ET

ABERKANE Magid, BIANCHERI Olivier, TONIN Pascal

Mai 2006

TABLE DES MATIÈRES

Introduction à Hurd# !

projet GNU

Le projet GNU fondé par Richard Stallman en 1983 a pour but d'écrire d'un système d'exploitation composé uniquement de logiciel libres.
C'est en 1986 que les discussions sur le futur noyau de système d'exploitation GNU s'intensifient. Le Noyau choisit au départ sort directement du MIT, mais il y a des nombreuses parties a réécrire et le choix se portera donc sur le microyau Mach, en 1988.

Avènement de LINUX

En 1992, alors que le projet GNU/Hurd fonctionnant sur GNU/Mach est encore en pleine phase de conception, Linus Torvalds met à disposition son noyau Linux qui fonctionne déjà et qui est entièrement sous licence GPL (Gnu Protected Licence). Deux discussion s'ouvrent alors :

L'adoption de Linux comme noyau officiel GNU Avec Linux, le projet GNU dispose enfin d'un noyau libre et pleinement opérationnel, cette polémique s'éteind rapidement après comparaison avec l'avancement de projet GNU/Hurd qui lui ne fonctionnera pas avant quelques années.

L'abandon du projet Hurd Linux gagne très vite une commauté impressionnante, il évolue rapidement et personne ne l'imagine disparâitre du jour au lendemain.
De nombreuses discussions visent à connaître le but visé par Richard Stallman en préservant le projet GNU/Hurd, qui est maintenant devenu inutile aux yeux de nombreuses personnes. Richard Stallman explique que avec GNU/Hurd il veut créer une nouvelle architecture de système, et qu'il veut prouver que la vision classique d'un système d'exploitation tel que Linux n'est pas la seule valable.
Le second argument est plus idéologique, en effet le but de l'O.S. GNU est de faire naître un système totalement libre, ce que Linux n'est pas à cause de la façon dont il est implémenté. GNU/Hurd et son implémentation permettront selon Stallman de permettre a une plus grande partie de développeurs de participer activement au projet GNU.
Sur son appui, l'équipe qui travaille sur GNU/Hurd continue son travail.

L'évolution de Linux et de Hurd

Linux a évolué et gagné aujourd'hui sa place phare au centre du projet GNU, de nombreux développeurs ont profité de son expansion pour apporter leur aide au projet GNU.
Cependant, la communauté qui implément aujourd'hui le noyau linux se fait de plus en plus rare, étant donné la complexité de sa programmation, argument qui motive toujours autant Richard Stallman pour pousser son projet tant attendu.
Hurd lui a très peu évolué car la communauté entièrement dédiée à linux lui a enlevé un grand nombre de testeur et programmeurs futurs, ce qui a ralenti son implémentation et sa mise a disposition.
Le noyau Mach ne tient pas la route niveau performances face au noyau Linux, entrainant le pessisme de nombreux utilisateurs linux qui se sont essayé à hurd.
2002 a été une année phare pour hurd avec la création du système Hurd debian, qui a aujourd'hui beaucoup progressé, mais qui attends énormément maintenant des futures évolutions du Hurd.
En effet, Hurd souffre de la dure comparaison avec linux, et même s'il lui est supérieur en de nombreux points, il n'est toujours pas prêt pour être adopté par le public linuxien.
Richard Stallman l'a avoué, linux est le projet phare de projet GNU pour encore pas mal d'années.

CHAPITRE 1

Le Noyau

Le point essentiel qui réside entre GNU/Hurd et Linux se situe au niveau de l'architecture, de la communication du noyau avec les services, les applications et le matériel.
Pour différencier les architectures de Hurd et de Linux nous allons tout d'abord nous intéresser à l'architecture monolitique avec son noyau. Nous expliquerons ensuite le fonctionnement de l'architecture à micro-noyau et nous finirons par la présentation de deux micro-noyaux qu'utilise le projet GNU/Hurd.

1.1 Architecture monolithique

1.1.1 Noyau monolithique non modulaire

Définition

Dans une architecture monolithique, le noyau et toutes les fonctionnalités (processus, IPC, gestion de mémoire, systèmes de fichiers, accès aux périphériques...) sont regroupés dans un même et seul bloc. Le noyau est exécuté par un seul fichier.

Le noyau monolitique se charge de :
- l'ordonnancement
- la gestion des processus
- entrées-sorties des périphériques
- la mémoire
- la pagination
- le système de fichiers
- couche réseau

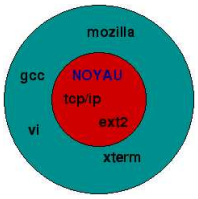


FIG. 1.1 – Schéma d'une architecture avec un noyau monolithique non modulaire

Avantage

- Vitesse d'exécution :
  Si le noyau est bien implémenté et fiable, le code étant dans un seul fichier, les capacités du matériel sont améliorées.

Inconvénients

- Maintenance :
  De part la grande quantité de code écrit pour le noyau monolitique, la maintenance du code est très difficile. il est très dur de lire et de comprendre le code à cause de nombreuses dépendances entre les fonctions du noyau. Peu de personne au monde est capable de comprendre et de modifier le noyau.

- Portabilité :
  Même si l'on peut utiliser des directives pre-processeur pour préciser ce qu'il faut compiler selon le matériel, la majorité des noyaux de ne sont pas portables à cause de l'évolution du code faite en parallèle à l'évolution du matériel.

- Sécurité :
  Comme les fonctionnalités sont dans le même espace que celui du noyau, le bogue de l'un d'entre eux peut provoquer le crash du système entier.

1.1.2 Noyau monolithique modulaire

Pour résoudre les problèmes des noyaux monolithiques, ces noyaux ont subis une évolution et sont devenus des noyaux monolithiques modulaires.
Linux utilise ce type de noyau.

Définition

Seules les fonctionnalités principales au sytème sont encore inclues avec le noyau.
Les autres fonctions sont déployées sous forme de modules dont le code ne fait plus partie de celui du noyau. Ainsi ces modules peuvent être chargés et déchargés indépendamment dans l'espace utilisateur ou celui du noyau.
Le noyau peut donc être constitué du strict nécessaire pour le démarrage et son initialisation.

Avantages

- Vitesse d'exécution :
  La vitesse reste la même que celle d'un noyau monolithique.

- Simplicité de conception et développement

- Maintenance :
  Le code est plus clair et mieux organisé car le code source du noyau est découpé en plusieurs blocs indépendants.

- Chargement des fonctionnalités en mémoire peut être fait à la demande.

Inconvénient

- Portabilité :
  Dans la partie théorique, la portabilité est difficile à cause de la grande quantité de code mais dans la partie pratique les noyaux modulaires sont les plus portés.

1.2 Architecture micro-noyau

1.2.1 Définition

Le micro-noyau est le noyau nouvelle génération, c'est un noyau modularisé qui à accès juste au matériel, à la mémoire, à la gestion des processus et de la communication entre eux.
Il a été crée pour minimiser les fonctionnalités : il ne reste plus qu'un nombre réduit de fonctions fondamentales dans le noyau. Les autres fonctionnalités sont représentées par des "services" qui ne s'exécutent pas dans l'espace du noyau mais dans celui de l'utilisateur.
Cependant le micro-noyau peut attribuer des droits aux composants de l'espace utilisateur pour élever leur niveau de privilège.

Les parties restantes en espace noyau sont :
- IPC
- Ordonnanceur basique
- Gestion de la mémoire basique

Les parties essentielles déplacées dans l'espace utilisateur sont :
- l'ordonnanceur
- la gestion de la mémoire
- le système de fichier
- la couche réseau

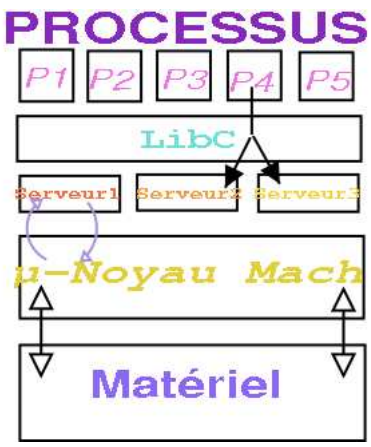


FIG. 1.2 – Schéma d'une architecture à micro-noyau

1.2.2 Avantages

- Conception et Evolution :
  Grace à la modularité, il est beaucoup plus facile de faire évoluer dynamiquement le micro-noyau. Si l'on veut rajouter par exemple un driver, il suffit juste de rajouter un serveur, le micro-noyau ne subissant aucun changement et ceci se faisant sans redémarrage sans pertubations des autres sessions des utilisateurs.

