PUB 93-104

Joël CLOSIER

MEMOIRE INDUSTRIEL ESIEA

1992

Sujet du mémoire

Développement d'un logiciel système pour un serveur de données qui est utilisé simultanément par un système UNIX et VMS. L'ensemble du système comprend environ 100 stations de travail, 100 gigabytes d'espace disque. Une connexion au projet SHIFT est à prévoir. Les données sont utilisées pour des analyses physiques dans l'expérience ALEPH.

Stage effectué au **Laboratoire Européen pour la physique des Particules**
du 1er janvier 1992 au 28 février 1993

Mémoire soutenu le 19 novembre 1992

Président du jury : **A.M. KEMPF** ,
Directeur du 3ème cycle de l'E.S.I.E.A.

Maître de stage : **J. KNOBLOCH** ,
Responsable du département logiciel dans l'expérience ALEPH au CERN

Rapporteur : **B. RONDEAU** ,
Responsable domaine systèmes internes de gestion/bureautique au Crédit Agricole du Loiret

à mes parents ....

Remerciements

Je tiens à remercier Monsieur Knobloch, mon maître de stage, qui m'a permis d'effectuer mon travail dans de très bonnes conditions et qui m'a aidé à m'incorporer au sein de son équipe.

Mes remerciements vont également à Ronald Hagelberg et à John Hilgart qui m'ont beaucoup appris sur les travaux du groupe et sur le métier d'ingénieur système.

Je remercie également Monsieur Rondeau qui m'a conseillé pour la gestion de mes projets, ainsi que pour la rédaction de ce mémoire.

Enfin, je ne peux pas oublier de remercier l'ensemble du groupe OFFLINE pour sa gentillesse, sa patience et son aide, sans lesquels la réalisation de mon travail aurait été difficile.

Merci à tous.

Introduction

Depuis des millions d'années, l'Homme vit dans un univers dont il essaie de comprendre le fonctionnement et de connaître la composition.

Dans les années cinquante, des physiciens européens établissent les plans d'un accélérateur de particules qui permettrait de mieux comprendre l'univers. Le CERN était né.

Actuellement, des millions de données ont été enregistrées et analysées. Elles ont permis la découverte de nouvelles particules.

Toutes ces données sont traitées grâce à l'aide d'un outil très utile : l'informatique. Cet outil permet d'analyser les données et aide considérablement le physicien dans l'élaboration de ces théories.
Les chercheurs font donc un travail de bonne qualité s'ils peuvent compter, entre autres, sur des systèmes informatiques performants et fiables.

Les dirigeants de l'une des quatre grandes expériences du CERN, ALEPH, veulent utiliser un nouveau type de matériel informatique en plus de celui en service actuellement.

La mission qui m'a été confiée était l'intégration de ce nouveau type de matériel au sein de l'expérience. J'étais également en charge de trouver une solution pour stocker de façon minimale toutes les données utilisées par ALEPH parmi les différents environnements de travail.

Le lecteur pourra découvrir dans les pages qui suivent comment l'intégration de ce nouveau type de matériel s'est réalisée et comment le problème de stockage des données en vue de leur utilisation a été résolu. Le lecteur pourra également trouver quelques réflexions sur mon expérience au CERN et ce que ce travail m'a apporté.

Abstract

Since millions of years, Man lives in a universe whose functioning and composition he tries to undertsand .

In the decade of the 50's, some European physicists set up a plan for an accelerator of particles which would allow them to better understand the universe. CERN was born.

Currently, millions of events have been recorded and analysed. They have allowed the discovery of new particles.

All this data are processed with the help of a useful tool : computing. This tool permits the physicist to analyse data and helps significantly in the development of his theories.

The researchers do high quality work, if, among other things, they can count on high-performance and reliable computing systems.

The managers of ALEPH, one of the four big experiments at CERN, want to use a new type of computing equipment in addition to those which are used at the moment.

The mission which has been given to me, was to integrate this new type of equipment inside the experiment, and to find an optimal solution to store the data which will be used by ALEPH in the different environement of work.

The reader will discover in the following pages how the integration of this new type of equipment has been done and how the problem of storing data has been solved. The reader will also find some thoughts on my experience at CERN, and what the work has brought to me.

Sommaire

I Le contexte du CERN

1 - Le CERN

a - Présentation

Nom	Laboratoire Européen pour la physique des particules (acronyme : Centre Européen pour la Recherche Nucléaire)
Adresse	1211 Genève 23 Suisse
Implantation	sur deux sites : Meyrin (Suisse) et Prévessin (France)
Date de création	1953
But	**La Recherche à caractère purement scientifique et fondamental pour :** • permettre à l'Europe de rattraper son retard dans le domaine de l'atome • démontrer que l'atome peut avoir des applications pacifiques • éviter la fuite des cerveaux vers les USA et l'URSS
Etats membres	Allemagne, Autriche, Belgique, Danemark, Espagne, Finlande, France, Grèce, Norvège, Pays-Bas, Pologne, Portugal, Royaume-Uni, Suède, Suisse
Budget	calculé au prorata du PNB de chaque état membre pour 1992 : 909 millions de francs suisses
Personnel CERN	3200 personnes sur les sites de Meyrin et Prévessin, composé d'ingénieurs, techniciens, personnel de maîtrise, secrétaires et personnels administratifs
Personnel rattaché au CERN	7000 personnes de tous pays dont 6000 sont des visiteurs (physiciens) et 1000 des attachés ou boursiers. Le CERN concentre 53% de l'ensemble mondial des physiciens travaillant sur l'atome.
Directeur général	Carlo Rubbia (Prix Nobel de physique en 1984)

b - Historique

1953	Fondation sous l'impulsion d'un groupe de scientifiques européens Etats membres : Allemagne, Belgique, Danemark, France, Grèce, Italie, Norvège, Pays-Bas, Royaume Uni, Suède, Suisse, Yougoslavie Projet de construction d'un accélérateur à protons de 26 GeV : Proton synchrotron (PS)
1954	Implantation sur le site de Meyrin près de Genève (Suisse)
1954	Construction d'un synchrocyclotron (SC), accélérateur de protons d'une énergie de 600 MeV, pour former les physiciens et les ingénieurs et tester le matériel mis au point pour l'accélérateur à 26 GeV
1959	Démarrage du Proton-Synchrotron (PS) , accélérateur de protons à 28 GeV . Première expérience un an après
1971	Début de l'étude de la collision de deux faisceaux de protons provenant du PS à l'aide des ISR (Intersecting Storage Rings). En 1972 première collision (première mondiale)
1971	Projet de construction d'un accélérateur de protons et d'antiprotons à 450 GeV : Super Proton Synchrotron (SPS).
1976	Démarrage du Super Proton Synchrotron (SPS). Utilisé pour accélérer des protons
1981	Lancement du projet de réalisation d'une grande machine à collisions de faisceaux électrons-positons : le LEP (Large Electron Positron collider : 27 km de long, 50 GeV) Début de l'utilisation du SPS comme machine à collisions proton antiproton
1983	Découverte des particules W et Z, et mise au point opérationnelle de la technique du refroidissement stochastique que récompensera en 1984 le prix Nobel de Physique à Carlo Rubbia et à Simon Van Der Mer
1989	Démarrage du LEP le 13 Juillet à 24h00

c - Les accélérateurs du CERN

La physique expérimentale au CERN se concentre aujourd'hui autour du collisionneur électrons-positons LEP (Large Electron Positron collider), anneau souterrain circulaire d'environ 27 kilomètres de circonférence. Les faisceaux d'électrons et de positons, leur antiparticule, y tournent en sens opposé à une vitesse très proche de celle de la lumière et y sont assujettis à des collisions en des points d'expériences déterminés. Ces expériences sont à l'heure actuelle au nombre de quatre (ALEPH, DELPHI, L3 et OPAL), réparties symétriquement sur l'anneau. Chacune possède un puissant détecteur de particules d'une taille comparable à celle d'un immeuble de quatre étages et un système qui enregistre et analyse les fragments de matière apparus.

Les faisceaux pour le LEP ainsi que pour d'autres expériences sont fournis par une suite d'accélérateurs interconnectés, construits entre 1959 et 1976.

Voici les caractéristiques des trois accélérateurs les plus utilisés au CERN :

Nom	Date de construction	Longueur	Types de particules	Energie Maximale*
PS	novembre 1959	628 m	proton-proton proton-antiproton électrons-positons ions lourds	28 GeV
SPS	1971	7 km	proton-proton proton-antiproton ions lourds	450 GeV
LEP	septembre 1983	27 km	électrons-positons	50 GeV

d - Les productions du CERN

Le complexe PS (dix machines) est le seul à produire des particules.

Le SPS effectue essentiellement des collisions sur des cibles externes de protons et d'ions lourds que lui fournit le PS. Il peut aussi collisionner un faisceau de protons avec un faisceau d'antiprotons.

* Les chiffres donnés sont les valeurs des énergies de chaque particule entrant en collision. On parlera donc pour le LEP d'un choc 50 GeV--50GeV.

Le LEP est une machine à collisions de faisceaux d'électrons avec des faisceaux de positons.

Préparations des faisceaux pour le LEP :

Aujourd'hui, PS et SPS sont tous deux utilisés pour préparer les faisceaux destinés au LEP : électrons et positons sont d'abord pris à 600 MeV à la sortie du LPI (LEP PreInjector) qui est un accélérateur linéaire de 600 m de long. Puis, le PS les accélère à 5 Gev avant de les céder au SPS qui les propulse à 20 GeV. Ces particules sont alors prêtes pour l'injection dans le LEP.

Dans le LEP, un millier de gros aimants maintiennent les faisceaux d'électrons et de positons dans des orbites précises centrées dans le tube en alliage d'aluminium où le vide est présent, pendant qu'un équipement de radiofréquence amène les faisceaux à des énergies de plus de 50 GeV.

L'installation complète du LEP représente 600 tonnes d'équipement sur 80 km^2.

Des dispositifs électrostatiques gardent séparés les faisceaux d'électrons et de positons pendant les phases d'injection et d'accélération. Une fois que les particules ont atteint leur énergie optimale, on fait entrer en collision les faisceaux aux quatre points expérimentaux. Dans ces zones expérimentales, afin d'augmenter le taux de collision, les faisceaux sont fortement compressés sous l'effet d'aimants quadripôles supraconducteurs.

e - La recherche au CERN

La recherche en physique des particules à haute énergie se concentre sur les quatre zones d'expériences du LEP : ALEPH, DELPHI, L3 et OPAL. Toutefois, le PS et le SPS, en plus de leurs fonctions d'injecteur pour le LEP, conservent une activité expérimentale pour des collisions et des études à plus basse énergie, par exemple, LEAR (Low Energy Antiproton Ring) et ISOLDE (Isotope Separator On-Line). En effet, l'état de la recherche actuelle en Physique nucléaire ne permettant pas de prévoir si la prochaine découverte viendra des résultats d'expériences à haute ou basse énergie, le CERN conserve ces deux axes de recherche.

Pour l'avenir, le CERN envisage la réalisation du projet LHC (grand collisionneur de hadrons) consistant à installer en anneau de puissants aimants supraconducteurs dans le tunnel du LEP pour faire entrer en collision des faisceaux de protons atteignant 8 TeV (8000 GeV).

2 - Division ECP

Le CERN compte quatorze divisions parmi lesquelles la division ECP (Electronics and Computing for Physics) qui emploie 315 personnes à plein temps.

La division ECP se sépare en deux branches : les sous-divisions Electronics et Computing.

- Electronics contient sept groupes :
  - trois groupes de conception en électronique
  - un groupe de micro-électronique
  - un groupe qui se consacre à l'optoélectronique
  - un groupe qui s'occupe de la CAD et d'infrastructures atelier
  - un groupe d'électronique

Constituée en majorité de physiciens et d'ingénieurs, cette branche a la charge de concevoir et de réaliser les infrastructures existant entre les détecteurs et le stockage sur bande magnétique.

- Computing rassemble cinq groupes :
  - un groupe sur les techniques de programmation
  - un groupe sur les architectures de lecture
  - un groupe pour les systèmes d'acquisition de données
  - un groupe de simulation, reconstruction et analyse
  - un groupe de support technique

De composition plus hétérogène (en terme de formation), la branche Computing s'occupe de toute conception et développement attenant à l'acquisition de données : digitalisation, processeurs VME, enregistrement et analyse de données. Le centre de calcul et la gestion réseau sont confiés à la division CN (Computing and Networking division).

Au sein d'un groupe, les membres sont soit directement attachés à des expériences, soit impliqués dans des projets dans lesquels ils sont souvent en collaboration avec d'autres groupes, d'autres divisions ou des utilisateurs extérieurs.

3 - L'acquisition de données au CERN

a - Quelles données acquiert-on ?

Les signaux bruts recueillis à la sortie des détecteurs se présentent sous forme d'impulsions électriques caractéristiques des événements retenus par le système d'acquisition de données. Ces événements correspondent aux interactions entre particules dans les zones expérimentales d'un accélérateur de particules. Ces signaux seront mis en forme puis digitalisés par différents "étages" d'électronique avant d'être saisis par un système informatique.

Les signaux électriques proviennent, en fait, de chacun des sous-détecteurs de l'expérience et permettent d'identifier les particules en masse, charge et moment.

b - Définition d'un système d'acquisition de données

A l'aide de ces données, on reconstruit les événements intéressants pour pouvoir les analyser. Ces données constituent les entrées du système d'acquisition de données.

Un système d'acquisition de données est constitué de différents modules, chacun requérant un certain degré d'indépendance tout en exigeant un haut niveau de coordination au sein de l'expérience. De plus, quelque soit sa complexité, chaque module doit posséder une interface utilisateur.

c - But et caractéristiques principales

Le but premier d'un système d'acquisition de données en physique des particules de haute énergie est de collecter des données issues des sous-détecteurs et, de les compresser, de les filtrer puis de stocker les événements sélectionnés pour une future analyse en différé. Ceci requiert une efficacité maximale ainsi qu'un contrôle permanent et vigilant de la validité des données acquises et de la performance du dispositif.

Le deuxième but est la surveillance du détecteur et de ses sous-détecteurs, ainsi que des signaux émis et reçus.

Par ailleurs, à cause de l'importance des débits de production des événements donc des données, un système d'acquisition de données doit maintenir un haut degré de parallélisme et un système temps réel, en utilisant autant que possible :

- Une architecture en pipeline
- Le stockage des événements dans des mémoires tampons

Systèmes multitâches

Un tel système peut exécuter plusieurs programmes à la fois.

En commutant d'une tâche (ou processus) à une autre plusieurs fois par seconde, les différents processus (analyse, contrôle, stockage, ..) au sein du système d'acquisition pourront être actifs simultanément sur leurs propres traitements.

Temps réel

La saisie des événements et leur enregistrement pour une analyse en différé doit essayer de suivre leur rythme d'arrivée, c'est-à-dire de production.

Par ailleurs, l'analyse de ces événements en ligne doit être rapide. Toutefois, ce temps d'analyse en ligne n'est pas déterminant car les événements sont automatiquement stockés et pourront donc être analysés en différé.

Le "deadtime" ou temps mort est le temps pendant lequel l'électronique ou l'informatique est incapable de traiter un événement. Il est à la fois dû au temps de réaction du système informatique, au signal de déclenchement d'un événement et au temps de lecture d'un événement.

Durant la saisie de l'événement **n**, si un événement **n+1** arrive, les données lues pour l'événement **n** sont surécrites, donc incohérentes, et par conséquent inexploitables.

En tenant compte de toutes ces contraintes de temps, un bon système d'acquisition temps réel devra minimiser le temps mort et assurer la cohérence des données associées à un état.

Systèmes non déterministes : accumulation d'événements

Au CERN, l'important dans l'acquisition de données est d'accumuler des statistiques donc d'obtenir un maximum d'événements. Il ne s'agit donc pas d'un échantillonnage au cours duquel on serait assuré d'acquérir des événements régulièrement dans le temps (une période d'échantillonnage) mais d'une prise discontinue d'événements en suffisamment grand nombre. Nous avons à faire à de la physique non déterministe.

Environnement matériel et logiciel complexe et hétérogène

Un système d'acquisition de données est constitué de différents sous-systèmes interconnectés. Chaque sous-système requiert un environnement matériel et logiciel adapté et souvent complexe (selon les exigences techniques de l'expérience).

Cette diversité requiert l'usage d'interfaces appropriées qui, selon les cas, sont disponibles sur le marché ou sont développées par le CERN.

Adoption de normes

Afin de minimiser l'hétérogénéité et de pouvoir y faire face dans les meilleures conditions, l'utilisation de standards est encouragée, que ce soit au niveau logiciel (systèmes d'exploitation, langages, protocoles de communication ..) ou matériel (microprocesseurs, bus réseaux...).

Adaptation aux besoins des expériences

Chaque expérience est unique. Elle a des caractéristiques physiques propres telles que :

- Fréquence d'arrivée et taille des événements
- Régime de production des collisions : avec ou sans burst
- Type de recherche pratiquée : acquisition de données pour un phénomène connu voire bien connu, ou recherche pure ou aveugle de nouvelles particules

donc des besoins matériels et logiciels spécifiques.

d - Les phases d'une acquisition de données

Aujourd'hui :

- les détecteurs peuvent générer des données brutes de plusieurs mégabytes par événement.
- Les supports modernes de stockage peuvent enregistrer jusqu'à plusieurs mégabytes d'informations par seconde.

Mais :

- Les capacités de calcul de l'analyse en différé sont limitées par la technologie et l'économie.

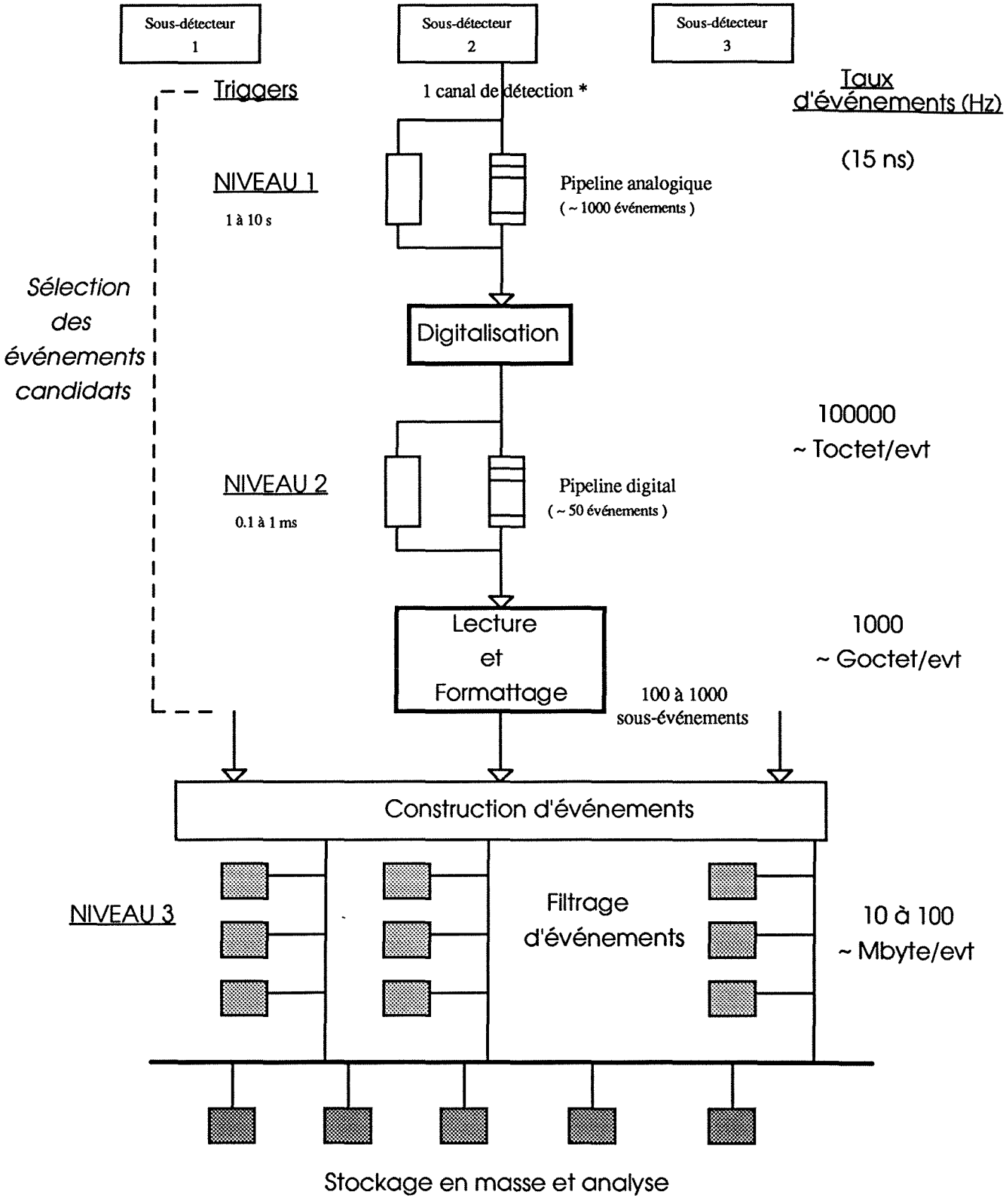
C'est pourquoi il apparaît essentiel de pratiquer autant que possible la compression des données, le formatage et le filtrage des événements avant d'enregistrer les données à des tailles et des taux acceptables.

