Ecole Nationale Supérieure d'Informatique et de

Mathématiques Appliquées de Grenoble

Institut National Polytechnique de Grenoble

Connexion d'une base de données relationnelle avec un serveur WWW

Jean Marc NEYRET-GIGOT

Année Spéciale Informatique

Juin 1998

SOMMAIRE

Je tiens à remercier Serge Rouveyrol pour l'aide qu'il m'a apporté pour la réalisation de ce projet, ses conseils, pour le temps qu'il m'a accordé sans compter, ainsi que pour les cafés et les cigarettes partagées…

J'y associerai mes camarades de cette année spéciale informatique, pour les discutions, le soutient, la bonne humeur, … ils m'ont grandement aidé à reprendre les études : Pierre Pujalon, Antoine Viana, Pascal Vergneau, Guillaume Durand, Gilles Lemoing et tous les autres.

I. Introduction

Le Web n'est plus maintenant un terme connu de quelques initiés et passionnés d'informatique. Il n'est pas de jour où, dans une publicité, un journal, à la télévision on ne voit un quelconque « http://… ». On assiste même à une promotion du Web par les pouvoirs politiques, relayée abondamment pas les médias.

S'il est apparu à l'origine pour faciliter la communication entre chercheurs en physique - il a vu le jour en 1989 au CERN de Genève à l'initiative de Tim Berners-Lee - il a depuis largement débordé son domaine. Le Web couvre maintenant pratiquement n'importe quel sujet possible.

Sa facilité d'emploi, même pour les non-spécialistes d'informatique, et sa couverture mondiale explique son succès.

Pourtant derrière les pages colorées, les images animées ou toute autre fantaisie rencontrée en surfant, le Web est avant tout un outil de communication et de transfert d'information. Très vite, se ressent le besoin de faire connecter des données avec ce moyen de communication et de les mettre sur le réseau.

C'est l'objet de ce mémoire : on montrera comment connecter une base de données désormais largement répandue : **MySQL** avec un serveur Web.

Après la création de pages HTML statiques, on s'aperçoit de l'utilité de pouvoir générer des pages dynamiques. Ceci est rendu possible grâce à **la programmation CGI**. Celle-ci rendra possible une interaction entre le client demandeur d'information et le serveur.

Maintenant que nous avons défini nos objectifs, il s'agit de lier les éléments de notre schéma. Heureusement il existe un langage fantastique qui réalisera nos souhaits : **Perl**.

Perl (Practical Extraction and Report Language) est un langage interprété extrèmement puissant, parfaitement adapté à la manipulation de texte et de fichiers. On peut pratiquement tout faire en Perl : gérer des processus, manipuler des répertoire ou fichiers, accéder à des bases de données … Perl possède en outre une série de modules objets adaptés à des besoins particuliers. Parmi ceux-ci CGI.pm et DBI qui permettent respectivement d'accéder à la programmation CGI et à manipuler des bases de données.

II. Le serveur WWW

II.1. Généralités

Le vecteur le plus connu de l'Internet est sans aucun doute le WWW, ou World Wide Web. Celui-ci englobe des modes de communication tels que gopher, telnet ou wais, maintenant en perte de vitesse, ou encore ftp ou les news.

Le protocole de base est fondé sur l'idée du transfert d'hypertextes (Hyper Text Transfert Protocol). Le langage associé à HTTP est HTML (Hyper Text Markup Language). Ce langage à balise a la propriété de surligner certains mots, renvoyant l'image d'un hyperlien – correspondant à une adresse distante, l'URL (Uniform Ressource Location).
La forme générale d'une URL est :

<service>://<user> :<password>@<host> :<port>/<url-path>

HTTP est un protocole client-serveur mais, au contraire de FTP par exemple, il n'y pas de connexion permanente entre les deux parties, de plus, il ne peut traiter qu'un objet à la fois. Par exemple, une image gif insérée dans une page sera chagée séparément de la page HTML elle-même. Chaque requête entre client et serveur est donc traitée de manière indépendante.

Comme protocole client-serveur HTTP peut, en plus de son rôle de transfert de document, envoyer le résultat d'un programme (soit après traitement des données du client, soit le résultat d'un programme) vers le serveur.

Les programmes qui font le lien entre le serveur et les applications sont appelées Gateway puisqu'ils jouent le rôle de passerelles. La **CGI** (Common Gateway Interface) correspond aux spécifications qui définissent les relations entre HTTP et les autres ressources.

La communication en mode non-connecté présente pourtant un inconvénient : le serveur sans mémoire ne peut rien conserver de ce qui vient d'être échangé. Le protocole sans état n'a par conséquent aucune mémoire des connexions précédentes. Par exemple une page retournée à la suite d'une réponse à un formulaire ne sait rien des valeurs saisies.
Pour identifier le client – ou seulement pour savoir s'il a le droit d'accéder aux données – on peut glisser dans les pages HTML des variables HIDDEN, invisibles du client. Pour conserver la mémoire de ce qui s'est passé, pendant la durée d'une connexion, on stocke par exemple l'identité d'un utilisateur dans un fichier. Ces information seront ensuite récupérées au gré des besoins.
On peut également stocker l'information du coté client par l'intermédiaire des cookies. Le serveur peut alors interroger le fichier cookies.txt qui existe sur la plupart des navigateurs modernes pour obtenir des informations.

Il existe plusieurs méthodes pour que le client reçoive les données du serveur.

❖ Méthode GET

Les données sont renvoyées dans l'URL. Elles ont passées par ce qui s'''appèle une *query-string* :

```
http://www.le_serveur.com/cgi-
bin/programme ?donnée1+donnée2
```

❖ Méthode POST

Les données sont envoyées dans le corps du message. Il n'y a pas de *query-sting*, on rajoute un champ *Content-Type* et *Content-Length*.
C'est la méthode qui sera employée icu.

❖ Méthode HEAD

Permet de savoir si un document existe, sans avoir à le télécharger entièrement.

II.2. Le serveur httpd et sa configuration

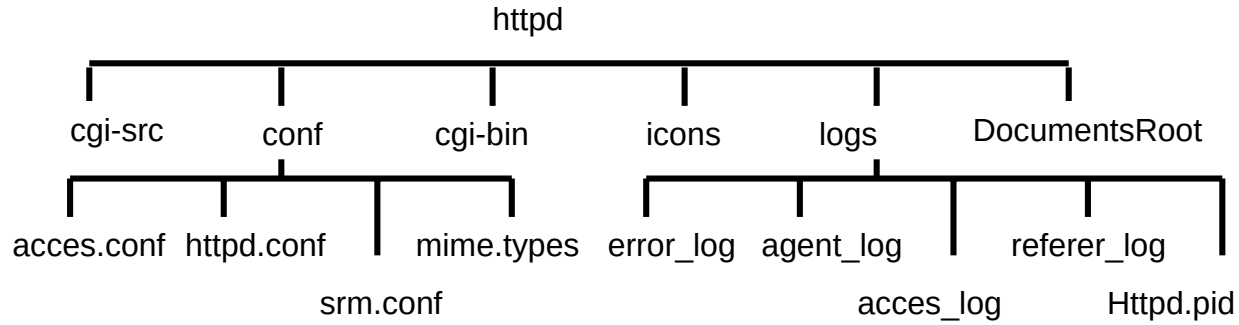
Le serveur (ou démon httpd) est un programme qui écoute les requêtes des clients sur un port de machine (80 par défaut). Il se charge de traiter selon les cas : le transfert des pages HTML au browser, l'exécution des programmes (CGI) sur la machine serveur :

II.2.1. Le serveur httpd

httpd se lance soit en ligne de commande (standalone), soit par inetd.
httpd peut être lancé avec les options :

-d répertoire	équivalent à l'option ServeurRoot de http.conf
-f fichier	Renvoie vers fichier au lieu de httpd.conf
-v	version du httpd

Arborescence des fichiers



II.2.2. Configuration du serveur

Trois fichiers interviennent dans la configuration d'un serveur :

- httpd.conf : fichier qui détermine le comportement du serveur pendant son lancement
- srm.conf : (Server Ressource Management) configure les emplacement des scripts et documents accédés par les clients
- acces.conf : sécurise les accès sur le serveur. Il détermine les droits d'accès sur les répertoires

Peu de variables d'environnement sont à modifier. Voici pour les fichiers httpd.conf et srm.conf les principales instructions utiles pour la configuration du serveur.

- httpd.conf

AccesConfig *fichier*	Spécifie l'accès au fichier acces.conf en référence absolue ou relative au répertoire ServerRoot (conf/acces.conf)
AgentLog *fichier*	Précise le fichier de traces des accès clients, type du browser, sa version (en standalone) (logs/agent_log)
ErrorLog *fichier*	Fichier de trace des erreurs (logs/error_log)
Group *[nom_groupe]*	Identifie le groupe propriétaire des copies httpd pour les réponses vers les clients (en standalone)
User *[nom_user]*	Idem group
IdentifyCheck *[on|off]*	Mode inetd détermine si le client est loggé sous son uid
MaxServers *nb*	Nombre de fils max pouvant être lancé pour le compte Group
PidFile *fichier*	Insère pid du serveur dans le fichier spécifié (logs/httpd.pid)
Port *numero*	Port de la connexion (8765)
ResourceConfig *fichier*	Endroit où se trouve le fichier srm.conf
ServerAdmin *email*	E-mail de l'administrateur du serveyr
ServerName *nom_serveur*	Nom de la machine ou alias de celle-ci
ServerRoot *répertoire*	Définit le chemin absolu du serveur de la racine du serveur httpd (/users1/jmneyret/sSs.mysql/serveur.www)
ServerType *[inetd|stand…]*	Spécifie le mode du serveur soit lancé à la main (standalone) soit par le démon inetd
StartsServer *nb*	Nombre de process qu'on peut lancer en concurence
TimeOut *secondes*	Temporisation pour les requêtes du client
TransfertLog *fichier*	Spécifie l'emplacement du fichier de trace sur les requêtes (logs/acces_log)

type de lancement inetd ou standalone

```
ServerType standalone

#le port
Port 8765

# User/Group: The name (or #number) of the user/group to run httpd as.
#  On SCO (ODT 3) use User nouser and Group nogroup
User jmneyret
Group speinf

#l'identification de l'administrateur
ServerAdmin jmneyret@ensisun.imag.fr

#le répertoire racine du serveur
# ServerRoot: The directory the server's config, error, and log files
# are kept in
ServerRoot /users1/jmneyret/sSs.mysql/serveur.www

#les traces
ErrorLog logs/error_log

# TransferLog: The location of the transfer log file. If this does not
# start with /, ServerRoot is prepended to it.
TransferLog logs/access_log

# PidFile: The file the server should log its pid to
PidFile logs/httpd.pid

#le nom du serveur
ServerName ensisun.imag.fr
```