- Maintenance :
  Les services étant indépendants et le code du micro-noyau étant réduit assez fortement, il est plus facile de maintenir et de faire évoluer le micro-noyau. On peut modifier ou échanger les composantes du système assez facilement grace à la modularité.

- Robustesse :
  Comme les services dits "à risque" sont beaucoup moins nombreux dans le noyau, le système est plus fiable.

- Sécurité :
  Comme les exécutions des éléments dans l'espace du noyau sont très limitées, les risques d'erreurs dans le noyau est très nettement diminué. Si un des services a un problème de fonctionnement, comme il se trouve en espace utilisateur, il ne peut crasher le système.
  De plus l'utilisateur peut lancer un sous-système Hurd sous sa session, ce sous-système se nommmant sub-Hurd. S'il advient un problème sous ce sub-Hurd cela n'affectera au pire que le sous-sytème, les autres utilisateurs n'étant pas affectés.

- Portabilité :
  De part sa petite taille et sa limitation de fonctions, le micro-noyau peut être porté facilement sur plusieurs systèmes.

1.2.3 Inconvénients

- Performance :
  Les services maintenant utilisés en espace utilisateur sont indépendants et communiquent souvent entre-eux et avec le micro-noyau. Le micro-noyau utilise beaucoup d'IPC pour ces communications ce qui entraine une certaine lourdeur et une limitation des performances.

- Complexité :
  L'architecture à micro-noyau est plus complexe, plus récente, la progression est donc globalement lente.

1.3 Micro-noyau associé à Hurd

1.3.1 GNU Mach

Historique

Micro-noyau libre écrit en C orienté objet et considéré obsolète par le projet GNU.
Il a un niveau de sécurité avancé (sécurité de niveau B3).
Il a été développé à l'université de Carnegie-Mellon (1985-1994) à Pittsburg et se base sur le premier micro-noyau Accent non compatible avec UNIX. Mach reprend les fonctionnalités de Accent excepté qu'il est compatible avec UNIX.
En 1988, Mach 3 est choisi comme micro-noyau pour le projet GNU.
En 1991, Mach 3 est diffusé sous licence compatible.
Mach étant portable sur plusieurs plateformes, et pouvant être implanté sur à peu près toute les machines existantes, Hurd l'a choisi comme base.
Il est l'une des premières implémentations réussies des micro-noyaux.
C'est le premier système à définir clairement les notions de tâches.

Interfaces

Mach s'organise en un ensemble d'interfaces par lesquelles on accède aux ports, messages, tâches, threads et mémoire virtuelle.

Une tâche Mach (processus Unix) correspond à un environnement d'exécution permettant d'accéder aux ressources systèmes par des ports et pouvant contenir un ou plusieurs threads.
Une tâche Mach ne contient pas implicitement un thread comme dans UNIX.

Un thread est une unité d'exécution de base et s'exécute dans le contexte d'une tâche.
Les threads Mach correspondent aux threads des autres systèmes d'exploitation modernes, ce sont un flot d'exécutions partageant des ressources avec les autres threads appartenant à la même tâche.

Le mécanisme de port que fournit Mach sert à la communication des différentes tâches. Un port Mach est semblable à celui utilisé dans la programmation socket, c'est à dire qu'il fournit un canal de communication entre deux tâches, même si celles-ci ne sont pas sur le même processeur.

Un message est une communication entre thread.

Gestion des processus et communication

Mach se charge d'un ordonnancement simple car seuls les threads sont ordonnancés et rentrent en compétition pour obtenir des ressources. Les tâches n'entrent donc pas en jeu.
Les threads sont placés dans une file d'exécution selon leur priorité (numéro entre 0 et 127, plus un thread utilise le CPU plus sa priorité est faible), file d'exécution globale (visible par tous les processeurs) ou locale (visible par un seul processeur).
Mach utilise la notion de quantum à temps variable. Commme l'on se trouve dans un système multiprocesseur, on peut avoir à un moment donné moins de threads en cours d'exécution qu'il n'y ait de processeurs disponibles. Un thread qui a épuisé son quantum de temps fixe est interrompu puis immédiatement réexécuté. Le changement de contexte étant couteux, cela dimininue les performances.

La communication sous Mach est basée sur les ports et messages, ce qui permet d'assurer une transparence de la localisation des objets et une sécurité de communication. L'échange de messages qui constitue la communication sous Mach se fait par "Copy on write", technique très rapide qui conserve l'original des modifications.

Gestion mémoire

Mach ne redéplace pas les blocs mémoire pour avoir la plus grande zone possible, c'est un thread interne au noyau qui gère l'allocation et la récupération mémoire.
Mach utilise l'algorithme basé sur le FIFO de la seconde chance pour sélectionner les pages à remplacer. Les pages sélectionnées sont envoyées au gestionnaire approprié qui peut se situer au niveau utilisateur. Cela permettant une meilleure performance qu'avec le gestionnaire par défaut. Cependant si le gestionnaire au niveau utilisateur n'arrive pas à diminuer le nombre de pages, le gestionnaire par défaut sera utilisé. Le rendement sera donc le meilleur possible.
Un thread qui demande une ressource sur le disque utilise un appel système contenant le port identifiant la ressource et le gestionnaire mémoire utilisé. L'objet est alors prêt à recevoir des requêtes. Le thread est placé dans un état d'attente si un défaut de page survient pendant l'accès à cet objet. Lorsque toutes les tâches ne veulent plus envoyer de données vers cet objet, le noyau libère le port associé à l'objet et donc la mémoire utilisée par l'objet.

Mach utilise des gestionnaires de mémoires externes. Ainsi les tâches voulant partager la même section de mémoire devront utiliser le même gestionnaire.

Hurd utilise actuellement le micro-noyau Mach.

1.3.2 L4Ka : :Pistachio

Il est le renouveau du micro-noyau de Hurd mais il n'en est qu'au stade expérimental(cependant diverses versions sont disponibles gratuitement).
Il est écrit en C++ et est en cours de développement à l'université de Karlsruhe en Allemagne qui collabore avec l'université de New South Wales en Australie.
Avec le micro-noyau L4, encore plus de services sont déplacés dans l'espace utilisateur.
Les communications ont été simplifié au maximum pour un meilleur traitement des IPC, les vérifications des permissions sont déléguées aux serveurs externes, la mémoire virtuelle et les pilotes des périphériques sont relégués à l'espace utilisateur.
Il est donc beaucoup plus petit et plus rapide que le micro-noyau Mach.

CHAPITRE 2

Les serveurs

2.1 Architecture multi-serveur

2.1.1 Définition

Les systèmes multi-serveurs sont constitués du micro-noyau et de multiples serveurs tournant en mode utilisateur.

Le Hurd est un ensemble de serveurs qui s'exécutent dans l'espace utilisateur au-dessus du micro noyau, pour fournir les services aux applications.

Chaque serveur fournit un certain nombre de services, sous formes d'appels RPCs (Remote Procedure Call).
Les serveurs fonctionnent comme une tâche Mach, et communiquent via les ports de Mach.

La communication entre les applications et les serveurs se fait alors par un mécanisme de communication inter-processus ou IPC.

Le service d'IPC est fourni par le noyau, via un appel système, qui se charge de transmettre l'information à la tâche destinataire.

Au-dessus du mécanisme d'IPC est généralement construit un système d'appel de procédures distantes (RPC) dont l'objectif est de masquer au programmeur le fait que la fonction appelée par le programme est exécutée par une autre processus.

2.1.2 Avantages

Modularité

Le système est conçu de manière très modulaire de façon à ce que chaque composant puisse être remplacé ou modifié.
L'utilisateur a ainsi la possibilité de changer de serveur si des besoins en ce sens se font sentir et ce, sans mettre en péril la sécurité ou la liberté des autres utilisateurs.

Si un serveur fait preuve d'instabilité, il n'entraîne pas l'ensemble des serveurs et le micronoyau dans sa « chute ».

De même, si l'on estime qu'un serveur ne remplit pas correctement sa tâche, celui-ci peut être remplacé dynamiquement par un autre serveur semblable, sans avoir à relancer le noyau (ie. la machine).

Avec les caractéristiques de transparence réseau, les serveurs peuvent être répartis sur plusieurs machines.
Lorsqu'une nouvelle tâche est créée, le système lui alloue des ressources sur la même machine ou sur une machine distante.

Code réentrant

Le code du système est réentrant. Le système peut donc servir plusieurs requêtes simultanément et de manière asynchrone, contrairement l'Unix classique où le noyau ne peut fournir qu'un seul service à la fois en suspendant les autres processus accédant au noyau. Mais les besoins en ressources d'un tel système sont bien plus grands en comparaison à Linux ou à FreeBSD.

