PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Conception d'une plate-forme de télécommunications pour desservir des laboratoires virtuels distribués
Auteur: Author:	Marthe Kassouf
Date:	1999
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Kassouf, M. (1999). Conception d'une plate-forme de télécommunications pour desservir des laboratoires virtuels distribués [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8823/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/8823/
Directeurs de recherche: Advisors:	Jean Conan, & Samuel Pierre
Programme: Program:	Génie électrique

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

UNIVERSITÉ DE MONTRÉAL

CONCEPTION D'UNE PLATE-FORME DE TÉLÉCOMMUNICATIONS POUR DESSERVIR DES LABORATOIRES VIRTUELS DISTRIBUÉS

MARTHE KASSOUF
DÉPARTEMENT DE GÉNIE ÉLECTRIQUE ET DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉLECTRIQUE)
JUIN 1999

©Marthe Kassouf, 1999.

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre reference

Our file Notre reference

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-48858-6

Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

CONCEPTION D'UNE PLATE-FORME DE TÉLÉCOMMUNICATIONS POUR DESSERVIR DES LABORATOIRES VIRTUELS DISTRIBUÉS

présenté par: KASSOUF Marthe

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. SAVARD Pierre, Ph.D., président

M. CONAN Jean, Ph.D., membre et directeur de recherche

M. PIERRE Samuel, Ph.D., membre et codirecteur de recherche

Mme. AIMEUR Esma, Ph.D., membre

À mes parents . . .

REMERCIEMENTS

Je tiens à remercier mon directeur de recherche, Monsieur Jean Conan, qui m'a fourni un support moral remarquable et qui m'a donné l'opportunité de conduire mes travaux de recherche dans ce projet de laboratoires virtuels. Mon effort n'aurait pas pu être fructueux sans l'encadrement judicieux de mon codirecteur de recherche, Monsieur Samuel Pierre, que je remercie sincèrement pour sa patience, son encouragement perpétuel, le choix du sujet et l'aide financière. J'exprime ma véritable reconnaissance au Centre de recherche en informatique cognitive et environnements de formation (LICEF) où j'ai effectué une partie de la recherche conduisant à ce mémoire.

Mes remerciements s'adressent également à tous mes collègues dans les différentes équipes du projet pour leurs conseils et leur collaboration. Finalement, je dois un grand merci à mes tantes ainsi qu'à tous les membres de leur famille pour l'hospitalité avec laquelle ils m'ont accueillie durant ces deux longues années de Maîtrise.

RÉSUMÉ

La convergence des domaines de l'informatique et des communications a donné lieu à tout un éventail d'applications réparties de plus en plus assujetties à la distribution de ressources. La notion de systèmes d'objets distribués semble être favorablement accueillie au sein de la communauté scientifique. Dans ce genre de système, des objets dispersés exploitent des mécanismes de communications divers pour solliciter les services l'un de l'autre. Avec la multiplication des technologies impliquées, une variété d'applications innovatrices voit le jour, dont le télé-enseignement mettant à contribution le concept de campus virtuel.

Un campus virtuel vise à promouvoir l'enseignement à distance dans tous ses aspects, y compris ceux d'expérimentation en laboratoire. On parle alors de laboratoire virtuel pour désigner un environnement distribué permettant de réaliser par simulation ou par télé-manipulation certaines activités pratiques d'apprentissage à partir de machines clientes distantes.

L'objectif principal de ce mémoire est de concevoir une plate-forme de télécommunications assurant l'accès à distance à des laboratoires virtuels. Deux phases sont nécessaires afin d'atteindre ce but : l'élaboration d'un modèle conceptuel et la réalisation ou implémentation informatique de ce modèle.

Selon le modèle conceptuel proposé, un laboratoire virtuel est considéré comme un environnement informatique complexe intégrant deux catégories d'objets. La première contient des éléments génériques, communément partagés par plusieurs laboratoires et constituant des outils et fonctionnalités de base. La deuxième catégorie d'objets fait référence aux outils et fonctionnalités spécifiques à un laboratoire donné. L'intégration

des outils spécifiques et de base ainsi que la transmission de l'ensemble à l'utilisateur final représentent les deux tâches essentielles de la plate-forme de télécommunications.

Le modèle conceptuel que nous avons proposé est en fait une architecture à trois couches comprenant :

- la couche d'adaptation aux outils et fonctionnalités spécifiques qui est la plus haute de l'entité conceptuelle et permet à un objet générique de s'ajuster aux exigences spécifiques d'un laboratoire particulier ;
- la couche d'outils et de fonctionnalités de base, où se réalise l'implémentation des objets génériques partageables ;
- la couche d'adaptation aux réseaux d'utilisateurs, le bas niveau de la plate-forme, qui assure l'interopérabilité entre les différentes architectures de communications dont disposent les clients (TCP/IP, ATM, etc).

Pour ce qui est de l'implémentation, la plate-forme de télécommunications essaie de respecter autant que possible les spécifications élaborées sur le plan conceptuel. Une approche orientée objet a été adoptée pour réaliser les trois couches. Pour implémenter les deux premières, le langage de programmation Java s'est révélé un des meilleurs candidats en égard aux requis de la plate-forme. Quant à la couche d'adaptation aux réseaux, nous avons opté pour l'architecture CORBA et l'interface HTTP qui nous ont offert un moyen satisfaisant pour maintenir plus ouvert l'environnement des laboratoires virtuels. Toutefois, le modèle conceptuel proposé laisse la liberté à n'importe quel outil générique de disposer de son propre mécanisme d'accessibilité à d'autres composants informatiques, y compris une base de données le cas échéant. Ce modèle présente plusieurs avantages, tels que la portabilité, la modularité et la réutilisabilité. Des tests conduits sur différents supports et réseaux de communications hétérogènes confirment la flexibilité de cette implémentation.

Le mérite principal de ce mémoire est d'avoir établi la preuve de faisabilité d'une plate-forme de télécommunications capable de supporter des laboratoires virtuels distribués. Ces derniers sont destinés au télé-apprentissage et rendent possible la réalisation en réseau d'activités pédagogiques pratiques, en tenant compte des aspects d'interopérabilité des réseaux d'accès. Les travaux futurs devraient porter sur la conception d'une méthodologie, de modèles et d'outils logiciels pour concevoir ce genre d'environnement complexe. Un effort particulier devrait être investi pour contrôler, voire limiter le recours à des applications et à des outils informatiques propriétaires, dans un souci de maintien du caractère ouvert et de la convivialité de la plate-forme.

ABSTRACT

The growth in popular use of telecommunication technologies will create more opportunities in electronic business and multimedia communications, but it will create also more challenges in organizing information and facilitating its efficient manipulation. Nowadays, network applications are heading toward more replication and resource distribution. A number of distributed object systems, where dispersed sets of objects request services from one another using different communication mechanisms, have been designed. Several innovative applications have been implemented, in particular the establishment of an electronic equivalent of a real academic campus. The so-called virtual campus promotes tele-learning with all of its aspects including laboratory tele-experiments and simulations proceeding from distant client machines.

This thesis's main objective is to conceive a telecommunication platform for remote access to distributed virtual laboratories. Two phases are required in order to achieve this goal: elaborating a conceptual model and carrying on an implementation process.

From the conceptual point of view, the virtual laboratory concept can be considered as a complex computer environment integrating two categories of objects. The first one contains generic elements, commonly shared among several laboratories, and building up basic tools and functionalities. As for the second group, it specifies specific tools and functionalities belonging to a given laboratory. Integrating both specific and basic tools, and delivering them to the requesting end-user are the main tasks of the telecommunication platform. Thus, the conceptual model we are proposing consists in a three-layered architecture comprising from top to bottom:

- the specific tools and functionalities adaptation layer: Generic objects are subject to some settings in order to fit into a particular laboratory scientific background,

- the generic tools and functionalities layer,
- the user network adaptation layer: Remote access is provided after dealing with interoperability issues among the different users communication stacks (e.g. TCP/IP, ATM).

During the implementation process, our computing approach strives in as much as possible to preserve the elaborated specifications of the conceptual design. With this objective in mind, the first two layers were implemented using the object-oriented programming facilities provided by Java, the first Internet development language. For the user network adaptation layer, we have moved toward an open structure using the CORBA architecture and the HTTP interface as the relevant solution in accessing virtual laboratories. Nevertheless, each generic tool may still involve its own underlying accessibility mechanism as well as other computer components such as databases. The current telecommunication platform design has many advantages including portability, modularity, and reusability. Several tests involving heterogeneous user access networks and platforms have proven so far the flexibility of this implementation.

The most valuable credit of our work is probably in showing the feasibility of a telecommunication platform promoting pedagogical collaborative distance learning activities throughout wide interoperable environments. Future research efforts must be carried out in conceiving a more systematical methodology and models which will enhance the use of computer communications facilities in the education sector. Many issues remain open such as the affordability of complex tele-learning facilities, security, quality of service, and reliable distributed resources connectivity. In general, the use of proprietary computer tools should be avoided as much as possible in order to maintain the open system characteristics and user-friendliness features of the platform.

TABLE DES MATIÈRES

DÉDICACE iv
REMERCIEMENTS v
RÉSUMÉ vi
ABSTRACT ix
TABLE DES MATIÈRES xi
LISTE DES TABLEAUX xiv
LISTE DES FIGURES xv
LISTE DES SIGLES ET DES ABBRÉVIATIONS xvii
LISTE DES ANNEXES xx

CHAPITRE I : INTRODUCTION 1

1.1 Éléments de problématique 1
1.2 Objectifs de recherche et résultats escomptés 3
1.3 Plan du mémoire 4

CHAPITRE II : INTEROPÉRABILITÉ DE RÉSEAUX HÉTÉROGÈNES 6

2.1 Problématique de l'interopérabilité 6
2.2 Mécanismes d'interopérabilité 8
2.3 Architecture CORBA 14
2.3.1 Généralités sur CORBA 15
2.3.2 Modèle de référence de CORBA 18
2.3.3 Mode opératoire de CORBA 21
2.3.4 Quelques caractéristiques de CORBA 22

2.3.5 Complémentarité Java-CORBA 25

CHAPITRE III : MODÈLE CONCEPTUEL DE LA PLATE-FORME DE TÉLÉCOMMUNICATIONS 29

3.1 Contexte d'utilisation de la plate-forme de télécommunications 29
3.2 Spécifications techniques et fonctionnelles de la plate-forme 36
3.3 Modèle conceptuel de la plate-forme de télécommunications 40
3.3.1 Description de la structure proposée 40
3.3.2 Architecture en triple couches 42
3.4 Organisation des différentes couches du modèle 44
3.4.1 Adaptation aux réseaux de communications 45
3.4.2 Outils et fonctionnalités de base 51
3.4.3 Adaptation aux outils et fonctionnalités spécifiques 56

CHAPITRE IV : IMPLANTATION DE LA PLATE-FORME DE TÉLÉCOMMUNICATIONS 59

4.1 Définition de la plate-forme de télécommunications 59
4.1.1 Réalisation de l'adaptation aux réseaux de communications 60
4.1.2 Mise en place des outils et fonctionnalités de base 63
4.1.3 Réalisation de l'adaptation aux outils et fonctionnalités spécifiques 64
4.2 Implantation de la plate-forme de télécommunications 65
4.3 Solution proposée 72
4.3.1 Implantation de l'architecture en triple couches 72
4.3.2 Mise en œuvre des outils de base 75

CHAPITRE V : MISE EN ŒUVRE ET EXPÉRIMENTATION 79

5.1 Mise en œuvre de la plate-forme de télécommunications 79
5.1.1 Préalables informatiques à la mise en œuvre 79

5.1.2 Interface utilisateur de la plate-forme de télécommunications 83
5.2 Expérimentation de la plate-forme de télécommunications 91

CHAPITRE VI : CONCLUSION 101

6.1 Synthèse des travaux et contributions 101
6.2 Limitations actuelles et recherches futures 103

RÉFÉRENCES 105

LISTE DES TABLEAUX

5.1 Notation des séances de tests 93
5.2 Résultats de manipulation du cahier de laboratoire en mode local avec Internet . . 98
5.3 Résultats de manipulation du cahier de laboratoire en mode étendu avec Internet . 98
5.4 Résultats de téléchargement du laboratoire de physique avec ADSL 99
5.5 Résultats de manipulation du cahier de laboratoire avec ADSL 100

LISTE DES FIGURES

2.1 Interaction client/serveur avec HTTP/CGI 11
2.2 Interaction client/serveur avec les *Servlets* 12
2.3 Interaction client/serveur avec RMI 14
2.4 Interaction client/serveur avec CORBA 15
2.5 Architecture inter-ORB 16
2.6 Architecture OMA 17
2.7 Modèle de référence de CORBA 19
2.8 Attachement de références transitoires 22
2.9 Attachement de références persistantes 23
2.10 Mise en œuvre d'un code Java 26
2.11 Cycle de programmation intégrant la plate-forme Java 27
3.1 Architecture proposée pour les laboratoires virtuels 31
3.2 Emplacement de la plate-forme de télécommunications 42
3.3 Architecture en triple couches de la plate-forme de télécommunications 44
4.1 Implantation des deux premières couches du modèle conceptuel 74
4.2 Implantation de la couche d'adaptation aux réseaux des utilisateurs 75
4.3 Organisation logique des outils de base 76
4.4 Accès au cahier de laboratoire 78
5.1 Modélisation d'une séance de laboratoire virtuel 84
5.2 Page d'accueil des laboratoires virtuels 85
5.3 Fenêtre de contrôle d'accès 86
5.4 Écran d'accueil du laboratoire de physique 87
5.5 Calculatrice scientifique générique 88
5.6 Fenêtre de validation d'accès à un cahier de laboratoire 88

5.7 Affichage du contenu d'un cahier de laboratoire 89
5.8 Confirmation de suppression de données du cahier de laboratoire 90
5.9 Fenêtre de mise à jour du cahier de laboratoire 91
5.10 Infrastructure de communications expérimentale 92
5.11 Délais de téléchargement du laboratoire de physique 94
5.12 Nombre de paquets chargés du laboratoire de physique 95
5.13 Nombre de paquets et délais d'établissement de connexion 96
5.14 Nombre de paquets et délais d'ajout d'un enregistrement 96
5.15 Nombre de paquets et délais de modification d'un enregistrement 96
5.16 Nombre de paquets et délais de suppression d'un enregistrement 97
5.17 Nombre de paquets et délais de rupture de connexion 97
5.18 Requêtes échangées entre client et serveur sur le lien ADSL 99
AI.1 Fenêtre de mise à jour de la politique de sécurité 109
AI.2 Fenêtre d'attribution des permissions à une source de code 109
AI.3 Fenêtre de sélection de permissions 110

LISTE DES SIGLES ET DES ABBRÉVIATIONS

AAL : ATM Adaptation Layer

ABR : Available Bit Rate

ADSL : Asymmetric Digital Subscriber Line

API : Application Programming Interface

ASN1 : Abstract Syntax Notation One

ATM : Asynchronous Transfer Mode

BSD : Berkeley Standard Distribution

CBR : Constant Bit Rate

CGI : Common Gateway Interface

CMIP : Common Management Information Protocol

CORBA : Common Object Request Broker Architecture

DCE : Distributed Computing Environment

DCOM : Distributed Component Object Model

DII : Dynamic Invocation Interface

DSI : Dynamic Skeleton Interface

ESIOP : Environment-Specific Inter-ORB Protocol

FTP : File Transfer Protocol

GDC : General DataComm

GIOP : General Inter-ORB Protocol

GPIB : General Purpose Interface Bus

GUI : Graphical User Interface

HTML : HyperText Markup Language

HTTP : HyperText Transfer Protocol

ICMP : Internet Control Message Protocol

IDL : Interface Definition Language

IEEE : Institute of Electrical and Electronics Engineers

IIOP : Internet Inter-ORB Protocol

IOR : Interoperable Object Reference

IP : Internet Protocol

IPOA : Internet Protocol Over ATM

IPX : Internet Packet Exchange

ISO : International Organization for Standardization

JDBC : Java DataBase Connectivity

JDK : Java Development Kit

JRE : Java Runtime Environment

JVM : Java Virtual Machine

LLC : Logical Link Control

LVEST : Laboratoires Virtuels en Sciences et en Technologies

MAC : Media Access Control

MIB : Management Information Base

NFS : Network File System

NI-DAQ : National Instrument-Data Acquisition

OBV : Object By Value

OFB : Outils et Fonctionnalités de Base

OFS : Outils et Fonctionnalités Spécifiques

OMA : Object Management Architecture

OMG : Object Management Group

ORB : Object Request Broker

OSF : Open Software Foundation

OSI : Open Systems Interconnection

OTP-N : Optical Transparent Packet Network

POA : Portable Object Adapter

PSTN : Public Service Telephone Network

RDA : Remote Data Acquisition

RFC : Request For Comment

RMI : Remote Method Invocation

RNIS : Réseau Numérique à Intégration de Services

RPC : Remote Procedure Call

RTCP : Réseau Téléphonique Commuté Public

SGBD : Système de Gestion de Base de Données

SII : Static Invocation Interface

SMTP : Simple Mail Transfer Protocol

SNMP : Simple Network Management Protocol

SPX : Sequenced Packet Exchange

SQL : Simple Query Language

SS7 : Signaling System Seven

TCP : Transmission Control Protocol

TFTP : Trivial File Transfer Protocol

UDP : User Datagram Protocol

URL : Universal Resource Locator

VI : Virtual Instrument

VISA : Virtual Instrument Software Architecture

WDM : Wavelength Division Multiplexing

XML : Extensible Markup Language

LISTE DES ANNEXES

ANNEXE I : **Configuration préalable du site client pour établir des connexions CORBA**. 108

CHAPITRE I
INTRODUCTION

À l'aube du troisième millénaire, un nouveau chapitre s'inscrit dans l'histoire scientifique de l'Humanité. Fixes ou mobiles, la multitude des réseaux existants englobe non seulement des centaines de composants matériels et d'équipements différents, mais également toute une gamme d'architectures et de techniques de communications spécialisées. Cette diversité technologique soulève le problème de l'interopérabilité qui s'avère toutefois indispensable si on veut satisfaire les besoins des usagers. De jour en jour, ces derniers deviennent plus exigeants sur le plan de la qualité des services attendus, notamment ceux reliées aux applications multimédias. Téléconférences, diagnostics médicaux à distance, outils de surveillance, sont quelques exemples d'applications parmi d'autres actuellement disponibles sur le marché. Le présent chapitre décrit brièvement le contexte, présente les éléments de problématique, précise les objectifs de recherche et les résultats attendus d'une plate-forme de télécommunications supportant des laboratoires virtuels distribués destinés en particulier au télé-enseignement.

1.1 Éléments de problématique

Depuis son existence, le réseau téléphonique à commutation de circuits s'est révélé une infrastructure exemplaire de fiabilité et d'efficacité, deux caractéristiques fortement recherchées au niveau de l'Internet. Bien que populaires, ni Internet ni le réseau téléphonique seuls ne semblent cependant être capables de satisfaire les besoins de connectivité du marché mondial. D'où l'introduction du mode de commutation de cellules communément utilisées dans ATM. Cette technologie, même dans sa phase

embryonnaire, promet de hautes performances dans un environnement multi-services. Pourtant, son déploiement ne peut avoir lieu en ignorant l'étendue de la base installée des technologies traditionnelles. Il convient donc de fournir des moyens de coexistence de l'ensemble des technologies, tout en ayant soin de réduire les coûts des ressources investies pour y parvenir.

Il est tout à fait illusoire de considérer les technologies de communications actuelles comme un effet de mode. C'est plutôt une vague de fond qui va changer toutes les données du marché, le mode de vie, la culture et l'aspect social même des communautés humaines de cette planète. Chaque jour, les concepts de base de la nouvelle ère de l'information s'enracinent davantage dans la vie quotidienne et ouvrent la voie à de nouvelles classes d'applications. Le projet de laboratoires virtuels en sciences et en technologies n'est qu'une illustration de ce fait dans un cadre pédagogique.

Parmi les avantages potentiels des laboratoires virtuels, nous pouvons mentionner :

- Réduction de coûts : L'organisme universitaire n'aura pas à se soucier des coûts des composants matériels d'un laboratoire réel tels que les instruments de mesure, les locaux et les espaces de travail ;
- Flexibilité : Chaque type de laboratoire engendre plusieurs manipulations avec des gammes variées de paramètres ;
- Réutilisabilité : Les modules logiciels constituant les différents laboratoires sont faciles à maintenir et sujets à des modifications et des améliorations ultérieures de plus en plus sophistiquées ;
- Portabilité : Les utilisateurs n'auront plus à se soucier des plates-formes qu'ils utilisent ni des heures d'accès aux laboratoires.

Plus les réseaux d'ordinateurs progressent, plus la variété des machines qu'ils relient et des liens qu'ils utilisent augmente. En effet, chaque catégorie de réseaux possède une organisation logique, un mode de commutation, un format de données et un niveau de

qualité de service qui lui sont propres. Ceci explique en partie l'existence d'un environnement hétérogène de réseaux publics et privés de dimensions illimitées posant un certain nombre de problèmes d'incompatibilité et d'interopérabilité.

Par ailleurs, de nouveaux protocoles ont été définis en vue d'adapter les réseaux à bas débit aux services de haut débit. De nouvelles variantes ont également été conçues afin de supporter du trafic à des débits variables. Parviendra-t-on un jour à réaliser le rêve d'une infrastructure de communications globale, multi-services, rapide et surtout fiable ? Pourra-t-on mettre au point une plate-forme de télécommunications qui rend transparente à l'utilisation la différence de protocole, de format de messages, de débit, et j'en passe, qui conditionne l'interopérabilité des réseaux d'accès à cet environnement de laboratoires virtuels ? Autant de questions de recherche auxquelles le présent mémoire ambitionne de donner une réponse.

1.2 Objectifs de recherche et résultats escomptés

L'objectif principal de ce mémoire est de concevoir une plate-forme de télécommunications capable d'assurer l'interopérabilité de réseaux hétérogènes servant d'infrastructure d'accès à des laboratoires virtuels distribués. Conceptuellement, cette plate-forme est une structure en triple couches au centre de laquelle se trouve la couche des outils et des fonctionnalités de base. De part et d'autre, on retrouve la couche d'adaptation aux outils et fonctionnalités spécifiques, ainsi que la couche d'adaptation aux réseaux.

L'idée consiste à modéliser un noyau de laboratoire générique dont des extensions progressives conduiront à la mise en place d'autres laboratoires spécifiques. Pour commencer, deux prototypes de laboratoires, un de physique et l'autre de génie électrique, ont été implantés. Cependant, rien n'empêche la réalisation ultérieure d'une liste non exhaustive de laboratoires : chimie, biologie, physique nucléaire, robotique,

etc. Un modèle générique conservera donc un ensemble le plus complet possible d'attributs externes communs à divers laboratoires spécialisés. Ces attributs externes donnent lieu à des outils et fonctionnalités de base (OFB) partageables par l'ensemble des laboratoires spécifiques. Chaque laboratoire est également doté de caractéristiques et de fonctionnalités propres désignées par outils et fonctionnalités spécifiques (OFS).

L'approche adoptée est à la fois orientée objet et inductive. Orientée objet, puisque les laboratoires spécialisés héritent un certain nombre de propriétés du laboratoire générique agissant comme ancêtre commun. Et inductive, car l'enrichissement du modèle générique résulte de l'ajout progressif de divers groupes d'attributs internes issus du développement d'un nombre suffisant de laboratoires spécialisés.

Suite à la réalisation de la plate-forme, on s'attend à un modèle client/serveur bel et bien dédié aux laboratoires virtuels spécialisés. Côté serveur, il s'agira d'un ensemble de processus parallèles qui reçoivent les requêtes des utilisateurs, leur accordent l'accès et leur fournissent les moyens facilitant l'exécution des manipulations souhaitées. Côté client et à travers une interface graphique interactive, l'utilisateur sera capable de choisir une simulation quelconque, sélectionner les paramètres convenables et visualiser les résultats correspondants. Les utilisateurs auront l'option de consulter les cahiers de laboratoires les uns des autres et de les remettre à une autre entité telle que le professeur. Le modèle fourni sera, autant que possible, une imitation de ce qui se passe dans un laboratoire réel, mais plutôt dissimulé derrière l'écran d'un ordinateur.

1.3 Plan du mémoire

Ce mémoire comprend six chapitres. Le chapitre deux expose le problème d'interopérabilité dans sa généralité et renseigne sur les techniques courantes de communications entre éléments hétérogènes distribués. Le chapitre trois présente le modèle conceptuel et la méthodologie adoptée pour la réalisation de la plate-forme de

télécommunications. Comme instance de son précédent, le chapitre quatre précise les détails d'implantation de cette plate-forme, alors que le chapitre cinq en expose la mise en œuvre et l'expérimentation. Le chapitre six, en guise de conclusion, effectue la synthèse des travaux et esquisse quelques indications de recherches futures en vue de combler les inévitables lacunes de réalisation.

CHAPITRE II
INTEROPÉRABILITÉ DE RÉSEAUX HÉTÉROGÈNES

L'exploitation de la théorie électrique et électromagnétique a ouvert la porte au déploiement de toute une gamme de techniques de transmission. Le concept de réseau de communication a trouvé sa première réalisation à travers le réseau téléphonique dédié aux messages vocaux. Par la suite sont arrivés les réseaux de données, et actuellement la tendance est vers des réseaux de services multiples caractérisés par un certain degré d'hétérogénéité. Ce chapitre étudie les aspects d'interopérabilité des réseaux hétérogènes. Dans un premier temps, nous exposons la problématique de l'interopérabilité. Par la suite, nous expliciterons les modèles opérationnels les plus courants pour développer des applications distribuées dans un environnement de réseaux hétérogènes. Pour finir, nous ferons une brève présentation de l'environnement CORBA qui est de plus en plus utilisé dans ce contexte.

2.1 Problématique de l'interopérabilité

Au fil des années, la dimension du marché informatique n'a cessé de croître. Des milliers de produits ne font que promouvoir des concepts divers de modèles client/serveur, d'environnements distribués, d'interfaces utilisateurs, de systèmes ouverts et de télématique. En dépit de toutes ses qualités et fonctionnalités, un produit doit satisfaire un critère fondamental afin d'être adopté dans un monde plein de composants, de plates-formes et de réseaux hétérogènes : l'interopérabilité. Ceci est non seulement une estimation de la capacité d'un produit à coexister, voire coopérer, avec d'autres produits dans des contextes distincts, mais aussi une référence à ce qu'on appelle le caractère ouvert. Plus concrètement, il s'agit de l'interopérabilité de systèmes

hétérogènes en termes de technologies, de protocoles de communications et d'outils d'implémentation. En effet, au fur et à mesure que les techniques de communications se multiplient, les barrières technologiques, réglementaires, temporelles et spatiales ne cessent de s'effondrer. Ainsi, les solutions propriétaires perdent de plus en plus de terrain car elles manquent de potentiel d'évolution.

Afin de comprendre le vrai sens de l'interopérabilité, il suffit de se référer au succès de l'infrastructure téléphonique. Grâce à leurs interfaces ouvertes, les communications par téléphone ont acquis une vaste portée géographique, peu importe les équipements disponibles : analogiques, numériques, cellulaires, terrestres et satellitaires. À l'heure actuelle, un défi majeur consiste à bien définir les principes d'inter-opération des services de téléphonie IP et de l'infrastructure classique des réseaux téléphoniques publics (Dalgic et al., 1999).

Des mesures d'interopérabilité sont prises en considération dans les milieux industriels pour les quatre premières couches de l'architecture OSI. En effet, ces mesures couvrent la couche physique avec toutes les spécifications d'interconnexion matérielle et les composants nécessaires : câbles coaxiaux, fibres optiques, concentrateurs, connecteurs, etc. Il en est de même pour la couche de liaison de données, notamment avec les sous-couches MAC et LLC. Quant aux couches de réseau et de transport, elles sont desservies par les protocoles IP et TCP assez populaires. Des spécifications ouvertes sont aussi prévues pour les couches de session, présentation et application.