❖ srm.conf

AccesFilename *fichier*	Fichier qui spécifie les permissions d'accès pour le répertoire ou *fichier* est placé (.htacces)
AddDescription *texte fichierID*	Description texte qui décrit un type de fichier défini par une extension, un nom de fichier, une référence absolue ou un fichier utilisant '*'
AddEncoding *type ext*	Action à associer, type, pour une extension définie
AddIcon *chemin nom1 [nom2..]*	Icône à utiliser sur un type de fichier, pour une session FTP
AddIconbyEncoding *chemin nom1 [nom2..]*	Idem AddIcon sauf que l'information encodée détermine l'icône à utiliser
AddIconType chemin nom1 [nom2..]	Idem AddIcon sauf que que le type MIME détermine l'icône à utiliser
AddType *type ext*	Remplace les définitions MIME spécifiées par les extensions ext trouvées dans le fichier mime.types
Alias *nom chemin*	Définir des alias pour des répertoires
DefaultType *type*	Type MIME par défaut (text/html)
DefaultIcon *chemin*	Icône par défaut
DirectoryIndex *fichier*	Fichier d'accès par défaut du site (index.html)
DocumentRoot *chemin*	Référence absolue depuis lequel httpd retrouve les documents (/home/ann1/jmneyret/sSs.mysql/serveur.www/DocumentRoot)
FancyIndexing *[on|off]*	Ajoute les icones, les fichiers, entêtes et pieds aux liste des fichiers automatiquement indexés (on)
HeaderName *fichier*	Nom du fichier utilisé pour l'entête de la liste des fichiers automatiquement indexés (HEAER)
IndexIgnore *type1, type2*	Type des fichiers ignoré lors de l'indexation des fichiers (*/. ? ?**~*#*/HEADER */README)
IndexOption *opt1, opt2*	Spécifie les paramètres d'indexation comme FancyIndexing, IconAreLink, SupressLastModfied
OldScriptAlias *nom chemin*	Idem Alias
ReadMeName *fichier*	(README)
Redirect *chemin URL*	Pour un site changeant d'adresse permet de le préciser aux clients
ScriptAlias *nom chemin*	Similaire à l'option Alias mais pour les scripts
UserDir *[répertoire|DISABLED]*	Répertoire d'accès pour les comptes utilisateurs (publc_html) par ex ensisun.imag.fr/~jmneyret

#répertoire où sont stockés les documents
DocumentRoot /home/ann1/jmneyret/sSs.mysql/serveur.www/DocumentRoot

UserDir: The name of the directory which is appended onto a user's home
directory if a ~user request is recieved.
UserDir public_html

DirectoryIndex: Name of the file or files to use as a pre-written HTML

DirectoryIndex index.html

#le répertoire où sont stockés les scripts CGI
ScriptAlias: This controls which directories contain server scripts.
ScriptAlias /cgi-bin/ /users1/jmneyret/sSs.mysql/serveur.www/cgi-bin/

❖ acces.conf

dans acces.conf, on trouve des commandes du type

Option *directives*
AllowOverride *directives*

All	Aucune restriction sur ce répertoire
ExecCGI	Autorisation d'exécuter des CGI dans ce répertoire
FollowSymLinks	S'il existe des liens symboliques alors httpd a le droit de les atteindre
Includes	Fichiers d'inclusion pour le serveur sont disponibles dans ce répertoire (< !--#include= ″fichier ″-->en html)
IncludesNoExec	Inclure les fichiers mais interdire l'option Exec (< !—#exec cmd=″cde″-->)
Indexes	Autorise les utilisateurs à retrouver les index générés pour ce répertoire
None	Aucune autorisation sur ce répertoire
SymLinksifOwnreMatch	Httpd suit les liens symboliques si le fichier ou le répertoire lié appartient au même utilisateur que le lien

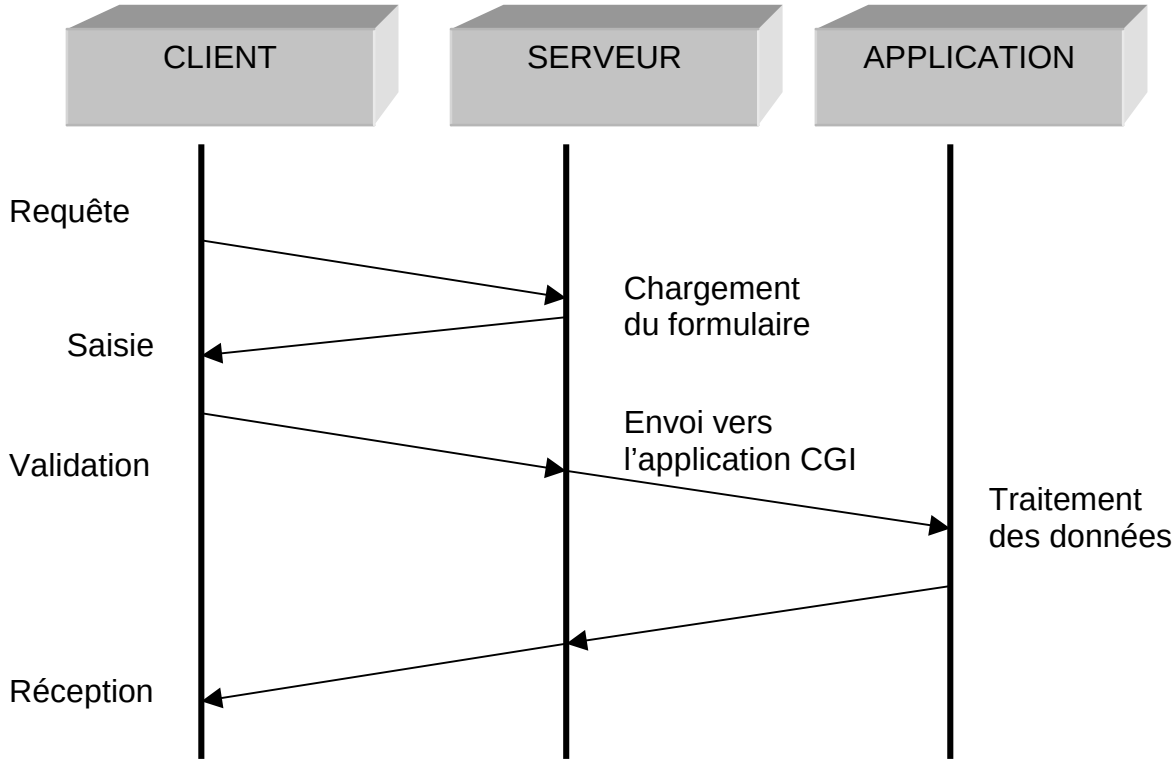
All	Pas de restriction sur les accès des fichiers dans ce répertoire
AuthConfig	Rend disponible les options AuthName, AuthType, AuthUserFile, AuthGroupFile
FileInfo	Rend disponible les options AddType et AddEncoding
Limit	Rend disponible l'utilisation de <Limit>…</Limit>
None	Tous les contrôles d'accès sur les fichiers de ce répertoire ne sont pas autorisés
Options	Rend disponible l'utilisation des directives de l'option Options

AuthName *nom*	Nom qui s'affichera sur la boîte de dialogue lors de l'entrée du mot de passe
AuthType *type*	Type d'autorisation sur ce répertoire (seul Basic est implémenté)
AuthUserType *fichier*	Fichier utilisé pour les utilisateurs avec leurs mots de passe par ex /users1/jmneyret/sSs.mysql/serveur.www/conf/.htpasswd
AuthGroupFile *fichier*	Liste des groupes utilisateurs pour l'authentification des utilisateurs

II.2. La programmation CGI

Lorsque un client clique sur un lien, l'adresse correspondante – l'URL – pointe sur un serveur distant et un service associé. Ce service, ou ressource peut être une simple page HTML ou un programme CGI. La figure suivante montre les relations entre un navigateur Web, un serveur et une application CGI.

A une requête du client, le serveur envoie à celui-ci la ressource demandée (dans notre exemple un formulaire) pour que le navigateur du client puisse l'afficher. Après la saisie du formulaire, et son transfert vers le serveur, le formulaire est envoyé vers l'application CGI.



Quand le formulaire arrive chez le client, chaque champ du formulaire a un nom particulier. Ce sont les variables associées aux informations que le client a saisies lorsqu'il a rempli le formulaire. Le formulaire lui même contient le nom du programme CGI (et son adresse) qui va traiter ce qui a été rentré par le client (balise HTML <FORM>). Quand le client appuie sur « submit », son navigateur rajoute à l'URL du programme CGI les informations qu'il vient de saisir. Ces informations sont ce qu'on appelle une chaîne de requête (query string), constituées de paires ″ nom=valeur″, où le nom correspond au nom du champ rempli et la valeur, les informations correspondantes.
En résumé, le navigateur va transmettre au serveur une URL ressemblant à :

http://www.le_serveur.com/cgibin/le_programme_cgi ?nom1=info1&nom2=info2

Dans cet exemple, il y a deux couples ″nom=valeur″ . Ils sont annoncées par ″?″ et séparés par ″&″

Le serveur, dès qu'il a reçu l'URL du navigateur, appelle le programme CGI en lui passant en paramètre les paires ″nom=valeur″ . Le programme effectue les tâches qui est sensé faire puis renvoie au serveur du code HTML qui est envoyé au client.

Les programmes CGI fournissent autre chose que la passage de paramètres ; ils peuvent être utilisés pour tout ce que peut gérer le navigateur (types MIME).
Ils ont l'avantage d'être programmable par n'importe quel type de langage (C, Shell, Perl).

❖ Les variables d'environnement

Les variables d'environnement contiennent toutes les informations en rapport avec les chemins d'accès, le client, le serveur et les données transmises.

GATEWAY_INTERFACE	Version CGI du serveur
SERVER_NAME	Adresse IP ou nom du serveur
SERVER_SOFWARE	Version du logiciel serveur
SERVER_PROTOCOL	Nom et version du protocole de requête
SERVER_PORT	Port où le serveur écoute
REQUEST_METHOD	Méthode de requête
PATH_INFO	Chemin d'accès
PATH_TRANSLATED	La PATH_INFO translatée
SCRIPT_NAME	Chemin du script courant
DOCUMENT_ROOT	Répertoire des documents
QUERY_STRING	Chaîne de requête passée au programme
REMOTE_HOST	Nom de la machine du client
REMOTE_ADR	Adresse de la machine client
AUTH_TYPE	Méthode d'identification utilisée
REMOTE_USER	Nom de l'utilisateur
REMOTE_IDENT	Identité de l'utilisateur
CONTENT_TYPE	Type MIME des données
CONTENT_LENGTH	Taille des données
HTTP_FROM	Adresse électronique du client
HTTP_ACCEPT	Types MIME supportés par le client
HTTP_USER_AGENT	Navigateur du client
HTTP_REFERER	URL du document qui a accédé au programme CGI

III. La base de données : MySQL

MySQL est probablement la base de données disponible en freeware la plus utilisée. Réputé pour sa rapidité MySQL est une base fondée sur le langage SQL. Elle est implémentée sur un mode client-serveur, avec du coté serveur un démon mysqld et du coté client toute une variéré de librairies et de programmes.

MySQL est disponible en version binaire ou compilable à partir des sources, et ce pour différentes plateformes (y compris Linux et Windows). Toutes les versions sont téléchargeables à partir de l'URL :

http://www.mysql.com/download.html

ou des sites miroir associés.