2.1.3 Inconvénients

Néanmoins ce type d'architecture possède quelques inconvénients : le premier est que chaque serveur développé nécessite la définition d'une instance pour l'utiliser (pour savoir quel est le serveur recherché et comment le joindre) ; le second est le coût très élevé des IPC (Inter Process Communication).
En effet, les systèmes multiserveurs réclament moins dappels directs au noyau mais plus de communication entre les différents serveurs.

2.1.4 Apports de Hurd

Cependant le Hurd apporte encore plus que les systèmes multiserveurs.

En effet, les utilisateurs sont autorisés et même à développer leurs propres serveurs.
Les applications communiquent uniquement avec des parties restreintes du noyau tandis que le reste est entièrement remplaçable de façon dynamique.

Ainsi les utilisateurs peuvent conserver la partie du système qu'il souhaite et ajouter leurs propres composants sans affecter lenvironnement des autres utilisateurs.

Le Hurd permet à l'utilisateur de lancer ses propres services systèmes, donc ses propres serveurs.
Il est alors possible de tester de nouvelles fonctionnalités sans perturber les autres utilisateurs et de changer dynamiquement un serveur.

2.2 Les principaux serveurs

2.2.1 Serveurs de base

Serveur exec

Le serveur exec crée la tâche (espace d'adressage) Mach, qui crée les ports correspondants (ports sur proc, auth, port de bootstrap), gère l'héritage de privilèges, l'enregistre auprès de proc et analyse le fichier pour lancer l'interpréteur au besoin (présence d'un hashbang ou non), et réalise le chargement de l'exécutable ELF dans l'espace d'adressage pour lui passer la main ensuite l'implémentation de l'appel système "execve" est répartie entre trois programmes.

La bibliothèque rassemble les vecteurs d'argument et d'environnement.

Elle envoie alors un message au serveur de fichier qui détient le fichier à exécuter.

Le serveur de fichier vérifie alors les permissions d'exécution et fait les changements nécessaires dans l'appel d' "exec".
Par exemple, si le fichier est setuid et que le serveur de fichier peut le faire, il changera l'id utilisateur.

Le serveur de fichier décidera également si les programmes qui ont eu accès à l'ancienne tâche peuvent toujours avoir accès à la nouvelle tâche.

Si le serveur de fichier augmente les permissions ou exécute une image illisible, alors l'exec devra être placé dans une nouvelle tâche Mach afin de garantir la sécurité.

Après avoir décidé de la politique à appliquer vis-à-vis de la nouvelle image, le système de fichier appellera le serveur d'exécution pour charger la tâche.

Ce serveur chargera l'image en utilisant la bibliothèque BFD (Binary File Descriptor).

BFD supporte un grand nombre de formats de fichiers objets (gzip, bzip2, ...) ; presque tous les formats supportés seront exécutables. Ce serveur gère également les scripts débutant par le hashbang, les exécutant avec le programme indiqué.

En fait, plusieurs serveur exec peuvent exister sur le même système, chacun étant conçu pour supporter différents formats d'exécutables. Cela permet à Hurd d'accueillir et d'exécuter des programmes originellement développés sous d'autres systèmes voire pour d'autres machines (à la condition que le serveur exec soit capable d'émuler la machine). Les serveurs actuellement développés permettront d'exécuter des programmes pour Linux, FreeBSD, Solaris et MS-DOS

Serveur init

Le serveur init est lancé après exec. Il s'occupe de l'arrêt de Hurd et de Mach, lance le script /libexec/rc, amorce les autres serveurs importants et reboot en cas de crash des serveurs importants comme proc

Serveur proc

Le serveur proc est le serveur de gestion des processus. C'est un élement-clé car Mach ne connait pas le concept de processus mais seulement celui de tâche (un espace d'adressage qui contient des ressources, comme des threads ou des ports). Il permet l'utilisation de la norme POSIX. Les processus sont ainsi, pour proc, des tâches Mach accompagnées de tous les renseignement POSIX (PID, permissions (UIDS,GIDS), hiérarchie d'héritage, environnements (envp), arguments (argv), appels relatifs aux processus (wait, fork, ... implémentés au travers de la glibc)).
En raison des restrictions de Mach, les programmes doivent fonctionner en tant que root pour identifier toutes les tâches du système. Mais plusieurs serveurs de processus peuvent co-exister, chacun d'eux gérant leurs propres clients, et donnant leur propre vision de l'univers. Ces fonctionnalités du serveur de processus qui ne nécessitent pas de privilèges root peuvent être implémentées sous forme de serveurs par les utilisateurs.

Serveur auth

auth est le serveur d'authentification de Hurd.

Chaque nouveau serveur lancé par l'utilisateur doit s'authentifier auprès de auth.
Evidemment, un serveur auth tournant en root sera toujours présent.

Tous les programmes font confiance à auth.

Un utilisateur peut cependant fournir ses serveurs auth pour ses propres besoins, ceci ne viole pas la sécurité du système.
Quand un service doit authentifier un utilisateur, il communique avec son serveur d'authentification de confiance.
Si cet utilisateur utilise un serveur d'authentification différent, la transaction échouera et le serveur pourra refuser de poursuivre la communication.

auth gère les UIDs POSIX et les tokens (droit permettant à une application d'effectuer certaines actions).
Un token circule auprès des applications et donne l'autorisation à une application de faire son travail.

Le serveur auth crée les tokens, les fait circuler, et les détruit. On peut également dupliquer les tokens, ce qui est utile lorsque plusieurs programmes ont besoin de la même information, au lieu que l'un attende son tour pour obtenir la même information, il pourra l'obtenir en même temps qu'un autre programme.

Serveur password

password est le serveur de mot de passes. Il fonctionne sous root.

Serveur pfinet

Bien que n'étant pas indispensable à Hurd, pfinet est tout de même important car il assure la gestion réseau.

Il s'agit d'un serveur pour TCP/IP qui implémente la famille de protocoles PF_INET(IPv4).

2.2.2 Serveurs de fichiers

Serveur ext2fs

Le serveur ext2fs est le principal serveur de fichier de Hurd, celui qui implémente le système de fichier ext2.

Serveur fatfs

Le serveur fatfs implémente le système de fichier fat

Serveur nfs

Le serveur nfs donne aux applications l'illusion de fichiers locaux quand ils sont sur un autre ordinateur du réseau, traduisant leurs demandes sur les systèmes de fichiers en demandes sur les réseaux et traduisant les blocs reçus en fichiers.

Serveur isofs

Le serveur isofs permet d'utiliser des systèmes de fichiers de type ISO 9660 (CDs ou DVDs de données) traduit les blocs dun CD ou DVD en fichiers ou répertoires pour les applications (iso9660)

Serveur ftpfs

Le serveur ftpfs est un serveur pour serveurs de fichiers FTP.

Transition : Tous les serveurs du Hurd sont des traducteurs, exceptés auth et proc (pour lesquels toute tâche créée dispose automatiquement d'un moyen de communiquer avec ces serveurs).

2.3 Les traducteurs

2.3.1 Présentation des traducteurs

Définition

Un traducteur (translator) est un serveur Hurd qui permet de créer un point d'accès vers une ressource pour peu qu'elle se présente sous une forme arborescente . C'est un serveur qui a pour but de gérer la lecture et l'écriture du fichier auquel il est associé

Ceci est valable pour les systèmes de fichiers comme ext2 et iso9660, mais également pour des éléments plus originaux comme les archives (tgz, zip, etc.), les serveurs ftp, nfs ou Web. Ce n'est pas tout, il est possible d'appliquer cette méthode à une base de données et ainsi de faire apparaître le contenu d'une base dans un répertoire du système de fichiers.

Hurd a perdu la notion de fichiers, il s'agit plutôt d'un arbre comportant des noeuds avec un programme qui écoute chaque noeud, et c'est ce programme qui va lire ou écrire dans le noeud en fonction de ce qu'on lui aura demandé de faire.

L'utilisateur n'aura jamais un accès direct à un fichier, le fichier sera chargé en mémoire, l'utilisateur écrira en fait dans la mémoire, et le traducteur se chargera d'écrire dans le fichier.

Il faut cependant préciser qu'il s'agit là de quelque chose de bien plus puissant qu'une simple simulation d'un système de fichiers. Pour preuve, les translators permettront, à terme, des opérations aussi pratiques que la compilation des sources contenues dans une archive sans décompression. Le translator implémente alors la compression/décompression à la volée, ne laissant apparaître à l'utilisateur que le contenu d'un répertoire classique.

