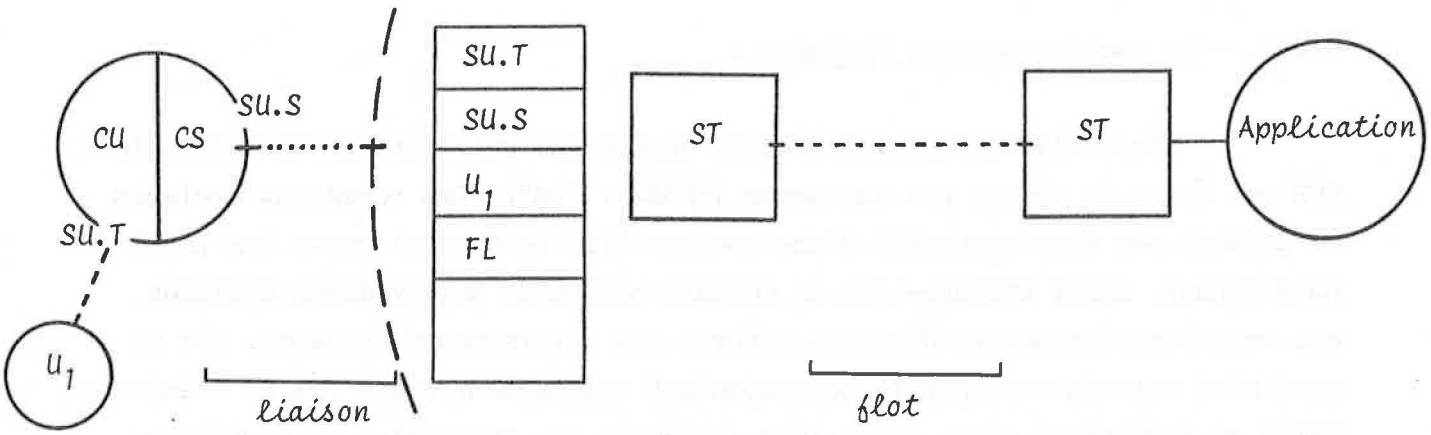


Figure 43

Lorsque l'utilisateur a demandé l'attachement d'autres appareils le CNEX indique que plusieurs liaisons sont ouvertes. Pour chacune d'elle la machine client va demander l'établissement d'un flot avec la porte correspondante du service Cyclades (porte service ou porte appareil) et créer un contexte liaison, avec le moyen de retrouver toutes les liaisons impliquées par le CNEX car c'est l'ensemble qui forme l'accès au service.

La rupture de la liaison utilisateur - service, qu'elle vienne du Correspondant service par un DCNX ou du réseau par la fermeture du flot correspondant, entraine la fermeture de toutes les liaisons appareils et de leurs flots associés ainsi que l'envoi d'un DCNX au Correspondant service.

Les évènements LCI sont facilement interprétables: ATTN en télégramme ATTENTION et PRTY en lettre PRIORITY. Inversement la machine client ne reçoit pas d'évènement VTP du réseau.

Comme dans la M:GPAS les IOCB de réception sont stockés dans le contexte SU, puisque c'est à elle qu'ils sont dédiés et non aux liaisons. Lorsque l'envoi d'une commande LCI au Correspondant service est nécessaire sur une liaison concernant cette SU, la machine client utilise un de ces IOCB stocké. Il n'y a pas de risque de prendre trop de buffer de réception puisque, interprètant le LCI, le client sait de quel côté est la main. L'annulation d'une réception se fait en rendant un de ces IOCB.

9.4.2. - La machine serveur Cyclades :

C'est elle qui permet l'accès du CCR aux utilisateurs Cyclades. le CCR est connu du réseau par une porte ('E3E2 : "TS"). Les terminaux Cyclades ne peuvent pas être connectés d'une manière fixe au Frontal comme les terminaux locaux, aussi déclare-t'on un certain nombre de SU.terminaux Cyclades qui vont être les points d'entrée offerts aux utilisateurs Cyclades. Ces SU sont bien entendu vues par le Correspondant utilisateur comme des SU standards. Elles se comportent comme des prises physiques sur lesquelles on peut brancher successivement plusieurs terminaux. C'est la machine serveur qui est chargée du branchement. Elle ne connait pas ces SU de manière statique mais uniquement lorsque le Superviseur les ouvre, exactement comme la M:GPAT vis à vis des voies logiques qu'elle gère.

A l'arrivée d'une ouverture de flot sur la porte "TS" le serveur cherche une SU.terminal Cyclades libre. S'il en trouve une il fait le branchement: il crée un contexte SU dans lequel il marque l'association SU - flot. Ce contexte n'a pas besoin d'exister tant qu'aucun branchement n'a été fait. Entre l'ouverture par le Superviseur et une demande de connexion à la porte

"TS" il suffit que le mécanisme d'accès aux SU indique que la SU existe.