Par ailleurs, Internet a réussi à devenir un environnement intégrant une variété de systèmes et de contextes pour l'exécution d'applications portables distribuées, dont les plus critiques sont les applications multimédias, avec toutes leurs contraintes de bande passante et de temps réel. Évidemment, toute intégration dans un environnement interopérable possède plusieurs attributs et nécessite des méthodologies bien claires (Prnjat et Sacks, 1999). D'autre part, de nouvelles technologies viendront joindre le

marché apportant plus de diversité, donc d'hétérogénéité, tant au niveau des réseaux locaux qu'étendus. Tel est le cas d'ATM et des réseaux optiques très sophistiqués WDM et tout récemment OTP-N (Bostica et al., 1999).

2.2 Mécanismes d'interopérabilité

Quelle que soit la taille d'un environnement hétérogène, la complexité d'interopérabilité doit être dissimulée derrière une interface utilisateur bien construite. C'est le cas du World Wide Web permettant à l'échelle planétaire un grand nombre d'applications réparties. Cependant, l'orientation de plus en plus sophistiquée des besoins en matière de solutions informatiques ne fait que poser plus de contraintes à son évolution. Un degré d'interactivité supérieur est alors indispensable vue la distribution des ressources dans des environnements différents d'une part, et la multiplication des transactions effectuées d'autre part.

Face à ce genre de situation, plusieurs tentatives ont eu lieu dans le but d'améliorer l'intergiciel (*middleware*) classique HTTP/CGI prédominant depuis 1995. Le problème crucial de toutes ces approches réside dans la difficulté de faire converser directement clients et serveurs. En d'autres termes, toute interaction entre les ressources (appelées communément objets) du client et celles du serveur exige la médiation du serveur Web et du protocole HTTP. Une nouvelle perspective pour les opérations entre objets distribués fut abordée par Harkey et Orfali (1998) sous le titre de Web orienté objet (*Object Web*). Il s'agit d'une infrastructure innovatrice qui tend à pallier les obstacles de répartition, d'interopérabilité et de portabilité des applications mettant en jeu des plates-formes hétérogènes. En dépit de sa récente définition, cette notion de Web orienté objet fut sérieusement prise en compte par la communauté scientifique, et les travaux de recherche et de développement correspondant sont déjà dans une phase avancée.

Par conséquent, le marché informatique assiste à une véritable compétition laissant la porte ouverte à une variété d'intergiciels promouvant, chacun à sa façon, des fonctionnalités assez utiles vis-à-vis des utilisateurs et également des programmeurs. Parmi ces intergiciels, mentionnons : les *sockets*, l'ensemble HTTP/CGI, les *servlets* et RMI.

Sockets : Par définition, il s'agit de points de communications munis d'un nom et d'une adresse réseau, et agissant comme interface d'accès au réseau en question. En général, ce n'est qu'un protocole brut de communications induisant un bas niveau de programmation et servant de couche inférieure à d'autres protocoles et mécanismes genre NFS et RPC. Leur introduction en 1981, à travers la version BSD 4.2 du système UNIX, leur a attribué une appartenance exclusive à la famille de protocoles TCP/IP dans ses modes de communications orientés connexion (TCP), non orientés connexion (UDP) et natifs (IP ou ICMP). Plus tard, des extensions spéciales ont été ajoutées pour d'autres plates-formes et d'autres architectures de communications, telles que IPX/SPX et ATM. Une telle hétérogénéité de contextes se reflète aussi dans la variété de langages de programmation structurée et orientée objet.

Bien qu'une modélisation orientée objet des *sockets* facilite l'intégration desdites entités clients et serveurs dans un environnement d'objets distribués, quelques lacunes peuvent être citées :

- l'absence de transactions : Il n'y a aucune garantie d'assurer l'atomicité des opérations impliquant des objets distribués ;
- le manque de possibilité de découverte dynamique d'entités distribuées : Il n'y a pas de moyen de mise à jour et de découverte d'objets connectés ou déconnectés au cours d'une session statiquement déclenchée entre deux entités données ;
- l'inexistence d'interfaces de description : Les données véhiculées sur les liens n'ont pas de représentation commune. Les chances de réussite d'opérations distantes

faisant intervenir des paramètres de types informatiques différents sont donc de moins en moins fortes.

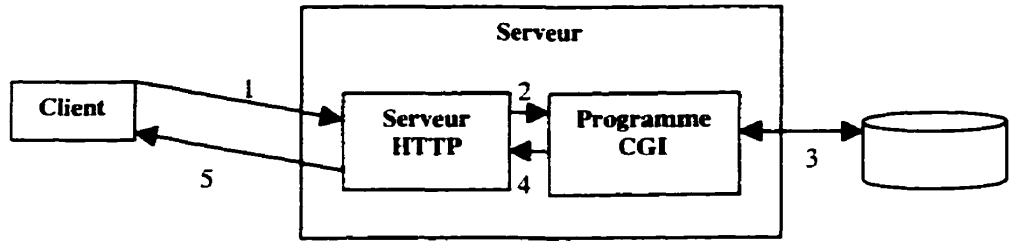
Ainsi, il s'avère plus adéquat d'opter pour plus d'abstraction, i.e. un niveau de programmation plus élevé dans lequel ce genre de déficiences est complètement masqué.

HTTP/CGI : Cet intergiciel apporte au Web la possibilité d'effectuer des échanges dynamiques à l'aide de documents statiques HTML. Introduit en 1990, le protocole HTTP est en fait un outil d'appel de procédures distantes, un peu à la manière de RPC au-dessus de la suite TCP/IP. Une interaction typique client/serveur moyennant HTTP/CGI est représentée à la Figure 2.1.

Malgré sa simplicité, HTTP/CGI introduit une grande surcharge puisque les requêtes successives d'un même client sont traitées tout à fait séparément. En revanche, le protocole HTTP permet à un fureteur de négocier, lors d'une connexion, le format de données qu'il est capable de manipuler. De sa part, le protocole CGI permet au serveur Web de lancer un programme quelconque et de lui acheminer les paramètres d'entrée sous forme de variables d'environnement. Il est possible toutefois qu'un code CGI orienté objet fasse parvenir des objets distribués. Ceci demeure restreint et lourd, à cause du principe de fonctionnement du protocole HTTP lui-même. En plus des problèmes que présentent les *sockets* dans un environnement d'objets distribués, HTTP/CGI souffre aussi des défauts suivants :

- une lenteur de traitement : Elle est parfois considérable et provient de la procédure multi-phases lors de l'établissement d'une liaison entre un client et un objet serveur ;
- une complexité d'interaction : Une séquence de questions/réponses nécessite plusieurs connexions entre client et serveur.

Actuellement, malgré la renommée de HTTP/CGI, ses aspects négatifs font l'objet de plus en plus de critiques, motivant ainsi les concepteurs de réseaux de communication à rechercher d'autres solutions.

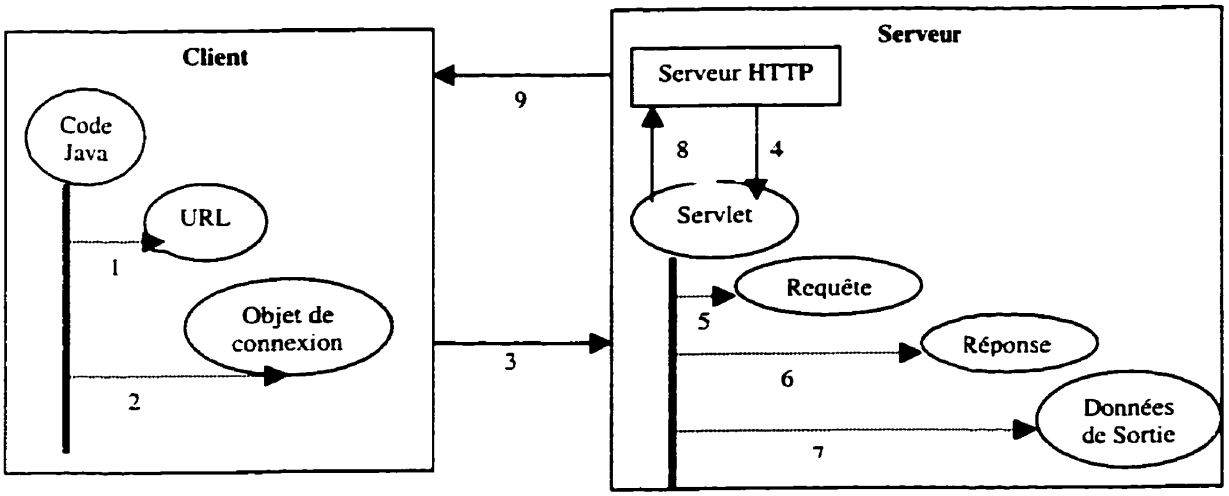


Légende :
1. Requête de client
2. Exécution du programme CGI
3. Interaction du programme avec des structures de données internes ou externes au serveur
4. Remise des résultats au serveur
5. Remise des résultats au client à l'intérieur d'un document HTML

Figure 2.1 Interaction client/serveur avec HTTP/CGI

Les *servlets* : En 1997, Javasoft introduisit un premier serveur Web en Java muni d'un nouveau type d'outils appelés *servlets*. Le serveur s'occupe de charger dynamiquement ou lors de son démarrage un *servlet* procurant des classes et des modules spécifiques aptes d'une part à remodéliser le protocole CGI, et d'autre part à coexister avec le protocole HTTP. Pratiquement, le scénario est un peu plus complexe que ses précédents, comme le montre la Figure 2.2.

Comparée à HTTP/CGI, l'approche des *servlets* apporte deux améliorations : une durée de vie plus longue d'un *servlet* ainsi qu'une possibilité de connexion permanente à une base de données. Ceci réduit la surcharge de traitement pour des requêtes successives, mais augmente par contre la durée de service.



Légende :
1. Création d'un objet URL symbolisant la connexion
2. Création de l'objet effectif de connexion
3. Émission de la requête HTTP
4. Chargement du *servlet* (s'il ne l'est pas déjà)
5. Création et communication de paramètres à l'objet requête puis exécution des opérations
6. Création de l'objet réponse
7. Génération de l'objet des résultats
8. Remise des résultats au serveur HTTP
9. Remise du document HTML réponse

Figure 2.2 Interaction client/serveur avec les *Servlets*

RMI : Attribué exclusivement à un contexte de programmation Java, l'API ou interface de programmation **RMI** n'est autre que l'intégration d'un modèle d'objets distribués manipulés, localement ou à distance, à partir de ce qu'on appelle des machines virtuelles (JVM) Java. Avant toute interaction client/serveur avec **RMI**, des étapes préalables doivent être effectuées des deux côtés (Harkey et Orfali, 1998) :

- l'implémentation de l'objet serveur ;
- la génération du code réel de l'objet serveur (*skeleton*) et de son code délégué ou stub téléchargeable côté client ;
- l'enregistrement de l'objet distribué en vue d'une accessibilité publique ;

- l'évocation de l'objet distribué.

RMI exige l'activation d'un gestionnaire de sécurité également sur la machine du client et celle du serveur, afin de pallier les menaces de déstabilisation de systèmes dues aux téléchargements et aux échanges subséquents. Typiquement, une interaction client/ serveur sous RMI prend la forme décrite à la Figure 2.3.

Il apparaît alors clair que l'approche RMI diffère de ses précédentes et apporte certains avantages :

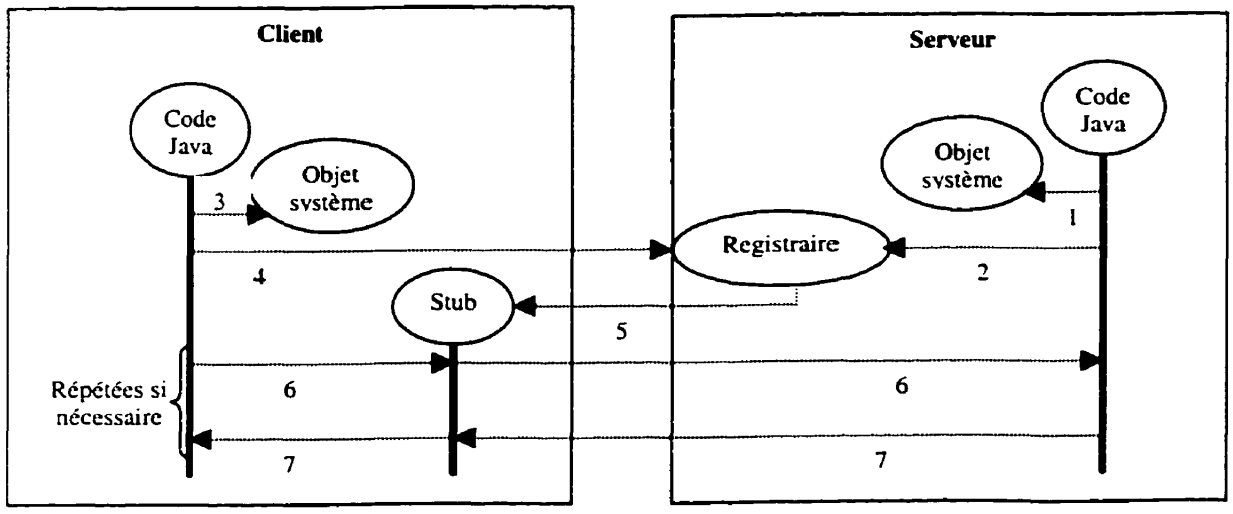
- des traitements plus rapides qu'avec HTTP/CGI ;
- une interface de description pour gérer les fonctionnalités de clients et de serveurs avec un niveau élevé d'abstraction ;
- des téléchargements dynamiques sécurisés de code et de classes Java ;
- un contrôle d'objets distribués y compris des mécanismes de ramasse-miette.

Par ailleurs, RMI ne permet pas la propagation de transactions ni la découverte dynamique d'objets répartis. L'inconvénient majeur de RMI provient de sa conception même basée exclusivement sur le langage Java, ce qui supprime toute sorte d'interopérabilité entre des objets générés par des langages informatiques différents. La possibilité de mise en échelle de grands systèmes distribués est elle aussi réduite.

En dépit de ses points faibles, RMI peut être considéré comme un premier pas vers un concept plus développé d'interaction dans un environnement ouvert d'objets distribués. Javasoft et OMG joignent leurs efforts afin de mettre au point un paradigme plus général. Là où l'homogénéité de RMI est rejetée, un autre acteur entre alors en scène sous le nom de CORBA.

2.3 Architecture CORBA

Dès sa fondation en mai 1989, le consortium OMG s'est fixé un objectif principal d'interopérabilité. Comptant actuellement plus de 700 membres internationaux, OMG a réussi en 1996 à atteindre son but à travers la spécification CORBA 2.0. Depuis, plusieurs extensions et fonctionnalités continuent à apparaître dans les versions successives de CORBA dont la dernière, CORBA 3.0, est prévue pour 1999 (Harkey et Orfali, 1998).



Légende :
1. Initialisation du gestionnaire de sécurité du serveur
2. Enregistrement de l'objet serveur
3. Initialisation du gestionnaire de sécurité du client
4. Localisation de l'objet distant
5. Téléchargement du stub correspondant
6. Invocation d'opération via le stub
7. Réception de résultats des opérations distantes via le stub

Figure 2.3 Interaction client/serveur avec RMI

2.3.1 Généralités sur CORBA

En fait, CORBA se présente sous la forme d'une famille de spécifications établies par OMG (Siegel, 1998). Deux aspects principaux caractérisent le mode opérationnel de CORBA. D'une part, tout objet doit être muni d'une interface IDL dissimulant les détails de son implémentation réelle, tels que les opérations effectuées, leurs paramètres d'entrée et de sortie et les exceptions qui risquent d'être générées par suite d'exécutions erronées. D'autre part, qu'il s'agisse d'objet local ou distant, les requêtes statiques ou dynamiques conservent une forme unique et sont gérées uniformément par les routines et les librairies constituant la couche ORB aux deux extrémités d'une connexion. L'association IDL/ORB préserve un niveau élevé d'encapsulation puisqu'il n'y a aucune contrainte sur les langages de programmation mis en jeu. Java, C, C++, Cobol, Smalltalk et Ada ont déjà des extensions équivalentes en IDL. La Figure 2.4 illustre des interactions typiques entre clients et objets serveurs.

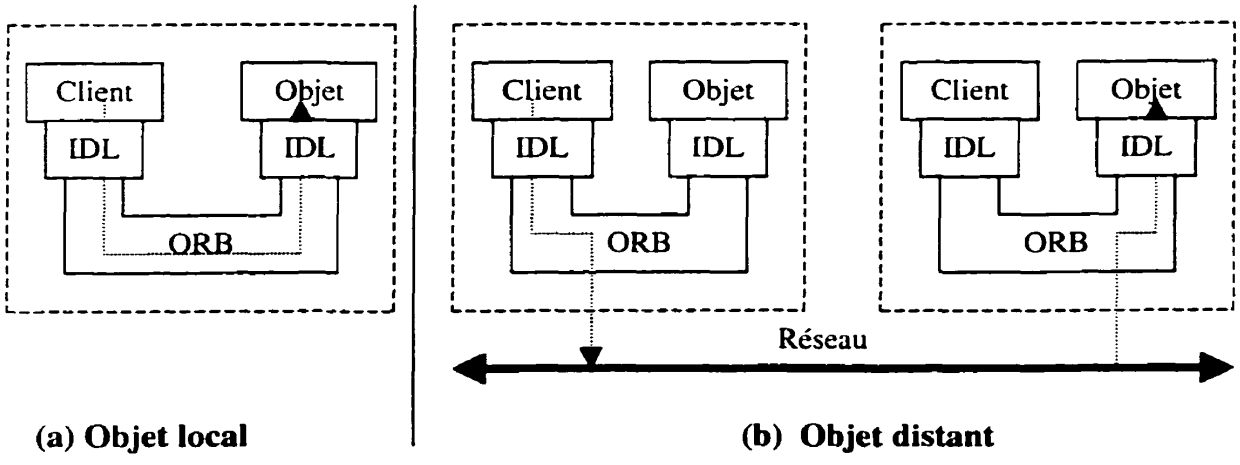


Figure 2.4 Interaction client/serveur avec CORBA

Un trait saillant du caractère distribué de CORBA réside dans le fait que les interfaces IDL des objets sont accessibles indépendamment des plates-formes et des ORB

provenant de différents vendeurs. De tels mécanismes sont rendus possibles grâce au standard générique GIOP. Ce dernier couvre tous les aspects d'interopérabilité au niveau des protocoles de communications de proche-en-proche, donc des couches physique, liaison de données et réseau. Conçu en principe pour opérer au-dessus d'un protocole de transport orienté connexion, GIOP conduit au protocole IIOP lorsqu'il est utilisé avec la suite TCP/IP, ce qui légalise l'exploitation de l'Internet en un ORB dorsal à travers lequel divers ORB peuvent interagir. Rien n'empêche pourtant la cohabitation de GIOP avec d'autres protocoles fiables, y compris les protocoles asynchrones, genre OSI, IPX, ATM et la famille SS7 de protocoles de télécommunications. Dans des contextes plus sophistiqués, des protocoles similaires à GIOP sont mis en œuvre et connus sous le nom de DCE/ESIOP (Harkey et Orfali, 1998). L'architecture inter-ORB est décrite à la Figure 2.5.

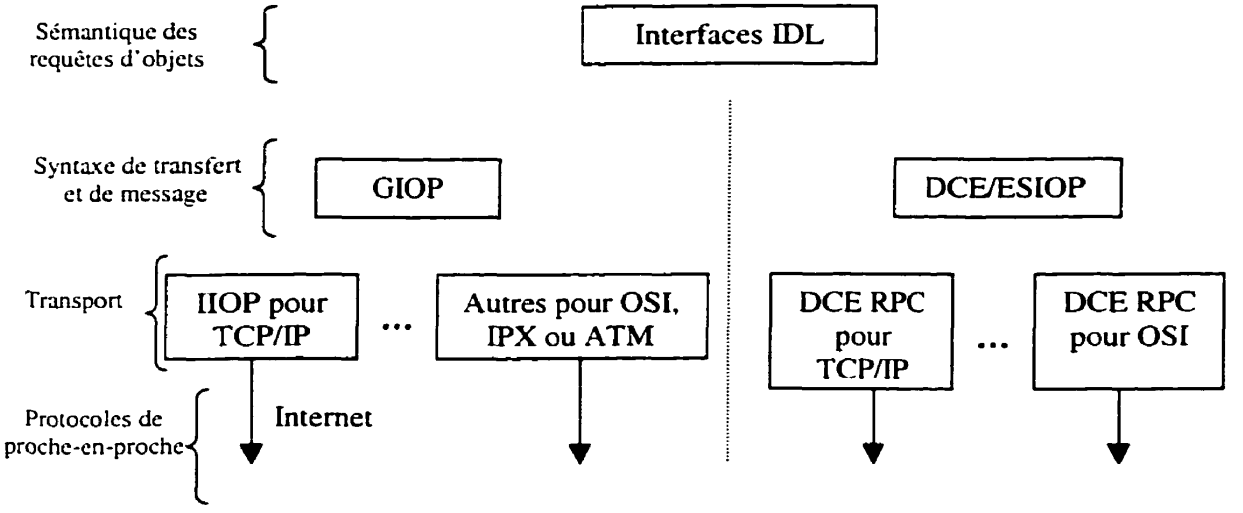


Figure 2.5 Architecture inter-ORB

Certaines spécifications OMG n'ont pas, pour le moment, atteint une phase définitive :

- Les composants CORBA : Ce sont une extension plus sophistiquée et plus abstraite du modèle existant d'un objet CORBA ;
- Les modes de communications CORBA pour circuit intégré et ceux en temps réel.

À côté de l'ORB, des services (*CORBAservices*) et des fonctions (*CORBAfacilities*) supportent l'interopérabilité de CORBA et constituent ce qu'on appelle l'architecture OMA représentée à la Figure 2.6.

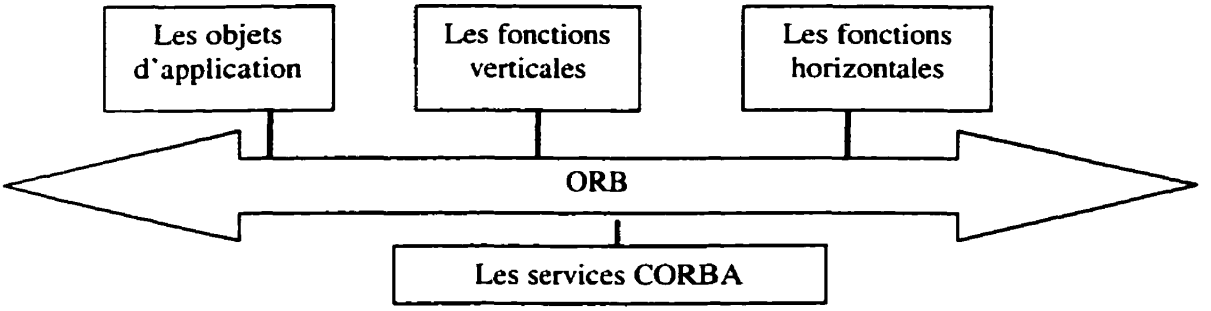


Figure 2.6 Architecture OMA

Accessibles à travers des interfaces IDL, les services CORBA constituent des services de base, de niveau système, indispensables à tout objet. Ils parviennent en complément à l'ORB facilitant notamment la création et le nommage d'objets distribués. Parmi les quinze services définis par OMG, quatre sont particulièrement importants :

- l'accès aux références d'objets distribués à l'aide desdits *Naming Service* et *Trader Service* ;
- la notification lors de l'occurrence d'événements ou de changements d'états d'un objet grâce aux *Event Service* et *Notification service* ;
- le support des sémantiques et des protocoles dédiés aux transactions, fourni par le *Object Transaction Service* ;
- la sécurité d'interopérabilité garantie par le *Object Security Service.*

D'autres services supportent la mise en œuvre ou le cycle de vie d'objets, les relations et les associations dynamiques entre eux, ou même les requêtes de consultation de bases de données selon la spécification SQL3.

Quant aux fonctions CORBA, elles sont considérées, vis-à-vis des applications, comme des services de niveau intermédiaire. Munies aussi d'interfaces IDL, ces fonctions sont séparées en deux groupes :

- les fonctions horizontales : Elles consistent pratiquement en quatre catégories communément utilisées par diverses applications : l'administration d'information, l'administration de systèmes, l'administration de tâches et l'interface utilisateur ;
- les fonctions verticales : Celles-ci constituent des services orientés vers des domaines spécifiques d'applications et implémentés sous forme d'objets standards.

Le dernier acteur de l'architecture OMA est bien incarné par des objets non standards, munis d'interfaces IDL et développés par des langages informatiques distincts qui dépendent des applications.

2.3.2 Modèle de référence de CORBA

Qualifié d'innovateur, le concept qu'introduit CORBA s'affirme de plus en plus comme conforme aux aspirations des entreprises aussi bien des utilisateurs individuels. La portabilité et l'interopérabilité résultent d'une collaboration bien organisée entre tous les composants du modèle informatique de CORBA, dont un schéma général est exposé à la Figure 2.7 (Schmidt, 1998).

Les composants essentiels de cette structure sont les suivants : servant, client, ORB, interface d'ORB, stubs IDL et *Skeletons* IDL, compilateur IDL, DII, DSI, adaptateur d'objets (*Object Adapter*), archivage d'interfaces (*Interface Repository*) et archivage d'implémentations (*Implementation Repository).*

- Servant : Ceci représente le corps d'un objet CORBA, c'est-à-dire l'implémentation en un langage informatique donné de ses méthodes définies auparavant en IDL. Un servant est identifié par une référence unique relative à l'objet évoqué par l'intermédiaire d'un processus serveur.

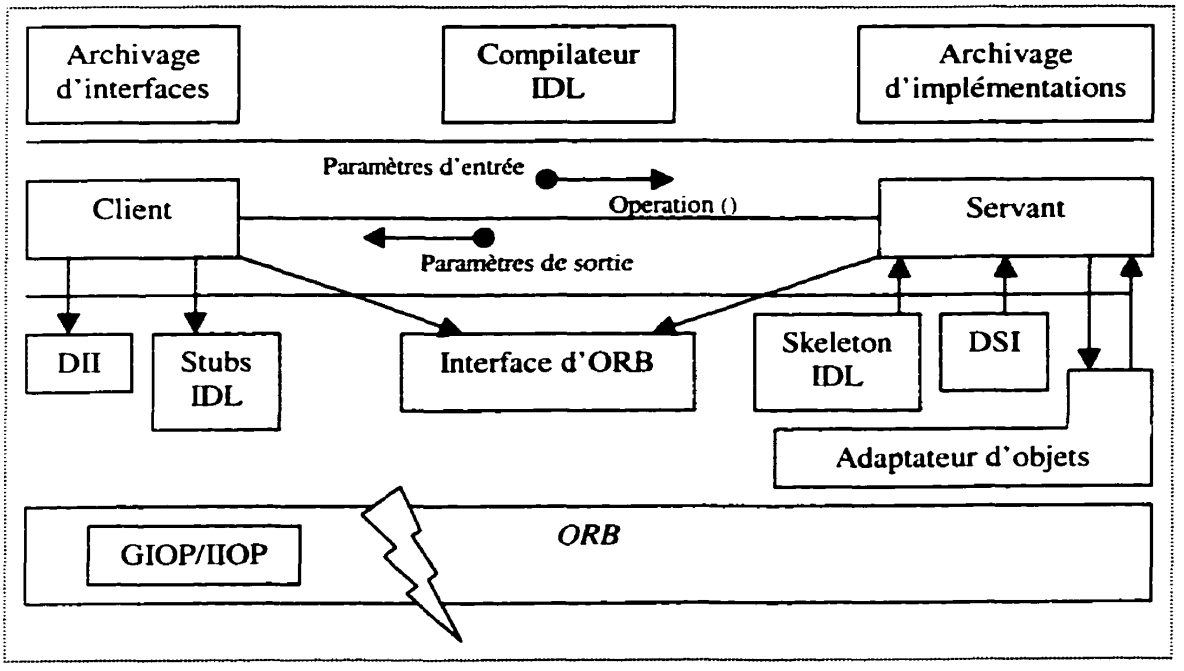


Figure 2.7 Modèle de référence de CORBA

- Client : Une fois la référence de l'objet voulu obtenue, le client ne fait qu'en évoquer des opérations et récupérer éventuellement leurs résultats.
- ORB : C'est l'élément de base des communications effectuées dans un environnement CORBA. Il offre des mécanismes sous-jacents pour localiser un objet, lui livrer les requêtes clients et leur retourner les réponses si elles existent.
- Interface d'ORB : Son objectif principal est de découpler les applications des détails d'implémentation des structures de l'ORB lui-même. Parmi les fonctions standards qu'assume cette interface, on peut citer : l'initiation et la suspension de l'ORB, la conversion des références d'objets en chaînes de caractère et inversement, ainsi que la mise en forme des requêtes dynamiques et de leurs paramètres à travers l'interface dynamique DII.
- Stubs et *skeletons* IDL : Ces programmes séparent les clients et les servants, respectivement, de l'ORB. Un stub joue le rôle d'interface statique SII capable

d'assigner aux données de l'utilisateur, des représentations communes au niveau des paquets véhiculés sur le réseau. Inversement, un *skeleton* intervertit les données reçues en un format adéquat avant de les remettre aux servants relatifs à l'application serveur en question.

