Mise en place d'un parc d'instrumentation sur le WWW

Emmanuel Alves Moreira
Laure Chao

Responsable de projet: **Patrice Kadionik.**

Option Technologie de l'Image et de la Communication, promotion 1999.

Remerciements

Nous tenons à remercier Patrice Kadionik sans qui le projet n'aurait pas lieu d'être et pour tous ses conseils et aides, Hicham El Alaoui pour sa contribution à notre apprentissage du Java™, Jean-Baptiste pour Stargate & KDE, Thomas Zimmer pour l'analyseur de réseau HP, ainsi que Marie-Claire Gaudin et ceux et celles qui nous ont apporté tout le soutien tant technique que moral à l'élaboration de notre travail.

Table des matières

I- Introduction

Ce document présente notre travail réalisé dans le cadre du projet de fin d'étude à l'ENSERB (Ecole Nationale Supérieure d'Electronique et de Radioélectricité de Bordeaux). Le projet consiste à permettre l'accès d'un Analyseur de réseau, localisé dans un laboratoire de micro-électronique, l'IXL, à tout utilisateur connecté sur Internet et possédant les droits d'utilisation.

Tout d'abord, le projet RETWINE dans le quel s'insère notre travail, sera présenté dans ce rapport en guise d'introduction. L'accent sera mis sur ce qui a déjà été développé et sur ce que nous devons ajouter. Ensuite les solutions technologiques retenues pour le développement de notre projet seront explicitées. Puis la réalisation sera décrite ainsi que les tests effectués pour la validation du travail. Enfin, à cause du délai assez court qui nous a été imparti, une partie sera dédiée à une amélioration possible du travail afin que d'éventuels concepteurs qui prendront la relève puissent compléter le projet.

Mis en place par le laboratoire IXL de l'Université de Bordeaux, le projet RETWINE (RemoTe Web Instrumentation Network) entre dans le cadre de l'action ODL (Open and Distance Learning). Son principal intérêt est de réaliser un véritable réseau d'instrumentation virtuelle. Tout un parc d'instruments de mesure très performants et par conséquent onéreux devra être partagé par chaque partenaire au projet. L'utilisateur pilote alors l'instrument à distance via son navigateur Internet.

Il s'agit de mettre en place un moyen pédagogique original pour apprendre en ligne à utiliser et maîtriser des équipements de mesure complexes dans le cadre d'une formation universitaire. Ce moyen pédagogique est basé sur les possibilités multimédia inhérentes au réseau Internet World Wide Web (WWW): environnement en réseau, manipulation de documents hypertextes, d'images, de sons...

D'autres universités ont rejoint l'IXL sur ce projet:

- Universidad de Madrid, Espagne.
- Fachhochschule Münster, Allemagne.

Le projet RETWINE a donc acquis une envergure européenne et a bénéficié d'un financement de la part de la Commission Européenne.

II- Présentation du projet RETWINE

II.1- Concept de base

L'idée de base a été de mettre en œuvre le concept d'application client/serveur du WWW: on a d'un côté un parc d'instruments de mesure pilotés par une application serveur accessible de n'importe où dans le monde par des applications clientes (ou " browser ") via le réseau Internet. Un système d'identification est de plus mis en œuvre pour restreindre l'accès au parc d'instruments mis à disposition aux personnes dûment enregistrées. L'application serveur propose en outre le panneau de face avant virtuel de chaque instrument qui pourra être ensuite piloté par le browser de la personne autorisée.

L'intérêt d'une telle approche est ainsi de mettre à disposition de la communauté universitaire un parc d'instruments de mesure performants et récents en vue d'apprendre à les utiliser mais aussi pour réaliser des mesures dans le cadre de recherches via le réseau Internet. On a donc ainsi mis en service sur le " Web " des instruments virtuels WWW.

II.2- Enjeu

Le test de composants et systèmes VLSI est fondamental pour le développement des procédés de fabrication, la caractérisation des composants et l'analyse de défaillance. Pour les tests électriques par exemple, on a à disposition une série d'équipements couramment utilisés: système de caractérisation VLSI, analyseur de paramètres électriques, analyseur de réseau, oscilloscope numérique, générateur de fonctions, analyseur de spectre...

La mise en place d'un cours universitaire pour des étudiants en micro-électronique concernant l'utilisation et la maîtrise de ces équipements est nécessaire mais n'est pas toujours possible à cause généralement du coût élevé de tels instruments (500 KF pour un analyseur de réseau d'occasion). Ce coût élevé est donc un frein majeur à la mise en place d'un cours universitaire pratique.

L'acquisition d'un telle instrumentation à des fins purement pédagogiques revient donc trop cher et de plus, les équipements seraient dépassés en quelques années. La durée de vie moyenne d'un cours universitaire pratique est d'environ 10-15 ans ; ce qui reste incompatible avec l'évolution rapide de la micro-électronique et de son instrumentation.

L'approche originale présentée maintenant est basée sur l'utilisation des concepts multimédia du Web. Cette approche d'enseignement à distance ou télé-enseignement est maintenant possible avec ce formidable moyen de communication qu'est le WWW ; ce qui va permettre aux étudiants dans le cadre d'une formation universitaire de maîtriser et d'avoir en plus une expertise dans l'utilisation d'instruments les plus complexes mais aussi les plus récents mis à leur disposition sur le Web.

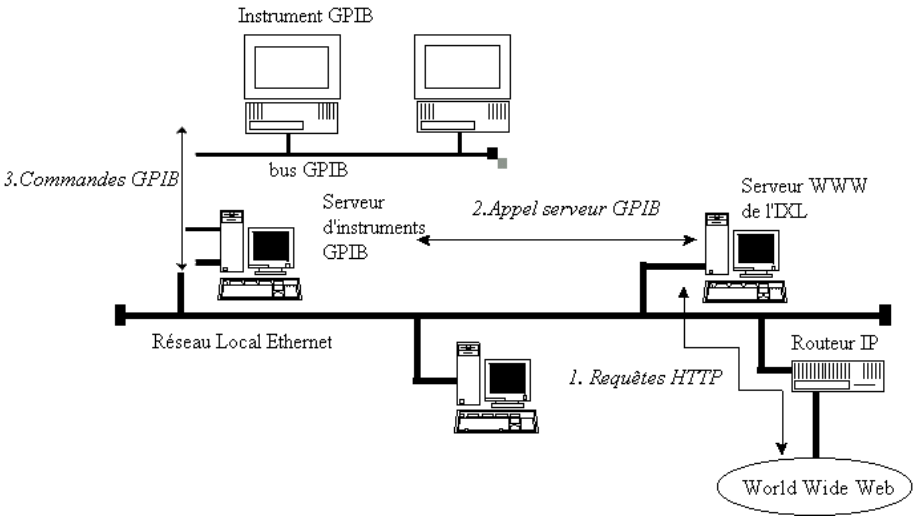
II.3- Avantage

La mise en place d'un parc d'instruments virtuels accessibles par le Web apporte des avantages incontestables. Par exemple:

- ✓ Les étudiants sont toujours en contact avec des instruments nouveaux et performants qui ne peuvent pas être toujours utilisés dans le cadre d'un cours pratique classique.
- ✓ Les étudiants des pays en voie de développement peuvent accéder aux instruments les plus modernes.
- ✓ La communication entre tous les étudiants est possible en mettant en place un groupe de discussion ou SIG (" Special Interest Group ").
- ✓ Le contrôle à distance des instruments virtuels est possible à partir d'autres continents; ce qui autorise à mettre en place un service de nuit local pour l'accès à l'instrument qui est alors non utilisé localement.

II.4- Stade actuel

Actuellement, quelques instruments ont déjà été intégrés et sont disponibles sur le WWW. L'architecture réseau actuelle correspond à la figure suivante:



Architecture d'implémentation RETWINE

Il est nécessaire d'avoir une station de travail (ou un PC) possédant une carte GPIB pour piloter les instruments IEEE-488. Une application serveur doit être développée à cet effet. Un serveur WWW doit être aussi mis en place et peut être installé sur la station de travail ou mieux sur une deuxième pour des raisons de sécurité. Il convient ensuite de développer les pages HTML où l'on définira les principales fonctions et la face avant de chaque instrument.

L'interaction entre l'application serveur et les requêtes valides reçues par le serveur WWW est réalisée à l'aide de requêtes HTML " GET " ou " POST "; ce qui oblige donc à écrire des scripts CGI (" Common Gateway Interface "). Ces pages deviennent alors accessibles par tout le monde par le Web. Un mécanisme d'identification sera enfin à mettre en place pour identifier par mot de passe l'accès au service d'instrument virtuel. La mise en place d'une base de données indépendante de celle du système d'exploitation est généralement offerte par le serveur WWW pour la gestion de mots de passe. Par ailleurs, un système de chien de garde interdit la connexion d'un autre utilisateur désirant utiliser un appareil déjà sollicité.

L'objectif de notre projet est de poursuivre dans cette direction. Nous nous proposons d'intégrer un instrument supplémentaire à ceux déjà disponibles; l'analyseur de réseau HP8510B. A terme, l'architecture actuelle de la figure de la page précédente va évoluer mais le principe restera le même.

II.5- Cahier des charges pour l'ajout du HP8510B

L'analyseur de réseaux Hewlett Packard 8510B doit être rajouté au réseau RETWINE d'instruments de mesure déjà implémentés. Le dispositif de pilotage à distance repose sur une architecture client-serveur.