La voie logique reliant un terminal Cyclades au Correspondant utilisateur recouvre l'ensemble des moyens nécessaires à la connexion selon le schéma complet:

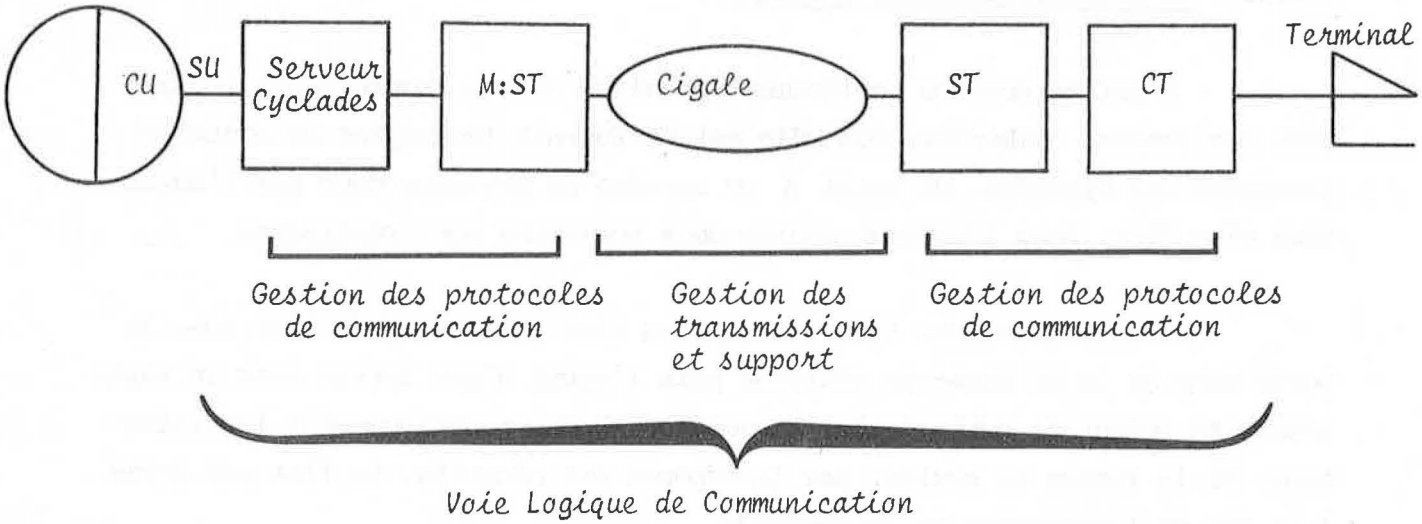


Figure 44

Pour montrer cette voie logique comme une voie standard le serveur va se servir du VTP comme la M:GPAT le fait du Protocole Appareil Terminal.

L'utilisateur Cyclades, comme tout utilisateur, envoie du LCE mais celui-ci arrive "enrobé" de VTP pour le contrôle de dialogue. Le serveur enlève le VTP pour ne délivrer au Correspondant utilisateur que le LCE. Inversement il doit envoyer vers l'utilisateur les réponses du Correspondant utilisateur accompagné du VTP traduisant le contrôle de dialogue implicite existant dans les échanges entre Correspondant et usagers.

Notamment les messages VTP PRIORITY, ATTENTION 1 et ATTENTION 2 vont être respectivement compris comme les évènements LCE (AC), (E1) et (E2) et répercutés comme tels chez le Correspondant utilisateur avec blocage de la voie.

Remarque : Pour simplifier leur tâche client et serveur demandent systématique-

ment le mode transparent appareil sur les flots, ce mode leur évitant d'avoir à traduire les fonctions d'adressage VTP. Mais si leur interlocuteur refuse ils ne ferment pas le flot pour autant.