Structure d'un système d'acquisition de données

La plupart des systèmes d'acquisition utilisés en physique des particules de haute énergie ont une structure arborescente. Les données sont tout d'abord collectées en parallèle à travers des branches reliées à des sous-détecteurs. Elles sont ensuite compressées, filtrées puis regroupées (les différentes branches d'acquisition se rejoignent) avant d'être à nouveau filtrées et compressées pour être finalement enregistrées au niveau du "tronc" de l'arbre. Les différents étages de cette acquisition sont eux-mêmes organisés en structure de type pipeline, farm et event builder. Chaque étage a pour but de réduire la taille et/ou la fréquence des événements à des valeurs compatibles avec les possibilités d'acquisition de l'étage suivant. Par extension, un tel étage d'acquisition est appelé "détecteur de niveau N".

Un modèle d'acquisition de données est proposé dans un schéma page suivante :

Modèle d'acquisition de données (LHC)



*au total 1 à 100 millions de canaux de détection

Remarques :

- taille et fréquence des événements sont des fonctions de l'expérience.
- Un système d'acquisition est constitué de plusieurs étages. Ces étages correspondent à un filtrage des événements intéressants à analyser qui est de plus en plus sélectif à mesure que l'on s'éloigne des détecteurs. La remarque concerne le fait que les possibilités d'acquisition de chaque étage sont directement liées à la complexité de l'algorithme de filtrage utilisé.

Au sein de tout système de lecture d'événements, on utilise une mémoire tampon à chaque fois qu'un processeur, ou une rangée de processeurs travaillent sur des données, tout particulièrement avant la gestion d'une branche d'acquisition d'un sous-détecteur, la construction d'un événement et son enregistrement en tant que donnée exploitable.

On espère en retour minimiser les temps morts.

Au sein d'un système d'acquisition de données, on aura donc besoin, près du détecteur, de nombreux processeurs dont la caractéristique principale soit leur grande vitesse de réaction suite à un trigger alors que l'on aura besoin, dans le tronc, de peu de processeurs plus longs à la réaction d'un trigger mais munis d'une grosse puissance de calcul pour pouvoir traiter des algorithmes complexes.

II L'expérience ALEPH

1 - Description de l'expérience

a - Qui est ALEPH ?

ALEPH est l'une des quatre grandes expériences du CERN qui est positionnée sur le LEP. Le LEP est situé dans un tunnel circulaire de 8.5 kilomètres de diamètre à une profondeur qui varie entre 50 et 150 mètres (ceci est dû au relief montagneux et de façon à être dans un même plan). Les électrons et positons circulent en sens opposé. Les paquets de particules sont dirigés pour se rencontrer à chaque point d'expérience, avec des paquets qui se croisent toutes les 23 micro-secondes. Actuellement l'énergie maximale des faisceaux du LEP est de 60 GeV, mais en fait l'énergie exploitée jusqu'à maintenant est de 45.6 GeV. La prise de données de l'expérience ALEPH a démarré en juillet 1989.

b - Quel est son rôle ?

Le rôle d'ALEPH est de déterminer, aussi précisément que possible, la nature, direction et énergie de chaque particule créée lors d'une collision entre un électron et un positon. Le nombre de particules créées varie selon chaque collision, mais la moyenne est de 30.

c - Quelques chiffres

Les dimensions du détecteur sont environ de 12 x 12 x 12 mètres, son poids de 3000 tonnes et son coût aux alentours de 250 millions de Francs.

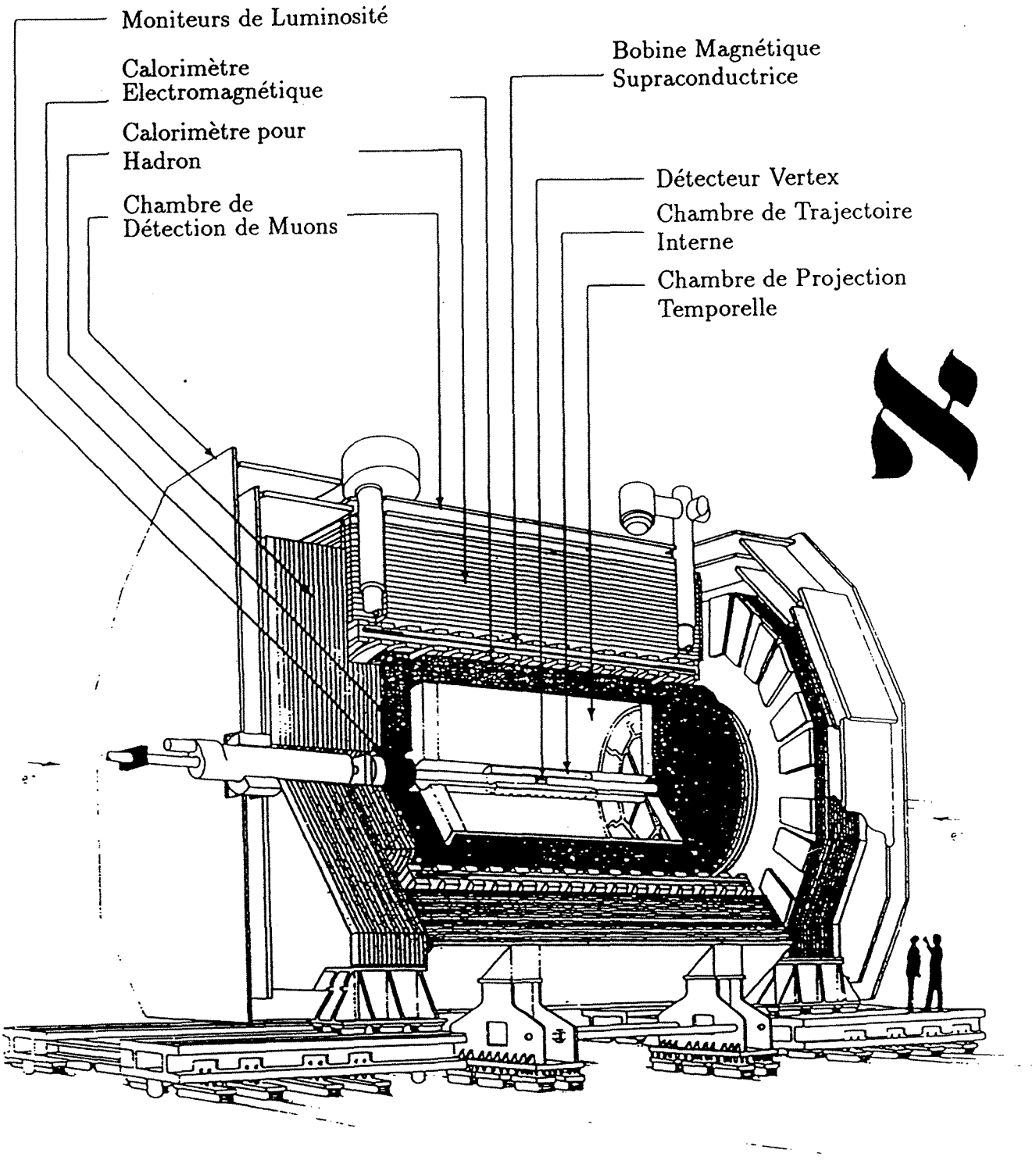
ALEPH est un bel exemple de coopération internationale puisque 400 personnes de 14 nationalités différentes travaillent sur cette expérience.

d - Description technique du détecteur

Les particules sont détectées dans six couches différentes. Les trois couches internes (détecteur Vertex, chambre de trajectoire interne et chambre de projection temporelle) mesurent les trajectoires des particules chargées. Leur trajectoire sont incurvées par le champ magnétique de la bobine supraconductrice, permettant la mesure des énergies des particules. Les particules neutres, n'ayant pas de charge électrique, ne laissent pas de trace. Leurs énergies sont mesurées dans deux couches de calorimètres (calorimètre électromagnétique et calorimètre pour hadron) qui entourent les trois premières. Parmi toutes les particules chargées connues, seuls les muons ont assez de pouvoir de pénétration pour traverser l'épaisseur de 1.2 mètre de la couche du support magnétique. Ils sont identifiés dans la chambre à muons, qui constitue la dernière couche du détecteur.
(voir schéma du détecteur ALEPH page suivante)

e - L'informatique dans l'expérience ALEPH

La partie *informatique* d'ALEPH est composé de deux groupes : ONLINE qui s'occupe de l'acquisition des données et OFFLINE qui s'occupe plus particulièrement de fournir et maintenir des logiciels pour utiliser les données recueillies.



Le Detecteur ALEPH

2 - Online

Le détecteur ALEPH est composé de 700 000 canaux électroniques, et l'acquisition de données est supposée générer plus de 500 mégabytes de données par seconde. Lors d'une collision de deux faisceaux de particules, le détecteur ALEPH et ses sous-détecteurs envoient des signaux électriques qui sont captés par des microprocesseurs. Les signaux provenant de chaque canal sont digitalisés à chaque fois qu'un trigger est produit et cette opération doit permettre de générer 500 mégabytes de données brutes par seconde. Un système appelé "Readout" doit réduire ce volume de données pour éliminer l'information non intéressante, pour collecter les digitalisations et les formater dans des structures de données convenant pour de futures analyses. Ces données sont stockées sur des disques.

Les données enregistrées par l'expérience ALEPH qui ont passées les niveau 1, 2 et 3 des triggers (voir figure page 16) sont écrites sur l'un des quatre disques prévus à cet effet. Quand un disque est plein, le flux de données en cours est terminé, les données sont copiées sur cassette, et le contrôle du disque récemment rempli est passé du système d'acquisition de données à l'outil d'analyse de données appelé *FALCON*.

3 - Offline

Depuis l'enregistrement des données jusqu'aux étapes finales d'analyse, que ce soit en mémoire, sur disque, ou sur bande, toutes les informations associées à un événement sont stockées dans une structure appelée "BOS banks". Le système BOS, qui gère la mémoire, simplifie l'entrée et la sortie des données, et les structure sur le modèle entité-relation. Cela permet de garder l'information ensemble dans des paquets associés logiquement qui peuvent être accédés de façon indépendante, et ajoutés ou enlevés d'un enregistrement d'un événement.

FALCON est une grappe constituée de 10 VAXstations. FALCON exécute le programme de reconstruction des données ALEPH, appelé JULIA, qui détermine la majorité des trajectoires et exécute la reconstruction calorimétrique, indispensable à l'analyse physique. Les sorties de JULIA restent sur disque après cette étape.

Après qu'un flux de données soit complètement traité, les sorties de JULIA sont copiées sur le système CERNVM et sur la grappe ALWS.
Une procédure sur CERNVM enlève les événements qui ne sont pas utiles pour des analyses futures, soit parce qu'ils ne contiennent pas de trajectoires et/ou d'énergie, et un DST (Data Summary Tape) est créé. Un résumé de chaque événement rejeté est gardé sur les DST. Quelques "banks" utilisées dans la reconstruction sont enlevées pour réduire la taille des événements. Parfois, les DST sont concaténés dans des fichiers contenant plus d'un flux de données.

Quand un DST est produit, un autre fichier appelé "event directory", ou EDIR, est aussi créé. L'usage d'un "event directory" permet une sélection et un accès efficaces des événements par flux de données, numéro d'événement ou classification physique.

Plus tard les DST sont encore filtrés pour produire des MINI-DST. A ce moment là, aucun événement n'est éliminé, mais les données sont intégrées, compressées et les "banks" additionnelles utilisées dans la reconstruction sont détruites.

La simulation de données va de paire avec son analyse. Les interactions simulées de particules dans ALEPH est une procédure à deux étages. Tout d'abord, les particules d'état initial sont sélectionnées selon les spécifications de l'utilisateur par le générateur d'événement appelé KINGAL. KINGAL lui-même contient des douzaines de générateurs individuels pour des processus physiques variés.

Le programme GALEPH combine des informations cinématiques avec la description du détecteur pour produire des digitalisations simulées. GALEPH peut simuler toutes les configurations passées du détecteur. Les données simulées

produites par GALEPH peuvent être lues de façon transparente par les programmes qui traitent aussi de vraie données.

L'outil primaire pour l'analyse physique dans ALEPH est le package ALPHA. Cet outil permet la manipulation de trajectoire et objets reconstruits, objets MONTE-CARLO et contient un large nombre de routines utiles pour la manipulation de données. Presque tous les outils d'analyse d'ALEPH sont conçus pour s'exécuter à l'intérieur de l'environnement ALPHA, bien qu'il soit possible d'analyser des données ALEPH avec un programme écrit par l'utilisateur.

Enfin la trajectoire et l'énergie des particules peuvent être visualisées grâce au logiciel DALI. (voir annexe page 93)

L'activité du groupe OFFLINE peut-être visualisée par un schéma décrit en annexe page 67.

III L'environnement de travail

1 - Description du travail effectué dans le groupe OFFLINE

a - Architecture du monde informatique utilisé dans le groupe

Le groupe utilise comme matériel :

- un IBM 9000 fonctionnant avec le système d'exploitation VM
- un CRAY XMP fonctionnant avec le système d'exploitation UNICOS
- un VAX 9000 fonctionnant avec le système d'exploitation VMS
- des stations de travail VAX (modèle 3100 et 4000) utilisant le système d'exploitation VMS
- des stations de travail DEC (modèle 5000) utilisant le système d'exploitation ULTRIX
- des ordinateurs SILICON GRAPHICS (340 S) fonctionnant avec le système d'exploitation IRIX

(voir schéma page 65)

Tous ces systèmes informatiques sont interconnectés grâce aux réseaux ETHERNET (câble coaxial), FDDI (fibre optique) et ULTRANET (fibre optique) à l'intérieur du CERN, ce qui permet à partir d'un terminal, de pouvoir accéder à tous les types d'ordinateurs, à condition toutefois d'avoir l'autorisation d'accès à la machine que l'on veut atteindre.

b - Utilisation des ressources informatiques

Les données qui sont utilisées par le groupe OFFLINE sont stockées sur des disques IBM, sur des cartouches ou sur les disques qui appartiennent à la grappe des VAXstations du groupe, appelée ALWS. Les différents logiciels qui sont maintenus par le groupe, sont installés sur cinq plates-formes différentes qui sont IBM, CRAY, VAX, DEC, et SILICON GRAPHICS. Comme l'ensemble de ces plates-formes n'utilise pas le même système d'exploitation, cela implique que les accès aux données ne vont pas se faire de la même manière. Il serait donc

intéressant de pouvoir stocker ces données en un minimum d'endroit afin d'économiser de l'espace disque et avoir moins de maintenance.

Le groupe OFFLINE utilise depuis l'existence de l'expérience ALEPH, le matériel IBM, CRAY et VAX. Afin d'avoir un peu plus d'autonomie, le groupe s'est doté de stations de travail d'un type nouveau, les DECstations, utilisant un autre système d'exploitation. Il faut donc apprendre à maîtriser ce nouveau matériel pour l'utiliser au maximum de ses performances.

Le groupe souhaite également s'intégrer rapidement au projet SHIFT qui utilise le matériel SILICON GRAPHICS, car il est conçu pour remplacer le CRAY qui arrive à la fin de ses jours au CERN (fin prévue pour avril 1993). En effet, le CERN a décidé de ne pas renouveler le contrat pour l'utilisation du CRAY car celui-ci était trop onéreux à utiliser et à maintenir.

2 - Description de mon travail au sein de ce groupe

Le travail que je dois effectuer au sein de ce groupe est de développer un logiciel système pour un serveur de données qui est utilisé simultanément par un système UNIX et VMS. L'ensemble du système comprend environ 100 stations de travail, 100 gigabytes d'espace disque. Une connexion au projet SHIFT est à prévoir. Les données sont utilisées pour faire des analyses physiques dans l'expérience ALEPH.

3 - Objectifs de mon travail

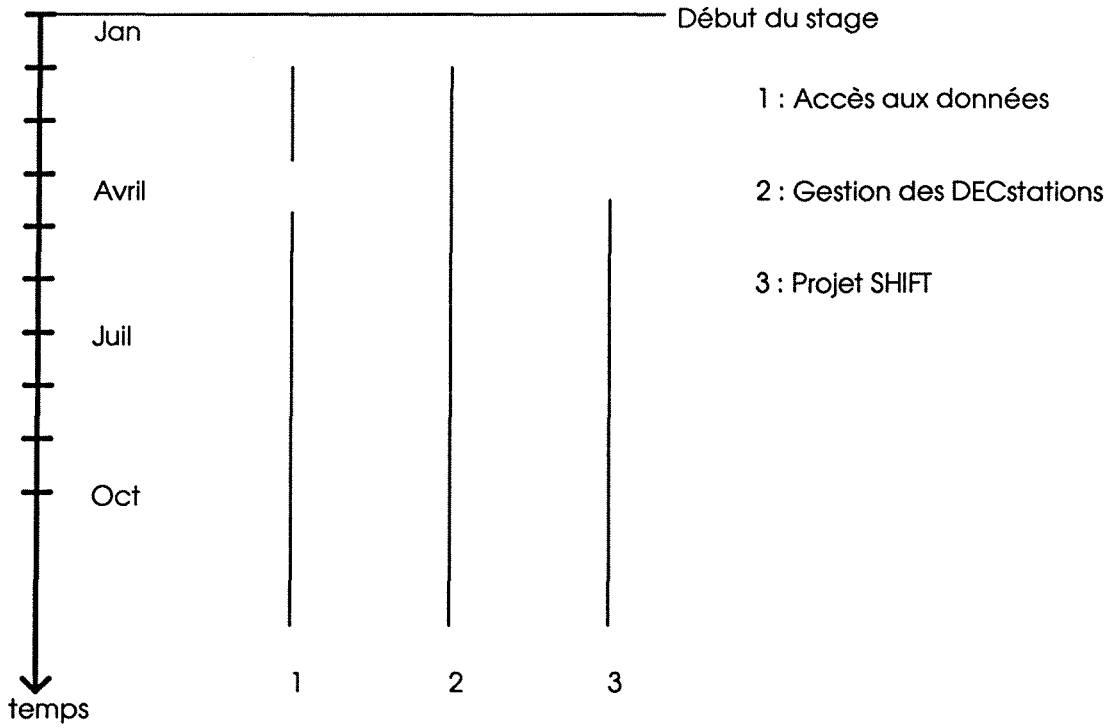
Trois objectifs principaux avaient été fixés à mon arrivée dans le groupe :

1. L'accès aux données pour l'expérience ALEPH, quelque soit le lieu de leur implantation au CERN

2. Mise en place pour le groupe OFFLINE d'un ensemble de stations de travail (DECstation) fonctionnant avec le système d'exploitation ULTRIX et installées en réseau

3. Intégration d'ALEPH dans le projet SHIFT.

Planification de mon travail dans le temps pour mes trois objectifs :



4 - Moyens mis à ma disposition

a - Moyens matériels

Lors de mon arrivée dans le groupe OFFLINE, le chef de groupe a mis à ma disposition un bureau avec une station de travail (DECstation modèle 5000/200) fonctionnant avec le système d'exploitation ULTRIX, ainsi qu'un téléphone. En effet , le téléphone est un outil indispensable pour la communication surtout sur un lieu de travail qui s'étend sur plusieurs centaines de mètres. Il en est de même avec le courrier électronique qui permet de dialoguer avec n'importe quel utilisateur ayant un compte sur une machine informatique.

