Info-Ci

Nº9 3 mars 1989

Bulletin édité par le Centre informatique
de l'Université de Lausanne

BSP - Dorigny
1015 Lausanne

Rédaction: A. Guerid, Centre informatique.
Impression: imprimerie Tex, Prilly

Le mot du délégué

Le système informatique de l'Université poursuit une évolution qui s'est accélérée ses derniers temps pour le grand bien de chaque utilisateur.

Dans le domaine de la téléinformatique, notre Centre vient d'élargir considérablement les prestations offertes par son réseau central. Une machine de communication utilisant le système UNIX a été installée sur le réseau. Cette innovation facilite l'intégration du nombre toujours croissant des stations de travail individuelles de la nouvelle génération.

Le système central de l'Université a également amélioré ses services pour l'accès aux deux machines CRAY des Ecoles Polytechniques Fédérales. Notre Université est actuellement la seule parmi les Universités suisses à posséder un système convivial pour communiquer avec les deux machines.

Quant aux cours de formation dispensés par le Centre Informatique, on remarque également une évolution très réjouissante. En effet, les cours de base en bureautique et pour l'utilisation du système central ont dû être dédoublés vu le nombre très important des participants.

Last but not least, notre Université a eu l'honneur de devenir membre d'un club très exclusif appelé A.U.C. (Apple University Consortium). Ce statut de membre nous permet entre autres de bénéficier de conditions très favorables lors de l'achat de matériel Apple.

Comme on le constate, notre CI fait preuve d'un dynamisme très réjouissant grâce à l'enthousiasme de tous ses collaborateurs. Nous les encourageons vivement à poursuivre cette voie.

Le Délégué à l'informatique,
G. Chapuis

Sommaire

Avis et nouvelles

Compilateur C sur ULYS.	**Page 2**
Insertion de graphiques Mac dansTEX/PC.	**Page 2**
Nouvelles du LAVC.	**Page 2**
Un lauréat du concours CRAY au CI.	**Page 3**
Un prix Seymour Cray pour un petit ver.	**Page 3**
Les gens qui font le CI.	**Page 4**
ULYS: archivage et sécurité.	**Page 5**
Cours d'informatique du CI: description.	**Page 5**
ULYS: ressources consommées.	**Page 8**

Annonces

- Cours du prochain semestre.
- Club Macintosh à l'UNIL.
- Cours sur graphisme.

Annexes

- Note sur le Mailing avec Word.
- Interface musical MIDI.
- Sortie de graphique UNIRAS sur laserWriter.
- Utilisation de la passerelle TCP/IP-DECNET.
- Sauvegarde des bases BASIS et INGRES.
- Bibliothèque NAG-FORTRAN: Des nouveautés.
- CRAY-2 et CRAY-X-MP: Des monstres au service des chercheurs de l'UNIL.

Avis et nouvelles

COMPILATEUR C SUR ULYS

Le VAXCluster ULYS est dorénavant pourvu d'un compilateur C. Même si ce langage de programmation n'est pas celui qui convient le mieux à l'environnement VMS (le langage C étant plus adéquat dans un milieu UNIX), son utilisation très répandue justifie sa présence sur ULYS.

Comme pour les autres langages, le compilateur VAX C produit, à partir d'un fichier source contenant des instructions C, un fichier objet dont on doit éditer les liens pour produire une image exécutable. A partir d'un fichier source "program.c" (le type par défaut des fichiers source C étant ".c"), la suite standard de commandes de compilation et exécution du programme est donc:

```
$ CC program
(crée l'objet "program.obj")
$ LINK program
(crée l'exécutable "program.exe")
$ RUN program
(exécute l'image)
```

Des informations sur le compilateur VAX C sont disponibles avec la commande HELP CC. En particulier, l'entrée "`HELP CC Link_libraries`" indique comment définir un nom logique pour accéder à la "VAX C Run-Time Library". Des notes sur la version du compilateur installée (V2.4) se trouvent dans le fichier:

`SYS$HELP:VAXC024.RELEASE_NOTES.`

Pour des informations complémentaires contacter J. Guélat/CI.
Amateurs de C, à vos marques...

Insertion de graphiques ou de dessins créés par Mac dans des fichiers Tex développés sur IBM AT.

L'insertion de graphiques, de dessins, voire de photos est déjà possible dans beaucoup de traitements de texte; je pense en particulier à Word.

A la base, TeX n'a pas été conçu pour remplir ce rôle. Néanmoins sa grande souplesse et le fait qu'un fichier TeX puisse être traduit en PostScript (par exemple avant d'être envoyé à une AppleLaserWriter) rendent toutes ces opérations théoriquement possibles. Il reste à les réaliser.

Je noterais tout d'abord que la version de TeX qui tourne sur Mac (Textures) permet l'insertion de toutes sortes de fichiers graphiques au moyen du Presse-papiers; pour cela je vous renvoie à la documentation correspondante.

Suivant les indications d'un TeXhaxer, j'ai mis au point une procédure permettant de récupérer des fichiers MacDraw, SuperPaint, etc. sous forme de fichiers PostScript, puis de les transférer sur IBM AT et enfin de les incorporer dans des documents en utilisant le logiciel DviLaser de Textset.

Les personnes intéressées peuvent contacter D. Cangemi de l'Institut de physique théorique à l'aide du courrier électronique : ULA :: DCANGEMI

Des nouvelles du LAVC

Quelques notes pour l'utilisation du Local Area Vax Cluster de l'UNIL (voir Info-CI nº7):

• Dans la version 3.3 du système de fenêtrage qui vient d'être installée, DEC a corrigé quelques erreurs; en particulier, les touches <CTRL Z> et <CTRLY> ne sont plus permutées sur le clavier Suisse romand.

• Dans cette version, un troisième type de

terminal est émulé (rappelons que la version précédente émulait un terminal alpha-numérique VT200 et un terminal graphique Tektronix 4010/14). Ce nouveau type est Regis, le standard graphique de DEC. Cette émulation est forte intéressante, car elle permets de faire tourner des programmes sur ULYS qui demandent une puissance de calcul importante et d'afficher les résultats sur la station de travail. Ceci est spécialement intéressant pour des applications graphiques (par exemple UNIRAS), car l'on peut ainsi profiter de la vitesse d'affichage élevée d'une station de travail.
Pour l'utilisation, donner:

`'LVT340'` comme type de terminal dans UNIRAS et `'340'` dans UNIGKS.

• Une station du LAVC est mise en libre service aux utilisateurs de la section de physique dans le BSP.

Pour des informations complémentaires contacter E. Niebur / CI.

Un lauréat du concours Cray au Centre informatique

Un premier assistant de la Section de physique collabore à temps presque complet avec la Centre informatique. Depuis le 1er novembre, ce poste est occupé par **M. E. Niebür**. Les utilisateurs du graphisme sur le Vax cluster Ulys ont certainement déjà pu apprécier ses qualités de conseiller au sein du groupe d'assistance. Récemment, le travail de thèse de M. E. Niebür a été récompensé par un prix de la société Cray. Il présente dans ce bulletin un léger aperçu de son travail.

Un prix Seymour Cray pour un petit ver

Ernst Niebur
Section de physique
Université de Lausanne

Les super-ordinateurs actuels permettent de faire des calculs extrêmement complexes. Un CRAY fait en une seconde des calculs qui prendraient plusieurs semaines à l'homme.

Cependant, il y a beaucoup de choses que l'homme ou même des animaux savent faire beaucoup mieux qu'un super-ordinateur. Un exemple est la vision : on n'a aucun problème pour voir et pour identifier des objets qui se trouvent autour de soi. Un autre exemple est la compréhension de la parole. Programmer ces tâches sur un ordinateur est très difficile.

Nous aimerions alors mieux comprendre comment l'homme et les animaux résolvent ces problèmes; en d'autres mots, nous aimerions comprendre le fonctionnement de leurs systèmes nerveux.

Dans ce travail, nous avons choisi un animal primitif dont le système nerveux est très simple et très bien connu. Nous avons essayé de comprendre le fonctionnement de son système nerveux, celui des vertébrés étant trop compliqué.

L'animal choisi est le nématode *Caenorhabditis elegans*, un petit ver d'un millimètre de longueur. Ce ver a exactement 302 neurones (cellules nerveuses) et chaque neurone a, en moyenne, 20 synapses (connections qui les relient avec d'autres neurones). On synapses. Nous avons donc le "plan du câblage" du système nerveux de ce ver. Notre but a été la définition des neurones modèles réalistes basés sur la neurophysiologie et de faire une simulation par ordinateur de ce système nerveux. La comparaison entre les résultats de la simulation et le comportement du ver avait pour objectif de démontrer si notre modèle était valable.

Nous avons d'abord étudié la locomotion de ce ver qui se meut par reptation, donc comme un serpent. Il nous a fallu trouver le schéma spatio-temporel d'excitation musculaire nécessaire à la production de ce mouvement. En d'autres mots, quel muscle à quel moment doit être excité pour générer la reptation ?

Nous avons créé un modèle mathématique des forces entre le ver et son environnement. Ce modèle nous a permis de simuler par ordinateur la reptation du nématode. Pour que le ver montré dans la figure ci-contre puisse aller dans la direction indiquée par la flèche, il doit contracter les muscles qui sont représentés par des lignes épaisses. *Caenorhabditis elegans* est capable d'exciter (et donc contracter) sélectivement ces muscles grâce à des capteurs d'élongation qui font partie de son système nerveux. Ainsi, nous avons pu expliquer la reptation de *Caenorhabditis elegans*. La simulation nous a fourni, en plus, des informations sur le mode de fonctionnement du circuit nerveux en question.

Ce travail a été fait dans l'Institut de physique théorique sous la direction du Prof. Paul Erdös.

Les gens qui font le Centre informatique

Nom et prénom	fonction	Téléphone
Jacot-Guillarmod Pascal	Direction	tél. 692.23.01
Béguin Martine	Secrétariat	**tél.692.23.11**
Pfersich Charles	Gestion	tél. 692.23.12
Renaud Guy-Daniel	Adjoint de gestion	tél. 692.23.12
Henchoz Daniel	Chef d'exploitation	tél. 692.23.13
Wenger Jacques	Responsable système	tél. 692.23.15
Genaine François	Pupitreur	tél. 692.23.06
Longchamp Jean-Paul	Responsable réseau	tél. 692.23.03
Mikhail Fayez	Micro-ordinateurs et réseaux	tél. 692.23.14
Guedri Taoufik	Opérateur	tél. 692.23.09/03
Ryter Philippe	Micro-informatique	tél. 692.23.02
Pernet Marie-France	Bureautique	tél. 692.23.05
Guélat Jacques	Programmation	tél. 692.23.10
Küffer Pierre	Graphique/statistiques	tél. 692.23.04
Niebur Ernst	Graphique	tél. 692.23.04
Guerid Abdelali	Bases de données	tél. 692.23.15

ULYS: archivage et sécurité

• Archivage: Comme il a déjà été dit dans Info-CI Nº 8, le Centre Informatique n'offre toujours pas de service d'archivage à la demande des fichiers peu ou rarement utilisés. Nous laissons donc aux utilisateurs, surtout s'ils se trouvent chroniquement à cours d'espace disque, le soin de faire eux-même le ménage de leurs fichiers. Il existe, pour ce faire, un utilitaire du système, BACKUP, dont l'usage est décrit dans une note technique parue dans ce même Info-CI Nº 8.
Par contre les comptes qui sont arrivés à échéance, sont, eux, systématiquement archivés sur bande magnétique avant d'être supprimés. Cette opération a lieu à la fin de chaque mois. A ce propos, lorsqu'un compte n'est plus utilisé, à la suite d'un départ par exemple, son titulaire ou son responsable devrait en informer le Centre Informatique sans tarder.

• Sécurite: Tout le monde a eu connaissance de l'affaire de contamination de réseaux d'ordinateurs par un "virus", qui a récement défrayé la chronique. Ce "virus" a heureusement été intercepté à temps sur celui de nos systèmes où il était arrivé, grâce à la vigilance de notre chef d'exploitation. C'est l'occasion de rappeler quelques **règles de sécurité élémentaires.** Chaque compte est en principe réservé au seul usage de son titulaire, qui est responsable de l'utilisation qui peut en être faite. Le Centre informatique se réserve la possibilité de bloquer les comptes qui seraient utilisés par plusieurs personnes sans son assentiment. D'autre part, nous invitons les utilisateurs à changer fréquement leur mot de passe, et à ne l'inscrire que dans un coin de leur mémoire. La limite de validité des mots de passe est de 45 jours. Elle pourait être raccourcie si celà s'avérait nécessaire.

Cours d'informatique du CI: description

Introduction au système VAX/VMS

(Enseignant: Jacques Guélat).

Ce cours est destiné aux personnes n'ayant jamais (ou presque) touché à un système VAX/VMS, mais possédant des notions d'un autre système d'exploitation (micro ou mainframe). Il est suffisant pour assurer à l'auditeur la possibilité d'une utilisation autonome du VAX-Cluster de l'UNIL. La matière couverte est approximativement celle qui se trouve dans le guide "Introduction to VAX/VMS" de la documentation VAX, avec des compléments sur les particularités de l'installation à l'UNIL, à savoir:

- Terminaux et branchement au VAXCluster.
- Commandes DCL.
- Fichiers et répertoires.
- Utilitaires de communication.
- Procédures de commandes.
- Développement de programmes.
- Bibliothèques et programmes d'utilité générale.

Introduction au système de gestion de base de données INGRES

*(E*nseignant: Abdelali Guerid).

Ce cours s'adresse à toute personne désireuse de connaître un système de gestion de bases de données relationnelles. Ce cours présente le système INGRES et ses interfaces utilisateurs, le langage d'interrogation SQL ainsi que les outils de développement.

L'organisation du cours est la suivante:

- premier jour: présentation de INGRES, définition, chargement et interrogation d'une table, langage SQL.
- deuxième jour: génération d'écrans, de rapports, développement d'une application en OSL et notions sur la gestion de la base.

Des connaissances élémentaires du système VAX/VMS sont requises.

Introduction au système de gestion de base de données documentaires BASIS.

(*E*nseignant: Abdelali Guerid).

Ce cours s'adresse à toute personne confrontée ou interressée par la gestion de textes (exemple: gestion de bibliographie). Ce cours présente le système BASIS, la définition et l'interrogation de la base, les différents modes de chargement ainsi que la mise à jour.

L'organisation du cours est la suivante:

- premier jour: présentation du système BASIS, recherche et affichage de documents.
- deuxième jour: définition d'une base, chargement et mise à jour.

Des connaissances élémentaires du système VAX/VMS sont requises.

Cours d'introduction Macintosh

(Enseignant: Marie-France Pernet).

Ce cours est destiné en priorité à toute personne n'ayant aucune connaissance du Macintosh mais désirant acquérir les bases nécessaires à un emploi judicieux de ce type de micro-ordinateur. La seule condition requise pour suivre ce cours est d'avoir une certaine pratique de la dactylographie.

Cours d'introduction au traitement de texte WORD

(Enseignant: Marie-France Pernet).

Ce cours est destiné à toute personne désirant connaître les bases du traitement de texte WORD. Après cette introduction, chaque participant doit être capable de taper des textes et de les mettre en forme. Il doit également pouvoir effectuer de la correspondance simple.

Cours de traitement de texte WORD avancé

(Enseignant: Marie-France Pernet).

Ce cours est particulièrement recommandé à toute personne ayant déjà acquis les connaissances élémentaires de WORD. Son but essentiel consiste à approfondir les fonctions avancées du traitement de texte, à savoir :

- le Mailing
- la Césure
- la Définition d'entêtes ou pieds de page
- la Création de tables des matières, d'index
- la Définition de styles et leur utilisation
- l'utilisation de la Division

  etc...

Introduction à l'émulation de terminal sur Macintosh

(Enseignant : Philippe Ryter).

Ce cours est recommandé à toute personne désirant utiliser les ressources d'un ordinateur central à l'aide d'un Macintosh. La matinée est consacrée à la présentation des logiciels Mac240 et VersaTerm-PRO et l'après-midi aux exercices. L'accent est mis avant tout sur les procédures de transfert de fichiers.
Des connaissances élémentaires du Macintosh et du système VAX/VMS sont requises.

Introduction à HyperCard

(Enseignant : Philippe Ryter).

Ce cours s'adresse à toute personne désirant s'initier aux méthodes permettant de gérer toutes sortes d'informations à l'aide de cet étonnant logiciel. Le concept de base du produit ainsi que des éléments du language HyperTalk seront présentés en matinée. L'après-midi sera consacré à un exercice dirigé permettant à chaque participant de créer une petite base de données.
Des connaissances élémentaires en programmation sont souhaitées, mais elles ne sont pas indispensables pour suivre ce cours.