Pour résumer la situation, nous pouvons dire qu'un translator est capable d'associer n'importe quel répertoire de l'arborescence à n'importe quel programme.
Il n'y a pas besoin des privilèges de root pour installer ou modifier un traducteur, il suffit des droits d'accès à l'inode auquel est rattaché ce traducteur. De nombreux traducteurs n'ont pas besoin d'un fichier concret pour fonctionner, ils fournissent des informations par leurs propres moyens. C'est pourquoi les informations sur les traducteurs sont associées aux noeuds de larbre.

2.3.2 Commandes

settrans

La commande settrans qui permet d'activer/désactiver un translator. Cette commande s'utilise de la manière suivante :

settrans [OPTION...] NODE [TRANSLATOR ARG...]

où NODE est le noeud à utiliser (dans bien des cas, un répertoire), TRANSLATOR... le translator (habituellement placé dans /hurd), ARG les arguments adéquats pour le translator en question et, enfin, OPTION est la ou les option(s) parmi les suivantes :

-a –active rend le noeud actif
-c –create crée le noeud s'il n'existe pas
-L –dereference si le translator existe, place de nouveau par-dessus
-p –passive rend le noeud passif
-t –timeout=SEC temps limite pour le démarrage du translator (en secondes)
-x –exclusive place le translator uniquement si aucun autre n'est déjà en place
-g –goaway demande au translator de s'arrêter
-k –keep-active laisse fonctionner le translator déjà actif
-f –force force l'arrêt du translator
-R –recursive arrête également les translators dépendants
-? –help affiche un résumé de l'aide
-V –version affiche la version du programme

On peut attacher un translator soit de manière passive ("-p") ou de manière active ("-a") (ou les deux).

Par exemple, pour l'exemple du ftpfs, on l'utilise ainsi :

settrans -cagp ftp-debian /hurd/ftpfs / ftp.fr.debian.org

ftp-debian correspond au noeud sur lequel il faut attacher le translator
/hurd/ftpfs est le chemin du translator à exécuter
et ftp.fr.debian.org sont deux arguments passés à /hurd/ftpfs

On peut aussi demander à settrans de détacher tout translator d'un noeud en utilisant :

settrans -g ftp-debian

showtrans

Cette commande permet de savoir si un noeud est lié à un translator passif.

toto@tata : showtrans /home
/hurd/ext2fs /dev/hd0s7

Cela nous indique qu'au noeud "/home" est associé le translator "/hurd/ext2fs" avec l'argument "/dev/hd0s7".
Ainsi, au prochain accès à /home, c'est cette commande qui sera exécutée

fysopts

Cet utilitaire permet de connaître et de modifier les options du translator actuellement attaché à un noeud.

Par exemple, pour connaître l'état d'un point de montage, on peut simplement faire :

toto@tata : fsysopts /home
/hurd/ext2fs –writable /dev/hd0s7

On peut aussi indiquer de passer un système de fichiers en lecture-écriture avec la commande suivante :

fsysopts /home –writable

2.3.3 Traducteurs actifs/passifs

Traducteurs actifs

En mode actif : Le mode actif connecte le translator au noeud jusqu'à ce qu'il meurt (quand la machine est redémarrée). Les sorties standard et d'erreur sont connectées au terminal courant lors du lancement.
Par exemple, pour monter une partition ext2fs, vous pouvez exécuter settrans -a -c /mnt/hurd/ext2fs /dev/hd0s4. L'option -c créant le point de montage s'il n'existe pas déjà (point de montage qui, à propos, n'a pas forcément besoin d'être un répertoire). Pour démonter, vous devez juste essayer settrans -a /mnt.
De plus, un traducteur peut être activé par un utilisateur (il ne nécessite pas de droits root) et peut utiliser n'importe quelle bibliothèque (il existe par exemple un traducteur ftpfs, qui permet de «monter» un serveur ftp sur son disque dur). L'utilisation et le développement de traducteurs sont donc simplifiés.

Traducteurs passifs

Le mode passif, en revanche, se rapproche plus du fonctionnement d'un daemon. On va écrire dans le système de fichiers le fait qu'à tel noeud est attaché tel translator. Ainsi, lorsqu'un utilisateur accède pour la première fois à ce noeud, le translator est automatiquement lancé par le système de fichiers. Si le translator meurt, il sera relancé automatiquement dès la prochaine sollicitation. Comme l'information est écrite dans le système de fichier, cela survit même au redémarrage. Dans ce mode, les sorties standard et d'erreur ne sont connectées à aucun terminal.

Différences

Les traducteurs actifs peuvent mourir ou être perdus. Dès que le traducteur actif est tué (par exemple, parce que vous avez redémarré la machine), il est perdu pour toujours. Les traducteurs passifs, eux, ne sont pas éphémères et sont conservés dans les inodes, même lors des redémarrages. Ceci jusqu'à ce que vous les modifiez avec le programme settrans, ou que vous détruisiez les inodes auxquels ils sont attachés. Ce qui veut dire que vous n'avez pas besoin de maintenir un fichier de configuration de vos points de montage (comme /etc/fstab sous Linux).

Il est à noter que les translators sont permanents par défaut. Ainsi, il ne vous sera pas utile de retaper ces commandes à chaque démarrage du Hurd. Lorsque vous accéderez pour la première fois au répertoire où est «traduite» votre partition GNU/Linux, le translator sera automatiquement lancé avec les paramètres déjà utilisés. Si vous désirez mettre en place un translator de manière temporaire (accès à un serveur NFS par exemple), vous devez le spécifier explicitement à la commande settrans.

2.3.4 Exemples d'utilisation

Ainsi, vous pouvez éditer votre site Web en tapant simplement : emacs /mnt/ftp/www.monsite.org/index.html
créer un point d'accès vers votre partition, utilisée comme racine pour votre système GNU/Linux :
settrans -c /linux
/hurd/ext2fs
/dev/hd0s2
Cette commande appelle le serveur /hurd/ext2fs comme translator entre le répertoire /linux de votre système de fichiers GNU/Hurd et la seconde partition du premier disque dur.

Utilisation d'un translator pour fournir les paramètres réseau à notre périphérique eth0.

settrans -fg /servers/socket/2
/hurd/pfinet
–interface=eth0
–adresse=192.168.0.1
–gateway=192.168.0.10
–netmask=255.255.225.0

L'utilisation des informations inscrites dans /etc/fstab. settrans s'effectue de manière légèrement différente de la commande mount de GNU/Linux. Sous GNU/Linux, vous pouvez utiliser umount /mnt/cdrom et umount /dev/cdrom alors qu'avec GNU/Hurd, vous ne devez utiliser que le point d'accès et non le périphérique.

Un traducteur ext2fs sera associé à / pour monter une partition. Un autre traducteur ext2fs sera probablement associé à /home pour monter une autre partition pour les répertoires utilisateurs. Lorsquun programme voudra accéder à /home/toto/tata, il y aura d'abord un appel au traducteur associé à /. Comme il ne trouvera pas le fichier, il demandera au traducteur associé à /home d'accéder à toto/tata.

Imaginons une archive bz2 qui se comporterait comme un répertoire. Les programmes pourraient se déplacer dedans, effacer ou créer des fichiers. La compilation se ferait directement "dans" l'archive, sans que GCC ne s'aperçoive de rien. Il faut bien comprendre que l'archive n'est pas affichée comme un répertoire mais est un répertoire. Il suffit alors de lire et d'écrire dans ce fichier pour communiquer avec la tâche, sans se préoccuper des mécanismes IPC. Enfin, des translators destinés à gérer les protocoles Internet sont d'ores et déjà implantés.

Le plus connu est le FTP transparent où un site FTP est projeté dans un répertoire. L'utilisateur n'a qu'à se déplacer à l'intérieur des sous-répertoires et à copier les fichiers, sans avoir à connaître la moindre commande FTP. Ce mécanisme est prévu pour d'autres protocoles tels que HTTP et SMTP. De tels translators utilisés avec une interface graphique telle que Motif ou GNOME permettront d'obtenir un système réellement intuitif et orienté utilisateur. Enfin, il faut noter que, sauf interdiction explicite, n'importe quel utilisateur peut poser un translator sur un noeud de l'arborescence et bien sûr développer ses propres translators

CHAPITRE 3

Hurd en Pratique

3.1 Les distributions GNU/HURD

Aujourd'hui il existe peu de distributions proposant une base GNU/Hurd.

3.1.1 Les distributions disponibles

Debian GNU/Hurd
Commencé en 2002 avec la naissance de la version 0.2 de hurd, elle a évolué pour simplifier l'installation et la configuration du système.
Aujourd'hui disponible en version CD installable, la distribution Debian de hurd intègre l'installeurs de paquets debian (apt) dont la version Hurd possède déjà plus de 3000 éléments portés pour Hurd.
Elle lance l'ensemble des serveurs (le Hurd) sur le micronoyau GNU/Mach.