- Compilateur IDL : Outre sa tâche principale d'assurer la correspondance transparente entre les définitions IDL et les différents langages informatiques, ce compilateur élimine certaines erreurs classiques en programmation réseau et optimise les compilations.
- DII : Par rapport à l'interface statique SII des stubs, l'interface dynamique DII permet à un client d'évoquer des opérations sur des objets distribués dont l'interface n'était pas connue préalablement lors de la compilation.
- DSI : C'est l'élément homologue au DII du côté d'un serveur. Les requêtes acheminées via cette interface sont remises à un servant n'ayant pas d'avance de connaissances des objets qu'il implémente.
- Adaptateur d'objets : Considéré comme un médiateur entre les objets CORBA et leurs implémentations en langages quelconques ou servants (Vinoski, 1998), cet élément se charge d'activer un objet inerte, d'associer un servant à un ORB, de démultiplexer les requêtes qui lui sont destinées et d'initier les opérations requises. Un ORB est alors apte à supporter une variété de servants ayant des propriétés, des styles d'implémentation et des durées de vie différentes.
- Archivage d'interfaces : Il s'agit exactement de l'endroit où sont stockées dynamiquement les informations relatives aux interfaces IDL et celles correspondant aux interfaces ORB.
- Archivage d'implémentations : Les informations qui y sont stockées permettent à un ORB spécifique de localiser et d'activer les servants d'objets. De plus, on y trouve d'autres renseignements sur le fonctionnement des servants tels que le contrôle administratif, l'allocation des ressources et la sécurité.

2.3.3 Mode opératoire de CORBA

La structuration du modèle de référence de CORBA ne fait que lui affecter un qualificatif de bus logiciel conceptuel (Seetharaman, 1998). La cohabitation des applications avec leurs contextes hétérogènes découle essentiellement de la présence des ORB. La simple mise en place d'interfaces et de modules adéquats permet alors à un objet d'être tantôt client tantôt serveur. Une interaction entre deux objets distribués requiert impérativement l'existence d'une référence unique à l'objet désigné serveur. Lorsque l'objet est créé, une référence d'objet (*Object Reference*) ou IOR lui est retournée par un adaptateur d'objets. Celle-ci est parfaitement opaque à une application et contribue à l'amélioration de l'indépendance des opérations vis-à-vis des langages, des protocoles et des plates-formes.

Un IOR est composé de trois champs :

- L'identificateur d'archivage : Il permet de retrouver la définition IDL de l'interface descriptive de l'objet correspondant.
- Le protocole et l'adresse : Ceux-ci renseignent sur l'emplacement de l'objet ainsi que le protocole de communication utilisé. Dans le cas de IIOP, le champ d'adresse n'est autre que la combinaison du nom ou adresse IP de l'hôte et du numéro de port TCP.
- La clé d'objet : Elle indique, d'une part, le nom de l'adaptateur d'objets capable d'accéder au servant de l'objet en question, et l'identificateur de l'objet proprement dit d'autre part.

Le processus de localisation d'un objet distribué s'appelle attachement (*Binding*). Par ailleurs, l'avènement de nouvelles spécifications d'adaptateurs d'objets a mis au point deux types de références (Vinoski, 1998). Le premier est celui des références transitoires dont la durée de vie est fonction de celle du processus serveur qui lui a donné naissance. Quant au deuxième type, il s'agit de références permanentes ou persistantes. Celles-ci

continuent à identifier le même objet, quelle que soit la machine sur laquelle s'exécute le serveur, ainsi que le nombre de fois où ce dernier redémarre.

Il faut noter que le deuxième champ d'une référence d'objet est remplacé par l'adresse d'archivage d'implémentations renseignant sur l'adaptateur d'objets, la commande d'activation du serveur ainsi que l'adresse courante de ce dernier. Le type d'un IOR lui est attribué par le code de l'application serveur lors de sa création. Cependant, des divergences significatives existent entre les mécanismes d'attachement de références transitoires et persistantes, représentés respectivement aux Figures 2.8 et 2.9.

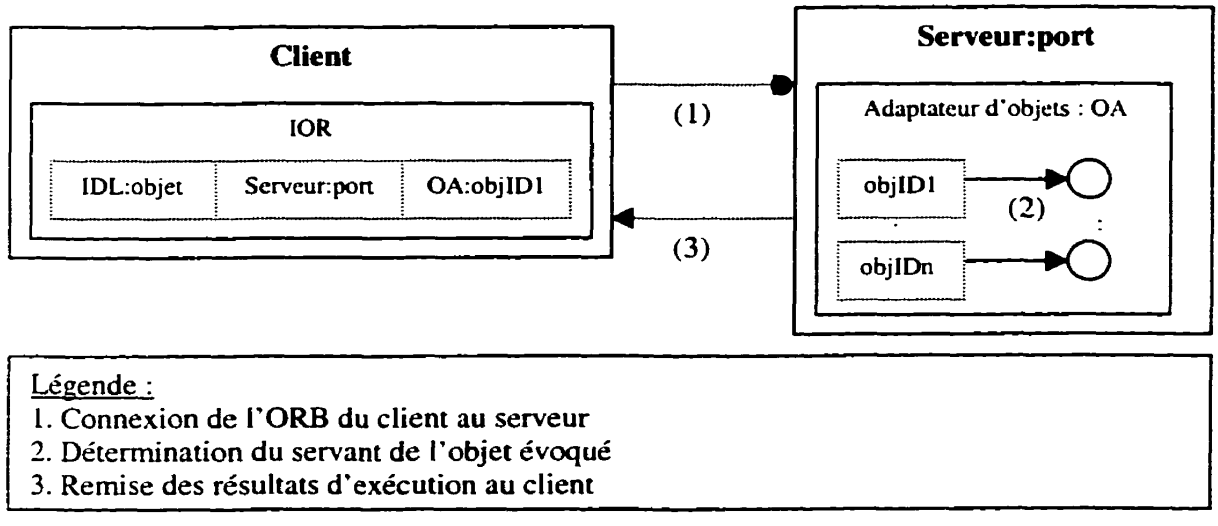
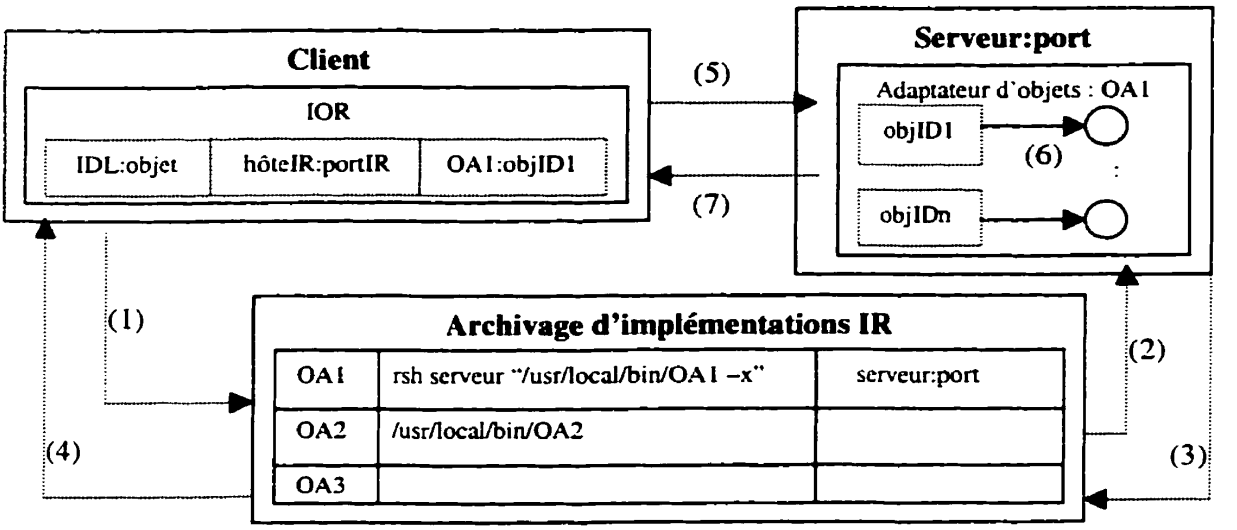


Figure 2.8 Attachement de références transitoires

2.3.4 Quelques caractéristiques de CORBA

Naturellement, une des solutions informatiques les plus contraignantes est celle des systèmes distribués. CORBA, ne faisant pas d'exception, se trouve confronté à plusieurs défis, surtout au niveau des fonctionnalités des ORB. Non seulement un ORB doit assurer la gestion et le multiplexage/démultiplexage des connexions de transport, mais aussi pouvoir effectuer des traitements concurrents d'une multitude de requêtes de clients. Dans un environnement hétérogène tel que CORBA, le traitement parallèle est

d'une importance primordiale puisqu'il permet la simplification de programmes, l'exploitation des puissances de calcul fournies par des systèmes multiprocesseurs et la réduction des délais de traitement. Idéalement, à chaque requête est attribué un processus léger (*thread*) disposant d'un certain nombre de ressources propres. Un *thread* est en fait une séquence unique d'instructions exécutées dans le contexte d'un processus (Barton et al., 1992). Au fur et à mesure que de nouvelles versions CORBA voient le jour, diverses stratégies de parallélisme sont intégrées dans les ORB (Schmidt, 1998).



Légende :
1. Connexion à l'archivage d'implémentations mentionné dans la référence d'objet
2. Activation du serveur sur l'hôte correspondant
3. Renvoi de l'adresse réelle du serveur à l'archivage d'implémentation
4. Retransmission de l'adresse du serveur au client
5. Connexion de l'ORB du client au serveur
6. Détermination du servant de l'objet évoqué
7. Remise des résultats d'exécution au client

Figure 2.9 Attachement de références persistantes

À côté du traitement parallèle, Henning (1998) évoque d'autres enjeux de la spécification CORBA comme ceux exposés ci-dessous :

- La mise en échelle : Des mesures préventives ont été proposées afin d'améliorer l'expansion de l'environnement d'objets distribués. Cette expansion fait face à deux

obstacles manifestés, d'une part, par la congestion résultant d'échanges excessifs de messages pour la localisation d'un serveur donné, et la taille limitée des archivages d'autre part.

- La migration d'objets : Théoriquement, un environnement distribué parfait ne présente pas de frontières entre les espaces ou les domaines d'adresses de ses objets, ce que l'infrastructure actuelle éprouve toujours de la misère à rendre réel.
- L'intégrité d'objets et de références : Le manque de fiabilité des communications entre systèmes hétérogènes peut être à l'origine d'une violation de l'intégrité des objets ou des références d'objets. Ceux-ci ne seront plus récupérables dans le système suite, par exemple, à des ruptures de connexion.

Cependant, l'architecture CORBA cherche toujours à maintenir sa viabilité avec l'évolution constante de la portée des applications. Trois fonctionnalités additionnelles sont prévues pour la version CORBA 3.0 (Vinoski, 1998). Il s'agit d'adaptateur d'objets plus flexibles (POA), de nouveaux modes d'échanges de messages, ainsi que des spécifications (OBV) pour passage de paramètres plus sophistiqués pour les opérations évoquées.

Finalement, il faut signaler la présence d'un compétiteur étroit de CORBA sur le plan des environnements d'objets distribués. Introduit par Microsoft en 1990, DCOM s'est inspiré principalement du modèle DCE établi par OSF et utilise les mécanismes DCE RPC pour les communications distribuées, ainsi qu'une version légèrement modifiée du DCE IDL pour la définition d'interfaces (Redlich, Suzuki et Weinstein, 1998). Depuis CORBA 2.1, l'interopérabilité de CORBA et DCOM fait partie intégrante de la spécification CORBA. Dans ce sens, des ponts statiques et dynamiques sont conçus pour assurer l'interconnexion de ces deux technologies.

Du point de vue structurel, CORBA et DCOM présentent plusieurs points communs, tout en ayant évidemment quelques différences. Par exemple et contrairement à DCOM,

CORBA n'exige pas un protocole unique pour l'implantation d'un ORB, laissant ainsi aux vendeurs leurs choix propriétaires d'implantations, à condition que l'interopérabilité globale soit respectée entre les divers ORB. Il revient donc au marché informatique de sélectionner un gagnant parmi ces deux architectures en se basant sur des critères variés tels que la facilité de transition à partir de ce qui existe et la conformité à l'orientation des applications sollicitées. Certaines recherches favorisent CORBA pour les opérations à temps réel à cause des solutions plus pratiques adoptées dans ce secteur.

Quant aux applications de CORBA, celles-ci sont nombreuses et couvrent plusieurs secteurs : bancaire, commercial, éducatif, médical, etc. À titre d'illustration, Tjandra et Wong (1999) présentent la conception d'une librairie numérique pour imagerie biomédicale manipulant des documents répartis sur plusieurs départements et hôpitaux. Cependant, une application classique de CORBA demeure l'administration de réseaux (Haggerty et Seetharaman, 1998 ; Bardout et al., 1998). Allant du simple, tel que SNMP, au plus abstrait, tel que CMIP, le concept d'administration de réseaux est basé sur une modélisation orientée objets des informations de gestion. Si la MIB représente une structure arborescente d'objets ou plutôt d'attributs décrits en ASN1, la description adoptée par l'ISO fonce plus profondément dans la théorie orientée objet. Toutefois, les acteurs sont les mêmes : des agents reportant les états des objets gérés qu'ils représentent et des gestionnaires surveillant l'ensemble des objets répartis. Avec l'avènement des réseaux de taille de plus en plus grande, des mécanismes plus spécialisés de communications distribuées sont indispensables. Dans tous ces scénarios, CORBA s'avère adéquat, et des produits sont déjà disponibles sur le marché, genre l'architecture d'administration ProSphere de GDC pour les réseaux ATM.

2.3.5 Complémentarité Java-CORBA

En dépit de la généralité du concept de Web orienté objet, la transition à partir du Web conventionnel préserve une simplicité surprenante, tant sur le plan structurel

qu'opérationnel. Cette innovation consiste en un environnement d'objets distribués d'étendue universelle, dans lequel on retrouve toujours des fureteurs et des serveurs. En outre, les langages de programmation qui ont contribué au succès des applications réparties sur le Web conserveront un rôle significatif lors de la mise en place du Web orienté objet. Ceci est bien le cas du langage Java.

Java est un langage orienté objet simple, portable et dynamique. Il présente plusieurs éléments communs aux langages C et C++, mais va plus loin en considérant la sécurité des ressources du système réservées lors de l'exécution du code Java. La particularité de ce langage provient du fait qu'il possède les avantages d'un langage interprété et également la performance d'un langage compilé (Anuff, 1996). La compilation d'un programme Java fournit un code intermédiaire, connu sous le nom de *Java Bytecodes*, susceptible par la suite d'être interprété sur une plate-forme quelconque. Un interpréteur est en fait la représentation de ce qu'on appelle machine virtuelle Java (JVM), dont l'implantation peut être réalisée au niveau matériel. Les outils de développement Java et les fureteurs ne sont autres que des interpréteurs, respectivement utilisés par les applications locales et les Applets Java. La Figure 2.10 illustre la mise en œuvre d'un code Java sur divers systèmes d'exploitation.

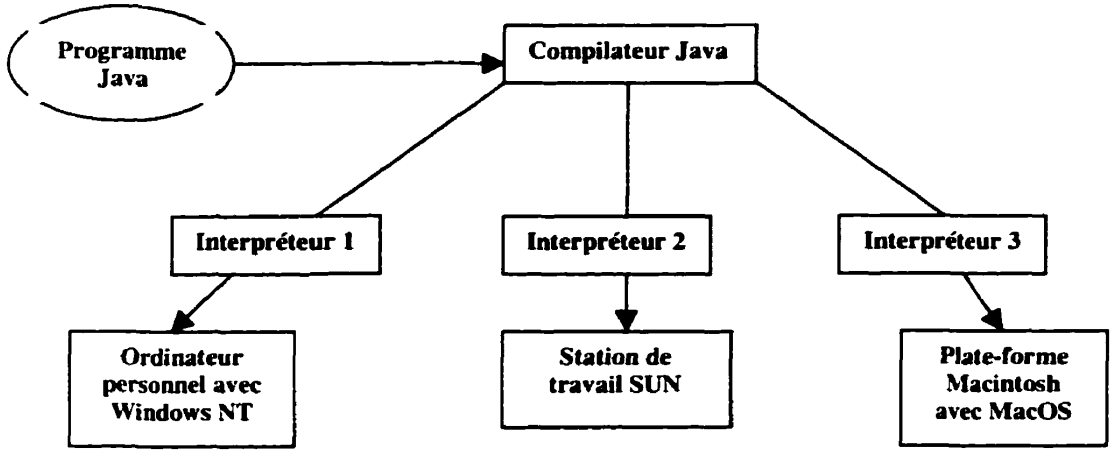


Figure 2.10 Mise en œuvre d'un code Java

À côté de la machine virtuelle Java, une interface de programmation ou API offre, à travers des librairies sous forme d'une hiérarchie de classes, plusieurs fonctionnalités utiles dont les composants d'interfaces utilisateurs graphiques GUI. L'association de l'interface de programmation et de la machine virtuelle Java conduit à une entité logicielle autonome indépendante du contexte dans lequel les applications Java sont lancées. La Figure 2.11 en est une illustration.

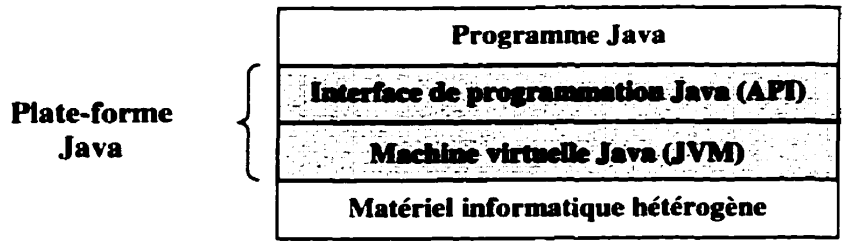


Figure 2.11 Cycle de programmation intégrant la plate-forme Java

Cependant, le langage Java tout seul s'avère une condition nécessaire mais pas suffisante pour le déploiement du Web de demain. Une infrastructure d'objets distribués est alors indispensable pour assurer un degré d'interaction valable. C'est dans ce contexte que l'architecture CORBA a fait son apparition. Java et CORBA sont donc deux parties qui se complètent. En effet, si CORBA assure l'interopérabilité ou la transparence au niveau des communications entre réseaux, systèmes et langages informatiques hétérogènes, Java réalise la transparence au niveau des implémentations.

Avec CORBA, Java n'est plus un simple compétiteur du langage HTML pour le développement de pages Web dynamiques. CORBA permet d'étendre le modèle d'objets de Java pour le rendre plus distribué. Cela veut dire que, tout en maintenant une interface légère, une Applet Java peut utiliser des composants répartis et déclencher des opérations dynamiques ou bien des transactions atomiques invoquant plusieurs entités. Bien plus, CORBA a pu introduire dans le Web un nouveau style de communications (*peer-to-peer*) entre les serveurs eux-mêmes, puisqu'un objet donné peut être à la fois

client et serveur. Par conséquent, il en découle, d'une part, un style de programmation Java d'objets offrant plus de services, et d'autre part des environnements sujets à de grandes mises en échelle.

En conclusion, la combinaison CORBA/Java nous apparaît bien convenir à la mise au point de certains aspects critiques pour l'implantation de la plate-forme de télécommunications. Comme d'autres applications réparties, les laboratoires virtuels trouveront non seulement un support efficace pour l'intégration d'outils spécifiques et génériques, mais également des interfaces d'accès souples et interactives. Le modèle conceptuel présenté au chapitre suivant aborde ces préoccupations.

CHAPITRE III
MODÈLE CONCEPTUEL DE LA PLATE-FORME DE TÉLÉCOMMUNICATIONS

Ce chapitre définit le modèle conceptuel de la plate-forme de télécommunications desservant des laboratoires virtuels distribués. Dans un premier temps, nous préciserons le contexte d'utilisation de cette plate-forme. Puis, nous en exposerons les spécifications techniques et fonctionnelles. Par la suite, nous en esquisserons l'architecture, pour finalement exposer les détails relatifs aux différentes couches qui composent cette plate-forme.

3.1 Contexte d'utilisation de la plate-forme de télécommunications

En dépit des multiples définitions qui lui ont été associées, le terme télé-apprentissage constitue en général une référence à l'utilisation des réseaux d'ordinateurs à des fins d'apprentissage (Collis, 1996). En tout temps, des apprenants encadrés par des formateurs, tuteurs ou professeurs, tous éventuellement dispersés dans l'espace et le temps, peuvent réaliser des activités de formation. D'où le concept de campus virtuel de plus en plus utilisé pour désigner cet environnement qui se veut similaire au campus traditionnel dit réel. Le défi qui se pose alors est d'intégrer autant que possible les attributs fondamentaux d'un campus réel dans des environnements d'apprentissage plutôt virtuels. En effet, si les connaissances sont suffisamment bien véhiculées à travers des cours et des documents textuels facilement livrables aux utilisateurs, les choses sont un peu plus complexes lorsqu'il s'agit d'un apprentissage pratique en laboratoire, somme toute nécessaire dans un grand nombre de disciplines scientifiques ou technologiques. Il est alors indispensable de recréer un concept générique de laboratoire,

et encore mieux, un environnement d'apprentissage distribué permettant d'effectuer des activités d'apprentissage à distance, similaires à celles ayant lieu dans les laboratoires conventionnels. Parmi les barrières à surmonter, mentionnons l'impossibilité de manipulation physique directe et les limitations des possibilités d'observation sensorielle, entre autres.

La méthodologie préconisée dans ce contexte consiste à définir et à développer, selon une approche inductive, différents types de laboratoires en fonction des caractéristiques spécifiques des champs de savoir ou des disciplines associées. Ensuite, les outils et fonctionnalités sur lesquels reposent ces divers laboratoires seront regroupés en deux grands ensembles : les outils et fonctionnalités de base (OFB) qui sont communs à plusieurs laboratoires, les outils et fonctionnalités spécifiques (OFS) qui sont plutôt propres à un laboratoire en particulier. Le caractère générique du modèle de laboratoire virtuel résultera du développement d'un nombre suffisant de laboratoires spécifiques, dont l'ensemble des fonctionnalités intégrées en constitue l'essence. Cependant, cet aspect générique repose principalement sur une plate-forme de télécommunications qui constituera un support concret pour les outils et fonctionnalités de base, comme l'illustre la Figure 3.1.

Outre l'accès aux diverses catégories de manipulations expérimentales et via ses interfaces, la plate-forme de télécommunications se charge de remédier au problème d'interopérabilité de réseaux. Ainsi, quels que soient les systèmes ou les moyens de communications dont ils disposent, les usagers parviendront à se servir de façon tout à fait transparente, de l'ensemble des facilités présentées par chaque laboratoire virtuel spécifique. Ainsi, toute une gamme de services partageables entre les divers laboratoires spécifiques est offerte aux utilisateurs, notamment des outils de traitement de texte, de calcul mathématique ainsi que des moteurs de simulation et d'analyse logique. Bref, selon chaque requête lancée par un utilisateur, la plate-forme de télécommunications renvoie les outils spécifiques adéquats accompagnés d'une copie de chaque outil

commun qu'elle possède. L'utilisateur est alors libre d'exploiter à sa guise les fonctionnalités qu'il juge utiles pour son expérience, sans avoir besoin de distinguer entre ce qui est de base et ce qui est spécifique. La mise en œuvre du modèle de la Figure 3.1 s'effectue par l'intermédiaire de deux prototypes de laboratoire : l'un en physique et l'autre en génie électrique. Par la suite, d'autres prototypes de laboratoires virtuels spécialisés peuvent être mis en place, une fois que les modules déjà implantés auront prouvé leur efficacité. Quel est alors le rôle de chacun des composants de ce modèle ?

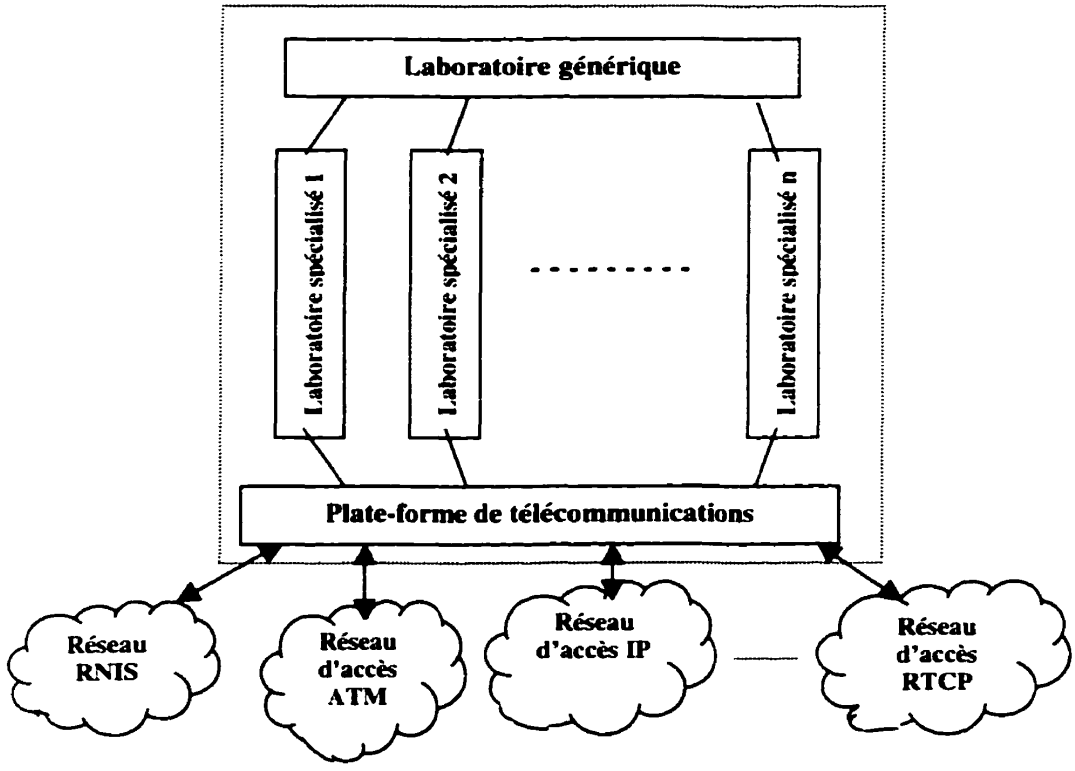


Figure 3.1 Architecture proposée pour les laboratoires virtuels

Le laboratoire de physique : Celui-ci fournit un environnement multimédia comprenant un certain nombre de simulations, munies de séquences filmées d'expériences réelles ainsi que d'explications animées et textuelles. Par ailleurs, la plupart des aspects

significatifs d'un laboratoire réel sont pris en compte, et les contraintes identiques à celles rencontrées en pratique sont conservées. Celles-ci portent, entre autres, sur l'imprécision de mesure, les limitations de spectre ou la précision des paramètres fixés par l'expérimentateur. Chaque simulation est susceptible d'inclure ses outils spécifiques, mais également d'autres outils communs empruntés au modèle générique de base. Il sera question, par exemple, de cahiers de laboratoire, d'instruments de mesure et d'analyse, et d'outils de calcul algébrique.

Dans une phase primaire, le laboratoire virtuel de physique intègre les fonctionnalités d'accès et d'affichage de documents multimédias, d'exploitation de manipulations interactives à distance, ainsi que des ressources de communications genre courrier électronique ou vocal. Par la suite, d'autres fonctionnalités plus sophistiquées seront envisagées. À titre d'illustration, on peut citer l'assistance en ligne tout au long de la durée d'expérience, l'élaboration d'outils de travail collaboratif et l'amélioration du contrôle et de l'échange à temps réel lors des expérimentations.