VAXcluster ULYS: ressources consommées

Nous donnons sous forme de graphiques, les différentes ressources globales consommées sur le VAXcluster ULYS en 1988:

Utilisation par mois des deux processeurs du VAXcluster

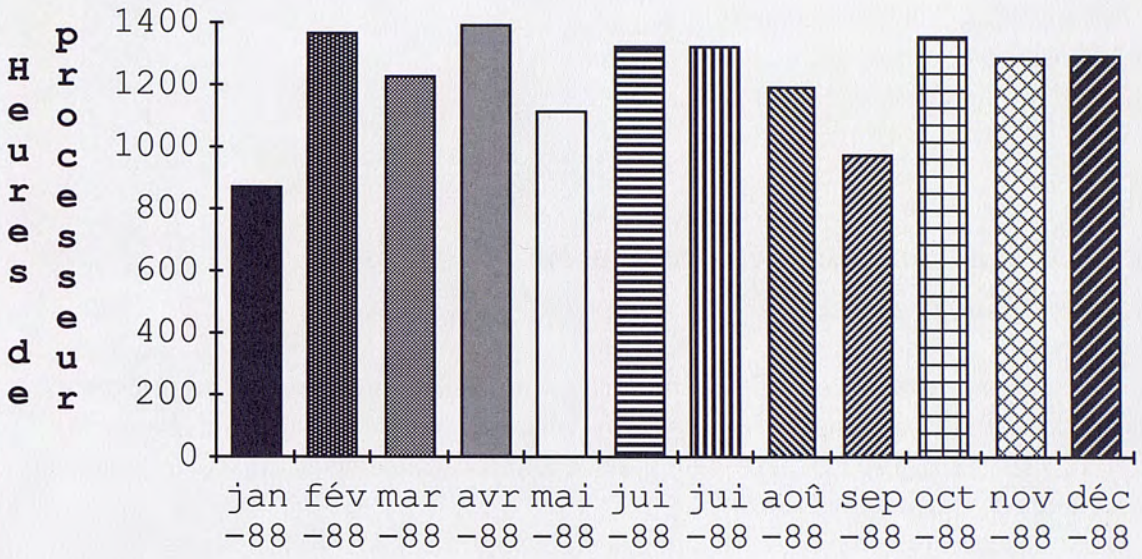


figure 1. Nombre d'heures de processeurs par mois

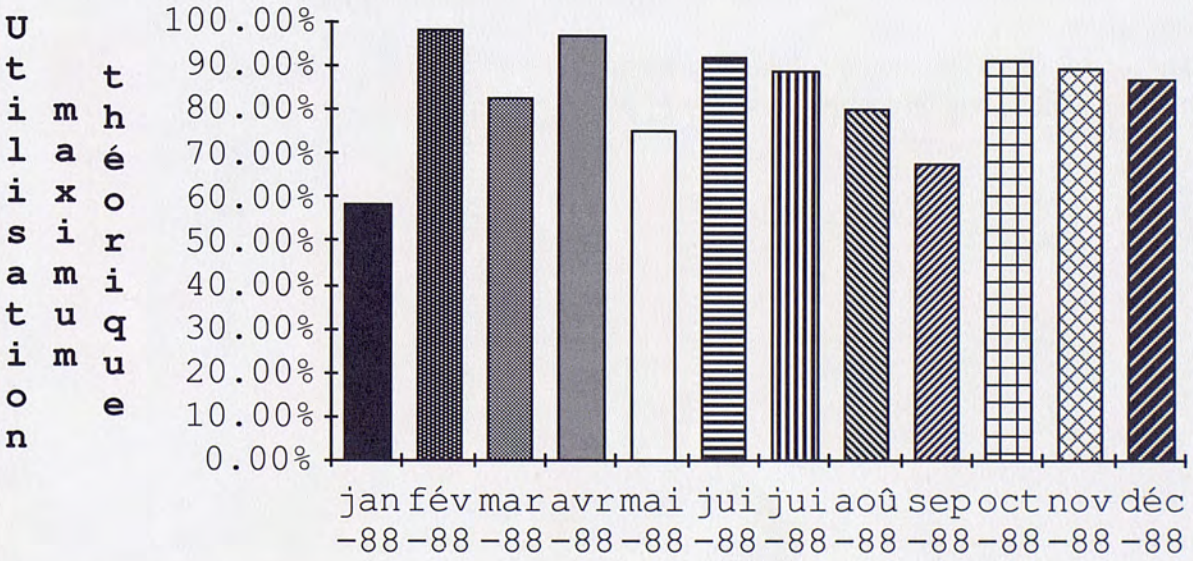


*figure 2. Utilisation relative au maximum théorique de 2*24*nb jours dans le mois*

Temps de connexion par mois sur le VAXcluster

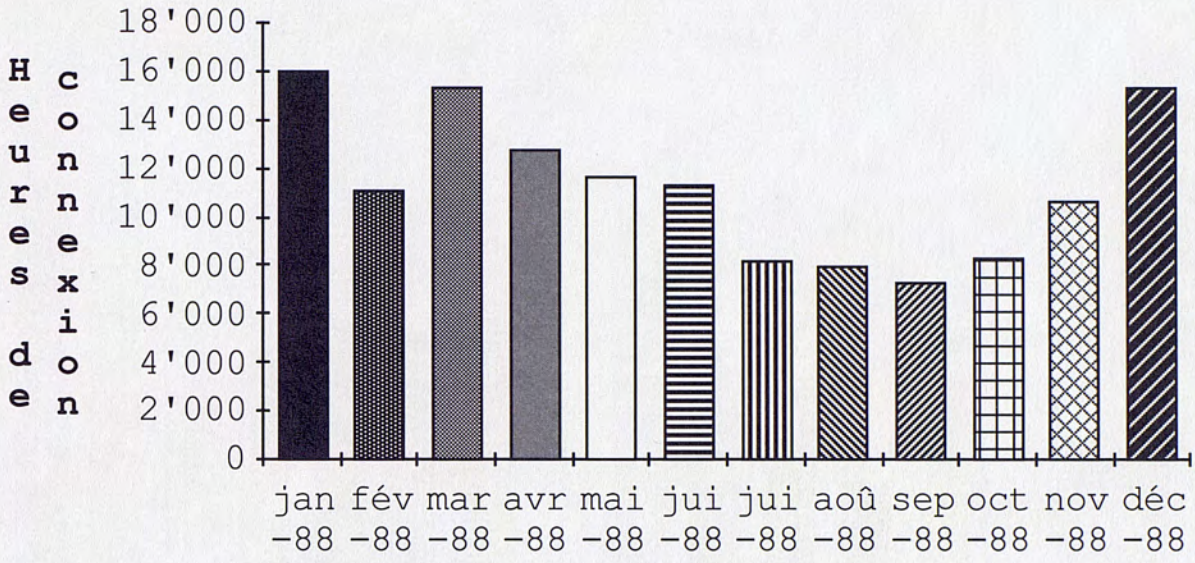


figure 3. Nombre d'heures de connexion par mois

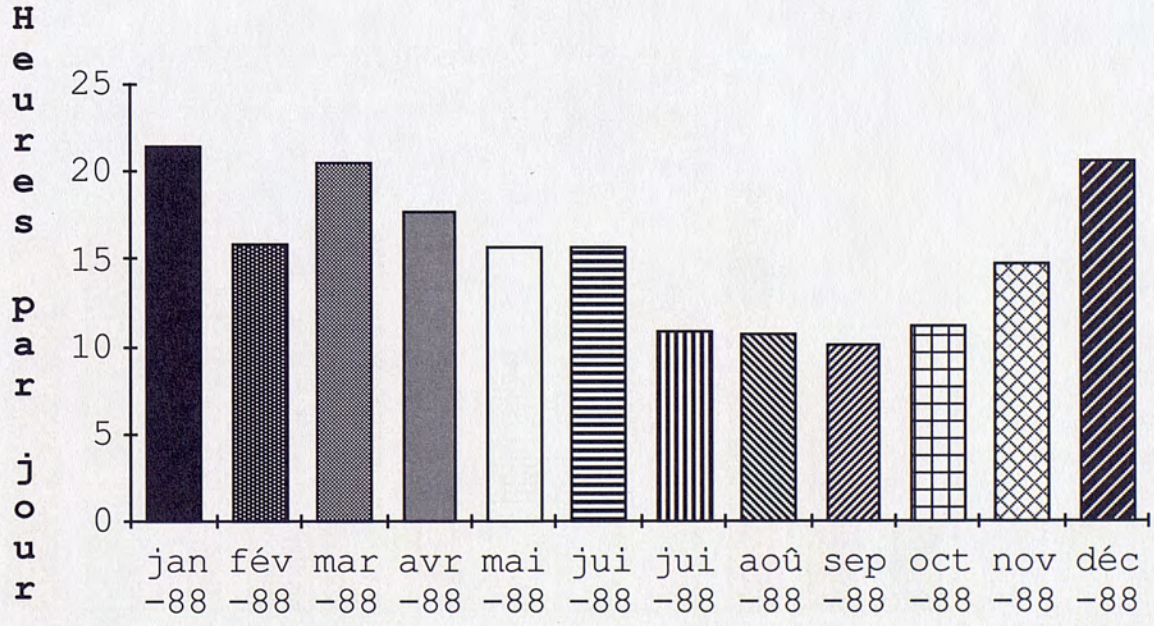


figure 4. Nombre moyen d'heures de connexion par jour

Entrées-sorties

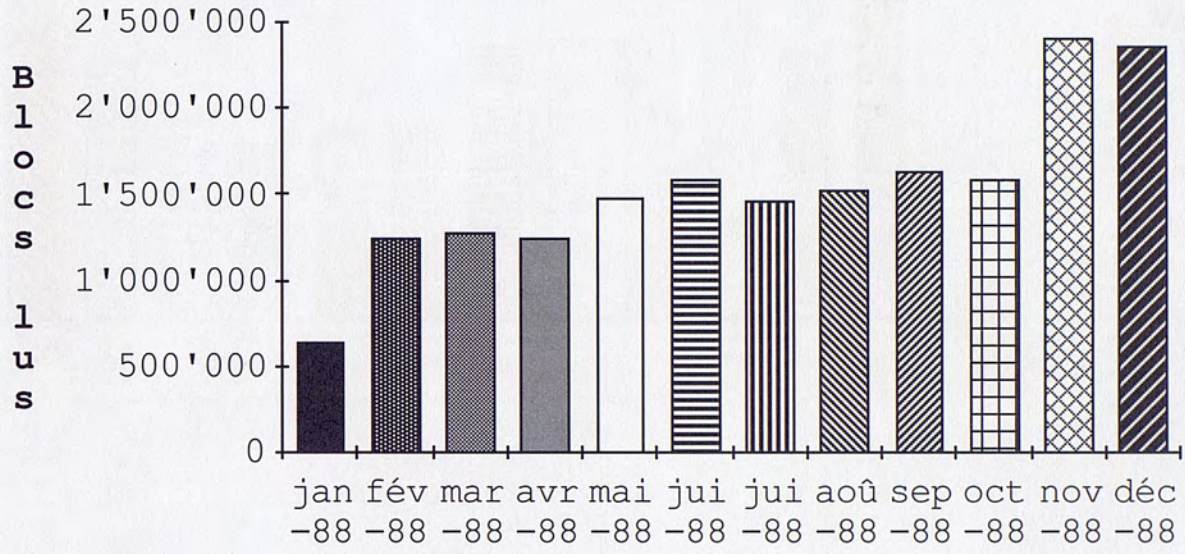


figure 5. Nombre de blocs lus par mois

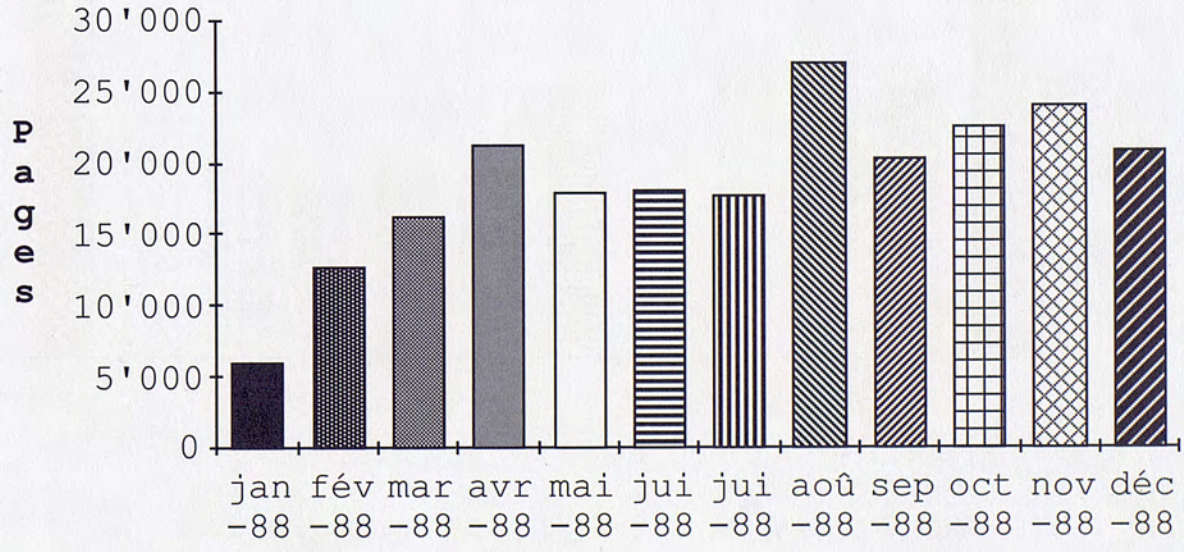


figure 6. Pages imprimées par mois

Enrées-sorties

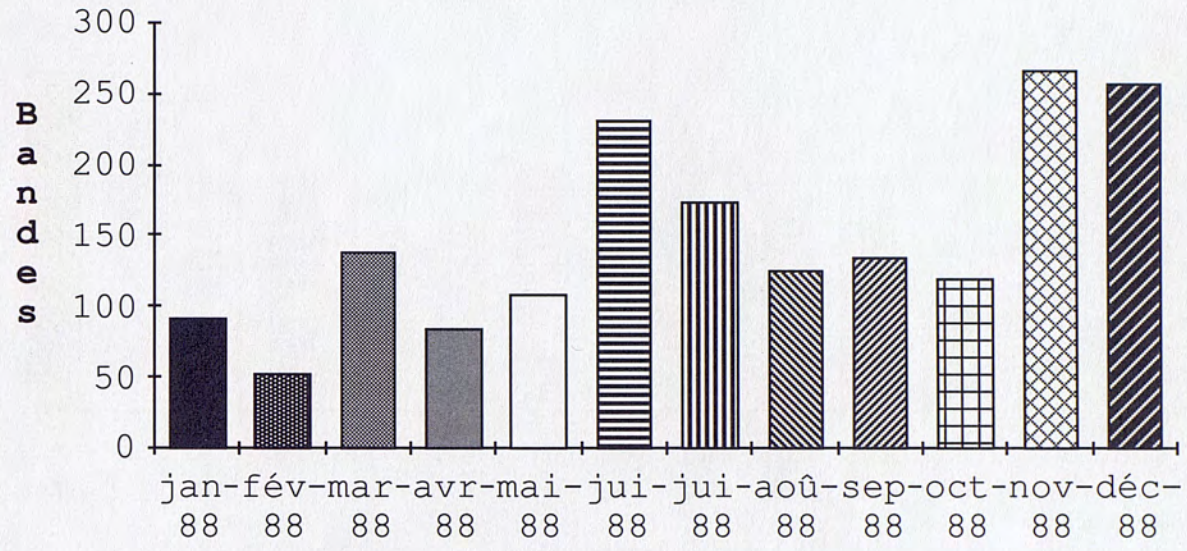


figure 7. Nombre de bandes magnétiques montées par mois

Sessions

figure 8. Nombre de sessions par mois

Sessions

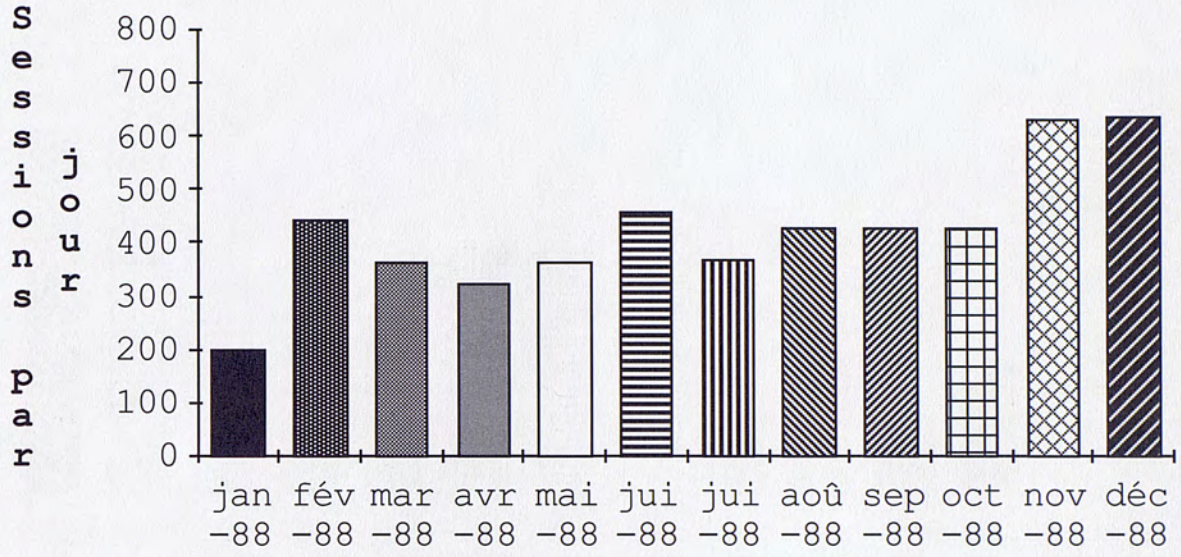


figure 9. Nombre moyen de sessions par jour

Situation en début de 1989

A la veille de la réfection de la climatisation de la salle des machines, un coup d'oeil rapide montre que le noeud ULA (respectivement ULB) fonctionne sans interruption depuis 129 jours (respectivement 139 jours), avec un taux d'inactivité inférieur à 4% (resp. 3%).

Nous finirons ces quelques remarques en rappelant que le programme de comptabilisation des ressources qui a permis la réalisation de ces figures a été écrit par M. B. Gabioud; qu'il en soit remercié.

Ce numéro d'Info-Ci a été tiré à 550 exemplaires

UNIVERSITE DE LAUSANNE **Centre informatique**

COURS ORGANISES PAR LE CENTRE INFORMATIQUE, semestre d'été 1989

Cours	*Date*	*Lieu*	*Enseignant*	*Inscription*
Introduction au VAX/VMS	18-19/04/89	BFSH1 salles nº126 (18/04/89), CP salle 201 nº121 (19/04/89)	J. Guélat	692.23.10
Introduction au VAX/VMS	02-03/05/89	BFSH1 salles nº126 (02/05/89), CP salle 201 nº121 (03/05/89)	J. Guélat	692.23.10
** *Macintosh: Introduction*	*19/04/89*	*BFSH2 salle nº4120*	*M.-F. Pernet*	*692.23.05*
** *Macintosh:* ***Word avancé***	*03/05/89*	*BFSH2 salle nº4120*	*M.-F. Pernet*	*692.23.05*
Macintosh: Introduction	26/04/89	BFSH2 salle nº4120	M.-F. Pernet	692.23.05
Macintosh: **Word avancé**	10/05/89	BFSH2 salle nº4120	M.-F. Pernet	692.23.05
Macintosh: émulateurs	17/05/89	BFSH2 salle nº4120	P. Ryter	692.23.02
Macintosh: **Hypercard**	24/05/89	BFSH2 salle nº4120	P. Ryter	692.23.02
Graphique GKS	20/04/89	CP salle nº201	E. Niebür P. Küffer	692.23.04
Base de données BASIS	05-06/06/89	CP salle nº201	A. Guerid	692.23.15
Base de données INGRES	12-13/06/89	CP salle nº201	A. Guerid	692.23.15

** *en priorité pour le personnel administratif*

Remarque: *Les cours ont lieu de 9H à 12H et de 14H à 17H. Ils s'adressent aux enseignants et au personnel de l'UNIL; les étudiants n'y ont pas accès. Ces derniers sont priés de consulter les calendriers des cours donnés dans leurs Facultés respectives.*

Un club Macintosh à l'UNIL ?

Dans le but de permettre un échange d'informations et d'expériences entre utilisateurs de Macintosh, le Centre informatique prévoit la création d'un groupe micro-informatique à l'UNIL. Le calendrier de ces rencontres mensuelles sera fixé dès réception d'un nombre suffisant de réponses.

Les personnes intéressées par ce projet peuvent m'envoyer le coupon-réponse ci-dessous à l'adresse suivante :

Ph. Ryter, Centre informatique, Bâtiment des sciences physiques (BSP), 1015 Lausanne-Dorigny

---

GROUPE MACINTOSH A L'UNIVERSITE DE LAUSANNE

FICHE D'INSCRIPTION

Nom : .....

Prénom : .....

Institut ou département : .....

Fonction : .....

Intérêts particuliers en micro-informatique : .....

.....

Remarques éventuelles : .....

Cours sur des logiciels graphiques

En octobre 1988, deux cours étaient donnés, l'un sur UNIGRAPH, l'autre sur UNIGKS. Le CI tâche organise des cours qui portent sur des logiciels graphiques d'intérêt général qui sont installés sur des ordinateurs centraux de l'Université. Normalement, ces cours comprennent deux heures d'enseignement *ex cathédra* et deux heures d'exercices sur ordinateur. Pour planifier au mieux les futurs cours à donner, les intéressés sont priés de renvoyer le talon ci-dessous au soussigné. Rappelons qu'une brève description des différentes modules graphiques a été donnée dans INFO-CI nº4.

