R.I.R.O.
(1re année, No 6, 1967, p. 3-25)

LE SYSTEME HALIOTIS (1)

par MM. DERVILLE (2), KAISER (3), PEIROTES (4), TELLIER (2)

Haliotis — (du grec Halios, marin) : ceux des mers tropicales sont recherchés pour leur coquille ; ceux des régions tempérées, appelés vulgairement « oreille de mer » sont comestibles.

Résumé. — Le système haliotis est destiné à assurer le traitement simultané d'un ensemble de fonctions, réunies en activités. Deux groupes de deux calculateurs, associés à des disques et à un pupitre de commande centrale forment l'outil de traitement dont la mise en œuvre est décrite. Quatre niveaux de hiérarchie régissent les programmes :

— le superviseur gère les activités du système,
— le moniteur gère les travaux d'une activité,
— le travail assure l'exécution d'une fonction,
— le module réalise une partie du travail.

Le principe et le fonctionnement du moniteur, l'organisation du programme travail, les tableaux de variables et les entrées et sorties sont décrits en détail.

1. — UTILISATION DES CALCULATEURS A BORD DES NAVIRES DE GUERRE

1.1. — L'amélioration très sensible du service rendu par les équipements traditionnels lorsqu'ils sont associés à des calculateurs, a contribué très fortement au développement de ceux-ci à bord des navires de guerre. On rencontre désormais des « systèmes informatiques » où l'information est gérée par un ou plusieurs calculateurs ou machines informatiques.

(1) Réalisé au Centre de Calcul Cœlacanthe de la Direction Technique des Constructions Navales.

(2) Ingénieur à la C.I.I.

(3) Ingénieur principal du Génie Maritime.

(4) Ingénieur du Génie Maritime.

Les matériels mis en jeu par ces systèmes comprennent :

— des calculateurs numériques ou analogiques,

— des dispositifs de mise en route et de surveillance de ces calculateurs,

— des organes de mémorisation de l'information,

— des appareils de liaison entre les calculateurs et les autres équipements :

* unités d'entrée-sortie et canaux d'accès,
* convertisseurs analogiques-digitaux et capteurs,
* convertisseurs digitaux-analogiques,
* des organes de liaison vers d'autres systèmes ;

— des moyens de communication entre l'homme et le système composé du calculateur et de ses programmes (ces moyens sont le support du langage de communication exploité par l'homme d'un côté, par le calculateur et ses programmes de l'autre) :

* lecteurs de divers types,
* organes de présentation sur support permanent : imprimantes, tables traçantes, perforatrices,
* consoles et pupitres de visualisation ;

— des équipements traditionnels dont la structure a souvent été modifiée pour mieux utiliser les possibilités du système : asservissements numériques, moteurs incrémentaux. La séparation entre le domaine de l'information et celui de l'action est davantage précisée que dans les équipements anciens.

Selon le degré d'intimité entre le calculateur et les équipements périphériques, on est amené à distinguer deux types d'emploi de celui-là :

— calculateur de commande, il est indispensable aux équipements pour remplir leur fonction,

— calculateur de centralisation des données, il collecte les informations en provenance des équipements et en extrait les éléments intéressant les utilisateurs du système.

La distinction entre ces deux emplois n'est jamais très franche et l'un et l'autre se rencontrent dans les systèmes usuels.

1.2. — Plusieurs cas d'utilisation sont possibles :

1.2.1. — Un calculateur est employé dans le cadre d'une fonction. Il est choisi pour satisfaire les besoins de celle-ci seulement. Cette solution peut se justifier si la fonction doit être assurée en permanence ou si les traitements doivent être extrêmement rapides. Toutefois, lorsque la permanence n'est pas demandée ou si une partie seulement des opérateurs du calculateur est employée, le rendement est mauvais et cette solution coûteuse. D'autre part, on constate qu'elle laisse fleurir des calculateurs qui diffèrent d'une fonction à l'autre par leurs technologies, leurs fiabilités et leurs modalités d'emploi.

1.2.2. — Un calculateur est géré en multiprogrammation pour servir plusieurs fonctions. Le rendement est meilleur que précédemment ; une synthèse des informations est possible. Cependant, cette solution est dangereuse, car toutes les fonctions ont un point commun qui doit être extrêmement fiable et insensible aux perturbations provenant de chaque fonction. Ce danger est écarté en doublant le calculateur, en créant donc un centre de calcul qui fournit des services à tout le bord. La sécurité de fonctionnement est augmentée, mais le rendement est amoindri car certains programmes prioritaires doivent exister en double, un dans chaque calculateur et les liaisons entre les deux calculateurs ne se font pas aisément.

1.2.3. — Un centre de traitement bâti sur un système multiprocesseur fournit des services multiples tout en assurant une sécurité excellente grâce aux possibilités de dégradation lente (les blocs de mémoires sont banalisés ; ils sont reliés à plusieurs blocs de calcul et à plusieurs unités d'échange avec les périphériques). C'est la solution d'avenir. On constate qu'elle est préconisée par tous les architectes de systèmes importants nécessitant une grande sécurité et fonctionnant en temps réel.

1.3. — Il convient de rappeler que la technologie des calculateurs militaires les distingue des autres.

En effet, ils sont appelés à fonctionner en « temps réel » et en permanence. On ne peut tolérer un taux de panne trop important. Mais c'est là aussi le lot de certains calculateurs du type industriel.

Ensuite l'environnement est sévère. Ils doivent résister aux chocs, aux vibrations, aux variations de température, être étanches au sable, à l'air salin, à l'humidité, à l'huile, rester insensibles aux rayonnements des émetteurs radio et radar.

(A titre d'exemple. Récemment, en cours d'installation, un calculateur embarqué est tombé d'une hauteur de 2 mètres. Mis sous tension, il a fonctionné normalement.)

2. — STRUCTURE D'UN SYSTEME INFORMATIQUE

2.1. — L'emploi d'un centre de traitement des données concerne les gros systèmes et se justifie par le service qu'il peut rendre :

— réaction rapide et efficace à l'aide de matériels automatisés,

— accroissement de l'efficacité des moyens de détection et des armes grâce à une meilleure utilisation de l'information recueillie,

— aide à la décision humaine.

