Serveur Web de Base de Données mSQL / perl HOWTO

par *Oliver Corff* ,
version française *Nicolas Caillaud* v0.1, 17 September 1997

Ce Mini HOWTO, très largement inspiré de l'article de Michael Schilli `Gebunkert : Datenbankbedienung mit Perl und CGI`, publié dans le magazine informatique allemand iX du mois d'aout 1997, explique comment construire une base de données client/server SQL sur le Web, utilisant HTML pour l'interface utilisateur.

Contents

1 A propos de ce Document

1.1 Lecteurs concernés

Ce document devrait être utile à ceux qui veulent mettre en place un serveur de base de données sur le Web, mais qui ne savent pas quels logiciels sont nécessaires, ni comment les installer. Il fournit toutes les informations utiles pour faire fonctionner une base de données SQL sur un serveur Web ; il *ne* rentre *pas* dans les détails de la programmation CGI, ni n'explique le langage SQL. D'excellents ouvrages existent sur ces sujets, et le but de ce document est seulement de fournir une plate-forme sur laquelle un utilisateur pourra étudier la programmation CGI et le langage SQL.

Pour faire tourner un système SQL sur une petite échelle (et non pas l'exemple classique du système de réservation d'une importante compagnie aérienne, ou le système de gestion de base de données d'une mission spatiale), il suffit d'avoir les logiciels décrits dans ce document, et la documentation les accompagnant. Le manuel utilisateur de msql (la base de données décrite ici) fournit suffisamment d'informations sur SQL pour vous permettre de construire votre propre base.

Le lecteur doit savoir comment récupérer des fichiers par `ftp` s'il n'a pas les CD-ROM adéquats, et comment construire des exécutables à partir des fichiers sources. En tout état de cause, toutes les étapes décrites dans ce document ont été testées sur un système réel, et devraient fonctionner sur le système du lecteur.

1.2 Conventions typographiques

Une commande utilisateur :

```
# make install
```

Affichage d'un programme :

```
    Program installed. Read README for details on how to start.
```

Exemple de fichier de code :

```
# Un commentaire
char lettre;
```

2 Introduction

On peut raisonnablement supposer que des bases de données contenant de gros volumes de données, ou un schéma relationnel compliqué (comme, par exemple, une base lexicale pour un langage parlé), doivent être accessibles à beaucoup d'utilisateurs à la fois. De même, il doit être possible d'utiliser diverses plates-formes matérielles et logicielles existantes pour construire le système final. Pour réduire les coûts de mise en oeuvre, une seule partie du système a réellement besoin de puissance : le serveur de la base de données ; les stations utilisateurs doivent seulement afficher des données et accepter des commandes utilisateurs, mais le traitement proprement dit est fait sur une seule machine, justifiant ainsi le terme "base de données client - serveur". De plus, l'interface utilisateur doit être facile à configurer, et requérir le moins de traitement possible sur le client.

Les éléments suivants (protocoles, logiciels et plus généralement concepts) sont utilisés pour construire un système conforme à ces critères :

Linux

est le système d'exploitation. Il s'agit d'une implémentation stable d'Unix, multi-utilisateurs, multi-tâches, avec support réseau complet (TCP/IP entre autres). A part le coût du support et de la transmission, il est gratuit et livré sous forme de distributions, qui incluent généralement l'indispensable, du Système d'Exploitation lui-même au traitement de texte, outils de développement logiciel, langage de script, générateurs d'interfaces, etc.

HTML

Le langage HTML (HyperText Markup Language) est utilisé pour construire des interfaces de systèmes réseaux comme des Intranets, des serveurs Web (WWW). HTML est extrêmement simple et peut être généré à partir de n'importe quel éditeur de texte ASCII.

Navigateurs

Les navigateurs sont des applications en mode texte (par exemple Lynx) ou en mode graphique (par exemple Mosaic, Netscape, Arena, etc.) destinées à la lecture et à l'affichage de documents HTML. C'est le seul logiciel directement manipulé par l'utilisateur de la base de données. Grâce aux navigateurs, on peut afficher différents types de données (texte ou images), et communiquer avec des serveurs HTTP (voir plus loin), hébergés par à peu près n'importe quel modèle d'ordinateur sur lequel un navigateur est disponible.

Serveurs HTTP

Un serveur HTTP fournit l'accès à une zone de l'ordinateur contenant les données publiques d'un réseau. Il supporte le protocole HTTP et fournit l'information demandée par l'utilisateur.

SQL

SQL (Structured Query Language) est un langage de manipulation de données dans une base relationnelle. Sa grammaire est très simple et constitue un standard largement supporté dans l'industrie. Les bases de données SQL sont au coeur même du concept classique de base de données Client - Serveur. Des systèmes SQL reconnus existent, comme Oracle, Informix, etc. ; on trouve aussi des bases comme msql, mysql, postgresql, pratiquement gratuites lorsqu'elles sont utilisées dans un cadre scolaire ou universitaire.

CGI

CGI (Common Gateway Interface) est l'interface de programmation entre le système supportant les données (dans notre cas, le système SQL) et le protocole réseau (HTML, bien sûr). Les interfaces CGI peuvent être construites en utilisant beaucoup de langages de programmation, dont l'un des plus populaires est perl.

Perl

Perl est un langage de script particulièrement puissant, qui combine les avantages du C, des différents shells, et des langages de manipulations de flux comme awk ou sed. Par exemple, perl possède de nombreux modules de manipulation de base de données SQL.