Une applet Java communique avec un serveur WWW. L'applet ouvre une socket sur le serveur pour y ordonner l'exécution d'un programme. Ce programme, le driver de l'instrument, commande via un bus GPIB, l'analyseur de réseaux HP 8510B et fournit en retour un résultat. Le serveur WWW doit gérer les scripts CGI pour l'exécution de commandes locales sur l'appareil, et reconnaître les applets Java qui seront transmises au client pour l'interfaçage avec le serveur. Ainsi, il doit donc être connecté directement ou indirectement à l'appareil via le bus GPIB.

Par ailleurs, le serveur doit disposer de moyens de protection afin de contrôler l'accès et l'utilisation de l'appareil d'instrumentation. La solution la plus générique doit être adoptée pour faciliter l'implémentation ultérieure d'autres instruments.

III- Mise en œuvre technologique

III.1- L'instrument HP8510B

III.1.1- Description de l'instrument

Le HP8510B est un analyseur de réseaux dont la gamme de fréquence va de 45 MHz à 26.8 GHz. L'appareil contrôle lui-même un générateur de fréquence et un testeur (cf. figure ci-après).

HP8510B accompagné du générateur de fréquence et du testeur.

L'instrument mesure les caractéristiques d'amplitude et de phase de réseaux et de composants tels que filtres, amplificateurs, atténuateurs et antennes. Deux types de mesure peuvent y être effectués: des mesures de réflexion et des mesures de transmission. Un signal d'entrée généré par une source radiofréquence contrôlée par le HP8510B est appliqué à l'appareil testeur. Ce signal est ensuite comparé au signal réfléchi à l'entrée du testeur et au signal transmis à la sortie de celui-ci.

Les mesures de réflexion sont réalisées en comparant les signaux incident et réfléchi. Les résultats dans les données de mesure sur les caractéristiques du composant ou du réseau comprennent le coefficient de réflexion, l'impédance, le taux d'onde stationnaire, les pertes… Les mesures de transmission sont réalisées en comparant les signaux incident et transmis. Les résultats dans les données de mesure sur les caractéristiques du composant ou du réseau comprennent le coefficient de transmission, le délai électrique, les pertes ou gain, le déphasage… Les caractéristiques inverses du composant à tester peuvent être mesurés en envoyant le signal incident sur sa sortie. On peut ainsi mesurer les paramètres S d'un quadripôle.

III.1.2- Le bus GPIB

Le HP8510B peut être piloté grâce à son interface IEEE 488 (Institute of Electrical and Electronics Engineers). Le bus GPIB (General Purpose Instrumentation Bus) est donc utilisé pour le télécommander via un PC, ou une station, équipé d'une carte IEEE.

Le bus GPIB est un interface standard de communication parallèle, à 8 bits et dont la vitesse de transfert des données est supérieure à 1Moctets/s. Il permet la liaison entre un système de contrôle (PC ou station) et jusqu'à 15 appareils possédant une adresse GPIB distincte.

III.1.3- Driver de l'instrument

Afin de piloter l'instrument à partir du système de contrôle (PC ou station), un driver est nécessaire. Ce programme consiste à envoyer des données à l'appareil et à en recevoir via l'interface GPIB. La carte IEEE que possède Treville, la station Sun du laboratoire IXL reliée au bus GPIB, est fournie avec une bibliothèque de fichiers écrits en langage C. Ainsi, le driver sera développé en C, comme les drivers des appareils déjà implémentés au réseau RETWINE.

III.2- L'interface graphique

III.2.1- La façade avant de l'instrument

Afin de présenter à l'utilisateur un outil le plus proche possible de l'instrument réel, l'interface graphique reprendra bien sûr la face avant de l'appareil. Pour cela, on part d'une photo de la face avant de l'instrument sur laquelle on place les composants graphiques. Cela dit, il faut prendre en compte les limites posées par une représentation graphique qui ne permettent pas d'avoir une interface absolument semblable à l'instrument réel. Ainsi, pour un meilleur confort d'utilisation, les menus seront placés dans une fenêtre séparée car il n'est pas possible de l'afficher directement sur la face avant de l'instrument.

III.2.2- Le choix de Java

Le choix de Java comme outil de développement de l'interface graphique s'impose de lui-même. En effet, nous sommes en train de développer une application client-serveur munie d'une interface graphique destinée à piloter un instrument via Internet. Pour atteindre cette objectif, nous avons donc besoin d'un outil adapté aux besoins d'une application multimédia et indépendant de la plate-forme. Avec ses bibliothèques de composants graphiques (qui font parfois penser au VHDL) pour leur utilisation, sa gestion événementielle et sa portabilité, Java répond tout à fait à nos critères.

III.2.3- Le choix du JDK

Le choix du *Java Developpment Kit (JDK)* est décisif dans la mesure où il va décider sur la façon dont l'interface sera implémentée. La bibliothèque de composants graphiques et la gestion événementielle ont évolué avec les versions du JDK, ce qui permet de développer plus efficacement des interfaces. Le projet RETWINE a pu dans une certaine mesure bénéficier de ces évolutions pour l'intégration de ses instruments dans le parc virtuel bien que certains problèmes en découlent également.

L'analyseur d'impédance et de gain/phase HP4194A

L'HP4194A est le premier appareil mis à la disposition d'utilisateurs via internet. **Il a été développé avec le JDK1.0.**

- L'interface graphique est constituée d'une photo de la face avant de l'instrument sur laquelle sont définies des zones qui génèrent des événements.
- La gestion événementielle n'est pas des plus pratiques.
- L'utilisation des *softkeys* dans une fenêtre séparée dont le concept sera repris par la suite.

L'analyseur de paramètres pour semi-conducteurs HP4155A

L'HP4155A est le second instrument du parc virtuel RETWINE. **Il a été développé avec le JDK 1.2.** Le JDK 1.2 est la version la plus récente qui a été utilisée pour le projet RETWINE. Ses points forts sont :

- Des composants graphiques tels que les *swing* permettant de réaliser facilement des interfaces de qualité.
- Une gestion événementielle remaniée (depuis le JDK 1.1).

Mais un problème non négligeable toutefois : actuellement, les machines virtuelles Java des navigateurs ne supportent pas tous le JDK 1.2.

L'analyseur de réseau HP8510B

Pour éviter ce problème avec les navigateurs et permettre aux utilisateurs d'accéder sans difficultés à l'instrument que nous avons implémenté, nous avons préféré revenir à un JDK plus ancien mais qui est supporté par les navigateurs. **Le choix du JDK 1.1.5 est un bon compromis.** Nous pouvons envisager de réaliser une interface graphique :

- Conviviale avec des composants graphiques posés sur la photo de l'instrument pour donner un peu plus de relief.
- Une gestion événementielle plus efficace qu'avec un JDK 1.0.

- Une interface supportée par les navigateurs contrairement au JDK 1.2 qui ne l'est pas encore.

Le parc d'instruments RETWINE

Il est tout à fait normal de faire évoluer les interfaces des instruments avec les évolutions des outils dont nous disposons (comme le JDK), ce qui explique les différences d'implémentation d'un appareil à l'autre. Mais il est cependant souhaitable de maintenir une certaine homogénéité entre les instruments. Il sera donc probablement nécessaire de songer à faire évoluer les instruments existants. Ainsi, il serait souhaitable de faire évoluer le HP4194A qui a été développé avec le JDK 1.0.

D'autres instruments seront ajoutés au parc virtuel, la stratégie de développement du projet RETWINE doit donc prévoir ces deux aspects que sont l'intégration de nouveaux instruments et l'évolution de ceux existants.

III.3- Communication Client/Serveur

III.3.1- Architecture Client/Serveur

L'architecture client/serveur est une technologie informatique très utilisée actuellement notamment dans le domaine des SGBD (Système de Gestion de Base de Données) ou de l'Internet. Deux acteurs entrent donc en jeu dans cette architecture, le client et le serveur, à travers un réseau le plus souvent.

Le serveur est un programme ou une machine programmée pour rendre toujours un même service suite à une requête qui lui est adressée. Le client est un programme ou une machine qui demande un service à un serveur en lui adressant une requête.

Le client/serveur est souvent utilisé pour accéder à de grandes bases de données qui sont stockées sur une machine, le serveur, et qui sont utilisées par plusieurs utilisateurs. Le serveur a pour mission de répondre aux requêtes que lui adressent les clients, en modifiant ou interrogeant la base de données qu'il gère et en envoyant ensuite la réponse attendue.