Ces deux derniers résultats sont atteints en élaborant une synthèse des renseignements obtenus sur l'environnement et en les présentant aussi bien au centre de commandement où sont prises les décisions fondamentales qu'aux centres secondaires de décision locale et d'action. Ceci

est rendu possible par le choix d'une structure possédant les qualités suivantes :

— rapidité d'action,

— capacité de traiter une grande quantité d'information et de renouveler celle-ci rapidement,

— séparation du travail donc centralisation des décisions fondamentales, mais délégation importante des décisions et des actes concernant chaque fonction.

Les moyens techniques qui facilitent l'obtention de ces performances sont :

— l'automatisation de la détection,

— la normalisation des transferts d'information dans un réseau rapide et la banalisation des interfaces,

— la définition d'un centre de synthèse et de décision où la communication soit aisée entre l'homme et la machine,

— l'automatisation de l'action, dès que la décision a été prise.

2.2. — On obtient ainsi la structure type d'un système informatique (voir fig. 3). Celui-ci comprend :

— des matériels capteurs d'informations sur le système ou sur son environnement,

— des organes de commande assurant une automatisation des actions du système,

— des supports de communication entre l'homme et l'ensemble formé par la machine et ses programmes,

— des moyens de communication entre systèmes.

3. — STRUCTURE DU SYSTEME « HALIOTIS »

3.1. — Le système dont il est question maintenant est un équipement de sous-marin, et les phénomènes qu'il s'agit de contrôler ou de commander ont une évolution relativement lente (beaucoup moins rapide que pour les radars et les engins aériens).

Le système comprend :

— comme capteurs d'information :
  * des sonars actifs et passifs,
  * des radars et appareils de contre-mesures électromagnétiques,
  * des périscopes ;

— comme organes de commande :
  * des asservissements analogiques auxquels les calculateurs envoient des consignes. Il s'agit par exemple de la commande des tubes lance-torpilles et des torpilles,

* des asservissements numériques pour lesquels le calculateur est un élément de la boucle de commande ;

— comme moyens de communication homme-machine :

* une table traçante pour la sortie de valeurs,
* un pupitre et une console de visualisation sur écran cathodique,
* des pupitres d'affichage de données alphanumériques également sur écran cathodique,
* un pupitre de commandes centralisées ;

— Il n'y a pas d'interface vers d'autres systèmes.

3.2. — Le système est organisé autour d'un centre de traitement des données (CTD) qui contient deux groupes de calcul CAE-133 et qui les affecte aux fonctions qui doivent être remplies.

Le calculateur numérique CAE-133/174 comporte :

— une mémoire à tores de ferrite de 32*K* mots de 15 chiffres binaires. Le cycle de mémoire est de 2 µs ;

— une unité de traitement simplifiée puisque ne comprenant que 6 registres pour son fonctionnement. Les instructions (logandes) sont à 1/2 adresse et sont élémentaires. La plupart des opérations sont réalisées par appel de sous-programme en mémoire (logramme). On obtient ainsi des temps d'opération plus importants que ne le laisserait prévoir la cadence de 1 mégahertz de l'horloge de base :

* addition, soustraction simple longueur : 20 microsecondes,
* addition, soustraction double longueur : 40 microsecondes,
* multiplication simple longueur : 65 microsecondes,
* division simple longueur : 80 microsecondes,
* opération en virgule flottante (addition : 140 µs ; multiplication : 200 µs),
* sinus, cosinus simple longueur : 350 µs.

Par contre les opérations logiques demeurent rapides :

* décalage logique de n positions : ($4 + n$ microsecondes),
* intersection : 5 microsecondes,
* normalisation : ($4 + n$ microsecondes) ;

— 1 canal *A* de liaison intercalculateurs, à 30 chiffres binaires, non simultané,

— 1 canal *B* d'accès à 30 chiffres binaires, non simultané,

— 1 canal *C* d'accès à 15 chiffres binaires, non simultané,

— 4 canaux *D* d'accès à 15 chiffres binaires, simultanés.

Les outils de programmation sont :

* un assembleur (logrammes et logandes),
* un compilateur temps réel,
* des programmes de service.

Ce calculateur, conçu vers 1960, ne possède pas de dispositifs facilitant la multiprogrammation :

* protection mémoire,
* horloge temps réel, et compteur de temps écoulé (1),
* registres d'index et de base,
* mode exécutif.

Un seul calculateur, malgré sa mémoire de 32K mots et ses quatre canaux d'entrée-sortie simultanés, est insuffisant pour les besoins du problème : ceci a conduit à l'utilisation de 2 calculateurs par groupe.

Le système d'interruption de ce calculateur, en contre-partie de sa simplicité, possède le désavantage de nécessiter une longue analyse pour détecter la cause de l'interruption, ce qui conduit à un temps de réaction assez élevé (1 à 2 ms).

3.3. — Les blocs mémoires du CTD sont insuffisants pour contenir les programmes de toutes les activités. En plus on veut disposer d'importants programmes de maintenance. On utilise une mémoire de masse à base de disques magnétiques militarisés. Le dispositif comprend 6 tourne-disques pouvant recevoir chacun un monodisque amovible et deux unités de liaison vers les calculateurs.

3.4. — Les interfaces du CTD et des fonctions sont alors les suivantes :

— les canaux *D*, simultanés,

— une centrale de codage de signaux analogiques, branchée sur un canal *D*,

— les unités 102 reliées au canal *C* et permettant le multiplexage de signaux logiques.

3.5. — Le schéma détaillé du CTD apparaît en figure 5 :

— la 141 est un lecteur-perforateur de bande perforée auquel est associée une machine à écrire. Elle sert aux chargements initiaux des calculateurs.

4. — ORGANISATION DES PROGRAMMES HALIOTIS

4.1. — Enoncé du problème

Le système décrit ci-dessus doit permettre de traiter à un instant donné une ou plusieurs fonctions. Cette (ou ces) fonction (s) constitue(nt) à cet instant ce que l'on appelle *l'activité* du système.

(1) Le compteur a été rajouté dans un périphérique : on l'appelle dans le texte « sablier ».

Le problème se pose de la manière suivante :

— l'activité du système à chaque instant dépend de décisions opérationnelles prises par le commandement du sous-marin en fonction de données extérieures au système.

N. B. — Il est cependant concevable qu'une des fonctions du système soit précisément l'aide à la décision du commandant. Nous n'envisagerons pas ce cas ici.