Installations Croisées
A partir d'un système GNU déjà installé (notamment GNU/Linux) on peut construire la base du système de GNU/Hurd.

3.1.2 Debian GNU/Hurd

Installation du système Debian GNU/Hurd

Voici une description non complète de l'installation de la version CD de la distribution Debian, à partir de la version K10 fournie de 4 CDs.
Il est apparemment possible de trouver sur internet une version K11 avec un CD minimal d'install par le réseau.

Partitionnement La Debian GNU/Hurd étant une distribution encore au stade expérimental, il est donc sage de garder une installation d'un système comme LINUX.

Format des partitions : Hurd ne dispose pas encore de serveur officiel supportant les systèmes de fichiers journalisés, même si les developpeurs de Hurd garde un code de partition (0X83) libre pour un jour la création d'un système de fichier propre a GNU/HURD.
Pour l'instant, les types de partitions qui ammorcent GNU/Hurd sont de type EXT2.
Chose très importante, GNU/Hurd lié a GNU/Mach à besoin d'un fichier d'échange conséquent

car dès que la mémoire vive est saturée, il se sert du fichier d'échange et de la pagination.
Attention : Si le système ne dispose pas d'une partition pour la pagination, le noyau paniquera !

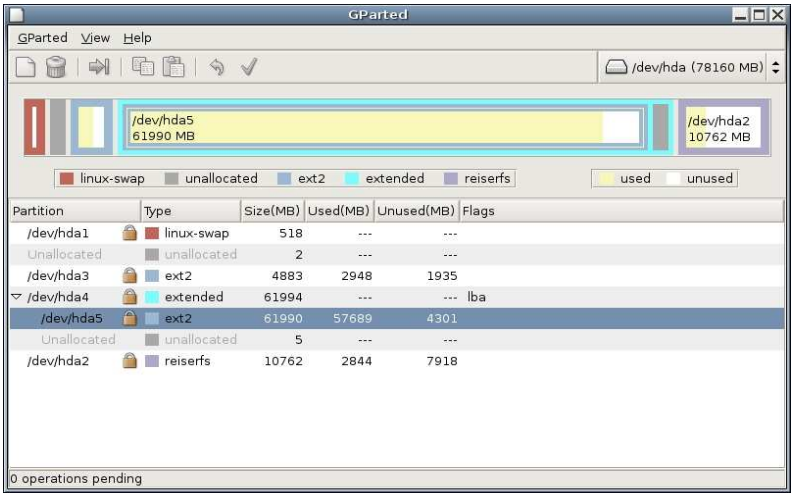
Quant à l'utilisation de la partition Swap, on peut très bien la partager entre linux et hurd sans aucun problème...

Nommage des disques : Dans les deux systèmes les périfériques sont représentés comme des fichiers placés tous dans le répertoire /dev

- GNU/Linux préfixe le nom de ses partitions par hd, puis lui attribue une lettre en fonction de sa position maître/esclave et du bus IDE duquel il dépends.
  Synthétiquement le nom d'une périférique pour un lecteur IDE devrait être :
  "hd[a-z][1,n]"

- GNU/Hurd dispose d'une numérotation similaire par un suffixe hd suivi lui d'un nombre de 0 à n désignant le périférique disque.
  Pour désigner la partition d'un disque, on suffixe le nom du disque de "s[1-n]" , on retrouve une correspondance directe avec les noms utilisés par linux, certes peu importante mais vitale pour amorcer un système et des partitions.

Modifier la table des partitions : Modifier une table de partition s'avère assez délicat, surtout si on dispose de plusieurs systèmes d'exploitations.
Il existe à partir d'un poste linux un utilitaire convivial : parted dont la version graphique provenant de gnome gparted est particulièrement facile et maniable.



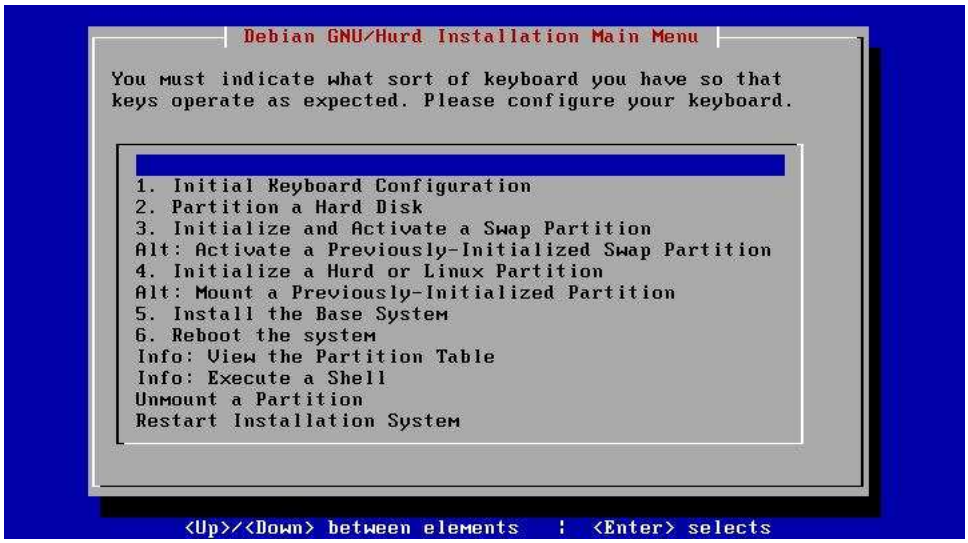
Avec les versions Debian actuelles, Hurd reconnait les partitions de plus de 2 GO.
Pour formater une partition hurd, il faut utiliser l'utilitaire mke2fs en ligne de commande et lui indiquer que hurd est le propriétaire du système de fichier avec l'option -o.
La commande exacte est *$> mkefs /dev/[identifiant du périférique disque] -o hurd*
L'installeur du CD debian propose un utilitaire de partionnement et Formate les partitions avec le même outil que précédemment.
L'utilitaire de formatage du CD est à utiliser sans risque, par contre l'utilitaire de partitionnement est un peu vétuste...
Il est donc sage d'utiliser un outil de partitionnement plus évolué comme gparted ou encore fdisk (sous linux) qui restent des références.

Installer le système de base

Méthodes d'installations : Après création de partitions dédiées au système Hurd, on passe maintenant à la mise en place du système de base.

A partir de l'archive GNU : Il est possible d'extraire une archive qui contient le noyau Mach, l'ensemble des serveurs Hurd et les utilitaires et commandes de base d'UNIX.
Ces archives sont téléchargeables à partir des mêmes serveurs qui proposent le téléchargement des CDs debian Hurd.
Sous un système Linux,après avoir monté la partition avec
$> mount /dev/[partition racine de hurd] /[chemin vers un répertoire où monter le système de fichier de hurd]
on peut extraire l'archive dedans avec
$> tar -xvzf /[chemin de montage de gnu-hurd].

A partir du CDROM debian : Après avoir amorcé la procédure d'installation du CDROM qui lance une ancienne version du noyau linux, l'installeur Debian s'execute.
Consécutivement il est demandé de :
- Choisir le type de clavier (presque inutile car ne configure pas mach pour utiliser ce type de clavier par la suit)
- Modifier la table des partitions (Réservé aux utilisateurs avisés).
- Choisir un périphérique d'échange (Mémoire Swap)
- Choisir une partition par deux fois, dont une doit être la racine du système de fichier hurd.
- Installer le système de Base : décompresse en fait l'archive contenant le système GNU/Mach
- réamorcer le système



Amorcer GNU/Mach : Le CD Debian/Hurd visant le côté pratique de la chose, il n'inclue pas l'installation d'un gestionnaire d'amorçage du genre Grub.
C'est pourquoi il est très utile de disposer d'un système GNU/Linux avec déjà un gestionnaire de multiboot installé.

L'amorçage du noyau Mach est assez facile, il faut se rappeller du bon préfixage pour lui indique le préfixe de partition racine.
Pour créer une entrée qui lancera Hurd dans grub , il faut ajouter ces quelques lignes au fichier /boot/grub/grub.conf ou menu.lst :

title [GNU/HURD] GNU/Mach
kernel (hd0,4)/boot/gnumach.gz root=device :hd0s5 -s

module (hd0,4)/hurd/ext2fs.static –multiboot-command-line=$kernel-command-line \

–host-priv-port=$host-port \
–device-master-port=$device-port) \
–exec-server-task=$exec-task -T typed $root $(task-create) $(task-resume)

module (hd0,0)/lib/ld.so.1 /hurd/exec $(exec-task=task-create)
La ligne kernel indique le chemin vers le noyau ainsi que le les paramètres qui lui sont donnés.
La première ligne module lance le serveur de fichier de fichier ext2 destiné à accueillir la partition racine.
La seconde ligne lance le serveur exec qui se chargera de la création des processus.
+ **Notes sur les variables :** Dans ces lignes de commande les variables que l'on passe entre $ plus accolades seront remplacées par grub lors du lancement des serveurs.