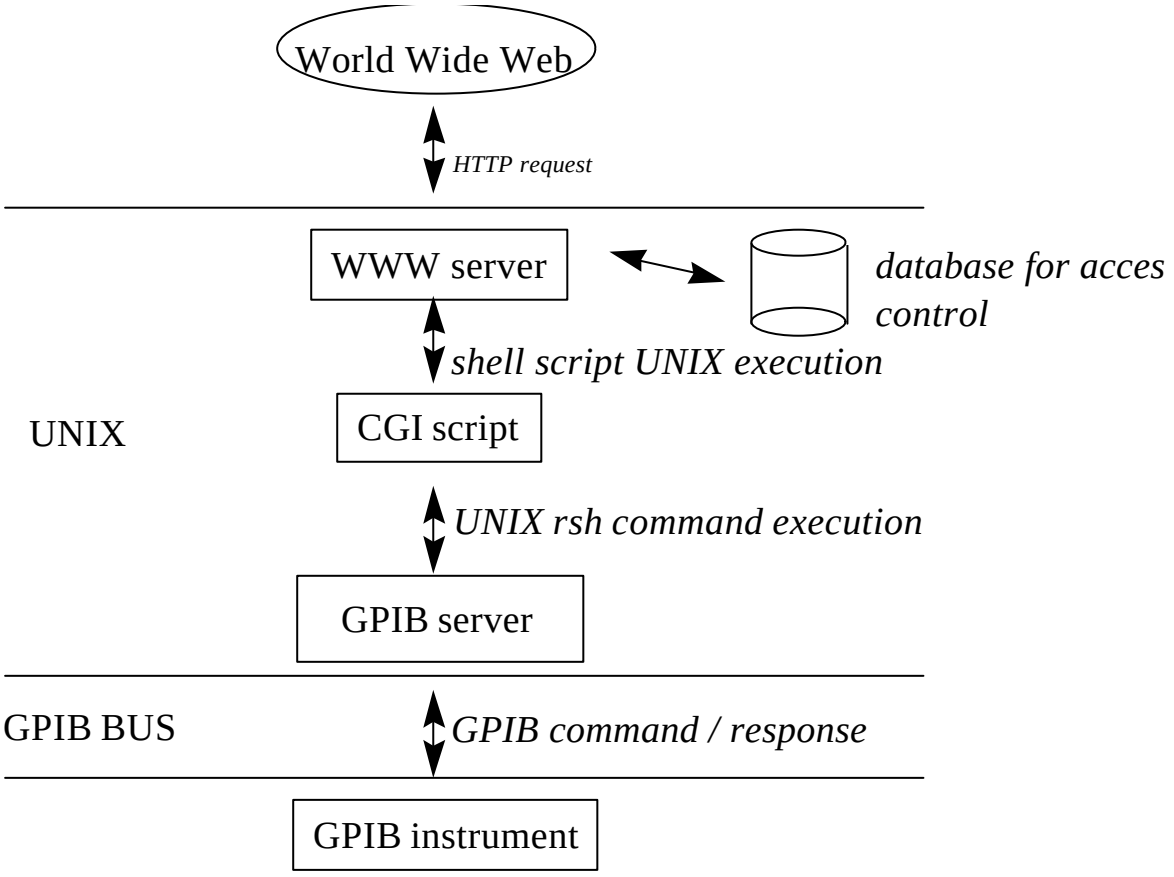
Dans l'exemple de l'Internet, les sites consultés constituent les serveurs, car ils rendent le même service au client qu'est le navigateur. Le service rendu par le serveur est le renvoie de la page du site demandé, et qui sera visualisée par le navigateur une fois téléchargée.

Cette architecture client/serveur permet de séparer complètement le serveur qui gère le stockage des données, du client qui cherche à y accéder. Un protocole commun au client et au serveur est utilisé afin d'établir leur communication et ses normes. Le protocole le plus largement utilisé est le TCP (Transport Control Protocol) qui garantit que les données transmises d'un serveur arriveront sur le client ou vice-versa, et dans l'ordre où elles ont été émises.

Par ailleurs, les serveurs connectés sur Internet sont accessibles grâce à une mnémonique qui est le nom de la machine hôte (par exemple www.yahoo.com ou java.sun.com). Chaque mnémonique est associée à un identifiant numérique représentant la machine hôte qui héberge le serveur : c'est l'adresse IP (Internet Protocol) constituée de 32 bits, notée sous la forme de 4 nombres séparés par des points. Toute machine reliée a Internet possède son adresse IP propre et unique.

D'autre part, plusieurs programmes peuvent tourner en même temps sur une même machine grâce à leurs systèmes d'exploitation. Si plusieurs serveurs sont hébergés sur une même machine, ils vont devoir partager l'accès physique au réseau sur cette machine. Un deuxième niveau d'identification est donc nécessaire pour désigner le programme avec lequel on désire faire la connexion sur une machine donnée. Ainsi, a chaque programme désirant communiquer sur Internet est associe un port unique de 16 bits (de 0 à 65535). Les ports compris entre 0 et 1023 sont réservés à des services particuliers et ne doivent pas être utilisés inconsciemment.

Ainsi, dans le cadre du projet RETWINE, le serveur Aramis du laboratoire IXL héberge les sites qui fournissent les documents Web RETWINE et les programmes de pilotage des appareils de mesure, programmes dont les requêtes se font à partir d'un navigateur client. Le numéro de port choisi est le 8080.



Communication client-serveur.

III.3.2- Scripts CGI

Pour faire appel au driver de l'instrument de mesure et le commander via le bus GPIB, le choix est porté sur les scripts CGI (Common Gateway Interface). Une autre possibilité était d'utiliser les servlets. Mais la solution des scripts CGI a précédemment été retenue pour la mise en place des instruments HP4194A et HP4155A. Etant donné la faible durée de temps pour le développement du projet, il était préférable de reprendre l'emploi du script CGI en l'adaptant au HP8510B.

Les scripts CGI permettent à l'utilisateur de disposer de documents Web réalisés de façon dynamique par un programme du serveur sur requête du navigateur. Le document ainsi produit renferme aussi bien des données extraites du serveur que le résultat du traitement d'un formulaire complété et expédié par l'utilisateur. Le programme peut être écrit en Perl, C, Shell, etc : le choix du langage utilisé n'est pas restreint. Les scripts pour les appareils existants ont été développés en Shell, on gardera donc le même principe.

Les programmes CGI sont lancés par le serveur HTTP sous la demande du client, et s'exécutent sous formes de processus indépendants. Les avantages du CGI sont leur simplicité, leur relative sécurité et le coût moyen de leur exécution. Le programme peut avorter sans endommager le serveur, du moins sur un système d'exploitation en mémoire protégée et à gestion multitâche tel qu'UNIX. De plus l'accès au serveur reste limité pour un programme CGI. Par ailleurs, le coût moyen d'exécution du programme est comparable à celui exigé par l'extraction d'un fichier.

Voici un exemple:

```
(GET/POST) /cgi-bin/mon_executable.sh?parametre
```

Cette ligne, appelée requête, est acheminée vers le serveur avec l'une ou l'autre méthode GET ou POST sous la forme d'un flux. Le serveur reconnaît au chemin `/cgi-bin` qu'il s'agit d'un programme CGI, et exécute le programme mon_executable.sh avec les entrées à traiter.

De plus amples détails seront donnés plus tard au sujet des entêtes du flux que le navigateur transmet au serveur.

IV- Réalisation

IV.1- Le serveur

IV.1.1- Driver C

Le driver utilisé est une surcouche de celui de National Instruments Corporation : le NI-488.2M Driver pour des stations Sparc. Les fonctions pour communiquer avec l'appareil via le bus GPIB sont celles fournies par la bibliothèque.

La communication se fait tout d'abord par :
- initialiser le bus GPIB,
- identifier la carte IEEE,
- identifier l'appareil en lui donnant une adresse GPIB: l'adresse 13 est attribuée au HP8510B.

Ces opérations ont été effectuées dans int init(), fonctions qui retourne le numéro identifiant l'appareil. Afin de debugger le pilotage du HP8510B, l'instruction "DEBUON;" est envoyé (fonction void start(int ud)): les commandes reçues par l'appareil sont alors affichées sur son écran.

Les commandes reçues sous forme de char* par le driver sont envoyées à l'instrument par la fonction GPIB :

```
void ibwrt(int identifiant_du_HP, char *commande, int longueur_du_char*)
```

Les réponses venant de l'appareil sont récupérées par :

```
void ibrd (int identifiant_du_HP, char *reponse, int longueur_du_char*)
```

Toutes commandes sont envoyées directement au HP8510B, et des mesures ne sont effectuées qu'après "SING;" ou "REST;" grâce à la commande "FORM4;OUTPFORM;" qui donne en retour les résultats sous forme ASCII. Les mesures sont ensuite stockées dans des fichiers.

Un message d'erreur est signalé pour tout problème rencontré dans le bus GPIB grâce aux variables ibsta et ERR.

La fonction void data_tofile(int ud, char *answer) génère trois fichiers de résultat contenant chacune une en-tête de description et les valeurs mesurées. Pourquoi trois?

Une applet de plotter déjà conçue pour traiter les mesures récupère deux colonnes de données (x,y) et construit la courbe correspondante y = f(x). Or le HP8510B ne donne pas les valeurs de fréquence lui-même: il donne les valeurs de module (ou de phase, de réél ou autre) sans donner les fréquences correspondantes. Celles-ci sont donc calculées par la fonction data_tofile et sont ensuite affichées dans le fichier qui sera utilisé par le plotter pour tracer la courbe.

Prenons l'exemple d'une mesure de module dans une bande de fréquence. Le HP8510B fournit les valeurs X = module et Y= 0 qui vont être stockées dans trois fichiers différents. Ainsi, le fichier HP8510B_mesure.tab contiendra deux colonnes de valeurs (fréquence F, module X), le fichier HP8510B_mesureXY.tab deux colonnes de valeurs (X,Y), le fichier HP8510B_mesureFXY.tab trois colonnes de valeurs (F,X,Y). Pour l'instant, seul le fichier HP8510B_mesure.tab est utilisé par le plotter.

Les fréquences sont calculées grâce au nombre de points de mesure demandé par l'utilisateur. Par défaut, la fréquence de début de mesure est de 45MHz, celle de fin est de 26.8GHz, et le nombre de points est de 201 (d'après l'état d'initialisation du HP8510B).

Des changements imminents sur le tracé des courbes seront effectués grâce à la collaboration de l'équipe universitaire allemande.