— Les diverses fonctions, et par suite l'activité du système elle-même, sont formées d'un ensemble de travaux (1) qui doivent être exécutés concurremment en respectant certaines contraintes de priorité, d'urgence et de synchronisme (ces trois termes seront définis de façon précise plus tard).

4.2. — Influence des données du problème sur la programmation

Les données du problème ont une influence indépendante du matériel sur la structure des programmes. Elles entraînent :

— la nécessité d'un programme *Superviseur* pour gérer les activités du système.

Le rôle de ce superviseur sera tout naturellement :

* la gestion des communications entre le commandement et le système ;
* la gestion des changements d'activité et des changements de configurations matérielles qui en résultent (affectation des calculateurs et des divers périphériques).

— la nécessité d'un programme *Moniteur* pour gérer les travaux constituant l'activité du système.

Les critères de gestion sont au nombre de trois :

* Priorité :

— la priorité d'un travail est choisie lors de la construction du système.

* Urgence :

— l'urgence tient au caractère « Temps réel » du système : certaines informations perdent toute valeur si elles ne sont pas traitées dans un délai bien déterminé.

* Synchronisme :

— par synchronisme nous entendons le fait que l'exécution d'une partie d'un travail peut être soumise à l'exécution préalable d'une partie d'un autre travail.

---

(1) Travail : Un travail est une sous-fonction opérationnelle rendant un service déterminé ; un travail est donc un programme défini par :

— ses échanges d'informations avec les autres travaux,

— les traitements d'informations qu'il effectue.

— La nécessité de fractionner les *travaux* en segments appelés *modules*.

Ce fractionnement est imposé par la condition de synchronisme que nous venons de voir. Les modules possèdent deux caractéristiques essentielles :

* l'exécution d'un module peut être soumise à l'exécution préalable d'un module appartenant à un travail différent ;
* l'exécution d'un module étant commencée, celle-ci *peut* être achevée quel que soit l'état d'avancement des autres travaux.

En conclusion, le problème posé conduit, indépendamment de toute contrainte matérielle, à une organisation des programmes en 4 niveaux hiérarchiques :

* un superviseur unique,
* des moniteurs de multiprogrammation en temps réel, chacun étant caractérisé par la liste des travaux qu'il doit gérer,
* des travaux,
* des modules.

4.3. — Influence du matériel sur la programmation

La nature du problème nous a conduit à la structure hiérarchique décrite plus haut. Il nous faut maintenant examiner la compatibilité de cette structure et du matériel.

Les contraintes matérielles sont les suivantes :

— deux calculateurs reliés par canal *A* travaillent en parallèle à l'exécution d'une activité donnée ;

— la structure du calculateur CAE-133 ne permet pas d'avoir facilement des programmes réentrants.

Les conséquences en seront :

— La nécessité de diviser l'activité entre les deux calculateurs, chacun recevant une partie des travaux ; ceci entraîne en particulier l'existence de deux moniteurs distincts (un par calculateur). Cependant chaque moniteur peut modifier les tables de l'autre.

— L'exécution d'un programme non réentrant, doit être impérativement achevée avant qu'on ne puisse à nouveau l'exécuter, sous peine de le détruire. Ceci entraîne la nécessité de fractionner les travaux en segments de programmes qui, une fois entamés devront absolument être terminés. (On retrouve ici la nécessité de fractionner les travaux en modules.)

Pour tenir compte de ces nouvelles contraintes, les modules devront posséder les caractéristiques suivantes :

— l'exécution d'un module peut être soumise à l'exécution préalable d'un module appartenant à un travail différent ;

— une fois l'exécution d'un module commencée, aucun autre programme n'a le droit d'utiliser les logrammes et sous-programmes tant que le module n'est pas terminé. Donc :

* le module commencé doit être terminé avant qu'un autre travail puisse prendre le contrôle ;
* une interruption survenant au cours du module ne peut être traitée que si le traitement n'utilise pas de logrammes ni de sous-programmes.

Il en résulte que le traitement de la plupart des interruptions doit être différé jusqu'à la fin du module interrompu : l'interruption a pour seul effet immédiat de placer un indicateur qui est examiné à la fin du module.

— Comme le traitement des interruptions est en général différé jusqu'à la fin du module en cours, il faut que le temps d'exécution de ce module ne soit pas *trop long*. En principe la durée maximale d'exécution d'un module sera de l'ordre de 50 ms, temps de réponse maximal admissible à une interruption.

5. — LE SUPERVISEUR

Le superviseur assure une liaison permanente entre l'extérieur et le Centre de traitement des données.

Il met en place les ressources nécessaires à une configuration du système (ou contrôle leur mise en place). Ces ressources sont les programmes dont il faut charger les groupes de calcul, mais aussi les matériels du C.T.D. (disques, unités de liaison, permutateurs de câble *D*...) utilisés par la configuration.

Il gère et maintient ces ressources communes à toutes les fonctions. C'est donc par le superviseur que doit s'effectuer toute liaison avec le C.T.D.

Il est appelé de deux manières :

— Soit par une interruption externe provenant des organes du C.T.D.

— Soit par une demande de concours qui lui est posée par un travail d'une fonction. Comme cette demande interrompt le traitement de la fonction, elle est faite sous le contrôle du moniteur de fonction.

Le superviseur comprend une partie fixe (module de liaison avec le pupitre de commande centralisée, module de liaison avec les disques, module de liaison avec la 141) et une partie variable avec la configuration.

Il existe un programme superviseur dans chaque calculateur dès que celui-ci est disponible pour les configurations. La partie fixe est identique dans tous les calculateurs, seule la partie variable, qui dépend de l'activité diffère d'un calculateur à l'autre.

Ces superviseurs doivent être considérés comme un programme unique réparti en 4 calculateurs et assurant le multitraitement des programmes.

Le pupitre de commande centralisée est l'emplacement par lequel les quatre parties du superviseur communiquent entre elles.

6. — LES MONITEURS DE MULTIPROGRAMMATION EN TEMPS REEL

6.1. — Généralités

L'analyse des fonctions fait apparaître un grand nombre de travaux qui doivent être exécutés concuremment.