3 Procédure d'installation

3.1 Matériel requis

Aucune supposition ne peut être faite sur les besoins en matériel d'un serveur de base de données. Cela dépend trop du nombre d'utilisateurs, du type d'application, de la charge du réseau, etc. Dans un environnement comprenant peu d'utilisateurs et un trafic réseau faible, un 486 ou équivalent, avec 16 MO de mémoire vive, peut être suffisant. Linux, le système d'exploitation, est très efficace en termes de ressources, et peut fournir suffisamment de puissance pour faire tourner un grand nombre d'applications en même temps. Bien sûr, un processeur plus puissant et plus de mémoire vive signifient plus de puissance, mais la quantité de mémoire vive est plus importante que le processeur. Plus le système a de mémoire vive, moins il est obligé, en cas de besoin, de swapper les processus les plus gourmands en mémoire sur le disque.

Avec un système équipé de 32 MO de mémoire vive et d'un bus PCI, les recherches et opérations de tri peuvent être faites sans avoir recours au(x) fichier(s) d'échange (swap), donnant d'excellents résultats.

L'installation décrite dans cet article a été faite sur un IBM 686 à 133MHz, avec 32 MO de mémoire vive et un disque dur IDE de 1.2 GO. La suite du document présente les étapes indispensables à une installation complète.

3.2 Logiciel

Les logiciels décrits dans cet article sont disponibles sur Internet, ou sur CD-ROM. Les produits suivants sont utilisés :

- Distribution Red Hat 4.2, parue pendant l'été 1997, disponible sur CD-ROM (Red Hat Linux Power-Tools : 6 CD-ROM complets et prêts à être utilisés) ou sur Internet, sur le site de

  RedHat `http://www.redhat.com` ;

- Base de données SQL msql : disponible en deux versions. Les différences entre ces deux versions résident dans le nombre maximum de transactions supporté, l'interface d'administration, etc. La version la plus ancienne, 1.0.16, est disponible sur les sites miroirs de Sunsite. L'exécutable au format ELF peut être trouvé sur *Sunsite* , ou sur CD-ROM (en l'occurrence le disque 4 de l'InfoMagic Linux Developper's Resource, ensemble de 6 CD-ROM, décembre 1996), ou chez *InfoMagic* . La version la plus récente, 2.0.1, peut être obtenue directement de la page Web d' *Hughes* , en Australie, ou sur de nombreux sites miroirs dans le monde ;

- Perl du CPAM : The Comprehensive Perl Archive Network. Sur le CD-ROM WalnutCreek, ISBN 1-57176-077-6, mai 1997 ;

- L'exemple de programme CGI de Michael Schilli, dans le magazine informatique iX d'août 1997, pages 150-152, disponible par *ftp* .

3.3 Installation de l'OS

Linux est installé à partir de la distribution Red Hat Linux 4.2. Pour réussir à l'installer, la machine doit avoir un lecteur de CD-ROM accessible à partir de MSDOS, un lecteur de CD-ROM bootable, ou bien encore une disquette de boot préparée selon les instructions du CD Linux.

Pendant l'installation, l'utilisateur peut sélectionner et configurer de nombreux paquetages logiciels. Il convient de sélectionner les suivants :

- support du protocole TCP/IP
- le serveur http Apache
- le langage Perl
- le système XWindow
- les navigateurs Arena (graphique) et Lynx (mode texte).

Tous ces paquetages sont fournis avec la distribution Linux. Si vous ne les installez pas maintenant, vous pourrez le faire plus tard en utilisant `glint`, le gestionnaire graphique de paquetages logiciels. Assurez-vous d'être connecté comme utilisateur `root` lorsque vous les installerez.

Il n'est pas du ressort de cet article de décrire l'installation réseau, ni la procédure d'initialisation. Pour cela, consultez la documentation en ligne (pages de manuel, HTML, texinfo) et imprimée (Bible Linux, etc.).

La procédure d'installation de Red Hat est très au point et nécessite peu d'interaction de la part de l'utilisateur, en dehors des choix courants (les noms de machines, par exemple). Une fois l'installation terminée, le système est prêt à tourner.

L'installation de XWindow n'est pas obligatoire pour le serveur, mais cela rend les accès locaux et les tests plus faciles. La procédure d'installation de XWindow peut être conduite par différents programmes ; XF86Setup offre le plus de facilité d'auto-test, et demande peu de connaissance des menus détails (la programmation de l'horloge vidéo, etc.). La seule contrainte est que le logiciel puisse détecter l'adaptateur vidéo. Des cartes graphiques accélératrices bon marché (comme les cartes basées sur le chip Trio S64, avant le S64UV+) fonctionnent sans aucun problème.

A partir de maintenant, nous supposons que le système tourne, et que Apache, Perl et XWindow ont été installés avec succès. Nous supposons de même que les fichiers et structure de répertoires sont tels que définis dans l'installation. Enfin, nous laissons le nom de la machine tel quel, et pour le moment, supposons que c'est `localhost`. Nous utiliserons ce nom pour tous les tests d'installation ; dès que le système fonctionnera, le véritable nom pourra être ajouté. Notez que l'installation réseau suppose d'éditer le fichier `/etc/hosts`, entre autres. Cela peut être pris en charge par les outils d'administration fournis à l'utilisateur root.