IV.1.2- Serveur HTTP

Le serveur Aramis du laboratoire de l'IXL est un serveur HTTP (Hypertext Transfert Protocol). Ce protocole HTTP définie les modalités de communication entre serveurs Web et clients en mettant en œuvre le protocole TCP/IP et s'appuie sur deux normes MIME (Multipurpose Internet Mail Extensions) et HTML (Hypertext Markup Language). La première norme permet d'encoder des données de types (son ou texte par exemple) avant transmission grâce à une connexion ASCII 7 bits. La deuxième norme permet à l'utilisateur de décrire la valeur sémantique de données textuelles sans toutefois agir sur le formatage qui incombe à l'outil de navigation. La version HTTP actuellement utilisée est le 1.0.

Quatre étapes de communication constituent un échange client/serveur:

- établissement de la connexion: le client met en place une connexion TCP avec le serveur sur le port 80 par défaut ou le port indiqué dans l'URL (dans notre cas pour Aramis, c'est 8080).

- Emission d'une requête: le client transmet un message au serveur en vue d'extraire le document correspondant à l'URL spécifié. Le format des données transmises se divise en une en-tête et un corps. L'en-tête se décompose en un mot-clé (GET, HEAD, POST, PUT), suivi de l'URL du document abrité dans un fichier du serveur, et ensuite la version du protocole reconnu par le client. L'en-tête permet de définir entre autre, le type MIME du corps et sa longueur. Deux paires de retour chariots/changements de ligne (\r\n\r\n en Java) concluent l'entête.

  Voici l'exemple d'une requête d'un document Web :

```
GET /index.html HTTP/1.0
Accept : text/html
Accept : text/plain
User-Agent : Lynx/2.4 libwww/2.1.4
```

  Cette requête consiste à demander le contenu d'une URL au serveur. Dans notre exemple, l'URL est relative. Le mot-clé Accept fournit les types de données acceptés par le client.

Le mot-clé User-Agent indique au serveur le type de navigateur utilisé pour permettre l'envoi de fichiers spécialement optimisés pour le client.

Prenons l'exemple maintenant de l'envoi par le client d'octets "STAR 90 MHz;" avec une requête sur le script CGI:

```
POST /cgi-bin/HP8510B_perform.sh HTTP/1.0
Content-type : application/octet-stream
Content-length : 14

"STAR 90 MHz;"
```

La méthode POST permet de transmettre au serveur des données contenues dans le corps du message. Ces données peuvent être de tout type et de grande taille. Dans l'en-tête sont définis le type de format MIME envoyé ainsi que la taille des données.

- Envoi de la réponse:

La réponse que le client reçoit du serveur commence par un code, suivi d'un en-tête MIME, d'une ligne vierge, puis du document requis ou d'un message d'erreur. En reprenant l'exemple de la requête pour le document Web index.html, on obtient une réponse du type :

```
HTTP/1.0 200 OK
Server : Apache/1.0.5
MIME-Version : 1.0
Content-type : text/html
Content-length :97

<HTML>
<HEAD>
Exemple de document html
</TITLE>
</HEAD>
<BODY>
Corps du document
</BODY>
<HTML/>
```

Dans l'en-tête sont spécifiés le protocole utilisé, la réussite (ou l'échec) du traitement de la requête, le logiciel du serveur, la version MIME, le type des données et leur taille. Vient ensuite le corps du message.

- Clôture de la connexion : la connexion peut être close sur l'initiative du client ou du serveur ou des deux. Pour chaque requête est ouverte une connexion réseau distincte, le serveur ne conserve aucune trace des précédentes connexions d'un client ni du résultat.

IV.1.3- Script shell

Lorsqu'une requête est effectuée pour appeler le script CGI, le serveur exécute celui-ci et lui passe par l'entrée standard les arguments du client. Un mécanisme de protection est alors mis en œuvre pour réserver les ressources matérielles du client. Celui-ci est identifié par l'adresse IP de sa machine. Pratiquement, un fichier est créé dont le nom comporte un préfixe déterminant l'appareil et un suffixe identifiant l'utilisateur. Il est alors simple de tester l'existence d'un fichier préfixé du nom de l'appareil si celui-ci est utilisé.

Un watchdog (chien de garde) est utilisé pour effacer régulièrement les verrous: un processus est lancé en tâche de fond, attend une période fixe déterminant le temps alloué au client avant libération de ressources pour finalement réinitialiser le verrou. Ainsi, si le client n'effectue pas de requête dans le temps réservé, l'appareil est libéré. Un message (USED ou NOT_USED) informe le client s'il peut accéder à l'appareil. Dans le cas affirmatif, i.e. dans le cas ou les ressources sont libres, le script invoque par rsh l'execution du driver de l'appareil en passant les arguments par l'entrée standard.

Les fichiers générés par le driver sont ensuite déplacés dans le répertoire adéquat afin que l'utilisateur puisse visualiser les résultats via son navigateur, sous forme de colonnes de mesures ou de courbes.

IV.1.4- Chargement de l'applet

Le chargement d'une applet et son exécution commencent lorsque le navigateur rencontre par exemple la balise :

```
<APPLET CODEBASE="http://aramis.ixl.u-bordeaux.fr:8080/pages-
web" CODE="HP8510B.class" WIDTH = "200" HEIGHT="300">
```

Le navigateur réserve une surface 200x300 pixels dans le document et se connecte par défaut sur le port 80 du serveur à moins que l'URL spécifié ne fournisse un autre port. Si `CODEBASE` n'apparaît pas, le socket utilisé est celui du serveur ayant expédié le document HTML. Le navigateur requiert du serveur Web l'extraction du fichier .class comme pour un fichier ordinaire.

Le serveur transmet alors une en-tête MIME et les données du fichier .class. Le navigateur réceptionne les données et les stocke dans un tableau. Le vérificateur de code analyse les données reçues à la recherche d'éventuelles erreurs. Si rien d'anormal n'est détecté, les données sont converties en classe Java grâce aux méthodes defineClass() et loadClass() de l'instance ClassLoader courante. La classe téléchargée est ensuite placée dans un espace indiquant sa provenance. Ainsi, les classes de meme nom provenant de sites différents ne sont pas confondus. Quand une classe fait référence à une autre classe, l'interpréteur Java recherche cette dernière en premier lieu dans le CLASSPATH de l'utilisateur. Une fois localisée, cette classe est créée à partir du fichier .class du disque dur de l'utilisateur.

Si la recherche ne donne rien, le navigateur contacte à nouveau le site de provenance de la première classe puis télécharge le fichier .class requis. Le navigateur procède ainsi pour la nouvelle classe et pour toute autre classe obtenue via l'Internet. L'exception ClassNotFoundException est signalée pour toute anomalie.

Pour raison de sécurité, les accès directs au fichiers locaux sont interdits. Des URL doivent être utilisés pour accéder à des images ou autres fichiers.

IV.2- L'interface client

IV.2.1- Principe

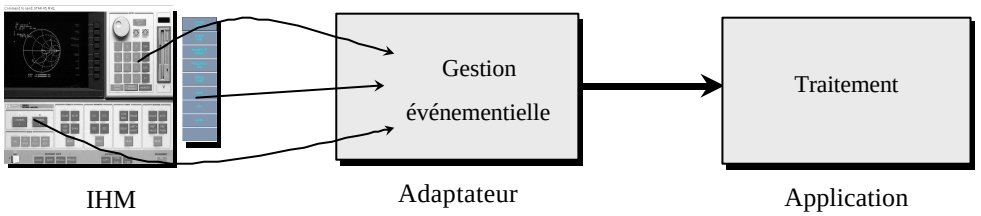
Le développement de l'interface s'est fait en mettant l'accent sur la modularité afin que le code puisse évoluer facilement et permettre une séparation nette entre l'interface graphique et le traitement des événements détectés. On distingue donc trois parties :

- **L'interface homme machine (IHM)** qui concerne les éléments graphiques de l'interface. Là aussi, l'aspect modulaire favorisera sa maintenance. Par ailleurs, la réalisation de cet interface s'est voulue rester la plus proche de l'instrument réel.

- **L'adaptateur** prend en charge la gestion des événements détectés sur l'IHM et aiguille l'application vers le traitement adéquat.

- **L'application** intègre toutes les méthodes susceptibles d'être sollicités pour traiter un événement sur l'IHM transmis par l'adaptateur.

Cette organisation du code permet de modifier aisément des fonctionnalités et permet aussi une gestion efficace des événements. Le schéma suivant illustre ce découpage du code :



Implémentation de l'interface.

IV.2.2- L'interface homme machine IHM

IV.2.2.1- Le découpage de l'interface

L'interface graphique se décompose en une fenêtre affichant la face avant de l'appareil, une zone de message au-dessus de cette image, et une fenêtre pour les menus des softkeys. Ce découpage est spécifié dans la classe *InstrumentGUI* qui constitue un squelette de l'interface:

- InstrumentPanel,

- InstrumentFrame,

- InstrumentLabel.