L'ordonnancement des travaux dans le temps est réglé par un moniteur dont le rôle essentiel consiste à sélectionner le travail à exécuter ; une fois cette tâche accomplie, le contrôle est donné au travail choisi qui pourra exécuter un ou plusieurs modules (toujours un nombre entier, comme il a été dit plus haut). Le contrôle revient au moniteur, soit en fin de travail, soit à la suite d'une interruption d'un sablier, et le cycle recommence.

6.2. — Table de déclaration des travaux

Le choix du travail à exécuter dépend à la fois de sa priorité et de son urgence ainsi que de contraintes de synchronisme avec les autres travaux.

Pour effectuer ce choix le moniteur dispose d'une table spéciale : la table de déclaration des travaux (TABDET) dans laquelle tous les travaux de l'activité en cours sont représentés.

Pour chaque travail, on trouve dans TABDET les divers paramètres de priorité, d'urgence et de synchronisme permettant d'effectuer le choix. On y trouve également un certain nombre d'autres renseignements. (Nous exposerons le contenu détaillé de cette table après avoir vu le fonctionnement de l'algorithme de choix.)

N. B. — L'algorithme de choix que nous allons présenter est le même quelle que soit l'activité du système. Autrement dit, les différents moniteurs correspondant aux différentes activités possibles du système, ne se distinguent que par leur TABDET.

6.3. — Algorithme de choix d'un travail par le moniteur

6.3.1. — Il existe deux catégories bien distinctes de travaux : les travaux périodiques et les travaux apériodiques. Pour les uns comme pour les autres les durées d'exécution peuvent être variables et parfois imprévisibles. C'est pourquoi le moniteur ne peut décider d'heures fixes pour le début et la fin de chaque travail.

On peut alors décrire de la façon suivante le déroulement d'un travail (T) :

— *A l'heure Ho l'exécution du travail* (T) *est demandée.*

* Si (T) est périodique : cette demande est faite par le moniteur lui-même ;
* Si (T) est apériodique : cette demande est faite soit par un autre travail, soit par un événement extérieur au système.

Dans tous les cas cette demande a pour effet de donner une valeur aux deux dates suivantes :

x : date à partir de laquelle le travail peut commencer ;

y : date à partir de laquelle le travail doit avoir commencé.

Nous voyons donc que le temps de réponse du système à la demande d'exécution du travail (T) sera :

* au minimum x — Ho,
* au maximum y — Ho.

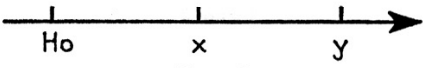


Figure 1

On pose x — $Ho = \delta P1$ et $y - x = \delta P2$:

$\delta P1$ et $\delta P2$ sont deux durées dépendant du travail et fixées au moment de son écriture.

$\delta P1$ est appelé « Temps d'attente minimum du travail (T) ».

$\delta P2$ est appelé « Fourchette d'initialisation du travail (T) ».

$\delta P1$ et $\delta P2$ étant connus x et y en sont déduits très facilement.

— *A l'heure H'o* $(x \leqslant H'o \leqslant y)$ *le moniteur a décidé que le travail* (T) *devait commencer.*

Le travail (T) commence donc immédiatement et exécute intégralement son premier module sans être interrompu (cf. § 4).

A la fin du premier module le travail s'alloue un temps de réponse limite θ_1 tel que l'exécution du module suivant dans le travail (T) débute avant la date $z = y + \theta_1$.

A la fin de chaque module i le travail change la date limite de démarrage du module suivant en faisant :

$$z := z + \theta_i$$

où θ_i est une durée dépendant de la suite du travail $(\theta_i \geqslant 0)$.

L'existence de cette date limite variable permet au moniteur de contrôler plus étroitement l'avancement du travail : si celui-ci prend du retard, le fait sera détecté plus tôt que si une date limite lointaine était fixée pour l'ensemble du travail.

— *Cette discussion est résumée par la figure 2,* dans laquelle on a introduit l'état du travail ; un travail peut être dans l'un des trois états suivants :

* en attente,
* prêt,
* en exécution.

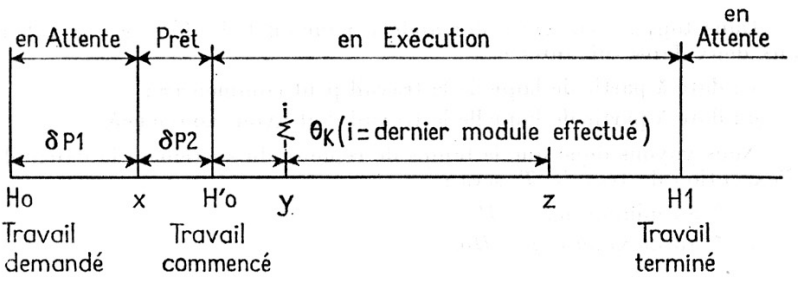


Figure 2

N. B. — 1) Le schéma est valable pour le travail (T) à la fin de son i-ème module.

2) Pour un travail apériodique le schéma s'applique parfaitement. Pour $H < H_0$ on a :

$$\begin{cases} x = \infty \\ y \text{ et } z = \text{indéterminés} \end{cases}$$

Lorsque le moniteur déclare le travail prêt ($H \geqslant x$) il fait à nouveau :

$$x := \infty$$

3) Pour un travail périodique il n'y a pas de demande explicite du travail ; donc H_0 n'existe pas à proprement parler.

Un travail périodique est demandé directement par le moniteur en faisant :

$$x := x + P \text{ (période)}$$

au moment où le travail est déclaré prêt.

La valeur de y pour la nouvelle période est mise à jour par le moniteur lorsque le travail a été exécuté entièrement en faisant

$$y := x + \delta P2$$

4) Il est entendu qu'un travail commencé ne peut être suspendu qu'à la fin d'un module.

6.3.2. — Le choix du travail à effectuer dépend de sa priorité et de son urgence (nous parlerons des conditions de synchronisme plus tard) :

— la priorité d'un travail est fixée au moment de la constitution de TABDET. Appelons-la π.

— l'urgence d'un travail peut être mesurée, d'après ce que nous avons vu en 6.3.1. par : (H étant l'heure courante)

$$DT = \begin{cases} . \ y - H \text{ si le travail a été demandé et est « en attente ».} \\ . \ y - H \text{ si le travail est « prêt ».} \\ . \ z - H \text{ si le travail est « en exécution ».} \end{cases}$$

En effet y a été défini comme la date limite pour commencer un travail demandé et z comme la date limite pour commencer le prochain module d'un travail entamé.