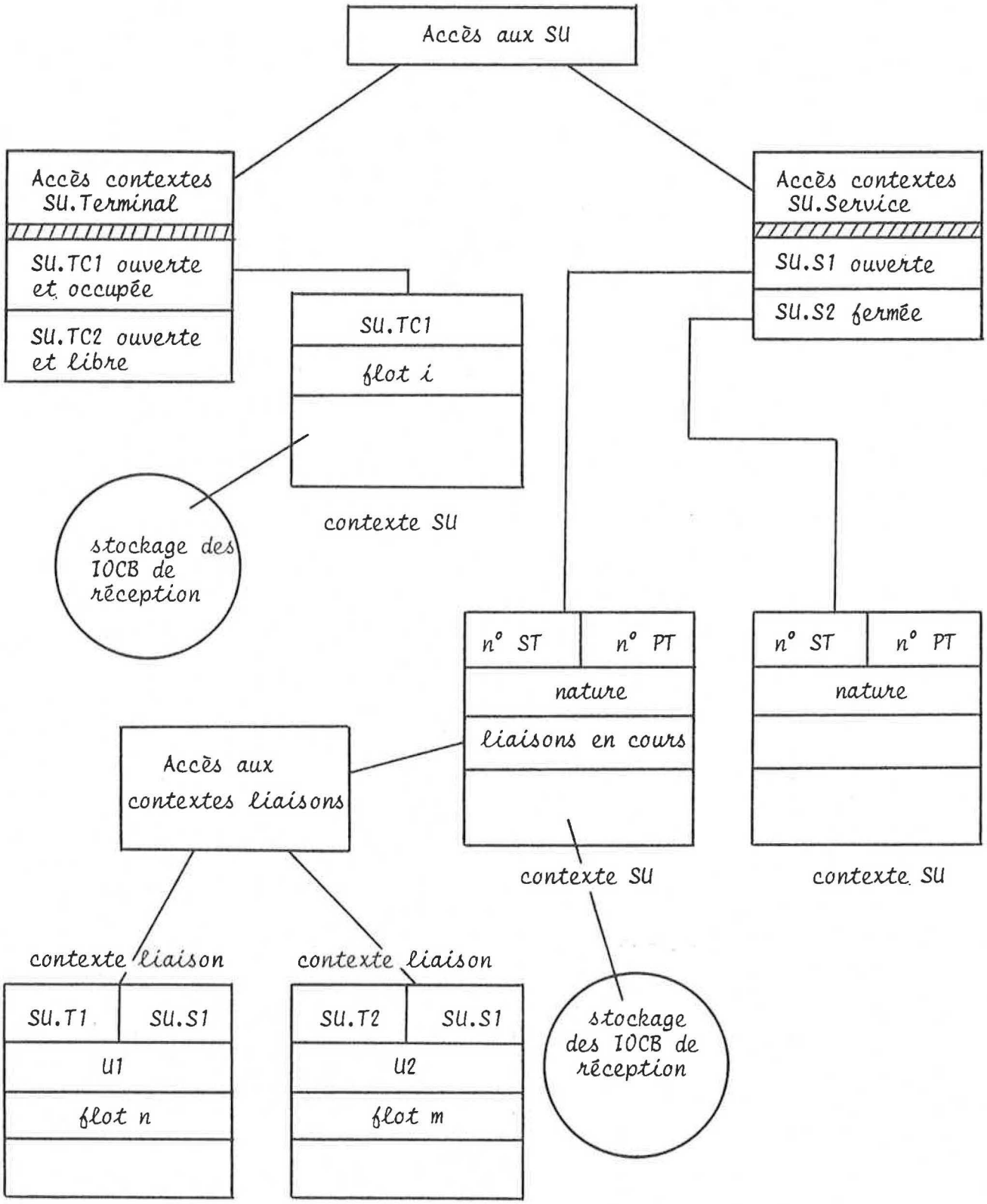
9.4.3. - Le service opérateur Cyclades :

C'est un service entièrement localisé sur le Frontal et qui possède ses propres commandes. Son rôle est de fournir les moyens de contrôle possibles sur Cyclades. Un appel à ce service ne provoque donc pas l'ouverture d'un flot. Pour l'instant seules deux commandes sont réalisées:

- ECHO déclenche l'ouverture d'un flot en service de base avec la porte écho de la ST distante désignée puis l'envoi d'une lettre avec un texte connu. Au retour de cette lettre le service répond positivement à l'utilisateur. Si la lettre ne revient pas la réponse est négative. Le flot est fermé à la fin de l'exécution de la commande.

- CIGA est l'envoi d'un message à l'opérateur d'un noeud. Elle est traduite par une demande d'émission de message STI.

D'autres possibilités seront offertes par ce service quand celles ci existeront sur Cyclades. Par exemple des diagnostics sur l'état du réseau.



Exemple : nous avons quatre SU, deux SU.Terminal et deux SU.Service
SU.TC1 est ouverte et un terminal Cyclades s'est connecté
SU.TC2 est ouverte et libre
SU.S1 est ouverte et possède deux liaisons (utilisateurs U1 et U2)
SU.S2 est fermée

Figure 45

CHAPITRE 10

CONCLUSION

10.1. - LIMITES DE L'ETUDE

Nous avons décrit ,d'un point de vue fonctionnel, un centre de calcul réparti sur plusieurs ordinateurs interconnectés entre eux, en nous fixant au départ un objectif précis: fournir à des utilisateurs divers un moyen simple de se servir des possibilités offertes par les matériels informatiques.

Tout au long de cette étude est aussi apparue une certaine méthodologie de travail: limiter les interventions sur les matériels et systèmes mis en jeux, utiliser au maximum des techniques existantes et développer de nouveaux outils uniquement en fonction de nos besoins.

Mais après la définition du CCR nous voulons maintenant en faire une réalisation effective. D'abord parce que le meilleur moyen de valider nos idées est de les mettre en pratique, ensuite pour montrer que le développement du centre de calcul (adjonction de service ou de matériel) est possible sans trop perturber l'exploitation, ce qui est une des contraintes fixées au Chapitre 2.