3.4 Le serveur HTTP

Le serveur HTTP fourni avec Linux est Apache, `httpd` pour le système. La page de manuel (`man httpd`) explique comment installer et démarrer le démon http (donc http*d*) mais, comme il a été indiqué plus haut, si l'installation s'est bien passée, le serveur HTTP doit tourner. Vérifiez l'arborescence des répertoires : le répertoire `/home/httpd` doit exister, avec trois sous-répertoires : `../cgi-bin/`, `../html/` and `../icons/`. Dans `../html/`, vous devez trouver un fichier `index.html`. Plus tard, nous modifierons ou remplacerons ce fichier par notre propre `index.html`. Toute la configuration se fait dans le fichier `/etc/httpd/conf/`. Le système est correctement préconfiguré et ne doit pas être modifié, si l'installation s'est faite sans problème.

3.5 Les navigateurs

Il existe trois types de navigateurs disponibles sous Linux : les logiciels purement textuels, comme Lynx, des logiciels simples et expérimentaux comme Arena (gratuit) et des logiciels commerciaux, comme Netscape (partagiciel !) avec support de Java intégré. Alors que Lynx et Arena sont fournis avec Linux, Netscape doit être récupéré par d'autres sources. Netscape est disponible sous forme de fichier binaire précompilé pour Linux sur architecture ix86 et tourne "tel quel" une fois l'archive décompressée.

3.5.1 Configuration de Lynx

Une fois Lynx démarré, il cherche une 'URL par défaut' qui n'existe pas toujours si le système n'a pas d'accès Internet permanent. Pour changer cette URL par défaut (ainsi que d'autres détails de configuration), l'administrateur doit éditer le fichier `/usr/lib/lynx.cfg`. Ce fichier est gros, environ 57000 octets, et contient des informations quelquefois contradictoires. Il établit son propre répertoire dans `/usr/local/lib`. Au début du fichier figure une ligne commençant par `STARTFILE`. Remplacez cette ligne par la suivante : `STARTFILE:http://localhost`, en vous assurant qu'il n'y a pas d'espace en trop :

```
# STARTFILE:http://www.nyu.edu/pages/wsn/subir/lynx.html
STARTFILE:http://localhost
```

Après avoir enregistré le fichier, Lynx doit maintenant ouvrir notre `index.html` s'il est lancé sans argument.

3.5.2 Configuration d'Arena

S'il est lancé sans argument, Arena recherche son URL par défaut. Cette URL est codée en dur dans l'exécutable, mais peut être redéfinie en utilisant la variable d'environnement `WWW_HOME`. L'administrateur système peut placer la ligne suivante dans le fichier `/etc/profile` : `WWW_HOME="http://localhost"`. Cette variable doit être exportée, soit par l'ajout de la ligne adéquate (`export WWW_HOME`), soit en ajoutant `WWW_HOME` à la ligne d'export courante :

```
WWW_HOME="http://localhost"
export WWW_HOME
```

A la prochaine connexion, la nouvelle URL par défaut d'Arena sera connue du système.

3.5.3 Installation et Configuration de Netscape

Netscape était un produit commercial, et n'est donc pas inclus dans les anciennes distributions Linux. Néanmoins, il est téléchargeable par Internet, ou accessible à partir de certaines compilations de logiciels sur CD-ROM. Netscape est fourni sous la forme de fichiers binaires précompilés pour les plates-formes les plus courantes. Avant de l'installer, il est utile de créer le répertoire `/usr/local/Netscape`, dans lequel l'archive sera décompactée. Tous les fichiers doivent rester à cette place (sauf la bibliothèque Java : suivez les instructions du fichier `README` fourni avec les binaires de Netscape), et il suffira de créer un lien symbolique avec `/usr/local/bin` par la commande :

```
# ln -s /usr/local/Netscape/netscape .
```

depuis le répertoire `/usr/local/bin/`.

Netscape est maintenant pret à être utilisé, et peut être configuré par le menu "Options". Dans "General Preferences", il y a un onglet intitulé "Appearance", avec un champ de saisie "Home Page Location". Tapez `http://localhost`, et n'oubliez pas de sauvegarder les options (par le menu "Options" – "Save Options") avant de quitter Netscape. Au prochain démarrage, Netscape présentera la page d'accueil d'Apache.

3.6 Les navigateurs avec Apache

Faites maintenant le premier test d'Apache avec un navigateur : lancez simplement l'un des navigateurs disponibles, et il affichera la page d'accueil `Apache : Red Hat Linux Web Server`. Cette page indique la localisation des fichiers et d'autres informations concernant l'installation du serveur http. Si cette page ne s'affiche pas, vérifiez que les fichiers cités plus haut sont bien en place et que la configuration du navigateur est correcte. Fermez les fichiers de configuration avant de démarrer de nouveau le navigateur. Si tous les fichiers sont installés et que le navigateur semble correctement configuré, examinez la configuration du réseau. Ou bien le nom de votre machine est différent de celui spécifié lors de la configuration, ou bien la configuration réseau n'est pas correcte. Il est particulièrement important que `/etc/hosts` contienne au moins la ligne suivante :

```
127.0.0.1                    localhost localhost.localdomain
```

qui suppose que vous pouvez vous connecter localement. Vous pouvez le vérifier en lançant une commande réseau qui réclame un nom de machine comme argument, comme `telnet localhost` (en supposant que `telnet` soit installé). Si cela ne fonctionne pas, la configuration réseau doit être controlée avant de continuer l'intallation.