On peut alors définir une fonction de choix par :

$$\boxed{C = \pi + \alpha DT} \quad (\alpha = \text{constante}).$$

Le moniteur choisira pour exécution, le travail ayant la fonction de choix minimum.

6.3.3. — *Conditions de synchronisme*

Le moniteur doit contrôler l'avancement parallèle des divers travaux, l'exécution d'un module d'un travail (T) pouvant être soumise à l'exécution préalable d'un module d'un travail (T').

Pour ce, il suffit que chaque travail dispose d'un mot, appelé mot « Autorisation d'exécution » ; les divers bits de ce mot correspondent aux conditions préalables, et le travail ne peut commencer ou continuer que si tous les bits spécifiés par un masque déterminé sont corrects.

6.4. — Fonctionnement du moniteur

Nous sommes maintenant en mesure de décrire le fonctionnement du moniteur.

6.4.1. — *La TABDET*

La TABDET contient les diverses informations que nous avons décrites plus haut. On aura donc pour chaque travail :

— *Des valeurs fixées au moment de l'écriture du travail* :

* Priorité π
* Temps d'attente minimum $\delta P1$
* Fourchette d'initialisation $\delta P2$
* Nombre d'instructions du travail
* Adresse d'exécution du travail
* Rang de suppression ρ

  Nous n'avons pas encore rencontré ce paramètre. Il s'agit du rang d'élimination du travail en cas de saturation du système.
  A un instant donné il existe un rang de suppression critique ρ_0 attaché au moniteur.
  Un travail ne pourra être exécuté que si :

$$\rho > \rho_0$$

— *Des valeurs tenues à jour par le moniteur ou d'autres programmes*

* État du travail
* Date x, conservé dans une mémoire appelée DAT1.
* Date y et date z.

— Comme z n'est pas déterminé tant que le travail n'est pas commencé et que y n'est plus intéressant dès que le travail est commencé, on peut conserver y et z dans une même mémoire appelée DAT2.
Avant la fin du premier module DAT2 contiendra y, puis DAT2 contiendra z.

* Urgence DT = DAT2 — H

DT n'a de sens que pour un travail demandé.

— *Des renseignements de service*

* Indicateur « Travail en mémoire rapide »
* Mot « Autorisation d'exécution » et son masque.

Le mot « Autorisation d'exécution » est tenu à jour directement par les divers autres travaux intéressés.

Le masque est tenu à jour par le travail lui-même à la fin de ses divers modules.

6.4.2. — *Le fonctionnement du moniteur*

A) *Balayage de la TABDET*

Nous avons adopté une organisation qui permet au moniteur de remplir ses diverses tâches au moyen d'un seul balayage de la TABDET.

Au cours de ce balayage le moniteur effectue, pour chaque travail, les opérations suivantes :

— Examen de la fourchette d'initialisation du travail. Si celle-ci est entamée, le travail est déclaré prêt, et le moniteur met à jour DAT1 :

$$\begin{cases} \text{DAT1} := \text{DAT1} + P \text{ pour un travail périodique} \\ \text{DAT1} := \infty \text{ pour un travail apériodique.} \end{cases}$$

— Comparaison de l'heure courante H à DAT2 (seulement pour un travail demandé).

Compte tenu de la définition de DAT2, H ne devrait jamais dépasser DAT2. Si H a dépassé DAT2 de plus d'une certaine valeur, il y a saturation du système : il faut retirer des travaux.

Pour ce, le moniteur augmente le rang de supression critique ρ_0 : l'élimination proprement dite sera faite par un sous-programme spécialisé.

— Calcul de l'urgence DT et de la fonction de choix (seulement pour un travail prêt).

Le balayage permet en outre :

— de déterminer les deux DT les plus petits ;

— de trouver le travail ayant la fonction de choix minimum.

C'est ce travail que le moniteur choisit.

B) *Opérations de services*

Avant de donner le contrôle au travail choisi, le moniteur doit effectuer deux opérations de service :

1) Réintégration éventuelle de travaux.

A l'aide du DT_{min} déterminée en A) on peut faire une estimation de la charge du système. On définit une fonction de charge F, calculée au moyen de la relation :

$$F_{nouveau} = \alpha F_{ancien} + (1 - \alpha) DT_{min} \quad (\alpha = C^{te})$$

Si F dépasse un certain seuil S, le moniteur diminue le rang de suppression critique ρ_0 ; la réintégration proprement dite sera faite par un sous-programme spécialisé.

2) Chargement du sablier.

Les 2 DT minimum calculés en A) permettent de déterminer le répit avant un dépassement possible du DAT2 le plus proche par l'heure courante.

On va donc placer la valeur de ce répit dans un compte-temps spécial appelé sablier. Ce sablier est décompté toutes les millisecondes et envoie une interruption quand il est vide. Le moniteur reprend alors le contrôle dès la fin du module en cours.

7. — LES TRAVAUX

7.1. — Généralités

Un travail correspond à l'exécution d'une sous-fonction opérationnelle. Il n'utilise toujours qu'un seul calculateur mais peut nécessiter des résultats fournis par un travail exécuté dans l'autre calculateur. Dans un tel cas le premier travail est arrêté jusqu'à ce que les résultats du second soient disponibles.

Comme un travail est fractionné en modules, le programme du travail comprend :

— l'ensemble de ses modules, chacun formant une unité de programme répondant aux propriétés énoncées plus haut au paragraphe 4,

— un programme standard, identique pour tous les travaux dont le rôle essentiel est d'enchaîner les divers modules.

Le fonctionnement de ce programme standard, appelé Programme principal Travail (PPT) est décrit plus complètement dans les deux paragraphes suivants.

7.2. — Rôle du programme principal travail (PPT)

Le rôle essentiel du PPT est d'enchaîner les divers modules composant le travail. Seul le PPT est en communication avec le Moniteur : on peut donc considérer que les modules sont des sous-programmes appelés successivement par le PPT. (Cependant à la différence d'un sous-pro-

gramme, ces modules appartiennent toujours en propre à un travail et un seul.)