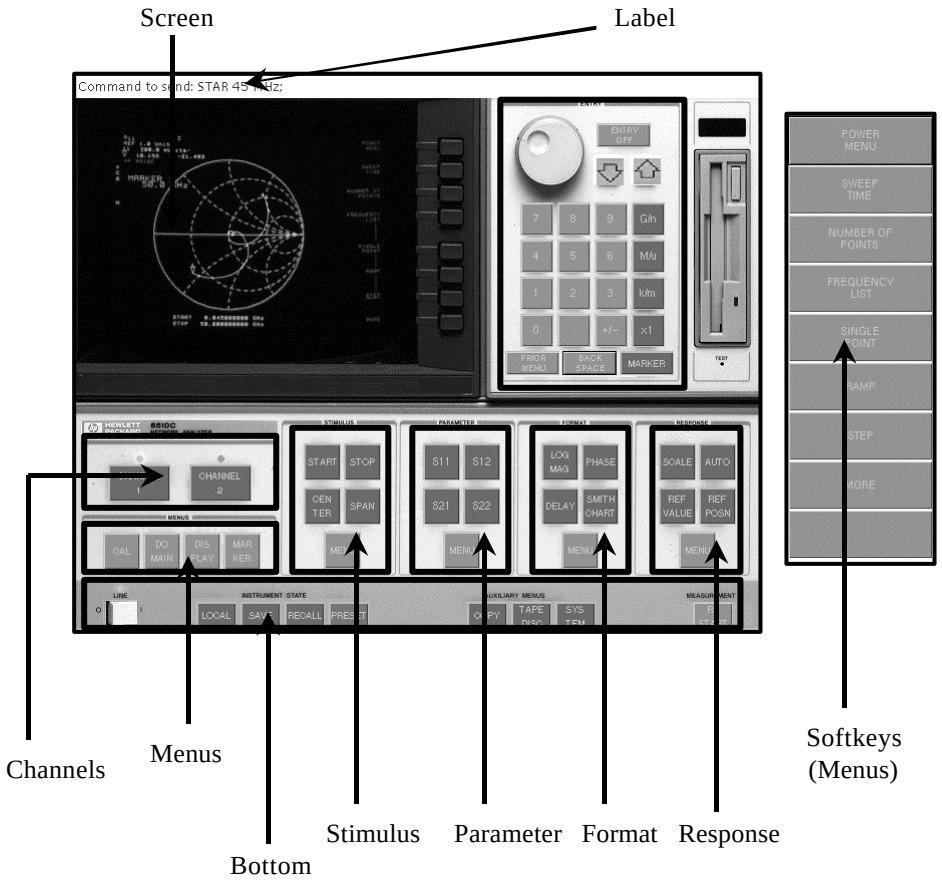
A cela s'ajoutent des objets *SwapImage* pour l'écran et les *LEDs* (*InstrumentScreen, InstrumentChanLED, InstrumentPowLED*) qui doivent s'allumer ou s'éteindre selon les instructions. La classe *HP_Interface* dérive de *InstrumentGUI,* et c'est dans cette classe que se construit l'ensemble de l'interface.

IV.2.2.2- La face avant de l'appareil

La fenêtre affichant la face avant de l'appareil (*InstrumentPanel*) a été découpée en plusieurs sections qui ont toutes été conçues suivant le même schéma. Pour chaque bloc, on distingue:

- Le *layout* des composants (coordonnées sur l'image)
- Le paramétrage des composants (taille, couleurs, polices,…)
- L'ajout des *listeners* destinés à détecter les événements.
- Le placement des composants dans le panel *instrumentPanel*.

Les différents blocs sont tous inclus dans le panel *instrumentPanel* de l'interface *HP_Interface* qui est instanciée dans l'applet. On voit ainsi apparaître la structure hiérarchique du code, ce qui est illustré dans la figure suivante :



Face avant de l'appareil.

IV.2.2.3- Palette, ImageResource et Constant

Ces éléments permettent aux différents blocs d'accéder aisément à des données sans avoir à les passer par arguments. Ainsi, au lieu de redéfinir des couleurs dans chaque objet de l'interface, *Palette* regroupe toutes les couleurs utilisées et modifier la palette de couleurs consiste simplement à modifier le fichier Palette. De même, *ImageResource* contient les images utilisées pour notre application (image de l'instrument, image pour certains boutons, leds,…). Et l'interface *Constants* se charge comme son nom l'indique des constantes utilisées dans la méthode *sendToGPIB* (transmettant les instructions à l'instrument).

IV.2.2.3- PanelImage

Cette classe est dérivée de la classe Panel de l'AWT. Sa méthode *paint* permet de faire afficher l'image de la face avant du HP8510B.

IV.2.2.5- CmdButton et ImageButton

CmdButton et *ImageButton* sont tout simplement des boutons avec des attributs supplémentaires. *CmdButton* intègre les commandes à envoyer à l'instrument et éventuellement un ordre de branchement sur un menu pour les *softkeys*. *ImageButton* se comporte comme *un CmdButton* mais on peut lui poser une image qui se décale lorsqu'on clique dessus. *ImageButton2* repose sur le même principe que *ImageButton* mais diffère de celui-ci par le fait que ce sont deux images qui se succèdent dans l'affichage suivant le clique de la souris.

IV.2.2.6- SwapImage

Cette classe définie des objets affichant une image (parmi deux ou trois) suivant le choix imposé. Elle est utilisée pour la visualisation de l'écran et des LEDs.

IV.2.2.7- Le menu des softkeys

Les menus de l'instrument sont affichés dans une fenêtre séparée. Ce sont des *Panels* implémentées par un *CardLayout* dans un *Frame* (appelé *InstrumentFrame*). Les menus sont instanciés et ajoutés à l'interface dans *HP_Interface*.

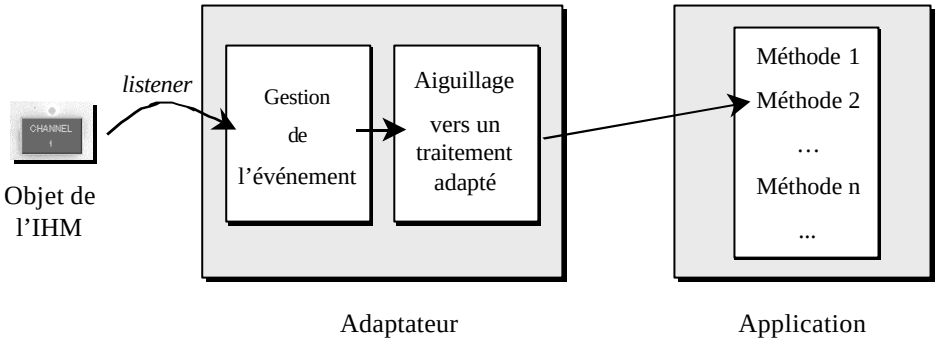
Ainsi, le *Frame* n'affichera que le menu appelé soit par un bouton de la face avant de l'appareil, soit par un bouton d'un autre menu. Chaque menu possède neuf *CmdButtons*. A chaque bouton actif est associé son *Listener*. Un bouton "PRIOR MENU" permet de remonter dans la hiérarchie des menus si une hiérarchie existe.

IV.2.2.8- Le label affichant des messages

Cette zone d'affichage (*InstrumentLabel*) permet d'informer l'utilisateur des commandes qu'il va envoyer, des commandes qu'il envoie et des erreurs possibles qu'il peut rencontrer lors de la communication client/serveur.

IV.2.3- Gestion et traitement des événements

Comme nous l'avons déjà évoqué, la gestion des événements est prise en charge par l'adaptateur et la classe *Application* est munie des méthodes nécessaires au traitement de l'événement. Ainsi, un composant graphique de l'IHM possède un *listener* qui détecte un événement et invoque l'adaptateur. Celui-ci identifie l'objet concerné et sélectionne la méthode de l'application répondant à l'événement survenu. La figure suivante illustre ce principe qui a l'avantage d'être simple et généralisable:



Traitement des évènements.

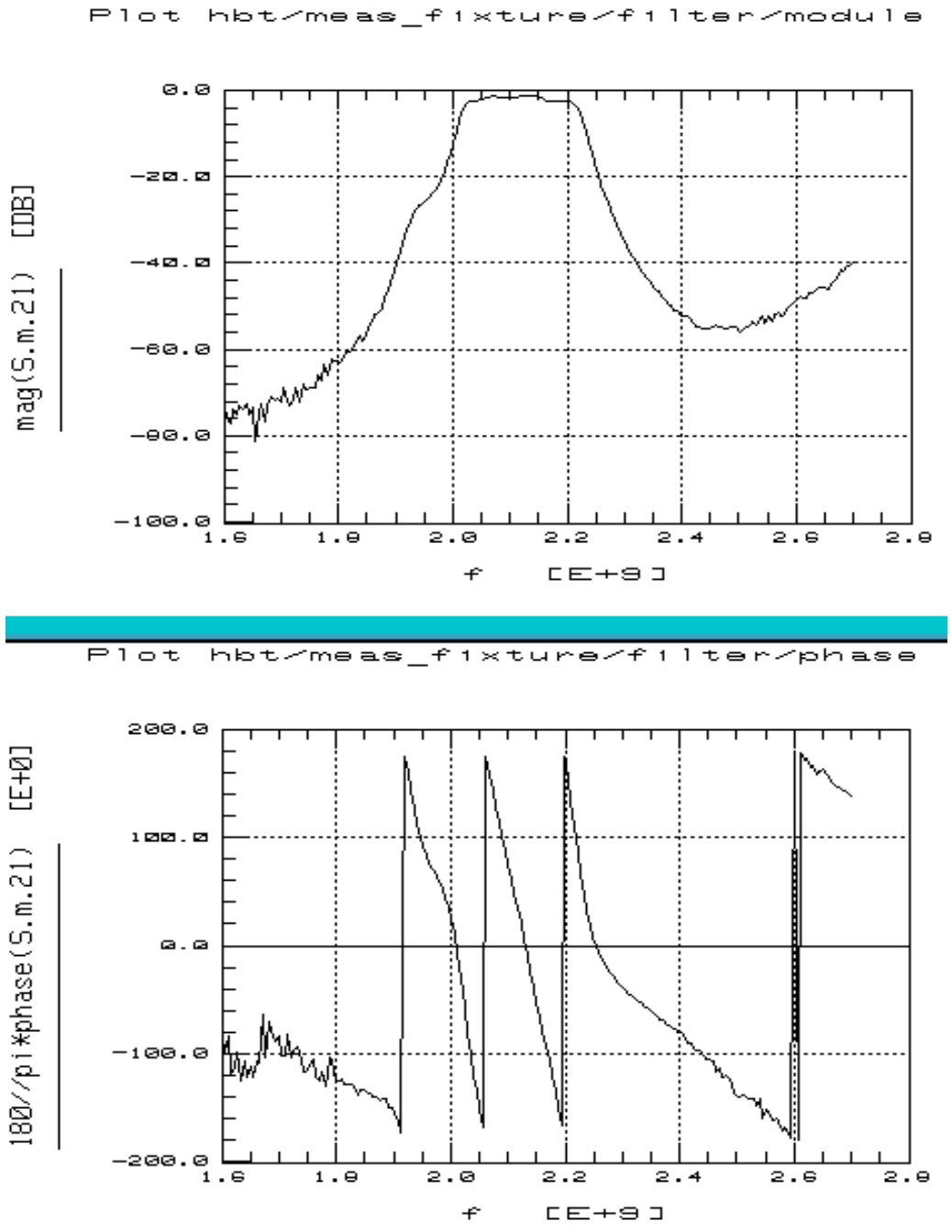
IV.3- Organisation des fichiers sources

Le serveur est structuré comme suit:

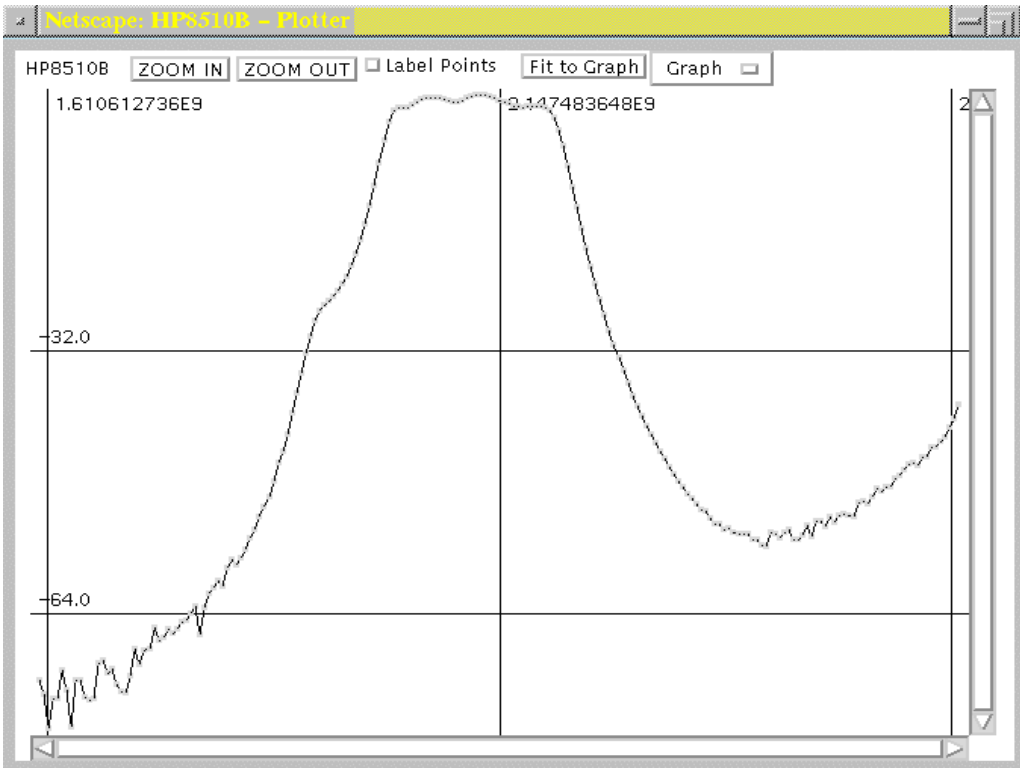
- Répertoire *pages-web* où sont contenus les fichiers *.html* et les fichiers *.class*.
- Répertoire *cgi-bin* où sont contenus les fichiers en *Shell script* et les exécutables.
- Répertoire *src* où sont contenus les fichiers *.java*, *.c*, et la documentation en *javadoc*. Les fichiers .java sont classés dans les répertoires *util*, *panel* et *menu* selon leur *package*.

V- Tests et Essais

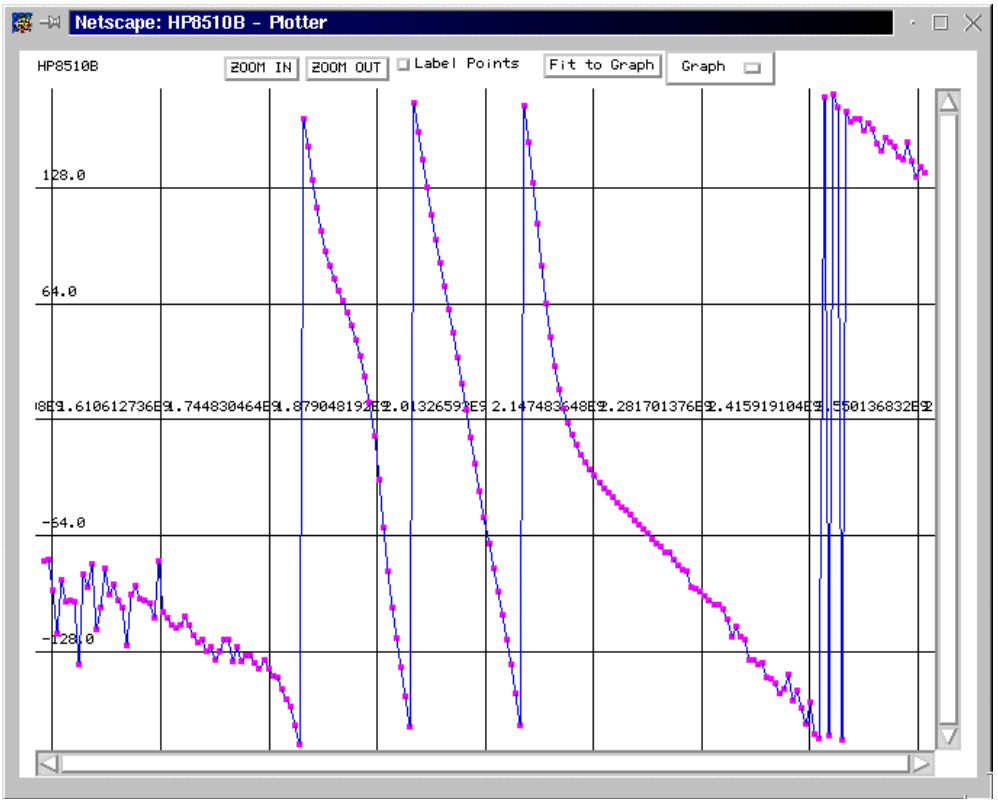
Des mesures sur un filtre passe-bande ont été réalisées afin de valider le projet. Les résultats obtenues lors d'un pilotage test de l'interface Java concorde parfaitement à ceux obtenus grâce à un logiciel (Iccap) pilotant le HP8510B. Sont données ci-après pour comparaison les courbes module et phase du paramètre S21 du filtre.



Paramètre S21 du filtre passe-bande, de 1.6GHz à 2,7GHz
avec le logiciel Iccap.



Paramètre S21 (module) du filtre,
Tracé du plotter à partir des mesures via le Web.



Paramètre S21 (phase) du filtre,
Tracé du plotter à partir des mesures via le Web.

VI-Manuel d'utilisation

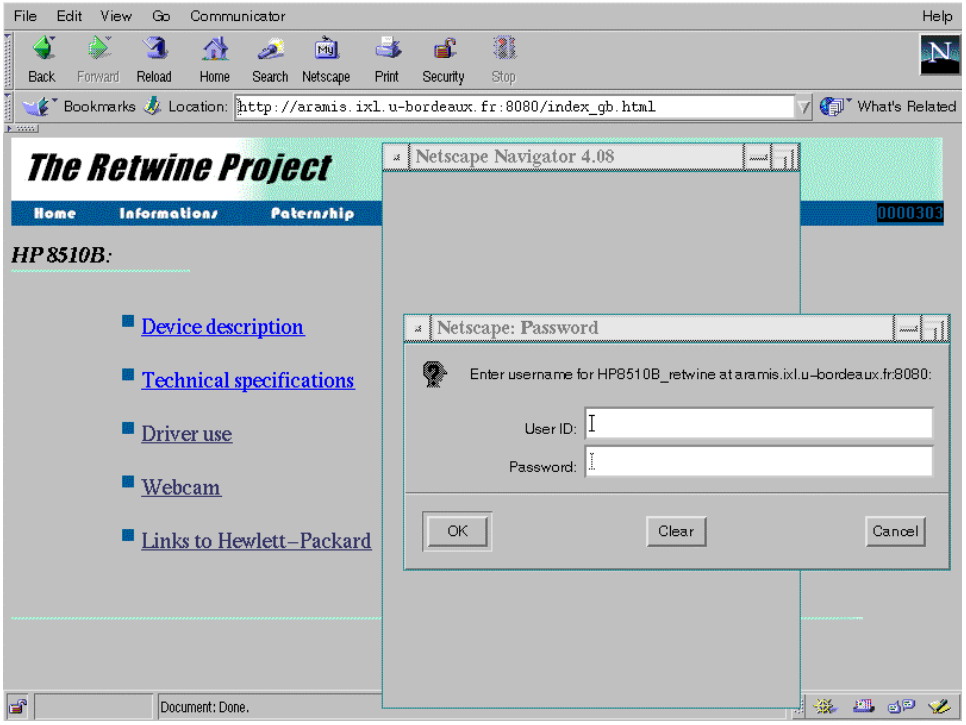
Cette partie donne un exemple d'utilisation de l'interface Java du HP8510B. Il est loin d'être exhaustif, et l'utilisateur devra se confronter aux manuels d'utilisation de Hewlett Packard pour l'analyseur de réseau. Un document Web doit être prochainement créé pour diriger l'utilisateur novice à ce genre d'appareils.