3.7 La Base de données et son installation

L'installation de la base de données demande à peine plus de préparation que les étapes précédentes. Il y a quelques moteurs de base de données SQL disponibles, avec des contraintes d'administration et d'exécution différentes ; l'un des plus simples est msql, dit aussi "Mini-SQL", écrit par David Hughes. Msql est un "partagiciel". En fonction de la version utilisée, les sites commerciaux sont redevables de 250 US$, voire plus, les utilisateurs privés d'au moins 65 US$, et seules les institutions scolaires et les entreprises à but non lucratif peuvent utiliser ce logiciel librement. Les termes exacts des droits figurent dans la documentation de la base de données. Les éléments données ici ne sont qu'indicatifs.

Tout d'abord, voici en quelques mots pourquoi l'auteur a choisi msql. Il y a tout d'abord une expérience personnelle. Alors qu'il cherchait une base de données, l'auteur a trouvé que msql était la plus facile à installer et à maintenir, et qu'elle couvrait un ensemble suffisamment large du langage SQL pour convenir à une utilisation classique. C'est seulement en écrivant ces lignes que l'auteur a découvert cette éloge dans la DBI FAQ d'Alligator Descartes (la FAQ interface perl aux bases de données) :

```
Du point de vue de l'auteur, si le volume de données est relativement faible, les tables
contenant moins d'1 million de lignes, avec moins de 1000 tables dans une base donnée,
alors msql constitue une solution parfaitement acceptable.  Cette base de données est
très bon marché, extraordinairement solide, et offre un excellent support, ...
```

Mqsl est disponible en deux versions, msql-1.0.16 et msql-2.0.1, qui diffèrent par leurs performances (cela n'est sensible que sur des petits projets) et les logiciels les accompagnant (la version la plus récente dispose de plus d'outils, de son propre langage de script, etc.). Nous décrirons les deux versions de msql, car leurs installations se distinguent par quelques aspects.

3.7.1 Installation de msql-1.0.16

Msql est disponible sous forme de sources et de binaires précompilés au format ELF. L'utilisation des binaires ELF rend l'installation plus simple, car l'archive `msql-1.0.16.ELF.tgz` contient une copie de l'arborescence d'installation, pour que les répertoires soient générés correctement lors du décompactage dans le répertoire `/`.

Si vous décidez de compiler msql-1.0.16 vous-même, et que vous voulez utiliser le paquetage MsqlPerl plutot que l'interface DBI (voir plus loin une présentation détaillée des différence entre les deux stratégies), alors attendez-vous à ce que MsqlPerl rapporte, lors des tests d'installation, des erreurs dans msql. Dans ce cas, une correction sera nécessaire, expliquée dans la documentation MsqlPerl (fichier `patch.lost.tables`). En l'occurrence, il faut inclure les trois lignes suivantes dans `msqldb.c`, après la ligne 1400, contenant `entry->def = NULL;` :

```
*(entry->DB) = 0;
*entry->table) = 0;
entry->age = 0;
```

Cette partie de code doit maintenant être la suivante :

```
freeTableDef(entry->def);
safeFree(entry->rowBuf);
safeFree(entry->keyBuf);
entry->def = NULL;
*(entry->DB) = 0;
*entry->table) = 0;
entry->age = 0;
```

La compilation de msql comprend plusieurs étapes. Après avoir décompacté l'archive contenant les sources, il faut créer un répertoire destination. Cela se fait avec la commande suivante :

```
# make target
```

Si tout se passe bien, le système répond avec

```
Build of target directory for Linux-2.0.30-i486 complete
```

Vous pouvez maintenant aller dans le répertoire que vous venez de créer, et taper d'abord la commande

```
# ./setup
```

La séquence `./` est nécessaire pour s'assurer que la commande `setup` exécutée est bien celle du répertoire courant, et non une autre qui aurait le même nom. On va maintenant vous poser quelques questions concernant le répertoire source, et la localisation du répertoire d'installation. Une fois que ces questions ont eu leur réponse, le système lance quelques tests pour vérifier que les logiciels nécessaires (compilateurs, utilitaires divers, etc.) sont présents, puis finalement répond

```
Ready to build mSQL. You may wish to check "common/site.h" although the defaults should
be fine.  When you're ready, type "make all" to build the software
```

Il faut alors taper

```
# make all
```

Si tout fonctionne comme prévu, nous devons alors lire :

```
make[2] : leaving directory '/usr/local/Minerva/src/msql' <- [msql] done Make of mSQL
complete. You should now install mSQL using make install

NOTE : mSQL cannot be used free of charge at commercial sites. Please read the
doc/License file to see what you have to do.

make[1] : Leaving directory '/usr/local/Minerva/src'
```

Tous les binaires doivent être accessibles, par exemple en créant des liens symboliques dans `/usr/local/bin/`. Déplacez-vous dans ce répertoire et tapez la commande

```
# ln -s /usr/local/Minerva/bin/* .
```

après quoi les liens sont correctement construits.

3.7.2 Test de msql-1

Après l'installation, il est maintenant possible de tester le fonctionnement de la base de données. Avant toutes choses, le serveur doit être démarré. L'administrateur système, grâce aux privilèges du compte utilisateur `root`, lance la commande

```
# msqld &
```

(n'oubliez pas d'ajouter le `&`, sinon msql ne tournerait pas en tâche de fond.), après quoi le message suivant doit apparaître :

```
mSql Server 1.0.16 starting ...

Warning : Couldn't open ACL file : No such file or directory
Without an ACL file global access is Read/Write
```

Ce message indique que tout fonctionne correctement, à part la configuration des droits d'accès. Pour le moment, il suffit de démarrer le serveur msql à partir d'un shell, mais vous pourrez par la suite vouloir le démarrer automatiquement au lancement du système. Cette commande doit alors être ajoutée dans un des scripts du répertoire `rc.d`. Seul l'administrateur système peut lancer la première commande typique d'une base de données (création de la première table) :

```
# msqladmimn create inventur
```

msql répond alors `Database "inventur" created`. Comme preuve supplémentaire, vous pouvez constater que le répertoire `/usr/local/Minerva/msqldb/` contient maintenant le répertoire `../inventur`, vide pour l'intant. Vous pouvez manipuler la nouvelle base avec les outils d'administration, qui sont décrits en détail dans la documentation msql.