En outre, le PPT effectue :

— la tenue à jour de certains mots dans la TABDET :

* mise à jour à la fin de chaque module du masque associé au mot autorisation d'exécution,
* mise à jour de DAT2 avant de céder le contrôle à un module ou au moniteur ;

— l'examen de l'indice de renvoi au moniteur.

(Cet indice est placé par un module quelconque du travail s'il nécessite un retour au moniteur.)

— l'examen de l'indice de fin de travail.

(Le dernier module exécuté d'un travail place cet indice, qui permet au PPT de constater que le travail est terminé.)

— l'examen de l'indicateur d'interruption différée.

Cet indicateur est placé lorsqu'une interruption n'est pas traitée immédiatement. Il faut donc l'examiner à la fin de chaque module.

7.3. — Fonctionnement du PPT

Pour assurer l'enchaînement des modules, le PPT dispose de deux indices :

— Indice de parcours.

Il est tenu à jour par le PPT et permet de repérer le module en cours d'exécution.

— Indice de ventilation.

Il est tenu à jour par chaque module et permet de repérer le prochain module à exécuter.

A ces indices sont associées diverses tables :

— Table des modules.

Cette table contient les adresses de début des divers modules.

— Table de ventilation.

A chaque module correspond une table de ventilation qui établit une correspondance entre l'indice de ventilation et le module successeur.

— Table de parcours.

Cette table contient les adresses des différentes tables de ventilation.

Pour trouver le prochain module à exécuter le PPT devra donc :

* chercher dans la table de parcours l'adresse de la table de ventilation du dernier module exécuté (repéré au moyen de l'indice parcours),
* chercher dans cette table de ventilation, grâce à l'indice de ventilation, le prochain module à exécuter,
* chercher dans la table de parcours l'adresse du module.

8. — LES TABLEAUX DE VARIABLES

Aux différents niveaux de programme sont attachées des tables de variables : informations en provenance de l'extérieur, résultats de calculs, etc.

On doit distinguer deux types de tables :

— Les tables dont la longueur est fixe et prédéterminée.

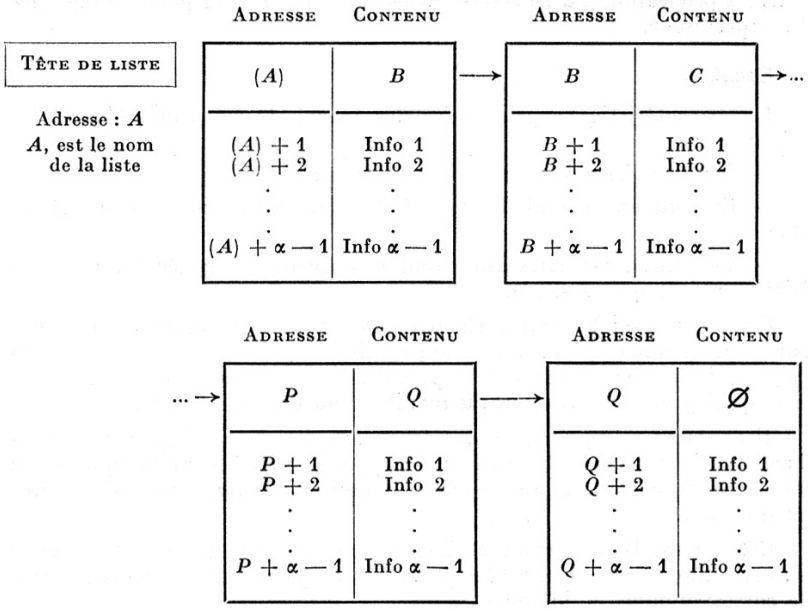
On affectera à ces tables une place de longueur fixe.

— Les tables dont la longueur est variable.

Un certain nombre de données sont transmises au C.T.D. à des instants aléatoires. De ce fait on peut avoir à un instant donné beaucoup d'informations d'un type et peu d'un autre : il serait très peu économique d'affecter à chaque type d'information une longueur de mémoire fixe.

C'est pourquoi toutes les informations de ce type sont rangées dans des listes.

N. B. — On rappelle qu'une liste est un ensemble d'éléments enchaînés les uns aux autres dans un certain ordre. Chaque élément est l'ensemble formé par α emplacements consécutifs de mémoire, le premier étant consacré à l'enchaînement :



Par convention, si la liste de tête contient $\varnothing$ (c'est-à-dire $(A) = \varnothing$) la liste est vide.

Dans le cas présent on a pris $\alpha = 5$. Comme un des emplacements de chaque élément est de toute façon consacré à l'enchaînement, l'utilisation maximale de la mémoire sera de 80 %. Néanmoins cette disposition reste avantageuse.

9. — LES ENTREES-SORTIES

9.1. — Généralités. Sujétions

Les échanges entre C.T.D. et l'extérieur peuvent se faire par l'intermédiaire de 4 types de canaux :

— les canaux de type *A*, *B* et *C* communiquent directement avec le calculateur (ce sont des canaux non simultanés) ;

— les 4 canaux de type *D* communiquent avec l'unité CAE-174 (ce sont des canaux simultanés).

Le traitement des Entrées-Sorties dépend du canal utilisé. Il y a trois procédures distinctes :

Canal B

Le canal *B* assure la communication entre le calculateur et les unités de commande des disques. Comme les disques sont gérés par le superviseur, les programmes d'Entrées-Sorties par canal *B* sont partie intégrante du superviseur.

Canal C

Les caractéristiques principales des transferts sur canal *C* sont les suivantes :

— Ces transferts arrêtent le calculateur ;

— Ce sont en général des transferts mot plutôt que des transferts bloc ;

— Les Entrées-Sorties sur canal *C* doivent en général être satisfaites aussitôt que possible.

C'est pourquoi les transferts par canal *C* sont effectués directement par le programme qui en a besoin ; celui-ci appelle des logrammes standards qui effectuent le transfert, puis il reprend le contrôle d'exécution, sans qu'il y ait intervention du moniteur ou du superviseur.

Il convient de remarquer que les opérandes de ces logrammes standards (variables à transférer et adresses) sont désignées symboliquement ; une table de correspondance donne les indications physiques nécessaires au transfert.

Cette disposition permet d'effectuer des changements éventuels dans les connexions sans changer les programmes (changements dans la table de correspondance uniquement).