Que nous le voulions ou non cette volonté de ne pas se limiter à une "étude papier" a influencé nos orientations et nos définitions. Et il ne fait nul doute qu'au fur et à mesure que nous avancerons dans le développement réel du CCR certains de nos choix "théoriques" qui semblent parfaits aujourd'hui seront remis en question.

Il faut donc bien considérer que plusieurs des descriptions d'entités faites dans les chapitres précédents sont surtout des buts à atteindre et qu'elles ne connaissent à l'heure actuelle qu'un début d'écriture ou de mise au point.

Néanmoins nous espérons arriver à la réalisation d'une maquette basée sur les principes énoncés et remplissant toutes les fonctions demandées. Mais ceci n'est qu'une étape. A partir de cette maquette, qui consti-

tuera une base de départ solide, des développements futurs sont d'ores et déjà prévisibles. Puisque nous aurons montré que nos idées peuvent aboutir à un produit valable il deviendra intéressant de les pousser plus en avant.

Sur quelques points précis nous pouvons indiquer des améliorations envisageables.

10.2. - DEVELOPPEMENT DU LCE

10.2.1. - Commandes imbriquées :

Les requêtes LCE ont une forme très simple afin d'être facilement utilisables et de permettre une extension aisée du langage. Mais cette simplicité tend à donner à chaque requête une fonction assez primitive, ce qui oblige l'usager, pour un travail complexe, à entrer de nombreuses lignes.

Par exemple un utilisateur désire faire compiler le fichier source algol SRCE dans le fichier objet OBJALG, puis lier ce dernier avec les modules OBJ1 et OBJ2 et enfin faire exécuter le programme ainsi obtenu. Il va donc appeler successivement les services ALGOL, LINK et EXEC, ce qui se traduit par la série de commandes:

```
ALGOL   SRCE  OBJALG
LINK    OBJALG  OBJ1  OBJ2
EXEC    OBJALG
```

L'écriture est quand même simplifiée par le fait que l'utilisateur n'est pas obligé de taper la fonction "fin d'appel" (DS) après chaque appel. Pourtant cette suite de requête peut être écrite en une seule ligne si l'on introduit dans la définition du LCE la notion de commandes imbriquées. Des délimiteurs spéciaux "[" et "]" par exemple, permettent de distinguer les commandes entre elles.

Il est alors possible d'écrire la séquence d'instructions précédentes:

EXEC [LINK [ALGOL SRCE OBJALG] OBJ1 OBJ2]

Le Correspondant utilisateur va d'abord interpréter la commande la plus intérieure, l'exécuter et la remplacer dans la ligne par son résultat, et ainsi de suite jusqu'à ce que toute la ligne ait été analysée [Whitney, LAN-7][Charlet, LAN-8].

Dans notre cas cela donne:

. exécution de ALGOL SRCE OBJALG qui est remplacée par OBJALG
. la ligne devient EXEC [LINK OBJALG OBJ1 OBJ2]
. exécution de LINK OBJALG OBJ1 OBJ2 qui est remplacée par OBJALG
. la ligne devient EXEC OBJALG
. exécution de EXEC OBJALG

Nous retrouvons donc bien les appels successifs aux trois services nécessaires. Ce mécanisme a le désavantage d'obliger le Correspondant utilisateur à garder stockée la ligne de l'utilisateur tant que celle-ci n'est pas entièrement analysée.

10.2.2. - Instructions procédures :

Au cours du dialogue externe si l'utilisateur appelle un service qui, pour une raison ou pour une autre, n'est pas accessible le Correspondant utilisateur le signale immédiatement. Mais si cet appel se trouve dans une procédure cataloguée et que les lignes suivantes sont des <Ligne E> destinées à l'application le Correspondant utilisateur va systématiquement déclarer une erreur chaque fois qu'il tentera d'en interpréter une puisqu'aucune liaison n'aura été établi. L'usager n'a plus alors que la ressource d'envoyer (AC) pour stopper l'exécution de la procédure.

Il semble donc intéressant de créer une instruction conditionnelle spéciale pour les procédures définie par:

<Instruction Cond> ::= /SI <Cond> ALORS {<Ligne proc>}* /FSI

<Cond> ::= SC | EC

SC (succès du call) est vrai si l'appel d'un service a réussi ou faux sinon et inversement pour EC (échec du call).

Une procédure peut ainsi avoir la forme:

```
?CS  SERVICE1
/SI  SC  ALORS
      .
      .
     suite de <Ligne E>
      .
      .
/FSI
?CS  SERVICE2
/SI  SC  ALORS
      .
      .
     suite de <Ligne E>
      .
      .
/FSI
```

Ce qui protège les <Ligne E> d'une tentative d'interprétation alors que l'appel au service qu'elles concernent a échoué.