3.7.3 Installation de msql-2.0.1

Une nouvelle version, plus puissante, du serveur mSQL d'Hugues est maintenant disponible. Son installation est un peu différente. Installer msql-2 de zéro nécessite les étapes suivantes. Copiez l'archive à l'endroit où vous souhaitez l'installer, par exemple `/usr/local/msql-2/`, puis décompressez-la :

```
# tar xfvz msql-2.0.1.tar.gz
```

Positionnez vous à la racine de l'arborescence d'installation et tapez

```
# make target
```

Placez vous dans le répertoire `targets` et vérifiez le type de votre machine. Il devrait y avoir un nouveau sous-répertoire `Linux-`*(votre version)*`-`*votre cpu)*. Allez dans ce répertoire et lancez l'utilitaire de configuration qui s'y trouve :

```
# ./setup
```

Il y a aussi un fichier `site.mm` qui peut être édité. Peut-être avez-vous déjà utilisé le répertoire `/usr/local/Minerva/` et souhaitez-vous le conserver intact ? Dans ce cas, changez la ligne `INST_DIR=...` pour indiquer le répertoire destination qui vous convient. Sinon, ne changez rien.

Maintenant, vous pouvez contruire la base de données :

```
# make
# make install
```

Si tout marche bien, vous verrez ce message :

```
[...]

Installation of mSQL-2 complete.

*********
**  This is the commercial, production release of mSQL-2.0
**  Please see the README file in the top directory of the
**  distribution for license information.
*********
```

Une fois que tout est installé correctement, vous devez vous préoccuper de certains détails d'administration. C'est ici que les différences avec msql-1 commencent. D'abord, un utilisateur `msql` est créé, et est responsable de l'administration de la base de données.

```
# adduser msql
```

Maintenant, vous devez changer les propriétaire et groupe de tous les fichiers dans le répertoire de mSQL en tapant :

```
# cd /usr/local/Minerva
# chown -R msql:msql *
```

Enfin, vous pouvez créer les liens symboliques pour tous les éxecutables de la base de données dans `/usr/local/bin/` en lançant la commande :

```
# ln -s /usr/local/Minerva/bin/* .
```

3.7.4 Test de msql-2

Démarrez maintenant le serveur de la base en tapant la commande `msql2d &`, vous devriez obtenir cette réponse :

```
Mini SQL Version 2.0.1
Copyright (c) 1993-4 David J. Hugues
Copyright (c) 1995-7 Hughes Technologies Pty. Ltd.
All rights reserved.

     Loading configuration from '/usr/local/Minerva/msql.conf'.
     Server process reconfigured to accept 214 connections.
     Server running as user 'msql'.
     Server mode is Read/Write.

Warning : no ACL file. Using global read/write access.
```

Tout est parfait. La base est compilée et installée, et nous pouvons maintenant continuer avec les modules perl puisqu'ils demandent la présence d'un serveur de base de données opérationnel pour les tests.

Au fait, ce moment est bien choisi pour imprimer la documentation complète livrée avec msql-2.0.1 :

```
# gzip -d manual.ps
# lpr manual.ps
```

Nous pouvons maintenant poursuivre la mise en place des interfaces, mais il est judicieux de laisser le nouveau serveur SQL tourner: cela ne fera que faciliter les tests des bibliothèques d'interface.

3.8 Les différentes Interfaces : DBI/mSQL, MsqlPerl, et Lite

Une phrase fréquemment citée dans le Camel Book (la documentation de référence de perl) affirme qu'il y a toujours plusieurs manières d'obtenir un résultat avec perl. Hélas, cela est vrai aussi avec notre application. Il y a trois méthodes pour accéder à une base de données msql par l'intermédiaire de CGI. Tout d'abord, la question est de savoir s'il faut ou non utiliser perl. Dans le premier cas (supposé dans ce document), il y a encore le choix entre deux types complètement différents d'interface. Si nous n'employons pas perl, il reste la solution d'employer le langage de script propre à msql, appelé Lite, qui est relativement proche du langage C, en plus simple.

3.8.1 DBI et DBD-mSQL

Au moment de la rédaction de ce document, c'est l'utilisation de l'interface générique de base de données appelée DBI qui est préférée. DBI a quelques avantages. Elle fournit un contrôle d'accès standard à de nombreuses base de données commerciales, et ce avec le même ensemble de commandes. La base de données en fonctionnement sur un système donné est alors interrogée par une interface qui masque efficacement les caractéristiques spécifiques de cette base au programmeur. Ainsi, DBI fournit une passerelle pratique pour travailler avec différentes bases de différents auteurs. Avec un seul script, il est possible de communiquer avec plusieurs bases de données différentes. Le lecteur interessé peut consulter la DBI-FAQ pour plus de détails. Il y a cependant un inconvénient : l'interface DBI est en cours de développement et aligne les versions à une allure galopante (quelques fois avec plusieurs mises à jour par mois). De même, les pilotes de bases de données sont fréquemment mis à jour, et peuvent être basés sur des versions spécifiques de l'interface de base de données. Les utilisateurs faisant une première installation doivent se limiter aux numéros de version donnés dans ce document, car d'autres versions peuvent poser des problèmes de compilation et de test, dont la résolution n'est pas une affaire de néophyte.