L'adresse du serveur Aramis est :

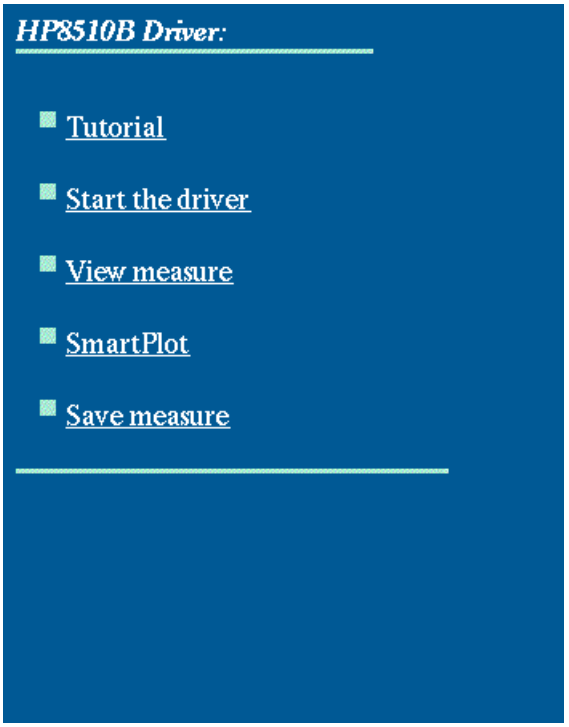
http://aramis.ixl.u-bordeaux.fr:8080

Après avoir demandé les documents dans Instruments, et sélectionné l'appareil HP8510B, une page propose plusieurs choix d'utilisation lié à l'instrument.

Actuellement, la documentation sur l'appareil (device description et technical specification) n'est pas encore prête. "Driver use" permet de piloter l'appareil. Une fenêtre d'accès pour l'utilisation apparaît et invite l'utilisateur à entrer son nom d'identification et son mot de passe (qui auront été préalablement enregistré par un responsable RETWINE).



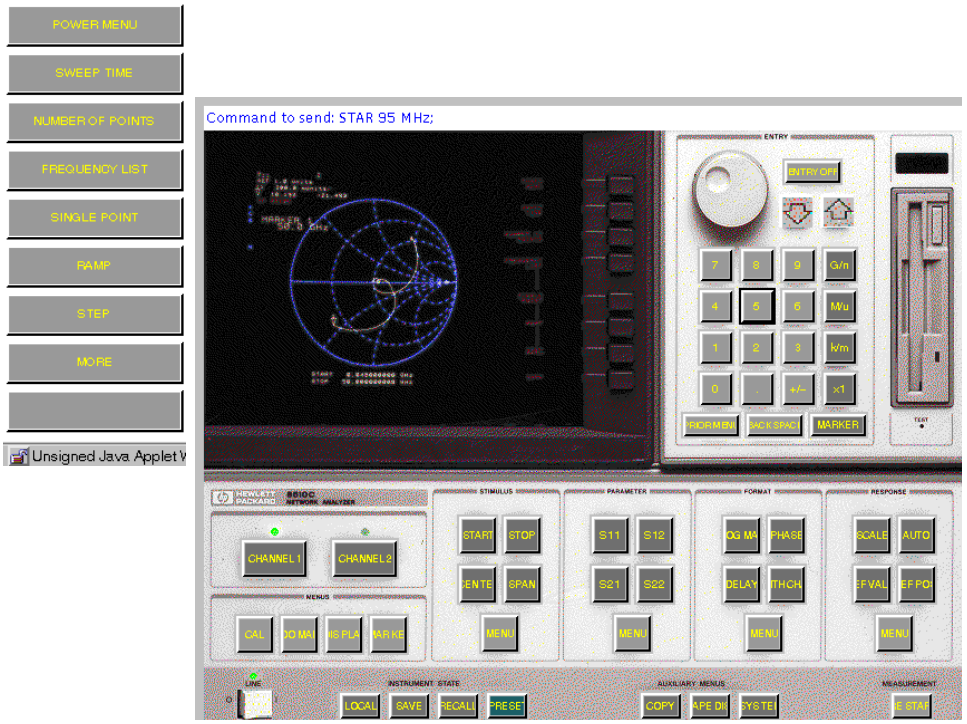
La fenêtre suivante invite l'utilisateur à consulter un guide d'utilisation, à piloter l'appareil, à visualiser les courbes obtenues via un plotter et un "smart plotter", et à visualiser les résultats dans un fichier txt facilement sauvegardable.



Le tutorial n'est pas encore disponible à l'état actuel. Le chargement de l'applet affichant la face avant du HP8510B dure un certain moment: que l'utilisateur prenne son mal en patience…

Un menu apparaît lorsque l'appareil est allumé: il correspond aux touches softkeys. En effet, l'écran étant trop petit pour faire afficher les fonctions de ces touches, la solution retenue est d'utiliser ce menu offrant neuf boutons (dont un certain nombre peut-être inactifs). L'utilisateur peut interagir sur l'interface seulement avec la souris. Les événements sur le clavier n'ont pas (encore) été pris en compte. On remarquera que le clavier numérique de l'interface offre une fonction 'back space" qui permet d'effacer les chiffres saisis par erreur.

Donnons ici un exemple pour régler les fréquences de début de mesure. Comme sur l'analyseur de réseau réel, il suffit de sélectionner 'Start" puis de saisir une fréquence donnée et de valider cette commande par l'unité voulue (GHz, MHz, KHz, Hz (x1)).



On donnera ici un exemple simple d'utilisation de l'interface. Le quadripôle testé est un filtre passe-bande préalablement connecté à l'analyseur de réseau, l'entrée au port 1, et la sortie au port 2.

Après avoir allumé l'appareil virtuel, il faut sélectionner:

- le canal adéquat (le canal 1 est choisi par défaut). Dans notre exemple, le channel 1 est choisi.

- le calibrage associé au type de mesure à effectuer:
  sur la face avant de l'appareil: Menus>Cal; puis dans le menu des softkeys:

  Calibration ON;
  ensuite le numéro de calibrage à rappeler (dans notre exemple, "Cal Set 1").

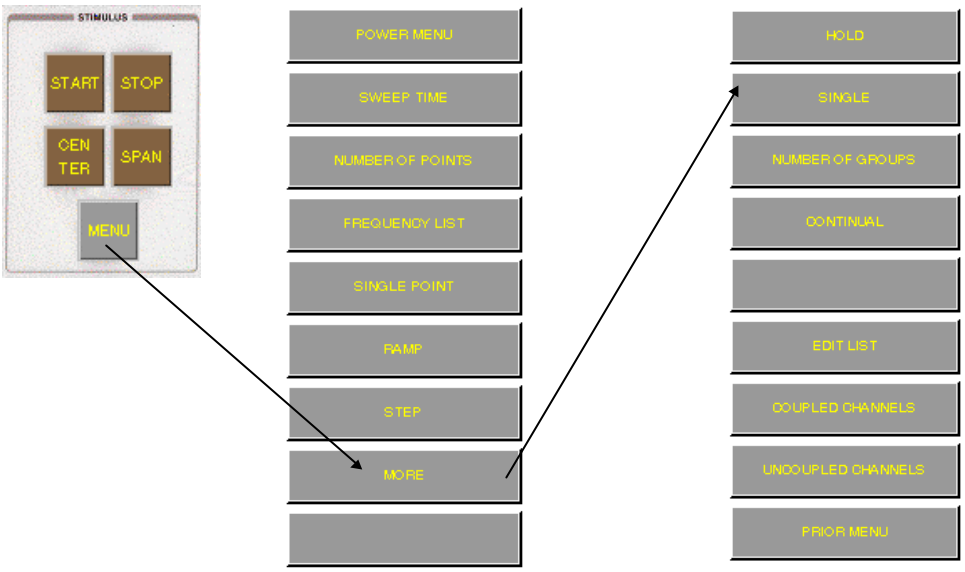
Actuellement, le calibrage ne peut se faire que localement; il faudra donc que l'utilisateur définisse le calibrage nécessaire afin qu'une personne du laboratoire s'en charge.

- le nombre de points de mesure (201 points par défaut): stimulus Menu>number of points.

- le paramètre S à mesurer (paramètre S21 par exemple).

- la courbe à obtenir (module sur échelle logarithmique, phase, abaque de Smith). Dans notre exemple, on visualisera la phase.

- la demande de mesure: soit le bouton "RESTART", soit le bouton "SINGLE" dans le menu "Stimulus>More".