Il est aussi possible qu'après l'échec d'un appel l'exécution des commandes suivantes n'offre plus d'intérêt. Il faut alors introduire des points de sortie de la procédure, exactement comme si l'utilisateur envoyait (AC), sous forme d'une instruction /EXIT.

En reprenant l'exemple du paragraphe 10.2.1 où les appels dépendent de ceux qui les précèdent on peut écrire:

```
?ALGOL  SRCE  OBJALG
/SI  SC  ALORS
?LINK   OBJALG  OBJ1  OBJ2
/SI  SC  ALORS
?EXEC   OBJALG
/FSI
/FSI
```

ou

```
?ALGOL  SRCE  OBJALG
/SI  EC  ALORS
/EXIT
/FSI
?LINK   OBJALG  OBJ1  OBJ2
/SI  EC  ALORS
/EXIT
/FSI
?EXEC   OBJALG
```

10.3. - ACCES AU RESEAU CYCLADES

Le développement du CCR entraine l'introduction sur le Frontal de nouvelles données et même de nouvelles FU dans le cas de l'adjonction d'ordinateurs. Or ce Frontal est un mini-calculateur aux ressources limitées, il va donc apparaitre nécessaire à un instant donné de le décharger d'une partie de son travail.

Un moyen d'éviter la croissance du logiciel Frontal est d'essayer de ne lui faire gérer que des protocoles de communication standards, c'est à dire de se servir le plus possible du Protocole Ordinateur Connecté avec les autres matériels du CCR.

Les protocoles Cyclades sont parmi les plus complexes, donc les plus coûteux, à utiliser. L'idée est de déporter leur gestion sur un autre mini-ordinateur qui devient le Frontal Cyclades du Frontal CCR.

La logique ne change pas, l'ensemble Frontal Cyclades et réseau Cyclades est vu comme un seul ordinateur, le protocole de communication entre ce dernier et le Frontal CCR étant le Protocole Ordinateur Connecté. C'est à dire qu'on revient au cas de connexion exposé au Chapitre 8 : par rapport à la figure 36 la machine client correspond à l'enveloppe réseau des services et la machine serveur est la gestion des terminaux locaux à "l'ordinateur Cyclades".