Pendant les deux premieres séquences de démarrage, il faudra donner au noyau l'option -s pour démarrer en mode **monoutilisateur**, on pourra donc créer une section temporaire dans notre fichier de configuration de grub destinée à cet effet.

Configurer le système de base : On dispose ici d'un système presque opérationnel, qui fonctionne sur un clavier qwerty standard.
En mode monoutilisateur,on doit exécuter le script ./native-install après avoir fixé la variable TERM
export TERM=mach. Il opère en deux étapes :

- Premiere Phase : Configuration des traducteurs
- Deuxieme Phase : Configurations du système et avec l'utilitaires Debconf.

Configuration de la distribution debian GNU/HURD : Maintenant le système est prêt a démarrer en mode multi-utilisateur, on obtient une ligne de commande où on peut taper :

Connection d'un utilisateur : Au démarrage seul l'utilisateur root est disponible. *$> login [nom d'utilisateur]*
On obtient alors la console mach dans laquelle tourne le un shell bash.

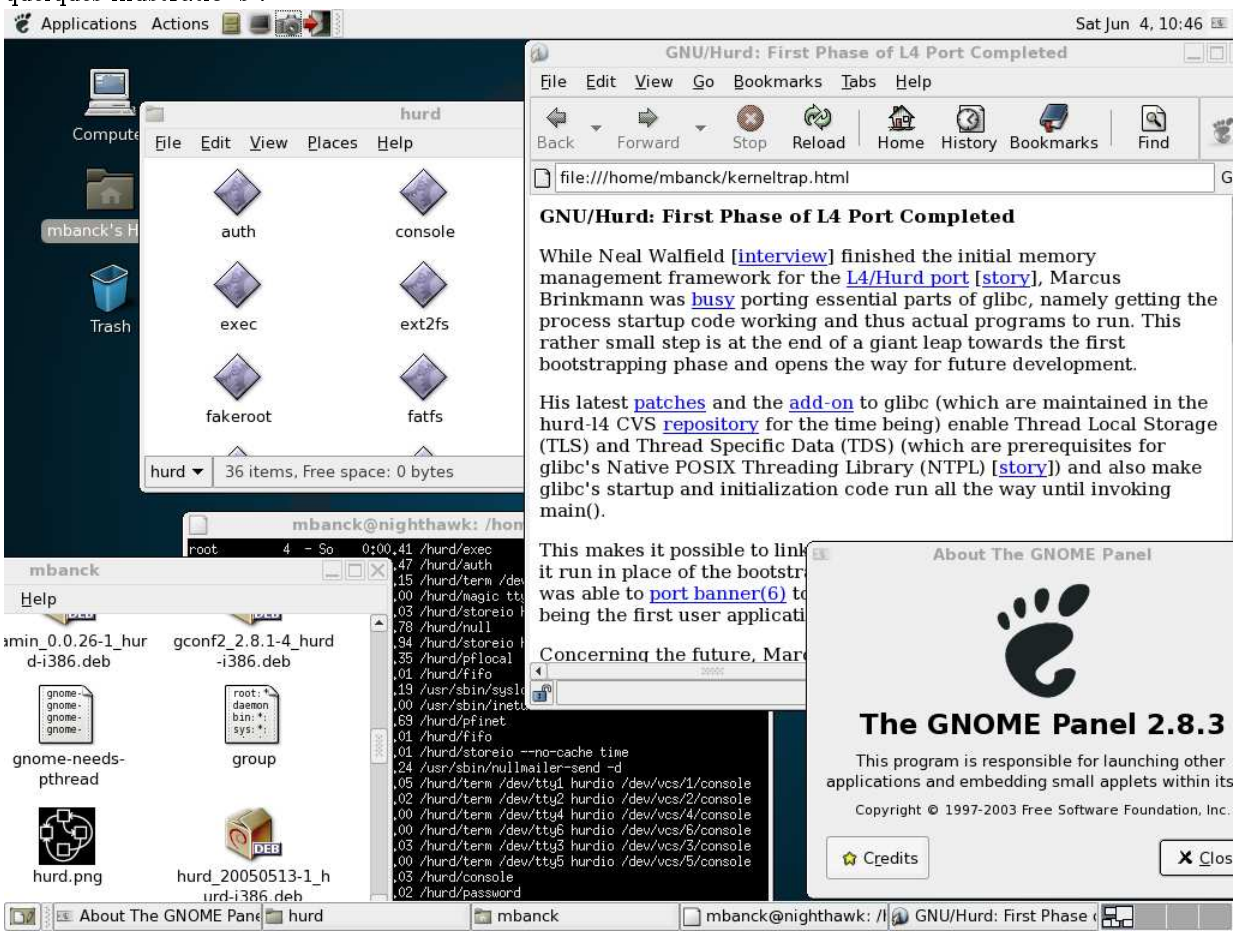
Gestion des périfériques : Comme sous GNU/Linux, l'ensemble des fichiers correspondant au périphériques se trouve dans /dev. Au besoin, on peut associer un traducteur au fichier périphérique qui interfacera le matériel géré par le noyau au travers du serveur associé au traducteur.
Pour créer un nom de fichier associé au périphérique, on passe par le script /dev/MAKEDEV, par exemple pour créer le fichier lié au lecteur cdrom :
$> ./MAKEDEV hd2 ... à partir du répertoire /dev.
Et on invoque un traducteur de cette façon :
settrans -a /cdrom /hurd/iso9660fs /dev/hd2 , ce qui correspond à la commande linux
$>mount /dev/hdc /cdrom -t iso9660fs
L'option -a indique que le traducteur est actif.

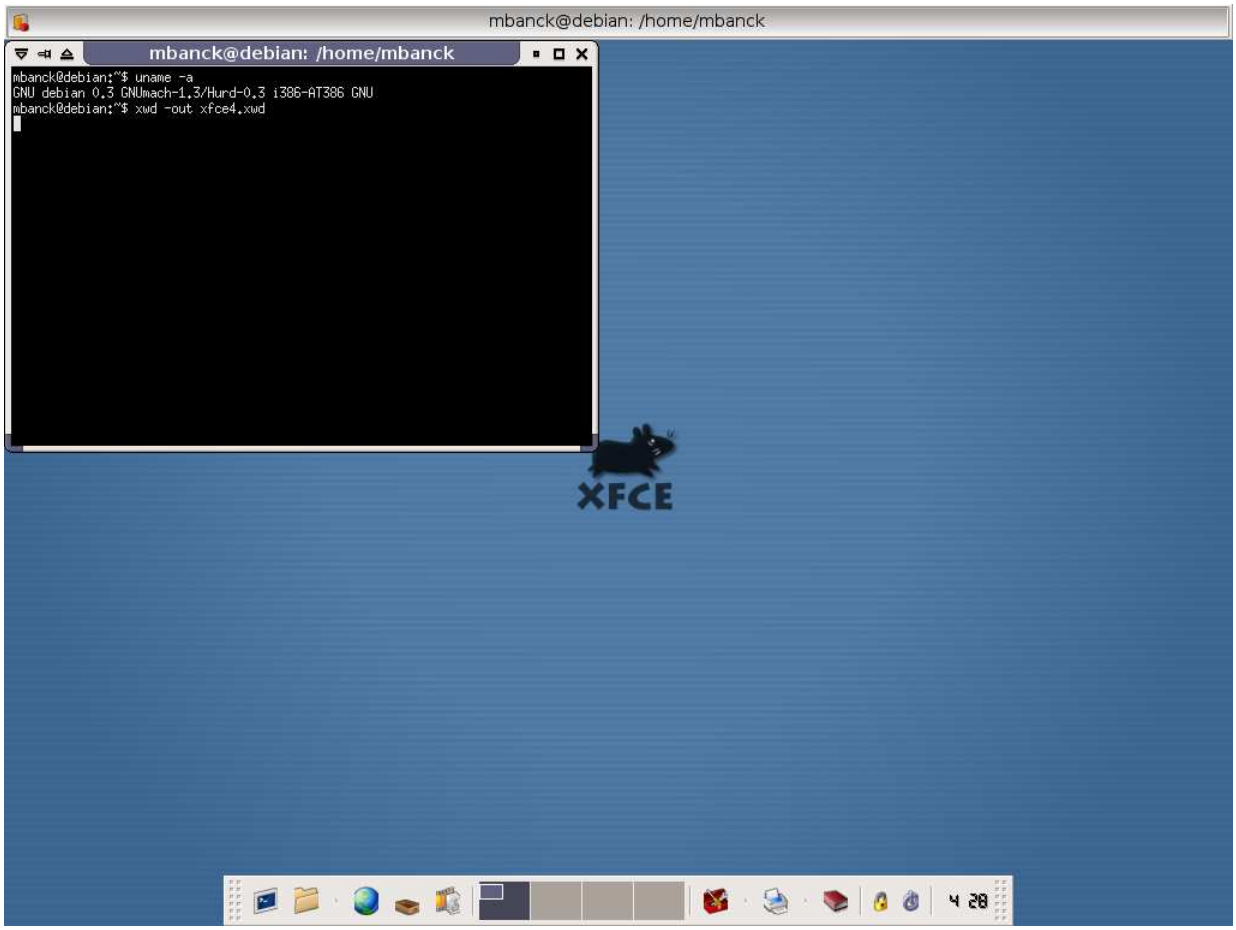
Configuration de fichier /etc/fstab : Son utilisation est similaire a celle de linux :