III.1. Installation du serveur MySQL

Après téléchargement, l'installation de la version source se fait classiquement par l'invocation des commandes :

```
# gunzip –9c mysql-VERSION.tar.gz | tar xvf –
```

Pour décompacter la distribution dans le répertoire courant

```
# cd mysql-VERSION
```

Aller dans le répertoire créé

```
# ./configure
```

Configure la version

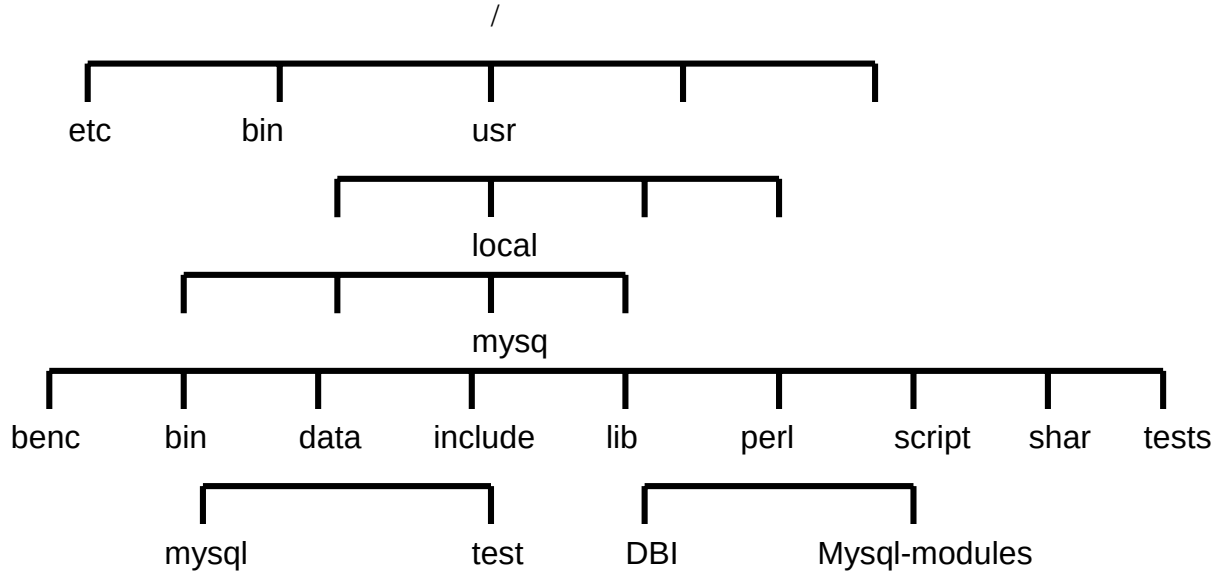
```
# make
```

Compile

```
# make install
```

Installe le tout

Traditionnellement, l'installation est faite dans `/usr/local/mysql`
L'arborescence de la distribution MySQL donne



III.1.1 Installation des modules DB Perl

De la même façon, l'installation des modules Perl : :DBI est réalisée à partir de la racine MySQL par :

```
# cd perl/DBI
# perl Makefile.PL
# make
# make install
# cd ../Mysql-modules
# perl Makefile.PL
# make
# make install
```

Pour installer correctement ces modules et disposer des librairies DBI, il faut – à cette date, juin 98 – avoir une version de Perl d'au moins égale à 5.004.

III.1.2 Lancement du démon

- MySQL sera définitivement en service après avoir lancé le script de démarrage. Celui-ci se commande à partir de la racine mysql :

```
# scripts/mysql_install_db
```

Ce script configure les variables d'environnement MySQL et démarre le démon. Il crée notamment les répertoires mysql et test dans ./data et remplit les tables, après les avoir créées, db, host,user et func. Ces tables interviennent dans la gestion de MySQL, en particulier pour la gestion des utilisateurs et des hosts.

- Le démon proprement dit se lance par la commande

```
# bin/safe_mysqld
```

Le démon peut être lancé par n'importe quel utilisateur de MySQL .

Selon l'endroit où MySQL a été installé, il faudra se placer dans `/usr/local/mysql` pour le lancer ou modifier le script `safe_mysqld` où le démon est placé par défaut dans `/my/gnu`.

III.1.3 Premiers pas avec MySQL

Pour vérifier que MySQL fonctionne correctement, on peut taper :

```
# bin/mysqlshow
```

```
+----------------------+
|      Databases       |
+----------------------+
|      mysql           |
+----------------------+
```

qui donne les bases de données disponibles

```
# bin/mysqlshow mysql
```

```
+----------------------+
|      Tables          |
+----------------------+
|      db              |
|      host            |
|      user            |
+----------------------+
```

qui montre les tables associées à la base mysql

ou encore exécuter une première requête :

```
# bin/mysql –e ″select host, db, user from db″ mysql
```

```
+---------------+-----------------+----------------+
|      host     |      test       |                |
+---------------+-----------------+----------------+
|      %        |      test_%     |                |
+---------------+-----------------+----------------+
```

qui donne les bases de données disponibles

Le démarrage et l'arrêt se font par les commandes suivantes :

```
# bin/mysql.serveur stop
# bin/mysql.serveur start
```

Très certainement l'invocation de ces commandes sera suivie d'un désagréable `acces denied`. Ce refus est provoqué par la table d'allocation des accès à la base. Celle-ci, à ca stade d'installation n'est pas encore configurée. Pour pouvoir accéder à MySQL, il faudra rajouter aux commandes `-u root` pour user=root. L'une des premières choses à faire après l'installation sera de créer des users et de leur associer des mots de passe, y compris pour `root` qui n'a pas encore de mot de passe personnel.

III.1.4 Configuration de la « table des privilèges »

L'une des particularités de MySQL est d'avoir un système original de droits d'accès aux bases de données. Pour MySQL, l'identité d'un utilisateur se fait à partir du couple user+host. Cette particularité fait qu'on peut avoir un utilisateur sur deux hosts différents (avec des privilèges différents) sans problèmes.

Tous les privilèges sont consignés dans les trois tables `user, host` et `db`. On peut tester les privilèges avec le script `mysqlacces`.

Field	Type	Key	Default
Host	char(60)	PRI	""
User	char(16)	PRI	""
Password	char(16)	-	""
Select_priv	enum('N','Y')	-	N
Insert_priv	enum('N','Y')	-	N
Update_priv	enum('N','Y')	-	N
Delete_priv	enum('N','Y')	-	N
Create_priv	enum('N','Y')	-	N
Drop_priv	enum('N','Y')	-	N
Reload_priv	enum('N','Y')	-	N
Shutdown_priv	enum('N','Y')	-	N
Process_priv	enum('N','Y')	-	N
File_priv	enum('N','Y')	-	N

Field	Type	Key	Default
Host	char(60)	PRI	""
Db	char(64)	PRI	""
User	char(16)	PRI	""
Select_priv	enum('N','Y')	-	N
Insert_priv	enum('N','Y')	-	N
Update_priv	enum('N','Y')	-	N
Delete_priv	enum('N','Y')	-	N
Create_priv	enum('N','Y')	-	N
Drop_priv	enum('N','Y')	-	N

Field	Type	Key	Default
Host	char(60)	PRI	""
Db	char(64)	PRI	""
Select_priv	enum('N','Y')	-	N
Insert_priv	enum('N','Y')	-	N
Update_priv	enum('N','Y')	-	N
Delete_priv	enum('N','Y')	-	N
Create_priv	enum('N','Y')	-	N
Drop_priv	enum('N','Y')	-	N

III.2. Configuration de la base de données

III.2.1 La base de données de la Médiathèque

Les données brutes de la médiathèque représentent une masse de plus 450000 références d'articles tirés de plus de 670 périodiques.
Ces données sont stockées selon un format très pratique :

- Les références sont rentrées par ligne
- Dans les colonnes, 10 champs permettent d'identifier les articles
  - nom d'auteurs
  - titre de l'article
  - titre du périodique
  - année de parution
  - numéro de volume
  - numéro
  - les pages dans le volume
  - numéro d'issn
  - champs réservé
  - classe

III.2.2 Stockage des données dans MySQL

a) Création de la table des données

On va reprendre la structure des données brutes pour la création de la table des données. On y incorpore également les données les moins informatives (le champs réservé est n'en effet presque jamais affecté) ou peu utilisées. MySQL permet très facilement de modifier les tables, de supprimer ou de renommer des colonnes.
A partir du shell MySQL :

```
> create table sSs_dat (nom_auteur char(100), titre_art char(255), titre_rev char(100), date char(10), volume char(10), numero char(10), page char(10), issn char(10), champ char(10), classe char(10)) ;
```

Il existe un nombre important de type de variables pour définir le type des attributs (float, char, double, varchar, plusieurs int différents, text, blob, …).

b) Chargement des données

MySQL possède un utilitaire particulièrement rapide pour rentrer les données dans la base. Les 71 mégaoctets de données brutes sont chargées en moins de cinq minutes contre près de douze heures par des moyens classiques.

Pour le chargement, les données doivent être placées dans le répertoire `data/sSs`.
Pour stocker le fichier de données brutes de janvier 1993 à juin 1998, sSs.930101-980608, on appelle :

```
> load data infile 'sSs.930101-980608' into table sSs_dat fields terminated by ',' enclosed by ' "' ;
```

La souplesse des commandes permet de définir précisément comment lire le fichier original à partir des « fields terminated by, lines terminated ou fields escaped ».
Les valeurs par défault sont :

Fields enclosed by : "
Fields escaped by : \\
Fields terminated by : \t
Lines terminated by : \n

Il faut veiller bien regarder les messages que fournit MySQL quand le chargement est terminé. Tout particulièrement les warning : ceux-ci signalent qu'une partie des enregistrements ne s'est pas déroulé de manière optimale. Ce peut être des troncatures de données ou des types de variables mal définies. Dans ce cas, il faut recommencer le chargement (heureusement il est rapide) en changeant le type des variables (varchar, varchar binary, …). Ou en augmentant la taille des types rentrés.

La fonction inverse qui récupère les données dans MySQL pour les écrire dans un fichier quelconque est :

```
> select … into outfile 'blabla.txt' fields terminated by ',' enclosed by '"' escaped by '\\' lines terminated by '\n' from …
```

c) Modifications des tables

La modification de pratiquement toute donnée dans les tables se fait grâce à la commande `alter`.

La syntaxe générale le cette commande est :

ALTER TABLE nom_de_la_table commandes ;

Parmi les commandes on notera,
ADD [attribut] définition_proposée : pour ajouter une colonne d'attribut
CHANGE [attribut] nom_a_changer définition_proposée : pour changer ou renommer une colonne
RENAME nouveau_nom_de_table : pour renommer une table
DROP [attribut] : pour supprimer une colonne
ADD INDEX nom_de_cle : pour créer un index

❖ Pour redéfinir la taille d'un attribut on pourra taper :

```
> alter table sSs_dat change titre_art titre_art
char(255) ;
```

❖ Pour créer un index, il faut d'abord définir les colonnes d'attribut sur lesquels ont veut placer un index comme NOT NULL :

```
> alter table sSs_dat change titre_rev titre_rev
char(255) NOT NULL, add index (titre_rev) ;
```

d) Sélection

❖ Pour sélectionner les numéros de volume et de numéro de la revue Acta Applicandae Mathematicae :

```
> select volume,numero from sSs_dat where titre_rev='
revue Acta Applicandae Mathematicae' ;
```

❖ Connaissant le numéro de volume et le numéro, on demande les titres de la revue commençant par 'Acta Applic' :

```
> select titre_art from sSs_dat where (titre like
'Acta Applic%') and volume='5' and numero='2' ;
```

e) Insertion

❖ Pour stocker dans la table *revues* tous les nom des périodiques de la table sSs_dat :

```
> insert into revues (titre_rev) select distinct
titre_rev fron sSs_dat ;
```

f) Utilisation d'index

Avec MySQL, pour créer un index, la commande classique SQL 'create index' existe, mais uniquement pour des raisons de compatibilité, dans les faits elle est inopérante. Pour créer des index, il faut avoir recours à la commande alter.