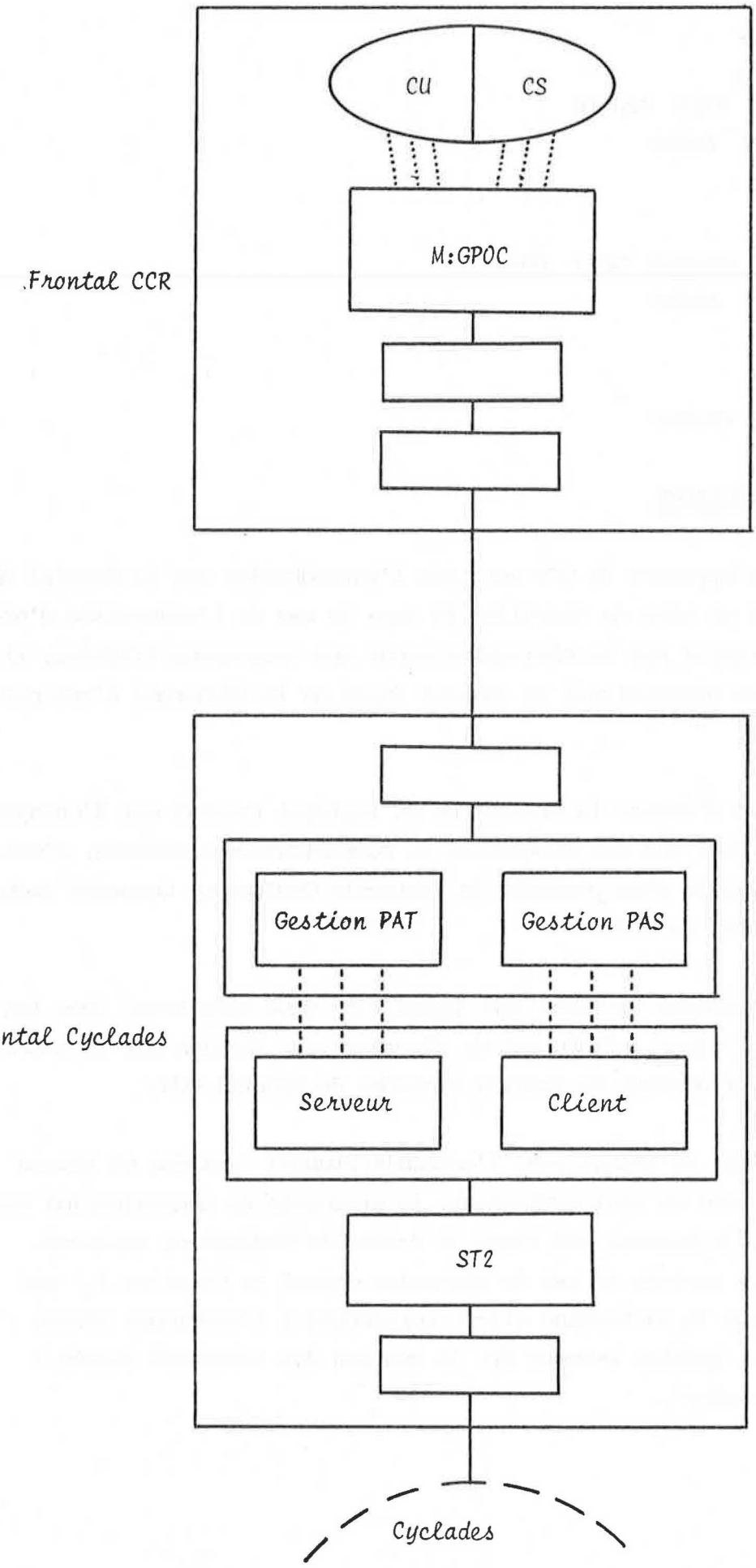

CU
CS
M:GPOC
Frontal CCR
Gestion PAT
Gestion PAS
Frontal Cyclades
Serveur
Client
ST2
Cyclades

Figure 46

Bien évidemment une telle solution ralentit les échanges avec le réseau externe, mais elle est plus souple lorsque des modifications doivent être apportées à la gestion des protocoles en fonction de l'évolution de Cyclades. Seule la bonne marche du Frontal Cyclades est gênée par la mise à jour du logiciel, le reste du CCR peut continuer à fonctionner normalement.

BIBLIOGRAPHIE

SYSTEMES ET RESEAUX : GENERALITES

GEN-1 POUZIN L.
"Architectures et variétés de réseaux"
Document Cyclades RES 500 (Aout 73)

GEN-2 POUZIN L.
"Network architecture and components"
Document Cyclades SCH 516 (Aout 73)

GEN-3 RODRIGUEZ F.J. , CHALLINE J.F.
"Structuration et communications dans un système d'exploitation"
Compte rendu des journées d'étude sur les structures résultant d'un groupement de processeurs , St Pierre de Chartreuse (Novembre 73)
pp. 26-51

GEN-4 KIMBLETON S. , MANDELL R.
"A perspective on network operating systems"
AFIPS national computer conference proceedings , vol. 45 (Juin 76)
pp. 551-560

GEN-5 MAC QUILLAN J.M.
"Strategies for implementation of Multi-Host computer networks"
Computer Communication Review , vol. 6 , n° 4 (Octobre 76) pp. 19-24

GEN-6 CHAMBON J.F. , LE BIHAN B.
"Architecture d'un Frontal en environnement télé-informatique (application au réseau Cyclades)"
Thèse de Docteur Ingénieur , Ecole des Mines de St Etienne (Octobre 76)

GEN-7 CHAMBON J.F. , LE BIHAN B.
"Architecture d'un calculateur Frontal en environnement télé-informatique"
Congrès AFCET , Gif-sur-Yvette (Novembre 76) pp. 881-890

GEN-8 WATSON R.W.
"User interface design issues for a large interactive system"
AFIPS national computer conference proceedings , vol. 45 (Juin 76)
pp. 357-364

GEN-9 LAFORGUE P.
"Construction de sous-système utilisant une machine abstraite. Réalisation autour du noyau GEMAU"
Thèse de Docteur Ingénieur , Grenoble I (Février 75)

GEN-10 GUIBOUD RIBAUD S.
"Mécanisme d'adressage et de protection dans les systèmes informatiques. Application au noyau GEMAU"
Thèse de Docteur d'état , Grenoble I (Juin 75)

GEN-11 GOUDA M.G. , MANNING E.G.
"Toward modular hierarchical structures for protocols in computer network"
Digest of papers , Compcon 76 (Septembre 76) pp. 246-250

GEN-12 POUZIN L.
"Network Protocols"
Document Cyclades SCH 517 (Septembre 73)

LANGAGES DE COMMANDE

LAN-1 RAYMOND J.
"NJCL un langage de commande pour réseau d'ordinateurs"
Contrat CRI , Grenoble I (Juillet 73)

LAN-2 DU MASLE J.
"An evaluation of the LE/1 network command langage designed for the SOC network"
in Unger ed , IFIP Working conference on command languages , Suède (Juillet 74) pp. 319-326

LAN-3 SERGEANT G. , FARZA M.
"Machine interprétative pour la mise en oeuvre d'un langage de commande sur le réseau Cyclades"
Thèse de Docteur Ingénieur , Toulouse Paul-Sabatier (Octobre 74)

LAN-4 CHUPIN J.C.
"Command languages and heterogeneous networks"
in Unger ed , IFIP Working conference on command languages , Suède (Juillet 74) pp. 311-318