3.8.2 MsqlPerl

MsqlPerl est une bibliothèque permettant l'accès direct à msql à partir de programmes écrits en perl. Elle n'utilise pas l'interface DBI et est très compacte. Bien qu'elle fonctionne très bien avec les deux versions de msql, son usage n'est pas conseillé par rapport à l'interface DBI, qui tend à se généraliser. Néanmoins, suivant le système, c'est une alternative intéressante, car la bibliothèque est petite et facile à installer. Notamment, il y a moins de dépendance par rapport aux numéros de version que celles constatées entre DBI et les pilotes de base de données.

3.8.3 Le langage de commande propre de msql : Lite

Enfin, msql-2 fournit son propre langage de commande : Lite. Ce langage est proche du C, assaini et complété avec quelques fonctionnalités du type 'shell' (d'une certaine manière, c'est une version spécialisée de perl). Lite est un langage simple et bien documenté dans le manuel msql-2. Le paquetage msql-2 livre aussi en exemple un application utilisant Lite.

Nous ne décrirons pas ici Lite, car il est trop spécifique de msql-2 (et déjà documenté !), et parce que le lecteur est censé avoir un certain intérêt pour perl, et en avoir de bonnes notions. Néanmoins, il est recommandé d'y jeter un coup d'oeil : il peut s'avérer la solution idéale dans un environnement ne mettant en oeuvre que msql-2 (en supposant donc qu'aucune autre base n'est utilisée), grâce à sa simplicité.

3.9 La solution standard : DBI et DBD-msql

Nous supposons que perl a été installé pendant la configuration du système, ou en utilisant le gestionnaire de paquetage mentionné plus haut. Aucun détail supplémentaire ne sera donné ici. Néanmoins, nous allons d'abord tester si notre version de perl est récente :

```
# perl -v
```

perl doit répondre avec le message suivant :

```
This is perl, version 5.003 with EMBED
        Locally applied patches:
          SUIDBUF - Buffer overflow fixes for suidperl security

        built under linux at Apr 22 1997 10:04:46
        + two suidperl security patches

Copyright 1987-1996, Larry Wall
[...]
```

Jusque là, tout va bien. L'étape suivante consiste à installer les bibliothèques générales perl pour les bases de données (DBI), le pilote msql (DBD-mSQL) et CGI. Le pilote CGI est nécessaire dans tous les cas. Les archives suivantes sont utilisées :

1. DBI-0.8.1.tar.gz
2. DBD-mSQL-0.65.tar.gz
3. CGI.pm-2.31.tar.gz (ou plus récent>

Une précision est nécessaire ici pour les débutants : le test décrit ici fonctionne très bien à condition d'utiliser *exactement* les versions recommandées de logiciels ; des combinaisons d'autres versions peuvent échouer à un moment ou à un autre. Le déboguage de combinaisons de versions incompatibles est une affaire de spécialistes des interfaces d'appel. Quelquefois, seule une méthode change de nom alors qu'elle fait la même chose, mais d'autres fois, les structures internes changent de manière significative. Donc, encore une fois, tenez-vous en aux numéros indiqués de versions si vous ne voulez pas de problème, et ce même si vous constatez que les versions ont encore changé dans l'intervalle. Il est normal de voir les versions évoluer rapidement, et vous devez vous attendre à des problèmes en installant d'autres versions que celles conseillées ici.

Il est très important que le pilote de base pour mSQL (DBD-mSQL) soit installé *après* l'interface générique DBI.

Nous commençons par créer le répertoire */usr/local/PerlModules* car il est important de conserver l'arborescence initiale de perl intacte. Nous pourrions aussi choisir un autre nom de répertoire ; cela n'a strictement aucune importance, malheureusement aucune recommandation n'est faite dans les fichiers README des différents modules perl. Une fois recopiées les archives précédemment citées dans `/usr/local/PerlModules`, nous les décompactons en tapant

```
# tar zxvf [archive-file]
```

pour chacune des trois archives. N'oubliez pas d'indiquer le vrai nom de l'archive à la commande `tar`. Le processus d'installation de ces trois archives est très standard ; seuls les messages de sortie concernant les étapes les plus importantes sont reproduits ici.

3.9.1 Installation de l'interface perl de base de données DBI

L'interface de la base de données doit avoir été installée avant le pilote spécifique à la base. Le décompactage de l'archive DBI crée le répertoire `/usr/local/PerlModules/DBI-0.81/`. Placez-vous dans ce répertoire. Il y a un fichier `README` (que vous devriez lire) et un makefile spécifique pour perl. Maintenant, tapez la commande

```
# perl Makefile.PL
```

Le système doit répondre avec un long message dont la partie la plus importante figure ci-dessous :

```
[...]
MakeMake (v5.34)
Checking if your kit is complete ...
Looks good
     NAME => q[DBI]
     PREREQ_PM => { }
     VERSION_FROM => q[DBI.pm]
     clean => { FILES=>q[$(DISTVNAME) /] }
     dist => { DIST_DEFAULT=>q[clean distneck disttest [...]
Using PERL=/usr/bin/perl

WARNING! By default new modules are installed into your 'site_lib' directories. Since site_lib directories
come after the normal library directories you MUST delete old DBI files and directories from your 'privlib
and 'archlib' directories and their subdirectories.

Writing Makefile for DBI
```