Un certain nombre de comptes ont été ouverts sur différents systèmes informatiques fonctionnant au CERN. Ces différents comptes me permettent d'accéder à partir de ma station de travail à différents services offerts au CERN et utiles dans mon travail.

b - Moyens humains

Dans le groupe OFFLINE, j'étais sous l'autorité du responsable logiciel, Monsieur KNOBLOCH, qui est également mon maître de stage pour mon mémoire industriel de fin d'études. Dans le bureau où l'on m'avait placé, j'étais en compagnie du responsable de la maintenance du système des DECstations. J'étais en contact permanent avec tous les responsables des différents logiciels qui sont maintenus dans l'expérience ALEPH ainsi qu'avec les utilisateurs des DECstations et des données.

J'avais des contacts privilégiés avec les différents ingénieurs systèmes et ingénieurs réseaux responsables des divers ordinateurs utilisés par l'expérience ALEPH.

IV Réalisation et prospectives

1 - Accès aux données

a - Etude d'opportunité

a-1 Où sont situées les données ?

Les données utilisées par le groupe OFFLINE, sont actuellement stockées à plusieurs endroits :

- cartouches accessibles par un robot géré par le centre de calcul
- disques attachés à l'IBM du centre de calcul
- disques des VAXstations de la grappe ALWS gérés par le groupe OFFLINE

Tous les supports de stockage appartiennent au groupe mais il n'en contrôle pas toute la gestion.

a-2 Analyse des besoins

Le groupe, acquérant et utilisant de nouveaux types de matériels comme les DECstations et SILICON GRAPHICS, souhaiterait utiliser les données déjà stockées sans avoir à les copier sur de nouveaux supports.

Il faut donc étudier l'accès aux données par rapport à leur lieu de stockage et d'utilisation. Cette étude portera sur le temps d'accès et la compatibilité entre le lieu de stockage et le lieu d'utilisation.

b - Etude de faisabilité

b-1 Compatibilité des données

Les données sont enregistrées dans un format standard appelé EPIO, lisible et décodable pour tous les types d'ordinateurs. EPIO est un format binaire (il

utilise le standard IBM), qui permet d'éviter les problèmes liés aux définitions de flottant et d'entier sur les différents types d'ordinateur.

Ce standard permet une grande liberté dans le choix du matériel à utiliser pour le stockage.

b-2 Etude de temps d'accès aux données

Pour connaître l'endroit où il serait judicieux de stocker ces données, il va falloir établir une liste des différentes possibilités de stockage ainsi qu'une liste des endroits à partir desquels vont être utilisées ces données. Ensuite il serait intéressant de connaître les différents temps d'accès aux données pour les multiples combinaisons proposées.

La liste des possibilités de stockage serait la suivante :

- disque des VAXstations
- disque des DECstations
- disque des SILICON GRAPHICS (projet SHIFT)

La liste des lieux d'utilisation de ces données serait la suivante :

- VAXstations
- DECstations
- stations de travail SILICON GRAPHICS

Dans le même temps, il est intéressant de tester les différents types de connexion entre ces trois systèmes. Au début, ces derniers étaient connectés à travers le réseau ETHERNET. Puis la solution de la fibre optique est apparue avec ses avantages et ses inconvénients.

Les différents tests nous ont donné le tableau suivant :

Site du fichier	Site de la CPU	Type de transport	Taux de transfert (kbytes / s)
dxal10	dxal10	local	1350
dxal10	dxal01	FDDI	883
dxal10	dxal01	ethernet (m branche)	599
dxal10	dxal03	ethernet (un pont)	545
dxal10	dxal05	ethernet (plusieurs ponts)	301
alws	dxal10	ethernet	141
SHIFT	dxal10	FDDI (RFIO)	1100
alws	dxal10	ethernet (RFIO)	360

dxal.. : DECstation alws : VAXstation SHIFT : SILICON GRAPHICS

Les résultats rassemblés dans le tableau ci-dessus, nous permettent de constater qu'il y a une grande différence entre les performances du réseau ETHERNET et FDDI. L'autre remarque qui s'impose est que les performances relevées avec RFIO sont meilleures pour les deux types de réseau que sans RFIO.

Qu'est-ce que RFIO (Remote File Input Output) ?

C'est un package qui permet de travailler sur des fichiers de données sans que ceux-ci soient directement visibles sur la machine où l'on travaille. On fait les manipulations sur les fichiers à travers le réseau. Ce package a été créé pour être utilisé surtout avec le réseau FDDI qui permet un plus grand débit.

b-3 Quelles sont les différentes possibilités envisageables ?

Trois possibilités se présentent à nous :

- Stocker les données sur les disques déjà existants des VAXstations et les rendre visibles pour les autres systèmes
- Stocker les données sur les disques des DECstations mais il faut acheter ces disques pour le stockage
- Stocker les données sur les disques des SILICON GRAPHICS (projet SHIFT) où ALEPH a déjà investi.

b-4 Quelle solution a été choisie et pourquoi ?

La solution à court terme qui a été choisie est en fait un mélange de deux solutions. Les données seront stockées sur les disques des VAXstations et sur les disques des SILICON GRAPHICS. Toutefois, le stockage sera différent entre les deux systèmes, car sur VAX les données sont stockées en permanence; alors que sur SILICON GRAPHICS elles ne seront présentes que lorsque l'utilisateur en aura besoin. En effet, les données seront copiées à partir des cartouches sur les disques durs des machines du projet SHIFT et y resteront un certain temps avant d'être détruites. Cette solution permet de travailler avec moins d'espace disque.

A long terme, on ne veut avoir qu'un seul lieu de stockage. Mais auparavant, la solution mixte doit fonctionner et elle nous donnera une indication précise de la solution finale à adopter.

c - Spécifications techniques

c-1 Description et réalisation du coté ALWS

Sur ALWS, les données sont stockées sur un grand nombre de disques qui sont accessibles grâce à MULTINET, logiciel de gestion de communication à travers le réseau ETHERNET.

Organisation de ALWS :

L'ensemble de la grappe ALWS est composé de:

- 2 bootnodes ayant chacun un disque système
- 14 serveurs desservant chacun sept disques
- 83 VAXstations

(voir schéma page 66)

Cela représente en tout 100 gigabytes d'espace disque, et toutes ces machines sont connectées par ETHERNET et FDDI.

Chacun des serveurs fait fonctionner le logiciel MULTINET.

Présentation général de MULTINET :

MULTINET permet de connecter les stations de travail VAX avec un réseau Internet et autorise la communication avec n'importe quel système d'exploitation qui fait fonctionner le standard TCP/IP, comme UNIX par exemple.

MULTINET offre les possibilités suivantes :

- transférer des fichiers entre le VAX et une autre machine qui utilise TCP/IP à travers le réseau,
- établir une connexion entre le VAX et d'autres machines TCP/IP sur le réseau,
- échanger du courrier électronique avec d'autres systèmes,
- écrire et utiliser des applications sophistiquées pour des communications de processus à processus entre le VAX et d'autres systèmes compatibles à travers le réseau.

Les caractéristiques de MULTINET sont les suivantes :

- Une bande passante haute autorisant ETHERNET à transférer des données à un débit de 10 mégabits par seconde et permettant à un grand nombre d'utilisateurs de se partager le réseau.
- Micro, mini, super-ordinateur et autres systèmes peuvent co-éxister sur le même réseau; Les ordinateurs utilisant le système d'exploitation VMS ne nécessitent pas de protocole réseau séparé.

- Les communications haut débit de programme à programme autorisent des opérations transparentes d'applications distribuées entre des systèmes dissemblables.

Fonctionnement général de MULTINET :

Un certain nombre de fichiers et de bases de données sont utilisés pour faire fonctionner MULTINET sur VAX. Ceux-ci autorisent de nombreuses opérations à travers le réseau et contiennent les noms et adresses de ceux qui sont autorisés à pouvoir accéder ou à être accédés par VAX.

Dans le cas qui nous concerne actuellement, nous allons utiliser le protocole NFS pour dialoguer avec les DECstations. Pour ce faire, il existe un fichier dans le noyau de MULTINET qui s'appelle "nfs.configuration", dans lequel sont enregistrés les disques qui sont autorisés à être accédés et les machines à qui on donne cette autorisation. Ce fichier contient également l'adresse du fichier où sont enregistrés les noms et mots de passe des utilisateurs autorisés à employer ce service (voir exemple en annexe page 68).
Le fichier "nfs.configuration" est chargé lorsque MULTINET est activé. Donc lorsqu'on effectue des modifications à l'intérieur de ce fichier, il faut forcer MULTINET à relire ce dernier pour que les changements soient pris en compte par MULTINET.

Lieu de stockage des données :

Nous avons vu que les fichiers de données sont situés sur différents disques. Il faut savoir que chaque disque a un nom. Sous le système d'exploitation VMS, les données sont regroupées sous un nom commun (LOGICAL). Dans notre cas particulier, le LOGICAL s'appelle AL$DATA et il regroupe tous les noms des disques où sont stockées les données.
Exemple : "AL$DATA" = "AL1F01$DKB0:[ALEPHDATA]"
= "AL2F03$DKB100:[ALEPHDATA]"
= "AL2F03$DKA500:[ALEPHDATA]"

Les données sont stockées journellement sur les disques où il y a de la place disponible. Il faut donc créer une procédure qui liste les différents disques contenus dans le LOGICAL AL$DATA, qui les compare avec la liste précédente et qui fait les mises à jour nécessaires du fichier de configuration "nfs.configuration" sur tous les serveurs.

Procédure automatique de mise à jour :

La procédure qui exécute toutes les actions décrites précédemment doit le faire régulièrement. Nous avons décidé que la période d'une journée était suffisante pour faire les mises à jour si nécessaire.

J'ai donc créé deux procédures dont une qui s'exécute en traitement par lots à quatre heures chaque matin, heure à laquelle l'activité du système est moindre. Ces deux routines font plusieurs tâches :

- sauver la liste des disques existants déjà
- mettre dans un fichier la valeur des LOGICAL
- créer une nouvelle liste avec les disques contenant actuellement des données
- comparer cette nouvelle liste avec l'ancienne et crée un fichier de mises à jour
- écrire le fichier "nfs.configuration" avec la nouvelle liste
- copier ce fichier de configuration sur tous les serveurs qui font fonctionner MULTINET
- redémarrer MULTINET sur tous les serveurs
- soumettre cette procédure à la queue de traitement par lots pour qu'elle soit de nouveau exécutée le lendemain à la même heure
- tester si le fichier de mises à jour est vide et le détruire si tel est le cas

(voir routine en annexe page 70 et 75)

Toutes les opérations qui sont exécutées, sont enregistrées dans un fichier. Cette sauvegarde permet de vérifier le lendemain que les opérations se sont déroulées sans incident.

c-2 Réalisation du coté DECstation

Du coté des DECstations, les données ne sont pas physiquement présentes mais elles sont visibles pour les utilisateurs de ces stations de travail, grâce au protocole NFS qui peut rendre "visible" des disques appartenant à d'autres systèmes. Ce protocole permet de monter des disques sur les DECstations et de pouvoir ainsi accéder aux données qui s'y trouvent.

Cette opération peut être réalisée grâce aux commandes ULTRIX *mount*, *umount* et *automount*. Elles permettent de monter des disques selon un type de protocole, à un endroit voulu et à partir d'une machine étrangère au système.
Mount est une commande manuelle alors que *automount* est un processus qui fonctionne en permanence sur la station de travail et qui est démarré lors de la mise sous tension de la machine.

La commande AUTOMOUNT :

La commande *automount* utilise des "cartes" qui sont lues lors du lancement du processus. Ces cartes ont la structure suivante :

location-point	options	mount-point

où :
location-point représente le nom du répertoire où vont être montés les disques et où seront visibles les données,
mount-point est le nom du serveur et du disque qui va être monté sur la DECstation,
options permet d'indiquer de quelle manière vont être installés ces disques (lecture seulement, lecture-écriture, temps d'accès déterminé, ...).

Exemple : On veut que les données qui sont dans les répertoires *alephdata* situés sur les disques *dka100, dka200* et *dka500* du serveur *al1f03* soient visibles sur le répertoire */alws*. La "carte" sera comme suit :

```
/alws/al1f03/dka100     -ro     al1f03:/al1f03$dka100/alephdata
/alws/al1f03/dka200     -ro     al1f03:/al1f03$dka200/alephdata
/alws/al1f03/dka500     -ro     al1f03:/al1f03$dka500/alephdata
```

(voir exemple concret en annexe page 77)

La commande automount est un processus qui existe sur chaque station. Par conséquent, si une "carte" change, il faut redémarrer le processus sur toutes les stations pour que le changement soit pris en compte. Pour cela, il faut d'abord "démonter" tous les fichiers systèmes (***umount -a***), tuer le processus d'***automount*** (***kill -9 n°_du_processus***), relancer le processus ***automount*** et "remonter" les fichiers systèmes (***mount -a***).

Procédure automatique :

On s'aperçoit que les "cartes" dépendent des disques et des serveurs des VAXstations. Donc si un changement intervient du coté ALWS, il faut qu'il soit répercuté du coté des DECstations. Il était souhaitable que cela se fasse de façon automatique comme cela se fait du coté des VAXstations.

J'ai donc été amené à écrire une procédure pour exécuter cette mise à jour si la nécessité apparaît. Cette procédure s'exécute chaque jour comme son homologue et après la première. La routine a été placée dans le fichier ***crontab*** du système. Ce fichier est une table qui est lue en permanence par le système et qui exécute les actions qui y sont décrites aux périodes souhaitées.

La routine exécute les tâches suivantes :

- copier le fichier de mise à jour qui se trouve sur ALWS sur le système
- mettre à jour les "cartes" et les "pages jaunes" du serveur si le fichier de mise à jour existe vraiment
- si une mise à jour a été faite le processus automount doit être redémarré sur toutes les machines

- mise à jour des liens sur les données

(voir procédure en annexe page 79)

Les liens sur les données :

Comme on l'a vu précédemment, les données sont dans des fichiers ayant des noms différents. D'après la construction des "cartes", on s'aperçoit qu'il ne va pas être facile à l'utilisateur de trouver un fichier de données s'il ne sait pas sur quel disque il est sensé se situer. C'est pourquoi, il a été décidé de créer un répertoire où se trouveraient tous les noms des fichiers de données. Ces noms de fichier de données étant des liens logiques qui pointeront sur l'endroit où se trouve réellement le fichier. Cela facilitera la tâche des utilisateurs.

Exemple de lien :
Soit le fichier ab1234.epio qui se trouve sur le disque AL1F01$DKB0:[ALEPHDATA]. On aura le lien suivant :
ab1234.epio -> /alws/al1f01/dkb0/alephdata/ab1234.epio

Il a donc fallu écrire un petit programme pour créer ces liens. Celui-ci parcourt l'ensemble des répertoires où ont été montés les disques et dès qu'il rencontre le nom d'un fichier, il crée le lien.

En utilisant le même principe, les données qui se trouvent sur SILICON GRAPHICS sont également visibles sur DECstations.

Remarque : Toutes les commandes qui viennent d'être décrites ne peuvent être utilisées que si on a les droits d'accès du super-utilisateur.

c-3 Réalisation du coté SHIFT

Installation des données :

Nous avons vu que du coté de SHIFT, les données n'étaient montées que temporairement sur les disques. Ces données proviennent des cartouches où elles sont stockées. Lorsqu'un utilisateur désire un fichier de données, il faut aller chercher la cartouche correspondant à ce fichier, l'installer sur le lecteur de cartouche et copier le fichier sur un des disques réservés à cet effet. Il va de soi que cette opération n'est pas effectuée par l'utilisateur mais par le robot de cartouche.

Une procédure a été créée par les responsables du système SILICON GRAPHICS pour aller chercher la bonne cartouche, vérifier qu'il y a de la place

disponible sur les disques réservés aux données et effectuer l'opération si elle est réalisable.
A partir de l'ossature de cette routine, nous avons effectué des modifications pour y mettre nos paramètres par défauts et quelques spécificités du groupe OFFLINE.

Nettoyage des données :

Etant donné que l'espace sur les disques des SILICON GRAPHICS n'est pas suffisant pour pouvoir installer toutes les données, il faut nettoyer ces disques pour qu'il y ait assez de place pour installer d'autres données. Une procédure a été écrite par les responsables du système SILICON GRAPHICS pour nettoyer les disques selon un certain nombre de paramètres comme la date de dernière utilisation d'un fichier, la date de dernière modification, la taille des fichiers, ... Cette procédure a été adaptée à notre cas particulier et elle ne peut être utilisée que par une personne ayant certains droits d'accès.

Liens pour les données :

Comme pour les DECstations, nous allons créer un lien sur les fichiers de données pour faciliter le travail de l'utilisateur. A la fin de l'opération d'installation d'un fichier de données, la procédure renvoie le chemin où a été installé le fichier. A partir de ce chemin, je crée un lien dans un répertoire avec le nom du fichier de données qui pointe sur le disque où se trouve le fichier.

d - Avenir du projet

Le travail que j'avais à effectuer est terminé mais il est incomplet. En effet, les tests réalisés avec le réseau FDDI sur le projet SHIFT ont été faits avec une fibre optique installée entre une DECstation test du centre de calcul et les SILICON GRAPHICS.

Suite à des problèmes administratifs et techniques, les tests n'ont pu être réalisés sur une fibre optique reliant nos propres stations et les ordinateurs du projet SHIFT.

Néanmoins, dès que la connexion sera établie, les procédures de test sont prêtes pour être mises en oeuvre et établir des résultats.

2 - Mise en place de l'ensemble des DECstations

Le travail que l'on m'a demandé de faire dans cette partie était un travail d'aide à la conception. En effet, quand je suis arrivé dans le groupe OFFLINE, la décision d'installer des DECstations en réseau avait été prise et un ensemble de trois machines existait déjà. Des tests avaient été réalisés pour connaître l'intérêt de s'équiper d'un nouveau type de matériel et l'efficacité de celui-ci. Les premiers tests étant concluants, le groupe avait décidé d'augmenter le parc de ses DECstations.

a - Installation du système

a-1 Architecture du système

Le système des DECstations est composé de :

- une station de travail qui fait office de serveur à laquelle sont connectés 10 disques durs d'un gigabyte chacun
- treize stations de travail qui sont clientes du serveur et qui ont chacune un disque local d'un gigabyte
- un lecteur de cassettes (Exabyte)
- un lecteur de disque compact (CDROM)

Les stations de travail clientes sont connectées au serveur grâce au réseau ETHERNET ou FDDI.

Le serveur est lui-même connecté au réseau ETHERNET du CERN, et peut donc dialoguer avec tous les autres systèmes informatiques présents au CERN, ainsi qu'avec les instituts ou entreprises du monde entier qui ont une connexion vers le monde extérieur.

Ce système est maintenu en état de fonctionnement grâce au travail de l'ingénieur système et de son adjoint, en l'occurrence moi-même durant mon stage.

a-2 Installation du serveur

Le lecteur se doute que le serveur est une pièce essentielle dans le système des DECstations. C'est pourquoi l'installation est une étape importante dans la vie du serveur car elle permet, entre autres, de bien définir son environnement pour son utilisation future.

Installation de la station de travail en autonome :

Lorsqu'un client achète une station de travail, celle-ci est livrée avec son système d'exploitation. En l'occurrence, une DECstation est livrée avec le système d'exploitation ULTRIX, qui est un dérivé d'UNIX.

Une fois que votre station est déballée, vous devez la configurer pour l'utilisation désirée.
La première chose à faire, avant la configuration physique, est de préparer sur papier la configuration voulue, en spécifiant, par exemple, la gestion de la mémoire (taille de la mémoire tampon et de la mémoire réelle), le partage en partition du disque local s'il existe, le nom de la machine, le format de l'heure utilisée, les différents répertoires principaux qui vont être utilisés avec l'allocation qui leur est réservée, les options du noyau du système d'exploitation qui vont être utilisées (les "pages jaunes", le type de sécurité, ..), etc ..

Cette préparation peut être réalisée grâce au guide d'installation d'une station de travail qui montre les différentes étapes de la configuration avec à l'appui des exemples concrets.