Le laboratoire de génie électrique : Inspiré en grande partie par des concepts incarnés dans le prototype de laboratoire de physique, le laboratoire de génie électrique maintient les mêmes perspectives d'assistance pédagogique et d'interaction des manipulations. En plus, ce laboratoire implante des concepts fondamentaux du génie électrique auxquels s'ajoute la dimension de télé-expérimentation. Il s'agit essentiellement d'effectuer des simulations moyennant des modèles numériques typiquement dédiés à l'acquisition, le traitement et l'analyse de signaux électriques. Ce genre d'expérience permet d'illustrer le fonctionnement d'équipements dispendieux, et aussi de mimer des processus qui ne peuvent pas être réalisés concrètement à cause des contraintes de temps ou du risque de danger que présente l'expérimentation réelle.

Outre l'environnement distribué d'applications assurant le partage de données entre les participants, le laboratoire virtuel en génie électrique offre d'autres catégories d'outils.

Ces derniers comportent des moyens nécessaires pour la vidéoconférence ainsi que de nouvelles manières de gérer, d'apprendre ou d'utiliser des systèmes de télécommande et de télémesure, favorisant spécialement des opérations à distance en temps réel de mécanismes réactifs. Dans cette perspective, ce laboratoire virtuel invoque des environnements de modélisation et de simulation pour l'exploitation de ressources matérielles et logicielles, partagées ou non, et qu'on retrouve dans les laboratoires traditionnels d'enseignement de génie électrique.

La plate-forme de télécommunications : C'est particulièrement grâce à elle que le projet de laboratoires virtuels doit en quelque sorte son originalité. Au-delà de sa raison d'existence primordiale en tant que fournisseur d'accès aux différents laboratoires virtuels spécialisés, cette plate-forme joue un rôle beaucoup plus important. En fait, c'est à ce niveau que les outils et fonctionnalités de base sont offerts aux utilisateurs. Ceci comprend plus précisément l'adaptation aux caractéristiques propres à chaque type de laboratoires spécialisés, et l'intégration de ces dernières aux outils destinés à l'usage commun entre tous les laboratoires. Il s'agit de fournir un cadre conceptuel et normatif aux échanges entre systèmes hétérogènes. Ceci sous-entend l'interopérabilité entre plusieurs classes de réseaux de communications, qui se distinguent non seulement par leurs architectures diversifiées, mais également par les techniques et les moyens matériels d'accès qu'ils mettent en jeu. Il revient donc à la plate-forme de télécommunications seule de pallier l'ensemble des problèmes d'incompatibilité due aux différences de protocoles, de formats de données, de débits de transmission, de modes de commutation, de formats d'adresse, et ainsi de suite.

En dépit de tout cela, la plate-forme doit être en mesure d'offrir une qualité de service digne des réseaux à haut débit de plus en plus recommandés. Elle doit rendre possibles des exécutions parallèles nécessaires au partage efficace de ressources d'instrumentation et de télémesure, souvent exploitées simultanément par divers utilisateurs. Il ne faut pas non plus oublier les tâches classiques qu'assume cette plate-forme : contrôle de

cohérence et d'intégrité des données transmises, contrôle de flux et d'erreurs normalement effectués par les nœuds d'un réseau de communications quelconque, etc.

Le modèle générique : Qualifié en partie d'ancêtre commun pour tous les laboratoires spécifiques, le laboratoire générique s'avère un élément indispensable à la mise en place du présent projet. Une telle modélisation consiste à rechercher et à conserver suffisamment de propriétés et d'attributs communs pour pouvoir les intégrer par la suite au sein des différents types de laboratoires. En d'autres termes, chaque laboratoire spécifique hérite d'un ensemble d'outils partageables et se voit doté également d'outils qui lui sont propres. Cependant, une approche inductive s'impose d'elle-même en vue de définir les constituants du modèle générique. Selon notre approche inductive, ceux-ci seront enrichis et modifiés au fur et à mesure qu'un nombre suffisant de laboratoires virtuels sera mis en place. En revanche, certains attributs peuvent être déjà identifiés pour constituer deux groupes principaux : les attributs externes et les attributs internes.

Le premier groupe recueille les propriétés à usage commun, dont les laboratoires auront besoin pour fonctionner et l'utilisateur pour se servir. Dans ce volet, six sous-groupes peuvent être cités :

- les outils de support : utilisés lors de la préparation et durant une expérience (tels que le cahier de laboratoire, etc.) ;
- les instruments de mesure : utilisés au cours d'une expérience (tels que la règle, le voltmètre, l'oscilloscope, etc.) ;
- le gestionnaire d'expérimentations à distance : servant d'interface entre les requêtes des clients et les réponses du serveur ;
- le gestionnaire de liaisons de bases de données : facilitant l'accès et la consultation des bases de données ;
- les outils de communications : permettant des échanges efficaces d'informations (tels que la messagerie électronique, les outils de vidéo et audioconférences, etc.) ;

- les outils de référence : utilisés au cours d'une expérience (tels que le tableau périodique des éléments dans un laboratoire de chimie, etc.).

Quant au second groupe d'attributs, il s'agit des propriétés associées à l'architecture permettant de présenter les expériences. Celle-ci est commune à tous les laboratoires virtuels, bien que chacun d'eux comporte des manipulations qui lui sont spécifiques. Les notions d'espaces d'expérimentation, d'expérience, d'interface et de système conseiller, font leur apparition au sein des attributs internes. Dans cette perspective, un attribut essentiel émerge et porte le nom de gestionnaire d'expériences. Celui-ci gère alors les cinq espaces proposés à l'expérimentateur, à savoir :

- l'espace *présentation* : présente l'expérience en elle-même (vidéos, images, etc.) ;
- l'espace *manipulation* : concerne la prise de mesures et la manipulation ;
- l'espace *analyse* : permet d'analyser les résultats obtenus à partir de l'espace *manipulation* ;
- l'espace *explication* : présente les explications des résultats ;
- l'espace *théorie et applications* : présente la théorie.

Au gestionnaire d'expériences s'ajoute le gestionnaire d'aide ou système conseiller. Ce dernier peut être décomposé en deux sous-systèmes :

- le système d'aide contextuel qui fournit à l'utilisateur une aide dans une situation donnée (telle que la prise de mesure avec un galon, etc.) ;
- le système de présentation qui fournit à l'utilisateur une aide sur la manière d'agir et de manipuler dans un laboratoire virtuel.

Ainsi, le modèle générique, malgré sa complexité, permet d'intégrer de manière cohérente un nombre important d'outils et de fonctionnalités de base partageables entre plusieurs entités éventuellement hétérogènes à travers une plate-forme de télécommunications évolutive et adaptative.

3.2 Spécifications techniques et fonctionnelles de la plate-forme

L'interconnexion de réseaux comportant une diversité de systèmes ouverts hétérogènes constitue un secteur très actif et particulièrement sensible. Les enjeux d'interopérabilité sont d'autant plus complexes qu'ils prennent appui sur l'informatique et les télécommunications. Situé au croisement de ces deux disciplines, le rôle associé à la plate-forme de télécommunications permet d'affecter un caractère dynamique au modèle proposé pour les laboratoires virtuels, surtout en ce qui concerne la flexibilité d'accès offerte aux utilisateurs en regard de la variété des réseaux dont ils disposent. Les spécifications techniques et fonctionnelles de cette plate-forme lui attribuent un statut de chef d'orchestre capable d'intégrer les outils et fonctionnalités de base et spécifiques, sous forme d'entités à part entière facilement livrables aux usagers distants. Cependant, l'ensemble des spécifications évoquées ci-dessus est mis en place selon une approche modulaire au niveau de la plate-forme. En d'autres termes, cette dernière est vue comme un regroupement de modules dédiés chacun à une tâche bien définie et faisant partie de l'interface des laboratoires. Ces modules se répartissent selon trois thèmes majeurs, à savoir :

- l'adaptation aux outils et fonctionnalités spécifiques ;
- l'intégration des outils et fonctionnalités de base ;
- l'adaptation aux réseaux de communications des usagers.

L'adaptation aux outils et fonctionnalités spécifiques : Au niveau fonctionnel, l'accessibilité correcte à tous les éléments propres d'un laboratoire spécialisé est impérative. Cela signifie que n'importe quelle manipulation sera en mesure de se réaliser avec le plus de fidélité possible chez l'expérimentateur. Les explications textuelles, les images animées et fixes, les graphiques ainsi que les données vocales, tout doit être reproduit à distance de façon identique au cas d'exécution locale. Ainsi, la plate-forme de télécommunications doit être en mesure, d'une part, de discerner et d'éviter les conflits éventuels entre les objets spécifiques requis par divers clients, mais d'autre part

d'effectuer certaines adaptations au niveau des outils de base qui sont fournis par le module suivant. Le deuxième volet de ce rôle est critique, puisqu'il est naturel que l'usage des objets partagés diffère esthétiquement d'une manipulation à une autre, nécessitant souvent une remise en forme de ces derniers conformément au contexte expérimental courant.

Du point de vue logiciel, ces exigences se traduisent par la capacité de la plate-forme, tant au niveau des interfaces que des langages de programmation dont elle se sert, de jouer sur quelques paramètres des outils de base en vue de rendre ces derniers plus adaptés au contexte spécifique. Ces exigences se traduisent aussi par la façon de gérer l'accès et la cohabitation des codes exécutables relatifs à chaque type de laboratoires virtuels.

Du point de vue matériel, la plate-forme doit contenir un espace de stockage, centralisé ou réparti, suffisamment grand pour pouvoir supporter tous les fichiers et documents relatifs à la multitude des laboratoires. De tels fichiers, générés par des compilateurs de langages de programmation variés et dans des contextes distincts, risquent d'avoir une taille gigantesque à cause de la nature multimédia des informations qu'ils renferment. Par ailleurs, il s'avère nécessaire d'avoir une vitesse de traitement plus ou moins élevée, au niveau du processeur et des boîtiers d'entrée/sortie des serveurs utilisés. Ceci permet ainsi des échanges rapides avec les réseaux des clients, et surtout un fonctionnement en mode parallèle plus adapté à ce genre de situations.

L'intégration des outils et fonctionnalités de base : L'aspect générique des laboratoires virtuels se manifeste essentiellement à travers ce module. Les spécifications fonctionnelles correspondantes portent sur la façon de rendre accessibles aux utilisateurs les outils et fonctionnalités à caractère commun jouissant d'une certaine flexibilité pour l'adaptation à un cadre expérimental plus spécialisé. Il s'agit donc de gérer l'intégration des copies de ces objets de base dans l'ensemble des objets envoyés en réponse à une

demande d'accès lancée par un usager distant. Qu'il soit générique ou spécifique, un outil doit maintenir son efficacité durant l'exécution sur le site client qui en a besoin. Cela veut dire que la mise en commun des outils et fonctionnalités de base doit avoir lieu sans dégrader les performances de tout ce qui est partagé, et également de tout ce qui lui est complémentaire et requis à titre spécifique.

Dans ce sens, les spécifications techniques que possède la plate-forme de télécommunications sont de nature purement logicielle. Une fois les fichiers et segments de codes relatifs aux objets spécifiques à un laboratoire virtuel identifiés et calibrés selon la manipulation requise, il revient aux éléments informatiques de la plate-forme de leur associer correctement des instances d'outils de base qui y sont implantés. L'articulation des outils spécifiques et génériques est confiée à une ou plusieurs interfaces de communications de la plate-forme supportant, entre autres, des protocoles divers de communications.

L'adaptation aux réseaux de communications des usagers : C'est à l'aide de ce module que la contribution pédagogique de laboratoires virtuels retrouve son sens et ses valeurs les plus riches. En plus d'être une simple interface vis-à-vis des utilisateurs distants, la plate-forme de télécommunications représente une clef de voûte en ce qui a trait à la fiabilité et à l'interactivité au sein de cette application de télé-enseignement. À ce stade, les spécifications fonctionnelles de la plate-forme se résument par l'interopérabilité entre les réseaux hétérogènes mis en jeu côté clients. Plusieurs notions en découlent. Au niveau physique d'une architecture de communications selon le modèle OSI, il s'agit d'assurer une adaptation aux formats des trames de données et surtout aux débits de transmission, ceux-ci étant susceptibles de varier significativement suivant les réseaux utilisés.

Puisque la plupart des échanges comportent des données multimédias souvent exigeant du temps réel, les facettes classiques qui émergent telles que l'isochronicité du trafic et

la synchronisation entre les différents flots de média d'une même session, trouveront ici une solution adéquate. À un niveau supérieur et toujours selon le modèle de référence OSI, le problème revient à interpréter le contenu des paquets transmis. Loin d'être une simple lecture, la présente tâche sous-entend le maintien de cohérence et d'intégrité des données échangées, la minimisation des pertes et des duplications de paquets dues en grande partie à la surcharge et à la congestion des liens, la garantie d'une transmission ordonnée des blocs de données, ainsi que la prévention ou la correction d'éventuelles erreurs dans ce qui est véhiculé. D'autre part, la plate-forme de télécommunications s'engage auprès des utilisateurs à respecter un seuil minimal de qualité de service, de sécurité et de transparence. Cela permet aux clients de jouir d'une reproduction quasi réelle des travaux qu'ils sont habitués à entreprendre dans les laboratoires réels. En ce qui concerne les spécifications techniques, l'interopérabilité souhaitée exige l'implémentation des architectures de communications correspondant aux différents réseaux utilisateurs, par exemple l'architecture TCP/IP pour l'Internet. En termes de composants matériels, la plate-forme doit posséder des cartes d'adaptation conformes aux techniques d'accès physiques évoquées par les diverses classes de réseaux mis en scène. Toutefois, les câbles et les moyens de connexion sont obligatoires pour raccorder la plate-forme à l'infrastructure de télécommunications existante.

Face aux diverses questions auxquelles la plate-forme doit remédier, cette approche modulaire possède de nombreux avantages, tels que :

- Indépendance : la plate-forme met ses ressources à la disposition des applications de laboratoires virtuels sans aucune forme de dépendance auprès d'une application particulière ;
- Portabilité : les outils mis en commun peuvent être ajoutés, retirés ou modifiés sans altérer le rendement général des laboratoires spécifiques ;
- Universalité : les utilisateurs distants ne sont pas contraints d'avoir une plate-forme particulière, et ne sont pas limités à des systèmes informatiques propriétaires ;

- Flexibilité : d'après cette organisation, chacun des modules de la plate-forme peut être conçu, implanté et amélioré sans le moindre changement dans les autres compartiments.

Par ailleurs, une telle approche modulaire permet un développement évolutif facilitant non seulement la conception et l'intégration progressives des outils et fonctionnalités de base, mais aussi l'adaptation de ceux-ci aux outils et fonctionnalités propres à chaque type de laboratoire spécifique. Le triple caractère évolutif, adaptatif et intégrateur de cette plate-forme en constitue l'aspect le plus novateur et le plus attrayant.

3.3 Modèle conceptuel de la plate-forme de télécommunications

L'objectif poursuivi ici est de concevoir une plate-forme offrant l'accès à un grand nombre d'utilisateurs au travers d'un réseau étendu et hétérogène. D'autres facteurs sont aussi pris en compte tels que les possibilités de supervision en ligne entre enseignants et apprenants, ainsi que les degrés de liberté dont jouit un utilisateur au moment où il effectue des expérimentations à distance en mode asynchrone. Dans cette perspective, le modèle conceptuel de cette plate-forme dispose d'une entité unique assurant la coordination des différentes tâches requises pour assurer la souplesse de manœuvre des applications. Ce genre d'entités tient toutefois à respecter les formalités protocolaires et techniques au sens d'une architecture de télécommunications.

3.3.1 Description de la structure proposée

Conceptuellement, la notion de laboratoire virtuel fait référence à un environnement complexe intégrant, de manière cohérente, un nombre assez important de fonctionnalités d'expérimentation traditionnelles devant être développées en lien direct avec les possibilités actuelles ainsi que potentielles qu'offrent les réseaux de communications. Un tel développement repose sur la plate-forme de télécommunications qui se tient à mi-

chemin entre les utilisateurs distants, d'une part, et les catégories de manipulations pratiques offertes par les laboratoires virtuels spécifiques, d'autre part. Quelle que soit la forme, voire le contexte, dont se sert un client pour lancer sa requête, la structure de télécommunications proposée doit être en mesure de lui faciliter l'accès aux connaissances scientifiques adéquates, ainsi qu'aux instruments de télémesure appropriés. À titre d'illustration, la Figure 3.2 renseigne plus concrètement sur l'emplacement de celle-ci par rapport aux laboratoires spécifiques et aux réseaux d'accès.

Indépendamment de ses exigences matérielles en termes de composants de stockage, de traitement, de lecture/écriture magnétique ou optique, de raccordement aux interfaces et de commutateurs de réseaux auxquels elle est reliée, la plate-forme de télécommunications puise toute sa puissance d'une entité logicielle chargée de gérer aussi bien l'adaptation aux réseaux que l'intégration des fonctionnalités spécifiques et génériques des laboratoires. Évidemment, cette entité occupe une ou plusieurs stations de travail ou serveurs de l'institution universitaire chargée de fournir l'accès aux laboratoires virtuels. Cependant, on se demande où se trouve exactement cette entité ?

Selon le modèle proposé, une requête lancée par un client au travers d'une pile de protocoles d'un réseau donné remonte les couches correspondantes implantées sur un serveur avant d'atteindre l'entité logique où elle sera interprétée. Une fois le contenu d'une requête identifié, l'entité logique fera appel aux codes constituant les outils spécifiques du laboratoire virtuel en question, puis elle leur associe une série d'outils de base après quelques ajustements éventuels de leurs paramètres d'adaptation. Le tout repassera par l'architecture de communications sous-jacente en vue d'être acheminé au bon destinataire.

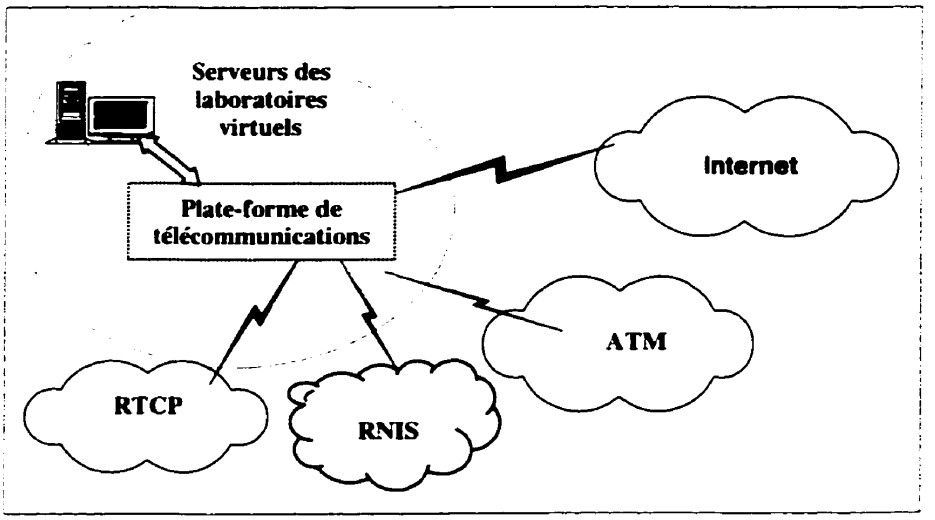


Figure 3.2 Emplacement de la plate-forme de télécommunications

3.3.2 Architecture en triple couches

La cohabitation des trois modules fonctionnels précédemment introduits est masquée par une entité logique méritant en quelque sorte la qualification de cerveau de la plate-forme. Une question se pose alors, comment les trois modules fondamentaux sont-ils implantés au sein de cette entité ?

En examinant de près les responsabilités de chacun de ces modules, certaines évidences se dégagent. Au niveau interne, les trois modules possèdent une autonomie de fonctionnement, tout en maintenant au niveau global une sorte d'harmonie imposée par les échanges de données. Lors de l'adaptation aux outils et fonctionnalités spécifiques, l'intervention auprès des outils communs du module de base se fait en leur passant des valeurs pour certains paramètres auxquels l'accès est autorisé. Cette opération s'effectue sans aucune modification au niveau de l'organisation interne de ces objets partagés. Une fois les outils génériques adaptés aux outils spécifiques, ils n'ont pas à se soucier de la structuration des programmes mettant en place les manipulations spécifiques sur

lesquelles portent les divers laboratoires. Enfin, lorsque l'ensemble des outils est remis au module de communications de la plate-forme, la transmission s'effectue indifféremment de l'enchevêtrement des outils spécialisés et communs envoyés à l'utilisateur. Ils seront perçus à ce niveau comme des blocs d'octets mis dans des paquets ayant une destination ainsi qu'une suite de protocoles de communications bien définies.

Donc, cette corrélation globale qui apparaît entre les trois modules de la plate-forme de télécommunications permet de conclure que ces modules collaborent en cascade. Ce mode opératoire est dominant dans les modèles d'architecture de communications situées à proximité de l'entité logique de la plate-forme, que l'on désigne communément par organisation en couches. Chacune de ces couches est dédiée à accomplir un rôle déterminé, tout en rendant service à la couche qui lui est supérieure et demandant service à celle qui lui est inférieure. La similitude constatée en comparant les modes opératoires des couches d'un modèle classique de communications à ceux des trois modules constituant l'essence de la plate-forme, inspire le modèle conceptuel de l'entité logique en question. Par conséquent, on a affaire à une architecture en trois couches, équivalente chacune à un module logique de la plate-forme de télécommunications proprement dite, comme l'illustre la Figure 3.3.

Les avantages de cette approche sont ceux qu'on peut énumérer pour n'importe quelle architecture multicouche, à savoir :

- Traitement organisé : la complexité d'un problème initialement posé en une seule pièce se voit de plus en plus réduite au fur et à mesure que chaque couche en traite une partie, ce qui facilite les procédures de traitements indépendants et parallèles.
- Séparation entre implémentation et spécification : les fonctions attribuées à une couche à travers une spécification ou une recommandation quelconque n'exigent aucun moyen d'implémentation particulier pour qu'elles soient mises en place et exécutées.

- Réutilisation de fonctionnalités : un service fourni par une couche inférieure peut être partagé entre plusieurs couches supérieures. Ceci est bien le cas de la mise en commun d'une liste d'outils génériques pour les laboratoires virtuels spécifiques, quelle que soit leur discipline scientifique.

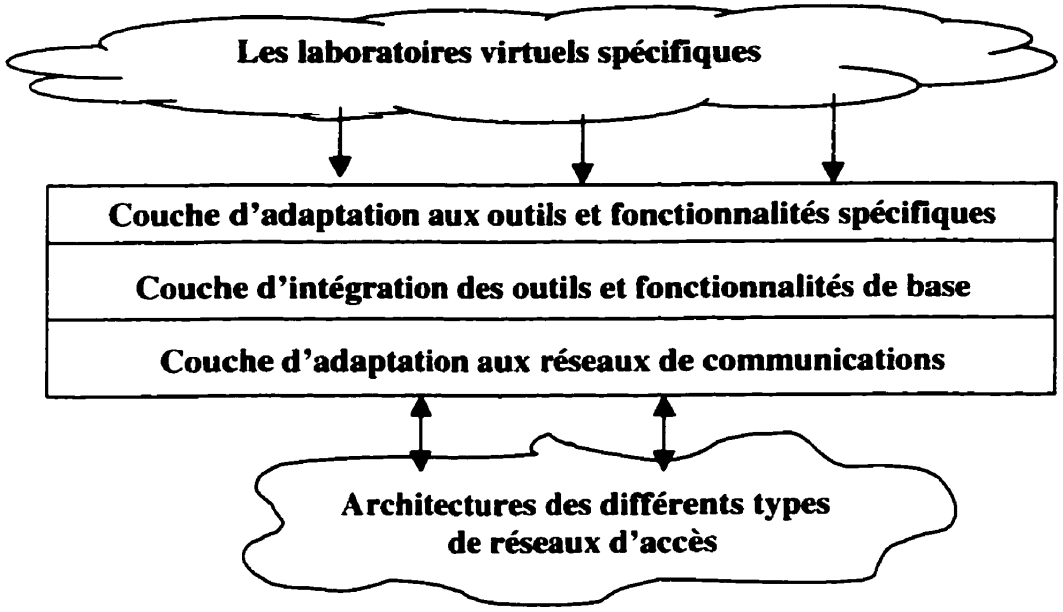


Figure 3.3 Architecture en triple couches de la plate-forme de télécommunications

3.4 Organisation des différentes couches du modèle

Le développement rapide des technologies de communications de plus en plus hétérogènes a conduit à un nouveau contexte dans lequel deux défis majeurs surgissent : l'interopérabilité et l'optimisation de ressources. Ces deux thèmes représentent brièvement les éléments de base dont notre modélisation de la plate-forme de télécommunications tente de mettre en œuvre. Ainsi, la conception et par conséquent l'implantation portent sur la structure individuelle de chacune des composantes de l'architecture en triple couches, ainsi que sur ses interfaces internes et externes.

3.4.1 Adaptation aux réseaux de communications

L'explosion du multimédia et la croissance des télécommunications fixes et mobiles ont contribué à la mise en place de nouvelles perspectives pour le développement de systèmes distribués. La présence d'une couche d'adaptation aux réseaux fournit alors le moyen approprié pour l'accessibilité à l'environnement des laboratoires virtuels.

Du côté des utilisateurs, la plate-forme de télécommunications ne voit en face d'elle que des réseaux de caractéristiques matérielles et logicielles tout à fait diversifiées. Telle qu'elle fut présentée dans la section précédente, cette couche peut être considérée comme un intermédiaire entre les deux sous-groupes de protocoles de communications, ceux de proche-en-proche et ceux de bout-en-bout définis dans le modèle de référence OSI. La première catégorie s'étale sur la couche physique, la couche de liaison de données ainsi que la couche réseau. Quant à la deuxième catégorie, elle comprend les trois couches les plus élevées de session, présentation et application, à côté de la couche transport. Ici, les traitements se font particulièrement en se basant sur les attributs relatifs aux paquets de données.

La couche d'adaptation aux réseaux de la plate-forme regroupe des fonctionnalités d'un niveau presque équivalent à celui de la couche transport. Bien sûr, les fonctions qu'offre la couche d'adaptation ne remplacent pas celles de la couche transport standard, mais viennent en complément et visent principalement l'interopérabilité entre des types variés de réseaux d'usagers. Il est alors clair que les services de cette couche sont indépendants du protocole de transport relatif à l'architecture de communications de chaque réseau utilisé. La couche d'adaptation effectue également des tâches complémentaires telles l'administration et la sécurité vis-à-vis des ressources appartenant à l'ensemble des laboratoires.

Afin de réaliser les fonctions d'interopérabilité entre les réseaux hétérogènes qui l'entourent, la couche d'adaptation doit répondre aux exigences suivantes :

- Opération en mode parallèle : la méthode idéale pour supporter des accès simultanés de la part de plusieurs usagers consiste en un traitement parallèle. Dès qu'elle reçoit une requête, la couche d'adaptation aux réseaux déclenche un processus propre capable de la traiter sans interférences avec les autres processus en cours ;
- Choix d'une architecture de communications adéquate : un client disposant d'un accès ATM ne peut pas être rejoint à travers une suite TCP/IP par exemple. La couche d'adaptation aux réseaux doit évoquer, pour chaque utilisateur, les interfaces et les intergiciels encapsulant les protocoles de communications que son réseau supporte et comprend ;
- Identification des utilisateurs : le site des laboratoires virtuels comporte en fait une partie publique et une partie privée. La première expose des informations pédagogiques et culturelles renseignant sur l'objectif général des laboratoires virtuels. Quant à la deuxième, elle fournit à des étudiants déterminés l'accès aux manipulations virtuelles. La distinction entre les deux catégories d'utilisateurs exige l'identification de chacun d'eux, de son adresse et des permissions qui lui ont accordées. Ce genre d'information est alors maintenu sur la plate-forme, consulté et mis à jour indépendamment des moyens de communications exploités par un client ;
- Garantie de qualité de service : de par la classification qui lui est attribuée à l'échelle des couches d'une architecture de communications, la contribution en termes de qualité de service que la couche d'adaptation peut procurer se manifeste par l'adaptation, d'une part, au taux de réception auquel les objets spécifiques et génériques des laboratoires virtuels lui sont remis, et d'autre part à la vitesse de transmission du support physique raccordant le réseau de chaque utilisateur. Pour un accès au moyen de lignes téléphoniques ou ADSL, le débit conventionnel est de l'ordre de quelques centaines de Kbps. Mais, pour un réseau ATM, quelle que soit la classe de trafic négociée par le client lors de l'établissement de la liaison CBR ou ABR, la vitesse de transmission peut atteindre les Mbps. Au niveau de la couche

d'adaptation, la méthode envisagée pour ce genre de situations consiste surtout en des mécanismes de compression et de décompression, appliqués aux données respectivement avant et après le passage par la pile adéquate de protocoles de communications standards du réseau utilisateur en question. Les algorithmes ainsi que les taux de compression varient éventuellement selon la nature de chacun de ces réseaux.