LAN-5 GOYER P. , DU MASLES J.
"Somme basic notions for computer network command languages"
in Unger ed , IFIP Working conference on command languages , Suède (Juillet 74) pp. 327-334

LAN-6 NAGEL H.H.
"On network languages"
in IRIA ed , 1st european workshop , computer networks , Arles (Avril, Mai 73) pp. 365-378

LAN-7 WHITNEY G.E. , MOOERS C.N.
"A formal definition of the scan algorithm of the TRAC T.64 language"
(Octobre 69)

LAN-8 CHARLET J.
"Système GEMAU - Le langage de commande"
Document GEMAU (Juin 73)

RESEAUX

RES-1 SOC Project - Report 1
Project departement , IBM France (Juin 71)

RES-2 DE WOUTERS H.
"How can a user submit execute jobs by a network and how could he follow their execution"
in IRIA ed , 1st european workshop , computer networks , Arles (Avril, Mai 73) pp. 113-136

RES-3 MANNING E. , HOWARD R. , O'DONNELL C. , PAMMETT K. , CHANG E.
"A UNIX-based local processor and network access machine"
Computer Networks , vol. 1 , n° 2 (Septembre 76) pp. 139-142

RES-4 HEBENSTREIT J.
"Projet M2 - Conception et réalisation d'un système multi-mini-ordinateur"
Congrès AFCET , Gif-sur-Yvette (Novembre 76) pp. 725-731

RES-5 BEAUFILS R. , BACKMANN S. , CABANEL J.P. , LAGASSE J.P.
"ARAMIS - Conception d'un réseau de mini-ordinateurs pour l'accès à des services spécialisés"
Contrat SESORI n° 74-98 , Lot n° 1

RES-6 FRANCHI P.
"Distribution of functions and control in RPCNET"
in IEEE Computer Society ed , 3rd annual symposium on computer architecture (Janvier 76) pp. 130-135

RES-7 WEBER S.
"Concentrateur Cyclades - Introduction aux concepts et à l'usage"
Support cours formation Cyclades (1977)

RES-8 ZIMMERMANN H.
"Proposal for a virtual terminal protocol (VTP)"
Document Cyclades TER 533.1 (Juillet 76)

RES-9 QUINT V.
"Complément de spécifications pour l'utilisation du Protocol Appareil Virtuel"
Document Cyclades TER 540.1 (Mai 77)

RES-10 ZIMMERMANN H. , ELIE M.
"Transport Protocol - Standard end-to-end protocol for heterogeneous computer networks"
Document Cyclades SCH 519.2 (Mai 75)

RES-11 POUZIN L.
"Presentation and major design of the Cyclades computer network"
Document Cyclades SCH 511 (Avril 73)

RES-12 GRANGE J.L. , POUZIN L.
"Cigale, la machine de commutation de paquets du réseau Cyclades"
Document Cyclades MIT 536 (Mai 73)

CENTRE DE CALCUL REPARTI

CCR-1 CHAMBON J.F. , GUIBOUD RIBAUD S. , LE BIHAN B. , TOSAN Y.
"Intéret et faisabilité d'un centre de calcul fondé sur un réseau de mini-ordinateurs"
Note interne , EMSE (Octobre 76)

CCR-2 CART M.
"Développement d'un logiciel de Frontal pour un centre de calcul réparti"
Rapport DEA , EMSE (Octobre 77)

CCR-3 ANGELIDES J. , CART M. , CHAMBON J.F. , TOSAN Y.
"Architecture globale du Frontal - Premières définitions"
Note interne , EMSE (Novembre 76)

CCR-4 ANGELIDES J. , CART M. , CHAMBON J.F.
"Spécifications externes et internes de la machine gérant les demandes d'entrées/sorties M:DM"
Note interne , EMSE (Avril 77)

CCR-5 ANGELIDES J. , CART M. , CHAMBON J.F.
"Spécifications externes de maintenance de la machine M:DM"
Note interne , EMSE (Mai 77)

CCR-6 TOSAN Y.
"La machine M:ST - Station de Transport du réseau Cyclades (ST.2)"
Note interne , EMSE (Juin 77)

CCR-7 TOURNOIS M.
"Spécifications de la machine M:GP Cyclades"
Note interne , INSA Rennes (Mai 77)

ANNEXES

ANNEXE A : Fonctions du LCE

- Liste des mots clés :

AC	AC
E1	E1
E2	E2
SC	Suppression de caractère
SL	Suppression de ligne
LL	Longueur de ligne
LP	Imprimante
LC	Lecteur
FI	Fichier
CL	Classe

- Liste des fonctions :

CS[ERV] Appel d'un service
Paramètre: nom du service , LP , LC , FI

DS[ERV] Fin d'appel d'un service

DA[TE] Demande ou changement de la date
Paramètre: date

HE[LP] Provoque l'impression de la liste de toutes les fonctions et de tous les services accessibles

ET[AT] Donne un certain nombre de renseignements contenus dans l'environnement utilisateur