Quand la première étape est faite, l'opération de configuration peut commencer et l'ingénieur système qui installe la nouvelle station de travail n'a plus qu'à répondre aux questions que la machine lui pose. A la fin de cette manipulation, la station de travail peut être redémarrée, et elle est prête à fonctionner de façon autonome.

La station de travail devient un serveur :

Maintenant que la station peut travailler, il va falloir continuer les initialisations pour que la station puisse :

- dialoguer avec le monde extérieur
- utiliser différents périphériques
- avoir plusieurs utilisateurs
- etc ...

Pour que la station de travail joue son rôle de serveur, nous allons lui ajouter des disques durs où l'on pourra stocker de l'information qui sera visible par les futurs clients du serveur.

Le serveur a un certain nombre de sorties pour recevoir différents périphériques. A chaque sortie correspond un numéro. Il faut donc numéroter chacun des périphériques, le numéro correspondant à la position d'un switch sur le dit périphérique. Il suffit alors de connecter les périphériques sur les sorties ayant le même numéro et de redémarrer la station de travail pour que les périphériques qui lui ont été adjoints, soient reconnus.

Après avoir installé physiquement les périphériques, nous allons les installer logiquement. Nous créons d'abord le périphérique avec la commande ***Makedev***. (***Makedev /dev/nom_du_périphérique***). Puis nous partitionnons et formatons, si cela est nécessaire, le périphérique avec les commandes ***chpt***, ***newfs*** et ***fsck***. Enfin, nous montons le périphérique pour qu'il soit accessible logiquement (commande ***mount***).

Si ce périphérique est installé de façon permanente, il est enregistré dans un fichier appelé ***fstab***, qui est lu au démarrage du serveur, ce qui permet de monter les différents périphériques de façon automatique. Ce fichier décrit le nom du périphérique, le nom du répertoire sur lequel il est monté , le type de montage (lecture écriture, quota, ..).

Vient ensuite l'initialisation de tous les services concernant le réseau sous ULTRIX (comme NFS qui s'occupe de gérer les fichiers systèmes, les Pages Jaunes (Yellow Pages Services), TCP/IP, BIND, etc ..). Pour faire cette initialisation, il faut tout d'abord que le CERN ait donné à la machine un numéro d'identification pour le réseau, ce numéro est unique, et est reconnu par tous les ordinateurs du monde qui ont un accès au réseau CERN.
Toutes ces initialisations sont écrites dans un fichier qui s'appelle ***rc.local*** qui est lu quand la machine démarre.

Il faut ensuite créer les comptes des utilisateurs et un répertoire leur appartenant où ils pourront travailler. Lors de la création, quelques fichiers d'initialisation de variables d'environnement et de définitions particulières, comme le langage de commande utilisé, sont mis dans son répertoire. Cela s'appelle le "squelette". Quand l'utilisateur se connectera sur la machine avec son nom et son mot de passe, ces fichiers seront exécutés et il sera dans un environnement qu'il pourra modifier par la suite pour le personnaliser, mais c'est le minimum requis.

a-3 Installation d'une station de travail cliente du serveur

L'installation d'une station de travail cliente d'un serveur s'effectue de la même manière que celle d'un serveur à quelques différences près. Tout d'abord, on oblige la station cliente a démarré à travers le réseau sur le serveur pour utiliser son système d'exploitation. Ensuite, on initialise les mêmes services que sur le serveur mais en déclarant la station comme cliente. Enfin, on enregistre la station cliente sur le serveur pour qu'elle puisse bénéficier des services du serveur.

b - Installation des logiciels maintenus par le groupe OFFLINE

Pour installer les différents logiciels sur le système des DECstations, il a fallu penser à une organisation sur le ou les disques pour installer ces divers produits. Il a donc été décidé de dédier un disque d'un gigabyte aux logiciels du groupe OFFLINE. Ce disque a été monté sur le répertoire *aleph* qui se trouve sur le serveur. Ce répertoire a été installé de façon à être accessible par toutes les stations clientes en lecture et en écriture pour certains sous-répertoires.

Architecture du répertoire aleph :

- /aleph (répertoire principal)
  - /bin (répertoire contenant des utilitaires pour le groupe OFFLINE)
  - /dbase (répertoire contenant les bases de données)
  - /data (répertoire contenant les liens sur les fichiers de données qui se trouvent sur les VAXstations)
  - /doc (répertoire contenant de la documentation sur les logiciels)
  - /edir (répertoire contenant les liens sur les fichiers de EDIR qui se trouvent sur les VAXstations)
  - /gal (répertoire contenant les exécutables du logiciel GALEPH)
  - /jul (répertoire contenant les exécutables du logiciel JULIA)
  - /lib (répertoire contenant les bibliothèques utilisées pour le linkage des différents logiciels)
  - /phy (répertoire contenant les exécutables du logiciel ALPHA)
  - /src (répertoire contenant les répertoires des sources de chacun des logiciels utilisés et maintenus par le groupe)
  - /test (répertoire où sont testés les nouvelles versions des logiciels)
  - /tmp (répertoire pour des fichiers temporaires)
  - /uphy (répertoire contenant des petits logiciels très spécifiques)

Les logiciels du groupe :

Tous les logiciels créés et maintenus par le groupe OFFLINE, sont écrits en Fortran 77. Sur les DECstations, il existe deux compilateurs Fortran : le MIPS et le DECfortran. Au début de la vie des DECstations le compilateur MIPS avait été choisi car il était plus performant que l'autre qui était très "vérolé". Mais depuis quelques mois, nous avons changé de compilateur et nous utilisons à présent le DECfortran nouvelle version (version plus fiable). Pourquoi ce changement ? : parce que les bibliothèques qui sont distribuées par le CERN et utilisées par les logiciels de ALEPH sont compilées avec ce nouveau compilateur plus performant que la version antérieure et parce que le premier, le MIPS, n'est plus maintenu par la société DIGITAL.

Ce changement de compilateur nous a obligé à recompiler tous les logiciels, car les deux compilateurs ne sont pas compatibles et les options de compilation changent légèrement.
Il a donc fallu re-tester tous les logiciels et vérifier que les résultats que l'on obtenait, étaient identiques à ceux obtenus précédemment. Ce qui ne fut pas toujours le cas, ceci étant dû aux options de compilation, en particulier les options d'optimisation.

Exemple : La fonction d'ouverture d'un fichier en Fortran est ***OPEN***. L'utilisation de cette option nécessite un certain nombre de paramètres. Avec le compilateur DECfortran, le paramètre READONLY est indispensable pour ouvrir un fichier alors qu'avec le MIPS il n'était pas nécessaire.

c - Formation des nouveaux utilisateurs

La gestion d'un système informatique nécessite aussi de faire de la formation, en particulier pour les nouveaux utilisateurs. Il existe plusieurs types d'utilisateurs : celui qui ne connaît pas UNIX et celui qui a utilisé un dérivé d'UNIX. Il faut donc lui apprendre les commandes de base d'UNIX ou lui donner les commandes équivalentes par rapport au système d'exploitation qu'il connaît.

Les nouveaux utilisateurs veulent aussi connaître l'éditeur qui est utilisé sur le système. En effet, l'éditeur est un outil indispensable pour travailler sur un ordinateur. Il faut donc prendre le temps de passer quelques heures avec eux pour leur montrer ce qu'il faut faire ou ne pas faire.

J'ai également été amené à former l'ingénieur système employé par le groupe OFFLINE et qui me remplacera à la fin de mon stage. Cette formation se situait essentiellement au niveau de l'apprentissage du matériel DIGITAL et de la construction de la grappe de DECstations.

d - Accès aux données par les logiciels

Les logiciels qui sont utilisés par le groupe font appel aux fichiers de données dont on a parlé dans le chapitre précédent. Cette ouverture de fichier se fait, en Fortran 77, grâce à une routine qui s'appelle *open*. Or il s'avère que l'ouverture de fichier ne se fait pas de la même manière selon le type d'ordinateur sur lequel on se trouve.

Il faut préciser que les fichiers de données ont un format un peu spécial (EPIO) qui nécessite des routines spéciales pour ouvrir, lire ou écrire des fichiers de ce type.

Il a donc fallu étudier comment faire une ouverture de fichier sur DECstations en tenant compte de la représentation des données sur DECstation qui n'est pas la même que sur un autre système UNIX.

Pourquoi avoir choisi d'utiliser les fichiers de données par l'intermédiaire d'un lien et non directement ?

La première explication à fournir est que cela facilite la tâche des utilisateurs qui n'ont pas besoin de savoir sur quel répertoire le fichier de données qu'ils veulent utiliser a été monté. La deuxième est que cela permet d'ouvrir un fichier à l'intérieur des logiciels avec un format standard. Ce format est composé du nom du chemin (qui est toujours le même), et du nom du fichier ou plus exactement du nom du lien qui pointe sur le fichier réel.

e - Avenir de ces stations de travail dans le groupe OFFLINE

Ces stations ont comme objectif d'être des outils de travail grâce à leur CPU qui n'est utilisée que par un ou deux utilisateurs et grâce à leur vitesse par rapport à des VAXstations. Il faut noter également que ces stations de travail ont l'avantage de ne pas être chères et d'être performantes.

Par contre, ce nouveau matériel oblige les personnes du groupe OFFLINE à maintenir les logiciels avec du code dépendant de ces nouvelles stations.

Celles-ci ont été installées, configurées et testées pendant mon stage. Elles peuvent démarrer leur activité. Les stations de travail seront maintenues en bon état de fonctionnement grâce à l'ingénieur système que j'ai formé et qui a travaillé avec moi pendant quelques mois.

3 - Intégration de ALEPH dans le projet SHIFT

a - Etude d'opportunité

Comme je l'ai expliqué quelques pages auparavant, l'expérience ALEPH utilise plusieurs systèmes informatiques pour analyser ses données. En particulier, les physiciens utilisent le CRAY pour faire un certain nombre de calculs à cause de sa puissance. Malheureusement, le CRAY est amené à disparaître du CERN en avril 1993, ce qui entraîne une perte de puissance de calcul assez phénoménale.

C'est pourquoi le CERN a conçu le projet SHIFT. Ce projet est réalisé avec du matériel SILICON GRAPHICS. Ce système informatique a plusieurs processeurs de calcul et servira de serveur de traitement par lots. Il est sensé prendre la place qui sera laissée par le CRAY.

Ce que propose SHIFT (Scalable Heterogeneous Integrated FaciliTy) c'est :

- un système approprié à petite échelle (département de physique) ou à grande échelle (centre de calcul)
- les applications ne voient qu'un simple système
- tous les disques et les cassettes sont disponibles depuis n'importe quel endroit avec la même bonne performance
- son faible coût
- son haut débit
- son hétérogénéité

Un des avantages que présente ce projet, c'est son faible coût. En effet, l'économie qui sera réalisée par les expériences pour utiliser le matériel SILICON GRAPHICS sera très importante.

Maintenant ce projet doit convaincre par son efficacité et sa puissance.

b - Etude de faisabilité

Lorsque ce projet a été proposé, l'expérience ALEPH s'est posée la question de savoir si elle pouvait être intéressée par celui-ci, et si le travail qu'elle avait à fournir pour l'utiliser était important.

Il fallait tout d'abord regarder le problème des données et de leur accès, et la portabilité des logiciels du groupe OFFLINE.

Accès aux données :

Le projet SHIFT proposait d'allouer un certain espace disque pour l'expérience, et d'utiliser les services du robot de cassettes comme sur l'IBM. C'est à dire que quand un utilisateur a besoin d'une cassette, il suffit d'exécuter une commande pour que le robot charge cette cassette et la copie sur le disque dur.
Cette manipulation est connue et utilisée par ALEPH sur IBM. Cela implique donc peu de changement.

Portabilité des logiciels :

D'après les renseignements fournis par les responsables du projet SHIFT, les logiciels écrits en Fortran 77 peuvent être compilés avec le compilateur MIPS. Le format des mots sur SILICON GRAPHICS est le même que sur APOLLO. Sachant que les logiciels fonctionnent déjà sur des machines APOLLO, le travail était facilité.

C'est pourquoi ALEPH décida de tenter l'aventure et de se joindre au projet SHIFT. C'était la deuxième expérience après OPAL à utiliser les machines SILICON GRAPHICS.

c - Spécifications techniques

c-1 Connexion physique avec les machines du projet SHIFT

Le projet SHIFT présente l'avantage de pouvoir se connecter sur le réseau ETHERNET et FDDI.

ALEPH a réussi à se faire installer une fibre optique directement connectée aux machines SILICON GRAPHICS. Cette connexion directe a pour but d'augmenter le débit de transfert des données entre les deux systèmes sans être perturbée par le trafic externe.

Les disques qui sont connectés aux machines SILICON GRAPHICS peuvent être rendus visibles pour les DECstations grâce au protocole NFS. Ce qui permet d'accéder aux données qui sont stockées sur SILICON GRAPHICS à partir des DECstations.

c-2 Mise en place des logiciels

Sur les machines SILICON GRAPHICS, un répertoire *aleph* a été créé et l'architecture qui se trouve à l'intérieur de ce répertoire est la même que sur

DECstations (voir page 41). Ceci afin de faciliter la tâche de l'utilisateur qui se sert des deux systèmes.

Ensuite, il a fallu installer les logiciels du groupe et les tester.
Nous avons tout d'abord vérifier si l'installation des logiciels pouvait se faire sans avoir à changer des parties du code. Mais on s'est aperçu que certaines routines devaient être modifiées pour pouvoir fonctionner comme nous le souhaitions.
Enfin, j'ai fait fonctionner les programmes et tester la durée d'exécution de ceux-ci pour la comparer avec le temps d'exécution sur les autres systèmes.

Type de CPU	Temps d'exécution par événement
IBM	0.018
CRAY	0.034
SILICON GRAPHICS	0.026

(voir les résultats complet en annexe page 81)

On peut noter que les SILICON GRAPHICS sont plus rapides que le CRAY, mais l'IBM reste le plus performant.

d - Espérance et exigence pour le projet SHIFT

Les espérances d'ALEPH sont axées sur la fibre optique en espérant qu'elle tiendra toutes ses promesses, car malheureusement il y a eu des retards dans son installation et aucun test véritable n'a pas pu être effectué.

D'autre part, les responsables du projet SHIFT ont écrit un package pour la manipulation des fichiers (ouverture, écriture, lecture, ..) à travers le réseau. Nous avons pu faire quelques tests à travers le réseau ETHERNET, mais les tests les plus intéressants sont à travers le réseau FDDI. Ce package est intéressant car il n'y a plus besoin de monter les disques de données sur un autre système, on lit directement à travers le réseau.

Pour le moment, le projet SHIFT n'a pas tenu toutes ses promesses. L'instabilité de l'ensemble des machines SILICON GRAPHICS entraîne beaucoup de perturbations et d'erreurs dans les tests qui sont effectués par nos soins.

Quant à mon travail, il est inachevé suite aux problèmes décrits précédemment mais quand la connexion de la fibre optique sera réalisée, les tests sont prêts à être activés.

V Méthodes et relations humaines

L'exposé qui va suivre est le fruit d'une réflexion et d'une expérience vécue tout au long de mon travail effectué au sein du groupe OFFLINE. Certaines constatations ou remarques seront néanmoins d'ordre général, d'autres plus personnelles ou plus dépendantes de mon contexte de travail.

Pourquoi ai-je décidé de parler de méthodes et de relations humaines? Parce qu'ils constituent, à mon sens, deux outils principaux du métier de l'ingénieur.

1 - Les méthodes

a - Connaissance d'un produit

Face à de nouveaux environnements et de nouveaux produits, la méthode que j'ai été amené à utiliser repose sur l'analyse du produit, en partant de sa fonctionnalité première (réponse au quoi?) jusqu'à une étude des détails de son implémentation (réponse au comment?).

Notons que la durée de cette phase d'analyse dépend de la présence et de la qualité de la documentation technique disponible sur ce produit.

b - Conception d'un produit

A partir de l'expression précise et détaillée des besoins du client, on étudie la faisabilité du système: recherche d'une solution optimale à un besoin donné et validation de cette solution par rapport au contexte global.

Remarque: La réponse à un besoin s'appuie éventuellement sur l'existence d'outils à même de résoudre une partie des exigences.

Au sein de cette méthode de conception, se succèdent deux types de démarches:

- Une analyse descendante qui permet de valider la faisabilité du produit.
- Une analyse montante menant à la définition détaillée de ce produit.

c - Formation

Il existe, à mes yeux, deux méthodes essentielles pour acquérir de nouvelles connaissances:

- La première, passive, consiste à suivre des **cours** qui présentent l'avantage d'apporter une synthèse des connaissances nécessaires sur un sujet donné et le désavantage de ne pas tenir compte des disparités de niveaux.

- La deuxième, c'est l'**auto-apprentissage**. Elle suppose une bonne définition du type et du cadre d'action de l'information recherchée. Ces précisions vont ensuite servir à interroger des spécialistes de la question et/ou à sélectionner les documents se rapportant au sujet.

Relation avec le gain de temps:

Pour tous les problèmes de l'ordre du détail, cette deuxième méthode réalise un gain de temps notable par rapport à un apprentissage au sein d'un cours. En revanche, pour des problèmes de grande envergure et/ou de grande complexité, elle demande un investissement en temps beaucoup plus coûteux, à niveau final de connaissances égal.

Cette différence de temps s'amenuise avec le nombre d'auto-formations. Car l'on apprend à se documenter, à se former.

D'autre part, en prenant deux personnes, l'une issue d'une formation de type cours, l'autre d'un auto-apprentissage, à niveau équivalent, il est courant que cette dernière ait à plus long terme une meilleure appréhension globale du sujet. En effet, l'auto-formation passe par une suite de synthèses personnelles des notions étudiées.

Au CERN, ces deux méthodes sont appliquées. Il est très facile de suivre des cours dans des domaines aussi divers que l'informatique, l'électronique ou le

management. Mais on peut également apprendre soi-même un nouveau produit, de nouvelles méthodes directement en l'utilisant au cours de son travail.

d - Développement d'un produit

Cette partie s'emploie à exposer une méthodologie de développement qui permet, à mon sens, d'assurer la **qualité** du développement d'un logiciel.

d-1 Qualité du code produit

- *Indépendance par rapport au contexte*

Exemples:

- Dans les programmes sources, les relations aux fichiers "include" doivent être indépendantes du chemin. Leur dépendance doit être spécifiée dans le fichier makefile. (voir annexe page 87)
- Les définitions de structures et de types ainsi que les variables globales et les définitions de constantes (instructions "# DEFINE") doivent être exportées vers un fichier "include".

- *Clarté du code*

Exemples :

- **Indentation**: Toute instruction doit être correctement indentée pour une meilleure lisibilité du programme.
- **Taille maximale** du programme source: Elle ne devrait pas excéder mille lignes de code, chaque fonction ne devant, si possible, pas dépasser deux cents lignes.
- **Séparation claire** entre deux fonctions: Chaque définition de fonction devra normalement comprendre un entête résumant sa fonctionnalité.
- **Documentation** des déclarations: Chaque variable, ... déclarée devra être explicitée.
- Tous les identificateurs de fonctions et de variables devront être suffisamment **explicites**.

- Tous les identificateurs de variables devront débuter par un **préfixe significatif** de leurs types: par exemple, "p_" pour les variables de type pointeur.

Toutes ces règles conclues de la pratique permettront de minimiser les efforts de maintenance.

d-2 Portabilité du code

Compte tenu de l'évolution rapide des systèmes d'exploitation, la portabilité est une contrainte non négligeable du code écrit. Cela est d'autant plus nécessaire que les expériences utilisent plusieurs types de matériel fonctionnant avec différents dérivés d'UNIX, pas toujours compatibles.

Il est, par exemple, déconseillé de faire appel à des variables par leur adresse physique (i.e. en mémoire). A titre d'un autre exemple, la représentation interne des nombres n'étant pas standard (octet de poids fort avant octet de poids faible ou vice-versa), il est conseillé d'utiliser les primitives de conversions (htons(), ntohs(),...) avant de manipuler ces nombres.

d-3 Maintenance

La maintenance d'un produit logiciel est évaluée, durant la vie de ce produit, en moyenne à quatre fois son temps de développement (tests compris). C'est une composante essentielle du produit. Il est donc fortement conseillé de garder ce chiffre à l'esprit lorsque l'on développe et de tout faire, à l'avance, pour minimiser le temps de maintenance.

d-4 Documentation technique

Pour tout produit logiciel, il faudrait écrire un document technique expliquant dans les grandes lignes les fonctionnalités du produit. Cette tâche est trop souvent omise ou peu soignée, ce qui revient au même. Cette documentation permet à un nouvel arrivant de pouvoir maintenir un produit quand son réalisateur n'est plus dans l'entreprise.