A renvoyer à: Ernst Niebur, Centre informatique, BSP 1015 Lausanne

Cours sur logiciels graphiques

NOM: **Prénom:**

Institut (adresse complète):

Numéro téléphone:

Adresse DECNET (par ex. ULYS::ENIEBUR):

Je serais intéressé(e) à suivre un cours sur le logiciel suivant (mettez une croix):

( ) UNIMAP
( ) UNIGRAPH
( ) UNIEDIT
( ) UNIGKS
( ) RASPAK

Je préfère que ce cours soit donné

( ) à la fin du semestre d'hiver 88/89
( ) au début du semestre d'été 89

UNIVERSITE DE LAUSANNE

Centre Informatique

Info-Ci

Annexes techniques

Numéro : 9

Parution : Mars 1989

UNIVERSITÉ DE LAUSANNE

CENTRE INFORMATIQUE
BUREAUTIQUE MACINTOSH
M-F. PERNET
692.23.05

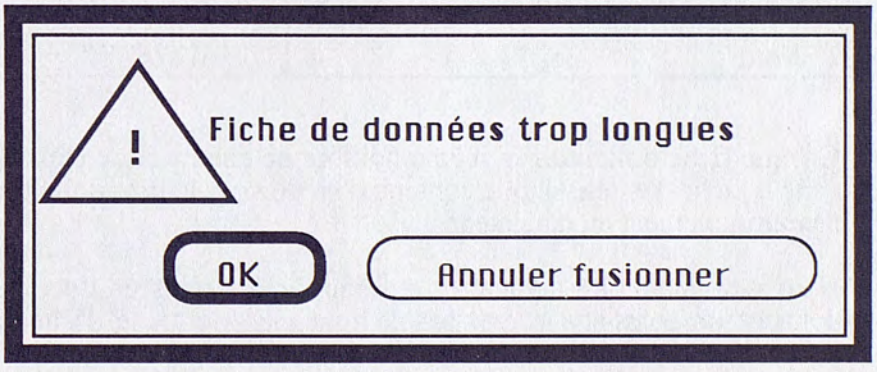
Note à l'intention des utilisateurs WORD concernés par la création de lettres standard ou plus couramment appelé Mailing

Rappel

Les lettres standard permettent de créer et d'imprimer des exemplaires personnalisés pour chaque destinataire. Pour accomplir cette tâche, vous devez préalablement préparer :

- un **document principal** contenant le corps de la lettre standard
- un **document de données variables** contenant spécifiquement les informations relatives à chaque exemplaire et qui seront fusionnées dans le document principal de la lettre standard.

Comme vous l'aviez peut être constaté vous-même, lors du fusionnement de ces deux documents, le message d'erreur suivant venait parfois s'afficher sur l'écran :



Celui-ci est provoqué pour la plupart du temps par le fait que le contenu d'un ou plusieurs de vos champs comportent une ***virgule***, un ***point-virgule*** ou encore un ***tabulateur***.

Après de multiples essais, la procédure à suivre pour la préparation de votre fichier de données variables, qui peut être créé par d'autres applications que Word telles que FileMaker II, Excel, etc., est la suivante :

Composition d'un document de données variables

- un paragraphe dit d'entête contenant la liste des noms des champs
- des fiches de données contenant le texte variable; chaque fiche représente en fait un exemplaire de lettre standard.

Création d'un document de données variables :

1. Tapez la fiche d'entête en séparant les noms des champs par des tabulateurs et abaissez la touche ***RETOUR*** à la fin de l'entête.

2. Tapez les fiches en séparant le texte variable de chaque champ par un tabulateur et terminez par la touche ***RETOUR*** à la fin de chaque fiche. Chaque texte variable doit être introduit dans l'ordre d'apparition des noms des champs décrits dans la fiche d'entête.

    - La fiche d'entête, ainsi qu'une fiche de données variables peut contenir jusqu'à 256 noms de champs ou champs de texte variable.
    - Les champs comportant des *virgules*, des *points-virgules*, des *tabulateurs* ou des *guillemets* doivent être encadrés par des *guillemets*. Dans le cas des guillemets, si vous désirez les faire imprimer, vous devez les doublez, voir les triplez comme dans l'exemple donné ci-dessous :

Exemples

Touche	*pour imprimer*	*introduire dans le fichier des données*
,	155, ch. des Boveresses	*"155, ch. des Boveresses"*
;	Lausanne; Morges; Nyon	*"Lausanne; Morges; Nyon"*
➡	Lyon Paris Lille	*Lyon ➡ Paris ➡ Lille*
" "	Microsoft "Word" 3.0	*"Microsoft ""Word"" 3.0"*
idem	"Word"	*""""Word""""*

- Chaque fiche doit avoir le même nombre de champs que ceux définis dans la fiche d'entête et si le contenu d'un de vos champs doit être vide, tapez simplement un tabulateur.
- N'utilisez jamais les caractères spéciaux « et » dans un document de données variables et n'insérez pas de ligne vierge entre les fiches.
- L'impression des divers exemplaires de la lettre standard se fait dans l'ordre d'apparition des fiches du document de données variables. Pour les obtenir dans un ordre quelconque, vous devez opérer préalablement un tri en choisissant un ou plusieurs champs comme critère de sélection.

M.I.D.I : Musical Interface Digital Instrument.

L'apparition d'un interface M.I.D.I dans la liste de commande du matériel Apple, à suscité de nombreuses questions concernant la signification et l'utilisation d'une telle extension.
Cet article, en présentant de façon détaillée la norme M.I.D.I et ses conséquences, espère répondre à ces questions et susciter des applications dans certains domaines de recherches nécessitant une production sonore contrôlée.
La signification du terme M.I.D.I., Musical Interface Digital Instrument, indique clairement la vocation musicale de cet outil. A titre d'information, une liste non exhaustive des applications musicales, avec quelques explications, est proposées.
Mais c'est dans les applications non musicales, que réside actuellement, pour la communauté universitaire, le plus grand intérêt du système M.I.D.I.

Interaction informatique-musique

aide à la composition, aide à l' arrangement :
L'intelligence artificielle et les systèmes experts devraient permettre d'intéressants développements, partant du principe que de nombreuses règles peuvent être utilisées, dans un style de musique donné.

édition de partition
La plupart des logiciels séquenceurs, étaient capable de transcrire sous forme de notation musicale standard, les informations MIDI, mais avec des lacunes en interdisant l'usage professionnel. Il semble qu'avec la deuxième génération de ces logiciels d'édition, qui apparaît maintenant sur le marché, tous les critères nécessaires soient réunis.

enseignement assisté par ordinateur
Les perspectives sont immenses, car il existe toute une série de qualités musicales, particulièrement réceptives à entraînement, en particulier "l'oreille ", la lecture à vue et la rythmique. La théorie musicale, notamment l'harmonie, l'arrangement et la composition, vont bénéficier de la faciliter à mettre en oeuvre un orchestre MIDI, plutôt qu'un orchestre de musiciens, pour tester certaines règles ou idées, directement par l'oreille.

synthèse de sons
L'échantillonnage digital (sur le même principe que les Compact-disc), associé avec un logiciel performant de synthèse et d'édition, offre des possibilités quasiment illimitées.

enregistrement multipistes
Le fait d'enregistrer la description de chaque événement musical (quelle note et quand), plutôt que le son lui-même, permet toutes les facilités, sans aucune altérations et pour un prix dérisoire, de l'enregistrement multipistes. Les bons séquenceurs actuels, offrent de 50 à 250 pistes, combinables de n'importe quelle manière les unes avec les autres. La seule limitation, est dictée par les limites de performances de la norme MIDI.

synchronisation sons-images (cinéma,vidéo,diaporama)
Il est possible d'utiliser les signaux de synchronisation du cinéma ou de la vidéo, pour contrôler le déroulement et le tempo de la musique.

éditions de Cd et D.A.T.
Le développement industriel de ces deux techniques, en tout cas de la première, à donné naissance à toute une série de chips performants, et l'on trouve des cartes à enficher sur le MAC II, qui permettent de travailler directement avec les signaux Cd et

D.A.T. L'avenir parait bien proche, dans lequel on pourra éditer ses propres Cd depuis son Mac.

contrôle en studio d'enregistrement

En plus de l'utilisation des moyens précités, les signaux MIDI peuvent être utilisés pour contrôler les effets et la table de mixage, l'ordinateur étant à même de remplacer plusieurs paires de mains

Utilisation non musicales

expériences nécessitant une production sonore

La production contrôlée, dans le temps, en fréquence,volume et composition harmonique, d'un son peut tirer parti des générateurs de sons très performants que l'industrie des instruments de musiques électroniques met à disposition à des prix très favorables, en raison de leurs relativement grande diffusion.

Contrôle dans le temps :

Le micro-ordinateur, via le réseau MIDI, va contrôler de façon précise le début et la fin de la note. En utilisant des logiciels séquenceur, la précision est de l'ordre de 1/3000 seconde. Mais la mesure du temps utilisée est celle de la musique (tempo=nb de temps par minute), ce qui va imposer une mini-gymnastique.

L'autre solution consiste à programmer son application. A ce stade, il sera bénéfique d'utiliser des langages incluant une librairie MIDI. Il existe notamment un Pascal et un Basique, qui sont fournis avec de telle extensions.

Contrôle en fréquence :

La norme MIDI prévoit un code pour chaque note (12 fréquences différentes dans un octave), ce qui pour un usage non musical est un peu limité.Mais elle prévoit aussi une modification de la fréquence de base, sous 14 bits, ce qui permet une précision de 0.6 Hertz minimum (dans l'octave 10000... 20000Hz)

Contrôle en volume :

un mot de 7 bits contrôle le volume, ce qui permet 128 niveaux différents.

Contrôle du timbre (harmoniques).:

un mot de 7 bits permet de sélectionner 128 programmes de sons par synthétiseur. Mais la structure harmonique du son, et les possibilités de la contrôler, dépendent du synthétiseur utilisé.

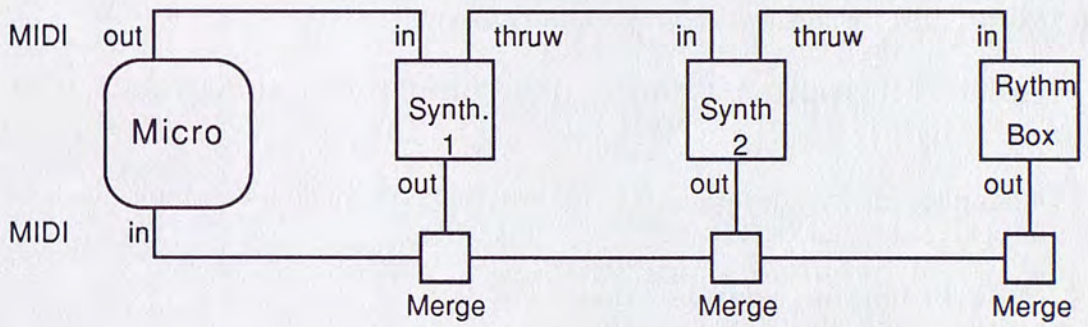
Les synthétiseurs analogiques qui fonctionnent, en combinant des sons purs, ou en filtrant des signaux carrés produits par des oscillateurs, ne permettront pas de contrôler de façon très précise le contenu harmonique, mais ils permettent de générer rapidement des sons très variés pour un prix modique.

les échantillonneurs (où sampler) travaillent avec une représentation numérique du son qui est transformée en signaux analogiques au moyen d'un convertisseur digital-analogique. Les paramètres importants concernent la quantité de mémoire disponible pour la représentation numérique du son, ainsi que le nombre de bits et la fréquence d'échantillonnage. Il est démontré que la bande passante est égale à la moitié de la fréquence d'echantillonnage. Les instruments de musique étant destiné à l'oreille humaine (20 à 20000 Hertz), la fréquence d'echantillonnage ne dépasse pas 40 kHz environ. Pour des raisons de comptabilités avec les Cd (Compact Disc) certains sampleurs travaillent à 44.1 Khz.et généralement sous 16 bits, ce qui leur donne une "qualité sonore" égale à celle du Cd. Généralement ces sampleurs sont aussi pourvu d'un préampli et d'un convertisseur analogique-digital, ce qui leur permet d'échantilloner une source sonore, au travers d'un microphone.

Il existe plusieurs logiciels Macintosh, permettant d'analyser et de traiter ces échantillons numérisés, ou d'en créer de nouveaux, par le calcul ou la représentation graphique, et de les réexporter vers le sampleur. De cette manière, on contrôle parfaitement la composition harmonique.

Remarques :
Le son digital nécessite beaucoup d'information, de l'ordre de 1 mégabyte pour 10 secondes, ce qui entraîne rapidement des problèmes de taille mémoire et de temps de transferts entre micro et instrument. Le disque optique offre une bonne solution pour la mémoire de masse (environ 600 Mbytes), d'autant plus qu'on pourra tirer parti de la comptabilité Cd-Rom et Cd-audio, et là le plus intéressant reste à venir.
Le transfert Micro-instrument est généralement réalisé de 3 manières :
Au travers du réseau M.I.D.I.: L'adressage spécifique à chaque constructeur, prévu par la norme, permet de transmettre toute l'information nécessaire, mais à la vitesse relativement lente de 31.25kbauds
Par l'intermédiaire d'un port sériel, qui peut-être plus rapide.
En connexion directe sur le bus : on trouve actuellement quelques cartes extrêmement performantes, qui remplissent les fonctions d'un sampleur, et qui se connectent sur le bus du Mac II. C'est une excellente solution pour des applications sophistiquées. On peut alors, sans être gêné par le temps, se livrer à toutes les manipulations souhaitées sur le son.
Programmation
Il existe actuellement pour le Mac, quelques langages de programmation incluant une librairie MIDI (notamment un Pascal et un Basic), permettant de réaliser des applications spécifiques.

Schéma d'un réseau MIDI :



Le type de branchement représenté ici, permet à un ordinateur de contrôler spécifiquement (sur son propre canal) 2 synthétiseurs et une boîte à rythme, piloté par un programme de séquenceur.La sortie Midi thruw redonne le signal tel qu'il est entré en Midi in, et permet de continuer la chaîne. La sortie Midi out délivre le seul signal issu du synthé (lorsque que quelqu'un en joue) . La boîte Midi Merge permet d'éviter les collisions lorsque 2 événements Midi sont générés simultanément depuis 2 synthétiseurs (temps réel) . Durant l'éxécution de la partition préétablie, trois musiciens pourront jouer sur les 2 synthétiseur et la boîte à rythme, et le micro-ordinateur sera capable d'enregistrer les ajouts spécifiques, issus de chaque instrument, sur leur piste spécifique. Il y a 16 canaux possibles, permettant de contrôler 16 synthétiseurs indépendants. Les signaux de l'ordinateur et de la boîte à rythme seront synchronisés.Ce type d'installation représente ce que l'on peut faire de plus sophistiqué avec un réseau MIDI.

Descriptif de la norme M.I.D.I

Cette partie technique va décrire les différents messages définis par la norme. Les codes seront donnés en hexadécimal, décimal et binaire. Chaque appareil MIDI , est livré avec un document : " MIDI implémentation chart ", qui décrit exactement son implémentation de la norme, généralement sous forme décimale et binaire. Cette description permettra de mieux cerner les possibilités et limites des systèmes MIDI . Elle permettra également de programmer son application, bien qu'il apparaisse souvent plus rationnel de se référer à une librairie MIDI préexistante

MIDI : Standard simple de communication qui permet d'interconnecter des appareils de musique et des computer sachant communiquer selon cette norme.
En 1982, durant la convention de l' Association Nationale des Manufacturiers de Musique (NAMM), les plus grandes compagnies (E-Mu Systems, Sequentials circuits, Roland, Yamaha, Korg ...) décidèrent d'établir un standard de transmission digitale pour les instruments de musique électronique. L'année suivante, la version 1.0 MIDI apparaissait.
Il s'agit d'un interface sériel asynchrone à 31.25 kilobauds selon un code de 10 bits consistant en 1 start bit, 8 bits de data et 1 stop bit.

On peut distinguer 2 grandes familles dans l'organisation du code MIDI : **Channel Messages et System Messages.**
Les premiers véhiculent toutes les informations spécifiques d'un canal (notamment les notes). **Les system messages** se subdivisent en 3 groupes, **exclusive, common, and real-time.**
System exclusive : chaque constructeur peut créer des messages system-exclusif à sa manière mais doit y inclure son code d'identité spécifique qui lui été attribué, de façon à se distinguer des messages des autres constructeurs. Ceci permet à l'utilisateur de mélanger les constructeurs tout en utilisant les spécificités sans trop de problèmes, par exemple charger des sons depuis un PC, soit sous formes de paramètres destinés à un synthétiseur, soit directement les valeurs échantillonnées (ou calculées) pour un sampleur, à travers une chaîne MIDI.
System Common : Ici les messages sont identiques pour tous les constructeurs, et concernent des actions d'intérêt général comme l'accordage.
Real-time : Les signaux mesurent le temps et assurent la synchronisation.

Channel Information (toutes les commandes spécifiques d'un canal MIDI)

Les 4 bits de poids forts de chaque channel statusbyte codent pour la commande, les 4 bits de poids faibles codent le canal choisi

$9x (144+x) note on pour le canal x
3 bytes :1001nnnn 0kkkkkkk ovvvvvvv

nnnn numéro du canal de 0 à 15 code pour le canal 1à 16

0kkkkkkk: no de la touche de 0 à 127
sur tous les claviers le do moyen = 60 et tous les autres do sont des multiples de 12.La tessiture possible s'étend donc de do =0 jusqu'à sol 10 octaves et demi plus haut .

0vvvvvvv: Velocity (La vitesse de descente de la touche est mesurée et fournit une approximation de la dynamique) de 0 à 127. 1 très léger ,127 très fort 0 note déconnectée, 64 pas de dynamique

$8x (128+x) note off sur le canal x
3 bytes :1000nnnn 0kkkkkkk 0vvvvvvv

0kkkkkkk: No de la touche de 1 à 127

0vvvvvvv: Velocity (Vitesse de remontée de la touche) de 1 à 127
Ce paramètre n'est généralement pas implémenté sur les synthétiseurs (exception Prophet T8 ...)

$AX (160+X) Pression en fin de course (after touch)
3bytes : 1010nnnn 0kkkkkkk 0vvvvvvv
0vvvvvvv : paramètre de pression de 0 à 127

$BX (176+X) Modifications de paramètres
3 bytes : 1011nnnn 0ccccccc 0vvvvvvv
0ccccccc : N0 du paramètre
Seules quelques valeurs de c sont décrites

Service des commandes de mode
124 Omni mode off
125 Omni mode on
126 Mono mode on
127 Poly mode on
0vvvvvvv : paramètres de 0 à 127
Pour les commandes de modes V=0
Pour les commandes à distance : si v=0 commande à distance et si v=127 commande locales

$CX (192+X) Changement du programme des sons
2 bytes : 1100nnnn 0ppppppp
0ppppppp : numéro du programme de 1 à 127

$DX (208+X) pression uniquement en MONO MODE
2 bytes : 1101nnnn 0vvvvvvv
0vvvvvvv : pression de fin de course uniquement en Mono Mode 0 à 127

$EX(224+X) Pitch Wheel
Déplacement de la commande de pitch
3bytes : 1110nnnn 0jjjjjjj 0mmmmmmm
0jjjjjjj 0mmmmmmm de 0,0 à 127,127 (14 bits)
d'abord LSB puis MSB Position minimum = 0,0 médiane = 0,64 max = 127,127

System exclusif (commandes spécifiques à chaque constructeur et appareil)

$F0 (240) system exclusive
Format : 11110000 0iiiiiii 0aaaaaaa 0bbbbbbb 0ccccccc ...etc...
Les données se terminent par n'importe quel byte de status

System realtime (synchronisation en temps réel)

Ces données sont toujours des bytes uniques servant à la synchronisation en temps réels des boîtes à rythmes , séquenceurs etc. Ces bytes peuvent apparaître n'importe quant et notamment entre des bytes de données. Ils sont valables pour tous les canaux et peuvent être ignorés lorsque l'appareil ne les utilise pas.
Un appareil en mode MIDI synchro off fonctionne sur sa propre mesure du temps (horloge interne), en synchro on il utilise la mesure du temps fournie par les signaux de synchronisation qu'émet la machine qui pilote le réseau MIDI.