Canal A et canaux D

Les caractéristiques de ces transferts sont les suivantes :

— Un transfert sur canal *A* entre les deux calculateurs d'un groupe arrête les deux calculateurs.

— Au contraire, un transfert sur canal *D*, une fois initialisé, laisse le calculateur libre d'effectuer toute autre opération. La fin de transfert génère une interruption.

— Par contre, les transferts sur canal *A*, aussi bien que sur canaux *D*, sont en général des transferts bloc.

— Enfin, les transferts sur canal *A*, aussi bien que sur canaux *D*, ne présentent en général pas un caractère d'urgence absolue.

Pour ces quatre raisons, les transferts sur canal *A* et sur canaux *D*, seront effectués au moyen d'un programme spécial de gestion des entrées-sorties, appelé par le moniteur en service dans les mêmes conditions qu'un travail ordinaire.

On appellera ce programme spécial de gestion des entrées-sorties : *G*.

9.2. — Le programme G

9.2.1. — *Liste des messages à transférer*

Une liste contient tous les messages à transférer sur canal *A* ou sur canaux *D*.

Les éléments de cette liste sont constitués de la manière suivante :

<table>
<tr><th>Bit / Mot</th><th colspan="5">15 1</th></tr>
<tr><td>1</td><td colspan="5">Adresse de l'élément suivant</td></tr>
<tr><td>2</td><td>15
Indice P</td><td>14
Indice E</td><td>13 à 7
Désignation du destinataire (symbolique)</td><td>6 à 2
Réservé</td><td>1
Indice V</td></tr>
<tr><td rowspan="2">3</td><td colspan="5">Indication du bit à modifier en fin de transfert</td></tr>
<tr><td>15 à 12
N° du bit</td><td colspan="2">11 à 3
N° de la mémoire moniteur concernée</td><td>2
Indice X</td><td>1
Indice W</td></tr>
<tr><td>4</td><td colspan="5">Première adresse de la zone de mémoire tampon</td></tr>
<tr><td>5</td><td colspan="5">Nombre de mots de la zone tampon</td></tr>
</table>

Indice P { Ø message non prioritaire
1 message prioritaire

Indice E { Ø transfert calculateur → périphérique
1 transfert périphérique → calculateur

Indice V { Ø normal
1 les mots 4 et 5 sont l'information à transférer

Indice X { Ø le bit à modifier doit être mis à Ø
1 le bit à modifier doit être mis à 1

Indice W { Ø le bit à modifier doit l'être en fin de transfert
1 le bit à modifier doit l'être lors de la réception du message (transfert par canal A)

Chaque programme ayant besoin d'un transfert ajoute un message dans cette liste.

Les bits à modifier en fin de transfert (cas des canaux D) ou en début de transfert (cas du canal A) ne sont autres que les bits témoins de transfert, du mot « autorisation d'exécution » de la TABDET.

9.2.2. — *Principe de fonctionnement du programme G*

Lorsque le programme de gestion des entrées-sorties prend le contrôle d'exécution, il tentera d'effectuer tous les transferts en attente sur canal A et sur canaux D. Cependant, si certains de ces transferts ne sont pas possibles immédiatement, en règle générale le programme n'attendra pas : ces transferts seront effectués au cours d'une prise de contrôle ultérieure du programme.

Le programme G commence par diviser la liste des messages à transférer en deux sous-listes :

— sous-liste des messages à transférer par canal A,

— sous-liste des messages à transférer par canaux D.

Première catégorie (canal A)

Les transferts par canal A se font pour les deux calculateurs (calculateur appelant et calculateur appelé) par l'intermédiaire de tampons spéciaux. Chaque calculateur dispose d'un tampon où il stocke les informations à transmettre et d'un tampon où il stocke les informations reçues.

Sur canal A, le calculateur appelant demande toujours à transmettre des informations ; si en fait le calculateur appelant a besoin d'information venant de l'autre calculateur, il lui transmet un message pour lui signaler ce fait ; quand le calculateur appelé est prêt à répondre, c'est lui qui prendra l'initiative de rappeler le 1er calculateur pour lui transmettre les informations demandées.

Deuxième catégorie (canaux *D*)

Le programme *G* examine un à un chaque message :

— Si le canal *D* concerné par le message est libre le transfert est immédiatement initialisé.

— Si le canal *D* concerné par le message est occupé le transfert est remis : il sera initialisé la prochaine fois que le programme *G* aura le contrôle d'exécution.

Quand le programme *G* a examiné les deux sous-listes, il rend le contrôle d'exécution au moniteur.

10. — CONCLUSION

Le système tel qu'il vient d'être décrit est en cours de programmation ; bien des solutions proposées devront subir l'épreuve de l'expérience avant d'être définitivement adoptées.

Il apparaît également que certains problèmes peuvent être résolus de plusieurs façons distinctes ; ce n'est qu'en comparant expérimentalement ces diverses façons qu'on pourra choisir la meilleure.

L'étude expérimentale se fait essentiellement par simulation. La structure modulaire du système de programmation rend d'ailleurs cette étude particulièrement aisée.

La simulation a deux aspects différents suivant le type de programmes à étudier :

— *Programmes de servitudes* (Superviseur, Moniteur, Entrées-Sorties, Interruptions).

Ces programmes peuvent être essayés en utilisant au lieu de travaux réels, des travaux fictifs.

Chaque module de ces travaux fictifs consiste à effectuer une atteinte ayant la même durée qu'un module réel.

On pourra par cette méthode, contrôler en particulier l'ordonnancement des travaux par le moniteur.

— *Programmes opérationnels*

L'étude de ces programmes nécessite une simulation des censeurs. Celle-ci peut se faire de deux façons :

— Par des matériels construits spécialement à cet effet.

— Par programme à l'aide de modules spéciaux remplaçant les modules d'entrées-sorties.

Enfin, il faut signaler, dans le cas de la rédaction de systèmes de cette taille par des équipes importantes de programmeurs, le rôle fondamental de la documentation, aussi bien en cours d'étude pour assurer l'homogénéité de la réalisation qu'en fin de programmation pour permettre la maintenance et la modification des programmes.