Dans les conditions de la base de données de la médiathèque, il est **capital** de créer des index judicieux car ils améliorent de manière spectaculaire la vitesse des requêtes.
La création d'index est une opération délicate : il faut toujours garder à l'esprit que créer un index c'est pour accélérer les requêtes. Si on créé des index trop gros (index multiples) on risque de ralentir les requêtes puisque la base cherchera dans un gros fichier. De plus ces index représentent une place mémoire importante : la base de donnée initiale représente 200MegaOctets, un index mutiple de 2 ou trois attribut fera facilement quelques dizaines de Mega voire dépasser 100 Mega.

Parmi les requêtes les plus fréquemment on aura besoin de réorganiser la base pour optimiser les requêtes suivantes:

- `> select titre_art, nom_auteur, page from sSs_dat where titre_rev='titre_rev' and volume='volume' and numero='numero' ;`
- `> select distinct volume, numero from sSs_dat where titre_rev='revue';`
- `> select * from sSs_dat where nom_auteur='auteur' and date='année' ;`

Après de nombreux essais, on aura intêret à regrouper les attributs 'volume' et 'numero'. Contre toute attente, il n'est pas intéressant d'y associer l'attribut 'titre_rev'. L'attribut le plus souvent appelé (titre_rev) devra être lié à 'nom_auteur' :

Index ind_rev :

```
> alter table sSs_dat add ind_rev(titre_rev, nom_auteur) ;
```

Et ind_vol,

```
> alter table sSs_dat add ind_vol(volume, numero) ;
```

g) Contrôle de la base de données

Une fois que l'on a rentré les données, créé des index, effectué des modifications et fait des « retouches », il est bon de visualiser ce qu'on a stocké dans la base. Pour cela, on dispose des commandes

- `> describe sSs_dat ;`

Qui donne des informations sur les tables

Field	Type	Null	Key	Default	Extra
nom_auteur	char(100)				
titre_art	char(255)				
titre_rev	char(100)		MUL		
date	char(10)				
volume	char(10)		MUL		
numero	char(10)				
page	char(10)				
issn	char(10)				
champ	char(2)	YES		NULL	
classe	char(10)	YES		NULL	

```
+-----------------+--------------------+-----------+------------+-------------------+----------------+
```

On peut vérifier l'état des différents attributs stockés dans la table :

- Champ et classe ont gardé leur valeur par défaut
- Les autres colonnes ont des index multiples mais on se sait pas comment ils sont liés entre eux

Pour le avoir il faut lister la table des index :

❖ `> show index from sSs_dat ;`

```
+-----------+---------+----------+-------------+-----------------+-----------+----------------+------+
|  Table    | Non     |  Key     |     Seq     |  Column         | Collation |  Cardinality | Sub  |
|           | unique  |  name    | in index    |   name          |           |              | Part |
+-----------+---------+----------+-------- -----+-----------------+-----------+----------------+---- --+
| sSs_dat |     1   | ind_rev |      1      | titre_rev      |       A   |  NULL      |  1    |
| sSs_dat |     1   | ind_rev |      2      | volume         |       A   |  NULL      |  1    |
| sSs_dat |     1   | ind_rev |      3      | numero         |       A   |  NULL      |  1    |
| sSs_dat |     1   | ind_aut |      1      | nom_auteur |       A   |  NULL      |  1    |
| sSs_dat |     1   | ind_aut |      2      | date           |       A   |  NULL      |  1    |
+-----------+---------+----------+-------------+-----------------+-----------+----------------+-------+
```

h) Enregistrement des données

L'opération la plus importante dans la gestion de la base de données est la validation des instructions. Celle-ci est lancée depuis le shell par :

```
> bin/mysqladmin –u root reload
```

IV. Connexion de MySQL avec un serveur Web

IV.1 Les modules objets de Perl

IV.1.1. CGI.pm

Le module objet CGI.pm est maintenant intégré à la distribution standard de Perl depuis la version 5.004. CGI.pm est disponible sur la plupart des plateformes (UNIX, Linux, …).
On présente ici une introduction succinte de CGI.pm, suffisante pour faire ses premiers scripts. Comme le reste de la documentation Perl, on peut avoir accès aux pages de manuel en lançant les commandes man ou perldoc.

Un programme Perl dans lequel on aura recours à CGI.pm commence par :

```
use CGI
```

Il est utile de se servir des fonctions déjà programmées par CGI. Ceci est intéressant pour éviter trop de frappe inutile et rendre les codes plus lisibles. On notera :

:cgi importe des méthodes de gestion d'argument (param(), …)
:form donne accès aux méthodes de gestion de formulaire (textfield(), …)
:html2 importe les méthodes générant HTML2
:netscape importe les méthodes spécifiques à l'HTML de Netscape
:standard appelle les fonctions cgi, form et html2
:all pour importer toutes les méthodes

Ces méthodes seront par exemple appelées par

```
use CGI qw( :standard)
```

La création d'un nouvel objet `$query` passe par l'invocation de

```
$query = new CGI
```

équivalent à

```
$query = CGI->new()
```

Appel de la méthode new() de la classe CGI et affectation dans l'objet `$query` (valable pour les méthodes POST et GET)

CGI.pm est particulièrement utile dans deux cas

a) Passer des paramètres

Pour récupérer des paramètres passés par un formulaire, un scrollpane ou autre, on dispose de :

```
$value = $query->param('nom_param')
```
pour un paramètre

```
@value = $query->param('liste')
```
pour une liste de paramètres

```
@keywords = $query->keywords
```
donne le nom des paramètres et non leur valeur

Si l'on se dispense de donner le nom des paramètres

```
$value = $query->param
```

on aura la liste des paramètres passés par CGI.

b) Ecrire du code html

Avec CGI, on peut très facilement, sans utiliser les balises html écrire des pages html.

Typiquement, on fait appel aux méthodes dont le nom est identique aux balises html On peut indistinctement les appeler par `methode()` ou `methode`.

```
print header
```
donne l'entête
```
print start_html
```
démarre la page html
```
print br
```
saut de ligne
```
print hr
```
dessine un trait
```
print end_html
```
termine une page html

L'écriture se fait simplement par l'appel de

```
print p('Quel est votre langage favori ?')
print 1('Perl')
```

On peut rajouter des options dans chaque méthode, par exemple, pour donner un titre et donner la couleur du fond :

```
print star_html(-title=>'Formulaire', -
BGCOLOR=>'blue')
print h1({-
style=>'color :red ;'},'formulaire')
```

Pour la création de formulaires intéractifs :

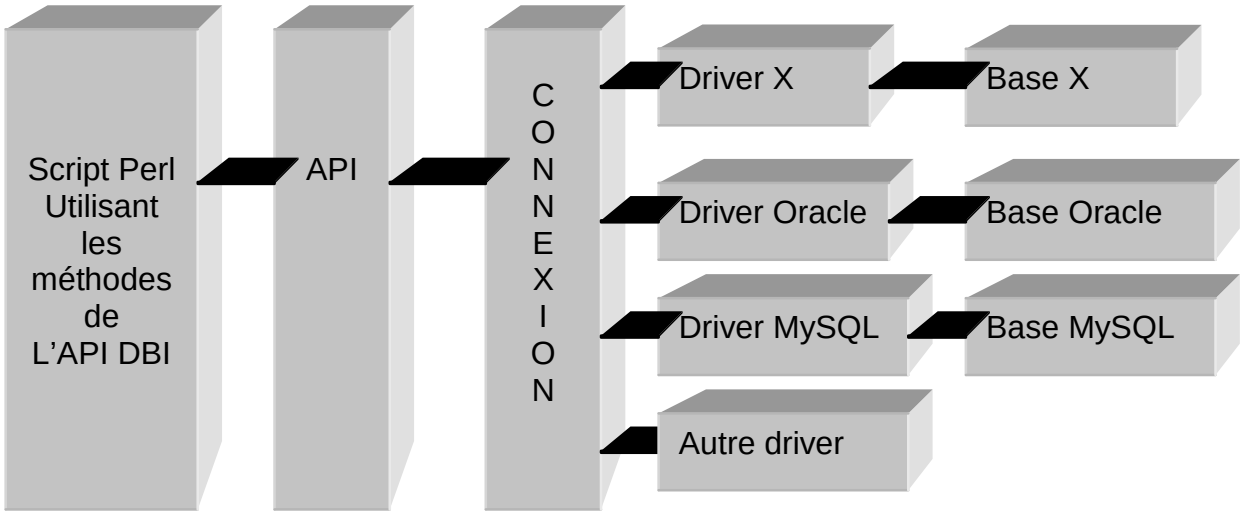
```
print start_form(    -methode=>$methode
                     -action=>$action
                     -encoding=>encoding)
print textfield(-name=>'nom du champ',
                     -default=>'valeur de depart',
                     -size=>50,
```

```
                    -maxlength=>80)
        print popup_menu('Au menu',
['Perl','Java','C'])

        %annees=(93=>'1993, 94=>'1994, …, 98=>1998)
        print scrolling_list(-NAME=> 'annees '
                    -VALUES=>[key %donnees],
                    -LABELS=>\%donnees,
                    -SIZE=>5,
                    -MULTIPLE=>1,)

        print password_field(-name=>'secret',
                        -default=>'starting value'),
                        -size=>50,
                        -maxlength=>80)

        print submit(    -name=>'Nom du bouton',
                    -value=>'valeur')

        print reset
        print end_form
```