$f8 (248) Horloge à la lecture (replay)
1 byte 11111000
24 de ces impulsions par noires sont émises à la lecture d'une séquence

$f9 (249) signal de fin d'une séquence de lecture
1 byte 11111001

$FA (250) Start
1 byte 11111010
Apparaît au démarrage d'une séquence.La première impulsion d'horloge suivra dans les 5 ms.

$FB (251) Continuation
1 byte 11111011
Signal émis lorsqu'une séquence interrompue à un certain endroit reprend depuis ce même endroit.
Par exemple après une pause.La prochaine impulsion d'horloge suivra dans les 5 ms.

$FC (252) impulsion de synchronisation
1 byte 11111100
émis durant une pause afin de conserver la mesure du temps

System Common

$F1, $F4, $F5, $F7 ne sont pas définis, et restent disponibles pour de futures développements.

$F2 (242) numéro de la séquence sous 14 bits (Song Position Pointer)
3 bytes : 11110010 0ttttttt 0ttttttt

$F3 (242) numéro de la partition sous 7 bits (Song select)
Correspond à la notion de piste (Track) pour un séquenceur (enregistreur)
2bytes : 11110011 0ppppppp No de partition de 0 à 127

$F6 (246) accordage
1 byte 11110110

System reset

$FF (255) reset du système
1 byte : 11111111
Tous les apparails se retrouvent dans leur état par défauts (état à l'enclenchement)

Les modes.

Il y a 4 modes de fonctionnement possibles pour les appareils MIDI, Omni Mode (on off) se combinant avec Monophonique et Polyphonique, qui sont mutuellement exclusifs.
omni on poly (mode 1): le synthé joue les notes de tous les canaux de façon polyphonique (simultanément...enfin presque)
Omni on mono (mode 2): le synthé joue les notes de tous les canaux de façon monophonique (le synthé joue toujours la dernière note reçue)
Omni off poly (mode 3) : Seul le canal choisi est pris en compte, et les notes sont jouées simultanément
Omni off mono (mode 4) : le synthé joue la dernière note reçue sur le canal choisi.
Le mode 2 semblant d'un intérêt musical très limité, les constructeurs ne proposent souvent que les modes 1,3 et 4.

Pour en savoir plus sur le flux de données MIDI, vous pouvez obtenir une copie de " MIDI Detailed Specification Document " à l'adresse International MIDI Association, 11857 Hartsook St., North Hollywood, CA 91607 . Tel (818) 505-8964

Pierre Küffer
responsable graphique

$FA (250) Start
1 byte 11111010
Apparaît au démarrage d'une séquence.La première impulsion d'horloge suivra dans les 5 ms.

$FB (251) Continuation
1 byte 11111011
Signal émis lorsqu'une séquence interrompue à un certain endroit reprend depuis ce même endroit.
Par exemple après une pause.La prochaine impulsion d'horloge suivra dans les 5 ms.

$FC (252) impulsion de synchronisation
1 byte 11111100
émis durant une pause afin de conserver la mesure du temps

System Common

$F1, $F4, $F5, $F7 ne sont pas définis, et restent disponibles pour de futures développements.

$F2 (242) numéro de la séquence sous 14 bits (Song Position Pointer)
3 bytes : 11110010 0ttttttt 0ttttttt

$F3 (242) numéro de la partition sous 7 bits (Song select)
Correspond à la notion de piste (Track) pour un séquenceur (enregistreur)
2bytes : 11110011 0ppppppp No de partition de 0 à 127

$F6 (246) accordage
1 byte 11110110

System reset

$FF (255) reset du système
1 byte : 11111111
Tous les apparails se retrouvent dans leur état par défauts (état à l'enclenchement)

Les modes.

Il y a 4 modes de fonctionnement possibles pour les appareils MIDI, Omni Mode (on off) se combinant avec Monophonique et Polyphonique, qui sont mutuellement exclusifs.
omni on poly (mode 1): le synthé joue les notes de tous les canaux de façon polyphonique (simultanément...enfin presque)
Omni on mono (mode 2): le synthé joue les notes de tous les canaux de façon monophonique (le synthé joue toujours la dernière note reçue)
Omni off poly (mode 3) : Seul le canal choisi est pris en compte, et les notes sont jouées simultanément
Omni off mono (mode 4) : le synthé joue la dernière note reçue sur le canal choisi.
Le mode 2 semblant d'un intérêt musical très limité, les constructeurs ne proposent souvent que les modes 1,3 et 4.

Pour en savoir plus sur le flux de données MIDI, vous pouvez obtenir une copie de " MIDI Detailed Specification Document " à l'adresse International MIDI Association, 11857 Hartsook St., North Hollywood, CA 91607 . Tel (818) 505-8964

Pierre Küffer
responsable graphique

Sortie de graphique UNIRAS sur LaserWriter

Les utilisateurs de l'UNIL disposent de logiciels graphiques très puissants de la maison UNIRAS (voir INFO-CI nº5 et nº 7). Ces logiciels peuvent utiliser une multitude de périphériques (imprimantes, tables traçantes, ...) pour sortir les graphiques, ceci aussi bien en noir/blanc qu'en couleur. Les utilisateurs qui ont des besoins spécifiques sont renvoyés à la documentation spécialisée de la maison UNIRAS, disponible dans tous les sites de l'UNIL.
Cependant, il nous semble utile de décrire le moyen d'accès à l'imprimante graphique la plus utilisée à l'UNIL, à savoir la **LaserWriter** de Apple.
Mutatis mutandis, la procédure présentée peut servir pour accéder aux autres imprimantes qui utilisent le langage PostScript.

Dans la première partie, nous décrivons comment on obtient du code PostScript à partir des logiciels UNIRAS, dans la deuxième partie, comment on transfère ce code sur l'imprimante.

1. Génération de code PostScript par des produits UNIRAS

1.1 Modules programmables (sauf UNIGKS)

On choisit le "driver" PostScript en appelant, avant toute utilisation de routine UNIRAS, la routine GROUTE avec les paramètres appropriés:

```
CALL GROUTE('sel mpost;ex')
CALL GOPEN
.
.(appel des routines UNIRAS)
.
```

Les résultats sont alors écrits dans un fichier `POST.DAT` qui est créé dans le répertoire courant.

1.2 UNIGKS et modules intéractifs

Dans le but de standardiser les procédures pour les autres logiciels, nous suggérons pour tous ces produits l'utilisation des fichiers UNIPICT. Notons cependant qu'il est possible de créer directement des fichiers PostScript dans les modules intéractifs: `(UNIMAP, UNIEDIT, UNIGRAPH)`.

1.2.1 UNIGRAPH

Le point 12 du menu principal `(MODIFY MENU)` écrit l'image produite dans un segment d'un fichier `UNIPICT.DAT`. Le numéro de ce segment peut être choisi entre 1 et 99.

1.2.2 UNIEDIT

Le menu `HARDCOPY` écrit l'image produite dans le segment 1 d'un fichier `UNIPICT.DAT`.

1.2.3 UNIMAP

Il faut descendre dans l'arborescence comme suit:

```
FILES
PICTURE
CREATE
```

Le "prompt "est maintenant:
`Element number>`

et on donne le numéro du segment (choisi entre 1 et 99) dans le fichier `UNIPICT.DAT` comme réponse. La prochaine fois que l'on donne ensuite la commande `PLOT`, le résultat est écrit dans le segment désigné ainsi du fichier `UNIPICT.DAT`.

1.2.4 UNIGKS

Il faut appeler la routine:

```
GOPWK(wkid,conid,4692)
```

qui ouvre la station de travail, `'wkid'` étant l'identificateur de la station en question, lequel est un entier, qui doit être différent des valeurs des autres stations ouvertes simultanément.

Le paramètre 'conid', également entier, peut être choisi librement entre 0 et 99, mais, de nouveau, il est préférable de ne pas choisir les mêmes 'conid' pour plusieurs stations. C'est, de nouveau, le numéro de segment dans le fichier `UNIPICT`. (ATTENTION: Ne pas confondre les segments `UNIPICT` avec des segments GKS, ce sont deux concepts entièrement distincts!)

Comme toujours en GKS, il ne suffit pas d'ouvrir la station (avec `GOPWK`), mais il faut l'activer (`donc GACWK(wkid)`) et il vaut mieux dé la désactiver (GDACWK(wkid)) et dé la fermer (GCLWK(wkid)) avant la fin du programme (voir le manuel GKS).

1.3 Transformation UNIPICT ==> PostScript

Pour les modules décrites sous 1.2, il faut alors transformer le fichier `UNIPICT` en format PostScript. Nous nous servons de la routine `Rascon` pour faire ceci (voir INFO-CI nº7). Alors, après la fin de l'exécution du module intéractif ou du programme qui contient des appels `UNIGKS`, on tape au niveau VMS:

```
$ Rascon
```

Ce programme reprend le fichier `UNIPICT.DAT` qui a été créé par les procédures décrites précédamment. Le "prompt" devient:

```
GROUTE>
```

On choisit le "driver" PostScript par les commandes:

```
GROUTE>sel MPOST
GROUTE>exit
```

Le programme `Rascon` pose alors une série de questions, qui permettent de varier encore la taille de l'image, de superposer plusieurs images, etc... Pour des applications standards, on peut répondre par des `<CR>` (valeurs par default) a toutes ces questions (une dizaine), tout en faisant attention à répondre avec le même numéro de segment que l'on avait choisi précédemment lors de la création de l'image.
En donnant comme réponse des `<CR>`, on aboutit automatiquement au niveau VMS.

Le Rascon créé un fichier `POST.DAT`, qui est en PostScript.

2. Transfert vers le Macintosh et impression sur la LaserWriter

Etant donné que le fichier `POST.DAT` ne contient que des caractères ASCII, le transfert de ce fichier vers le Mac se fait comme d'habitude (voir INFO-CI nº7):

```
$Run UNIL$library:vaxtomac post.dat
```

qui crée le fichier '`post.mact`'. Ce dernier peut être transféré vers le Mac, par exemple avec `Kermit`. Il est préférable d'écrire le fichier sur le kermit du Mac tout de suite en format de WORD (voir sous Kermit Options). S'il est dans un autre format, il faut encore le transformer en format Word.

A ce stade, il faut éventuellement profiter pour détruire les fichiers créés sur le VAX, dont `Post.dat et Post.mact`. (le fichier `UNIPICT.DAT` est d'utilité générale, mais pour des applications simples il peut être effacé également à ce moment). Tous ces fichiers se trouvent dans le répertoire courant.

Ensuite il faut ouvrir le fichier créé sur le Mac par Word et mettre tout le texte en style 'PostScript' (dans le menu 'Définir les styles'). Noter que ce style ne figure pas par default dans la liste des styles, mais il est connu par Word: Si on le tape dans la fenêtre 'Style', il apparait!

Finalement, on imprime le texte comme habituellement avec le Mac.

Ernst Niebur / CI

UTILISATION DE LA PASSERELLE TCP/IP-DECNET

En raison du nombre de plus en plus grand de machines tournant avec un système d'exploitation UNIX installées sur le réseau de l'Université de Lausanne, le Centre Informatique a installé une passerelle permettant de mettre en communication des machines UNIX et VMS. Cette passerelle consiste actuellement en une machine DEC de type MicroVAX II tournant le système d'exploitation ULTRIX et un programme spécialisé appelé DECNET-ULTRIX. Cette situation est provisoire et sera remplacée dans le courant de l'année 1989 par l'installation d'un logiciel TCP/IP sur le cluster ULYS directement.

Les trois applications TCP/IP suivantes sont actuellement supportées :

TELNET	:	**Etablissement d'une session interactive sur une autre machine du réseau.**
FTP	:	**Transferts de fichiers entre deux machines.**
MAIL/SMTP	:	**Envois de messages sur d'autres machines.**

Pour pouvoir utiliser les commandes décrites dans ce document, vous devez disposer d'une machine UNIX tournant un logiciel de communication TCP/IP et cette machine doit avoir été connectée sur le réseau Ethernet de l'université par les soins du Centre Informatique exclusivement; ce dernier vous attribuera également une adresse TCP/IP, unique à l'échelle du monde entier, et que vous ne devrez changer sous aucun prétexte ! D'autre part, le fichier des machines connues sur le réseau (généralement /etc/hosts sur les machines UNIX) doit contenir au minimum l'adresse de votre machine et celle de la passerelle DECNET-TCP/IP du Centre Informatique.

Remarques :
Dans les exemples ci-dessous, les commandes tapées par l'utilisateur seront en caratères gras et les caractères de contrôle seront représentés à l'aide du symbole <CTRL> (ex: <CTRL_D>). Soit également un username MDUPONT avec un password TULIPE défini sur les deux machines ULYS et SUN1.

Le système d'exploitation UNIX ne considère pas les caractères minuscules et majuscules comme équivalents. Il donc est important de donner les commandes sur les machines UNIX en respectant les deux types de caractères.

Prenons comme exemple la figure 1 ci-dessous afin d'illustrer les points suivants :

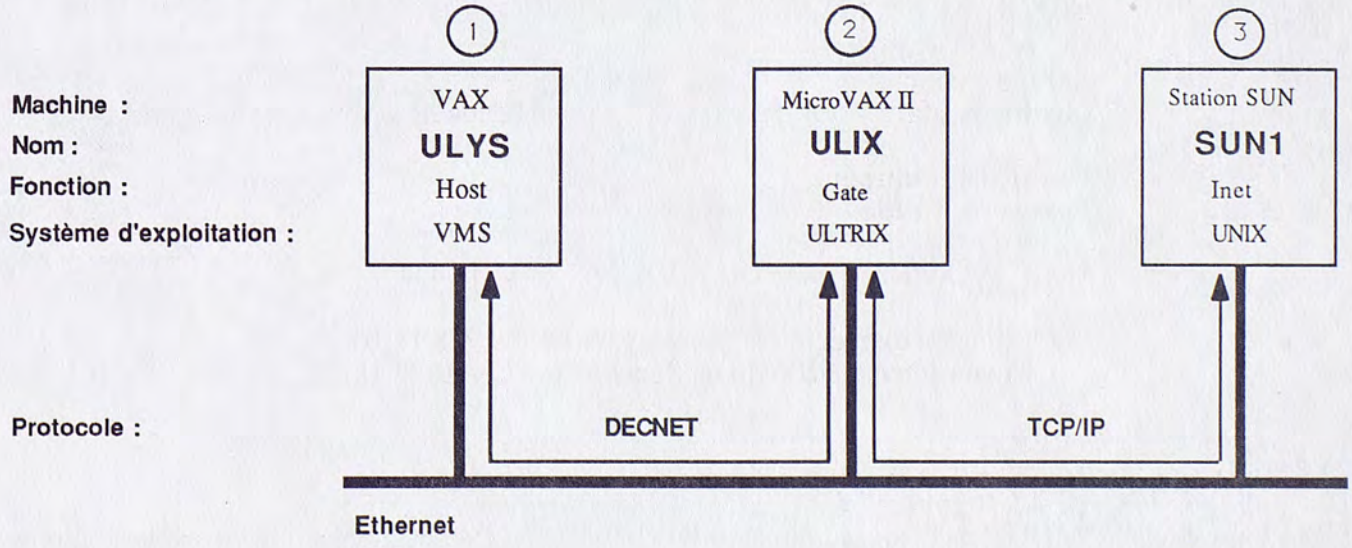


Fig. 1

On entend par remote login, l'établissement d'une session interactive sur une autre machine. Il n'est pas possible de rendre la passerelle ULIX transparente lorsqu'on réalise une connection de ce type; cette opération doit se dérouler en deux temps :

1.1 Connection depuis machine VMS vers une machine UNIX

```
ULA$ set host ulix              {connection de 1 vers 2 selon figure}

(ulix)

login: sun1!                    {connection de 2 vers 3 selon figure}
Trying...
Connected to sun1.
Escape character is '^]'.

4.2 BSD UNIX (sun1)

login: mdupont
Password: tulipe
Last login: Tue Nov  8 11:28:38 from 128.178.173.3
Sun UNIX 4.2 Release 3.5EXPORT (SUN1) #1: Tue Nov 8 08:17:06 PST 1988
Ok
sun1%  .....

Travail sur la station UNIX....

sun1% exit
sun1% logout
Connection closed by foreign host.
%REM-S-END, control returned to node _ULA::
 ULA$
```

1.2 Connection depuis machine UNIX vers une machine VMS

```
sun1% telnet ulix               {connection de 3 vers 2 selon figure}
Trying 128.178.172.1 ...
Connected to ulix.
Escape character is '^]'.
ulix login: ula::               {connection de 2 vers 1 selon figure}

Username: mdupont
Password: tulipe

UNIL VAXcluster, node ULA, VAX 8550, VMS 4.7

   Last interactive login on Thursday,  8-DEC-1988 17:10
   Last non-interactive login on Sunday,  6-NOV-1988 18:24
ULA$

Travail sur le systeme VMS...
ULA$ logout
 MDUPONT  logged out at  8-DEC-1988 17:12:49.60Connection closed by foreign
host.
sun1%
```

2.- TRANSFERT DE FICHIERS

Pour réaliser un transfert de fichier, on utilise la commande **COPY** sous VMS et la commande **ftp** sous UNIX.

Le transfert de fichier est possible dans les deux sens indépendamment de la machine sur laquelle on travaille; ceci signifie que depuis une machine VMS on peut copier un fichier vers ou depuis une machine UNIX et que depuis une machine UNIX on peut copier un fichier vers ou depuis une machine VMS. Sur la machine VMS il suffit de changer l'ordre des arguments de la commande COPY et sur la machine UNIX, il faut utiliser soit le PUT soit le GET de FTP suivant le sens du transfert.

2.1 Transfert d'un fichier en étant connecté sur une machine VMS.

L'exemple ci-dessous illustre un transfert de fichier dans le sens VMS --> UNIX.

Syntaxe:
COPY local_file gate"username@inet password"::"remote_file"

exemple {copie d'un fichier de 1 vers 3} :

ULA$ **copy/log toto.com ulix"mdupont@sun1 tulipe"::"toto"**
%COPY-I-CREATEDSTM, ULIX"mdupont@sun1 password"::"toto" has been created in stream format
%COPY-S-COPIED, SYS$COMMON:[SYSMGR]TOTO.COM;43 copied to ULIX"mdupont@sun1 password"::"toto" (47 records)

Il est également possible, en suivant la même syntaxe, d'exécuter les commandes VMS suivantes sur une machine UNIX:

APPEND, DELETE, DIRECTORY et TYPE.

2.2 Transfert d'un fichier en étant connecté sur une machine UNIX.

L'exemple ci-dessous illustre un transfert dans les sens UNIX --> VMS.

exemple :

sun1 % **ftp ulix** {connection de 3 vers 2 selon figure}
Connected to ulix.
220 ulix FTP server (Version 4.1 Wed Sep 30 20:25:18 EDT 1987) ready.
Name (ulix:mdupont): **ula::mdupont** {connection de 2 vers 1 selon figure}
331 Password required for gateway access ula::mdupont.
Password: **tulipe**
231 Access control info received.
ftp> **put**
(local-file) **toto.unix**
(remote-file) **toto.vms**
200 PORT command okay.
150 Opening data connection for ula/mdupont/password::toto.unix (128.178.172.4,1067).
226 Transfer complete.
local: toto.unix remote: toto.vms
225 bytes sent in 0.03 seconds (7.3 Kbytes/s)
ftp> **quit**
221 Goodbye.
sun1 %

> Remarque : lorsque le canal logique a été ouvert par FTP entre la machine UNIX et la machine VMS, il est possible d'utiliser la plupart des commandes habituelles, autres que celles pour le transfert de fichier , de FTP.