Type	Chemin du périférique	point de montage	type de partition	options		
Partition d'échange	/dev/hd0s4	none	swap	sw	0	0
Partition commune	/dev/hd0s6	/home	ext2	defaults	1	1
Lecteur	/dev/hd2	/cdrom	iso9660fs	ro,noauto	1	1

Installation de quelques applications portées pour hurd : Ce sont majoritairement des applications GNU comme gcc 4.0, bison, flex et bien d'autres mais aussi de nombreuses librairies... Elles sont toutes disponibles sur le CD1 de la debian GNU/HURD.
Pour les installer, il faut d'abord monter le cdrom de debian, puis lancer le script du cd par la commande
/cdrom/upragre/install.sh, le script demande la confirmation sur l'installation de paquets non certifiés debian, il faut bien sur répondre oui et voilà un système presque fonctionnel installé !.

Installation de XFREE86 4.0 Toujours sur le même CD, on peut utiliser la commande */cdrom/upgrade/gui.sh* pour installer un serveur X porté pour GNU Hurd.
Les gestionnaires de fenêtres tels que KDE ou GNOME ne fonctionnent pas encore car même si l'implémentation des threads sous GNU/Mach est faite et assez évoluée, de nombreuses parties des pthreads POSIX ne sont pas encore implémentées, ce qui empêche donc de les faire complétement fonctionner.
On peut apparement trouver des implémentations qui permettent de les faire tourner, en voici quelques illustrations :





Utilisation du gestionnaire de Paquets Debian a porté pour sa distribution son gestionnaire aptitude.

Ajouter les paquets disponibles sur les CDROMS : Il suffit de monter le cdrom et de taper la commande
$>apt-cdrom add ce qui ajoutera le cdrom dans le fichier source.list de configuration d'aptitude et qui créera le cache de l'arborescence des paquets des 4 CDS.
On pourra ainsi entrer les commandes
$>apt-get nomdulogiciel pour installer automatiquement le programme voulu ainsi que toutes ses dépendances, seul reste l'ennui de devoir monter et démonter les cdroms demandés pour l.

L'installation Croisée

Beaucoup plus pénible à réaliser, l'installation croisée du système hurd permet pourtant de garder un contrôle total sur le système.

3.2 Les changement apportés par GNU/HURD

Le système d'exploitation GNU/Hurd a été maintenu face a GNU/Linux pour que le monde des logiciels libres dispose d'une architecture plus puissante, fonctionnelle et portable qu'un simple

système UNIX.

3.2.1 Les avancées

Les traducteurs

L'ensemble des traducteurs fournit la plupart des fonctionnalités impressionnantes du système GNU/Hurd, ils apportent de nombreuses améliorations du point de vue sécurité, fonctionnalités d'utilisation.

Wide home system directories Sa désactivation a pour effet de traduire seulement pour l'utilisateur connecté les répertoires auxquels il a accès, il ne peut voir ceux des autres.
Ainsi, le système se préoccupe seulement des répertoires associés à un traducteur et invoquera un nouveau traducteur automatiquement si l'utilisateur courant a les droits pour qu'un traducteur soit associé au répertoire.

La Console mach et la console de hurd Au démarrage, GNU/Hurd lance la console mach sur laquelle on peut attacher la console hurd.
Cette console est très proche des la console linux déjà connue (implémentée par exemple par agetty) et qui est facilement détachable avec les touches CTRL + ALT + RETOURN ARRIERE.
Le système d'exploitation restant encore en phase expérimentale et l'intégration de cette console se fait encore lentement.

Traducteur de systèmes Le point fort des traducteurs est d'améliorer la portabilité des applications, il a ainsi été évoqué de se servir des traducteurs comme d'interfaces avec des programmes compilés pour d'autres machines, servant ainsi d'émulateur.
Aujourd'hui vu l'évolution du système bien sur aucun traducteur ne propose cette fonctionnalité...

Développement et modularité Le réel point fort des traducteurs est leur intégration au sein de la philosophie du monde libre.
Richard Stallman a évoqué la difficulté de programmation d'un noyau comme linux, qui ne juge pas exactement libre, car peu accessible au point de vue programmation.
Les traducteurs permettront eux aux développeurs de créer plus librement l'accès à des systèmes de fichiers, des fonctions réseau avancées ou des interfaces avec des périphériques.
Comme exemple on peut donner celui de ftpfs qui prend en charge la connection ftp comme un terminal unix.
On peut aussi imaginer qu'un traducteur serait capable de proposer l'accès à un appareil du type Scanner en proposant la liste des fichiers scannés...

Bien sur ce monde reste encore peu étendu et ne verra véritablement le jour que si un jour Hurd remplace totalement linux et sa communauté, ce qui permettra de voir l'avènement de nombreuses fonctionnalité sur ce système.

3.2.2 Les différences

Autre noyau, autres idées, autres conventions, quelques divergences entre les mondes Unix/Linux et GNU/Hurd existent.

3.3 Hurd en évolution

3.3.1 Les projets portant sur Hurd

Portage sur le noyau L4

les avantages : Le noyau L4 développé par l'équipe L4Ka en allemagne est très prometteur, il intègre les dernieres évolutions en terme de micronoyau et attends des performances nettement supérieures a celle de mach.
De nombreuses personnes ont alors proposé de se servir du noyau L4 pour faire tourner l'ensemble des serveurs Hurd.
Aujourd'hui le noyau GNU-Mach n'évolue plus et la communauté autour de hurd attend énormément de L4 et de son successeur L5.

le probleme : Théoriquement, changer de noyau devrait être entièrement réalisable selon la philosophie de hurd et c'est donc avec joie que beaucoup de personne ont pensé que le salut par L4 arriverait rapidement...
Ce qu'il faut rappeler c'est une seule l'équipe développait GNU-Mach et les serveurs HURD, bref tout le système, et qu'ils ont dramatiquement lié les serveurs de Hurd au noyau Mach.
Comble de d'ironie, le dernier cri en matière d'architecture système qui se voulait portable au possible ne l'était plus...

Portage de Hurd sur L4 Bien sûr, le projet L4 est né il y a quelques années et depuis les serveurs GNU deviennent compatibles avec L4, et on commence à voir apparaître des noyaux L4 qui font tourner le Hurd, mais certainement pas toutes ces fonctionnalités.
Par exemple, debian Hurd installe les entêtes L4, le noyau lors de l'installation du système GNU, bien sûr il doit certainement necessiter une configuration poussée et simplement changer le chemin vers le noyau dans Grub ne fonctionne pas.

L4 et Gnuppix Le livecd Gnuppix intègre le noyau L4 comme base du système hurd, on peut donc y porter un grand intérêt.
Malheureusement, aujourd'hui Gnuppix a entièrement disparu de la toile comme de nombreuses références à des projets sur Hurd.

Gentoo Hurd Un projet non officiel et une annonce de projet officiel ont été diffusés pour annoncer la naissance de projets visant à adapter la distribution Linux Gentoo sur un système Hurd.

Portage : L'équipe du projet non officiel de gentoo annonce qu'un patch est disponible pour Hurd afin d'adapter le système de gestion de sources, nommé PORTAGE, sous GNU/HURD.
Portage est la base de la distribution gentoo, qui installe n'importe quel programme en compilant sa source, au préalable empaquetée et préparée dans un paquet nommé ebuild.
Ce système d'installation est bien plus évolué que celui de toute autre distrubtion linux, permettant un masque de version, des optimisation spécifiques et automatiques à la compilation ainsi qu'un système bien construit et très bien documenté.