Comme le programme l'indique, tout va bien, et nous pouvons poursuivre avec l'étape suivante :

```
# make
```

Si aucun message d'erreur n'apparaît (les traces détaillées affichées sur l'écran *ne* sont *pas* un message d'erreur), nous pouvons tester la librairie nouvellement installée avec la commande

```
# make test
```

Sur l'affichage, guettez les lignes suivantes (vous pouvez toujours revenir en arrière avec la touche `[Shift]-[PgUp]`) :

```
[...]
t/basics...........ok
t/dbidrv...........ok
t/examp............ok
All tests successful.
[...]
DBI test application $Revision: 1.1.1.1 $
Switch: DBI-0.81 Switch by Tim Bunce, 0.81
Available Drivers: ExampleP, NullP, Sponge
ExampleP: testing 2 sets of 5 connections:
Connecting... 1 2 3 4 5
Disconnecting...
Connecting... 1 2 3 4 5
Disconnecting...
Made 10 connections in 0 seconds ( 0.00 usr  0.00 sys = 0.00 cpu)

test.pl done
```

La dernière étape est l'installation de tous les fichiers dans leurs répertoires respectifs. La commande suivante s'en occupe :

```
# make install
```

Il n'y a plus rien à faire. Si pour quelque raison que se soit, l'installation échoue et que vous vouliez la recommencer, n'oubliez pas de taper d'abord la commande

```
# make realclean
```

Cela supprimera toutes les traces laissées par la précédente installation. Vous pouvez aussi supprimer les fichiers installés en copiant le contenu de l'écran (montré ici abrégé)

```
Installing /usr/lib/perl5/site_perl/i386-linux/./auto/DBI/DBIXS.h
Installing /usr/lib/perl5/site_perl/i386-linux/./auto/DBI/DBI.so
Installing /usr/lib/perl5/site_perl/i386-linux/./auto/DBI/DBI.bs
[...]
Writing /usr/lib/perl5/site_perl/i386-linux/auto/DBI/.packlist
Appending installation info to /usr/lib/perl5/i386-linux/5.003/perllocal.po
```

dans un fichier, en remplaçant `Installing` par `rm`. Si vous avez appellé ce fichier `uninstall` vous pouvez alors taper

```
# . uninstall
```

ce qui effacera les derniers fichiers installés.

3.9.2 Le pilote msql de perl : DBD-mSQL

Le pilote msql pourra être installé seulement *après*l'installation réussie de l'interface perl générique de base de données.

Les étapes de l'installation sont pratiquement les mêmes que les précédentes, donc commencez par taper

```
# perl Makefile.PL
```

Là, le système doit répondre avec un avertissement vous demandant de lire la documentation accompagnant le logiciel. Ensuite, il va détecter où se trouve msql, et vous demande quelle version vous utilisez :

```
$MSQL_HOME not defined. Searching for mSQL...
Using mSQL in /usr/local/Hughes

-> Which version of mSQL are you using [1/2]?
```

Entrez la version correcte. Quelques lignes de texte suivent. Guettez les suivantes :

```
Splendid! Your mSQL daemon is running. We can auto-detect your configuration.

I've auto-detected your configuration to be running on port: 1114
```

Vous pouvez maintenant tester le pilote en tapant

```
# make test
```

Encore une fois, plusieurs lignes sont affichées. Si elles se terminent par

```
Testing: $cursor->func( '_ListSelectedFields' )/ This will fail.
     ok: not a SELECT in msqlListSelectedFields!
Re-testing: $dbh->do( 'DROP TABLE testaa' )
     ok
*** Testing of DBD::mSQL complete! You appear to be normal! ***
```

tout va bien, et vous pouvez lancer l'installation du pilote en tapant

```
# make install
```

Vous êtes prêt à continuer et pouvez sauter le paragraphe suivant.

3.10 L'interface MsqlPerl

Si vous décidez d'utiliser l'interface globale MsqlPerl, aucun pilote particulier n'est nécessaire ; seule l'archive `MsqlPerl-1.15.tar.gz` est utilisée, puisque, comme cela a déjà été dit, MsqlPerl fournit une interface directe entre perl et le serveur de base de données, sans utiliser l'interface DBI. L'installation et le test sont très faciles.

Après avoir tapé `perl Makefile.PL`, l'utilitaire make peut être activé. Vous devez d'abord indiquer où se trouve mSQL. S'il est dans `/usr/local/Minerva/`, la réponse par défaut peut être validée.

Ensuite, tapez `make test`. Avant cela, vous devez vous assurer qu'il y a bien une base nommée `test` et que vous avez les droits d'écriture et lecture dessus. Cela peut être fait avec

```
# msqladmin create test
```

3.11 Bibliothèque CGI de perl

L'installation de l'interface CGI de perl est la plus simple des trois étapes. Lancez les commandes dans l'ordre donné, et voilà :

```
# perl Makefile.PL
# make
# make install
```

Contrairement aux autres pilotes, cette interface n'a pas d'option de test (`# make test`), alors que les autres modules *doivent* être testés dans tous les cas.

Un sous-répertoire avec les exemples CGI est créé. Vous pouvez en copier le contenu vers `/home/httpd/cgi-bin/` et utiliser un navigateur pour jouer avec les scripts.