IV.1.2. DBI

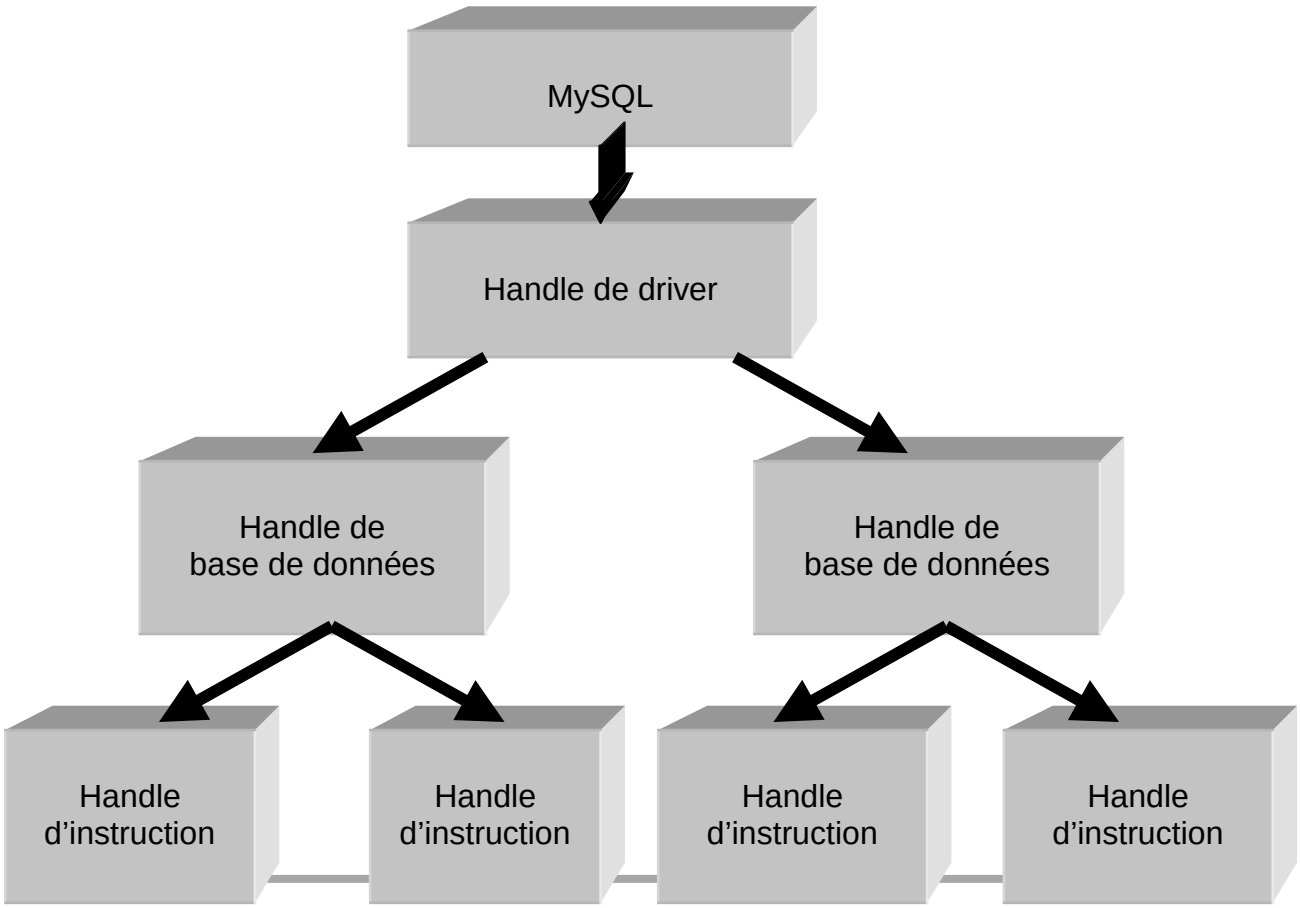
DBI est est une API (Application Programming Interface) pour le langage Perl. DBI fournit un ensemble de fonctions, variables et conventions donnant une interface de base de donnée **indépendante** du type de base de donnée utilisée.

a) Architecture d'une application DBI



b) Handles - Manipulateurs

Les manipulateurs (handles) sont des objets Perl retournés par les différentes méthodes DBI que le programmeur peut utiliser pour accéder aux données pour les différentes couches.
Dans DBI les handle peuvent être schématiquement représentés par :



En général les notations et conventions utilisées, qu'on retrouve souvent, sont les suivantes :

DBI nom statique de la classe
$dbh objet de manipulation de base de données
$sth objet de manipulation d'instructions
$drh objet de manipulation de driver
$rc code de retour (pour les booléens)
$rv code de retour (pour les integer)
@ary liste de valeurs retournées par la base (une colonne de données)
$rows nombre de colonnes traitées
$fh manipulateur de fichier

c) Préparation des requêtes

DBI « prépare » généralement une instruction SQL pour une exécution. Une instruction « préparée » est souvent identifiée par l'identificateur $sth.
La séquence typique pour les requêtes est :

connect,
    prepare
        execute, fetch, fetch, …finish,
        execute, fetch, fetch, …finish,
        execute, fetch, fetch, …finish.

ou s'il ne s'agit pas de requêtes :

connect,
    prepare
        execute,
        execute,
        execute.

d) Connexion

La plupart du temps, on verra un appel de la méthode qui charge le driver. En effet, DBI, est indépendant du type de base, il peut donc appeler (dans le même programme éventuellement) plusieurs types de base. Il faut donc préciser le type de base, et donc de driver, correspondant.
MySQL contient un module DBI dans sa distibution, on peut donc se dispenser des commandes suivantes.

```
$drh = DBI->install_driver('type_de_base')
```

L'établissement d'une connexion se fait par :

```
$dbh = DBI->connect($source_de_données, $username, $password) ;
```

La connexion (session) est établie avec la base de données et retourne un manipulateur de base de donnée ($dbh).

On peut regrouper plusieurs informations pour la valeur '$sources_de_données'.

DBI : Nom_de_driver : nom_de_la_base
DBI : Nom_de_driver : nom_de_la_base@hostname
DBI : Nom_de_driver : nom_de_la_base~hostname !port
DBI : Nom_de_driver : database=nom_de_la_base ;host=hostename ;port=port

e) Exemple

```
my $dbh = DBI -> connect("dbi :Oracle :$data_source ", $user, $password)
    || die "Impossible de se connecter à la base $data_source : $DBI : :errstr ";

my $sth = $dbh->prepare(q{
    SELECT nom, tel
    FROM mon_agenda
}) || die "Impossible de préparer l'instruction : $DBI : :errstr ";

my $rc = $sth->execute
    || "Impossible d'executer : $DBI : :errstr ";

print "La requête donne$sth->{NOMBRE_DE_CHAMPS} champs \n\n" ;
print "$sth->{NOM}->[0] : $sth->{NOM}->[1]\n" ;
while (($nom, $tel\n) = $sth_>fetchrow_array) {
        print "$nom : $tel\n ";
}

#prévient l'utilisateur si un problème est survenu
warn $DBI : :errstr if $DBI : :err

$sth->finish ;
```

IV.2 Connexion des données de la médiathèque avec le serveur

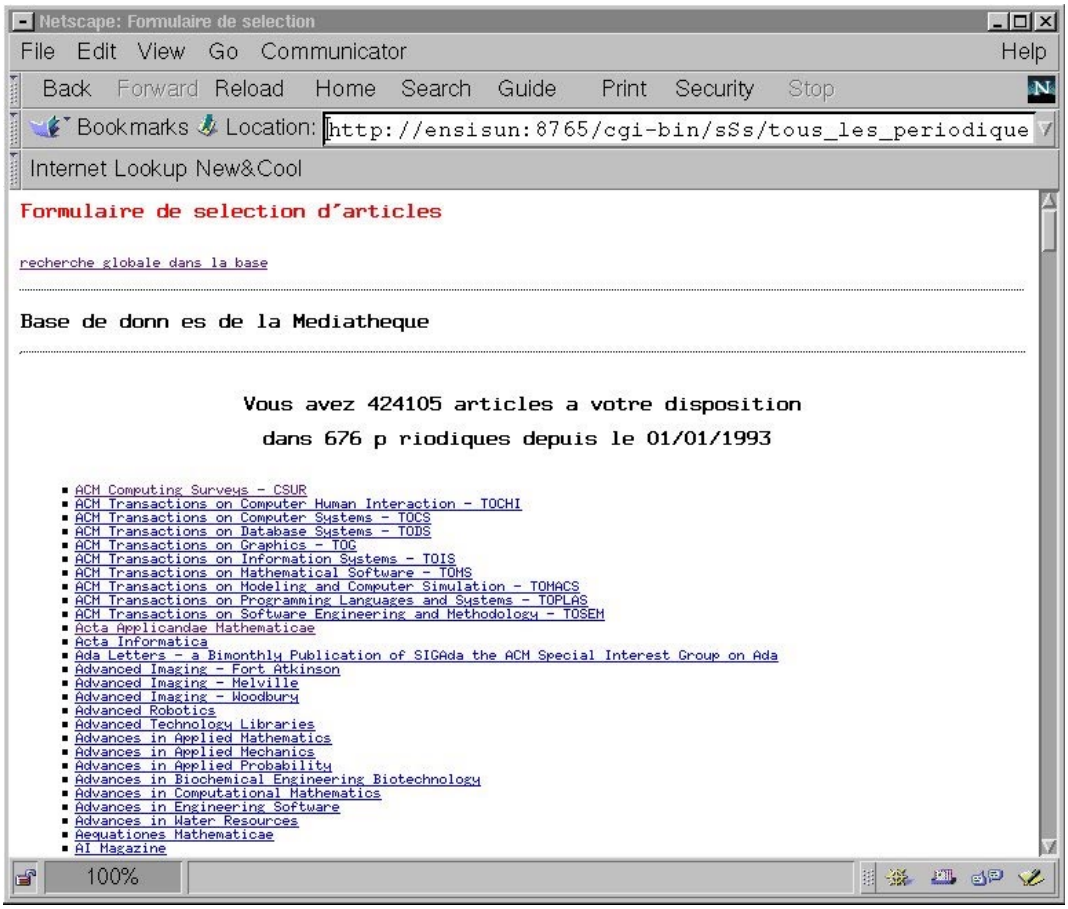
IV.2.1. L'application client-serveur

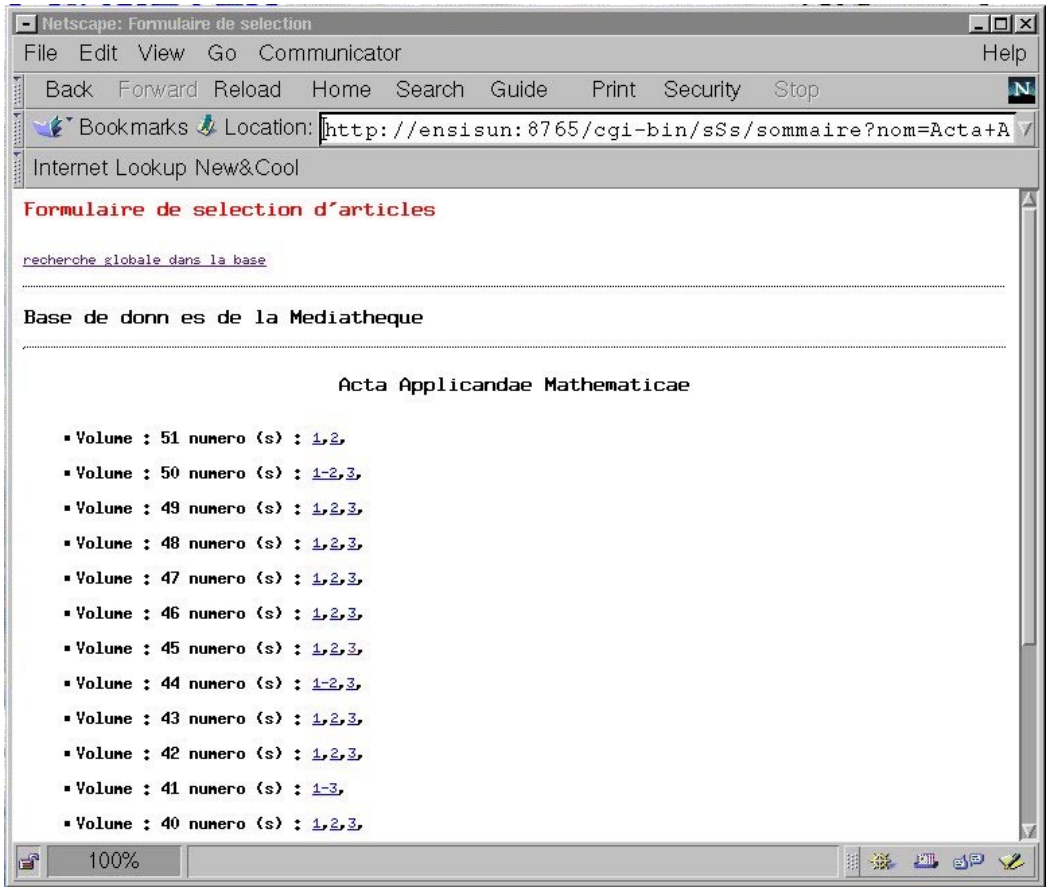
L'application développée dans ce projet comporte deux parties.