Au CERN, ce temps n'est souvent pas pris en compte dans le développement d'un projet.

d-5 Documentation utilisateur

Le document réservé à l'utilisateur comporte les étapes d'installation du produit dans son environnement d'exécution, de démarrage du ou des programmes avec les mises en garde nécessaires. L'implémentation, la compilation, les dépendances... doivent être transparentes à l'utilisateur.

Il est souvent très apprécié de l'utilisateur de réaliser des "man" pages, c'est à dire des fichiers textes de documentation accessibles interactivement au clavier. (voir annexe page 94)

d-6 Les fichiers "Makefiles"

Par souci de clarté et d'efficacité, il est préférable de n'écrire qu'un unique fichier makefile par projet. Ce dernier doit alors contenir toutes les dépendances par rapport aux fichiers sources, objets, include ainsi qu'aux libraires (voir annexe page 87). Ce fichier permet de compiler, installer un logiciel avec un minimum de manipulation pour l'utilisateur.

On utilise également des primitives appelées "auto rules". Ce sont des règles de compilation dont certains paramètres sont réglables par l'utilisateur.

On ajoute généralement les deux primitives suivantes:

- install, qui gère l'installation des fichiers exécutables, des librairies et des fichiers include dans les répertoires spécifiés auparavant.

- clean, qui détruit tous les fichiers générés par la compilation.

e - Organisation de son travail

S'organiser est à mon sens la plus passionnante des responsabilités mais une des plus difficiles.

En effet, s'organiser c'est gérer autant d'impératifs que d'imprévus. C'est aussi savoir évaluer la durée d'une tache, quelle qu'elle soit et aussi courte qu'elle soit. C'est encore savoir s'adapter aux imprévus. Et enfin, s'être défini, en connaissance de cause, un planning et s'y tenir au mieux.

S'organiser devient encore plus complexe quand on y intègre le suivi d'un travail que l'on a délégué.

A chacun de définir la méthode qui lui convient. Personnellement, mes objectifs à long terme étant fixés, j'ai eu à m'organiser à moyen et à court terme.

Sur les conseils de mon parrain de stage, je me suis établi un planning selon la méthode d'un schéma PERT, pour répartir mes différentes tâches et fixer mes objectifs dans le temps. A la fin de mon travail, j'ai pu comparer mes prévisions avec le temps réellement passé pour chacune des tâches pré-définies.

Planning établi au mois de février

Activités Nb jours	jan. 20	fev. 20	mars 22	avril 20	mai 19	juin 22	juil. 23	août 22	sep 21	oct. 22	**total** 211
Réunions	1	1	1	1	1	1	1	1	1	1	**10**
Conférences	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	**5**
Séminaires			5			5			5		**15**
Apprentissage	16	5									**21**
Projet 1		6	7		2	1	1	1	1	5	**24**
Projet 2		6	7	13	12	8	10	4	2	5	**67**
Projet 3				3	1	5	10	4	3	5	**31**
Mémoire				1	1	1	1	7	3	5	**19**
Congés			1	1	1			5	5		**13**
Total par mois	17.5	18.5	21.5	19.5	18.5	21.5	23.5	22.5	21.5	21.5	**205**

Projet 1 : travail sur l'accès aux données
Projet 2 : mise en place des DECstations
Projet 3 : intégration d'ALEPH dans le projet SHIFT

Planning réel au début du mois d'octobre

Activités Nb jours	jan. 20	fev. 20	mars 22	avril 20	mai 19	juin 22	juil. 23	août 22	sep 21	oct. 22	**total** 211
Réunions	1	1	1	1	1	1	0.5			1	**7.5**
Conférences	0.5	0.5	0.5	0.5	0.5					0.5	**3**
Séminaires			5			5			10		**20**
Apprentissage	16	5									**21**
Projet 1		6	7		1	1	1			5	**21**
Projet 2		6	7	13	12	8	9	3	1	3	**62**
Projet 3				3	2	5	9	5	1	5	**30**
Mémoire				1	1	1	3	9	4	5	**24**
Congés			1	1	1			5	5	2	**15**
Total par mois	17.5	18.5	21.5	19.5	18.5	21	22.5	22	21	21.5	**203.5**

L'analyse comparative des deux tableaux fait ressortir que le temps passé pour les projets a été inférieur aux prévisions. Une explication peut être donnée par le retard de mise en place de la fibre optique.

Cette planification met en lumière le fait que j'ai passé plus d'un quart de mon temps à apprendre et écouter, et la moitié de mon temps a été consacrée à l'exécution de mes projets.

f - Rédaction d'un document technique

Pour rédiger un bon rapport technique, il faut en avoir lu des bons et des moins bons, et s'être rendu compte que les trois qualités de base d'un bon document technique sont:

- La **clarté**
- La **logique**
- le fait d'**être complet**

2 - Les relations humaines

A chaque fois que s'instaure un dialogue, par exemple, dans un rapport de client à fournisseur, à chaque fois que l'on aborde la notion de travail en équipe, il naît une relation humaine que l'on cherche, dans un contexte professionnel en tout cas, à rendre efficace et profitable pour les deux parties.

a - Où interviennent les relations humaines et sous quelles formes?

Le relationnel est indissociable des phases d'analyse des besoins et d'analyse du produit fini.

En effet, en phase d'analyse des besoins, le client soumet ses desiderata au fournisseur qui en prend bonne note et est éventuellement chargé d'en aider la définition. Il s'instaure donc un dialogue, une discussion argumentée qui doit être constructive et efficace. Le relationnel est ici d'autant plus important que cette phase établit souvent le premier contact client - fournisseur et définit la vie du produit.

En phase d'analyse du produit fini, un des tous derniers contacts client - fournisseur dans la mise en oeuvre d'un projet, les relations humaines prennent tout autant d'importance. Cet épisode est en effet le moment où le client porte un jugement sur le produit réalisé, accepte ou non le produit, éventuellement au prix de telle ou telle modification, et où l'on étudie sa conformité aux exigences. Ce contact prend alors la forme d'une discussion positive et non ambiguë ayant pour but, soit de mettre en évidence les derniers souhaits d'évolution du client pour son produit, soit de constater l'adéquation des fonctionnalités du produit avec les besoins du client.

En dehors de ces phases d'analyse, les relations humaines interviennent aussi au quotidien dans la recherche et la communication d'informations, c'est à dire à la fois:

- Dans toute demande d'informations ou d'explications auprès de personnes compétentes sur un sujet précis.
- Dans la disposition à offrir sa connaissance sur un sujet donné

Cet échange nécessite une certaine disponibilité d'esprit et une certaine souplesse, ainsi qu'une grande rigueur. Car **communiquer** un savoir, même quand les connaissances sont acquises et "digérées" **n'est pas trivial**. La clarté de l'expression et de la démonstration améliorent grandement la qualité d'un échange.

Enfin, les relations humaines sont très présentes au moment où le concepteur présente son produit à l'utilisateur. En effet, ce dernier doit alors faire preuve d'une grande clarté et d'une certaine adaptation à son interlocuteur qui n'a souvent pas le même niveau de préoccupation ni le même vocabulaire technique.

b - A qui et à quoi servent les relations humaines ?

La liste est longue. J'aborde donc ici les facettes qui me paraissent les plus importantes dans un contexte de travail, dans le sens où elles peuvent être le moteur de toute une équipe.

Toute relation humaine est le **véhicule d'informations et d'idées** et est à ce titre la matière première d'une équipe.
Puisque l'expérience, la personnalité et l'appréhension d'un problème varient d'un homme à l'autre, on mesure très vite l'importance d'un liant. Ce liant n'est autre que la **communication** dans une relation homme-homme.
Prenons l'exemple d'un débat de groupe: Comment s'assurer que les idées émises par chacun seront bien reçues et que la discussion avancera si personne ne tient compte de l'importance du support de communication ?
Ce support de communication est essentiellement un langage de communication (mots, transparents, documents écrits,...) qui est propre à chacun.

Une relation humaine prend également toute son importance dans la **confrontation ou le partage d'idées**. Or c'est la matière d'une réunion de travail. Et plus qu'un état d'esprit, c'est bien une méthode de travail.

Véhicule d'idées et aide à la confrontation de ces idées, les relations humaines servent aussi à favoriser l'**intégration** d'une personne au sein d'une équipe, à créer un réel et sain **climat** de travail et à développer et entretenir la **motivation** de tous.

Quelques bons témoins de la qualité de la communication dans une équipe sont :

- son activité,
- son efficacité,
- sa motivation,
- son comportement face à la difficulté.

Je dirai enfin que les relations humaines servent à chacun quotidiennement.

Une relation humaine bien vécue, c'est-à-dire vecteur de confiance, d'investissement et de communication est généralement source de progrès. Alors existe-t-il des règles de base qui permettent de la rendre efficace et profitable aux deux parties? Profitable est pris ici au sens d'un gain de temps et de clarté.

c - Que faire pour rendre profitable une relation humaine ?

Mon expérience à ce jour me permet de dégager trois **principes fondamentaux de communication**:

- Bien définir son message (question ou réponse)
- Savoir écouter
- Savoir s'exprimer

Il me semble effectivement primordial de prendre le temps de définir précisément (au préalable et pour soi) le message que l'on cherche à faire passer ou l'information que l'on souhaite obtenir. Ce principe se retrouve bien sûr dans toute préparation de réunion mais il est conseillé de l'appliquer plus généralement à tout échange d'informations (prises ou données) car il permet un gain de temps souvent considérable, en tous cas jamais superflu. Pour celui qui donne l'information comme pour celui qui la prend.
Ce principe est encore plus vrai quand l'échange d'informations se fait dans une langue étrangère, comme cela a été mon cas durant mon expérience au CERN.

Les deux autres règles sont beaucoup plus couramment citées comme piliers de la communication dans l'entreprise.

Je ne tirerai que deux leçons:

- Je pense que l'écoute fait partie intégrante de l'échange et qu'il est presque plus facile de s'exprimer que d'écouter.

- Je soutiens qu'une mauvaise expression (orale ou écrite) est souvent pire car source d'ambiguïté et de confusion donc de perte de temps que pas d'expression du tout.

Conclusion

Grâce aux connaissances théoriques et aux méthodes acquises pendant ma scolarité, j'ai pu aborder mon mémoire de fin d'études sur de bonnes bases.

Durant mes dix mois passés au CERN, j'ai pu mettre en application les méthodes apprises et approfondir les connaissances générales dans certains domaines.

En particulier, j'ai pu approfondir trois aspects de l'informatique qui sont le développement de logiciel, la communication entre ordinateurs à travers un réseau et enfin la gestion d'un système informatique complexe.

Mon expérience m'amène à la réflexion suivante : le métier d'ingénieur est composé pour moitié de technique et pour autre moitié de communication et relations humaines. Ce qui correspond bien à la formation que j'ai reçue.

Les projets sur lesquels j'ai travaillé pendant dix mois étant très intéressants, et l'un d'eux n'étant pas achevé, j'ai décidé avec l'accord de Monsieur Knobloch de rester quelques mois supplémentaires pour achever le travail que j'ai commencé.

Glossaire

ALEPH

A detector for LEP PHysics. Nom d'une expérience du LEP

ALPHA

Package standard de manipulation de données pour leur analyse dans ALEPH.

ALWS

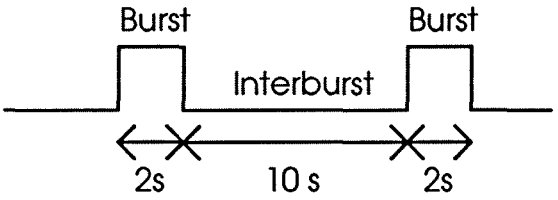
Ensemble de stations de travail VAX et de disques du groupe OFFLINE dans ALEPH

BOS

Package de gestion de mémoire dynamique utilisé pour les entrées/sorties et stockage interne des données d'ALEPH

Burst

Un faisceau organisé en bursts est un faisceau dans lequel les événements (issus des collisions entre particules) ne se produisent, dans un cycle donné, que pendant un certain intervalle de temps que l'on contrôle en fonction de l'expérience.



CERN

Centre Européen pour la Recherche Nucléaire. Acronyme : Laboratoire Européen pour la physique des particules.

CERNVM

Plate-forme IBM basée au centre de calcul du CERN

CPU

Central Process Unit. L'unité centrale est constituée d'un ou d'un ensemble de microprocesseurs.

CRAY

Super-ordinateur CRAY XMP utilisé comme serveur de traitement en lots au CERN

DECstation

Type de stations de travail fabriquées par la société DIGITAL.

DELPHI

Detector with Lepton, Photon and Hadron Identification.

digitization

Information brute créée par un élément d'un sous-détecteur (dans des données vraies ou simulées).

DST

Data Summary Tape. Sous-ensemble des données d'ALEPH avec des événements qui ne sont pas dû à la destruction des collisions e^+e^- et omission de quelques BOS banks de données brutes.

EDIR

Event DIRectory. Un index pour des événements autorisant un accès aléatoire plus rapide basé sur le numéro de l'événement et sa classe.

EPIO

Experimental Physics Input Output. Type de fichier pour le stockage des données indépendant de la machine.

ETHERNET

Système de câblage de haute performance (10 Mbit/s) qui lie les ordinateurs et des périphériques sur un réseau.

eV

Electronvolt. Unité égale à la variation d'énergie cinétique d'un électron qui subit une variation de potentiel de un volt.

Event builder

Système dont le but est de collecter des données associées à un événement depuis un ensemble de sources, de les ordonner (il reconstruit des événements à partir des données associées récoltées) et de les redistribuer vers un ensemble de destinations.

FALCON

Facility for ALeph COmputing and Network. Grappe destinée à la reconstruction des données pratiquement en ligne.

Farm

Système serveur CPU de traitement de différents processus à qui il envoie les données à traiter et les récupère. Il en est le maître et gère toutes ces entrées/sorties.

FDDI

Réseau utilisant comme support une fibre optique.

GALEPH

Programme Monte Carlo du détecteur ALEPH, qui produit des données simulées.

GEANT

Package Monte Carlo du CERN avec des routines standards pour décrire la géométrie et la modélisation des processus physiques. Utilisé par GALEPH.

GeV

Unité d'énergie correspondant à 10^9 électronvolt

Internet

Un réseau créé pour la liaison de deux ou plus petits réseaux de même ou de différents type avec un routeur ou une passerelle.

IP

Internet Protocol. Protocole responsable de la direction des paquets d'information depuis une machine vers une autre à travers Internet (méthode d'encapsulation).

JULIA

Job to Understand LEP Interactions in Aleph.

LEP

Large Electron Positron collider

LPI

Lep PreInjector. Canon à électrons permettant de générer des faisceaux d'électrons et de positons pour le LEP.

L3

LEP proposal 3

MeV

Unité d'énergie correspondant à 10^6 électronvolt

MINI-DST

Format abrégé pour les données d'ALEPH, utilisant un sous-ensemble des "banks" des DST échelonné et intégré pour économiser de l'espace de stockage.

NFS

Network File System. Offre un support pour partager des fichiers et des répertoires sur un réseau de machines hétérogènes.

OPAL

Omni Purpose Apparatus for LEP

Package

Ensemble de fonctions spécialisées écrites pour un produit donné.

Pipeline

Mémoire tampon servant d'unité de stockage en ligne des données associées aux événements expérimentaux.

Positon

Antiparticule de l'électron.

PS

Proton Synchrotron

RISC

Reduce Instruction Set arChitecture

SHIFT

Scalable Heterogenous Integrated Computing Facility. Projet CERN pour rassembler un large système de serveurs de traitement en lots fonctionnant sous le système d'exploitation UNIX.

SPS

Super Proton Synchrotron

TCP

Transmission Control Protocol. Pour les erreurs de transmissions et de réception de Datagrams à travers un réseau IP.

Trigger
Signal électrique de déclenchement.

ULTRIX
Version hybride du système d'exploitation UNIX fonctionnant sur la plate-forme DEC.

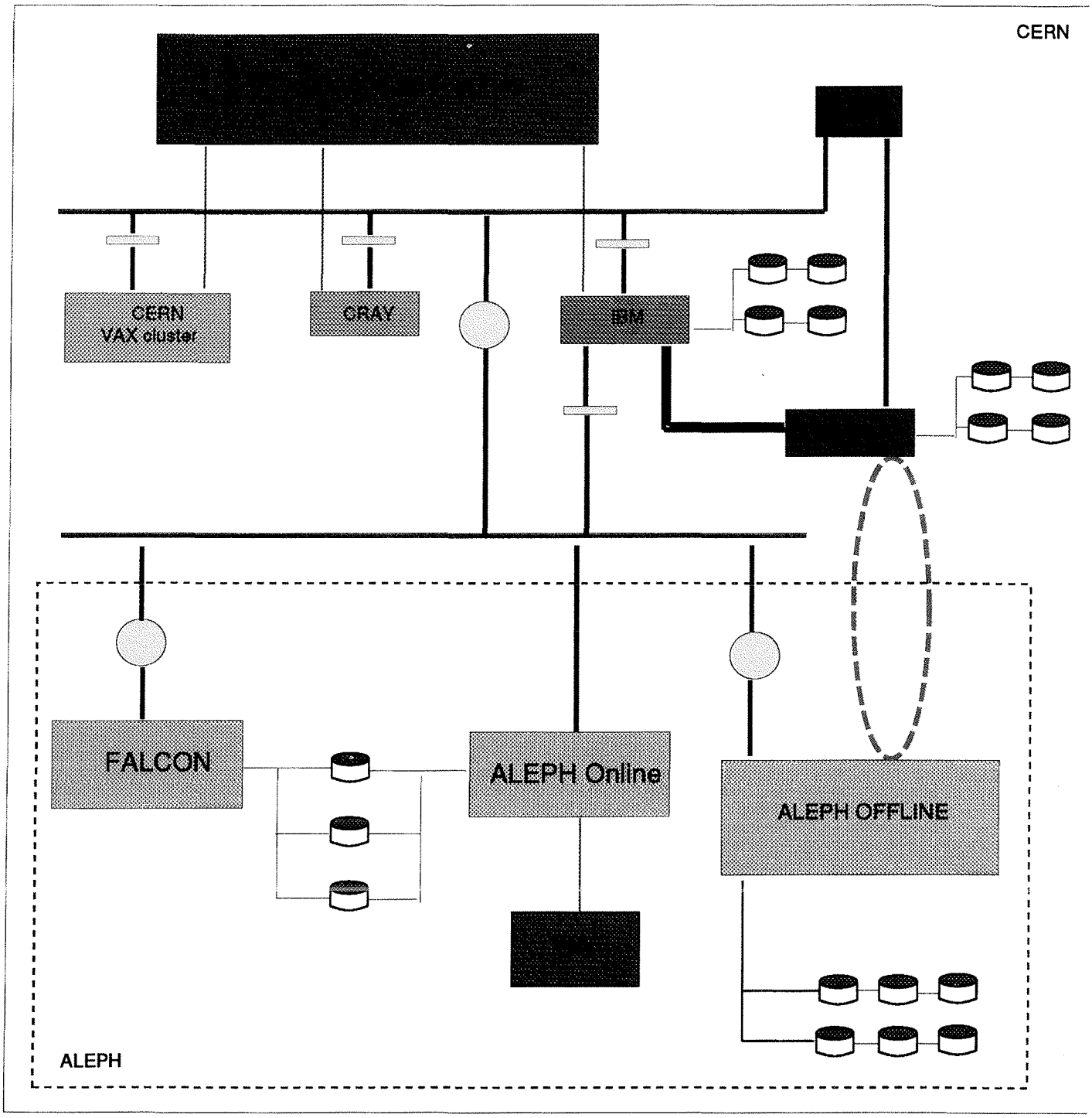
VAXstation
Type de stations de travail fabriquées par la société DIGITAL.

VM
Virtual Machine. Système d'exploitation de la plate-forme IBM.

VMS
Virtual Memory System. Système d'exploitation qui fonctionne sur les plates-formes VAX.