3.- ENVOIS DE MESSAGES

3.1 Envois d'un message depuis une machine VMS vers une machine UNIX.

Syntaxe dans le programme mail : to : **gate::"username@inet"**

exemple : {envois d'un message de 1 vers 3 selon figure}

ULA$ **mail**

MAIL> **send**
To: **ulix::"mdupont@sun1"**
Subj: **Essai**
Enter your message below. Press CTRL/Z when complete, or CTRL/C to quit:
Ce message est un message de test !
<CTRL_Z>
MAIL> **exit**
ULA$

3.2 Envois d'un message depuis une machine UNIX vers une machine VMS.

Syntaxe au niveau du shell : **mail username%host.DNET@gate**

exemple : {envois d'un message de 3 vers 1 selon figure}

sun1% **mail mdupont%ula.DNET@ulix**
Subject: **Message de test**
Ce message a ete envoye depuis la machine SUN1 vers la machine ULA
<CTRL_D>
EOT
sun1%

4.- RESUME DES COMMANDES

Depuis VMS	Depuis UNIX
remote login	
$ set host gate login : **inet!**	**% telnet gate** login : **host::**
Depuis VMS	**Depuis UNIX**
Transfert et accès aux fichiers	
gate"inet!username password"::"filename" ou **gate"username@inet password"::"filename"**	**%ftp gate** ..... Name (gate:): **host::username** Password (gate:host::username): ..... ftp>

mail

$mail
MAIL> **send**
to : **gate::"username@inet"**

% mail username%host.DNET@gate

D. Henchoz / CI

Méthodes de sauvegarde des bases de données BASIS et INGRES

1. Introduction.

Dans le numéro précédent de INFO-CI (voir INFO-CI nº8), le Centre informatique décrit la politique de sauvegarde des fichiers utilisateurs. Dans cet article, on examine le cas particulier des bases de données sur VAX: BASIS et INGRES.

2. Sauvegarde des bases BASIS.

Le système BASIS n'a pas d'utilitaire spécialisé pour assurer une sauvegarde interne. En outre, BASIS implémente les fichiers directement dans le répertoire courant de l'utilisateur. La sauvegarde dans ce cas est similaire à celle d'un fichier VMS habituel.

3. Sauvegarde des bases INGRES.

Contrairement à BASIS, INGRES possède du point de vue de la sauvegarde les caractéristiques suivantes:

- gestion des locations des fichiers utilisateurs.
- présence de plusieurs commandes pour la sauvegarde interne soit de tables soit de base entière.

Ces différentes commandes sont récapitulées ci-dessous:

Commande	**Utilisateur autorisé**	**Champs d'action**
COPYDB	DBA et utilisateur final	Tables
UNLOADDB	DBA	Tables et bases
CKPDB	DBA	Bases

Le choix de la méthode approprié est déterminé par le type d'utilisateur et par l'objet à sauvegarder. COPYDB est plus facile à utiliser, mais ne permet de sauvegarder que des tables. Par contre, UNLOADDB est plus complète et permet accessoirement de déplacer des objets d'une base vers une autre, de modifier la base etc... Enfin, CKPDB qui peut être combinée avec du "journaling", fera l'objet d'un article ultérieur.

4. Utilisation de la commande COPYDB.

4.1 Syntaxe

La syntaxe de la commande COPYDB est la suivante:

```
$ COPYDB [-uusername] [-c] [-ddirectory] nom_base {nom_table}
```

où [ ] indique une option et { } indique une répétition éventuelle.

Les différentes options signifient:

- **u**username, pour spécifier l'utilisateur dont on veut sauvegarder les objets.
  Cette option n'est accessible qu'à l'utilisateur dit "SUPER USER".

- **c**, pour sauvegarder en mode texte.

- **d**directory, pour spécifier le répertoire où l'on veut mettre les objets.

- nom_base est la base en question

- nom_table est la table à sauvegarder.

4.2 Déchargement de la table.

La commande COPYDB fonctionne en deux temps. En premier lieu, elle génère les deux fichiers de commandes COPY.OUT et COPY.IN qui permettent respectivement de décharger et de charger la table. En second lieu, l'utilisateur doit effectivement décharger la table.

Le déchargement effectif se fait en tapant (se placer auparavent dans le répertoire de travail):

```
$ ingres nom_base <COPY.OUT
```

Ceci aura pour effet de créer le fichier nom_table.XX contenant les données de la table nom_table et où XX désigne le code interne de l'utilisateur (code attribué lors de la définition de cet utilisateur dans INGRES).

A ce stade, l'utilisateur, pour avoir la sauvegarde de sa table, doit transférer sur bande magnétique les fichiers: COPY.IN, COPY.OUT et nom_table.XX .

4.3 Restauration

Se placer dans le répertoire de travail et y mettre les fichiers sauvés précédemment.

Pour restaurer la table, il faut simplement taper la commande:

```
$ ingres nom_base <COPY.IN
```

5. Utilisation de la commande UNLOADDB.

Le fonctionnement de la commande UNLOADDB est très similaire à celui de COPYDB sauf qu'elle s'applique à toute la base.
Il faut s'assurer au préalable qu'aucun utilisateur n'est connecté à la base en question.

5.1 Syntaxe

La syntaxe de la commande UNLOADDB est la suivante:

```
$ UNLOADDB [-uusername] [-c] [-ddirectory] nom_base
```

où les différentes options ont la même signification que précédemment.

5.2 Déchargement

Dans un premier temps, la commande UNLOADDB génère les fichiers suivants:

- UNLOAD.ING et RELOAD.ING (pour le déchargement/chargement de la base)
- COPYXX.OUT et COPYXX.IN (pour le déchargemnt/chargement des différents objets de la base); XX étant toujours le code d'un utilisateur.

Le déchargement effectif intervient en tapant la commande:

```
$ @UNLOAD.ING
```

Cette commande créèra une série de fichiers correspondant aux tables du catalogue de INGRES.

A ce stade, l'utilisateur, pour avoir la sauvegarde de sa base, doit transférer sur bande magnétique tous les fichiers mentionnées.

5.3 Restauration

De nouveau, il faut se placer dans un répertoire de travail et y mettre les fichiers précédemment sauvegardés.
La restauration de la base se fait alors en tapant la commande:

```
$ @RELOAD.ING
```

6. Conclusions.

Avec ces différentes commandes, tout utilisateur de INGRES ou de BASIS, qu'il soit utilisateur final ou DBA a la possibilité de sauvegarder son travail. Nous ne saurions trop l'encourager à le faire.

A. Guerid / CI

La bibliothèque NAG-FORTRAN: des NOUVEAUTES.

1. Introduction.

Dès novembre 88, la bibliothèque NAG_FORTRAN installée à l'UNIL a fait peau neuve par l'introduction de la **nouvelle version "Mark13"**. D'autre part, elle s'est complétée par l'adjonction de deux nouveaux modules des plus utiles: un **"ON-LINE HELP"**, style HELP de VMS, qui permet d'obtenir à l'écran la majeure partie des manuels NAG-FORTRAN et un **Supplément Graphique** offrant des possibilités de visualiser les données et résultats faisant partie d'une application utilisant éventuellement NAG. Ces deux ensembles rendent la bibliothèque NAG-FORTRAN encore plus complète et conviviale, c'est pourquoi son utilisation est fortement recommandée.

2. La nouvelle version "Mark13" de la bibliothèque.

Des 688 algorithmes numériques contenus dans la version "Mark12" de la bibliothèque NAG-FORTRAN, le 95% a été conservé dans la version 13. Plus précisément, 33 routines obsolètes ont disparu. Leur remplacement est assuré par l'utilisation de routines existantes remplissant, comme cas particulier, les mêmes fonctions, ou par des nouvelles routines plus performantes (cf. tableau 1). Une centaine de nouvelles routines ont été introduites dans la version 13. La nouvelle bibliothèque offre donc maintenant un choix parmi 746 algorithmes numériques. Des nouveautés ou améliorations ont été introduites dans les chapitres suivants (le nombre de droite indique le nombre de nouvelles routines):

• C02:	zéros de polynômes	(1)
• C06:	FFT symétriques et bidimensionnelles	(5)
• D01:	intégrales	(3)
• D02:	problèmes des valeurs initiales dans les EDO	(11)
• D03:	EDP elliptiques	(1)
• E01:	interpolations préservant la monotonicité	(8)
• E02:	ajustement de courbes et surfaces par splines	(3)
• F01:	décomposition QR complexe	(4)
• F02:	décomposition aux valeurs singulières réelles et complexes	(3)
• F04:	systèmes linéaires presque bloc-diagonaux	(3)
• F06:	rotations dans le plan	(30)
• G01:	distributions discrètes	(3)
• G02:	estimation robuste	(4)
• G07:	estimation robuste (nouveau chapitre)	(3)
• G13:	test de diagnostic	(2)
• S:	fonctions spéciales	(7)

Certaines routines ont été améliorées ou corrigées, ce qui peut éventuellement avoir pour conséquence que les résultats produits par ces routines soient légèrement différents. Certains paramètres peuvent avoir changé de signification. Voici la liste des routines dont il faut se méfier:

C05NBF	D01AJF	D01ANF	D02CBF	E04FDF	E04GEF	F02AGF
C05NCF	D01AKF	D01APF	D02CGF	E04GBF	E04HEF	F02AJF
C05PBF	D01ALF	D01AQF	D02CHF	E04GCF	E04HFF	F02AKF
C05PCF	D01AMF	D02CAF	E04FCF	E04GDF	F02AFF	

Routines éliminées	*Remplacement recommandé*
D02QBF	D02QDF
E04CGF	E04JAF
E04DBF	E04DGF
E04DEF	E04KAF
E04DFF	E04KCF
E04EBF	E04LAF
E04UAF	E04UCF
F01CNF	SCOPY/F06EFF
F01CPF	SCOPY/F06EFF
F01CQF	F06FBF
F01CSF	SSPMV/F06PEF
F01DAF	SDOT/F06EAF
F01DBF	X03AAF
F01DCF	CDOTU/F06GAF
F01DDF	X03ABF
M01AFF	M01DAF
M01ABF	M01DAF
M01ACF	M01DBF
M01ADF	M01DBF
M01AEF	M01DEF et M01EAF
M01AAF	M01DEF et M01EAF
M01AGF	M01DFF et M01EBF
M01AHF	M01DFF et M01EBF
M01ALF	M01DBF, M01ZAF et M01CBF
M01AMF	M01DBF, M01ZAF et M01CBF
M01ANF	M01CAF
M01AQF	M01CBF
M01ARF	M01CBF
M01BAF	M01CCF
M01BBF	M01CCF
M01BCF	M01CCF
M01BDF	M01CCF
P01AAF	P01ABF

Tableau 1. Remplacement des routines obsolètes.

Pour ce qui concerne l'utilisation de la bibliothèque NAG-FORTRAN sur le VAX-Cluster de l'UNIL, rien n'a changé par rapport à la version précédente. En effet, l'édition des liens d'un programme utilisant cette bibliothèque se fait toujours avec la commande:

LINK mon_programme, NAG$LINK1/LIBR, NAG$LINK2/LIBR

ou plus succinctement: LINK mon_programme, NAGLIB/OPT

Il peut être utile de rappeler ici que la bibliothèque NAG-FORTRAN est une version double précision, ce qui implique quelques modifications à apporter dans les exemples du manuel (cf. INFO-CI N° 8).

La documentation de la bibliothèque est toujours constituée de sept volumes dont la structure du texte est inchangée. Il faut toutefois noter l'apparition d'un index par mots-clés dans le contexte. L'utilisateur a d'autre part accès à certaines informations dans le répertoire $UNIL0:[NAG.DOC], à savoir les fichiers imprimables suivants:

UN.DOC	"Users' Note": détails d'installation.
ESSINT.DOC	"Essential Introduction to the NAG Fortran Library".
SUMMARY.DOC	Bref résumé des routines.
CALLS.LIS	Liste des routines appelées.
CALLED:LIS	Liste des routines appelantes.

3. Le "ON-LINE HELP": utilitaire d'aide intégré.

Pour pallier à la difficulté qu'éprouvaient certains utilisateurs de la bibliothèque NAG-FORTRAN d'accéder à la documentation (en particulier aux manuels), le Centre Informatique a acquis le module "NAG On-line Information Supplement" qui fonctionne comme le HELP de VMS et qui rend accessible au terminal une partie de l'information contenue dans les manuels. Cette dernière devrait être suffisante pour une utilisation "normale" de la bibliothèque. Cependant cette information disponible au terminal ne remplace pas celle contenue dans les manuels et les utilisateurs devront consulter la documentation de base pour obtenir des détails plus approfondis sur certains algorithmes non triviaux.

Malheureusement, la version actuellement installée du HELP ON-LINE (Mark12) ne correspond pas à celle de la bibliothèque (Mark13), ceci étant dû à des délais du distributeur. La version Mark13 sera installée dès que possible. Comme sus-mentionné, 95% des routines n'ont pas changé d'une version à l'autre, ce qui rend le HELP actuel tout à fait utilisable...

3.1. Utilisation du "ON-LINE HELP".

La bibliothèque d'aide de NAG a été rajoutée aux autres bibliothèques d'aide de VMS de telle manière qu'il faut utiliser la commande suivante pour y accéder:

$ HELP @NAGVMSHELP

On se retrouve alors dans un menu principal où tous les chapitres de la bibliothèque NAG-FORTRAN sont listés ainsi que deux entrées spéciales: "LOCAL" et "HELP".

L'entrée "LOCAL" permet d'obtenir des informations sur l'installation et l'utilisation de la bibliothèque NAG à l'UNIL. Il est vivement recommandé à l'utilisateur débutant de consulter cette information.

L'entrée "HELP" permet d'obtenir le titre de tous les chapitres de la bibliothèque NAG-FORTRAN. On obtient cette liste avec la commande du menu principal:

@NAGVMSHELP Topic? **HELP CHAPTERS**

Une information plus détaillée sur un chapitre particulier (par exemple D02) s'obtient en entrant les commandes:

@NAGVMSHELP Topic? **D02**

@NAGVMSHELP D02 Subtopic? **HELP**

La liste de toutes les routines du chapitre, accompagnées d'une description de leur fonction, est affichée. On peut dès lors faire le choix de l'algorithme qui nous intéresse. La description de la routine choisie (par exemple D02BAF) est obtenue avec la commande:

@NAGVMSHELP D02 Subtopic? **D02BAF**

La partie "1. Fonction" de la description du manuel apparaît à l'écran. Sont alors aussi disponibles les parties:

2. Spécifications:	description de l'appel à la routine et du type des paramètres.
5. Paramètres:	description détaillée de chaque paramètre de la routine, en particulier sa valeur à l'entrée et à la sortie de la routine.
6. Erreurs:	description du code d'erreur (paramètre IFAIL)
Version:	date et version de la bibliothèque auxquelles la routine a été introduite.

4. Le Supplément Graphique NAG.

Le Supplément Graphique NAG (**SG-NAG**) offre à l'utilisateur de la bibliothèque NAG-FORTRAN un moyen d'affichage des données et résultats sous forme graphique tout en restant dans le formalisme des routines NAG, c'est-à-dire sans exiger de l'utilisateur la connaissance d'un logiciel graphique évolué, tel UNIRAS; son apprentissage et son utilisation sont donc de ce fait simplifiés.

La structure du SG-NAG est divisée en trois niveaux :

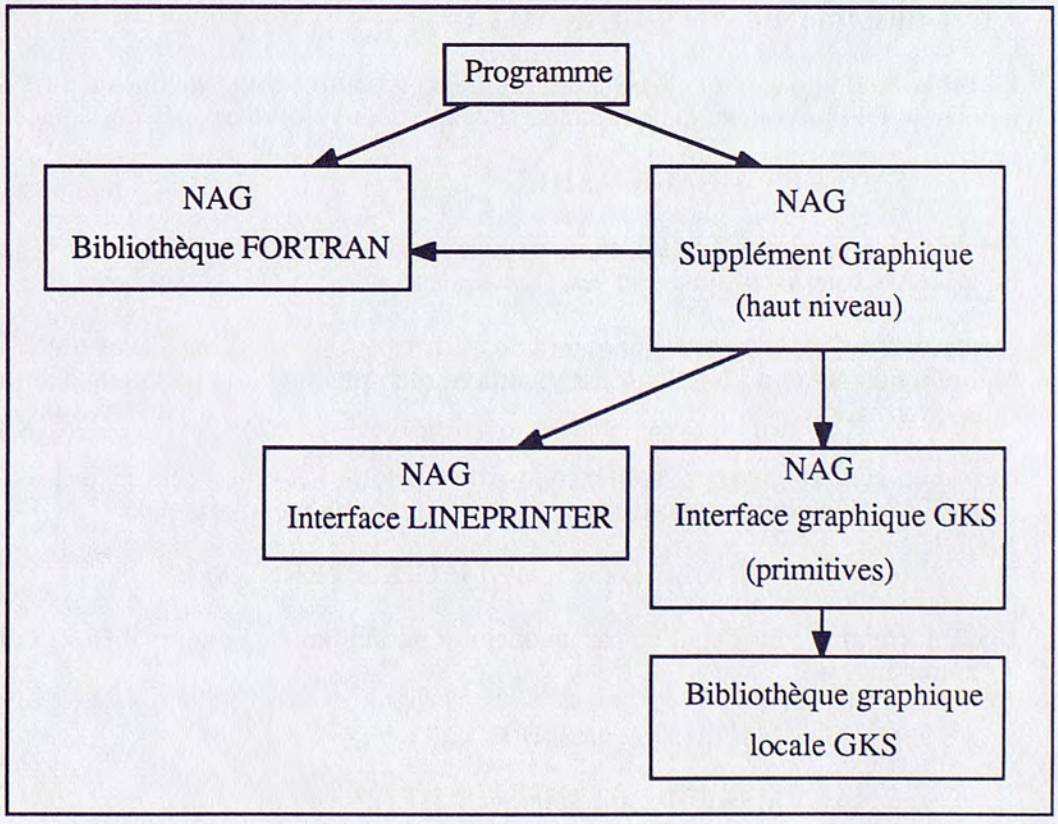


Figure 1. Structure du SG-NAG.

a) "**NAG Graphical Supplement**": un ensemble de routines de haut niveau (nommées J06Axx à J06Vxx) utilisant un petit ensemble de primitives graphiques et, éventuellement, des routines de la bibliothèque NAG-FORTRAN.

b) "**NAG Graphical Interface**": l'ensemble des primitives graphiques faisant des appels directs au logiciel graphique local (nommées J06Wxx à J06Zxx).

c) "**Local plotting package**": les routines du logiciel graphique local.

En n'utilisant que des routines de haut niveau dans son programme, l'utilisateur se prévient alors de toute dépendance vis-à-vis des particularités d'une implémentation locale de l'interface graphique. De tels programmes deviennent alors transportables d'un site NAG à l'autre.

4.1. Documentation.

Toutes les routines du SG-NAG sont décrites dans un volume "Graphical Supplement" disponible en consultation au Centre Informatique. Comme dans toute la documentation NAG, on y trouve de nombreux exemples, souvent prêts (ou presque) à être utilisés tels quels. Comme pour la bibliothèque NAG-FORTRAN, les descriptions de ces routines sont aussi accessibles avec le ON-LINE HELP, donc sur tout terminal!