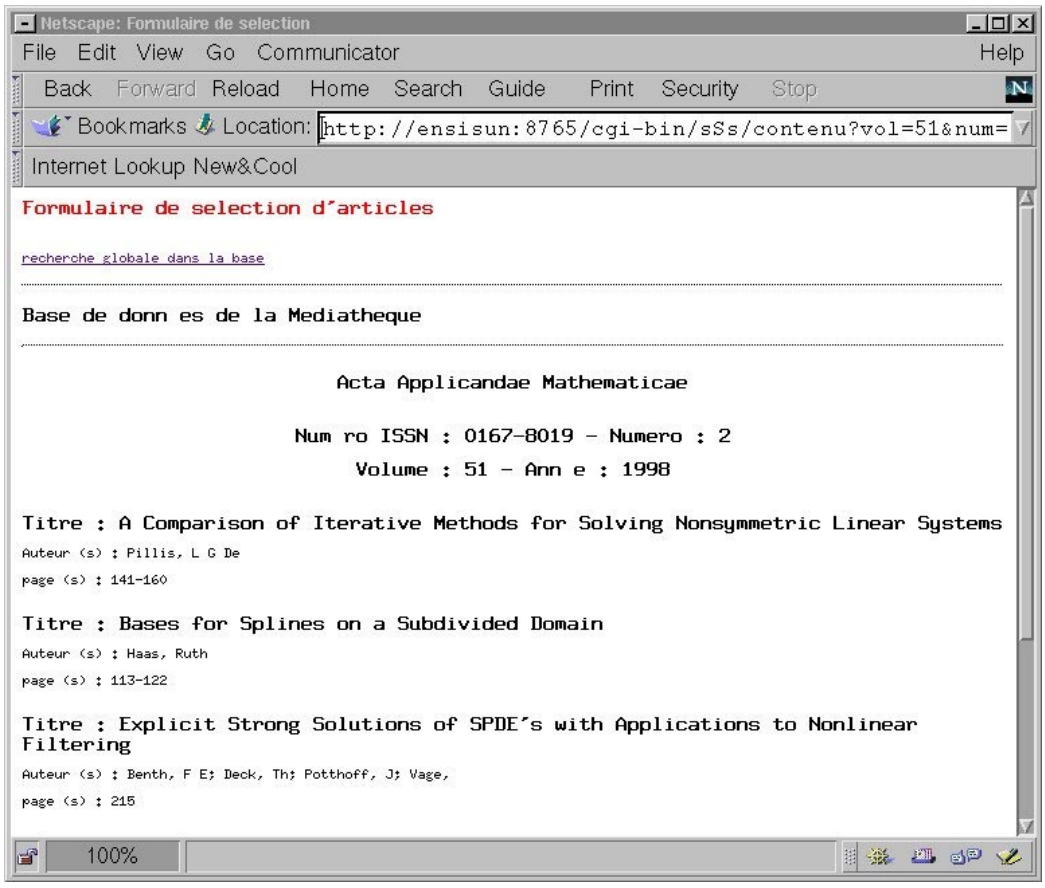
- Une application en trois pages où le client choisit les éléments qu'il désire.
  - une page où sont listés les périodiques disponibles : l'utilisateur peut cliquer sur le nom de la revue qu'il souhaite
  - la page suivante présente les volumes et les numéros du périodique choisi
  - la dernière page donne tout le contenu de la sélection :nom d'auteur, titre d'article, pages
- Une page de recherche globale sur toute la base de la médiathèque. Elle se présente sous la forme d'un formulaire. On peut choisir 4 critères de recherche :
  - selon le nom de l'auteur
  - selon le titre de la revue
  - selon l'article
  - selon le numéro ISSN

  On peut saisir une partie seulement des caractères. Ainsi, si un utilisateur ne se souvient plus exactement du nom d'un auteur ou du nom d'une revue, il peut taper 'Acta Appli' et la base cherchera la revue commençant par ses caractères : Acta Applicandae Mathematicae'.

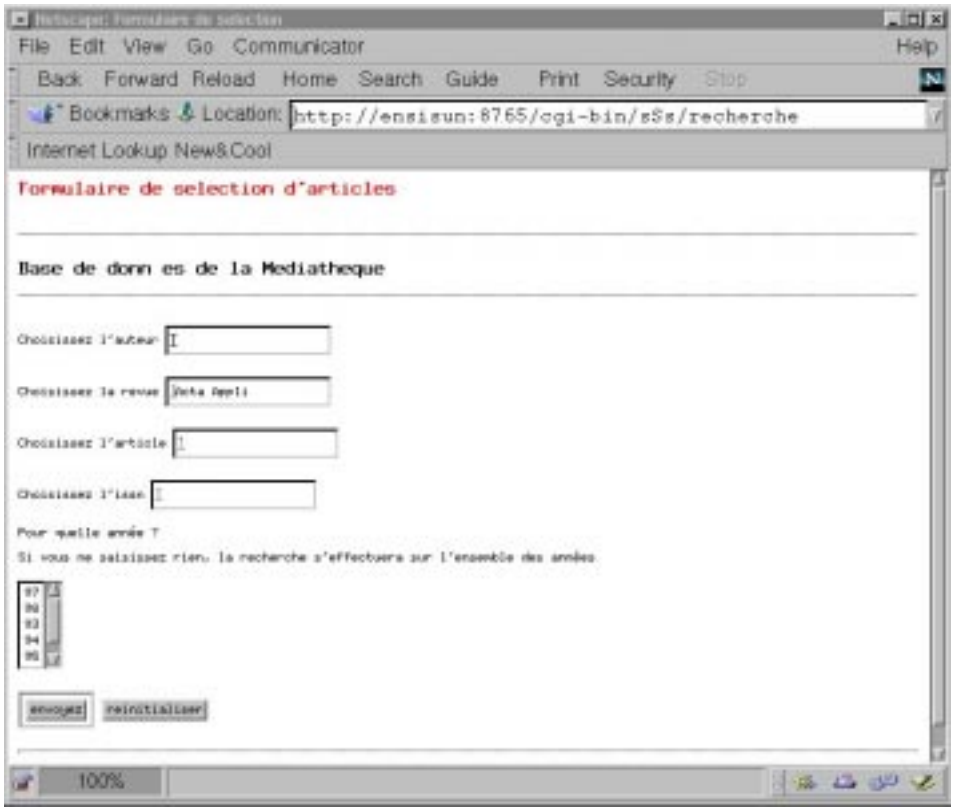
a) Les sorties écran – Requêtes

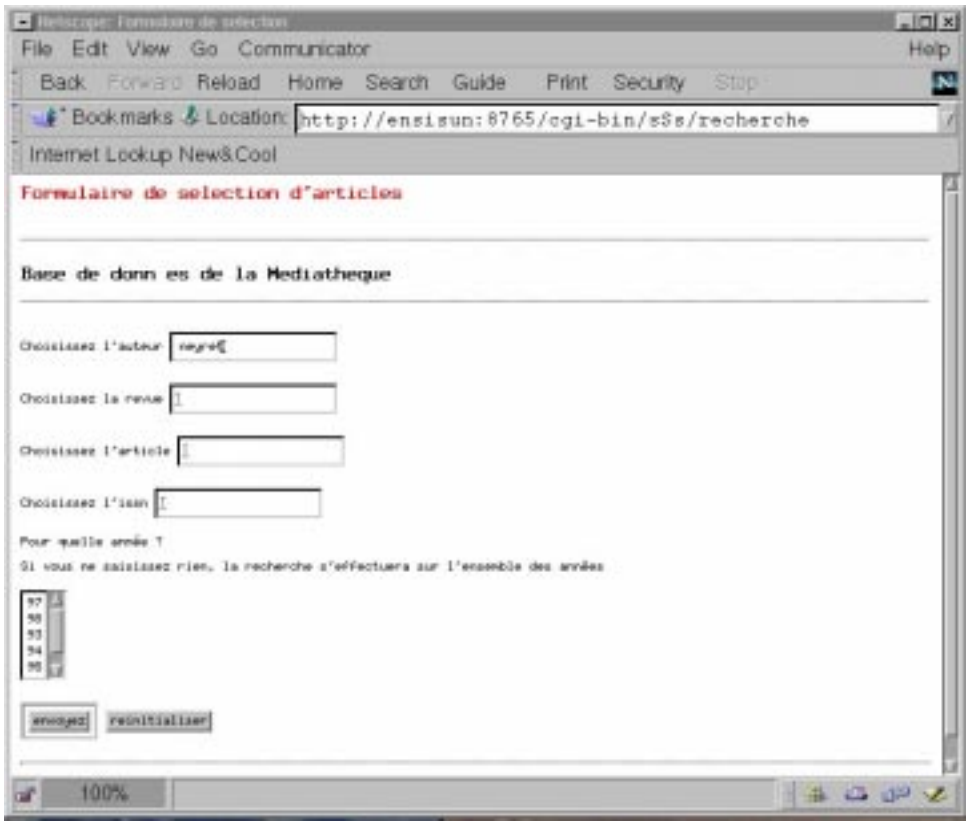


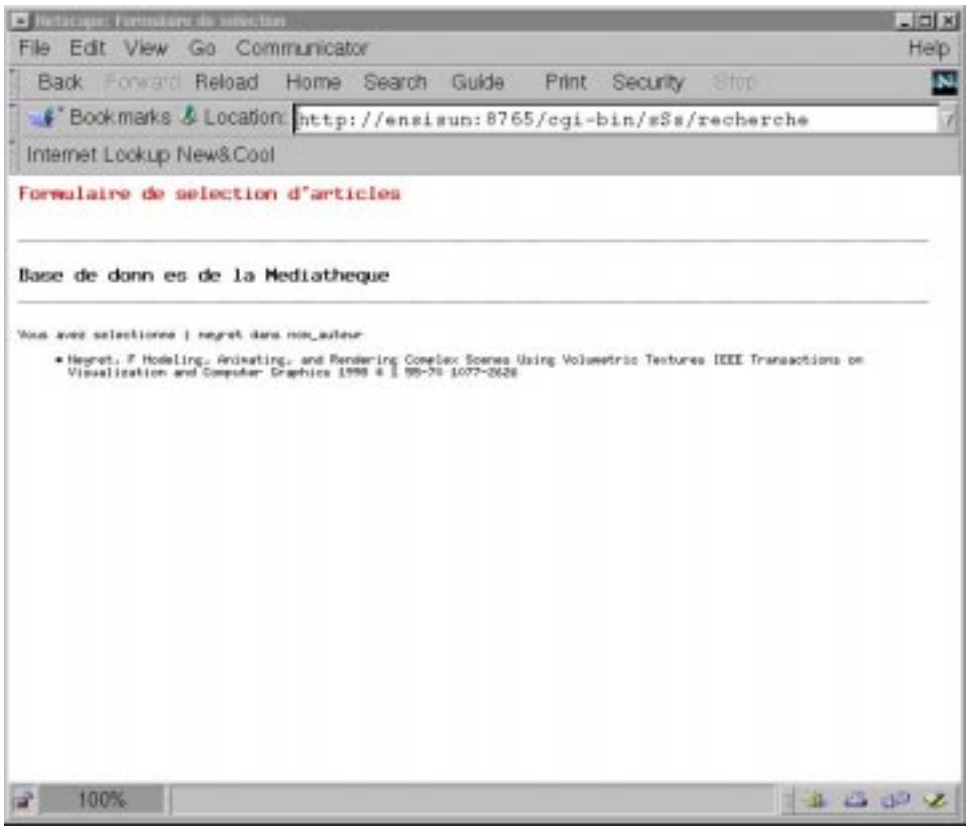

Netscape: Formulaire de selection
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://ensisun:8765/cgi-bin/sSs/sommaire?nom=Acta+A
Internet Lookup New&Cool
Formulaire de selection d'articles
recherche globale dans la base
Base de donn es de la Mediatheque
Acta Applicandae Mathematicae
Volume : 51 numero (s) : 1,2,
Volume : 50 numero (s) : 1-2,3,
Volume : 49 numero (s) : 1,2,3,
Volume : 48 numero (s) : 1,2,3,
Volume : 47 numero (s) : 1,2,3,
Volume : 46 numero (s) : 1,2,3,
Volume : 45 numero (s) : 1,2,3,
Volume : 44 numero (s) : 1-2,3,
Volume : 43 numero (s) : 1,2,3,
Volume : 42 numero (s) : 1,2,3,
Volume : 41 numero (s) : 1-3,
Volume : 40 numero (s) : 1,2,3,
100%