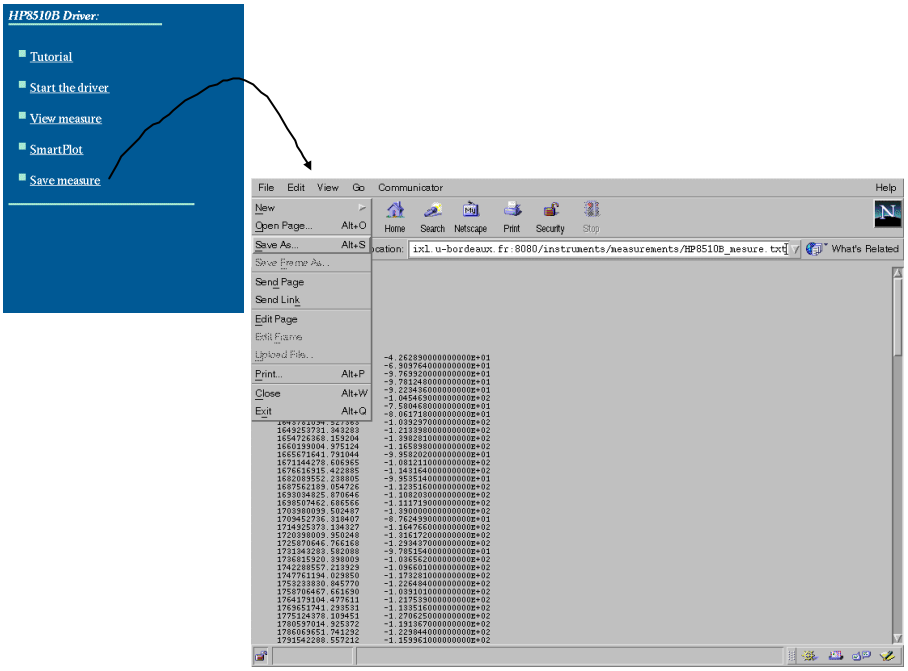


Ou encore

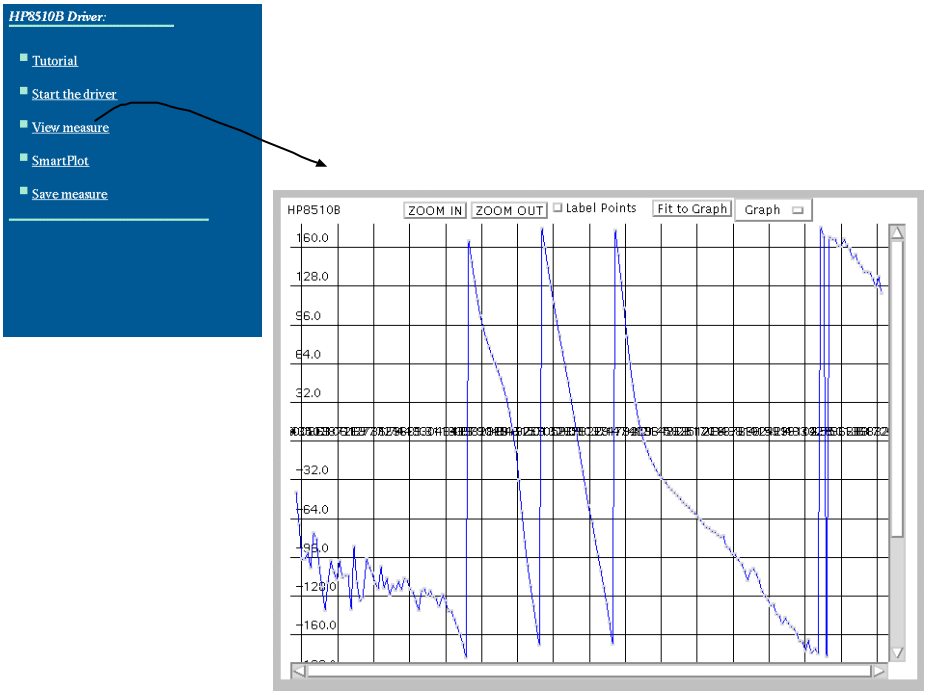
Des boutons n'ont pas de fonctionnalité attribuée parce qu'ils n'offrent pas d'utilité pour un pilotage à distance et avec un tel interface. Par exemple, le marker, la mise au point de l'échelle pour l'affichage sur l'écran de l'appareil (réel), les différents types de calibration, les commandes touchant au lecteur de disquette ou au système de l'appareil, ou les réglages qui font appel à une visualisation sur l'écran ne peuvent être implémentés pour l'instant. Un message dans le label situé au-dessus de la face avant de l'appareil met alors en garde l'utilisateur. Si l'interface évolue dans l'avenir, ces fonctions devront alors être intégrées.

This command cannot be used remotely. Sorry...

Après un message affirmant que la commande de mesure a été envoyée et que l'instrument est de nouveau prêt, les résultats peuvent être sauvegardés avec l'option "save measure".



L'option "view measure" permet d'utiliser le plotter pour afficher les courbes.



Ceci était juste une description succincte pour débuter et de nombreux réglages peuvent toujours être effectués.

VII- Perspectives

Ce travail élaboré au cours de ces trois mois a permis de rajouter un nouvel instrument au parc RETWINE. L'utilisation de cet appareil est satisfaisant mais il reste des améliorations à effectuer.

Il serait souhaitable qu'une description de l'état réel de l'appareil soit visualisable par l'utilisateur. Ceci permettrait une meilleure utilisation de l'analyseur de réseau. Par exemple, cet appareil possède une plage de fréquences d'utilisation de 45MHz à 26.8GHz. Si l'utilisateur par mégarde ou par ignorance saisie une fréquence de début de mesure de 40MHz, le HP8510B réel se met automatiquement à 45MHz. Il faudrait ainsi avertir l'utilisateur de cette erreur. Ainsi, une fenêtre d'affichage de certains paramètres serait utile en guise d'information. Il faudrait pouvoir intégrer :

- les fréquences de début et de fin de mesure,
- le nombre de points de mesure,
- le nombre de mesures à effectuer pour le moyennage (number of groups et affichage de H (Hold)),
- les indications sur les types de mesures
  - average → affichage de A,
  - smoothing → affichage de S,
  - electrical delay → affichage de D,
  - auto delay → affichage de la valeur correspondante.
- Récupérer sur l'interface les messages d'erreur dues au bus GPIB ou à la connexion client/serveur.

D'autres menus peuvent être rajoutés ou activés suivant l'évolution qu'on voudra apporter à l'interface. On peut par exemple afficher un écran dynamique en s'approchant ainsi de plus près à l'instrument réel. Il serait également utile de prévenir par un message que le chargement de l'applet est long afin de faire patienter l'utilisateur.

D'autre part, les documents HTML sur la spécification de l'appareil et le guide d'utilisation (tutorial) doivent être écrits. Des adaptations sur les fichiers de résultats devront également être réalisées afin d'utiliser le SmartPlotter qui vient juste d'être développé par l'équipe de l'université allemande.

Par ailleurs, pour le projet RETWINE en lui-même, il sera préférable de développer en Java 1.2 les prochains instruments à rajouter au parc si les navigateurs supportent cette version. En outre, un PC sous environnement LINUX devrait prochainement regrouper toutes les fonctions du serveur (carte IEEE et serveur Web). Une Webcam devrait également y être installée.

VIII- Conclusion

L'analyseur de réseau HP8510B est maintenant intégré au parc RETWINE et disponible via le WWW. Il faudra cependant apporter des améliorations comme celles que nous avons proposées dans ce rapport. Le site doit aussi être complété avec des pages décrivant l'instrument et un tutorlial.

Ce projet, intéressant et complet, fait appel à des connaissances et des technologies demandées actuellement dans le monde de la communication. Tous les aspects de l'installation de l'instrument sur le parc RETWINE ont été abordé depuis le pilotage jusqu'à la récupération et le visualisation des mesures. Il a aussi été très formateur sur le plan technique dans le cadre d'un projet de fin d'études. Par ailleurs, la conduite de ce travail nous a permis d'avoir une certaine autonomie dans l'acquisition de nouvelles connaissances (apprentissage du langage Java ou de l'architecture client/serveur), condition essentielle et nécessaire pour tout ingénieur.

Le projet RETWINE présente un intérêt certain et a tout pour séduire d'éventuels partenaires et se développer rapidement et des projets analogues de pilotage à distance de matériels très divers devraient voir le jour.

Abréviations

CGI	Commun Gateway Interface
GPIB	General Purpose Instrumentation Bus
HTML	HyperText Markup Language
HTTP	Hypertext Transfert Protocol
IP	Internet Protocol
JDK	Java Development Kit
MIME	Multipurpose Internet Mail Extensions
RETWINE	REmoTe Web Instrumentation Network
TCP	Transport Control Protocol

Bibliographie

Programmation Réseau avec Java
Elliotte Rusty Harold, traduction de Manuel Makarévitch
Editions O'REILLY, 1997.

Java client-serveur, JDK 1.1, Java Beans, JDBC, Corba/RMI, Marimba Castanet
Cédric Nicolas, Christophe Avare, Frédéric Najman
Collection Fi System, Editions Eyrolles, 1998.

HTML et la programmation de serveurs Web
Philippe Chaléat, Daniel Charnay
Editions Eyrolles, 1996.

Java par la pratique
Patrick Niemeyer & Joshua Peck, traduction de Eric Dumas
Editions O'Reilly International Thomson, 1996.

Webgraphie

Références sur HTTP: http://www.eisti.fr/eistiweb/docs/normes/

Cours de Java : http://www.eteks.com
http://athena.alcyonis.fr/

Références sur Java : http://java.sun.com

Tutorial de Sun : http://java.sun.com/docs/books/tutorial/

Site de l'IXL : http://www.ixl.u-bordeaux.fr

Site de Retwine : http://aramis.ixl.u-bordeaux.fr :8080/

Site de Hewlett Packard: http://www.hp.com

Annexes