4.2. Utilisation du SG-NAG.

Tout programme utilisant le SG-NAG **doit** suivre les étapes suivantes:

a) Appel à une routine d'initialisation du logiciel graphique local utilisé.

b) Appel à la routine d'initialisation du système graphique NAG (c'est la routine J06WAF).

c) Appels aux routines d'ajustement du système de coordonnées.

d) Appel à n'importe quel routine du SG-NAG (si possible uniquement des routines de haut niveau) nécessaires à la construction du graphique.

e) Appel à la routine de fin de graphique J06WZF **ou** appel à la routine de nouveau graphique J06WDF et retour, dans ce cas, à c).

Le point a) dépend de l'installation et sera traité ultérieurement. Le point c) mérite quelques commentaires. Imaginons que les données ou résultats à afficher soient compris dans les intervalles respectifs:

$$0.0 \leq X \leq 40.0 , \quad 0.0 \leq Y \leq 20.0$$

Ce sont les **coordonnées de l'utilisateur.** Il faut alors les ajuster aux **coordonnées du monde**, compréhensibles par le SG-NAG, grâce à l'appel:

```
CALL J06WBF (X1, X2, Y1, Y2, MARGE)
```

où:
```
X1 = 0.0
X2 = 40.0
Y1 = 0.0
Y2 = 20.0
```

MARGE = 1 ou 0 (selon qu'une marge est désirée ou non)

Cette routine transformera alors les coordonnées de l'utilisateur en coordonnées du monde comprises entre 0 et 1. Ce dernier carré est affiché sur la plus grande portion disponible de l'écran.

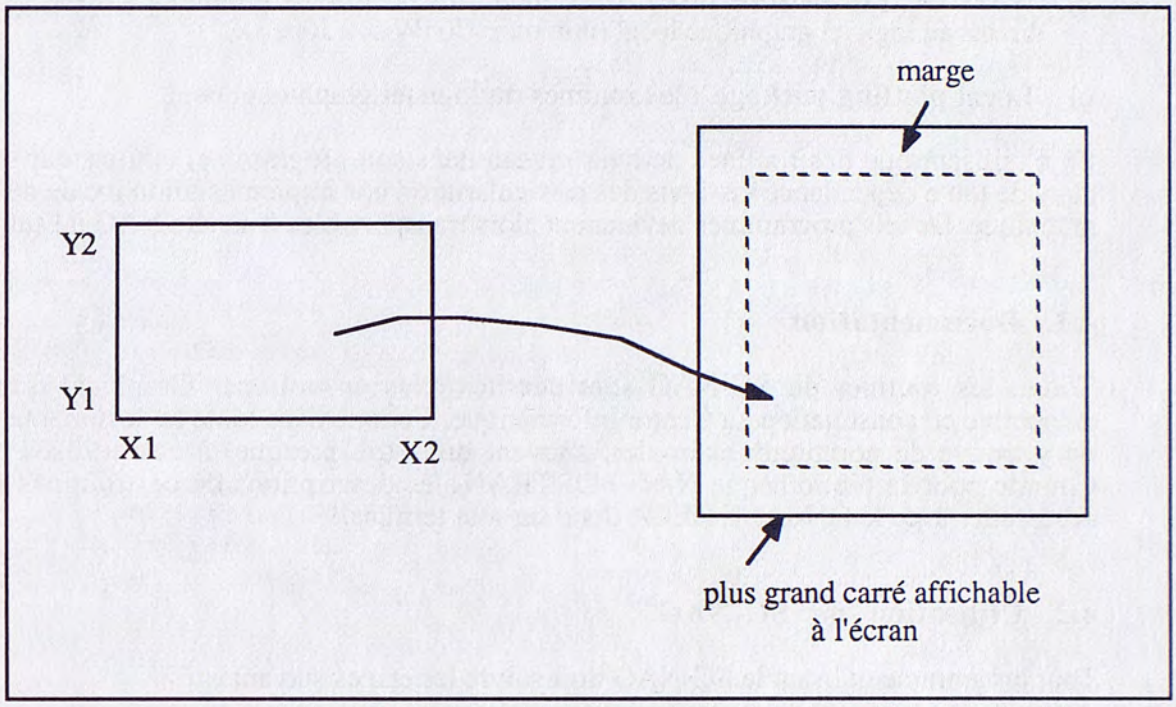


Figure 2. Changement de système de coordonnées.

Avec le routine J06WCF, il y a d'autre part possibilité de définir une fenêtre (partie de l'écran) où le graphique est affiché. Des détails supplémentaires concernant ces deux routines se trouvent dans la partie "Essential Introduction" du manuel.

4.3. Implémentation à l'UNIL.

La bibliothèque NAG-FORTRAN étant une version double précision, le SG-NAG l'est aussi; il faut donc faire attention aux exemples du manuel où des modifications s'imposent (cf. INFO-CI N° 8). Les routines GKS étant, elles, en simple précision, des conversions double à simple sont effectuées avant chaque appel à l'une de ces routines.

Comme sus-mentionné, il existe deux versions du SG-NAG à l'UNIL: l'une utilisant GKS, donc réservée aux terminaux et imprimantes graphiques, l'autre ne faisant appel à aucune base graphique et donc utilisable sur n'importe quel terminal alphanumérique (évidemment, la qualité des graphiques s'en ressent...). A deux endroits seulement, l'utilisateur verra une différence suivant son choix:

a) dans la routine d'initialisation du logiciel graphique local utilisé (routine notée XXXXXX dans les exemples du manuel);

b) au moment de l'édition des liens.

4.3.1. Terminaux graphiques.

L'utilisation des routines de GKS est alors autorisée. La phase d'initialisation de GKS (point a) ci-dessus) se résume à l'appel des trois routines de définition de l'environnement graphique suivantes:

```
CALL GOPKS (ERRLUN, WS)
CALL GOPWK (WKID, WKCON, WKTYP)
CALL GACWK (WKID)
```

La première ouvre l'unité logique pour les messages d'erreurs: ERRLUN. ERRLUN = 6 enverra les messages d'erreurs au terminal. Le paramètre WS précise la taille de l'espace de travail interne à GKS. WS = 0 est autorisé.

La seconde ouvre la station de travail. WKID est l'identificateur de la station. WKID = 1 précise que c'est la première station qu'on considère (l'ouverture d'une seule station est autorisée dans le contexte du SG-NAG). WKCON est le numéro d'unité logique de la station. WKCON = 1 est une valeur conseillée (à moins que l'unité 1 soit déjà utilisée!). Finalement, WKTYP est le type de station. En voici quelques exemples utilisés à l'UNIL:

WKTYP	**Station**	
340	VT340	(terminal)
4014	Tektronix 4014	(terminal)
4692	Tektronix 4692	(plotter; dans ce cas, il y a création d'un fichier auxiliaire: cf. INFO-CI no 8)

La dernière active la station de travail ouverte.

Ces trois appels suffisent à l'utilisation du SG-NAG. Il est clair que l'appel à n'importe quel autre routine de GKS sera reconnu, ceci étant déconseillé pour des raisons de portabilité. L'utilisation d'un certain nombre parmi celles-ci est toutefois recommandé pour des tâches spécifiques (cf. "Appendix to the Essential Introduction for the NAG GKS Interface").

Lors de l'édition des liens (c'est-à-dire après la compilation du programme), il faut appeler la bibliothèque GKS qui ne se trouve que sur ULA. Il faut donc suivre les points suivants:

a) Se brancher sur ULA (exécuter la commande SET HOST ULA au besoin).

b) Exécuter la procédure UNIL$LIBRARY:UNIRAS_LOGIN (on peut insérer la commande @UNIL$LIBRARY:UNIRAS_LOGIN dans son fichier LOGIN.COM).

c) Après compilation de son programme, on édite les liens avec la commande:

```
LINK mon_programme, NAG$GRLIB/LIBR,-
                    NAG$GKS/LIBR,-
                    NAG$LINK1/LIBR, NAG$LINK2/LIBR,-
                    GKSLIB/LIBR
```

La première bibliothèque contient les routines de haut niveau du SG-NAG, la seconde est l'interface graphique, les 3^{e} et 4^{e} sont les routines de la bibliothèque NAG-FORTRAN et la dernière est la bibliothèque GKS. Cette commande peut aussi se faire de manière plus concise:

```
LINK mon_programme, NAGGKS/OPT
```

4.3.2. Terminaux non-graphiques.

La situation est plus simple, dans ce cas, puisqu'aucune bibliothèque graphique locale n'est nécessaire. La phase d'initialisation peut ici se résumer à l'unique appel:

```
CALL X04ABF (1, NOUT)
```

qui redirectionne les sorties graphiques sur l'unité NOUT (le terminal en général: NOUT = 6).

L'édition des liens se fait (depuis ULA ou ULB indifféremment) avec la commande:

```
LINK mon_programme, NAG$GRLIB/LIBR,-
                    NAG$LPR/LIBR,-
                    NAG$LINK1/LIBR, NAG$LINK2/LIBR
```

On remarque que, dans ce cas, aucune bibliothèque graphique locale n'est nécessaire (cf. figure 1). On peut aussi entrer cette commande ainsi:

```
LINK mon_programme, NAGLPR/OPT
```

4.4. Exemple.

On dispose d'un ensemble de couples (X_i, Y_i). On désire dessiner une courbe Y=f(X) ajustée à ces points.

4.4.1. Recherche de la bonne routine NAG.

Grâce au "ON-LINE HELP" décrit ci-dessus, nous allons trouver la routine NAG qui va résoudre notre problème. Voici la démarche suivie:

```
$ HELP @NAGVMSHELP

@NAGVMSHELP Topic? HELP CHAPTERS
```

On repère alors le chapitre contenant les routines graphiques NAG, à savoir le chapitre J06 dont on demande une description:

```
@NAGVMSHELP Topic? J06 HELP
```

Différents types de graphiques sont alors proposés. On choisit celui qui nous intéresse, c'est-à-dire la section "Curves":

```
@NAGVMSHELP J06 Help Subtopic? CURVES
```

Quatre routines nous sont alors proposées. Nous choisissons la première: J06CAF. L'exemple fourni dans la description de cette routine est repris pour la suite de cette démonstration. On utilisera l'interface avec GKS.

4.4.2. Le programme FORTRAN.

Voici le code source FORTRAN du fichier XYPLOT.FOR:

```
C     J06CAF EXAMPLE PROGRAM TEXT
```

```
C     NAG GRAPHICAL SUPPLEMENT, MARK 1 RELEASE. NAG COPYRIGHT 1981.
C     .. Local Scalars ..
      DOUBLE PRECISION XMAX, XMIN, YMAX, YMIN
      INTEGER          I, IFAIL, METHOD, N, NIN, NOUT
C     .. Local Arrays ..
      DOUBLE PRECISION AX(30), AY(30)
C     .. External Subroutines ..
      EXTERNAL      J06AAF, J06AHF, J06BAF, J06CAF, J06WAF, J06WBF,
     *              J06WCF, J06WDF, J06WZF, J06XFF, X04AAF
c     *               J06WCF, J06WDF, J06WZF, J06XFF, X04AAF, XXXXXX
C
      DATA             NIN/5/, NOUT/6/
C
C     .. Executable Statements ..
C
C     SELECT OUTPUT CHANNEL FOR ERROR MESSAGES
C
      CALL X04AAF(1,NOUT)
C
C     INITIALISE PLOTTING DEVICE
C     *** SEE ESSENTIAL INTRODUCTION, SECTION 6, FOR DETAILS ***
C
c        CALL XXXXXX
      call  gopks(NOUT,0)
      call  gopwk(1,1,340)
      call  gacwk(1)
C
C     READ IN NUMBER OF DATA POINTS, LIMITS OF A SUITABLE
C     DATA REGION, AND DATA POINTS THEMSELVES
C
      READ (NIN,FMT=99999) N
      READ (NIN,FMT=99998) XMIN, XMAX, YMIN, YMAX
      READ (NIN,FMT=99997) (AX(I),AY(I),I=1,N)
C
C     INITIALISE NAG GRAPHICS SYSTEM
C     AND MAP DATA REGION ONTO VIEWPORT.
C
      CALL J06WAF
      CALL J06WBF(XMIN,XMAX,YMIN,YMAX,1)
      CALL J06WCF(0.0D0,1.0D0,0.0D0,1.0D0)
C
C     SELECT HIGH QUALITY CHARACTERS
C
      CALL J06XFF(2)
C
C     DRAW CURVE USING MONOTONIC METHOD
C
      METHOD = 1
      IFAIL = 10
      CALL J06CAF(AX,AY,N,METHOD,IFAIL)
C
C     ADD AXES AND TITLE, AND MARK POINTS ON CURVE
C
      CALL J06AAF
      CALL J06AHF('J06CAF EXAMPLE PLOT 1')
      IFAIL = 10
      CALL J06BAF(AX,AY,N,0,2,IFAIL)
C
C     ADVANCE FRAME AND DRAW CURVE USING CUBIC BESSEL METHOD
C
      CALL J06WDF
C
      METHOD = 2
```

```
      IFAIL = 10
      CALL J06CAF(AX,AY,N,METHOD,IFAIL)
C
C     ADD AXES AND TITLE, AND MARK POINTS ON CURVE
C
      CALL J06AAF
      CALL J06AHF('J06CAF EXAMPLE PLOT 2')
      IFAIL = 10
      CALL J06BAF(AX,AY,N,0,2,IFAIL)
C
C     TERMINATE PLOTTING
C
      CALL J06WZF
      STOP
C
99999 FORMAT (/,I3)
99998 FORMAT (4F10.4)
99997 FORMAT (2F10.4)
      END
```

Comme on peut le constater dans ce programme, l'utilisation du SG-NAG est très simple. Les adaptations qu'il a fallu faire pour définir l'environnement GKS sont marquées en gras dans le code.

4.4.2. Le fichier de données.

Voici le contenu du fichier d'entrée XYPLOT.DAT:

```
J06CAF EXAMPLE PROGRAM DATA
 11
0.0       10.0      0.0       10.0
0.0       1.0
1.0       1.1
2.0       1.1
3.0       1.2
4.0       1.3
5.0       7.2
6.0       3.1
7.0       2.6
8.0       1.9
9.0       1.7
10.0      1.6
```

4.4.3. La compilation et l'édition des liens.

Pour faciliter les opérations de compilation et d'édition des liens, on peut définir une petite procédure de commande qui effectuera le travail. Ceci est très pratique, surtout si l'on doit recommencer plusieurs fois... Voici le contenu de la procédure XYPLOT.COM:

```
$for xyplot.for
$lin xyplot,nag$grlib/libr,nag$gks/libr,-
            nag$link1/libr,nag$link2/libr,-
            gkslib/libr
```

Avant d'exécuter cette procédure, il faut s'assurer qu'on soit connecté sur ULA; l'appel à la bibliothèque GKS sera alors reconnu.

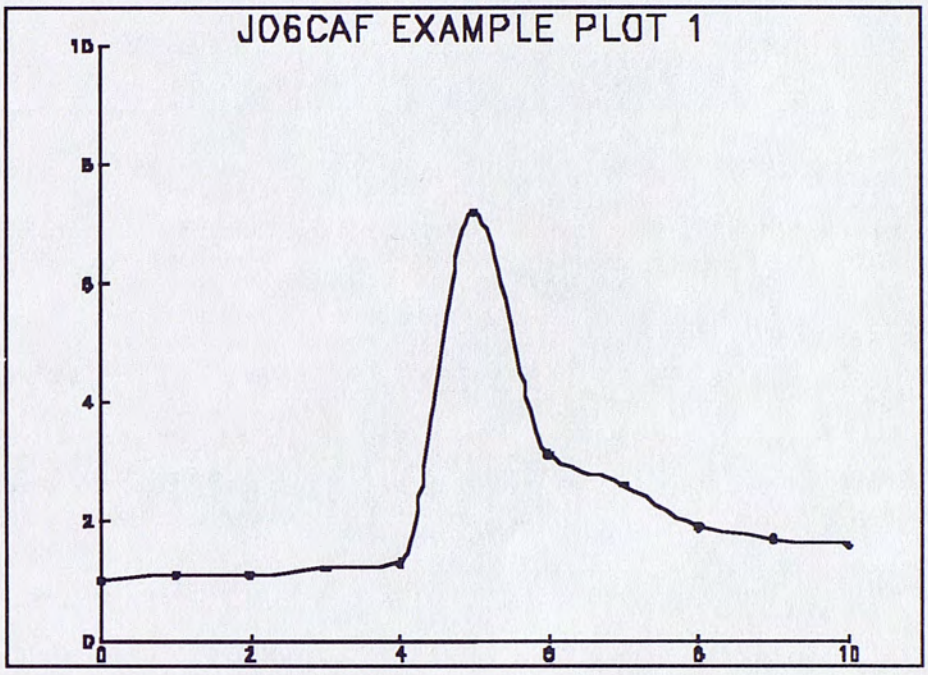
4.4.4. Les résultats.

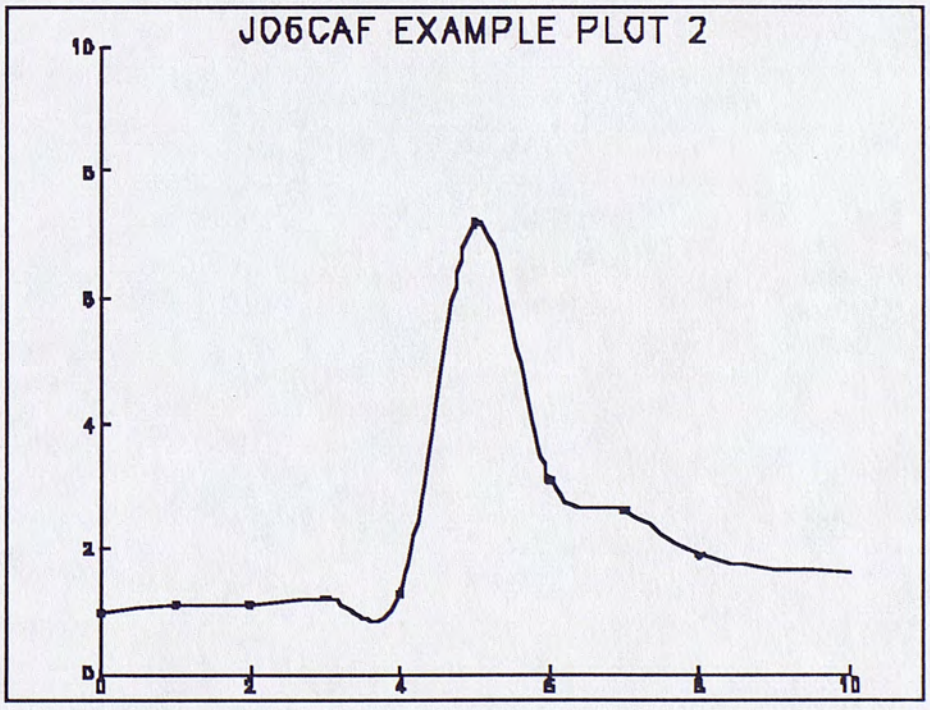
Après accomplissement heureux de la procédure ci-dessus, un fichier XYPLOT.EXE a été créé. On peut alors exécuter le programme ainsi:

```
$ DEFINE/USER SYS$INPUT XYPLOT.DAT

$ RUN XYPLOT
```

Les résultats suivants sont alors affichés à l'écran: *J. Guélat / CI*





CRAY-2 et CRAY-X-MP:

Des monstres au service des chercheurs de l'UNIL.