MS[SAGE] Envoi d'un message à un autre utilisateur
Paramètre: identificateur de l'utilisateur destinataire , suite de caractères constituant le message

MO[DIF] Modification des caractères de contrôle
Paramètre: AC , E1 , E2 , SC , SL , LL
N'est accessible que d'un terminal du Frontal

IU[TIL] Introduction d'un utilisateur
Paramètre: Identificateur de l'utilisateur , mot de passe , CL

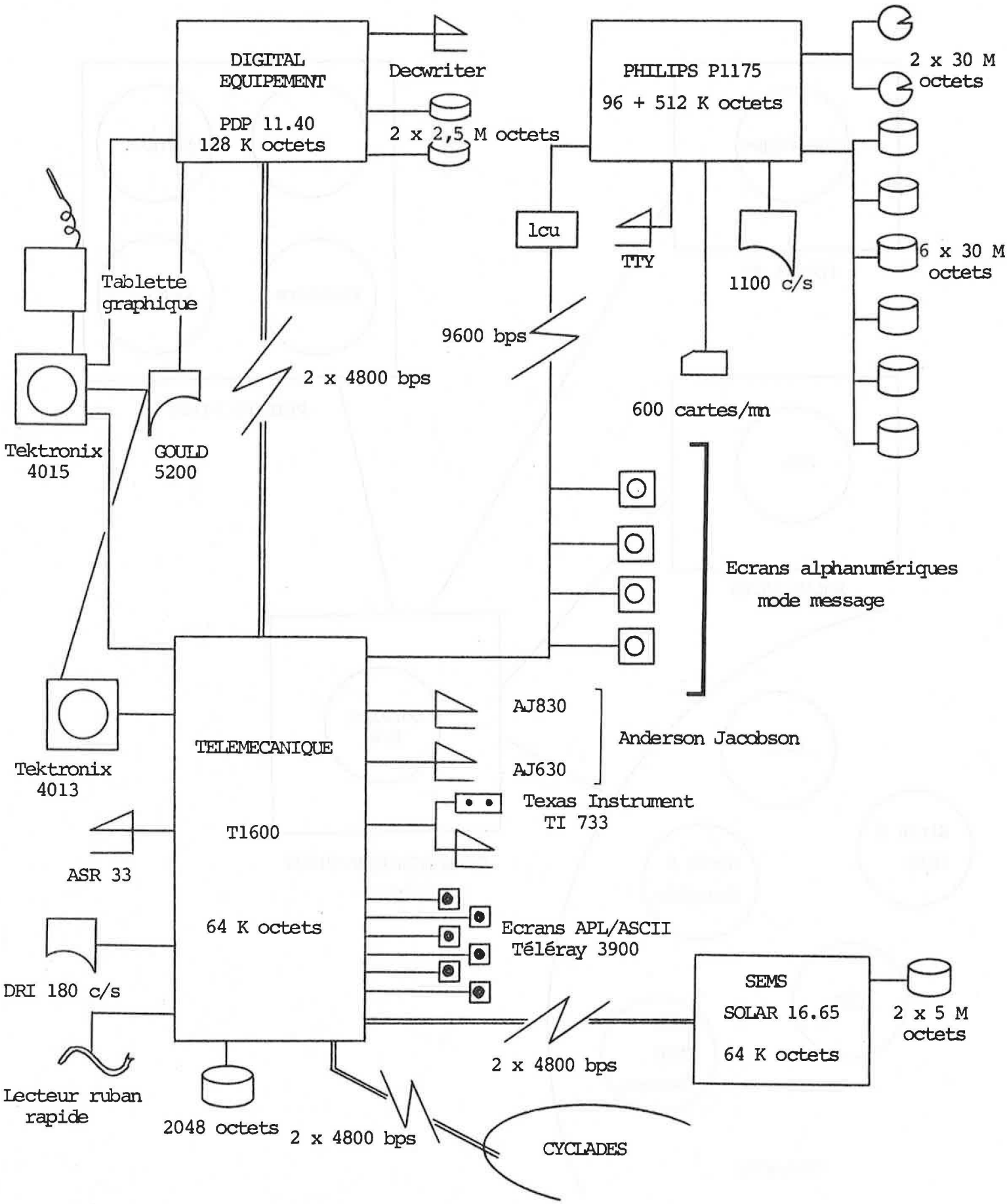
SU[TIL] Suppression d'un utilisateur
Paramètre: Identificateur de l'utilisateur à supprimer

MU[TIL] Modification des caractèristiques d'un utilisateur
Paramètre: Identificateur de l'utilisateur , CL

KU[TIL] Tuer un utilisateur (Logout forcé)
Paramètre: Identificateur de l'utilisateur

KA[NY] Déconnexion forcée de tous les utilisateurs ayant appelé un service donné
Paramètre: nom du service

ANNEXE B : Moyens matériels du centre de calcul



ANNEXE C : Exemple de répartition des services