Dès qu'un client réclame une manipulation particulière et sélectionne les paramètres qui vont avec, la couche d'adaptation de la plate-forme lui attribue une session où tous les échanges ultérieurs concernant son expérimentation puissent avoir lieu d'une manière aussi simple que transparente. L'utilisateur se contente de l'exécution animée de son expérience sans s'apercevoir de tous les événements qui ont lieu ni sur la plate-forme ni même sur sa machine.

Dans un sens ou dans l'autre, les données utiles qui se trouvent au niveau de la couche d'adaptation de la plate-forme verront leurs formats changés au fur et à mesure qu'elles transitent d'une couche à l'autre à l'intérieur de la pile de protocoles de communications propres au réseau de l'utilisateur destinataire. À titre d'exemple, les datagrammes de données reçus sur une carte Internet vont être désencapsulés et débarrassés des entêtes des couches MAC, IP et TCP, avant d'atteindre les interfaces de la couche d'adaptation aux réseaux, telles que HTTP et RMI. Réciproquement, une fois que la couche d'adaptation a rempli ses fonctions vis-à-vis des objets spécifiques et de base recommandés par l'utilisateur, les blocs de données utiles seront encapsulés et se verront dotés des entêtes relatifs aux couches de l'architecture du réseau de communications évoqué. Dans le cas d'un réseau ATM, les données utiles, éventuellement sous forme de paquets IP, sont groupées d'abord sous forme de segments munis des entêtes de la couche AAL. Par la suite, ces segments sont découpés en cellules de cinquante trois octets incluant les cinq octets d'entête de la couche ATM proprement dite. Le flot de

cellules atteint enfin l'utilisateur distant à travers les commutateurs et le réseau ATM approprié.

Outre l'interopérabilité, la couche d'adaptation aux réseaux tient compte aussi de divers aspects existant en partie dans n'importe quel environnement de réseaux, à savoir :

- la gestion des requêtes de clients ;
- la sécurité ;
- l'administration des ressources de laboratoires virtuels ;
- l'évolutivité.

La gestion des requêtes de clients : Devant la possibilité d'accès simultanés de plusieurs usagers à une même manipulation ou à des manipulations différentes, la couche d'adaptation doit être en mesure de gérer l'ordre des traitements demandés. Pour les requêtes provenant d'une même architecture de communications, la règle du premier arrivé premier servi semble être convenable. Cependant, l'affaire se complique dans le cas de requêtes reçues en même temps, mais à travers des piles de protocoles de communications différentes. Dans ce cas, la couche d'adaptation procède à une distribution prioritaire du trafic selon les qualités de service offertes par chaque type de réseaux. À titre d'illustration, considérons deux clients qui déclenchent au même moment l'ouverture de deux sessions, l'un à travers un lien ATM et l'autre à travers un lien Internet via le réseau téléphonique. Le premier bénéficie d'un statut plus prioritaire que le deuxième, essentiellement parce que la qualité de service revêt ainsi une plus grande importance. Elle se manifeste, par exemple, à l'aide de la limitation du délai de transmission qu'offre la largeur de bande en ATM, de loin supérieure à la capacité des lignes téléphoniques. Il convient alors, à la couche d'adaptation aux réseaux, de satisfaire le premier client, ensuite le deuxième, ce dernier pouvant tolérer un certain retard plus ou moins habituel pour la majorité des transactions effectuées à travers l'Internet.

La sécurité : On entend par sécurité l'aptitude de la couche d'adaptation aux réseaux de la plate-forme de télécommunications à isoler les ressources critiques et à les protéger de toute intrusion non admise. Dans notre contexte, les ressources critiques dont on entend parler ne sont autres que les outils et fonctionnalités tant spécifiques que de base auxquels l'accès reste sensible et qui doivent être contrôlés. La couche d'adaptation jouit dans ce sens d'un pouvoir non négligeable, puisqu'elle constitue l'unique porte par laquelle les clients parviennent à utiliser les laboratoires virtuels spécifiques. Elle doit donc être en mesure de prévenir les interférences éventuelles entre les trafics de différentes sessions pouvant avoir un impact sur les objets stockés localement sur le serveur. D'autre part, la couche d'adaptation doit veiller à ce que les requêtes des utilisateurs ne puissent pas directement ou indirectement modifier les structures ou menacer l'intégrité des données propres aux laboratoires virtuels. Une solution envisagée consiste à respecter l'atomicité de chaque accès. Pour cela, la plate-forme traite une seule requête à la fois, et les copies des objets requises sont transmises correctement toutes ensemble, ou bien elles ne sont pas livrées à la destination. Dans ce dernier cas, il y a échec de la demande globale d'accès de l'utilisateur en question.

Par ailleurs, il s'avère aussi nécessaire d'implanter des mécanismes d'authentification des usagers distants. La sélection des utilisateurs admissibles peut être raffinée jusqu'au niveau des simulations individuelles. Dès qu'une requête est reçue, la couche d'adaptation n'autorise pas le déclenchement d'une session d'expériences, avant que l'utilisateur ne soit complètement reconnu.

L'administration des ressources de laboratoires virtuels : Dans le contexte d'administration de réseaux, la plate-forme de télécommunications est susceptible d'abriter un ou plusieurs agents. Chacun correspond à une des architectures de communications qui y sont implantées et demeure donc accessible par leurs protocoles d'administration standards genre SNMP ou CMIP.

Un agent a naturellement un double rôle. De un, il doit être en mesure de répondre aux requêtes lancées par le gestionnaire du réseau en question concernant les états des ressources distribuées constituant notre plate-forme. De deux, il doit être capable de notifier le gestionnaire lors de l'occurrence d'événements suspects sur une composante quelconque de la plate-forme. La couche d'adaptation aux réseaux ne jouit pas du statut d'agent bien sûr, mais peut intégrer tous les moyens permettant de gérer et d'avoir des statistiques sur l'ensemble des outils des laboratoires virtuels. À titre d'exemple, on peut citer la fréquence d'accès à une manipulation particulière, ou encore la moyenne pour un paramètre donné, calculée aux détriments de ses valeurs remises comme résultats par les utilisateurs durant une séance d'expérimentation donnée.

L'implantation de l'application d'administration des ressources des laboratoires virtuels se fait en ajoutant, dans le cadre de cette couche, des applications munies d'interfaces et de mécanismes standards de gestion de réseaux compatibles avec les architectures de communications utilisées. Les modules et systèmes informatiques groupés sous le titre de plate-forme de télécommunications peuvent alors implémenter des entités propres de gestion, comme la MIB pour une administration SNMP. De cette façon, les objets propres aux laboratoires virtuels seront à la portée des gestionnaires éparpillés sur un ou plusieurs réseaux locaux de l'unité académique donnée.

L'évolutivité : L'accroissement constant du nombre et de la taille des réseaux interconnectés à la plate-forme de télécommunications, ainsi que les changements rapides au niveau de l'environnement supporté imposent à la couche d'adaptation de la plate-forme en particulier un certain niveau d'évolutivité et d'adaptabilité. Le problème soulevé ici est donc celui du nombre et du type d'interfaces desservant l'ensemble des utilisateurs. Mais aussi, la couche d'adaptation aux réseaux, dans tous ces éléments, doit présenter une certaine flexibilité vis-à-vis de la multiplicité des protocoles de communications sous-jacents. Un maximum de charge et donc de vitesse de traitement

doit être possible sans toutefois dégrader les performances des applications de télé-expérimentation.

3.4.2 Outils et fonctionnalités de base

L'un des objectifs souvent poursuivis par les concepteurs d'applications consiste en l'optimisation des ressources. Dans cette perspective, l'existence de la couche d'outils et fonctionnalités de base de la plate-forme de télécommunications nous apparaît d'une grande importance. Classée deuxième dans l'architecture conceptuelle de l'entité logique de la plate-forme, cette couche fournit une liste d'outils de base communément intégrés aux différents laboratoires virtuels spécifiques. Que veut-on dire par outils et fonctionnalités de base ?

Afin de comprendre ce que signifient ces termes, une analogie avec les environnements de programmation orientés objet nous semble opportune. Ce qui est partagé entre les laboratoires spécifiques peut être assimilé à des groupements de structures possédant chacun un thème ainsi que des propriétés qui le distingue des autres. Chaque structure, de par ses qualifications, est susceptible d'être parfaitement vue, dans une optique orientée objet, en tant qu'une classe dont les instances sont dotées d'attributs propres et de méthodes de manipulation exclusives. À titre d'exemple, la fonctionnalité de prise de mesure se traduit par un groupement de classes correspondant chacune à un instrument de mesure précis tel qu'un baromètre, un rapporteur ou un voltmètre. À ce moment là, les outils ne sont autres que les objets dérivés de ces classes. Par conséquent, chaque outil possède des attributs et des méthodes d'ordre privé faisant partie de procédures qui lui sont tout à fait internes et invisibles aux utilisateurs. Celles-ci sont surtout fondamentales à tout outil générique afin qu'il puisse prouver un caractère commun en même temps qu'un fonctionnement autonome au sein de n'importe quel laboratoire virtuel spécifique tournant sur un site client. Or, certains outils de base requièrent parfois des moyens de contact avec d'autres objets éparpillés dans un même cadre expérimental.

Ce type d'interactions peut être réalisé à l'aide d'attributs et de méthodes publiques définies également dans les classes dont dérivent ces outils et invoquant parfois des variables d'environnement de serveurs.

Il convient de noter que toute remise à niveau des outils de base, et plus généralement toute forme de contact prévu entre eux et l'environnement pratique au sein duquel ils sont introduits, ne met pas en cause l'aspect générique du modèle proposé. Par contre, c'est un moyen qui facilite leur adaptation et leur accorde une souplesse et une flexibilité, peu importe la nature du laboratoire virtuel spécifique auquel accède un utilisateur. Ceci justifie la mise en œuvre, pour les outils de base, des classes ou structures munies d'un certain nombre de degrés de liberté et de paramètres de réglage, aptes à préserver le principe d'optimisation de ressources, sans toutefois perdre l'utilité ou dégrader la qualité souhaitée dans chaque domaine spécifique considéré.

Ainsi, le rôle de la couche d'outils et fonctionnalités de base devient de plus en plus clair et consiste, de façon primordiale, en l'implantation desdits outils génériques. Les objets résultant, ainsi que les outils spécifiques recommandés, sont par la suite transmis par l'intermédiaire de différentes interfaces jusqu'à leur destinataire.

Au fur et à mesure que de nouveaux laboratoires spécifiques seront développés, la liste des outils de base deviendra plus riche. À une étape donnée, aucune forme finale pour les éléments partageables offerts par la couche d'outils et fonctionnalités de base ne prétend être établie une fois pour toute. Ceci découle du fait que l'ajout d'un nouveau laboratoire spécifique est susceptible, non seulement d'amorcer la création de nouvelles fonctionnalités plus sophistiquées, mais aussi d'affecter les structures de base déjà en opération pour ce qui existait auparavant. La présence de deux prototypes de laboratoires spécifiques permet actuellement d'effectuer un inventaire primitif des outils et fonctionnalités requises. Ces dernières sont consommées soit à égalité par les deux

prototypes, soit pour le moment par l'un d'eux, mais dont la présence sera nécessaire pour des prototypes potentiels dont le développement est planifié dans le futur proche.

Regroupés par thème, des structures voire des classes d'objets peuvent être énumérées pour chaque fonctionnalité, à savoir :

- la prise de mesure ;
- le calcul mathématique ;
- le traitement de texte ;
- l'appareillage électrique élémentaire ;
- les communications ;
- l'interaction locale.

La prise de mesure : Évidemment, celle-ci couvre une gamme diversifiée d'instruments de mesure pour toutes les grandeurs physiques répandues dans les laboratoires scientifiques. En partant de ce qui est déjà en place, certains outils de mesure peuvent être extraits des cadres spécifiques de physique et de génie électrique pour servir plus tard au sein des laboratoires virtuels à venir. Parmi ces outils, on distingue :

- des outils de mesure géométrique, tels que la règle pour les longueurs et le rapporteur pour les angles ;
- un outil de mesure de temps qu'est le chronomètre ;
- des outils de mesure de grandeurs atmosphériques tels que le thermomètre pour la température et le baromètre pour la pression ;
- des outils de mesure de masses tels qu'une balance ;
- des outils de mesure de grandeurs électriques tels qu'un multimètre, un ampèremètre pour le courant, un voltmètre pour la tension et un ohmmètre pour les résistances.

Le calcul mathématique : N'importe quelle discipline scientifique fait appel à des outils de calcul mathématique en allant des opérations simples aux systèmes matriciels complexes. Bien sûr, le type des équations et des méthodes de résolution varie

significativement d'un laboratoire à un autre. À titre d'illustration, la couche d'outils et de fonctionnalités de base offre les objets suivants :

- une calculatrice disponible à tout instant ;
- des logiciels mathématiques sophistiqués tels que Matlab, Mathematica et Maple. Ces outils munis de moyens de calcul algébrique puissant et de facilités graphiques poussées, peuvent être lancés dans des fenêtres propres sur la console de l'utilisateur. Dans un premier temps, l'usager leur communique manuellement les données de son expérience en laboratoire. Dans le cadre des développements futurs, cette acquisition sera accomplie automatiquement à partir de la fenêtre dans laquelle se déroule la manipulation virtuelle.

Le traitement de texte : Dans un laboratoire conventionnel, un étudiant doit remettre à son enseignant un rapport portant sur les tests, les résultats ou d'autres remarques importantes concernant le déroulement de la séance d'expérimentation. Cette rubrique porte principalement sur le cahier de laboratoire, dont la présence est exigée durant n'importe quelle manipulation. L'organisation ainsi que la mise en page de ce rapport sont sujettes à des modifications, selon les exigences de chaque laboratoire spécifique. Le développement ultérieur des codes des modules logiciels impliqués rendra possible la redirection spontanée des résultats, obtenus à l'issue des réalisations virtuelles, automatiquement vers les cahiers de laboratoire correspondants.

L'appareillage électrique élémentaire : Quelle que soit l'orientation scientifique d'un laboratoire, il est fréquent d'y trouver des appareils électriques contribuant au fonctionnement des systèmes et des équipements spécialisés. Parmi les outils de cette catégorie, on peut reconnaître :

- une alimentation électrique stabilisée à courant et tension réglables ;
- des interrupteurs pour déclencher ou arrêter l'alimentation de processus électriques, mécaniques ou autres ;
- des générateurs de signaux à basse ou haute fréquence.

Les communications : Au cours d'une séance d'expérimentation arbitraire, les étudiants ont besoin de communiquer entre eux ou bien avec leurs enseignants, à des fins de vérification, de commentaire et d'explication des problèmes posés ainsi que des résultats obtenus. De même, à la fin de la séance, les étudiants doivent remettre leurs rapports à leurs professeurs. Ceci doit être aussi pris en compte lors de l'inventaire des objets de base. En réalité, ces outils sont fournis par le fureteur ou le système lui-même, en dehors des composantes privées des laboratoires virtuels. C'est le cas de la messagerie électronique, de l'outil de communications interactives "*Chat*", ou bien d'autres commandes d'échanges de messages offertes par les systèmes d'exploitation installés sur les sites clients, genre la commande "*Write*" de UNIX. Une implémentation redondante exclusivement pour les laboratoires virtuels, d'outils de communication similaires à ceux énumérés auparavant, est inutile puisqu'ils sont inhérents aux programmes clients, et leur rendement est largement approuvé par le public. Cependant, leur exploitation peut être mise en relief à travers des astuces de l'interface graphique, affichables sur la console de visualisation des utilisateurs.

L'interaction locale : À travers sa console où se déroule l'expérimentation voulue, l'utilisateur jouit des moyens d'interaction avec son propre site. Cette catégorie d'objets de base comprend un outil d'agrandissement par zones de l'écran (*Zoom*), ainsi que des outils d'enregistrement des résultats textuels et graphiques. Ces derniers n'ont pas besoin d'une implémentation spéciale puisqu'ils sont déjà fournis par le système d'exploitation de la machine, mais des boutons spéciaux de la console permettent à l'usager de les activer sans avoir à revenir explicitement aux mécanismes conventionnels d'accès au système de fichiers local.

Que ce soit dans ces catégories ou dans d'autres, de nouveaux objets sont prévus mais ne peuvent apparaître sur la liste commune, avant d'être sollicités dans les milieux spécifiques où leur usage s'avère nécessaire. En fait, l'établissement de l'ensemble des

outils de base, construit à partir d'une approche inductive demeure un objectif ultime qui dépasse le cadre de ce projet.

3.4.3 Adaptation aux outils et fonctionnalités spécifiques

Il est évident que l'optimisation de ressources, mise en relief à travers la couche d'outils et de fonctionnalités de base, est un atout primordial pour cette application. Pourtant, la performance souhaitée se détériore significativement si ces ressources s'avèrent inutiles dans des situations expérimentales particulières, et incapables par suite de s'adapter au contexte spécifique dans lequel une place leur a été réservée. L'adaptation aux outils et fonctionnalités spécifiques est une condition nécessaire pour une expérimentation efficace vis-à-vis des utilisateurs, mais aussi pour un modèle réussi du point de vue des concepteurs. C'est ce que cette couche placée en tête de la plate-forme de télécommunications réalise.

Tout en maintenant une perception de classes et d'objets pour identifier les fonctionnalités et outils, la couche d'adaptation aux outils et fonctionnalités spécifiques ne présente pas de nouvelles classes ni de nouvelles structures. Elle constitue plutôt un pont à travers lequel les codes effectifs des laboratoires virtuels spécifiques se dotent de l'ensemble d'objets de base pour trouver ensuite la bonne voie conduisant aux utilisateurs distants. Par conséquent, cette couche veille à ce que les objets de base se débarrassent d'un mode de fonctionnement figé et adoptent surtout un aspect conforme au laboratoire spécifique au sein duquel ils sont insérés. À titre d'exemple, le réglage de l'échelle de pesée d'une balance est indispensable lorsqu'on passe d'un laboratoire de physique, où les masses mises en jeu varient entre quelques grammes et des centaines de kilogrammes, à un laboratoire de chimie où la matière manipulée, souvent de l'ordre des milligrammes, ne peut dépasser les dizaines de grammes. Ou encore, l'échelle de temps d'un chronomètre, réglée à des micro ou millisecondes pour des signaux de contrôle régulièrement déclenchés dans les processus d'automatisme en génie électrique, doit être

initialisée de nouveau lorsque ce chronomètre enregistre la durée d'action d'une enzyme ou d'un catalyseur dans un métabolisme biologique.

Avec plus d'abstraction, les programmes simulant les séances de laboratoires multidisciplinaires, générés souvent au moyen de logiciels tout à fait différents, sont perçus eux-mêmes en tant qu'objets ou instances de classes de codes exécutables correspondant chacune à une manipulation particulière. Bien sûr, cette approche orientée objets est tout à fait extensible pour prendre en compte des notions plus profondes envisagées pour des étapes ultérieures de ce projet. Il est question, par exemple, d'héritage établi entre des super-classes définissant les éléments essentiels d'un laboratoire générique et des sous-classes qui leur ajoutent d'autres caractéristiques plus raffinées facilitant la mise en scène de simulations de plus en plus spécifiques. Bref, la couche d'adaptation aux outils et fonctionnalités spécifiques a affaire à des objets distribués de ses deux côtés. En amont, ce sont donc les exécutables et les librairies supplémentaires des expériences disponibles, et en aval se trouvent les objets de base distribués et livrables dès l'occurrence d'un accès client quelconque.

Ainsi, l'adaptation aux outils et fonctionnalités spécifiques consiste tout simplement à jouer sur les paramètres publics des objets communs. Un ajustement pour certaines variables, une remise à zéro pour d'autres, un nouveau calibrage d'instruments de mesure et une activation d'options graphiques plus poussées, sont quelques échantillons des tâches sollicitées au niveau de cette première couche de la plate-forme de télécommunications. Il faut noter qu'à ce stade les programmes mettant en œuvre les laboratoires virtuels proprement dits sont uniquement accessibles en tant qu'entités à part entière. Cela veut dire que les fonctions d'adaptation demeurent opaques vis-à-vis des environnements informatiques et des plugiciels utilisés par un laboratoire spécifique quelconque. Le type et les conséquences d'adaptation varient en fonction du type de simulation.

En somme, la couche des outils et fonctionnalités de base apparaît sous forme d'un rassemblement de structures muettes, dont la couche supérieure d'adaptation aux outils et fonctionnalités spécifiques tire et remodélise des instances. La frontière entre cette couche et sa suivante est translucide, dans le sens que les deux couches sont tellement entrelacées qu'il n'est pas souvent évident d'évoquer les services de l'une sans avoir recours à ceux de l'autre. Une ambiguité assez remarquable incite à poser plusieurs questions sur la limite, la portée, les constituants et la contribution précise de chaque couche de l'architecture conceptuelle de la plate-forme de télécommunications. Un tas d'idées que seules l'implantation et la mise en œuvre du modèle suggéré sont susceptibles d'éclaircir à l'aide de solutions concrètes et des éléments de réponses réels.

CHAPITRE IV
IMPLANTATION DE LA PLATE-FORME DE TÉLÉCOMMUNICATIONS

Comme tout modèle conceptuel, l'architecture en triple couches de la plate-forme de télécommunications permet d'étendre la portée des laboratoires virtuels. Le grand défi que pose la conception de cette plate-forme demeure le choix, parmi une pluralité de solutions, de celle qui non seulement satisfait à court terme quelques aspects de base de la plate-forme, mais surtout celle qui sera apte à s'adapter à toute évolution technologique future. Dans cette perspective, ce chapitre peut être vu comme une instance de son précédent restructurée et décrite par les termes de l'environnement informatique sélectionné, à savoir Java/CORBA. Les sections qui suivent présentent une définition plus raffinée des composants de la plate-forme conceptuelle. Par la suite, les alternatives et les approches concernant l'implantation de celle-ci sont exposées, avec leurs restrictions et leurs avantages, pour justifier finalement l'adoption d'une option adéquate d'implantation.

4.1 Définition de la plate-forme de télécommunications

La mise en œuvre des principes de base élaborés dans le modèle conceptuel du chapitre trois peut être traduite différemment selon l'approche informatique adoptée. L'aperçu que présente le chapitre deux sur les technologies de communications répandues prévoit de multiples projections pour l'architecture en triples couches de la plate-forme. Or, dans un contexte comme celui des laboratoires virtuels, la reproduction d'une expérimentation réelle fait appel à des étudiants, des professeurs, des instruments, des appareils de mesure, ainsi que d'autres accessoires de laboratoires. Ce sont là autant

d'éléments distribués liés entre eux par des types et à des degrés assez variés d'interactions. Dans ce sens, la modélisation orientée objet utilisée lors de la conception se recommande pour l'implantation de la plate-forme. Cependant, le caractère évolutif que doit préserver celle-ci n'exclut pas une compatibilité avec des technologies non orientées objets. Le résultat est alors un compromis consistant à établir une structure hybride d'objets distribués apte à intégrer des outils possédant de l'immunité, non seulement face à l'hétérogénéité de langages informatiques, mais aussi face à la diversité de techniques et de mécanismes de communication.

4.1.1 Réalisation de l'adaptation aux réseaux de communications

Dans une perspective de système distribué, plusieurs serveurs fonctionnent en harmonie afin de fournir un accès aux laboratoires virtuels. Le rôle de cette couche du modèle conceptuel est accompli alors à l'aide des fonctions d'une série d'interfaces et d'intergiciels standards. Il s'agit en premier lieu de serveurs HTTP dont la portée universelle a déjà réussi l'épreuve d'interopérabilité. Quant aux différents serveurs d'outils génériques et de laboratoires spécifiques, certains parmi eux possèdent intrinsèquement les moyens nécessaires pour un fonctionnement satisfaisant en milieux hétérogènes. D'autres, par contre, posent quelques restrictions.

À travers l'ORB, CORBA permet en grande partie de combler les besoins en question. L'interopérabilité désirée se trouve traitée dans tous ses aspects par des extensions concrètes du protocole GIOP, le cas échéant IIOP. La compatibilité de GIOP avec diverses architectures de communications orientées connexion fait en sorte que les outils de base invoquant CORBA soient à la portée de plusieurs réseaux d'utilisateurs. Le protocole ESIOP introduit de sa part une extension pour les protocoles non orientés connexion. Les ORB construits aux détriments de piles de protocoles distinctes interopèrent entre eux à l'aide de passerelles.

L'occurrence simultanée de plusieurs requêtes de clients risque de ralentir le rythme auquel la plate-forme doit desservir ses utilisateurs. Pour tout ce qui est en rapport avec HTTP, le port TCP évoqué par défaut est le port 80. Cependant, pour des applications à grande échelle, d'autres numéros de ports peuvent être alloués (Berners-Lee et al., 1996), assurant ainsi la continuité de service des serveurs HTTP et, par conséquent, des outils génériques et spécifiques qui en dépendent. Si ceux-ci font partie d'autres systèmes ayant des mécanismes indépendants d'invocation d'objets distants, tels que CORBA, DCOM et RMI, des politiques propres de parallélisme sont adoptées.

Un autre élément à considérer dans un environnement de systèmes hétérogènes est la qualité de service. Celle-ci est étroitement liée aux types des réseaux de communications de clients ainsi qu'aux outils informatiques mis en jeu pour le développement des laboratoires virtuels. La couche d'adaptation aux réseaux met à la disposition des utilisateurs, autant que possible, des versions multiples d'outils spécifiques et génériques capables de fournir des fonctionnalités identiques mais dans des contextes différents. À titre d'illustration, un outil de base accessible via un ORB IIOP est parfaitement exploité dans un environnement TCP/IP. Dans un contexte ATM, par exemple, une autre version de ce même objet est disponible à travers un ORB implantant une extension GIOP/ATM. Ceci est tout à fait réalisable puisque l'apparition perpétuelle d'outils informatiques de plus en plus sophistiqués permet d'implanter sur une même plate-forme plusieurs ORB assurant la transmission de l'information à travers plusieurs familles de protocoles. Même plus, une même architecture, TCP/IP par exemple, est aussi susceptible d'être servie par une collection d'ORB provenant de fabricants différents et se distinguant l'un de l'autre par des modes opératoires distincts.

Par ailleurs, la multiplication d'outils et de systèmes peut constituer une menace pour la sécurité des échanges effectués. La plate-forme de télécommunications est alors appelée à assurer la protection des ressources de laboratoires virtuels situées au-delà de la couche d'adaptation aux réseaux. Effectivement, ceci est réalisable en deux temps. Au niveau de

l'accès aux laboratoires spécifiques, l'interface utilisateur autorise uniquement les usagers qui sont reconnus par le système, c'est-à-dire authentifiés par des mots de passe et d'autres renseignements privés. D'autre part, la sécurité constitue un enjeu de plus en plus critique pour les protocoles et les moyens de communications standards. Dans ce sens, des recherches ont été menées dans le but de concevoir un protocole conjoint à HTTP pour sécuriser les échanges entre clients et serveurs HTTP (Rescorla et Schiffman, 1996). Le groupe OMG pour sa part fait apparaître aussi la sécurité sous forme d'une partie intégrante de son modèle proposé.