1. Introduction.

Avec deux milliards d'octets (2 Gbytes) de mémoire centrale et un milliard d'opérations arithmétiques par seconde ("peak performance": 960 MFLOPS) pour le CRAY-2/256 de l'EPFL et, respectivement, 64 Mbytes de mémoire et 1/2 milliard d'opérations par secondes pour le CRAY-X-MP/28 de l'ETHZ, une puissance de calcul hors du commun est offerte aux chercheurs de l'université. La présence sur le VAX-Cluster de l'UNIL de stations CRAY permet, d'autre part, un accès aisé à ces machines. Avec des notions élémentaires du système d'exploitation UNICOS 4.0 (dérivé du UNIX système V de AT&T), l'utilisateur peut travailler dans un environnement possédant des produits bien connus tels les compilateurs FORTRAN, Pascal et C, les bibliothèques mathématiques Eispack, Linpack, Nag 12 et IMSL, ainsi que les logiciels graphiques UNIRAS et EGM. Des progiciels plus spécialisés, tels, par exemple, MARC et PHOENICS (physique), GAMESS (chimie) et LAMPS (programmation mathématique), sont aussi disponibles.

Le temps CPU des deux superordinateurs a été réparti dans les différentes hautes écoles suisses, la part du lion revenant aux deux écoles polytechniques. L'Université de Lausanne pourra profiter d'une part d'environ 2% (environ 15 heures par mois) sur chacune de ces machines pour des tâches courantes. De plus, une part importante (30% à l'EPFL et 21% à l'ETHZ) de l'utilisation du temps CPU a été réservée pour les "meilleurs" grands projets auxquels les chercheurs de toute la Suisse peuvent participer.

En vue d'une utilisation optimale des superordinateurs, l' utilisateur doit garder en mémoire les points suivants: ces machines sont, avant tout, des "number crunchers", c'est-à-dire des machines à calculer. Il est hors de question d'y faire du traitement de texte, par exemple (même si les textes sont énormes). Ces supermachines ne diffèrent guère d'un ordinateur conventionnel si ce n'est qu'elles possèdent une architecture permettant de réaliser des opérations arithmétiques très rapidement (vectorisation) et que la présence de plusieurs processeurs permet l'élaboration d'algorithmes parallèles. Les programmes soumis devront donc utiliser au mieux ces possibilités.

Le but de cet article est d'indiquer à la personne intéressée la marche à suivre pour obtenir l'ouverture d'un compte sur un CRAY et de l'aider dans la soumission des premières tâches d'initialisation. Des exemples de tâches types, telles la compilation et l'exécution d'un programme FORTRAN ou le transfert de fichiers du VAX au CRAY et inversement, sont

décrits et commentés. Par contre, seules des informations ponctuelles sont fournies en ce qui concerne l'utilisation du système UNICOS. Le lecteur est prié de se reporter aux manuels de références pour de plus amples détails.

2. Ouverture d'un compte sur un CRAY.

Tout utilisateur d'un CRAY doit remplir au préalable une *inscription d'utilisateur* dont les formulaires peuvent être obtenus au Centre Informatique. Il devra faire viser sa demande par le responsable du centre de frais sous lequel il travaillera. Des informations générales sur l'espace disque et le temps CPU requis lui seront demandées (cf. INFO-CI N° 8 pour des informations sur les limites). A ce propos, il faut noter qu'il existe sur CRAY une zone SCRATCH utilisable par n'importe qui et que les stations CRAY installées sur ULYS permettent le rapatriement et le stockage des fichiers sur le VAX de l'université.

Il existe, en outre, un certain nombre de restrictions concernant l'utilisation d'un CRAY, en particulier celles du département de la Défense qui en interdit l'accès à des ressortissants de certains pays* . Pour ces personnes, il est conseillé de préparer leurs tâches sur ULYS et de les faire exécuter sur le CRAY par une personne autorisée.

Dès création du compte, l'utilisateur reçoit un *username* et un *password* lui permettant d'accéder au CRAY, ainsi que ses limites d'utilisation de l'espace disque pour le stockage de fichiers permanents.

Le choix entre CRAY-2 et CRAY-X-MP dépend du type de l'utilisation qu'on désire en faire: les tâches mal vectorisées (quasi-scalaires) iront de préférence sur le CRAY-X-MP alors que celles requérant beaucoup d'entrées/sorties seront plutôt dirigées sur le CRAY-2 (grande mémoire centrale); le choix d'un logiciel spécialisé peut aussi dicter celui de la machine, certains logiciels n'étant installés que sur l'une des deux machines.

3. Support technique et documentation.

Le support technique de base aux utilisateurs du CRAY-2 est assuré par la PAT (Permanence d'Aide Technique) de l'EPFL qu'on peut atteindre au numéro de téléphone suivant: 693.22.09. Le SIC (Service Informatique Central) fournit lui aussi de l'aide aux utilisateurs: tél. 693.22.11.

En ce qui concerne le CRAY-X-MP, le support est assuré par le Centre de Calcul de l'ETHZ (01-256.22.11).

---

* Afghanistan, Albanie, Bulgarie, Cambodge, Corée du Nord, Cuba, Hongrie, Laos, Mongolie, Pologne, RDA, Roumanie, R.P. de Chine, Tchécoslovaquie, URSS, Vietnam

Les manuels CRAY sont en consultation publique dans les salles du SIC-EPFL. Un certain nombre de ces manuels sont en vente à la Librairie Polytechnique. La liste qui suit indique ceux qui sont consultables au Centre Informatique de l'université:

- UNICOS Overview for Users
- UNICOS Primer Reference Manual
- UNICOS User Commands Reference Manual
- UNICOS System Calls Reference Manual
- UNICOS File Formats & Special Files Reference Manual
- UNICOS Performance Utilities Reference Manual
- UNICOS Text Editor's Primer
- CFT77 Reference Manual
- Segment Loader (SEGLDR) Reference Manual
- CRAY-2 UNICOS Libraries, Macros and Opdefs Reference Manual
- DEC VAX/VMS Station Reference Manual
- DEC VAX/VMS Summary of Differences for UNICOS installations

Il est possible d'obtenir des informations au terminal directement:

- en se connectant sur *info* depuis le réseau EPNET2;
- en utilisant la commande UNICOS *man* donnant accès aux pages du manuel "UNICOS User Commands Ref. Man.";
- en utilisant la commande UNICOS *news* fournissant des nouvelles sur les installations locales;
- en consultant le répertoire */doc* sur le CRAY-X-MP (documentation en allemand).

4. Horaire.

A l'heure actuelle, le superordinateur CRAY-2 fonctionne sous le contrôle d'opérateurs tous les jours ouvrables de 08:00h à 19:30h. En dehors de ces heures, le système fonctionne en mode automatique. Pour des raisons d'entretien, le système est indisponible les jours et heures suivants:

lundi	de 05:00h à 11:00h
mardi, mercredi et vendredi	de 07:30h à 08:00h
jeudi	de 06:00h à 10:15h

Table 1. Indisponibilité du CRAY-2.

Le CRAY-X-MP est indisponible pour cause d'entretien les jours et heures suivants:

lundi	de 06:00h à 10:00h
jeudi	de 06:00h à 10:00h

Table 2. Indisponibilité du CRAY-X-MP.

5. Accès au CRAY.

L'accès au CRAY peut se faire de deux manières différentes:

a) Accès via une frontale.

Deux ordinateurs frontaux sont disponibles à l'EPFL: un VAX11-780 et un CDC-CYBER (180-855). Chacun de ces ordinateurs possède un logiciel de communication avec le CRAY appelé *station* . Il est donc possible d'accéder au CRAY-2 via une station VMS et une station NOS/VE. L'interface CRAY-frontale (FEI: Front End Interface) est géré par le sous-système USCP (Unicos Station Call Protocol). A Zürich, l'accès par frontale VAX/VMS (EZCRAY), CDC NOS/VE ou CDC NOS est assuré.

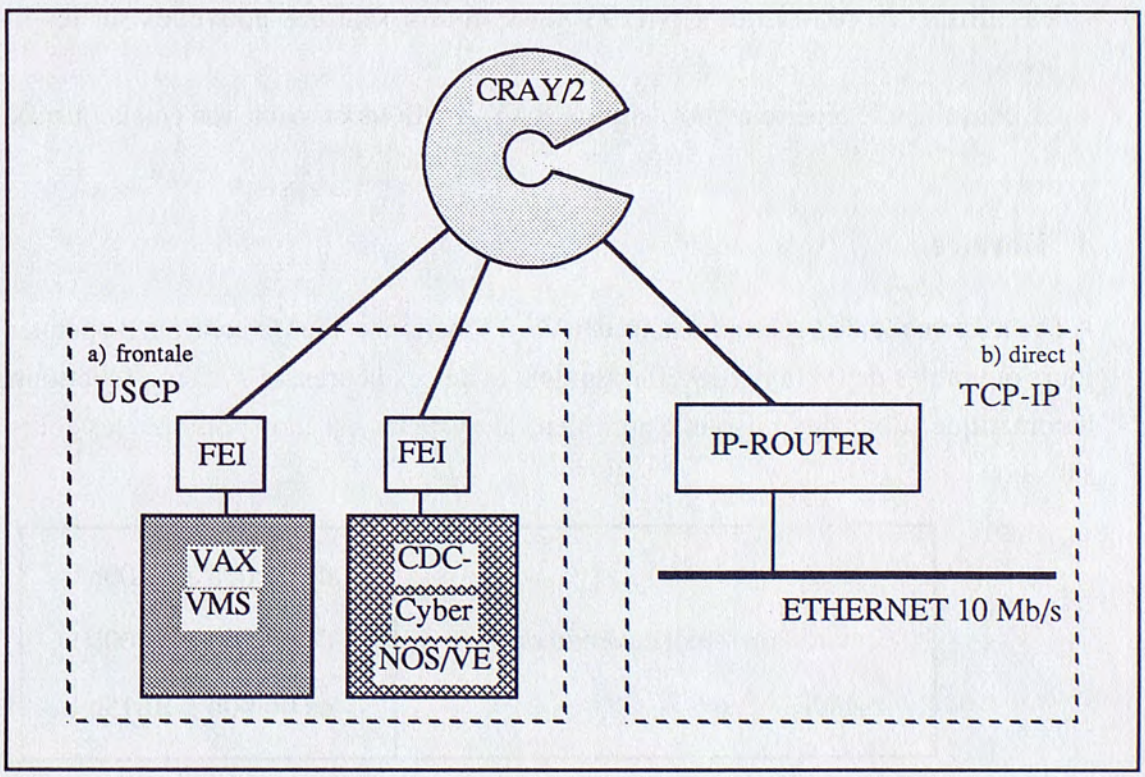


Figure 1. Accès au CRAY-2.

b) Accès direct.

Prévu pour début 1989, l'accès au CRAY-2 par le réseau EPNET2 permettra le branchement interactif à partir d'une station utilisant le protocole TCP/IP. Pour le CRAY-X-MP, ce genre d'accès existe mais est réservé à certains utilisateurs pour des raisons de sécurité.

Pour les utilisateurs de l'UNIL, des stations "remote" ont été installées sur le VAX-Cluster ULYS. Ces stations permettent d'atteindre par DECnet les CRAY via leur ordinateur frontal VAX respectif dont la présence devient alors totalement transparente à l'utilisateur. Le transfert des fichiers du VAX ULYS vers un CRAY, et inversement, se fait aisément (voir plus loin les commandes *fetch* et *dispose*).

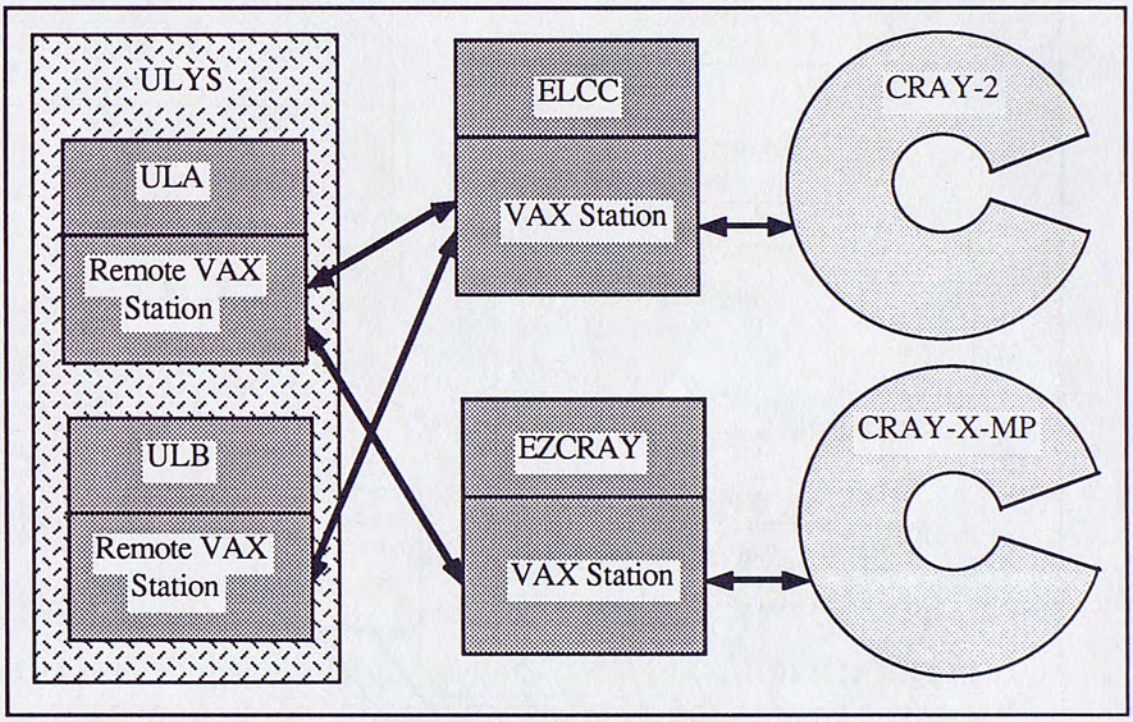


Figure 2. Station "remote".

Comme on peut le constater sur cette figure, l'accès au CRAY-X-MP ne peut se faire que depuis ULA, ceci étant dû à la politique de l'ETHZ de restriction dans la distribution des stations remote. Il faut donc se connecter sur ULA pour travailler sur le CRAY de Zürich alors que celui de Lausanne est atteignable depuis ULA ou ULB indifféremment.

6. Interactif vs Batch.

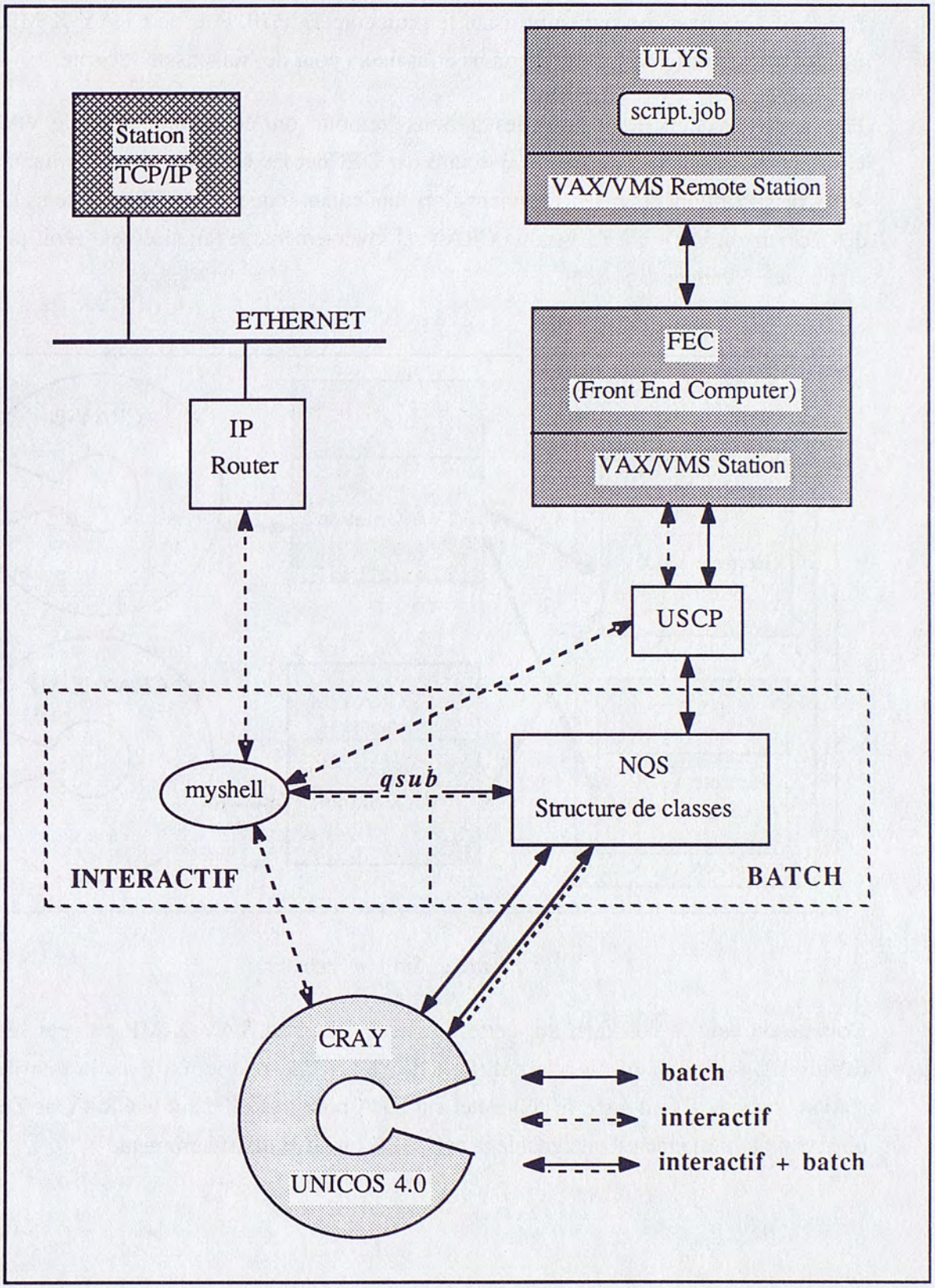


Figure 3. Interactif vs batch.

Les superordinateurs CRAY peuvent être utilisés soit interactivement, soit en traitement par lots ("batch"). L'accès interactif par frontale ne peut se faire que depuis une station attachée, c'est-à-dire la station de l'ordinateur frontal. A l'heure actuelle, seule l'utilisation en mode "batch" des CRAY est autorisée à monsieur tout-le-monde. L'utilisateur prépare sur la station un fichier de commandes UNICOS (un *shell-script*). Ce fichier est alors transmis au CRAY en passant par l'interface NQS (Network Queueing System) qui classe les tâches selon les ressources (espace mémoire, temps CPU) qu'elles requièrent. Interactivement, il est aussi possible de soumettre des tâches batch grâce à la commande UNICOS *qsub*. La description du manuel de cette commande fournit de précieuses indications sur l'utilisation des commandes NQS.