Annexes

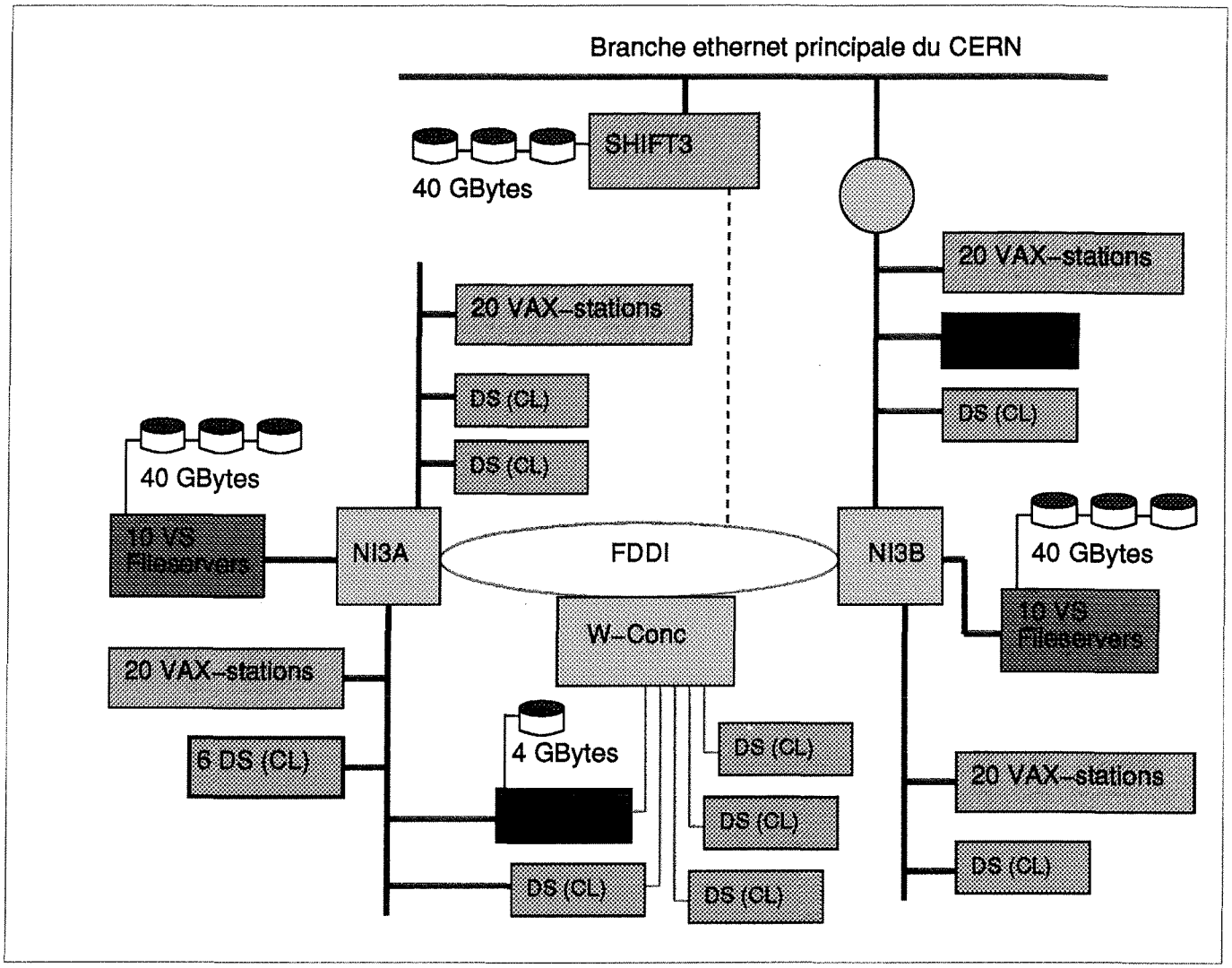
ALEPH au CERN



Legende :

- Disque
- Ethernet
- Ultranet
- FDDI
- Ordinateurs d'ALEPH
- Ordinateurs autre qu'ALEPH
- Concentrateur

ALEPH : grappe UNIX (DXAL) et grappe VAX (ALWS)



Legende :

DMS serveur

DS DMS client

MS – YP serveur maitre
SS – YP serveur esclave
CL – YP client
DS – DECstation

UNIX

Serveur

Client

VS – VAXstation

ALWS

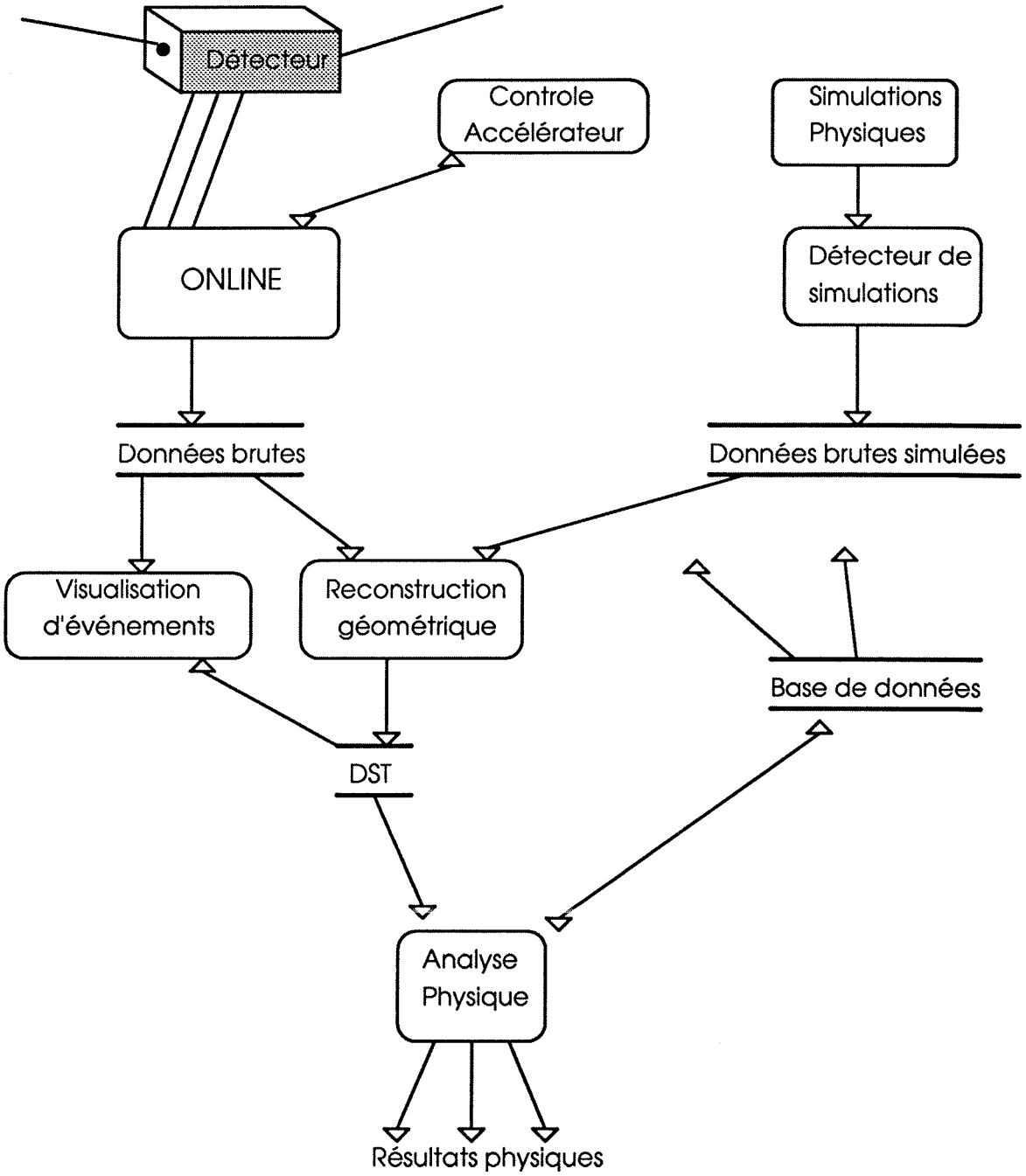
Concentrateur

Ethernet

Disque

FDDI

Organisation du traitement des données



Le fichier NFS.CONFIGURATION

```
!
!    MultiNet NFS configuration
!
!    Written by MultiNet NFS Configuration Utility 3.0(31)
!         by CLOSIER at Mon Jun 29 09:35:58 1992
!
!    Do *NOT* edit this file. Use the NFS Configuration utility
!    to change your configuration.
!

! Debugging
set nfsdebug              0
set rpcdebug              0
set filecache-debug       0

! Transports
set number-of-rpc-transports   10

! Cache parameters
set number-of-duplicate-requests-cached 250
set file-cache-timer-interval  30
set read-only-flush-age       180
set read-write-flush-age       60
set file-info-flush-age      1200
set directory-info-flush-age   300
set file-info-idle-flush-age   600
set directory-info-idle-flush-age     150
set maximum-cache-files       3000
set maximum-cache-buffers      500
set maximum-open-channels       50
set maximum-filesystem-files   3000
set maximum-filesystem-buffers 500
set maximum-filesystem-channels 50
set maximum-queued-removes     25
set seconds-before-writeback   0
set use-directory-blocking-asts 1
set use-file-blocking-asts     1
set maximum-dirty-buffers      0
set maximum-write-jobs         0


! Exported Filesystem: AL1$SCRATCH0:[ALEPH]
export AL1$SCRATCH0:[ALEPH]  dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11
dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2$SCRATCH0:[ALEPH]
export AL2$SCRATCH0:[ALEPH]  dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11
dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2$SCRATCH1:[ALEPH]
export AL2$SCRATCH1:[ALEPH]  dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11
dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1$USER0:[000000]
export AL1$USER0:[000000]  dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12
dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2$USER0:[000000]
export AL2$USER0:[000000]  dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12
dxal13 ptdec1 ptdec2 dxmp01 dxmp02
```

! Exported Filesystem: AL2$USER2:[000000]
export AL2$USER2:[000000] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1$USER3:[000000]
export AL1$USER3:[000000] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F01$DKA0:[ALEPHDATA]
export AL1F01$DKA0:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F01$DKA100:[ALEPHDATA]
export AL1F01$DKA100:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F02$DKB200:[ALEPHDATA]
export AL1F02$DKB200:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F02$DKB500:[ALEPHDATA]
export AL1F02$DKB500:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F02$DKB700:[ALEPHDATA]
export AL1F02$DKB700:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F03$DKB400:[ALEPHDATA]
export AL1F03$DKB400:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2F03$DKB100:[ALEPHDATA]
export AL2F03$DKB100:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2F03$DKB200:[ALEPHDATA]
export AL2F03$DKB200:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2F03$DKB300:[ALEPHDATA]
export AL2F03$DKB300:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL2F04$DKB0:[ALEPHDATA]
export AL2F04$DKB0:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F08$DKB200:[ALEPHDATA]
export AL1F08$DKB200:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F08$DKB300:[ALEPHDATA]
export AL1F08$DKB300:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

! Exported Filesystem: AL1F03$DKB200:[ALEPHDATA]
export AL1F03$DKB200:[ALEPHDATA] dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07 dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1 ptdec2 dxmp01 dxmp02

nfs-passwd-file 1dia2:[vms$common.multinet]ALOFFLINE.PASSWD

Le programme AUTOMOUNT.C

```
/*************************************************************/
/*                                                           */
/*     AUTOMOUNT.C written by Joel CLOSIER                   */
/*          april 1992                                       */
/*          Version : 1.0                                    */
/*                                                           */
/*     Automount.c reads the logicals of AL$DATA             */
/*These logicals are converted in a file called 'logical.txt' */
/*and they are compared to the old file 'logical.sav'. All the*/
/*changes are written in a file called 'update.txt'.         */
/*The file of the configuration for MULTINET is written and is*/
/*called 'nfs.txt'.                                          */
/*                                                           */
/*************************************************************/



#include <time.h>
#include <stdio.h>

FILE *fre,*fwr,*fw;
char bufl[80];
char name[80];


/*************************************************************/
/*     procedure to extract the name of a disk               */
/*                                                           */
/*  USE : line (line of the file which contains the value of */
/*      a LOGICAL)                                           */
/* Format of line is : "al$dat" = "al1f03$dkb200:[mcdata]"   */
/* Format of name is : al1f03$dkb200:[alephdata]             */
/*  RETURN : name (name of a disk on ALWS)                   */
/*************************************************************/
char *extract(char line[80])

{
int i=0,j=0,size;

 size = strlen(line);
 strcpy(name,"\n");
 while (i <= size)
  {
   j=0;
   while ((line[i] != "") && (i <= size))
    {
     i++;
    }  /*  end while not the first "  */
   i++;
   while ((line[i] != "") && (i <= size) && (line[i] != '.'))
    {
     name[j] = line[i];
     j++;
     i++;
    }  /*  end while not the second "  */
   if (line[i] == '.')
    {
     name[j] = ']';
     j++;
     i++;
    }
```

```
    name[j] = '\0';
    i++;
    if ((strcmp(name,"AL$DATA")!=0) &&
        (strcmp(name,"AL$DAT90")!=0) &&
         (strcmp(name,"AL$DAT91")!=0) &&
          (strcmp(name,"AL$DAT92")!=0) &&
           (strcmp(name,"AL$REPRO")!=0) &&
            (strcmp(name,"AL$MCDATA")!=0) &&
             (strcmp(name,"AL$RMINI")!=0) &&
              (strcmp(name,"NDATA")!=0))
     {
      return;
     }  /* end if real name of a disk  */
    else
     {
      name[0] = '\0';
     }
   }  /*  end while not end of line  */
  return;
}  /*  end procedure  extract */


/*************************************************************/
/*     procedure which compare two files                     */
/*                                                           */
/* USE : sav (file number 1), re (file number 2) and opt     */
/*      (add or delete)                                      */
/* This routine checks the file number 2 and looks if it     */
/* can find the same string in the file number 1. If the     */
/* string is not found, the line is written with the option  */
/* opt in the file fw (update.txt)                           */
/* RETURN : file update.txt with the right updating          */
/*************************************************************/
void comp(sav,re,opt)

char *sav,*re;
char opt;

{
FILE *fs,*fr,*jo;
char bufsav[80],bufsav1[80];
char ecrit[80];
int find;

 fs = fopen(sav,"r+");
 if (fs == NULL)
  {
   fs = fopen(sav,"w+");
   fclose(fs);
   fs = fopen(sav,"r");
  }
 while (!feof(fs))
  {
   fgets(bufsav,80,fs);
   find = 0;
   fr = fopen(re,"r+");
   while ((!feof(fr)) && (find != 1))
    {
     fgets(bufsav1,80,fr);
     if (strcmp(bufsav,bufsav1) == 0) find = 1;
    }  /*  end while not end of new file  */
   fclose(fr);
   if (find == 0)
    {
     ecrit[0] = opt;
     ecrit[1] = ' ';
```

```
    ecrit[2] = '\0';
    strcat(ecrit,bufsav);
    fwrite(ecrit,strlen(ecrit),1,fw);
   } /* end if bufsav not find */
 } /* end while not end of sav file */
 fclose(fs);
} /* end procedure compare */


/*************************************************************/
/*     procedure to create the file nfs.configuration     */
/*                                                           */
/* USE : files nfs.doc and logical.txt                */
/* This routine copies nfs.doc in nfs.txt and adds all the */
/* disks found in the file logical.txt                */
/* RETURN : file nfs.txt (nfs.configuration)          */
/*************************************************************/
void writenfs(void)

{

FILE *fin,*fout;
char strin[160],buf[80];
int index=0;
long now;

 now = time((long *) 0);
 fin = fopen("aleph$general:[tools]nfs.doc","r+");
 fout = fopen("aleph$general:[tools]nfs.txt","w+");
 if (fin != NULL)
  {
   while (!feof(fin))
    {
     fgets(buf,80,fin);
     if (index == 4)
      {
       buf[strlen(buf)-1] = '\0';
       strcat(buf,ctime(&now));
      }
     fwrite(buf,strlen(buf),1,fout);
     index++;
    } /* end while not end of file */
   fclose(fin);

   fin = fopen("aleph$general:[tools]logical.txt","r+");
   while (!feof(fin))
    {
     fgets(buf,80,fin);
     if (strcmp(buf,"") != 0)
      {
       strcpy(strin,"\n");
       fwrite(strin,strlen(strin),1,fout);
       buf[strlen(buf)-1] = ' ';
       strcpy(strin,"! Exported Filesystem: ");
       strcat(strin,buf);
       strcat(strin,"\n");
       fwrite(strin,strlen(strin),1,fout);
       strcpy(strin,"export ");
       strcat(strin,buf);
       strcat(strin," dxal00 dxal01 dxal02 dxal03 dxal04 dxal05 dxal06 dxal07")
;
       strcat(strin," dxal08 dxal09 dxal10 dxal11 dxal12 dxal13 ptdec1");
       strcat(strin," ptdec2 dxmp01 dxmp02 rsalmp01");
       strcat(strin,"\n");
       fwrite(strin,strlen(strin),1,fout);
      } /* end if string not empty */
```

```
   } /* end while read something */
  } /* end if file read is not empty */
  fclose(fin);
  strcpy(strin,"\n");
  fwrite(strin,strlen(strin),1,fout);
  strcpy(strin,"nfs-passwd-file $1$dia2:[vms$common.multinet]ALOFFLINE.PASSWD");
  fwrite(strin,strlen(strin),1,fout);
  fclose(fout);
} /* end procedure writenfs */


/*****************************************************************/
/*                main program                                   */
/*                                                               */
/* This routine creates the file logi.txt which contains         */
/* LOGICAL al$data, al$mcdata and ndata.                         */
/* It extracts the name of the disk from logi.txt with the       */
/* routine extract().                                            */
/* It compares the file logical.txt and the file logical.sav*/
/* to know if any change have appeared and writes these          */
/* changes in the file update.txt.                               */
/* It creates the file nfs.txt with writenfs().                  */
/*****************************************************************/
main(void)

{
 fre = fopen("aleph$general:[tools]logi.txt","r+");
 if (fre == NULL)
  {
   system("sh log al$data/output=aleph$general:[tools]logi.txt");
   fre = fopen("aleph$general:[tools]logi.txt","r+");
  } /* end if file does not exist */
 fwr = fopen("aleph$general:[tools]logical.txt","r");
 if (fwr != NULL)
  {
   system("copy aleph$general:[tools]logical.txt aleph$general:[tools]logical.sav");
  } /* end if file exists */
 fwr = fopen("aleph$general:[tools]logical.txt","w+");
 while (!feof(fre))
  {
   fgets(bufl,80,fre);
   extract(bufl);
   if (strcmp(name,"\0") != 0)
     {
      strcat(name,"\n");
      fwrite(name,strlen(name),1,fwr);
     } /* end if disk name */
  } /* end while fre not empty */
 fclose(fre);
 system("del aleph$general:[tools]logi.txt;*");
 fre = fopen("aleph$general:[tools]logi.txt","r+");
 if (fre == NULL)
  {
   system("sh log al$mcdata/output=aleph$general:[tools]logi.txt");
   fre = fopen("aleph$general:[tools]logi.txt","r+");
  } /* end if file does not exist */
 while (!feof(fre))
  {
   fgets(bufl,80,fre);
   extract(bufl);
   if (strcmp(name,"\0") != 0)
     {
      strcat(name,"\n");
      fwrite(name,strlen(name),1,fwr);
     } /* end if disk name */
  } /* end while fre not empty */
```

```
fclose(fre);
system("del aleph$general:[tools]logi.txt;*");
fre = fopen("aleph$general:[tools]logi.txt","r+");
if (fre == NULL)
 {
  system("sh log al$rmini/output=aleph$general:[tools]logi.txt");
  fre = fopen("aleph$general:[tools]logi.txt","r+");
 }  /*  end if file does not exist  */
while (!feof(fre))
 {
  fgets(bufl,80,fre);
  extract(bufl);
  if (strcmp(name,"\0") != 0)
    {
     strcat(name,"\n");
     fwrite(name,strlen(name),1,fwr);
    }  /*  end if disk name  */
 }  /*  end while fre not empty  */
fclose(fre);
system("del aleph$general:[tools]logi.txt;*");
fre = fopen("aleph$general:[tools]logi.txt","r+");
if (fre == NULL)
 {
  system("sh log ndata/output=aleph$general:[tools]logi.txt");
  fre = fopen("aleph$general:[tools]logi.txt","r+");
 }  /*  end if file does not exist  */
while (!feof(fre))
 {
  fgets(bufl,80,fre);
  extract(bufl);
  if (strcmp(name,"\0") != 0)
    {
     strcat(name,"\n");
     fwrite(name,strlen(name),1,fwr);
    }  /*  end if disk name  */
 }  /*  end while fre not empty  */
fclose(fre);
fclose(fwr);
fw = fopen("aleph$general:[tools]update.txt","w+");
comp("aleph$general:[tools]logical.sav","aleph$general:[tools]logical.txt",'d');
comp("aleph$general:[tools]logical.txt","aleph$general:[tools]logical.sav",'a');
fclose(fw);
system("del aleph$general:[tools]logi.txt;*");
system("del aleph$general:[tools]logical.sav;*");
system("purge aleph$general:[tools]*.*");
writenfs();
}  /*  end of main program  */
```