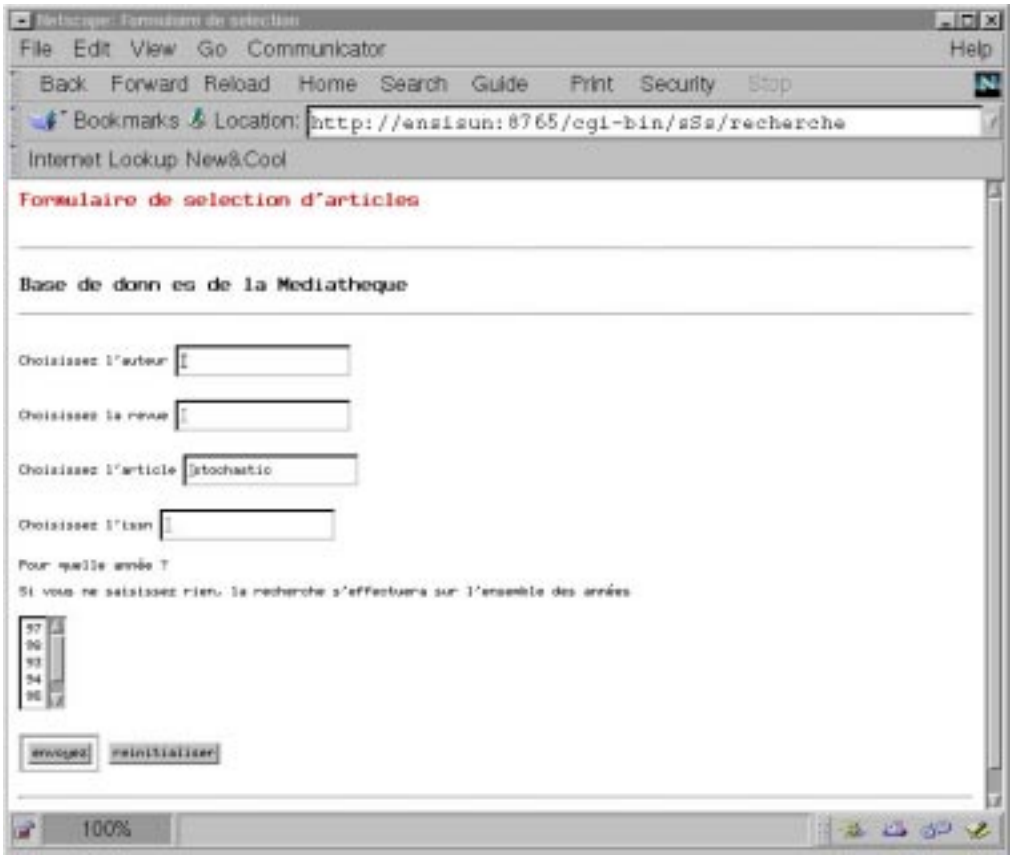

Netscape: Formulaire de selection
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://ensisun:8765/cgi-bin/sSs/contenu?vol=51&num=
Internet Lookup New&Cool
Formulaire de selection d'articles
recherche globale dans la base
Base de donn es de la Mediatheque
Acta Applicandae Mathematicae
Num ro ISSN : 0167-8019 - Numero : 2
Volume : 51 - Ann e : 1998
Titre : A Comparison of Iterative Methods for Solving Nonsymmetric Linear Systems
Auteur (s) : Pillis, L G De
page (s) : 141-160
Titre : Bases for Splines on a Subdivided Domain
Auteur (s) : Haas, Ruth
page (s) : 113-122
Titre : Explicit Strong Solutions of SPDE's with Applications to Nonlinear Filtering
Auteur (s) : Benth, F E; Deck, Th; Potthoff, J; Vage,
page (s) : 215
100%

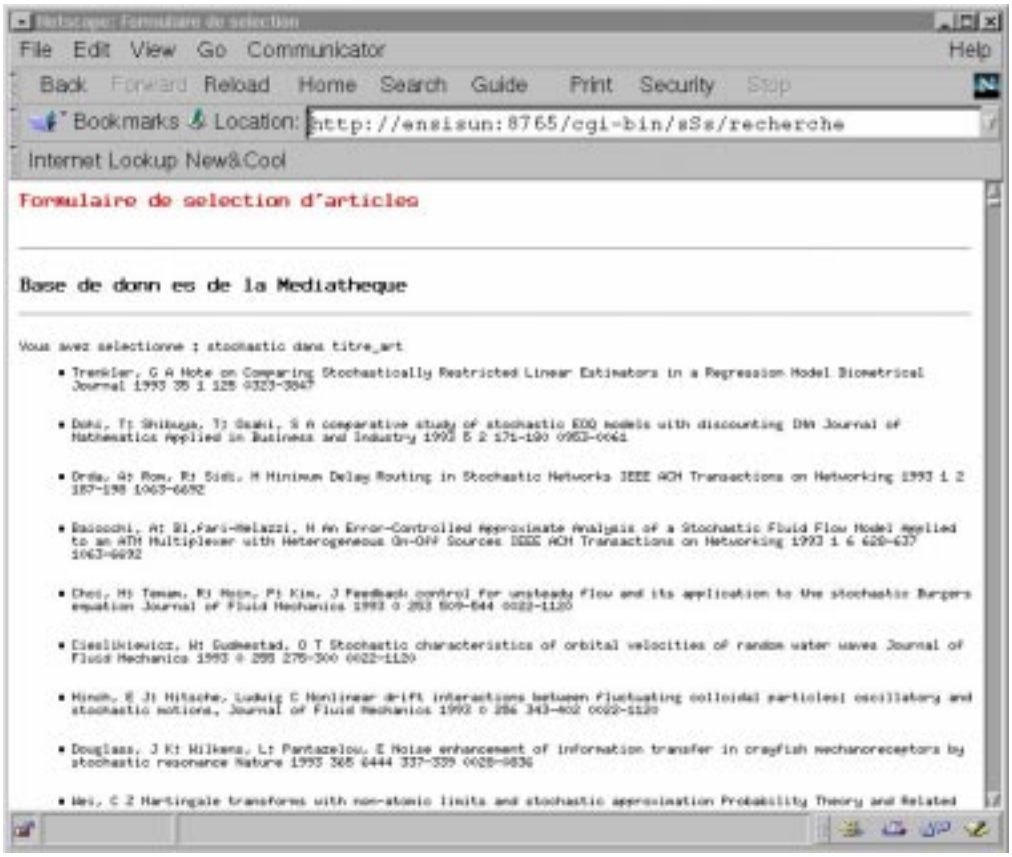
b) Les sorties écran – Recherche globale dans la base




File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://ensisun:8765/cgi-bin/sSs/recherche
Internet Lookup New&Cool
Formulaire de selection d'articles
Base de donn es de la Mediatheque
Choisissez l'auteur
Choisissez la revue
Choisissez l'article
Choisissez l'isbn
Pour quelle année ?
Si vous ne saisissez rien, la recherche s'effectuera sur l'ensemble des années
envoyez
reinitialiser
100%


File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://ensisun:8765/cgi-bin/sSs/recherche
Internet Lookup New&Cool
Formulaire de selection d'articles
Base de donn es de la Mediatheque
100%


File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://ensisun:8765/cgi-bin/sSs/recherche
Internet Lookup New&Cool
Formulaire de selection d'articles
Base de donn es de la Mediatheque
Choisissez l'auteur
Choisissez la revue
Choisissez l'article stochastic
Choisissez l'ann
Pour quelle année ?
Si vous ne saisissez rien, la recherche s'effectuera sur l'ensemble des années
envoyez
reinitialiser
100%


File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://ensisun:8765/cgi-bin/sSs/recherche
Internet Lookup New&Cool
Formulaire de selection d'articles
Base de donn es de la Mediatheque
Vous avez selectionne : stochastic dans titre_art

IV.2.2. Les scripts Perl

❖ Tous_les_periodiques

```
#!/usr/bin/perl -w
#cgi-bin/tous_les_periodiques: programme selectionnant
les #periodiques disponibles dans la base

use CGI qw(:standard);
use DBI;

print header, start_html(-title=>'Formulaire de
selection ',-BGCOLOR=>'white');

print h1({-style=>'Color: red;'},"Formulaire de
selection d'articles"), br;
print "<A HREF=/cgi-bin/sSs/recherche>recherche
globale dans la base</A>", br,br,hr;
print h1("Base de donn&eacute;es de la
Mediatheque"),hr;

my$req=0;
$query=new CGI;

print h1({-align=>CENTER},"$nom"), br;
print start_form();


$requet1="SELECT COUNT(*) FROM sSs_dat";
$requet2="SELECT COUNT(*) FROM revues";
$requet3="SELECT * FROM revues";

&selection_db($requet1,$requet2,$requet3);
print end_form(), hr();
print end_html;


#####################################################
sub selection_db {

    my($chaine1, $chaine2, $chaine3)=@_;

    $req=$decr=$name;
    $drh=DBI->install_driver('mysql');
    die "Impossibilite de charger le driver mysql\n"
unless $drh;

    $dbh=$drh->connect('sSs','root','');
```

```
    die "Impossibilite de se connecter a la
base:$DBI::errstr\n" unless $dbh;

#####################################################
# 1ere requete
#####################################################

    $sth=$dbh->prepare("$chaine1");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;

    #execution de la requete
    $sth->execute;

    while (($name,$value,$page) = $sth->fetchrow()) {
    # 1ere PAGE : on a 2 parametres
    # nom=nom de la revue, val=TITRE_REVUE
    # name=nombre d'articles
    print h1({-align=>CENTER},"Vous avez $name
articles a votre disposition");
    }
    $sth->finish;

#####################################################
# 2 eme requete
#####################################################


    $sth=$dbh->prepare("$chaine2");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;

    #execution de la requete
    $sth->execute;
   while (($nb_revues,$revues) = $sth->fetchrow()) {

    print h1({-align=>CENTER},"dans $nb_revues
p&eacute;riodiques depuis le 01/01/1993 "),br;
    }

    $sth->finish;

#####################################################
# 3 eme requete
#####################################################

    $sth=$dbh->prepare("$chaine3");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;
```

```
    #execution de la requete
    $sth->execute;

    $adr="http://ensisun:8765/cgi-bin/sSs/sommaire";

    print "<UL>";
    while (($nb_revues,$revues) = $sth->fetchrow()) {
    $ref=$nb_revues;
    $ref=~ tr/ /+/;
    print "<LI><A HREF=$adr?nom=$ref>$nb_revues</A>";
    }
    print "</UL>";
    $sth->finish;

    $dbh->disconnect;
}
```