La structure des classes NQS pour le CRAY-2 est décrite dans INFO-CI N° 8. Une nouvelle classe a été rajoutée depuis: il s'agit de la classe dite "long" autorisant des tâches d'au plus 5 heures de temps CPU et 32MW de mémoire centrale. Une autorisation spéciale doit être demandée pour pouvoir accéder à cette classe. Pour le CRAY-X-MP, la structure des classes NQS est la suivante:

Classe	Limite CPU (secondes)	Limite mémoire (MWords)	# max.de jobs simultanés	Priorité ("nice-value")
express	10	1	16	0
little	20	2	12	1
low	200	3	9	2
medium	2000	4	5	4
minster	14400	6	3	5
monster	36000	6	2	6
super	aucune	6	1	6
toll	aucune	8	1	6

Table 3. Classes de NQS pour le CRAY-X-MP.

Ces classes peuvent être modifiées selon les besoins. En utilisant la commande UNICOS *qstat*, on obtient une information à jour et plus détaillée sur les limites de chaque classe.

7. Utilisation des stations CRAY sur ULYS.

Le logiciel de communication avec les superordinateurs CRAY installé sur ULYS s'intitule "CRAY Remote VAX Station". Il s'agit d'un programme interactif mettant à la disposition de l'utilisateur un certain nombre de commandes permettant, entre autres, de

- soumettre des tâches sur le CRAY;

- à partir d'une tâche en exécution sur le CRAY, transférer des fichiers du VAX au CRAY et inversement;
- contrôler les queues d'attente et d'exécution des tâches.

Avant d'appeler ce programme, l'utilisateur doit inclure dans son fichier LOGIN.COM la commande:

```
$ @UNIL$LIBRARY:CRAY_LOGIN
```

qui initialise noms logiques et symboles nécessaires au bon fonctionnement du programme. Par défaut, c'est la station du CRAY-2 qui est initialisée. La commande CXCHOOSE permet de changer de station. Par exemple:

```
$ CXCHOOSE ETHZ     ! Sélectionne la station pour Zürich.
$ CXCHOOSE EPFL     ! Revient à la station de défaut, c'est-à-dire celle pour
                      Lausanne.
$ CXCHOOSE          ! Liste les stations disponibles.
```

Il faut se rappeler (cf. fig.2) qu'il n'y a pas de station pour Zürich sur ULB! Après avoir choisi la bonne station, on peut appeler le programme avec la commande: $ CRAY. L'écran se divise alors en deux parties, la partie supérieure étant réservée aux messages de la station, la partie inférieure à l'entrée des commandes (après le texte d'invitation "CRAY>"). On trouve sur la bande médiane l'indication *logged-on* si la station est en fonction, c'est-à-dire si l'on peut soumettre des tâches au CRAY. Sinon, l'indication *logged-off* est affichée, signifiant une interruption de la communication (heures d'entretien des ordinateurs CRAY, panne des frontales, ...) Quelques commandes utiles sont données dans la table 4.

La queue de "staging" à laquelle il est fait allusion dans la table 4 est une queue d'attente entre le VAX et le CRAY. C'est une bonne idée de la consulter lorsque qu'on a perdu la trace d'une tâche. Il se peut, par exemple, que, par manque d'espace sur VAX, un fichier ait été retenu dans cette queue. Après avoir libéré des blocs sur le VAX, on peut réactiver le transfert avec la commande: RELEASE jsq† . Si cela n'est pas possible, il faudra abandonner le transfert par une commande KILL.

Les commandes station peuvent être entrées directement depuis le niveau DCL sans entrer dans le contexte CRAY. Il suffit pour cela de faire précéder les commandes en question par le mot "CRAY". Par exemple, on peut faire exécuter un shell-script sur le CRAY avec la commande DCL:

```
$ CRAY SUBMIT script.job
```

---

† Pas (encore) valide sous UNICOS!

Commande Station	Fonction
HELP [commande station]	Donne des informations sur les commandes station.
EXIT	Sort du contexte CRAY et retourne au niveau DCL.
SUBMIT fichier	Soumet une tâche au CRAY.
SET TERM INFO	Emet des messages lors de chaque transfert CRAY-VAX.
STATUS	Affiche l'état des tâches batch en cours d'exécution. On y trouve, en particulier, le *jsq* (Job Sequence Number) des tâches.
STATUS/OWNER	Affiche l'état de ses propres tâches.
STATUS/CLASS=class-id	Affiche l'état des tâches d'une classe NQS particulière.
STATUS/INPUT	Affiche les tâches en attente.
STATCLASS	Affiche la structure des classes NQS.
SNAP fichier	Envoie une copie d'écran dans un fichier VMS.
SHOW QUEUES	Affiche le contenu de la queue de "staging" entre le CRAY et le VAX (utile si on ne retrouve plus ses tâches...)
REMOVE	Supprime les tâches de la queue de "staging".
RELEASE	Relâche un fichier, en transfert vers la frontale, qui était bloqué ("HOLDING").
KILL jsq	Tue une tâche. Si la tâche est en attente: elle est enlevée de la queue; en exécution: elle est avortée; terminée: les sorties (en queue) sont détruites.

Table 4. Commandes station.

8. Structure d'une tâche batch.

Le fichier VMS contenant une tâche batch à envoyer sur le CRAY par la commande SUBMIT (que nous appellerons plus succinctement *fichier batch*) est constitué des trois parties suivantes:

Identification de l'utilisateur
Commandes NQS
Shell-script (commandes UNICOS)

a) Identification de l'utilisateur.

La première ligne de tout fichier batch doit contenir le nom et le mot de passe de l'utilisateur sous la forme:

```
# user=username  pw=password
```

Cette ligne doit être en **minuscules**. L'apparition du mot de passe à même le fichier est fort peu désirable, c'est pourquoi il est recommandé d'interdire l'accès à ces fichiers. Une manière plus élégante est l'utilisation de la commande locale *crsub* (voir plus loin). Dans ce cas, le nom et le mot de passe de l'utilisateur sont demandés interactivement au terminal.

b) Commandes NQS.

Les commandes NQS permettent de spécifier certaines options associées à la tâche. Aucune commande NQS n'est obligatoire: la portion NQS du fichier batch peut être vide; dans ce cas, les valeurs par défaut des options sont considérées, c'est-à-dire celles de la classe *express*. Toutes les options disponibles sont décrites dans la commande *qsub* qui est le moyen interactif d'envoyer des tâches batch. Les commandes NQS ont la forme:

```
# QSUB option valeur  # commentaire
```

Voici les plus courantes (les valeurs de défaut sont celles du CRAY-2; celles du CRAY-X-MP sont données entre parenthèses):

Option	Valeur	Défaut	Signification
-a	date & heure	immédiat	Traitement de la tâche à la date citée.
-r	nom	nom du fichier batch	Assigne un nom à la tâche (7 caractères considérés). Le fichier de retour sur VAX portera ce nom (avec le type .CPR).
-lm	limite	4 MW (1 MW)	Mémoire maximale par processus.
-lM	limite	4 MW (1 MW)	Mémoire maximale par tâche.
-lt	limite	20 sec (10 sec)	Temps CPU maximal par processus.
-lT	limite	20 sec (10 sec)	Temps CPU maximal par tâche.
-o	fichier	stdout	Redirection du fichier de sortie.
-s	shell	/bin/sh	Spécification du shell.

Table 5. Principales options NQS.

c) Shell script.

La troisième partie d'un fichier batch contient les commandes UNICOS. Par défaut, les tâches batch s'exécutent en Bourne-shell. L'option NQS -s permet de choisir le C-shell, ce qui n'est pas recommandé par l'EPFL.

9. Exemples de fichiers batch.

Une fois que le fichier batch a été construit selon la structure décrite précédemment, il peut être envoyé sur le CRAY avec la commande CRAY SUBMIT. Si le fichier ne contient pas l'instruction QSUB -r, le fichier de retour portera le nom de l'utilisateur (éventuellement complété par des _), avec le type CPR. Par exemple, la commande:

```
$ CRAY SUBMIT myscript.job
```

retournera un fichier de résultats nommé "username.cpr". Si, au contraire, le fichier myscript.job contient la commande NQS suivante:

```
# QSUB -r alpha
```

le fichier de retour portera le nom "alpha.cpr".

9.1. Première tâche: changer le mot de passe.

La première tâche pour tout nouvel utilisateur consiste à définir son mot de passe. Son premier fichier batch doit donc être de la forme suivante:

```
# user=username  pw=initial_password                    1
#
passwd -b username << EOD                               2
initial-password                                        3
personnal_password                                      4
EOD                                                     5
```

La première ligne est la ligne d'identification de l'utilisateur. Il n'y a pas de commandes NQS dans cet exemple. La tâche sera donc traitée dans la classe *express*. La deuxième ligne fait appel à la forme batch de la commande UNICOS *passwd*. Les lignes 3 et 4 forment le contenu du "here file" donné à la commande *passwd* comme entrée standard. L'ancien et le nouveau mot de passe y figurent. La ligne 5 marque la fin de ce fichier d'entrée.

Encore une fois, ce fichier contiendra en toutes lettres votre mot de passe, ce qui est indésirable (voir la commande locale *crpasswd*). A l'EPFL, on recommande de changer de mot de passe au moins une fois par mois.

9.2. Transfert de fichiers.

Le transfert de fichiers de VAX à CRAY et inversement se fait à l'aide des commandes UNICOS suivantes:

fetch : Transfert un fichier du VAX vers le CRAY. Si un fichier de même nom existe déjà sur le CRAY, le contenu de ce premier fichier sera remplacé par le nouveau.

acquire : Transfert un fichier du VAX vers le CRAY, sauf si un fichier portant le même nom existe déjà sur le CRAY, auquel cas le transfert n'a pas lieu.

dispose : Transfert un fichier du CRAY vers le VAX.

Ces commandes possèdent de nombreuses options dont on trouve une description dans le manuel. L'exemple de fichier batch qui suit, indique une manière de faire. Il transfert le fichier texte "lettre.txt" du VAX vers le CRAY, compte les mots de ce fichier et renvoie le résultat dans le fichier "nbmots.res".

```
# user=username  pw=password                                    1
# QSUB -r ctwords                                               2
#
# Transfert du VAX au CRAY.                                     3
fetch lettre -dST -fCB -t'$DISK LETTRE.TXT'                     4
#
# Comptage des mots.                                            5
echo "Le fichier contient `cat lettre | wc -w` mots." > nbmots  6
#
# Transfert du CRAY au VAX.                                     7
dispose nbmots -dST -t'$DISK NBMOTS.RES'                        8
#
rm lettre nbmots                                                9
```

La première ligne est la ligne d'identification de l'utilisateur. La seconde est la seule commande NQS de ce fichier batch. Elle assigne un nom à la tâche. Les lignes 3,5 et 7 sont des commentaires qui ne sont pas interprétés par le shell. A la ligne 4, on importe sur le CRAY le fichier contenant le texte à analyser. L'option "-dST" indique que le fichier à importer réside sur un disque du VAX. L'option "-fCB" précise le format de transfert. Le champ texte de l'option -t est la partie VMS de la commande. Les options de la commande *dispose* de la ligne 8 ont les mêmes significations. Le coeur du programme (comptage des mots) se résume à la ligne 6, la dernière ligne ne faisant que du ménage sur le CRAY.

9.3. Compilation et exécution d'un programme FORTRAN.

Le fichier batch suivant montre un exemple de compilation d'un programme FORTRAN dont le code source est en partie seulement dans le fichier lui-même. Des statistiques sur l'exécution de la tâche sont demandées.

```
# user=username  pw=password                                          1
# QSUB-r FORTRAN   # request name                                      2
#
# Initiate accounting.                                                3
ja                                                                    4
#
# Create FORTRAN source program (as a here file).                     5
cat <<'END_OF_PROGRAM' > source.f                                     6
       program intable                                                7
       integer ind                                                    8
       real z,tab(8)                                                  9
       read(5,'(8f7.2)') tab                                         10
       read(12,'(2x,f10.4)') z                                       11
       call elem(z,tab,ind)                                          12
       print*,' RESULTATS'                                           13
       write(6,*) tab,z,x                                            14
       end                                                           15
END_OF_PROGRAM                                                       16
#
# Create input data files (as here files).                           17
cat <<'END_OF_DATA1' > fort.12                                        18
    100.5000                                                         19
END_OF_DATA1                                                         20
cat <<'END_OF_DATA2' > data2                                         21
 123.45  10.34 234.45 100.50 765.98 765.54 543.23 888.88             22
END_OF_DATA2                                                         23
#
# Transfer subroutine elem.f from VAX to CRAY2.                      24
fetch elem.f -dST -fCB -t'$DISK ELEM.F'                              25
#
# Compile main and subroutine.                                       26
cft77 source.f ; cft77 elem.f                                        27
#
# Link main and subroutine.                                          28
segldr source.o elem.o                                               29
#
# Execute program.                                                   30
a.out < data2 > output                                               31
#
# List and purge current CRAY directory.                             32
ls -l >> output                                                      33
cat output                                                           34
rm a.out *.o  *.f fort.* data 2 output                               35
#
# Terminate accounting.                                              36
ja -ct | nasa                                                        37
```

Excepté pour les deux premières lignes qui ont les mêmes fonctions que dans l'exemple précédent, les lignes commençant par un # sont des commentaires non interprétés par le shell. A la ligne 4, un appel à la commande *ja* est réalisé pour indiquer au shell que la collecte de statistiques commence à cet endroit. Le fichier UNICOS "source.f" contenant le programme principal est créé aux lignes 6 à 16. Ce programme lit un réel sur l'entrée standard (unité 5) et un tableau de réels sur l'unité 12. A noter qu'un tel fichier FORTRAN porte, par défaut, le nom "fort.12". L'appel à la routine "elem" retourne l'indice dans le tableau du réel lu. L'impression des résultats est faite sur la sortie standard (unité 6). La ligne 16 indique la fin de la définition du fichier "source.f". Deux fichiers de données sont ensuite créés aux lignes 18 à 23. Le code de la routine "elem" est ensuite importé du VAX par une commande *fetch*. La compilation du programme (ligne 27) fait appel au compilateur FORTRAN CFT77 de UNICOS. Après édition des liens (ligne 29), le programme est exécuté avec redirection des entrées et sorties standards (ligne 31). Un liste des fichiers du répertoire courant est ajoutée au fichier de sortie "output" à la ligne 33. Ce fichier peut être conservé si on le désire (on pourrait le rapatrier sur le VAX, par exemple). Après un peu de ménage sur CRAY (ligne 35), on indique au shell que la collecte de statistiques est terminée (option -t de la commande *ja*) et on demande l'impression de statistiques sur chaque commande du script (option -c). La sortie de cette commande est passée dans le filtre *nasa* qui rend le format de la sortie adéquat pour une imprimante.

Le contenu du fichier "FORTRAN.CPR" retourné sur VAX est le suivant (soumission sur le CRAY-X-MP de l'ETHZ):

```
 BATCH JOB STARTED:  Mon Jan 9 13:58:32 MEZ 1989

 This job is now running in: /scr/327.sn420
 All output files will be removed when this job terminates.
 Copy all output files you wish saved to /u/div/u5825.
 ---------------------JOB OUTPUT FOLLOWS---------------------------

      Welcome to UNICOS 4.0

Donnerstag, 12. Januar 1989:
   Kein CRAY-Betrieb von 06.00 Uhr - 10.00 Uhr
   Grund: Wartung                RZETH-Betrieb
news: documentation
  RESULTATS
 123.45,  10.34, 234.45, 100.5, 765.98, 765.54, 543.23, 888.88, 100.5, 4
total 80
-rwx------   1 u5825    div       298920 Jan  9 13:58 a.out
-rw-------   1 u5825    div           57 Jan  9 13:58 data2
-rwx------   1 u5825    div          159 Jan  9 13:58 elem.f
-rw-------   1 u5825    div          536 Jan  9 13:58 elem.o
-rw-------   1 u5825    div           13 Jan  9 13:58 fort.12
-rw-------   1 u5825    div           93 Jan  9 13:58 output
-rw-------   1 u5825    div          163 Jan  9 13:58 source.f
-rw-------   1 u5825    div         1888 Jan  9 13:58 source.o
```

```
Job Accounting - Command Report
===============================

Command  Started    Elapsed    User CPU    Sys CPU    I/O Wait
  Name      At       Seconds     Seconds    Seconds    Seconds     SBU's
========= ======== ========== ========== ========== ========== =========
ja        13:58:34     0.1630     0.0024     0.0054     0.1050      0.00
cat       13:58:34     0.1450     0.0027     0.0049     0.0767      0.00
cat       13:58:34     0.3480     0.0026     0.0049     0.2124      0.00
cat       13:58:35     0.2234     0.0026     0.0050     0.0847      0.00
sh        13:58:35     0.5267     0.0051     0.0244     0.1103      0.00
uscpbloc 13:58:38      0.6202     0.0008     0.0048     0.2500      0.00
fetch     13:58:35     3.6350     0.0037     0.0233     0.0837      0.00
cft77     13:58:39     1.1734     0.0517     0.0202     0.9692      0.00
cft77     13:58:40     1.0703     0.0609     0.0195     0.9717      0.00
segldr    13:58:41     0.9536     0.3008     0.0530     0.1864      0.00
a.out     13:58:42     0.1829     0.0040     0.0101     0.1000      0.00
ls        13:58:42     0.3261     0.0599     0.0118     0.0937      0.00
cat       13:58:42     0.1631     0.0023     0.0044     0.0550      0.00
rm        13:58:43     0.3579     0.0024     0.0104     0.0371      0.00
--------------------JOB OUTPUT ENDS-------------------------------
Changing directory to /u/div/u5825
Removing temporary directory /scr/327.sn420
```

Comme indiqué dans les trois premières lignes de cette sortie, les tâches s'exécutent dans un sous-répertoire de */src* (scratch). Tous les fichiers créés sont perdus à la fin de la tâche, à moins qu'on ne les sauve explicitement (ce qui rend la ligne 35 inutile dans ce cas). Cette politique d'économie d'espace n'est valable que sur le CRAY-X-MP de Zürich. Comme on peut le constater ci-dessus, les fichiers exécutables générés sont énormes (même pour un programme minuscule). Il est donc recommandé de ne pas les conserver.

10. Commandes locales.

Comme on l'a vu précédemment, il s'avère que le mot de passe de l'utilisateur d'un CRAY apparaît dans les fichiers batch qu'il soumet, ce qui n'est pas désirable. Pour pallier à cet inconvénient, des commandes locales ont été développées à l'UNIL. En voici la liste actuelle:

crsub : Soumet des tâches sur CRAY. Le nom d'utilisateur et le mot de passe sont demandés interactivement.

crpasswd: Change le mot de passe sur CRAY.

crman: Consulte le manuel "UNICOS User Commands Reference Manual".

Une description de ces commandes peut être obtenue au terminal en utilisant la commande suivie d'un point d'interrogation. Par exemple, une description de la commande *crsub* s'obtient avec la commande:

```
$ CRSUB ?
```

Voici quelques exemples d'utilisation de la commande *crsub*. Imaginons qu'un fichier batch "test.job" ait été préparé sans la première ligne d'identification de l'utilisateur. Pour soumettre cette tâche sur le CRAY-2 de l'EPFL, on utilisera alors la commande:

```
$ CRSUB test
```

Les options de la commande *crsub* permettent , entre autres, de choisir l'ordinateur CRAY sur lequel on désire travailler et d'indiquer, au niveau de la commande, le nom d'utilisateur. Ceci peut être exploité en définissant, par exemple, un symbole du type:

```
$ CRSZ :== 'CRSUB' -C ETHZ -U USERNAME
```

La commande

```
$ CRSZ test
```

enverra alors directement la tâche sur le CRAY-X-MP de l'ETHZ en ne demandant interactivement que le mot de passe de l'utilisateur.

Toute proposition concernant l'enrichissement de cet ensemble de commandes locales est la bienvenue.

J. Guélat / CI