Le programme COPYNFS.COM

```
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$!       Routine to update the file nfs.configuration for the data
$!
$!  Written by Joel CLOSIER (June 1992)
$!
$!  This routine update the file nfs.configuration with AUTOMOUNT
$!  We update the password file (dec users)
$!  We copy the new file called nfs.txt on all the fileserver and we
$!  restart Multinet. (nfs.txt was created by automount)
$!
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$ set noon
$ set proc/priv=all
$! @al1f03$dkb200:[alephdata.nphy]sandydef.com
$ @al$sandy:sandydef.com
$ run al2$user2:[closier.test]automount
$ copy dxal10::"/etc/yp/src/passwd" $1$dia2:[vms$common.multinet]aloffline.passwd
$ pu $1$dia2:[vms$common.multinet]aloffline.passwd
$!
$!   copy nfs.configuration for all the fileserver
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f1.multinet]nfs.configuration
$ purge $1$dia2:[sys1f1.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f2.multinet]nfs.configuration
$ purge $1$dia2:[sys1f2.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f3.multinet]nfs.configuration
$ purge $1$dia2:[sys1f3.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f4.multinet]nfs.configuration
$ purge $1$dia2:[sys1f4.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f5.multinet]nfs.configuration
$ purge $1$dia2:[sys1f5.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f7.multinet]nfs.configuration
$ purge $1$dia2:[sys1f7.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia2:[sys1f8.multinet]nfs.configuration
$ purge $1$dia2:[sys1f8.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f1.multinet]nfs.configuration
$ purge $1$dia3:[sys2f1.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f2.multinet]nfs.configuration
$ purge $1$dia3:[sys2f2.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f3.multinet]nfs.configuration
$ purge $1$dia3:[sys2f3.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f4.multinet]nfs.configuration
$ purge $1$dia3:[sys2f4.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f5.multinet]nfs.configuration
$ purge $1$dia3:[sys2f5.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f6.multinet]nfs.configuration
$ purge $1$dia3:[sys2f6.multinet]
$ copy aleph$general:[tools]nfs.txt $1$dia3:[sys2f7.multinet]nfs.configuration
$ purge $1$dia3:[sys2f7.multinet]
$!
$!  restart multinet on all the fileserver
$ mcr sysman
set env/node=(al1f01,al1f02,al1f03,al1f04,al1f05,al1f07,al1f08)
do multinet netcontrol nfs restart
do multinet netcontrol rpcmount reload
set env/node=(al2f01,al2f02,al2f03,al2f04,al2f05,al2f06,al2f07)
do multinet netcontrol nfs restart
do multinet netcontrol rpcmount reload
exit
$!
$!  delete the file which contains the updating if it is empty
$ buf = f$file_attributes("aleph$general:[tools]update.txt","eof")
```

```
$ if (buf .eqs. 0) then del aleph$general:[tools]update.txt;*
$!
$!   Restart the job for tomorrow
$ submit /queue=sys$batch/after="tomorrow+4"/log_file=al2$user2:[closier.test]copynfs.log/restart
al2$user2:[closier.test]copynfs.com
$ set proc/priv=(noall,tmpmbx,netmbx)
```

```
#
# scratch areas
/alws/al1scratch0 -rw,hard,intr,timeo=300      al2f05:/al1$scratch0/aleph
/alws/al2scratch0 -rw,hard,intr,timeo=300      al1f04:/al2$scratch0/aleph
/alws/al2scratch1 -rw,hard,intr,timeo=300      al1f04:/al2$scratch1/aleph
#
# user disks
#/alws/al1user0 -rw,hard,intr,timeo=300              al2b02:/al1$user0/000000
/alws/al1user3 -rw,hard,intr,timeo=300    al2f03:/al1$user3/000000
#/alws/al2user0    -rw,hard,intr,timeo=300    al2b02:/al2$user0/000000
/alws/al2user2    -rw,hard,intr,timeo=300    al1f01:/al2$user2/000000
#
# data areas on ALWS
#
# aldata
#
# al1f02 via al2f02
/alws/al1f01/dka0/alephdata        al1f01:/al1f01$dka0/alephdata
/alws/al1f01/dka100/alephdata      al1f01:/al1f01$dka100/alephdata
/alws/al1f01/dka500/mcdata         al1f01:/al1f01$dka500/mcdata

# al1f02 via al2f02
/alws/al1f02/dkb0/alephdata        al1f02:/al1f02$dkb0/alephdata
/alws/al1f02/dkb100/alephdata      al1f02:/al1f02$dkb100/alephdata
/alws/al1f02/dkb200/alephdata      al1f02:/al1f02$dkb200/alephdata
/alws/al1f02/dkb400/alephdata      al1f02:/al1f02$dkb400/alephdata
/alws/al1f02/dkb500/alephdata      al1f02:/al1f02$dkb500/alephdata
/alws/al1f02/dkb700/alephdata      al1f02:/al1f02$dkb700/alephdata

#
# al1f03 via al2f02
/alws/al1f03/dkb100/alephdata      al1f03:/al1f03$dkb100/alephdata
/alws/al1f03/dkb400/alephdata      al1f03:/al1f03$dkb400/alephdata
/alws/al1f03/dkb700/alephdata      al1f03:/al1f03$dkb700/alephdata

# al1f04
/alws/al1f04/dkb100/alephdata      al1f04:/al1f04$dkb100/alephdata
/alws/al1f04/dkb200/alephdata      al1f04:/al1f04$dkb200/alephdata
/alws/al1f04/dkb300/alephdata      al1f04:/al1f04$dkb300/alephdata
/alws/al1f04/dkb400/alephdata      al1f04:/al1f04$dkb400/alephdata
/alws/al1f04/dkb500/alephdata      al1f04:/al1f04$dkb500/alephdata
/alws/al1f04/dkb700/alephdata      al1f04:/al1f04$dkb700/alephdata

#
# al1f05
/alws/al1f05/dka0/alephdata        al1f05:/al1f05$dka0/alephdata
/alws/al1f05/dka100/alephdata      al1f05:/al1f05$dka100/alephdata
/alws/al1f05/dka200/alephdata      al1f05:/al1f05$dka200/alephdata
/alws/al1f05/dka400/alephdata      al1f05:/al1f05$dka400/alephdata

# al1f07
/alws/al1f07/dkb0/alephdata        al1f07:/al1f07$dkb0/alephdata
/alws/al1f07/dkb100/alephdata      al1f07:/al1f07$dkb100/alephdata
/alws/al1f07/dkb200/alephdata      al1f07:/al1f07$dkb200/alephdata
/alws/al1f07/dkb300/alephdata      al1f07:/al1f07$dkb300/alephdata
/alws/al1f07/dkb500/alephdata      al1f07:/al1f07$dkb500/alephdata
/alws/al1f07/dkb700/alephdata      al1f07:/al1f07$dkb700/alephdata

#
# al1f08
/alws/al1f08/dkb0/alephdata        al1f08:/al1f08$dkb0/alephdata
```

```
/alws/al1f08/dkb100/alephdata        al1f08:/al1f08$dkb100/alephdata
/alws/al1f08/dkb200/alephdata        al1f08:/al1f08$dkb200/alephdata
/alws/al1f08/dkb300/alephdata        al1f08:/al1f08$dkb300/alephdata
/alws/al1f08/dkb400/alephdata        al1f08:/al1f08$dkb400/alephdata
/alws/al1f08/dkb500/alephdata        al1f08:/al1f08$dkb500/alephdata

#
# al2f01
/alws/al2f01/dka0/mcdata        al2f01:/al2f01$dka0/mcdata
/alws/al2f01/dka100/alephdata        al2f01:/al2f01$dka100/alephdata
/alws/al2f01/dka200/alephdata        al2f01:/al2f01$dka200/alephdata
/alws/al2f01/dka400/alephdata        al2f01:/al2f01$dka400/alephdata
#/alws/al2f01/dka500/alephdata        al2f01:/al2f01$dka500/alephdata
/alws/al2f01/dka500/mcdata        al2f01:/al2f01$dka500/mcdata

#
# al2f03 gets distibuted from al2f02
/alws/al2f03/dkb100/alephdata        al2f03:/al2f03$dkb100/alephdata
/alws/al2f03/dkb200/alephdata        al2f03:/al2f03$dkb200/alephdata
/alws/al2f03/dkb300/alephdata        al2f03:/al2f03$dkb300/alephdata
#
# al2f04
/alws/al2f04/dkb0/alephdata        al2f04:/al2f04$dkb0/alephdata
/alws/al2f04/dkb200/alephdata        al2f04:/al2f04$dkb200/alephdata
/alws/al2f04/dkb300/alephdata        al2f04:/al2f04$dkb300/alephdata
/alws/al2f04/dkb400/alephdata        al2f04:/al2f04$dkb400/alephdata
/alws/al2f04/dkb500/alephdata        al2f04:/al2f04$dkb500/alephdata
#
# al2f05 via al2f04
/alws/al2f05/dkb0/mcdata        al2f05:/al2f05$dkb0/mcdata
/alws/al2f05/dkb200/mcdata        al2f05:/al2f05$dkb200/mcdata
/alws/al2f05/dkb300/mcdata        al2f05:/al2f05$dkb300/mcdata
/alws/al2f05/dkb400/alephdata        al2f05:/al2f05$dkb400/alephdata
/alws/al2f05/dkb500/alephdata        al2f05:/al2f05$dkb500/alephdata
/alws/al2f05/dkb700/alephdata        al2f05:/al2f05$dkb700/alephdata
#
# al2f06 via al2f02
/alws/al2f06/dkb0/alephdata        al2f06:/al2f06$dkb0/alephdata
/alws/al2f06/dkb100/alephdata        al2f06:/al2f06$dkb100/alephdata
/alws/al2f06/dkb200/alephdata        al2f06:/al2f06$dkb200/alephdata
/alws/al2f06/dkb300/alephdata        al2f06:/al2f06$dkb300/alephdata
/alws/al2f06/dkb400/alephdata        al2f06:/al2f06$dkb400/alephdata
#
# aledir (already included in above list)
#
# /alws/al1f02/dkb100/edir  al1f02:/al1f02$dkb100/alephdata/edir
#
#
# mcdata
#
# mcedir (already included in above list)
#
/alws/al2f06/dkb300/alrepro        al2f06:/al2f06$dkb300/alrepro


/alws/al2f03/dkb0/alephdata        al2f03:/al2f03$dkb0/alephdata
/alws/al1f01/dka200/rmini        al1f01:/al1f01$dka200/rmini
/alws/al1f01/dka400/alephdata        al1f01:/al1f01$dka400/alephdata
/alws/al2f03/dkb500/alephdata        al2f03:/al2f03$dkb500/alephdata
/alws/al1f05/dka500/rmini        al1f05:/al1f05$dka500/rmini
/alws/al1f01/dka0/alephdata        al1f01:/al1f01$dka0/alephdata
```

Le programme ALWSUPDATE

```
#!/usr/bin/csh
#  Program to update the map for automount
#  and the link for the epio and edir files
#
#  Closier Joel June 1992
#
cd /etc/yp/src
#
# copy the file from ALWS which contains the information to update the map
dcp alws/aldata/coteroti::'aleph$general:[tools]update.txt' /etc/yp/src/update.txt
#
#  test if the file exists to do the changes
if (-f /etc/yp/src/update.txt) then
#
# update the file auto.alws_user with creating a file called new.txt
 update_auto
 if (-f /etc/yp/src/new.txt) then
   mv -f /etc/yp/src/auto.alws_user /etc/yp/src/auto.alws_user.sav
   mv -f /etc/yp/src/new.txt /etc/yp/src/auto.alws_user
#
# update the map for YP
  make -f /etc/yp/src/Makefile
#
# kill and restart the automount process
  /lsoft/local/bin/upautomount
  echo 'rsh dxal00 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal01 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal02 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal03 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal04 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal05 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal06 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal07 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal08 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal11 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal12 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  echo 'rsh dxal13 /usr/local/bin/psudo /lsoft/local/bin/upautomount &' >> /tmp/upau
  su -f manager < /tmp/upau
  rm -f /tmp/upau
 endif
 rm -f /etc/yp/src/update.txt
endif
#
# Creates logical links in the area /aleph/data
#  for every *.epio file on ALWS disks with names
#  of the form /alws/al*f*/d*
#
#echo 'Making logical links to the ALWS epio files'
echo 'rm -f /aleph/data/*' >> /tmp/makeli
echo 'set files = `ls' /alws/al*f*/d*/*data/*.epio 'ls' /alws/al*f*/d*/*data/*/*.epio` >> /tmp/makeli
echo 'foreach i (${files})' >> /tmp/makeli
echo '  ln -s -f $i /aleph/data/$i:t' >> /tmp/makeli
echo 'end ' >> /tmp/makeli
#
echo 'set files = `ls' /alws/al*f*/d*/*data/*/*/*.epio` >> /tmp/makeli
echo 'foreach i (${files})' >> /tmp/makeli
echo '  ln -s -f $i /aleph/data/$i:t' >> /tmp/makeli
echo 'end ' >> /tmp/makeli
#
#echo 'Making logical links to the ALWS edir files'
echo 'rm -f /aleph/edir/*' >> /tmp/makeli
```

```
echo 'set files = `ls` /alws/al*f*/d*/*data/edir/*.edir`' >> /tmp/makeli
echo 'foreach i (${files})' >> /tmp/makeli
echo '  ln -s -f $i /aleph/edir/$i:t' >> /tmp/makeli
echo 'end ' >> /tmp/makeli
echo 'set files = `ls` /alws/al*f*/d*/*data/medir/*.edir`' >> /tmp/makeli
echo 'foreach i (${files})' >> /tmp/makeli
echo '  ln -s -f $i /aleph/edir/$i:t' >> /tmp/makeli
echo 'end ' >> /tmp/makeli
echo 'exit (0)' >> /tmp/makeli
#
chmod +x /tmp/makeli
su -f aldata < /tmp/makeli
rm -f /tmp/makeli

exit
```

```
 ------------------------------------------------
          A L P H A  113.14  20.34.26 21/08/92

                running on CERNVM

               ALEPHLIB Version 13.90
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
0BREADC---- NAME.....NR +-------------------------------------------------------+
 --------- FDBA      0 FDBA 'ADBSCONS DAF *'
 --------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FILI      1 FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FILI      2 FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FILI      3 FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FILI      4 FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FILI      5 FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FILI      6 FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 --------- FFMT      0 FFMT 'BANKAL FMT * ! CARD'
 --------- HIST      0 HIST 'ALPT HIS A'
                * **NEVT 200
 --------- DEBU      0 DEBU 0
                + ENDQ
0_QWFILE_ Input / output files
 Selected events :    25097
 Current input file :   4097 events
           =  4099 run/event/other records
 No FILO card : No event output written
 Histogram file : ALPT HIS A
0_QWTIME_ : total   init  25097     1   term   TIME   time
       time   time  events  event   time   card   left
     8966.89   4.54 8961.18  0.357   1.18  15.00   0.57
 _QMTERM_ Stop.

########################################################################

 ------------------------------------------------
          A L P H A  113.14  11.39.43 20/08/92

                running on CERNVM

               ALEPHLIB Version 13.90
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
0BREADC---- NAME.....NR +-------------------------------------------------------+
 --------- FDBA      0 FDBA 'ADBSCONS DAF *'
 --------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
                * **FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
 --------- FFMT      0 FFMT 'BANKAL FMT * ! CARD'
 --------- HIST      0 HIST 'ALPT HIS A'
                * **NEVT 200
```

```
---------- DEBU       0 DEBU 0
                + ENDQ

0_QWFILE_ Input / output files
 Selected events :     2982
 Current input file :   2982 events
            =  2983 run/event/other records
 No FILO card : No event output written
 Histogram file : ALPT HIS A
0_QWTIME_ : total    init    2982      1    term    TIME    time
        time    time  events  event    time    card    left
      1155.75    4.53 1150.06   0.386    1.16   15.00    0.54
 _QMTERM_ Stop.

#########################################################################
 ------------------------------------------------
          A L P H A  113.14  09:55:16 24/08/92

               running on CRAY

              ALEPHLIB Version 13.90
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space

 BREADC---- NAME.....NR +-------------------------------------------------------+
              * **FDBA 'ADBSCONS.DAF'
 ---------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      1 FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      2 FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      3 FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      4 FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      5 FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      6 FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      7 FILI 'ALDATA  ! EPIO ! CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      8 FILI 'ALDATA  ! EPIO ! CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
 ---------- FILI      9 FILI 'ALDATA  ! EPIO ! CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
              * **FILI 'ALDATA  ! EPIO ! CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
              * **FILI 'ALDATA  ! EPIO ! CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
              * **FFMT 'BANKAL.FMT ! CARD'
 ---------- HIST      0 HIST 'CRAYAL.HIS'
              * **NEVT 200
 ---------- DEBU      0 DEBU 0
              + ENDQ

 _QWFILE_ Input / output files
 Selected events :    40083
 Current input file :  10084 events
          =  10088 run/event/other records
 No FILO card : No event output written
 Histogram file : CRAYAL.HIS

 _QWTIME_ : total    init   40083      1    term    TIME    time
        time    time  events  event    time    card    left
      1325.73    0.49 1325.17   0.033    0.07   15.00    1.17
 _QMTERM_ Stop.

#########################################################################

 ------------------------------------------------
          A L P H A  113.14  16:44:36 21/08/92

               running on CRAY

              ALEPHLIB Version 13.90
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
```

```
BREADC---- NAME.....NR +-------------------------------------------------------+
           * **FDBA 'ADBSCONS.DAF'
--------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
           * **FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0611.1.SL SIZE 200 W' ! 3000 QQB  PEAK RU
           * **FFMT 'BANKAL.FMT ! CARD'
           * **HIST 'CRAYALPT.HIS'
           * **NEVT 200
--------- DEBU      0 DEBU 0
           + ENDQ


_QWFILE_ Input / output files
Selected events :    4551
Current input file :  4551 events
          =  4553 run/event/other records
No FILO card : No event output written
No histogram output requested

_QWTIME_ : total   init   4551     1   term   TIME   time
      time   time  events  event   time   card   left
    155.19   0.48  154.65  0.034   0.06  15.00   1.29
_QMTERM_ Stop.

#########################################################################

-------------------------------------------------
         A L P H A  113.14  17:07:45 21/08/92

              running on CRAY

             ALEPHLIB Version 13.90
-------------------------------------------------
_QUIBOS_ Init BOS with 500,000 words working space

BREADC---- NAME.....NR +-------------------------------------------------------+
           * **FDBA 'ADBSCONS.DAF'
--------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      1 FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      2 FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      3 FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      4 FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      5 FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      6 FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
           * **FILI 'ALDATA  ! EPIO ! CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FILI 'ALDATA  ! EPIO ! CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
           * **FFMT 'BANKAL.FMT ! CARD'
--------- HIST      0 HIST 'CRAYAL.HIS'
           * **NEVT 200
--------- DEBU      0 DEBU 0
           + ENDQ
_QWFILE_ Input / output files
Selected events :    29287
```

```
 Current input file :   8287 events
            =  8290 run/event/other records
 No FILO card : No event output written
 Histogram file : CRAYAL.HIS

 _QWTIME_ : total   init  29287     1   term   TIME   time
       time   time  events  event   time   card   left
     981.92   0.49  981.36  0.034   0.07  15.00   1.08
 _QMTERM_ Stop.

#########################################################################

 ------------------------------------------------
          A L P H A  113.10  11.33.12 19/08/92

               running on SHIFT3

              ALEPHLIB Version 13.80
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
0BREADC---- NAME.....NR +-------------------------------------------------+
 --------- FDBA      0 FDBA '/aleph/dbase/adbs169.daf'
 --------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
               * **FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0608.1.SL SIZE 200 W' ! 3000 QQB  PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
               * **FILI 'ALDATA  ! EPIO ! CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
 --------- FFMT      0 FFMT '/aleph/doc/bankal.fmt ! CARD'
 --------- HIST      0 HIST 'ALPT.HIS'
 --------- NEVT      0 NEVT 200
 --------- DEBU      0 DEBU 0
              + ENDQ

0_QWFILE_ Input / output files
 Selected events :     200
 Current input file :   200 events
            =  201 run/event/other records
 No FILO card : No event output written
 Histogram file : ALPT.HIS
0_QWTIME_ : total   init   200     1   term   TIME   time
       time   time  events  event   time   card   left
      38.18   0.54   37.50  0.188   0.14  15.00 9960.81
 _QMTERM_ Stop.
logout
Job ended at Wed Aug 19 11:33:56 MST 1992

#########################################################################

 ------------------------------------------------
          A L P H A  113.10  11.40.39 19/08/92

               running on SHIFT3

              ALEPHLIB Version 13.80
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
0BREADC---- NAME.....NR +-------------------------------------------------+
 --------- FDBA      0 FDBA '/aleph/dbase/adbs169.daf'
 --------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
```

```
--------- FILI      1 FILI 'ALDATA  | EPIO | CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      2 FILI 'ALDATA  | EPIO | CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      3 FILI 'ALDATA  | EPIO | CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      4 FILI 'ALDATA  | EPIO | CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      5 FILI 'ALDATA  | EPIO | CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      6 FILI 'ALDATA  | EPIO | CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
                * **FILI 'ALDATA  | EPIO | CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  | EPIO | CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  | EPIO | CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  | EPIO | CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  | EPIO | CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
--------- FFMT      0 FFMT '/aleph/doc/bankal.fmt | CARD'
--------- HIST      0 HIST 'ALPT.HIS'
                * **NEVT 200
--------- DEBU      0 DEBU 0
                + ENDQ
0_QWFILE_ Input / output files
 Selected events :    59999
 Current input file :   9000 events
            =  9003 run/event/other records
 No FILO card : No event output written
 Histogram file : ALPT.HIS
0_QWTIME_ : total   init  59999     1   term   TIME   time
       time   time  events  event   time   card   left
     9852.45   0.47 9851.84  0.164   0.14  15.00  146.55
 _QMTERM_ Stop.
logout
Job ended at Wed Aug 19 16:54:52 MST 1992

##########################################################################

 ------------------------------------------------
          A L P H A  113.10  03.22.04 22/08/92

               running on SHIFT3

              ALEPHLIB Version 13.80
 ------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
0BREADC---- NAME.....NR +-------------------------------------------------------+
--------- FDBA      0 FDBA '/aleph/dbase/adbs169.daf'
--------- FILI      0 FILI 'ALDATA  | EPIO | CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      1 FILI 'ALDATA  | EPIO | CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
--------- FILI      2 FILI 'ALDATA  | EPIO | CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
                * **FILI 'ALDATA  | EPIO | CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  | EPIO | CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
--------- FFMT      0 FFMT '/aleph/doc/bankal.fmt | CARD'
--------- HIST      0 HIST 'ALPT.HIS'
                * **NEVT 200
--------- DEBU      0 DEBU 0
                + ENDQ

0_QWFILE_ Input / output files
 Selected events :    38998
 Current input file :   9000 events
            =  9003 run/event/other records
 No FILO card : No event output written
 Histogram file : ALPT.HIS
0_QWTIME_ : total   init  38998     1   term   TIME   time
       time   time  events  event   time   card   left
     6138.22   0.57 6137.51  0.157   0.14  15.00 3860.78
 _QMTERM_ Stop.
logout
Job ended at Sat Aug 22 05:58:22 MST 1992

##########################################################################
```

```
 ------------------------------------------------------------
          A L P H A  113.10  14.01.15 25/08/92

                running on SHIFT3

              ALEPHLIB Version 13.80
 ------------------------------------------------------------
 _QUIBOS_ Init BOS with 500,000 words working space
0BREADC--- NAME.....NR +-------------------------------------------------------------+
 --------- FDBA      0 FDBA '/aleph/dbase/adbs169.daf'
 --------- FILI      0 FILI 'ALDATA  ! EPIO ! CART AM0598.1.SL SIZE 200 W' ! 3000 QQB PEAK RUN
                * **FILI 'ALDATA  ! EPIO ! CART AM0599.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0600.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0601.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0602.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0603.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0604.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0605.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0608.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0609.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0610.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
                * **FILI 'ALDATA  ! EPIO ! CART AM0611.1.SL SIZE 200 W' ! 3000 QQB PEAK RU
 --------- FFMT      0 FFMT '/aleph/doc/bankal.fmt ! CARD'
 --------- HIST      0 HIST 'ALPT.HIS'
 --------- NEVT      0 NEVT 3000
 --------- DEBU      0 DEBU 0
                + ENDQ

0_QWFILE_ Input / output files
 Selected events :     3000
 Current input file :   3000 events
             =  3001 run/event/other records
 No FILO card : No event output written
 Histogram file : ALPT.HIS
0_QWTIME_ : total    init    3000      1    term    TIME    time
        time    time  events   event    time    card    left
      478.55    0.55  477.84   0.159    0.16   15.00 9520.44
 _QMTERM_ Stop.
logout
Job ended at Tue Aug 25 14:09:27 MST 1992
```