3.12 Check-list de l'installation

Nous avons effectué les étapes suivantes, dans cet ordre:

1. Installation de Linux avec support réseau
2. Installation d'un serveur http, par exemple Apache
3. Installation d'un navigateur, par exemple Arena, Lynx ou Netscape
4. Installation d'un serveur SQL, par exemple msql
5. Installation d'une interface perl SQL convenable
6. Installation des fichiers CGI

A la fin, vous devez faire un peu de ménage. Toutes les arborescences des sources msql et les modules perl peuvent être détruites sans inconvénient (cependant, vous ne devriez pas détruire les fichiers archives !) puisque les binaires et la documentation sont maintenant dans des répertoires différents.

4 Un exemple de Base de Données

Après avoir terminé la procédure d'installation, nous pouvons maintenant lancer l'application donnée en exemple. En fonction de la version de msql installée et de l'interface perl utilisée, nous devrons modifier un peu ce programme.

Tout d'abord, le fichier `index.html`, dans le répertoire `/home/httpd/html/` doit être modifié pour appeler l'application exemple. Nous pouvons mettre notre base (que nous pouvons appeler `database.cgi` ou `inventur.cgi`) dans `/home/httpd/html/test`.

Nous ajoutons une ligne parmi les suivantes dans `index.html` (à choisir, bien sûr, en fonction des choix d'installation) :

```
<LI>Test the <A HREF="test/database.cgi">Database, DBI:DBD-mSQL style!</A>
<LI>Test the <A HREF="test/inventur.cgi">Database, MsqlPerl style!</A>
```

Vous ne devez en principe choisir qu'une seule des deux lignes précédentes, mais vous pouvez, si vous avez installé les deux types d'interface, laisser les deux lignes telles quelles. Vous pourrez alors comparer les performances.

4.1 Adaptation de l'exemple de script pour MsqlPerl

Il est nécessaire d'indiquer, dans notre exemple de script, qu'il faut utiliser l'interface MsqlPerl. La modification doit être faite à plusieurs endroits. D'abord, au début du fichier, il faut changer la clause `use` :

```
# use DBI;                # Generisches Datebank-Interface
use Msql;
```

Ensuite, à la ligne 27, MsqlPerl n'exige pas la mention d'un pilote particulier :

```
# $dbh = DBI->connect($host, $database, '', $driver) ||
$dbh = Msql->connect($host, $database) ||
```

A partir de la ligne 33 et pour tout le script, changez toutes les occurrences de `do` par `query` :

```
# $dbh->do("SELECT * FROM hw") || db_init($dbh);
$dbh->query("SELECT * FROM hw") || db_init($dbh);
```

Enfin, dans le laius MsqlPerl, la ligne 207 peut être mise en commentaire :

```
# $sth->execute || msg("SQL Error: $sth->errstr);
```

De plus, il peut être nécessaire de remplacer tout les appels `errstr` tels que celui de la ligne précédente par `errmsg`. Cela dépend de la version utilisée.

Après ces modifications, le script doit tourner correctement.

4.2 Adaptation de l'exemple pour msql-2

La syntaxe SQL a subi des changements durant le développement de msql-2. Le script original n'exécutera pas les instructions d'initialisation de la table, aux lignes 45 – 48. Le modificateur `primary key` n'est plus compris par msql-2, et doit être supprimé :

```
  $dbh->do(<<EOT) || die $dbh->errstr;   # Neue Personen-Tabelle
      create table person (
# We do not need the 'primary key' modifier anymore in msql-2!
#           pn    int primary key,    # Personalnummer
            pn    int,                # Personalnummer
            name  char(80),           # Nachname, Vorname
            raum  int                 # Raumnummer
          )
EOT
  $dbh->do(<<EOT) || die $dbh->errstr;   # Neue Hardware-Tabelle
      create table hw (
# We do not need the 'primary key' modifier anymore in msql-2!
#           asset   int primary key,    # Inventurnummer
            asset   int,                # Inventurnummer
```

```
                name    char(80),           # Bezeichnung
                person  int                 # Besitzer
              )
    EOT
```

Malheureusement, ce script particulier acceptera maintenant les enregistrements avec des numéros personnels identiques ; le modificateur msql-1 `primary key` était justement là pour éviter cela. La documentation msql2 indique comment utiliser la clause `CREATE INDEX` pour créer des entrées uniques.

5 Conclusion

Si vous avez installé msql-2 sur votre système, vous pouvez regarder les exemples de programmes écrits avec Lite, le langage de script de msql-2.

Chaque version de msql est livrée avec un minimum d'outils d'administration, qui permettent à l'utilisateur de créer et détruire des tables (`msqladmin`) et d'examiner la structure de la base de données (`relshow`).

Msql deuxième génération (c'est-à-dire msql-2) possède quelques utilitaires de plus : `msqlimport` et `msqlexport`. Ils permettent d'insérer et d'extraire des données de la base SQL, à partir de et vers des fichiers. Ces utilitaires peuvent être utilisés pour, *en une seule passe*, charger ou extraire de grandes quantités de données, et cela sans que l'utilisateur ait à se soucier d'écrire *une seule* ligne de perl, de SQL, ni même de n'importe quoi.

Si vous voulez écrire votre propre script perl de gestion de base de données, vous trouverez suffisamment d'aide dans les fichiers d'exemples, et dans la volumineuse documentation en ligne qui est livrée avec le module DBI.

Dans tous les cas, vous êtes maintenant prêts à publier vos données sur votre réseau, et même sur le Web.