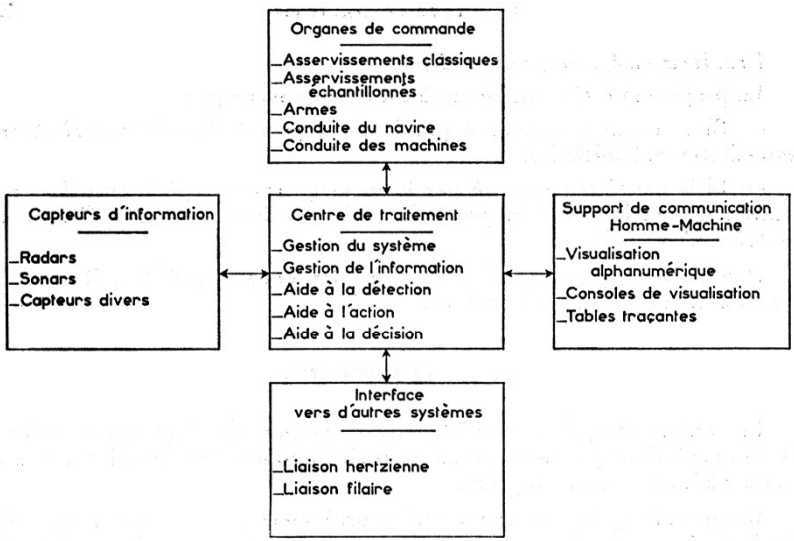


Figure 3
Structure type d'un système informatique

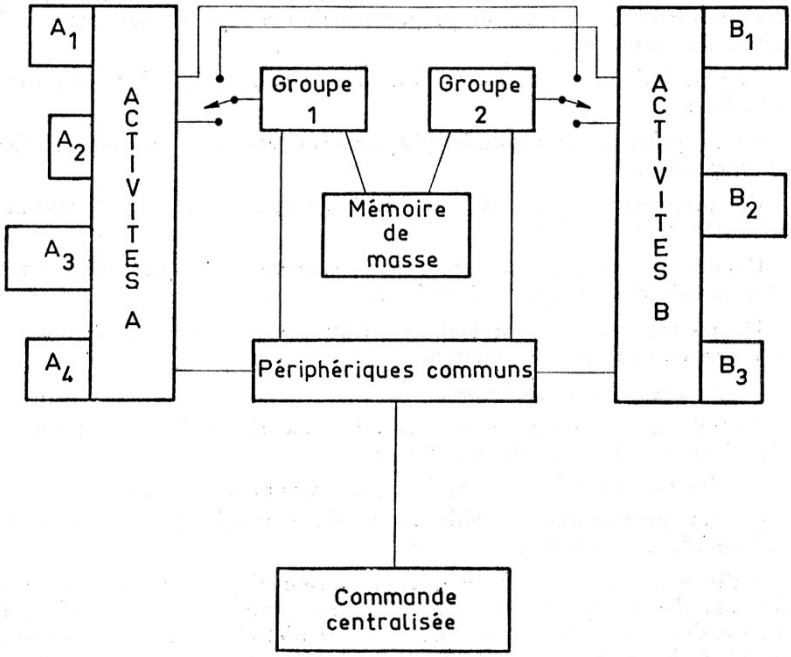


Figure 4
Schéma du système Halistis

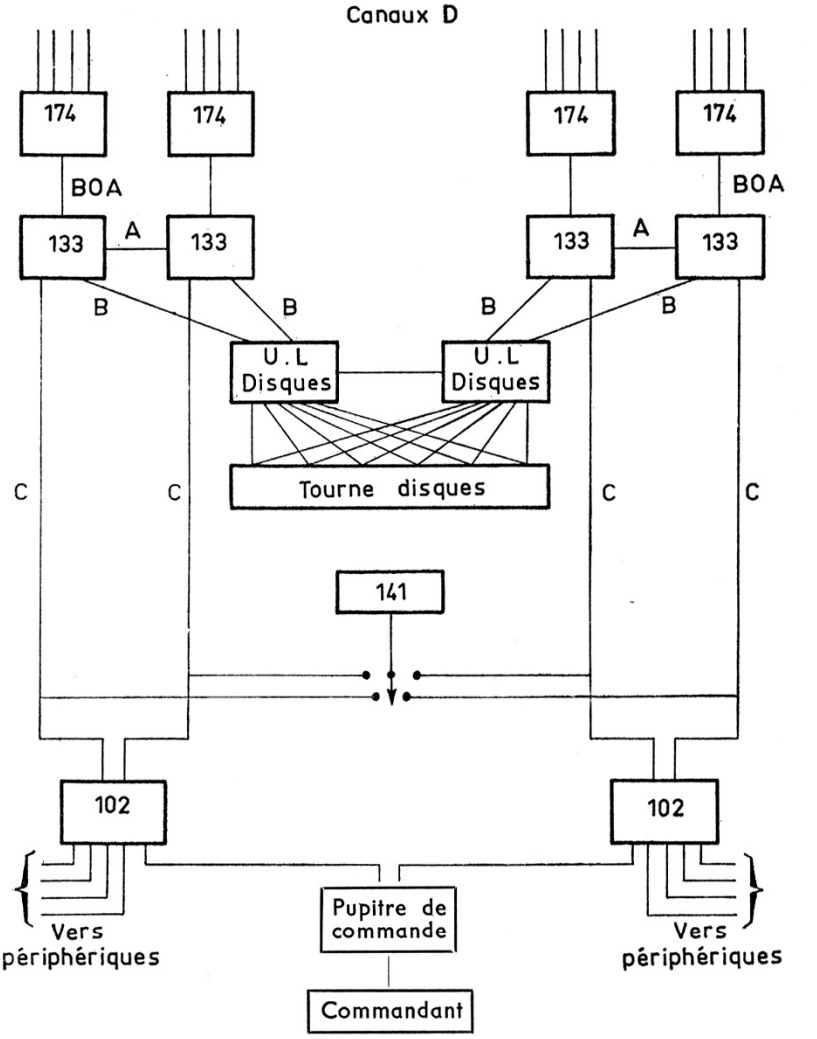


Figure 5
Schéma du centre de traitement des données

BIBLIOGRAPHIE

DEDIEU, *Étude relative à la simulation d'un système modulaire de calcul*, Laboratoire d'informatique de Toulouse, étude faite pour la D.T.C.N., Convention nº 925.525.