Le fichier MAKEFILE

```
#
# sudo makefile
#
CFLAGS=-O
DESTDIR=/usr/local/bin
ADMDIR=/usr/local/system
LOGDIR=/var/adm

all: psudo

psudo: psudo.o pclrusr.o
        cc $(CFLAGS) pclrusr.o psudo.o -o psudo

install: psudo
        install -m 4755 -o root psudo $(DESTDIR)

        if ![-d $(ADMDIR) ]; then \
        mkdir $(ADMDIR);\
        chmod go-rwx $(ADMDIR);\
        fi

        echo "You will have to edit the contents of $(ADMDIR)/sudoers."

        touch $(ADMDIR)/psudoers
        chmod go-rwx $(ADMDIR)/psudoers

        if ![ -d $(LOGDIR) ]; then \
        mkdir $LOGDIR);\
        chmod go-rwx $(LOGDIR);\
        fi

        touch $(LOGDIR)/psudo.log
        chmod go-rwx $(LOGDIR)/psudo.log


clean:
        rm -f pclrusr.o psudo.o make.out psudo



**********************************
/***   SUDO - run a command as root   ***/

#include <stdio.h>
#include <strings.h>
#include <ctype.h>
#include <sys/time.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/param.h>
#include <pwd.h>

#ifndef MAXHOSTNAMELEN
#define MAXHOSTNAMELEN 64
#endif MAXHOSTNAMELEN

#define ALERTMAIL "root"
#define LOGFILE "/var/adm/psudo.log"

extern char *ctime();
extern long time();
```

```
char *userfile = "/usr/local/system/psudoers";
char *progname;
long now;

void log(), errexit(), firsthostname();
int checkdoer();
char *isadoer();

main(argc, argv)

int argc;
char *argv[];

{
 char doerline[512];
 char cmd[512];
 char *dp;
 struct passwd *pw;
 int uid,pid;

 progname = argv[0];
 if (argc <2)
  {
   fprintf(stderr,"usage: %s cmd\n",progname);
   exit(1);
  }

 /*  remenber who this user really is  */

 uid = getuid();

 /*  set userid to be root and group id to be daemon  */

 if ((setuid(0)) < 0) eperror("setuid");

 if ((setgid(3)) < 0) eperror("setgid");

 dp = &doerline[0];
 pw = getpwuid(uid);
 now = time((long*) 0);

 if ((dp = isadoer(pw->pw_name,pw->pw_passwd)) == NULL)
  {
   log(pw->pw_name,"FAIL ",argc,argv);
   fprintf(stderr,"%s:I don't know you, and I'm telling!\n",*argv);
   if ((pid = fork()) == 0) mailmsg(pw->pw_name,argv,argc);
   if (pid == -1) eperror("fork");
   exit(1);
  }

 argv++, argc--;
 checkdoer(dp,*argv,cmd);

 if (strcmp(cmd,"all") == 0)
  {
   log(pw->pw_name,"",argc,argv);
   execvp(*argv,argv);
   eperror(*argv);
  }

 if (cmd[0] == '\0')
  {
   log(pw->pw_name,"FAIL ",argc,argv);
   fprintf(stderr,"%s: I know you, but I can't let you: %s\n",progname,*argv);
   exit(1);
```

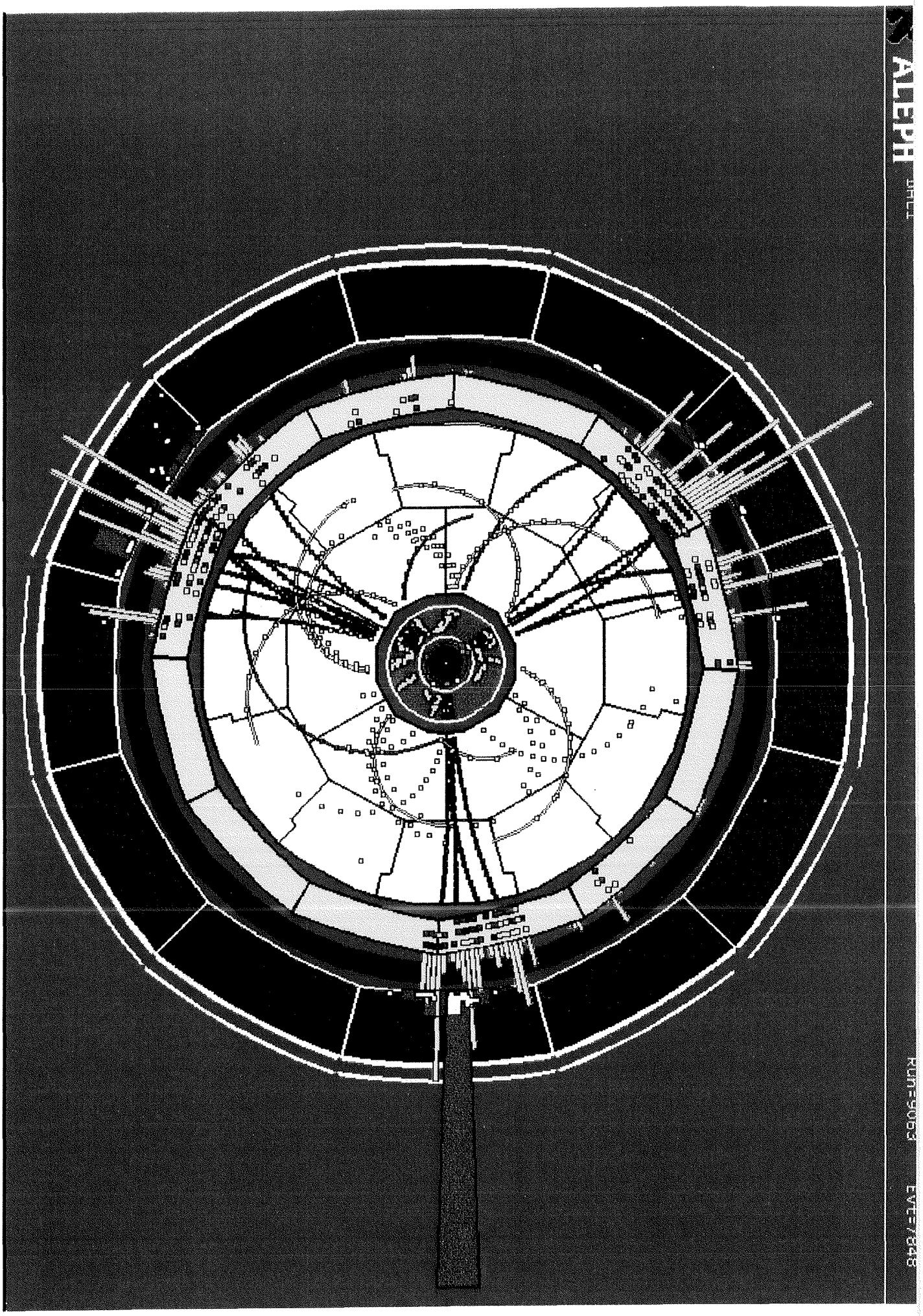
```
 }

 /* Do a specific command with hard-coded path from doer file */

 log(pw->pw_name,"",argc,argv);
 execv(cmd,argv);
 eperror(*argv);
}


/*****************************************************************/
/* isadoer look for a user in USERFILE                           */
/* return doer entry on success, else NULL                       */
/*****************************************************************/
char *isadoer(name,password)

char *name,*password;

{
 register FILE *fp;
 char buf[BUFSIZ];
 struct stat statb;

 if (stat(userfile,&statb)) eperror(userfile);
 if (statb.st_uid != 0) errexit("%s must be owned by root\n",userfile);
 if (statb.st_mode & 022) errexit("bad modes on %s\n",userfile);
 if ((fp = fopen(userfile,"r")) == 0) errexit("couldn't open %s\n",userfile);
 while ((fgets(buf,BUFSIZ,fp)) != NULL)
  {
   if (buf[0] == '#') continue;
   if ((strncmp(buf,name,strlen(name))) == 0)
    {
     if (not_timed_out(name,password)) return(buf);
     return(NULL);
    }
  }
 return(NULL);
}



/*****************************************************************/
/*   log this command in the log file                            */
/*****************************************************************/
void log(username,info,argc,argv)

char *username,*info, **argv;
int argc;

{
 register FILE *fp;
 fp = fopen(LOGFILE,"a");
 if (fp == NULL) errexit("can't open %s\n",LOGFILE);
 fprintf(fp,"%20.20s :",ctime(&now));
 fprintf(fp,"%s",info);
 fprintf(fp,"%7.7s :",username);
 while (argc--) fprintf(fp,"%s ",*argv++);
 fprintf(fp,"\n");
 (void) fclose(fp);
 return;
}
```

```
/*************************************************************/
/*  eperror - print system error message and exit          */
/*************************************************************/
eperror(s)

register char *s;

{
 fprintf(stderr,"%s :",progname);
 perror(s);
 exit(1);
}




/*************************************************************/
/*  errexit - print formatted error and exit also send mail  */
/*************************************************************/
void errexit(fmt,arg)

register char *fmt,*arg;

{
 FILE *popen(),*fd;
 char hostname[MAXHOSTNAMELEN],cmd[80];

 fprintf(stderr,"%s: ",progname);
 fprintf(stderr,fmt,arg);
 if ((fd = popen(cmd,"w")) == NULL) return;
 firsthostname(hostname,MAXHOSTNAMELEN);
 (void) sprintf(cmd,"/usr/ucb/mail -s \"HELP! %s@%s has problems.\" %s ",
   progname,hostname,ALERTMAIL);
 if ((fd = popen(cmd,"w")) == NULL) return;
 fprintf(fd,"%s: ",progname);
 fprintf(fd,fmt,arg);
 (void) pclose(fd);
 exit(1);
}




/*************************************************************/
/*  checkdoer - check to see if user is permitted to do      */
/*  command request.                                         */
/* dp points to a sudoer file entry of the form:             */
/* 'user ['all,','/dir/dir/cmdl,/dir/dir/cmdN'] ie)          */
/* coggs /bin/wall,/etc/vipassw./etc/adduser                 */
/* operator shutdown                                         */
/* If 'all' is found then string "all" is returned in res.   */
/* If command passed in ap is found, then that command, along */
/* with its full path are passed back in res. If nothing is  */
/* found, then NULL is returned in res                       */
/*************************************************************/
checkdoer(dp,ap,res)

char *dp,*ap,*res;

{
 char *cp0,*cp1,*cp2;

 cp0 = dp;
```

```
 while (isalnum(*cp0)) cp0++;    /* skip past user field */
 while (*cp0)            /* search until end of line */
  {
   while (isspace(*cp0)) cp0++;   /* skip to beg of cmd field */
   if (strncmp(cp0,"all",3) == 0)
    {
     (void) strcpy(res,"all");
     return;
    }
   cp1 = cp0;
  /* find end of this entry */
   while (*cp1 != '\n' && !isspace(*cp1)) cp1++;
   cp1--;  /* point to last caracter of this command */
   cp2 = cp1;
   while (*cp2 != '/') cp2--;   /* backup to head of command */
   cp2++;
   cp1++;
  /* if command issued is found then pass whole path back */
   if (strncmp(cp2,ap,strlen(ap)) == 0)
    {
     (void) strncpy(res,cp0,cp1-cp0);
     return;
    }
  /* if it got to here, then command not found yet move pointer
     past next comma and keep looking */
   while (!isspace(*cp0) && *cp0 != '\n') cp0++;
   if (*cp0 == '\n') break;
    else continue;
  }
 *res = NULL;
 return;
}
```

```
/*************************************************************/
/*       mailmsg - Snitch on person                          */
/*************************************************************/
mailmsg(user,argv,argc)

char *user,**argv;
int argc;

{
 FILE *popen(),*fd;
 char cmd[80],hostname[MAXHOSTNAMELEN];

 firsthostname(hostname,MAXHOSTNAMELEN);
 (void) sprintf(cmd,"/usr/ucb/mail -s \"*SECURITY* %s@%s tried to execute %s\" %s ",user,hostname,*argv,ALERTMAIL);
 if ((fd = popen(cmd,"w")) == NULL) return;
 fprintf(fd,"%s@%s tried to do a\n\n",user,hostname);
 while (argc--)
  {
   (void) fputs(*argv++,fd);
   (void) fputc(' ',fd);
  }
 (void) fputs("\n\nThought you might want to know.",fd);
 (void) pclose(fd);
}
```

```
/*************************************************************/
/* firsthostname - return hostname without domain stuff      */
/*************************************************************/
void firsthostname(n,l)

char *n;
int l;

{
 (void) gethostname(n,l);
 n[l-1] = 0;
 if (n=index(n,'.')) *n = 0;
}
```


ALEPH
RUN=9063
EVT=7848

```
.\" SCCSID: @(#)psudo.8  8.2    26/6/92
.TH PSUDO 8
.SH NAME
psudo - execute a command as root
.SH SYNOPSIS
.B psudo
command
.SH DESCRIPTION
.I Psudo
allows a permitted user to execute a command as root.
.I Psudo
determines who is an authorized user by consulting the file
.I /usr/local/system/psudoers.
If a match is found
.I command
is executed with root id.
Lines in
.I psudoers
beginning with a
.I '#'
or ' ' are considered comments and are ignored. The lines in the psudoers
must have the format :
.nf
.b

    # comment
    #
    user1  /path/path/cmd1 /path/path/cmd2
    user2  all

.r
.fi
In this example, Lines beginning with '#' are considered comments. User1
may invoke 'cmd1' or 'cmd2' only. User2 on the other hand is permitted
to execute any command.

.I Psudo
will send mail to root
if a user that does not appear in
.I psudoers
executes
.I psudo
, or if the file
.I psudoers
is not owned and exclusively writable by root.
All
.I psudo
invocations cause a log file
.I /var/adm/psudo.log
to be appended with date, time and command attempted.

.SH DIAGNOSTICS
.nf
psudo: I know you, but I can't let you: cmd
psudo: I don't know you and I'm telling !
psudo: bad modes on psudoers

.fi
.SH BUGS
Shell builtins such as
.T 'cd'
```

will fail.

It would be nice if entire groups could be enabled.
.SH FILES
.nf
/usr/local/bin/psudoers - list of authorized users
.br
/var/adm/psudo.log - record of all invocations of psudo
.fi
.SH SEE ALSO
su(1)

Bibliographie

[1] ALEPH : a detector for electron-positron annihilations at LEP. Nucl. Instr. and Meth. A294 (1990) 121-178;

[2] ALEPH contributions to the real-time 89 conference Williamsburg, Virginia (2 july 1989). ALEPH 89 - 116,DATACQ 89 - 19;

[3] Le CERN un exemple de coopération internationale;

[4] Cooperative Computing, L. Robertson, CERN school of computing , Sept 92

[5] Architecture of future data acquisition systems, L. Mapelli, CERN school of computing , Sept 92