Les ressources ont besoin d'être contrôlées ou gérées par une certaine entité. En termes d'administration de réseaux, plusieurs applications sont répandues en réseau local aussi bien que sur le Web. De son côté, CORBA permet d'enrichir le modèle classique gestionnaire/agent afin de développer des relations plus complexes et conformes aux services de gestion (Haggerty et Seetharaman, 1998). Dans le cadre des laboratoires virtuels, la couche d'adaptation aux réseaux tient à intégrer les outils des divers laboratoires dans le spectre des objets administrés par le réseau local dont la plate-forme fait partie. Il est alors possible de configurer les outils de base, et éventuellement les outils spécifiques, comme éléments d'un modèle générique global d'administration, parrainé par CORBA par exemple. Il en découle plusieurs avantages tels que : une mise en échelle plus aisée et plus contrôlable, un support ouvert d'interfaces standards et de produits d'administration commercialisés, un découplage entre l'interface utilisateur et le code réel des objets, ainsi qu'une indépendance vis-à-vis des implémentations informatiques d'objets.

En somme, la mise en place d'un nombre suffisant d'outils génériques d'une part, et le développement de laboratoires spécifiques moyennant des catégories variées de logiciels d'autre part, contribuent à élargir le profil de la couche d'adaptation aux réseaux. Celle-ci évolue perpétuellement en assumant des fonctions de plus en plus variées et sophistiquées.

4.1.2 Mise en place des outils et fonctionnalités de base

De par leur présentation initiale dans le modèle conceptuel, les outils et fonctionnalités de base trouvent dans la perspective orientée objets une incarnation adéquate. À première vue, une fonctionnalité quelconque correspond pratiquement à un groupe d'objets, distribués ou non, fournissant des opérations convergeant autour d'un thème spécifique. Cela peut être la prise de mesure ou le traitement de texte. L'implémentation d'un objet peut être accomplie à l'aide de divers langages de programmation structurée ou orientée objet.

Dans un environnement CORBA, les objets sont désormais récupérables chacun par une référence (IOR) unique, transitoire ou permanente, dont la génération et la gestion sont assurées par des composants du modèle de référence de CORBA. Un mécanisme de référence assez semblable est utilisé pour l'interface RMI qui exige des références d'objets pour accéder à ceux-ci à travers une architecture RMI particulière au-dessus de TCP/IP. L'intégration d'outils génériques conçus et mis en œuvre avec des langages variés demeure complètement transparente. Une telle approche multidimensionnelle conduit à une implantation hybride de la couche conceptuelle d'outils et fonctionnalités de base. Ainsi, on peut trouver côte à côte : des objets distribués écrits en C ou C++ évoqués au travers d'un ORB, d'autres objets écrits en Java et accessibles via l'interface HTTP classique ou bien via RMI, ainsi que des objets dont le code est en Perl et activé à travers l'interface HTTP/CGI.

Cette diversité de choix améliore non seulement la flexibilité du modèle générique mais surtout en augmente le degré de distribution. À titre illustratif, une requête de mise à jour du cahier de laboratoire peut être redirigée, par un serveur frontal, vers un autre système où se trouve une base de données représentant physiquement ce cahier de laboratoire. Dans le cas d'un objet de référence valide dans un environnement CORBA, l'ORB entreprend un tel processus d'acheminement.

Il faut faire attention, cependant, au maintien de l'intégrité référencielle et l'optimisation de l'espace mémoire global du système distribué afin de limiter le gaspillage de ressources causé par des objets non pertinents, deux enjeux pour lesquels les architectes de CORBA, entre autres, n'ont pas encore trouvé une solution convaincante (Henning, 1998).

En résumé, la couche d'outils et de fonctionnalités de base de la plate-forme de télécommunications est mise en forme par l'association d'objets distribués éventuellement dans l'espace et faisant appel à des langages de programmation multiples. À ce niveau, l'interopérabilité est due en grande partie aux langages de programmation mobiles, surtout Java, ainsi qu'aux plugiciels et aux interfaces exportant les applications de laboratoires virtuels au sein de fureteurs Web.

4.1.3 Réalisation de l'adaptation aux outils et fonctionnalités spécifiques

De par leur caractère générique, les outils de base doivent être fonctionnels au sein de deux ou plusieurs laboratoires spécifiques. Il s'avère alors nécessaire de doter chaque outil de base de paramètres à accès publique, peu importe le langage d'implémentation utilisé.

Ce genre d'adaptation prend place à un niveau supérieur à celui d'un intergiciel quelconque et dépend étroitement des outils de programmation d'applications qui le complémentent. Typiquement, dans un environnement CORBA, DCOM ou RMI, une requête a pour objectif de lancer une opération sur un objet distribué et d'en recueillir les résultats. Il en est de même pour d'autres intergiciels tels que HTTP/CGI ou RPC, qui permettent simplement l'exécution d'un programme ou bien d'un bloc d'instructions sur un site distant et d'en récupérer la sortie. En tout cas, ce genre d'adaptation n'est pas en mesure de changer l'apparence graphique ou bien la façon de manipuler un objet de base afin qu'il soit utile dans le contexte particulier que pose chaque laboratoire spécifique.

Logiquement parlant, la mise à niveau des outils de base doit avoir lieu une fois que le client évoque le laboratoire en question. Ainsi, ce dernier ne verra en face de lui qu'un ensemble d'outils prêt à tout usage, sans la perception des différences d'implantation de deux outils quelconques. Le réglage des outils génériques est alors confié à ce qui peut porter le nom d'interface d'appel dont dispose la plate-forme et qui constitue la seule clef d'accès à un laboratoire virtuel donné. Il faut noter que chaque laboratoire spécifique est muni d'une interface d'appel propre ayant une recette d'adaptation d'outils génériques dictée par la discipline scientifique particulière de celui-ci. Il revient donc à chaque interface d'appel d'activer son laboratoire spécifique, d'adapter les outils génériques au contexte courant et de fournir à l'utilisateur une interface graphique souple. À ce propos, l'expérience du développement de réseaux, essentiellement Internet, crédite le langage Java qui a tant répondu à des exigences de cahiers de charge similaires.

Idéalement, des contacts entre outils génériques et spécifiques doivent être tolérés afin de consolider l'homogénéité de manipulation. Ceci impose à la structure interne, tant au niveau de la modélisation que de l'implémentation, de chaque laboratoire spécifique des conditions de compatibilité et d'intégration avec l'environnement interopérable du modèle générique qu'offre la plate-forme. Évidemment, l'adaptation des outils et fonctionnalités de base sera de plus en plus efficace, en autant que lesdites conditions précédentes sont respectées.

4.2 Implantation de la plate-forme de télécommunications

À ce stade, une majorité des fonctionnalités proposées dans le modèle conceptuel trouvent des moyens concrets pour leur mise en œuvre. La diversité des outils évoqués vise à créer une plate-forme de télécommunications dont le mode de service est aussi ouvert que possible. Cependant, la procédure suivie pour atteindre cet objectif se heurte à plusieurs obstacles. En effet, certains découlent de la structure informatique propre des

laboratoires spécifiques, d'autres sont dus aux ressources de transmission et de traitement de l'information, d'autres encore sont d'origine conceptuelle provenant de l'environnement hétérogène dans lequel se trouve la plate-forme. Ainsi, on peut énumérer les limitations suivantes :

- Des langages de programmation particuliers : Afin de mieux représenter les éléments de la discipline scientifique à laquelle il est dédié, un laboratoire spécifique utilise souvent des langages informatiques spéciaux. En d'autres termes, ces derniers conviennent parfaitement à leur milieu pédagogique, mais ils manquent parfois de mécanismes télé-informatiques pour un accès à distance et présentent souvent un trait propriétaire vis-à-vis du reste de l'environnement des laboratoires virtuels. Ceci affecte l'interopérabilité, l'intégration ainsi que l'interaction avec d'autres composants, notamment des outils génériques écrits, par exemple en Java ou bien accessibles via CORBA. À titre informationnel, c'est le cas du langage *Lingo* qu'intègre l'environnement *Director* utilisé dans le laboratoire de physique. Ce langage s'avère complètement loin de la modélisation d'objets distribués adoptée pour des outils de base.
- Une surcharge d'exécution distante : Normalement, un laboratoire spécifique doit être à la portée de plusieurs clients en même temps. La nature individuelle d'une séance de laboratoire, d'une part, et l'interaction souhaitée lors d'une expérimentation virtuelle d'autre part, rendent difficile la tâche d'exécutions simultanées distantes sur le serveur. Par suite, il vaut mieux prévoir l'exécution locale, en grande partie sur le site client, des simulations de laboratoires accompagnées toutefois de requêtes bien définies pour l'exploitation de ressources partageables stockées sur le serveur. Ceci allège l'application de serveur mais introduit d'autres problèmes, en termes de délai et d'espace de stockage, lors du téléchargement des programmes exécutables et de leur démarrage.
- Une restriction sur la portabilité : Ceci est en fait une conséquence directe de l'exécution locale, puisque les entités exécutables générées par certains laboratoires ne peuvent fonctionner parfois que dans des conditions très particulières. Celles-ci

peuvent dépendre du système d'exploitation du site client. Par exemple, des fichiers avec une extension (.exe) ne sont pas compréhensible sur une station de système d'exploitation UNIX. D'ailleurs, il ne faut pas oublier le temps de téléchargement qui risque d'être long à cause de la taille souvent très grande des fichiers multimédias transmis.

- Des contraintes de temps réel : Dans un contexte aussi délicat que celui des laboratoires virtuels, des manipulations peuvent mettre en jeu des ressources communes évoquées en temps réel. Le cas typique est celui d'une carte d'acquisition de signaux électriques du laboratoire de génie électrique construit à l'aide de la version 5.0 de *Labview*™. Celle-ci est branchée au serveur mais elle est utilisée à tour de rôle par les clients. L'acheminement de ce genre de données numériques est alors la cible d'éventuelles défaillances au niveau des liens et des intergiciels de communication, y compris la version courante de CORBA.

Compte tenu de ce qui précède, l'accès d'un client à un laboratoire virtuel se trouve dissocié en deux étapes : l'activation explicite des modules propres au laboratoire spécifique et celle des outils de base. La première tâche dépend significativement du cadre informatique original dans lequel fut générée la simulation virtuelle du laboratoire en question. Quant à l'utilisation des outils génériques, elle se fait d'une manière homogène, quel que soit le contexte scientifique exploité.

Bien qu'une telle séparation ne mette pas en cause le caractère générique de la plate-forme, elle affecte cependant la transparence de l'intégration des outils partageables dans chaque laboratoire spécifique. En effet, les outils et fonctionnalités spécifiques possèdent des capacités rudimentaires en ce qui concerne les communications dynamiques et l'établissement de flots d'information d'entrée/sortie avec leurs homologues génériques. Il en résulte certaines limitations de manipulation au niveau de l'interface utilisateur, ainsi que quelques interventions manuelles et configurations préalables au niveau du site client. Si ceci s'avère acceptable dans cette première phase

de développement des laboratoires virtuels, il ne constitue pas une solution conforme au modèle conceptuel visé à plus long terme.

Il vaut mieux présentement opter pour une solution de compromis. Cela implique une implantation intérimaire de la plate-forme de télécommunications qui fournit aux utilisateurs un accès de "meilleur effort", tout en maintenant le caractère évolutif. À court terme, on peut tolérer l'existence de laboratoires spécifiques agissant à titre d'entités autonomes et dépourvues de mécanismes intrinsèques pour l'incorporation du matériel générique. Des outils de base vont être mis en supplément, moyennant des approches informatiques dotées d'un niveau élevé de degré de liberté. En d'autres termes, l'aspect générique sera plus ou moins préservé avec les prototypes de laboratoires spécifiques existants, mais de plus en plus amélioré au fur et à mesure que de nouveaux laboratoires sont mis en œuvre. Ces derniers seront sujets à respecter les grandes lignes, voire la perspective de développement ouvert de la plate-forme, dans le but d'une coexistence efficace avec les outils génériques déjà opératoires et ceux à venir. À titre illustratif, un laboratoire spécifique peut être conçu lui-même comme une collection d'objets distribués. L'interaction de ceux-ci entre eux et avec les outils de base externes suit des règles et des protocoles bien définis conformément à la discipline scientifique de ce laboratoire. À l'heure actuelle, certaines alternatives sont disponibles pour l'implantation d'une plate-forme apte à supporter, au moins temporairement, les deux prototypes des laboratoires de physique et de génie électrique.

Le laboratoire de physique : Il s'agit d'une reproduction d'un laboratoire réel particulièrement riche en instruments de mesure et en matériel expérimental, ce qui justifie la présence quasi excessive de l'effet visuel et de l'animation dans le laboratoire virtuel. Le produit *Director* de *Macromedia* fournit un environnement informatique pertinent pour implémenter le laboratoire de physique. Le langage de programmation utilisé, *Lingo*, n'est autre qu'un langage orienté objet qui permet une puissante interactivité reposant sur des combinaisons de graphisme, de texte, de son et de vidéo.

Dans le cas d'une exécution locale, le laboratoire virtuel n'est autre qu'un fichier exécutable avec l'extension (.exe). Par ailleurs, la distribution sur un réseau de communication peut se faire de deux façons. La première, qualifiée de primitive, consiste tout simplement à télécharger le code exécutable, pour le lancer ensuite sur le site client, éventuellement après avoir effectué quelques configurations d'ordre sécuritaire. La deuxième façon est beaucoup plus sophistiquée et fait parvenir un autre produit de *Macromedia* appelé *Schockwave*. En fait, *Schockwave* n'est autre qu'un module intermédiaire qui autorise, à quelques fonctionnalités près, un fureteur à déclencher l'exécution d'un programme généré sous *Director* sous forme de pages Web. Plus précisément, c'est un engin d'Internet qui est dédié à la répartition d'applications multimédias sur une faible bande passante. Évidemment, le succès de *Schockwave* dans le cadre de *Director* est dû au fait que ce dernier supporte un ensemble de standards Internet. Ceci comprend un support intégré de protocoles, tels que HTTP et FTP, de même que d'autres fonctionnalités, telles que l'interprétation de documents XML et l'intégration de fichiers HTML. En outre, des échanges de données en double sens sont aussi possibles grâce au support de l'interface CGI. Une fonction importante réside dans la possibilité de convertir automatiquement le code *Lingo* en une Applet Java facilement compréhensible par une machine virtuelle Java quelconque. Ceci constitue donc un mécanisme de distribution sur Internet équivalent à *Schockwave*.

Cependant, les deux méthodes suggérées possèdent des inconvénients. Pour les exécutables, le téléchargement risque d'être très lent. De plus, l'utilisateur est amené à établir des permissions spéciales pour que son système local de fichiers soit accessible explicitement. Cela réduit à la fois la portabilité et la souplesse de l'exécution distante, surtout que les plates-formes munies de systèmes d'exploitation UNIX ou LINUX sont exclues. Ce dernier obstacle existe aussi avec *Schockwave* qui est conçu spécifiquement pour les systèmes d'exploitation Windows 95/98/NT et Apple OS 8. La conversion en Java permet de surmonter partiellement ce problème, puisque le code Java résultant est saisissable par un interpréteur d'une station Sun, par exemple. Mais, cette alternative

souffre, elle aussi, d'une lenteur d'exécution considérable, ainsi que d'une certaine incompatibilité avec quelques instructions *Lingo*. Un autre moyen fut envisagé pour établir un lien entre les outils spécifiques du laboratoire de physique et des outils de base écrits en Java. Ceci est faisable en passant par l'intermédiaire de code Javascript pour une remise en forme des flots de données échangés avec *Schockwave*. Toutefois, la réalisation de ce choix s'est avéré complexe et non fiable dans un sens ou dans l'autre.

Somme toute, peu importe la solution choisie afin de manipuler à distance le laboratoire virtuel de physique, les performances obtenues restent assez modestes. Une remodélisation de ce laboratoire remédierait peut-être au rendement minimal actuel, mais risque bien sûr de sacrifier le caractère interactif spécial qu'introduit *Director* si jamais ce dernier est écarté.

Le laboratoire de génie électrique : Orienté vers une discipline encore plus pointue, ce laboratoire met en relief des aspects classiques du génie électrique, notamment pour ce qui est du traitement de signaux. La mise en forme de maquettes électriques virtuelles peut être facilement effectuée à l'aide du système *Labview* de programmation graphique parallèle et multitâche de *National Instrument*. En effet, il s'agit de regrouper en ce qu'on appelle instruments virtuels ou VI, des composants électroniques variés tels que des palettes de contrôle, des gauges, des thermomètres, des lampes, des interrupteurs, etc. L'interface, ainsi construite à l'aide du langage graphique G, est compilée pour fournir un code exécutable à une vitesse comparable à celle d'un code C compilé. Les instruments électriques, réels ou virtuels, impliqués dans une application sont gérés à l'aide de contrôleurs standards manipulés eux aussi à travers des librairies et des interfaces standards de *Labview*, genre VISA et GPIB.

Dans ce contexte, des applications à part entière sont obtenues en mode local mais aussi en mode distant. La connectivité aux instruments éloignés est envisageable de différentes manières. Pour le laboratoire de génie électrique, trois alternatives sont

prédominantes. La première correspond directement à l'acquisition de données et à l'accès physique aux cartes et à l'équipement électronique distribué. Le résultat est un fichier exécutable qui permet à un client de capturer et d'analyser les signaux reçus ou transmis au travers d'une interface GPIB implantant la norme IEEE 488. GPIB supporte des mécanismes divers de communications, entre autres les architectures TCP/IP et ATM. Cette approche ne pose pas de limite sur le nombre d'utilisateurs simultanément connectés, mais manque de performance à cause de téléchargement des exécutables.

La deuxième alternative consiste en un modèle client/serveur dédié à l'acquisition de données en entrée/sortie au niveau d'instruments éparpillés sur un réseau local ou étendu. Ceci nécessite l'installation de trousses du logiciel NI-DAQ tant du côté du client que du serveur. La procédure d'accès à distance utilisée porte le nom de RDA et se sert de la famille de protocoles TCP/IP. Par ailleurs, un scénario d'acquisition de données distantes est complètement transparent avec RDA et est accompli suite à une configuration adéquate des entités clients pour qu'elles puissent au moins reconnaître le serveur évoqué. Ainsi, un client sera apte à recevoir des signaux issus d'un ou de plusieurs serveurs en même temps. Un obstacle important réside cependant du côté serveur puisque sa capacité de servir plusieurs clients concurrents est réduite et risque d'atteindre seulement un seul client à un moment donné.

Finalement, *Labview* comporte un outil de développement pour Internet. Connu sous le nom de *Internet Development Toolkit for G*, cet outil permet de construire des serveurs HTTP capables de convertir les instruments virtuels en images incorporables dans des documents HTML et perceptibles par l'intermédiaire de fureteurs Web. D'autre part, des mécanismes spéciaux peuvent être utilisés pour veiller sur la sécurité des pages HTML en question. En outre, cet outil de programmation offre la possibilité de générer des instruments virtuels encapsulant du code CGI pour des échanges plus dynamiques de requêtes entre clients et serveurs. Les instruments virtuels sont aussi dotés d'autres

fonctionnalités plus classiques telles que l'émission de courrier électronique et la transmission de fichiers à un serveur FTP.

Bien que cette dernière approche semble être la plus simple à implémenter, son adoption dépend en fait de l'aspect expérimental particulier sur lequel les simulations présentement mises en œuvre tentent de mettre l'accent. Un travail substantiel demeure indispensable afin d'adapter le laboratoire de génie électrique au cadre ouvert de fonctionnement que la plate-forme de télécommunications essaie d'élaborer. Il s'ensuit aussi que cette dernière est assujettie à prévoir des moyens d'interopération avec les environnements assez spécifiques et complexes des laboratoires virtuels spécifiques.

4.3 Solution proposée

L'adoption d'une solution définitive dans le contexte de la conception d'une plate-forme de télécommunications garantissant une certaine interopérabilité doit prendre en compte un nombre de critères dont certains furent exposés au cours des sections précédentes. La solution que nous présentons ici ne prétend pas satisfaire l'intégralité de ces critères. Elle est plutôt une solution temporaire qui constitue en soi une preuve de faisabilité d'un concept que les travaux de recherche futurs viendront renforcer et étendre.

4.3.1 Implantation de l'architecture en triple couches

L'accès aux laboratoires virtuels est en fait un accès à un site Web logé sur un serveur HTTP. Des pages d'accueil offrent à l'utilisateur un aperçu général des objectifs pédagogiques des laboratoires. La navigation se fait à l'aide de liens HTML dont certains servent aussi de commandes de démarrage pour les expérimentations proprement dites. Les pages en question auront pour but de lancer côte à côte le laboratoire voulu sur la plate-forme du client, ainsi qu'une console autonome d'outils de base.

En ce qui concerne la première tâche, l'activation des codes est achevée de façon différente selon le prototype de laboratoire désigné. Pour le laboratoire de physique, les tendances actuelles semblent favoriser le recours à l'outil *Shockwave* pour exporter le code *Lingo* en format HTML. En revanche, la méthode d'exploitation du laboratoire de génie électrique passera plus probablement par la procédure d'accès distant RDA.

Quant à l'ensemble des outils génériques, il est conçu de manière à garder une certaine compatibilité, quelle que soit l'approche finale adoptée. En effet, la console des outils de base est une Applet Java qui implémente pour le moment trois outils, à savoir : un chronomètre, un cahier de laboratoire et une calculatrice. L'adaptation de ceux-ci au contexte spécifique d'un laboratoire donné se fait lors du téléchargement de l'Applet, et plus précisément en se basant sur des classes auxiliaires associées à chaque outil. En d'autres termes, tout laboratoire spécifique possède une interface d'appel propre qui comprend ce genre d'Applet Java. Ceci incarne tout simplement l'implantation de la couche des outils et fonctionnalités de base du modèle conceptuel, ainsi que celle de la couche d'adaptation aux outils et fonctionnalités spécifiques, comme l'illustre la Figure 4.1.

En ce qui concerne la couche conceptuelle d'adaptation aux réseaux des utilisateurs, elle est implantée sous divers aspects. D'abord, l'Applet Java évoquée fait intervenir l'interface HTTP du Web. Ceci sous-entend l'implication de la famille TCP/IP via Internet d'une part, et reporte l'hétérogénéité des architectures de communications vers un niveau plus bas selon l'échelle du modèle OSI, d'autre part. La contribution des protocoles ATM, par exemple, sera donc au niveau de la liaison de données, ce qui fait que les données échangées sont pratiquement des datagrammes IP mais encapsulés à l'intérieur de cellules ATM. Cette technique porte le nom de IPOA. Il en est de même pour les réseaux d'accès ADSL, qui sont par nature des liens de transmission de bas niveau, avec la seule différence que les paquets IP sont véhiculés sur des supports physiques distincts et à des vitesses différentes.

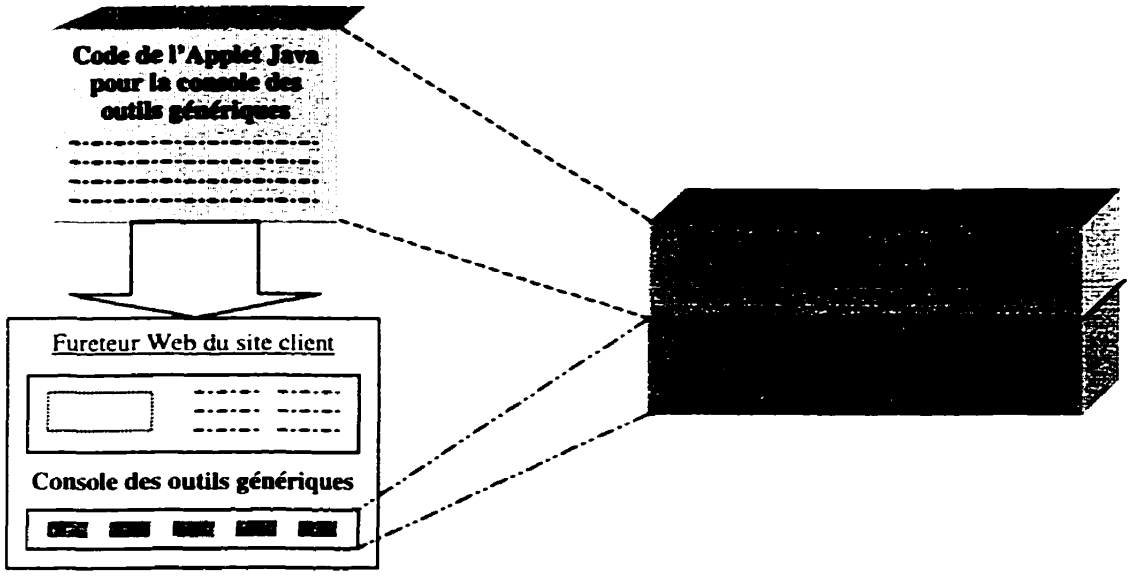


Figure 4.1 Implantation des deux premières couches du modèle conceptuel

De la même manière et à côté de l'interface HTTP, un outil générique qui explicite l'échange de requêtes selon l'architecture CORBA n'a pas à se soucier de l'hétérogénéité des réseaux de communication utilisés. L'infrastructure sous-jacente, reposant le plus souvent sur le protocole IIOP, permet à l'ORB de bien mener les opérations distantes et d'en communiquer les résultats aux clients.

Désormais, l'architecture CORBA aussi bien que les autres mécanismes et interfaces, genre CGI ou RMI, constituent une composante fondamentale de l'environnement des laboratoires virtuels se voulant générique. Dans tous les cas, l'interopérabilité est assurée de façon transparente à l'utilisateur et également au concepteur, puisqu'elle découle des produits standards mis en jeu. La Figure 4.2 renseigne plus spécifiquement sur la position de la couche d'adaptation aux réseaux des utilisateurs.

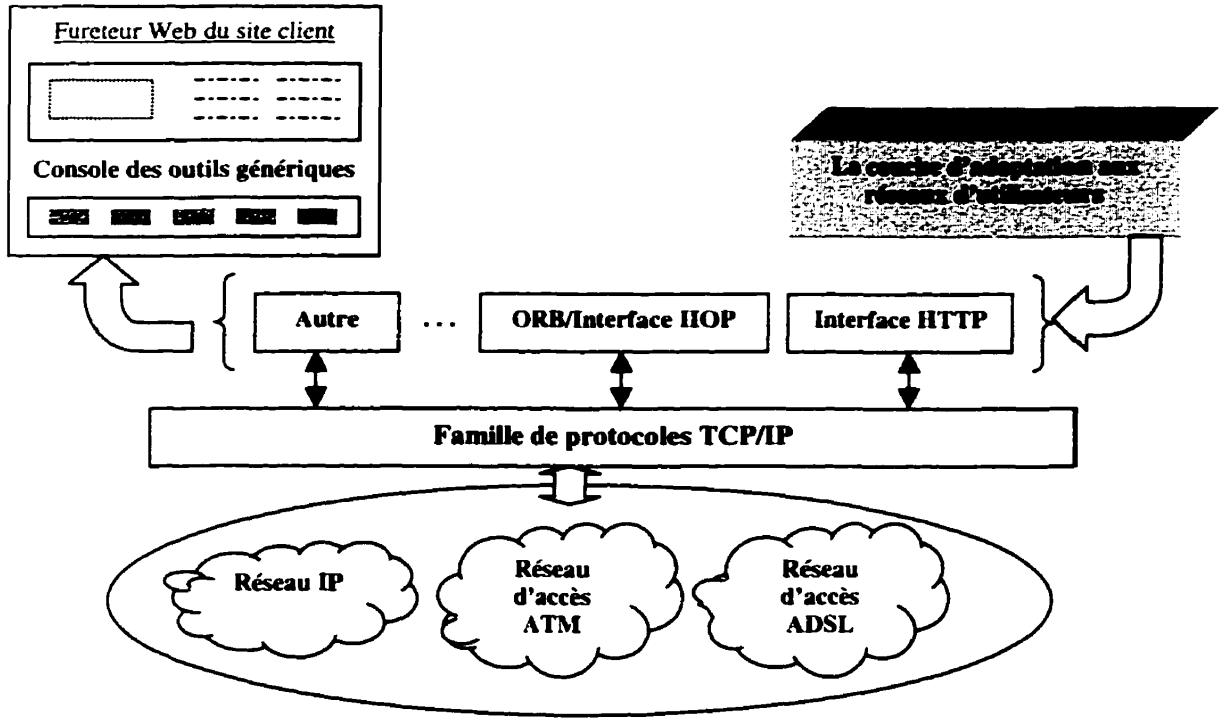


Figure 4.2 Implantation de la couche d'adaptation aux réseaux des utilisateurs

4.3.2 Mise en œuvre des outils de base

Parmi les outils de base implantés, chacun peut invoquer un outil informatique différent sans affecter ceux de ses homologues. Dans cette perspective, l'Applet Java consiste en un processus parent unique qui crée un processus enfant propre à chaque outil de base évoqué. Tant que l'outil en question est actif, le processus en charge demeure vivant en arrière plan. Puisque les outils implantés sont dédiés chacun à un service distinct des autres, les ressources allouées à un outil diffèrent de celles nécessaires à n'importe quel autre. Ceci permet d'alléger la synchronisation entre les processus enfants concurrents, et préserve toujours ce qu'on appelle l'exclusion mutuelle dans l'environnement d'exécution de sites clients. La structure logique de l'Applet Java est représentée à la Figure 4.3.