❖ Sommaire

```
#!/usr/bin/perl -w
#cgi-bin/sommaire: programme selectionnant les volumes
et les #numeros correspondant a un periodique

use CGI qw(:standard);
use DBI;


print header, start_html(-title=>'Formulaire de
selection ',-BGCOLOR=>'white');

print h1({-style=>'Color: red;'},"Formulaire de
selection d'articles"), br;
print "<A HREF=/cgi-bin/sSs/recherche>recherche
globale dans la base</A>", br,br,hr;
print h1("Base de donn&eacute;es de la
Mediatheque"),hr;


$query=new CGI;

$nom_revue=$query->param("nom");

print h1({-align=>CENTER},"$nom_revue"), br;
print start_form();

$requet="SELECT DISTINCT volume,numero FROM sSs_dat
WHERE titre_rev='$nom_revue'";

&selection_db($requet);

print end_form(), hr();
print end_html;

#########################################################
sub selection_db {

    my($chaine)=@_;

    $drh=DBI->install_driver('mysql');
    die "Impossibilite de charger le driver mysql\n"
unless $drh;

    $dbh=$drh->connect('sSs','root','');
    die "Impossibilite de se connecter a la
base:$DBI::errstr\n" unless $dbh;
```

```
    $sth=$dbh->prepare("$chaine");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;

    #execution de la requete
    $sth->execute;

    while (($name,$value) = $sth->fetchrow()) {

    push(@val_num,$value);
    push(@val_vol,$name);
}

    $count=-1;
    $valref=-999;
    foreach $i (0 .. $#val_vol) {

        $valnum=pop (@val_num);
        $valvol=pop (@val_vol);

        if ($valvol ne $valref) {
        $count+=1;
        $stock_num[$count]=$valnum;
        $stock_vol[$count]=$valvol;
        $inter=$valnum;
        } else {
        $stock_num[$count]="$inter"."+"."$valnum";
        }
        $valref=$valvol;
        $inter=$stock_num[$count];
    }

    $adr="http://ensisun:8765/cgi-bin/sSs/contenu";
    $req=$nom_revue;
    $req=~ tr/ /+/;
    foreach $j (0 .. $#stock_vol) {
    @sortie=split(/\+/,$stock_num[$j]);

    if ($count) {
        print "<UL>";
        print "<LI><B>Volume : $stock_vol[$j] numero
(s) : </B>";
        while (@sortie) {
        $val=pop @sortie;
        print "<A
HREF=$adr?vol=$stock_vol[$j]&num=$val&nom=$req>$val</A
><b>,</b>";
        }
    } else {
```

```
        # s'il n'y a pas de numero de volume on sort
les numeros
        # selon la largeur du champs
        print "<UL>";
        print "<LI><B>Numeros : </B>",br,br;

        foreach (@sortie) {
        printf "<A
HREF=$adr?vol=$stock_vol[$j]&num=$_&nom=$req>%$20s\n</
A><b>,</b>",$_;
        }
    }
    print br;
    print "</UL>";
    }
    $sth->finish;

    $dbh->disconnect;
}
```

❖ Contenu

```
#!/usr/bin/perl -w
#cgi-bin/contenu: programme selectionnant le titre des
articles, le nom des #auteurs, les pages
#de sélection d'article

use CGI qw(:standard);
use DBI;

print header, start_html(-title=>'Formulaire de
selection ',-BGCOLOR=>'white');
print h1({-style=>'Color: red;'},"Formulaire de
selection d'articles"), br;
print "<A HREF=/cgi-bin/sSs/recherche>recherche
globale dans la base</A>", br,br,hr;
print h1("Base de donn&eacute;es de la
Mediatheque"),hr;


$query=new CGI;

$vol=$query->param("vol");
$num=$query->param("num");
$nom_revue=$query->param("nom");

print h1({-align=>CENTER},"$nom_revue"), br;

print start_form();
#on selectionne le titre des articles
$requet1="SELECT DISTINCT date,issn,volume FROM
sSs_dat WHERE titre_rev='$nom_revue' and volume='$vol'
and numero='$num'";
$requet2="SELECT DISTINCT titre_art,nom_auteur,page
FROM sSs_dat WHERE titre_rev='$nom_revue' and
volume='$vol' and numero='$num'";
&selection_db($requet1,$requet2,$requet3);

print end_form(), hr();
print end_html;


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

sub selection_db {

    my($chaine1, $chaine2)=@_;

    $req=$decr=$name;
```

```
    $drh=DBI->install_driver('mysql');
    die "Impossibilite de charger le driver mysql\n"
unless $drh;

    $dbh=$drh->connect('sSs','root','');
    die "Impossibilite de se connecter a la
base:$DBI::errstr\n" unless $dbh;

#####################################################
# 1ere requete
#####################################################


    $sth=$dbh->prepare("$chaine1");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;

    #execution de la requete
    $sth->execute;

    while (($name,$value,$page) = $sth->fetchrow()) {
    print h1({-align=>CENTER},"Num&eacute;ro ISSN :
$value - Numero : $num");
    if ($vol eq '0' ) {
        print br;
    }
    else {
        print h1({-align=>CENTER},"Volume : $page -
Ann&acute;e : $name"),br;
    }
    }

    $sth->finish;

#####################################################
# 2 eme requete
#####################################################

    $sth=$dbh->prepare("$chaine2");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;

    #execution de la requete
    $sth->execute;

    while (($name,$value,$page) = $sth->fetchrow()) {

    print h1("Titre : $name ");
    print p("Auteur (s) :  $value");
    print p("page (s) : $page"),br;
```

```
    }
    $sth->finish;

    $dbh->disconnect;
}
```

❖ Recherche

```
#!/usr/bin/perl -w
#cgi-bin/recherche: programme sous forme de formulaire
qui propose
#une selection aux choix :
#       - soit tous les articles
#       - soit tous les nom d'auteurs
#       - soit les n° ISSN
#       - soit tous les periodiques
#               correspondant a la saisie du client
#on peut choisir les annees, par defaut la requete est
faite sur 93-98


use CGI qw(:standard);
use DBI;

print header, start_html(-title=>'Formulaire de
selection ',-BGCOLOR=>'white');
print h1({-style=>'Color: red;'},"Formulaire de
selection d'articles"), br;
print hr();
print h1("Base de donn&eacute;es de la
Mediatheque"),hr;

if (param()) {
    if (param("nom_auteur")) {
    $name=param("nom_auteur");
    $requet1="nom_auteur";
    }
    elsif (param("issn")) {
    $name=param("issn");
    $requet1="issn";
    }
    elsif (param("titre_art")) {
    $name=param("titre_art");
    $requet1="titre_art";
    }
    else {
    $name=param("titre_rev");
    $requet1="titre_rev";
    }
    if (param("annees")) {
    $requet2=param("annees");
    }
    &selection_db($requet1,$requet2);

    }else{
```

```
        print start_form();

        print p("Choisissez l'auteur",
textfield("nom_auteur"));
        print p("Choisissez la revue",
textfield("titre_rev"));
        print p("Choisissez l'article",
textfield("titre_art"));
        print p("Choisissez l'issn",
textfield("issn"));
        %annees=(# on rentre les valeurs associees
aux donnees
                # dans la table de hachage %donnees
                93 => "1993",
                94 => "1994",
                95 => "1995",
                96 => "1996",
                97 => "1997",
                98 => "1998",
                );


        print p("Pour quelle ann&eacute;e ?");
        print p("Si vous ne saisissez rien, la
recherche s'effectuera sur l'ensemble des
ann&eacute;es");
        print scrolling_list(
                    -NAME => "annees",
                    -LABELS => \%donnees,
                    -VALUES => [keys %annees],
                    -SIZE => 5,
                    -MULTIPLE => 1,
                    );

        print p(submit("envoyez"),
reset("reinitialiser"));
        print end_form(), hr();
    }
print end_html;


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

sub selection_db {

    my($chaine1,$chaine2)=@_;
    print p("Vous avez selectionne : $name dans
$chaine1");
    if ($chaine2) {
    print p("pour l'ann&eacute;e 19$chaine2");
```

```
    }
    $type=$name;
    $drh=DBI->install_driver('mysql');
    die "Impossibilite de charger le driver mysql\n"
unless $drh;

    $dbh=$drh->connect('sSs','root','');
    die "Impossibilite de se connecter a la
base:$DBI::errstr\n" unless $dbh;

    if ($chaine2) {
    $sth=$dbh->prepare("SELECT * FROM sSs_dat WHERE
($chaine1 like '%$name%') and (date like
'19$chaine2')");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;
    } else {
    $sth=$dbh->prepare("SELECT * FROM sSs_dat WHERE
($chaine1 like '%$name%')");
    die "Impossibilite de faire une
requete:$DBI::errstr\n" unless $sth;
    }
    #execution de la requete
    $sth->execute;

    $rc=$sth->rows;
    my $table = $sth->fetchall_arrayref;
    my ($i,$j);
    for $i (0 .. $#{$table}) {
    print "<ul><li>";
    for $j (0 .. $#{$table->[$i]}) {
        print "$table->[$i][$j]\t";
    }
    print "</ul>",br;
    }

    if ($rc eq 0) {
    print br;
    print p("$name n'est pas reference dans la
base");
    }

    $sth->finish;

    $dbh->disconnect;
}
```

V. Conclusion

Les modules objet Perl sont d'une grande utilité dans ce type de projet. Avec un seul langage puissant et facile d'emploi on dispose de:

- CGI.pm

- Passer des paramètres aux programmes CGI
- Ecrire du code html

- DBI pour

- Installer le driver approprié quel qu'il soit
- Préparer les requêtes avec un langage standard : SQL
- Exécuter des requêtes
- Recueillir les données issues de la base
- Traiter et afficher les données

VI. Bibliographie

- Introduction à Perl

Randal L Schwartz et Tom Christiansen
Ed. O'Reilly

- Serveur WWW et programmation CGI

Xavier Ducret Juin 1998
ASI ENSIMAG

- Le langage Perl

Cécile Poiraud Juin 199
ASI ENSIMAG

- PerlToolkit – 3 volumes

Ed. O'Reilly

- Configuration du serveur Apache – Aspects sécurité

Claude Gross FEVRIER 1997
CNRS/UREC

- DBI- The database Interface for Perl5

Alligator Descartes et Tim Bunce
The Perl Journal SPRING 1997

- MySQL Reference Manual for Version 3.21.30

http://www.mysql.com

- Man Perl, Perdoc CGI, DBI, PerlObj, dbi.faq

- Sites Web

Nouveau Guide Internet par Gilles Maire :
HTML, CGI, Perl …
http://www.imaginet.fr/~gmaire

MySQL :
http://www.mysql.com