Disponibilité : Même on trouve énormément de références a ce projet, tout comme Gnuppix, toutes les sources et installations ont entièrement disparues.

Conclusion

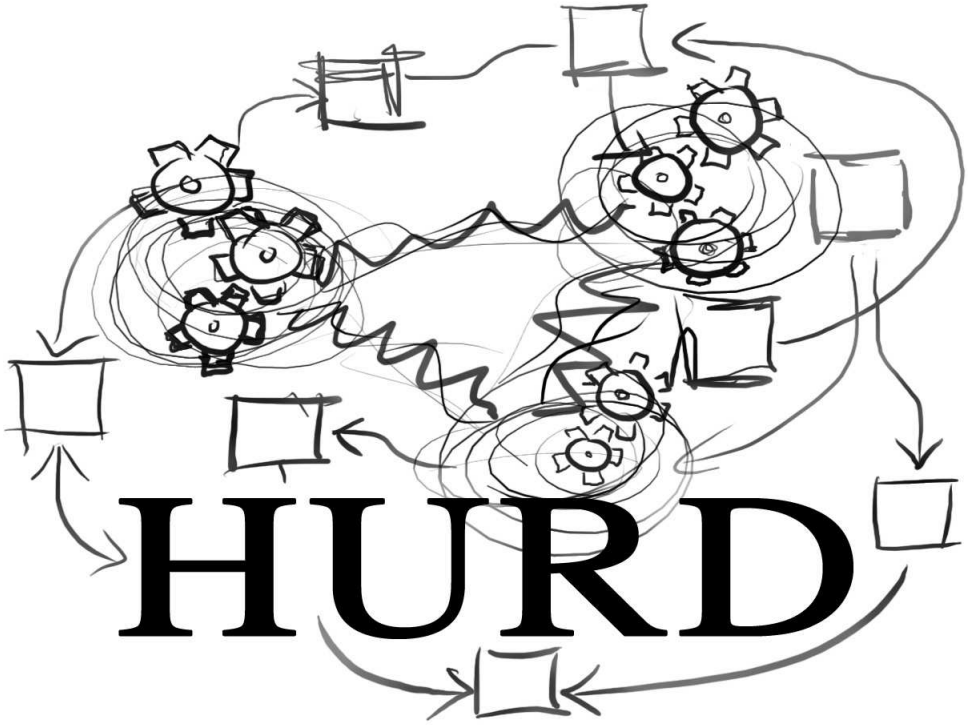
3.4 L'avenir des micro-noyaux

Les micro-noyaux gagnent de plus en plus de popularité, leur concept commence à plaire à de nombreuses personnes.
Aujourd'hui, presque aucun grand système d'exploitation n'utilise de micro-noyau pur, ils utilisent des versions enrichies et hybrides comme Windows NT et MacOSX.
Ce dernier est d'ailleurs l'adaptation du micro-noyau Mach enrichi avec des parties du système BSD, pour rééquilibrer les performances du système certaines parties d'un système monolithiques sont intégrées dans le noyau.

Hurd , un système a microyau pur ? Hurd lui correspond à une vision puriste du micronoyau, en intégrant aujourd'hui L4 il se rapproche de plus en plus de la version Nano-noyau, seulement si L4 apporte un gain notable niveau performances, les choix des systèmes tels que MacOS voir WindowsNT n'ont pas été fait par hasard, faire tourner une partie proche du noyau augmente sensiblement ses performances, et face aux fonctionnalités, les performances sont aujourd'hui un choix draconien dans la sélection d'un système d'exploitation.

3.5 L'avenir de Hurd face à Linux

Pour de nombreux utilisateurs de linux, hurd peut être décrit de la façon qui suit :



En effet même Linus Trosvalds ne manque pas de remarque ironiques sur Hurd du genre "Droguez vous et vous finirez comme les gens de Hurd", et il n'est pas le seul a penser que le système tel que le rêve Richard Stallman n'est pas la bonne approche.
Pour avancer celà il s'appuie sur la complexité de communication du noyau vers le serveurs, ou vers les serveurs entre eux qui donne au système un temps de réaction pénible.

Le projet Hurd abouttira-il ? D'une certaine façon la réponse est évidemment oui, le système est aujourd'hui opérationnel et utilisable tout comme une distribution linux.
De nombreux projets vont se mettre en place pour proposer d'ici quelques années des versions de hurd faisant tourner GNU-Mach ou encore L4 voire L5 à la portée d'utilisateurs moins avancés.
Car, s'il est vrai que Hurd fait tourner de nombreux logiciels, il n'est toujours pas à la portée de n'importe qui et son installation voir surtout sa configuration et sa prise en main nécessitent aujourd'hui énormément de travail et de recherches.
Sur ce point, on peut le mettre en corrélation avec Linux étant donné qu'il en reste proche, même si les divergences entre les deux mondes entraînent de nombreuses complexités pour la configuration du système pour l'utilisation.

Aujourd'hui Hurd manque énormément de ressources qui seront certainement apportées en partie par les développeurs et la communauté de L4.
Hurd semble dans le creux de la vague, il possède de nombreux atouts pour décoller, énormément de projets et d'outils et il se divise déjà en plusieurs branches telles que Mach/OSKit, Mach/Hurd et L4/Hurd.
C'est évidemment par L4 que passent tous les espoirs concernant Hurd, qui souffre d'une comparaison avec son Cousin monolithique...

Il est probable que Hurd trace l'avenir des systèmes a micro-noyau, mais celà lui profitera-il ?
En effet, le monde des systèmes d'exploitation reste en changement perpétuel et il y aura certainement d'autres noyaux semblables a Hurd qui pourront soit prendre sa place comme l'a fait Linux, soit lui servir de base pour le propulser en avant du projet GNU et donner naissance enfin à GNU OS.

Annexes

Guide d'installation du Hurd
http ://web.walfield.org/pub/people/neal/papers/hurd-installation-guide/french/hurd-install-guide.html

Noyau

http://www.clubic.com/wiki/Noyau
http://fr.wikipedia.org/wiki/Noyau_(informatique)

Noyau Linux
http://www.pcinpact.com/definition_117.htm
Micro noyau GNU Mach
http://www.trustonme.net/didactels/147.html

Présentation de Linux
http://glinux.freezee.org/quoilinux.php
http://linux.ensimag.fr/introlinux.html

Présentation de Hurd
http://www.supinfo-projects.com/fr/2004/decouverte_hurdfr/
http://nemy.bisouilles.net/Doc/HURD.pdf
http://membres.lycos.fr/nek/os/hurd.htm
http://magnux.free.fr/hurd/
http://wiki.hurdfr.org/index.php/Le_Hurd
http://www.hurdfr.org/pages/doc/pres.pdf
http://okki666.free.fr/docmaster/articles/linux091.htm
http://kilobug.free.fr/hurd/pres-en/abstract/html/
http://www-igm.univ-mlv.fr/~dr/XPOSE2002/Hurd/
http://www.empyree.org/informatique/hurd.html
http://www.chez.com/keep/KoM/hurd.htm
http://fr.wikipedia.org/wiki/GNU_Hurd
http://fr.wikipedia.org/wiki/Hurd
http://medias.2005.rencontresmondiales.org/topics/os/papers/le-mignot-hurd.pdf
http://medias.2005.rencontresmondiales.org/topics/os/slides/brinkmann-hurd.pdf
http://syn.sceen.net/linux_party/hurd.pdf
http://www.clubic.com/wiki/HURD

Les traducteurs
http://chl.be/glmf/www.linuxmag-france.org/old/lm10/hurd.html

La sécurité sous Hurd
http://web.walfield.org/pub/people/neal/papers/

The GNU Hurd Reference Manual
http://es.gnu.org/~jemarch/downloads/hurd.pdf

Le Hurd sur SourceForge
http://hurd.sourceforge.net

Le projet Hurdfr
http://www.hurdfr.org

Page officielle GNU
http://www.gnu.org/software/hurd/
http://www.gnu.org/software/hurd/hurd.html
http://www.gnu.org/software/hurd/faq.en.html
http://www.gnu.org/software/hurd/devel.html
http://www.gnu.org/brave-gnu-world/issue-1.fr.html

Projet Debian/Hurd
http://www.debian.org/ports/hurd/

Présentation de Linux
http://glinux.freezee.org/quoilinux.php
http://linux.ensimag.fr/introlinux.html

Guide d'installation du Hurd
http://web.walfield.org/pub/people/neal/papers/hurd-installation-guide/french/hurd-install-guide.html

Projet Debian/Hurd
http://www.debian.org/ports/hurd/

Divers
http://linuxfr.org/2005/02/05/18247.html
http://www.cs.pdx.edu/~trent/gnu/hurd/