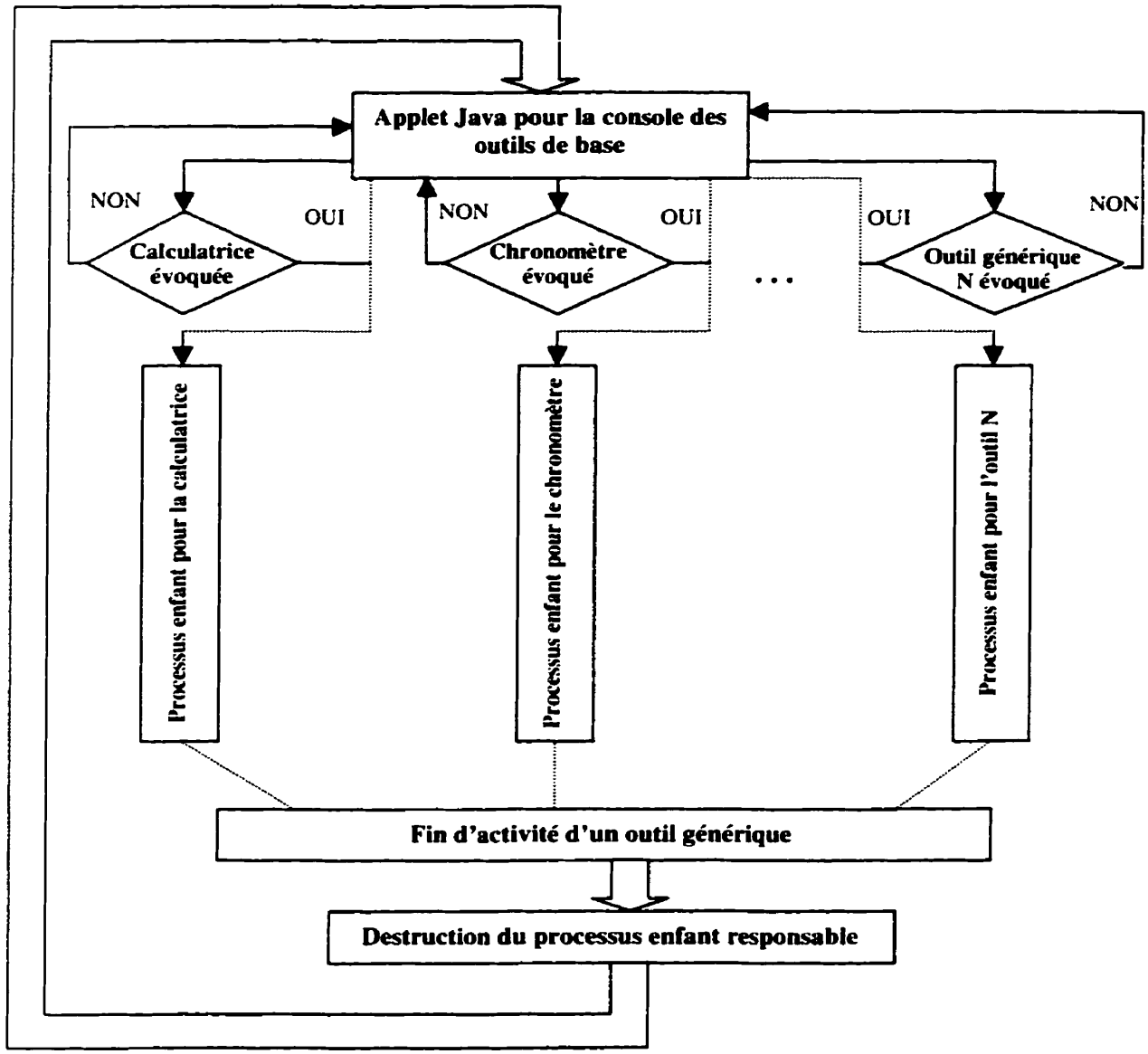


Figure 4.3 Organisation logique des outils de base

À titre illustratif, le chronomètre n'est autre qu'un ensemble de classes en Java. Leur fonction collective revient à enregistrer un délai temporel moyennant l'horloge interne du site client sans aucune intervention de la part du serveur. Le scénario est identique

pour la calculatrice qui fait recours à des fonctions graphiques plus poussées du langage Java.

Ce qui est le plus intéressant est le cahier de laboratoire. En effet, celui-ci est une illustration parfaite du caractère ouvert de la plate-forme de télécommunications. Il s'agit d'une représentation tabulaire faisant partie d'une base de données consacrée au système de laboratoire virtuel. Comme tout système de base de données, des structures relationnelles permettent de regrouper les informations des laboratoires eux-mêmes, des étudiants éligibles à y accéder, ainsi que celles d'autres unités administratives en charge. Sans trop rentrer dans les détails, le cahier de laboratoire est donc une table à laquelle les clients accèdent de façon dynamique au cours d'une séance d'expérimentation.

Toutefois, un client doit passer par l'intermédiaire d'un serveur de base de données. Les échanges entre ce serveur et les clients ne sont autres que des requêtes CORBA acheminées et construites avec l'assistance d'un ORB supportant le langage Java. Le contenu de celles-ci est, par ailleurs, des expressions SQL interprétées puis exécutées par le serveur à l'aide de l'interface JDBC et des librairies SQL que fournit Java. La saisie des données ou leur visualisation du côté du client sont bien facilitées par l'interface interactive que présente Java. Quant à l'administration de la base de données et la gestion des requêtes de clients, elles sont confiées au serveur lui-même, loin des interventions de clients. Ceci protège les données sauvegardées et les rend totalement accessibles à l'entité académique concernée. Un scénario comme celui-ci porte le nom de modèle client/serveur trois tiers. Un accès typique au cahier de laboratoire est illustré à la Figure 4.4.

En conclusion, cette solution constitue une étape vers une solution de long terme. On aspire plus tard à un degré supérieur d'interactivité de sorte que les liens entre outils génériques et spécifiques soient de plus en plus robustes. Une telle perception exige une profonde revue des prototypes mis en œuvre. La plate-forme de télécommunications sera

ainsi en mesure de s'adapter aux futurs changements, tout en conservant les principes de base sur lesquels la modélisation actuelle est établie.

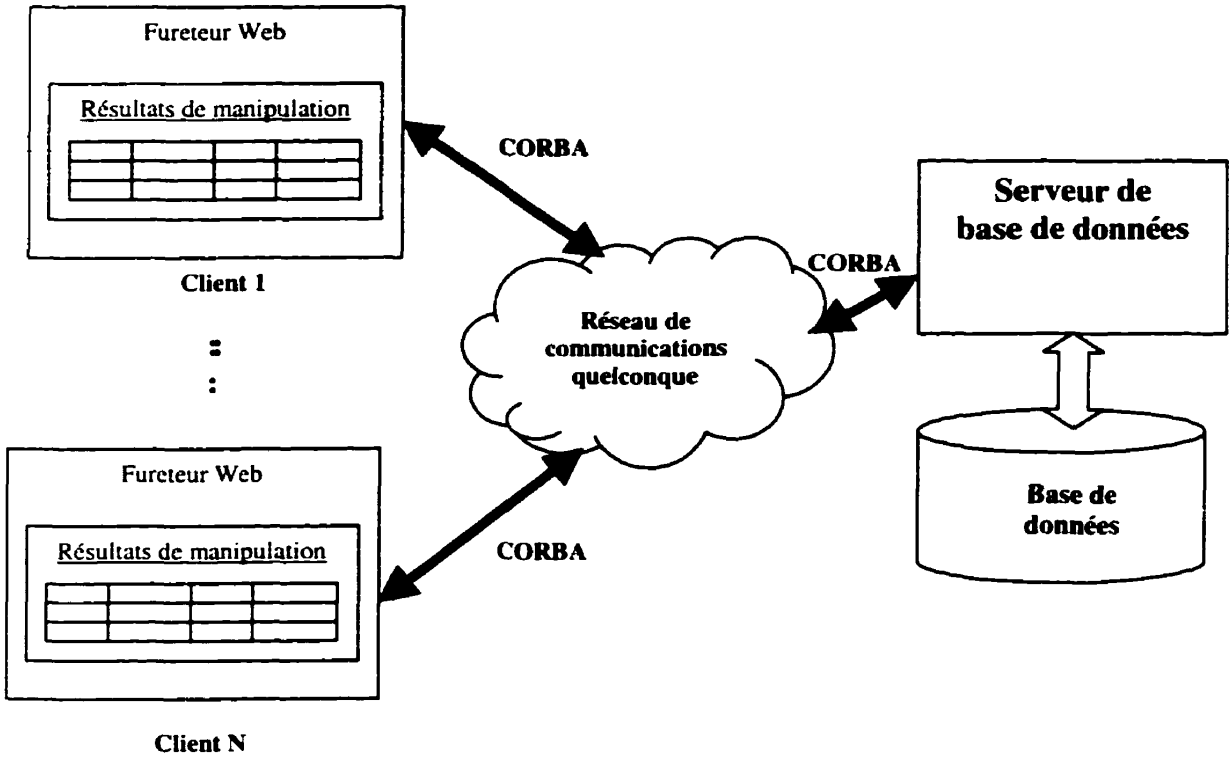


Figure 4.4 Accès au cahier de laboratoire

CHAPITRE V
MISE EN ŒUVRE ET EXPÉRIMENTATION

La performance de la plate-forme de télécommunications peut être mesurée en se basant sur des critères tels la fiabilité des liens, les caractéristiques opérationnelles de la machine du client elle-même, et la capacité des serveurs. Ce chapitre fait état de la mise en œuvre et de l'expérimentation de cette plate-forme. Il débute par une description de l'interface utilisateur permettant la manipulation des laboratoires virtuels. Par la suite, il évalue l'infrastructure hétérogène de communications déployée entre plusieurs postes et dédiée à des fins de tests. Finalement, des résultats d'expérimentation seront rapportés et analysés.

5.1 Mise en œuvre de la plate-forme de télécommunications

En dépit de son rôle implicite dans le fonctionnement de chaque laboratoire, la plate-forme de télécommunications ne jouit pas d'une interface autonome. Sa contribution apparente se résume, sur l'écran d'un client, à une série de boutons, relatif chacun à un outil de base donné. La part la plus importante du rôle de cette plate-forme, en matière de télécommunications, demeure toutefois invisible à l'utilisateur. Par ailleurs, certaines exigences matérielles et logicielles doivent être remplies en vue d'un fonctionnement global cohérent.

5.1.1 Préalables informatiques à la mise en œuvre

La console d'outils de base est en soi une Applet Java insérée dans une page HTML téléchargée à partir d'un serveur HTTP publique. Elle est capable donc de fonctionner

avec un système d'exploitation quelconque à condition, bien sûr, que ce dernier supporte un fureteur Web. Cependant, les structures actuelles des prototypes de laboratoires de physique et de génie électrique permettent de préciser certaines spécifications au niveau du système client, à savoir :

- la nature du système d'exploitation : Il s'agit pour le moment de Windows 95/98/ NT 4.0 ou plus, ainsi que de MacOS (Power PC) ;
- la nature du processeur : 486/DX (Pentium recommandé) ;
- la mémoire vive : 32 MB minimum (64 MB ou plus recommandé) ;
- l'espace disque : 125 MB minimum ;
- les fureteurs recommandés : Netscape Navigator 3.0, Internet Explorer 3.0 ou leurs versions ultérieures.

Le client doit avoir sur sa machine la version complète de l'environnement d'exécution JRE 1.2 de plate-forme Java 2, ou encore du JDK 1.2. En effet, le JDK 1.2 est un outil de développement Java qui ne fait pas partie, à l'heure actuelle, des versions courantes de fureteurs. Le cas échéant, ceux-ci sont munis de versions antérieures du JDK 1.1. Par rapport à ce dernier, des améliorations significatives sont apportées par le JDK 1.2 : une interface graphique plus riche, une politique de contrôle d'accès plus sécuritaire, des mécanismes RMI plus avancés, un support CORBA pour l'interopérabilité et la connectivité de services à distance, etc. À titre d'exemple, les permissions spéciales du JDK 1.2 permettent à une Applet Java de déclencher l'exécution d'un fichier (.exe) remis par le serveur à un client. Dans ce cas, une configuration préalable est nécessaire afin que le client autorise tout code provenant d'une source étrangère à accéder à son système de fichier local.

Par ailleurs, afin de pouvoir démarrer les Applets écrites en JDK 1.2 au sein des différents fureteurs existants, un plugiciel Java est aussi recommandé. Il permet de s'abstenir de l'environnement d'exécution du fureteur tout en fournissant au code Java une machine virtuelle Java propre (JRE) capable d'interpréter les instructions générées

auparavant par un compilateur Java JDK 1.2. Le JRE 1.2 étant installé sur le site client, il suffit donc d'évoquer le module d'extension Java, au moment du téléchargement de l'Applet via l'interface standard HTTP, pour que l'environnement JRE soit activé. La version courante des fureteurs, Netscape Navigator et Internet Explorer, sera ainsi en mesure de supporter les nouvelles versions de Java. Évidemment, une telle opération ne requiert pas l'intervention de l'usager et se fait de façon automatique à chaque fois que la page Web appropriée est acquise.

En ce qui concerne le support CORBA, un ORB Java compatible avec la spécification CORBA/IIOP 2.0 est fourni sous forme de librairies du JRE 1.2 lui-même. À travers l'Applet, certains outils génériques peuvent initialiser un ORB, pour se connecter ensuite à tout objet distribué identifiable par une référence d'objet unique (IOR). Une configuration de la politique de sécurité est indispensable pour que l'Applet chargée localement puisse établir une connexion CORBA et manipuler l'objet distant en question. Cette étape est en fait effectuée une seule fois à l'aide de l'outil *policytool* du JRE 1.2. Les accès ultérieurs pourront se dérouler sans problème. L'annexe I fournit en détail la démarche préalable à suivre.

Cependant, la liste des spécifications exigées du côté client n'est pas exhaustive et est assujettie, à tout instant, à l'addition de nouvelles extensions en fonction des produits utilisés par les prototypes de laboratoires virtuels mis en service. Dans cette catégorie, on peut trouver le plugiciel *Shockwave* pour le laboratoire de physique, ou bien d'autres extensions spéciales de *Labview* pour l'acquisition et le contrôle de données manipulées par des instruments virtuels distribués.

En ce qui concerne le serveur, l'affaire est un peu plus délicate à cause de la multiplicité des entités, génériques et spécifiques, desservant les clients. En effet, l'approche inductive adoptée pour construire le modèle générique de laboratoires virtuels fait en sorte que l'ajout d'un nouveau laboratoire introduit au moins un outil informatique

additionnel. Il en découle éventuellement l'intervention d'un ou de plusieurs serveurs selon la modélisation propre au laboratoire spécifique en question. Tel est le cas, par exemple, du laboratoire de génie électrique où le client peut contrôler à distance un moteur pas-à-pas en liaison directe avec un serveur *Labview* et un protocole bien déterminé.

À côté des serveurs relatifs aux laboratoires spécifiques, certains outils génériques, faisant partie d'un environnement d'objets distribués CORBA par exemple, requièrent eux aussi des serveurs propres. Dans ce cas, ces objets agissent eux-mêmes à titre de serveurs auxquels n'importe quel accès de clients s'effectue en respectant le modèle opératoire de l'architecture CORBA. Dans notre réalisation, le cahier de laboratoire fourni par la plate-forme de télécommunications fait allusion à ce type de serveur.

Une troisième catégorie de serveurs est celle des serveurs HTTP ou bien des serveurs Web. Dans ce sens, ce qu'on désigne par serveur joue le rôle d'entités frontales auxquelles accèdent les clients directement à partir de URL bien déterminées. Le premier membre de cette catégorie est le serveur HTTP qui distribue les différents documents HTML correspondant aux laboratoires virtuels. Ainsi, l'organisation du site Web dédié au projet comprend une page d'accueil, des informations renseignant sur les aspects techniques et pédagogiques des laboratoires scientifiques, des moyens adéquats pour déclencher les simulations virtuelles, des liens HTML faisant référence à d'autres sites intéressants, etc. Il faut noter que ce genre de serveurs n'est pas restreint uniquement au site d'accueil des laboratoires virtuels, puisque des outils de base peuvent à leur tour impliquer d'autres serveurs HTTP. Par exemple, l'outil générique de communication entre usagers exige le recours à un serveur HTTP privé à travers lequel les clients peuvent s'échanger de l'information tout en utilisant explicitement l'outil *NetMeeting* de Microsoft. Comme le montre la Figure 5.1, la modélisation actuelle d'une séance de laboratoire virtuel se base sur plusieurs schémas client/serveur.

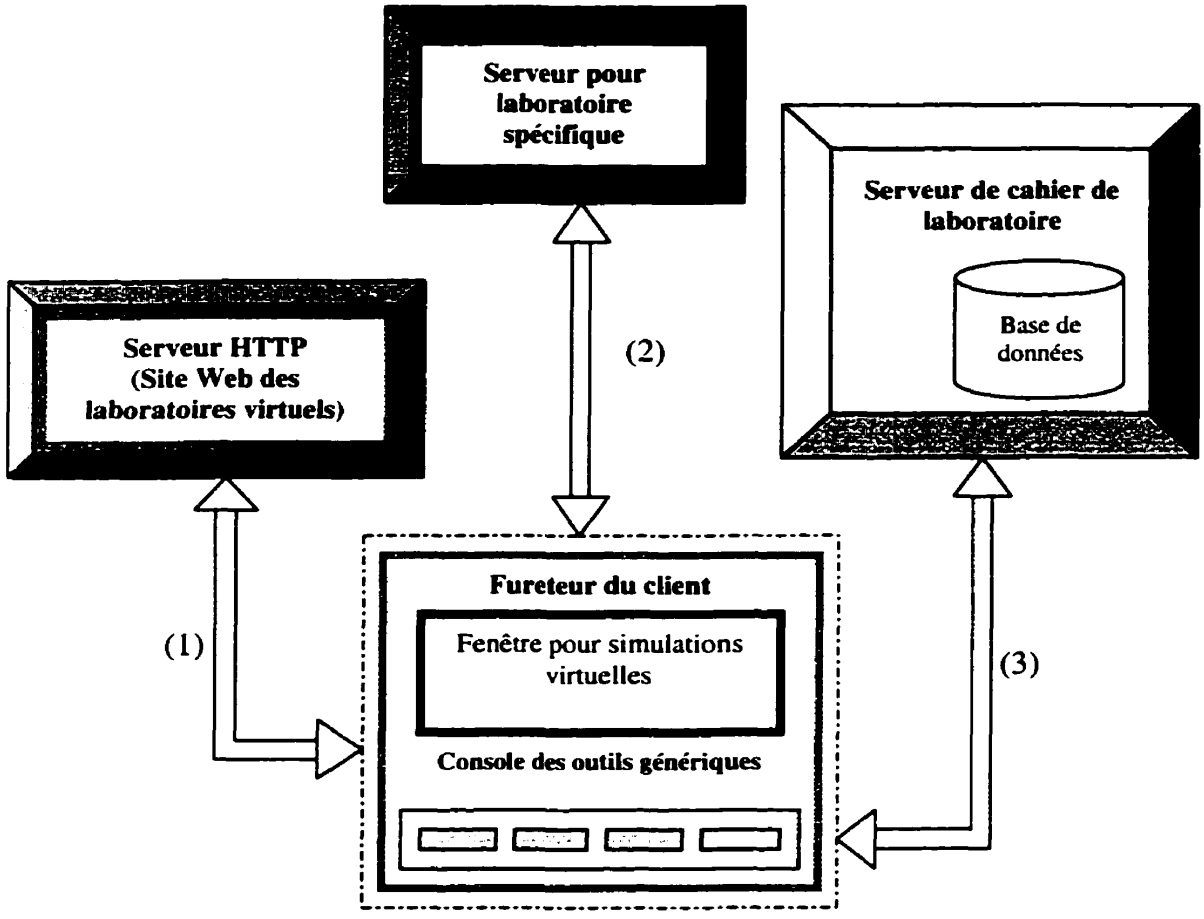
En termes de systèmes distribués, les divers serveurs peuvent coexister sur la même machine ou bien occuper chacun une machine distincte. L'emplacement de l'un desdits serveurs est fonction des caractéristiques matérielles et logicielles du système sur lequel il est implémenté. Pour tout serveur HTTP, les outils exigés sont ceux d'un serveur HTTP classique pouvant exister sur des plates-formes et dans des environnements de nature différente, telles que des stations de travail UNIX ou Windows NT. D'autre part, l'identité de tout serveur relatif à un laboratoire virtuel spécifique est dictée par l'infrastructure informatique exploitée lors de la mise en œuvre du laboratoire en question.

Quant au serveur du cahier de laboratoire, il n'est autre qu'une application autonome écrite en Java JDK 1.2. L'interopérabilité de l'architecture CORBA fait en sorte que la plate-forme sur laquelle ce dernier serveur est implanté est tout à fait quelconque. Or, puisque nous avons utilisé un système de gestion de base de données (SGBD) Access, le serveur est obligatoirement une plate-forme Windows. Le choix de la base de données et celui de la plate-forme du serveur du cahier de laboratoire sont complètement arbitraires. Il en est de même pour d'autres outils génériques pouvant évoquer l'usage d'autres serveurs codés en C, ou tout autre langage répandu d'un environnement CORBA ou DCOM. Même plus, un outil de base peut demander la présence d'autres environnements et d'autres serveurs avec lesquels les interactions auront lieu moyennant des interfaces particulières, telles que HTTP/CGI et RMI.

5.1.2 Interface utilisateur de la plate-forme de télécommunications

Contrairement à l'impression que donne la diversité des technologies utilisées lors de l'implantation de la plate-forme de télécommunications, l'accès à un laboratoire spécifique et aux outils génériques revêt une parure graphique assez conviviale. La souplesse d'utilisation d'un fureteur Web permet en grande partie de dissimuler la

complexité de l'adaptation et de l'intégration de l'ensemble des outils mis en jeu. La page d'accueil du site des laboratoires virtuels est représentée à la Figure 5.2.



Légende :
1- Téléchargement d'une page HTML à l'aide de l'interface HTTP.
2- Déroulement d'une simulation virtuelle utilisant un module d'extension spécifique ou autre.
3- Consultation du cahier de laboratoire à l'aide de requêtes CORBA.

Figure 5.1 Modélisation d'une séance de laboratoire virtuel

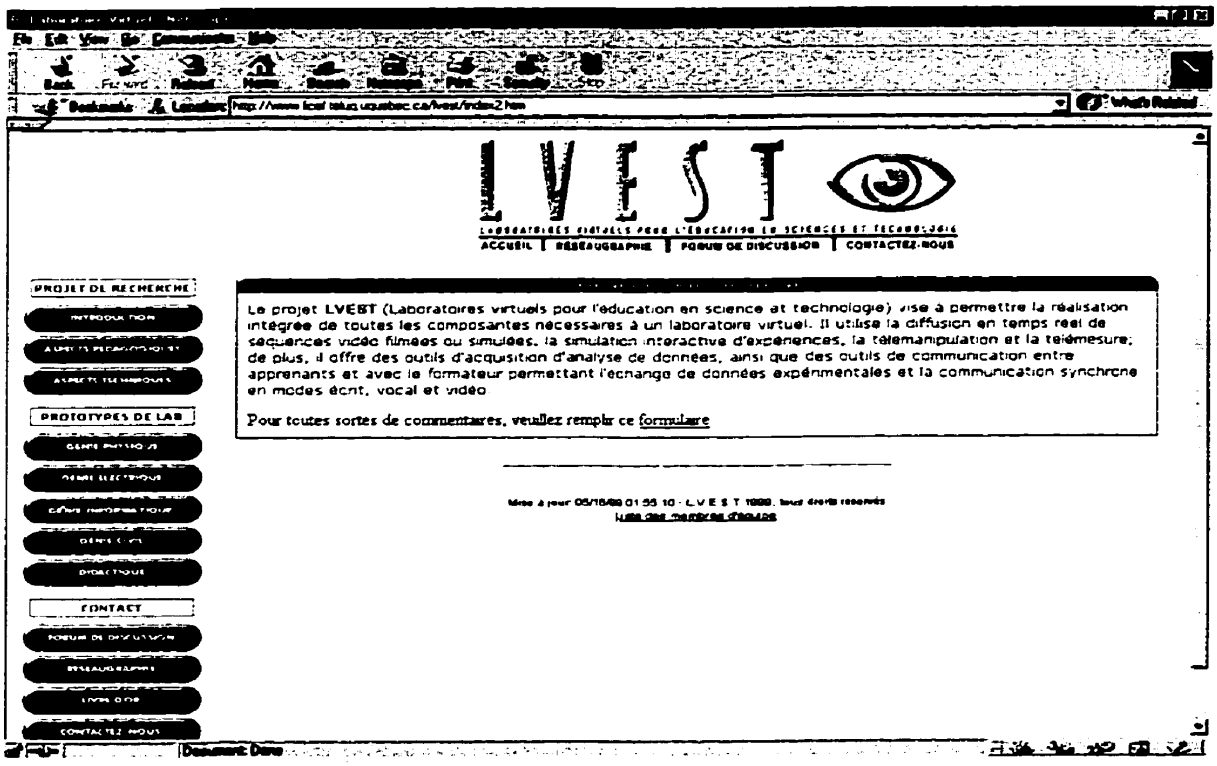


Figure 5.2 Page d'accueil des laboratoires virtuels

Conformément à ce qui a été prévu lors de la conception, le site d'accueil des laboratoires virtuels comporte des sections publiques et d'autres privées. Les pages privées sont réparties selon les différents laboratoires et ne sont accessibles qu'à travers les matricules des étudiants éligibles. La couche d'adaptation aux réseaux d'utilisateurs se charge d'effectuer la validation de chaque usager auprès d'un contrôleur central jumelé à une base de données. L'utilisateur devra fournir son nom d'usager ainsi qu'un mot de passe à l'aide de la fenêtre de contrôle d'accès représentée à la Figure 5.3.

Une fois l'utilisateur autorisé, il pourra bien entamer la manipulation souhaitée. Ainsi, une nouvelle page Web apparaît sur son écran comportant côte à côte la console des outils génériques et la liste des manipulations disponibles. D'autres fenêtres peuvent être utilisées pour que les simulations virtuelles y prennent place. À titre d'illustration, la page Web propre au laboratoire de physique se présente à la Figure 5.4.

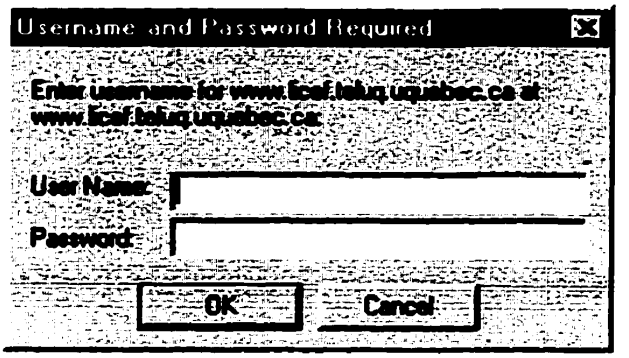


Figure 5.3 Fenêtre de contrôle d'accès

La console d'outils génériques, accompagnant toutes les manipulations, fournit l'accès aux outils de base à l'aide de boutons. Il suffit d'appuyer sur l'un de ces derniers pour activer l'outil correspondant. La Figure 5.4 présente les trois options actuellement implantées, en attendant le développement futur de l'ensemble des outils génériques. On retrouve donc trois boutons et un champ de texte :

Le chronomètre : Une fois ce bouton sélectionné, la temporisation est alors déclenchée. L'étiquette du bouton "Chronomètre" est ensuite remplacée par "Chronomètre Actif". Il suffit d'appuyer sur ce bouton pour arrêter le chronométrage et retrouver l'état initial du bouton de la console. La temporisation enregistrée est affichée dans le champ textuel.

La calculatrice : Ce bouton fait apparaître une calculatrice scientifique. Une fois la fenêtre correspondante ouverte, l'utilisateur aura le choix d'effectuer certaines opérations élémentaires de calcul algébrique et trigonométrique. D'autres options sont aussi possibles telles que la mémorisation de résultats, le changement d'unités angulaires et le tracé de fonctions polynomiales de degré inférieur ou égal à 5. Le développement potentiel au niveau de cette calculatrice, comme d'autres outils génériques, prévoit l'ajout de nouvelles fonctions et l'amélioration de celles qui existent. La version actuelle de notre calculatrice est représentée à la Figure 5.5.

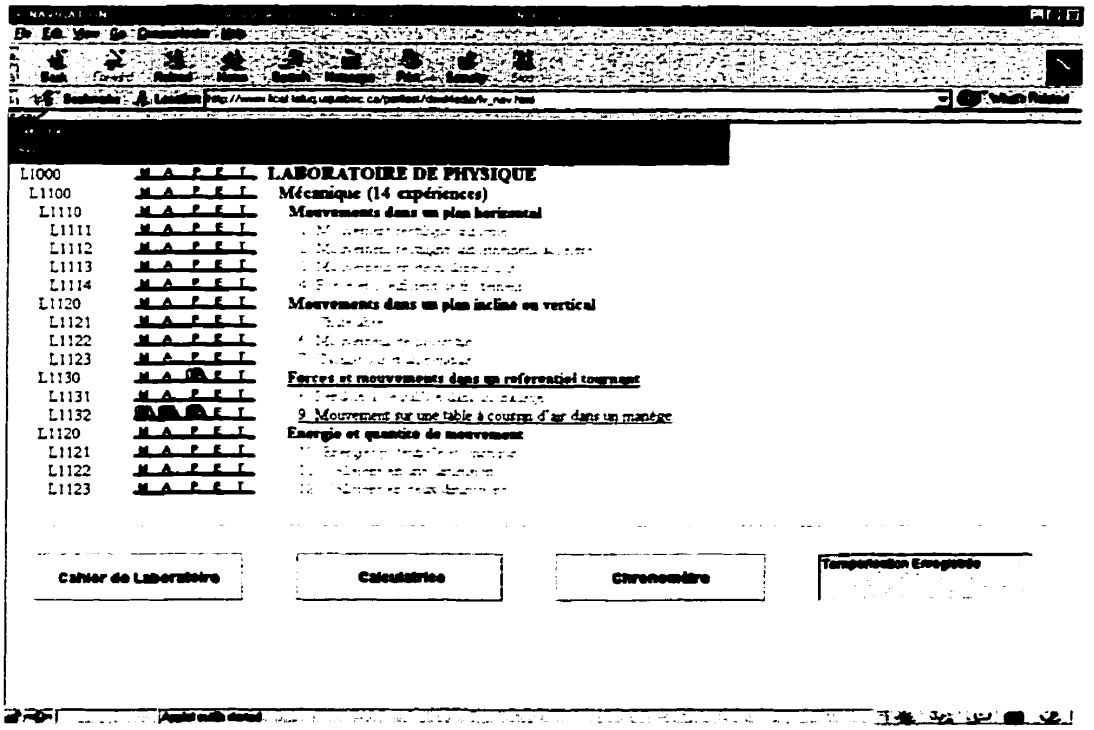


Figure 5.4 Écran d'accueil du laboratoire de physique

Le cahier de laboratoire : La flexibilité de traitement de l'information à l'aide de bases de données offre un choix idéal pour mémoriser les sorties de mesures effectuées au sein de diverses catégories de simulations. Il suffit d'ajouter de nouvelles tables au schéma de la base et de modifier le nombre et le type des champs de ces tables pour que chacune d'elles soit le synonyme d'un cahier de laboratoire déterminé. Les informations relatives aux laboratoires, aux manipulations dérivées, aux enseignants et tuteurs, ainsi qu'aux étudiants empruntent un modèle relationnel au sein duquel des règles bien définies préservent l'intégrité et la consistance des données. Dans cette perspective, chaque manipulation fait partie d'un laboratoire spécifique et est identifiée par un code unique. De même, chaque étudiant assistant à un cours quelconque est récupéré par un matricule unique dans tout le système. Ces deux derniers identificateurs permettent alors de retrouver les résultats d'expérimentation d'un étudiant au cours d'une séance de laboratoire particulière. Au début de tout accès à un cahier de laboratoire, le matricule de

l'étudiant et le code de la manipulation doivent être saisis. Ceci est effectué à travers la fenêtre de la Figure 5.6.

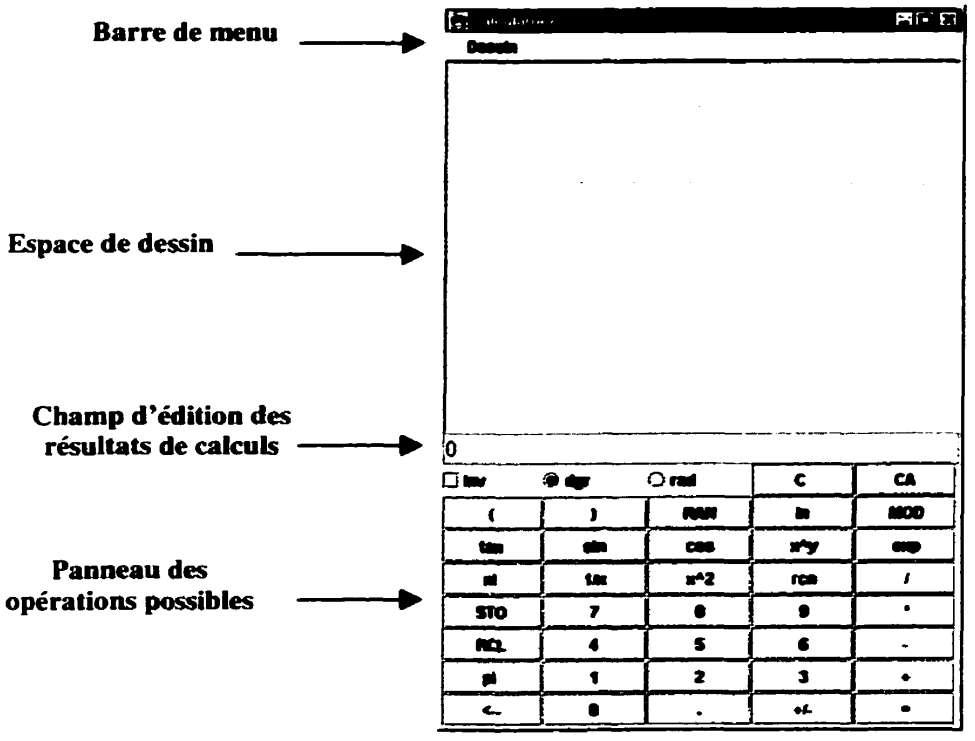


Figure 5.5 Calculatrice scientifique générique

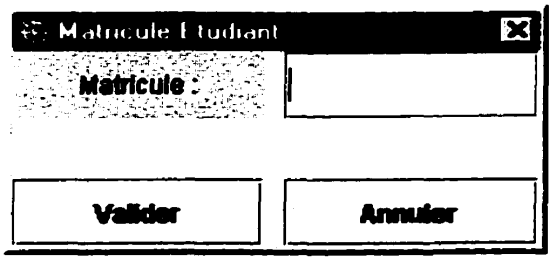


Figure 5.6 Fenêtre de validation d'accès à un cahier de laboratoire

Dès que l'étudiant et la manipulation qu'il effectue sont connus, les résultats correspondants s'affichent à l'écran dans une fenêtre pareille à celle de la Figure 5.7.

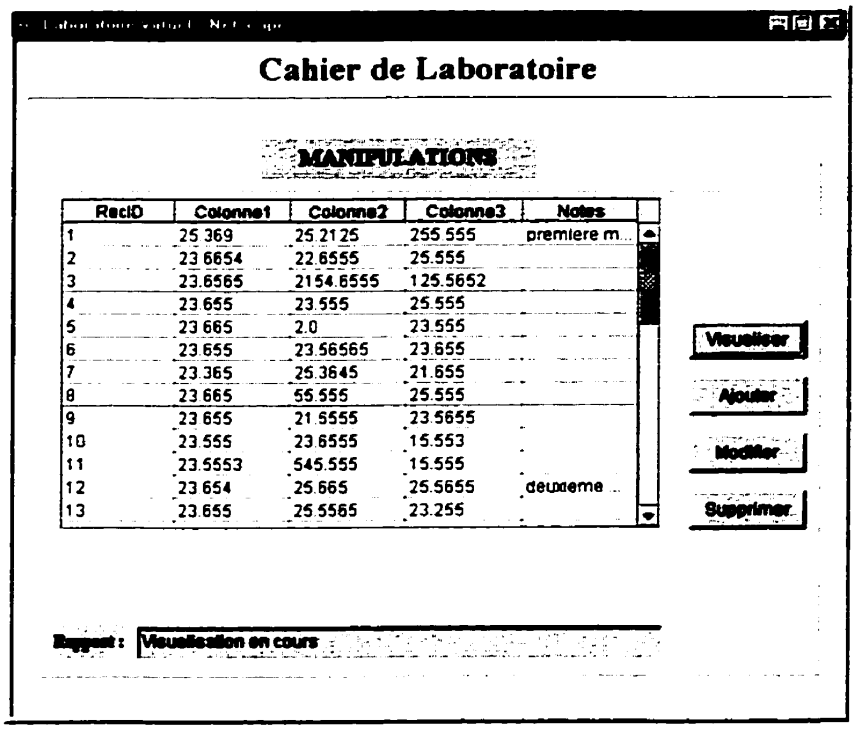


Figure 5.7 Affichage du contenu d'un cahier de laboratoire

Comme c'est le cas avec n'importe quelle base de données, les opérations classiques effectuées dans un cahier de laboratoire sont la visualisation, l'addition, la suppression et la modification de données. Dans les deux derniers cas, l'enregistrement est sélectionné d'abord en cliquant là-dessus par le bouton gauche de la souris, ensuite l'opération en question est déclenchée en cliquant sur le bouton correspondant.

Pour une suppression, l'utilisateur est appelé à confirmer son choix en appuyant sur le bouton "Valider" de la fenêtre représentée à la Figure 5.8. L'option "Annuler" empêche l'élimination physique de l'enregistrement sur la base de données.

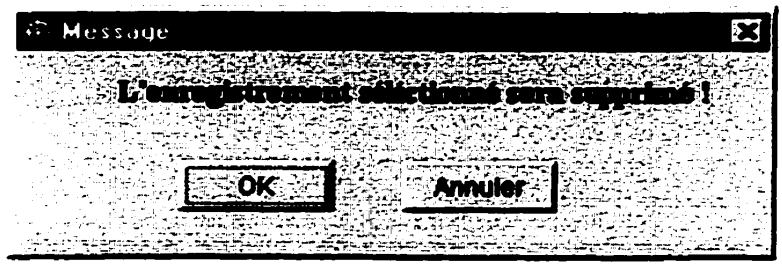


Figure 5.8 Confirmation de suppression de données du cahier de laboratoire

En ce qui concerne la mise à jour d'une information existante, une autre fenêtre renfermant les anciennes données s'ouvre et l'utilisateur est appelé à éditer ses nouvelles valeurs. Cette même fenêtre, représentée à la Figure 5.9, s'ouvre avec des champs vides lorsque l'utilisateur désire ajouter un nouvel enregistrement sans avoir besoin de sélectionner un autre préalablement existant. Le bouton "Valider" permet d'enregistrer les nouvelles données dans les champs respectifs de la base de données. Quant à l'option "Annuler", elle laisse cette dernière intacte. Il faut noter aussi que lors de l'occurrence d'erreur, le message d'erreur correspondant est remis à l'utilisateur dans une boite de dialogue. L'erreur peut apparaître suite à une manipulation erronée de l'interface utilisateur ou bien suite à une opération invalide effectuée sur la base de données.

La manipulation de la base de données et donc des différents cahiers de laboratoire se passe comme s'il s'agit d'une structure de données locale. Le client ne s'aperçoit ni de l'existence de plusieurs serveurs distants ni des mécanismes de communications sous-jacents.

Le développement de nouveaux outils génériques enrichit, sans doute, l'interface utilisateur. En présence d'un grand nombre d'outils de base, ceux-ci seront rangés dans des sous-menus illustrant chacun une catégorie ou bien une fonctionnalité de base. De

tels sous-menus seront activés à partir d'une barre ou menu principal en haut de la page principale de chaque laboratoire spécifique.

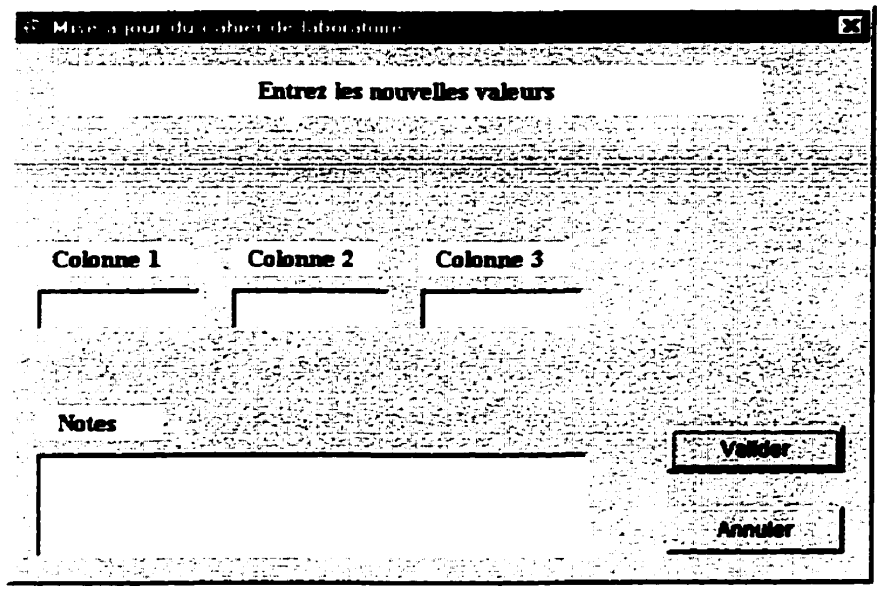


Figure 5.9 Fenêtre de mise à jour du cahier de laboratoire

5.2 Expérimentation de la plate-forme de télécommunications

Dans le but d'évaluer la performance de la plate-forme de télécommunications, nous avons proposé quelques scénarios d'expérimentation reposant sur une certaine infrastructure de communications. Ainsi, les postes de travail se répartissent sur deux localisations géographiquement éloignées l'une de l'autre et sont reliés entre eux par des liens de communications de types différents. La Figure 5.10 renseigne sur la disposition des sites et la nature des supports de communications qui relient ces sites : Internet avec TCP/IP, et la technologie ADSL.

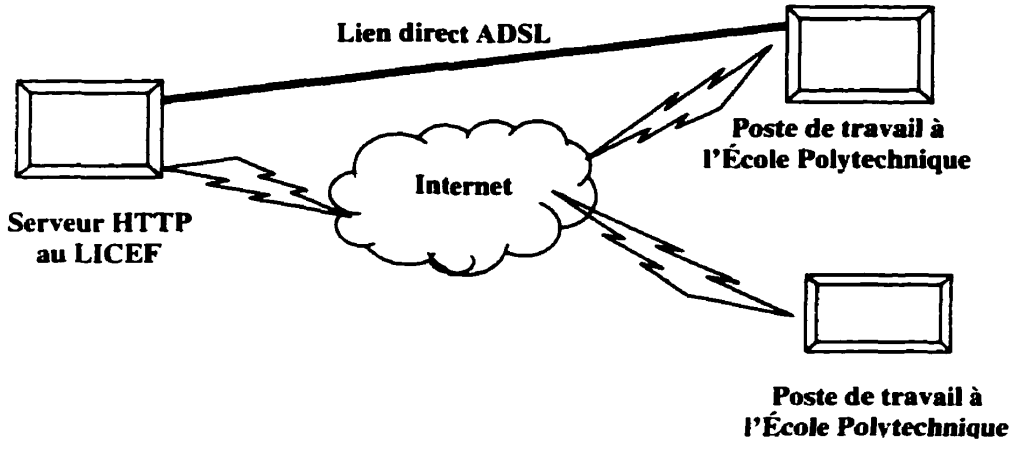


Figure 5.10 Infrastructure de communications expérimentale

Famille de protocoles TCP/IP : Celle-ci est en fait une architecture de communications complète, capable de fournir des fonctionnalités similaires à celles du modèle OSI. Au niveau de l'application, une multitude de protocoles gèrent les échanges d'informations variées, telles que le courrier électronique (SMTP), les documents HTML (HTTP), les fichiers de données (FTP et TFTP), ainsi que les commandes d'administration de réseaux (SNMP). Tous ces protocoles ainsi que d'autres se servent de protocoles de transport orientés connexion (TCP) ou non (UDP).

Au niveau réseau, le protocole IP occupe la place la plus importante et présente, en quelque sorte, l'image de réseau virtuel vis-à-vis des couches supérieures. Les datagrammes IP sont véhiculés sans aucun contrôle de flux ni d'erreur.

La technologie ADSL : C'est une composante du groupe de modems xDSL conçu dans le but d'amener à l'ordre des mégabits par seconde la capacité de transmission des liens déjà installés du réseau téléphonique. Ces derniers sont de plus en plus contraints à des trafics de haut débit provenant essentiellement de la demande croissante de multimédia au sein des applications réparties. Dans sa forme la plus simple, une connexion ADSL

nécessite l'installation de deux émetteurs-récepteurs aux deux extrémités d'une liaison. Les techniques de traitement numérique utilisées pour ADSL permettent théoriquement une capacité de 10 Mbps. Cependant, la capacité courante est à peu près de 6 Mbps pour la liaison descendante (du poste central vers l'utilisateur) contre un peu moins de 1 Mbps pour la liaison ascendante (de l'utilisateur vers le poste central). Les signaux transmis dans un sens et dans l'autre appartiennent aussi à des bandes de fréquences différentes afin de réduire les interférences. Cette asymétrie de répartition de bande passante s'explique du fait que ADSL était prévu, par les compagnies de téléphone, pour la distribution des signaux compressés de vidéo sur demande sur des paires torsadées en cuivre des communications téléphoniques traditionnelles.

La performance des mécanismes de communications exploités au niveau des outils de base et des simulations expérimentales a été évaluée en comparant des graphiques illustrant la variation des délais et du nombre de paquets TCP/IP échangés lors du téléchargement, par HTTP, de l'ensemble des outils spécifiques et génériques du laboratoire de physique, ainsi que lors de la manipulation d'un cahier de laboratoire via CORBA. Le Tableau 5.1 présente les notations adoptées pour les séances de tests.

Tableau 5.1 Notation des séances de tests

Sigle	École Polytechnique	LICEF
Matin	**P1**	**L1**
Après-midi	**P2**	**L2**
Soir	**P3**	**L3**

Les Figures 5.11 et 5.12 présentent respectivement les délais et les nombres de paquets correspondants en utilisant Internet pour accéder au laboratoire de physique qui intègre les outils de base, les informations textuelles et les séquences vidéo des manipulations.

Le laboratoire de physique : Quels que soient la période et l'endroit d'expérimentation, le laboratoire de physique, avec l'ensemble de ses outils génériques et spécifiques, requiert approximativement le même nombre de paquets TCP/IP, dont la taille varie entre 60 et 300 octets. Selon la Figure 5.12, ce dernier oscille autour de 450 paquets pour les pages HTML, 1400 paquets pour une première séquence de vidéo, 5300 paquets pour une deuxième séquence de vidéo et 1400 paquets également pour la manipulation de la table à coussin d'air comportant pratiquement de l'information textuelle.

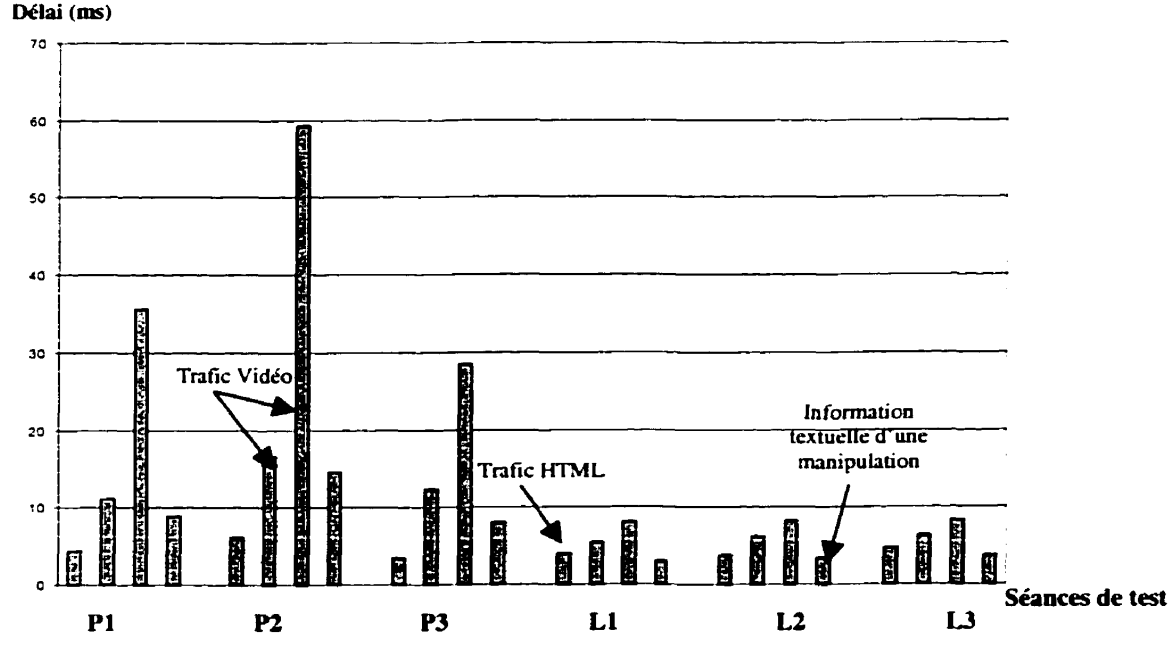


Figure 5.11 Délais de téléchargement du laboratoire de physique

En revanche, les délais de transmission, entre un client et le serveur Web installé au LICEF, dépendent de deux facteurs essentiels : le type local ou étendu du réseau de client et la charge de ce réseau. Pour le premier facteur, les séries de mesures L1, L2 et L3 de la Figure 5.11 présentent des délais pratiquement identiques pour un type de média donné, peu importe la période d'accès au laboratoire virtuel de physique. Dans ce cas, le client et le serveur HTTP sont tous les deux en réseau local faiblement chargé. La congestion de réseau apparaît, par contre, dans les séries de mesure P1, P2 et P3, où le

client est à l'École Polytechnique. Pour les quatre types de médias, les délais sont supérieurs aux valeurs précédentes dans un rapport de 2 ou même 3. Les plus grands délais sont observés surtout en période d'après-midi lorsque les liens sont le plus chargés.

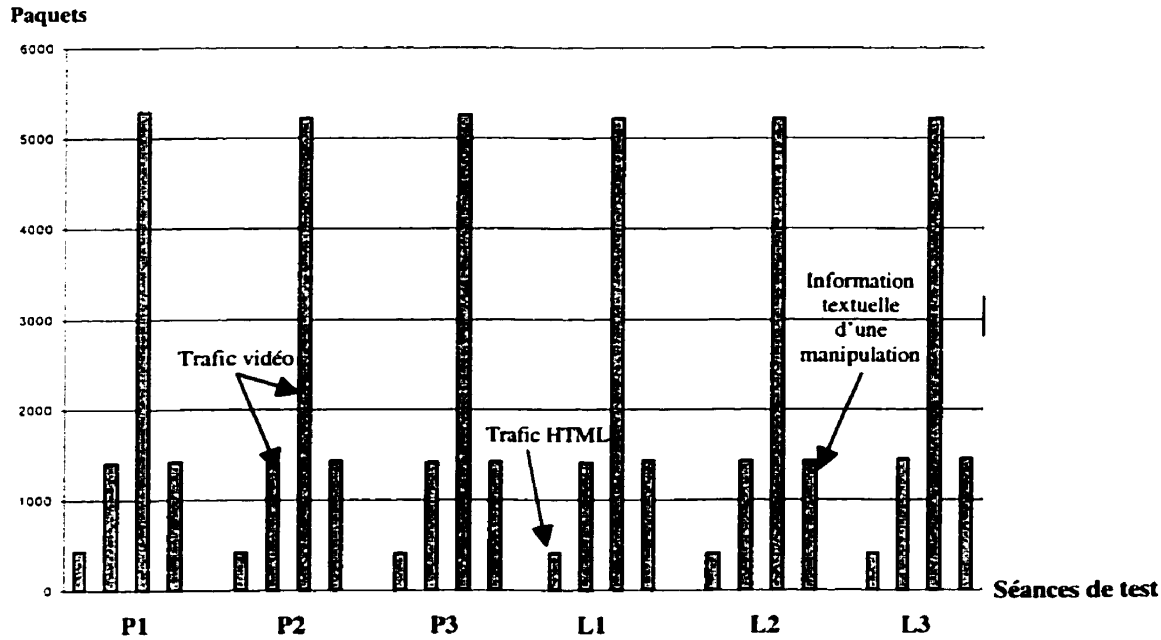


Figure 5.12 Nombre de paquets chargés du laboratoire de physique

Quant aux échanges CORBA, nous avons dressé des graphiques semblables aux précédents, mais qui sont répartis par type d'opération de cahier de laboratoire. Le chronomètre et la calculatrice sont chargés implicitement avec le laboratoire de physique. L'utilisation de ceux-ci est parfaitement statique et ne fait plus intervenir des échanges dynamiques. Par contre, la manipulation du cahier de laboratoire nécessite la communication dynamique avec le serveur de base de données, placé sur une machine à part. Les Figures 5.13, 5.14, 5.15, 5.16 et 5.17 illustrent respectivement le coût sur Internet des opérations d'ouverture du cahier de laboratoire, de l'ajout d'un enregistrement dans la base de données, de modification et de suppression d'un enregistrement existant, ainsi que la fermeture du cahier de laboratoire.

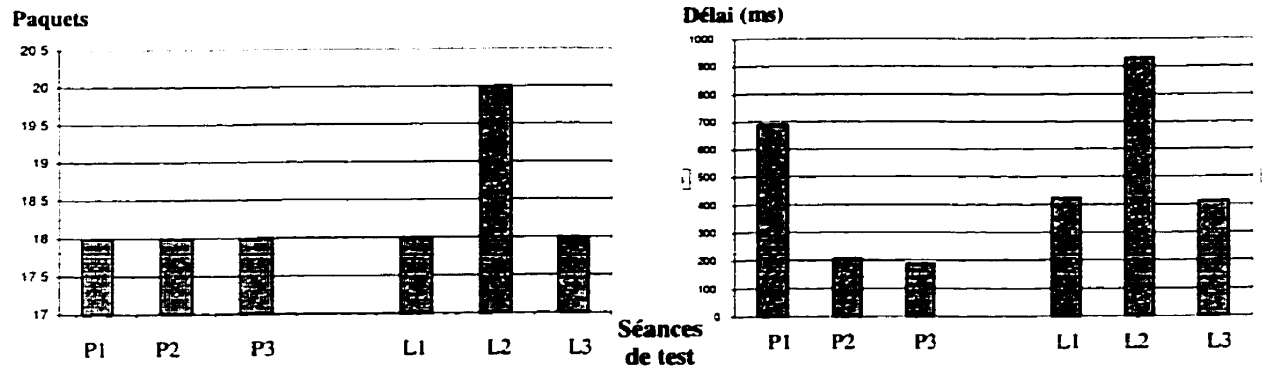


Figure 5.13 Nombre de paquets et délais d'établissement de connexion

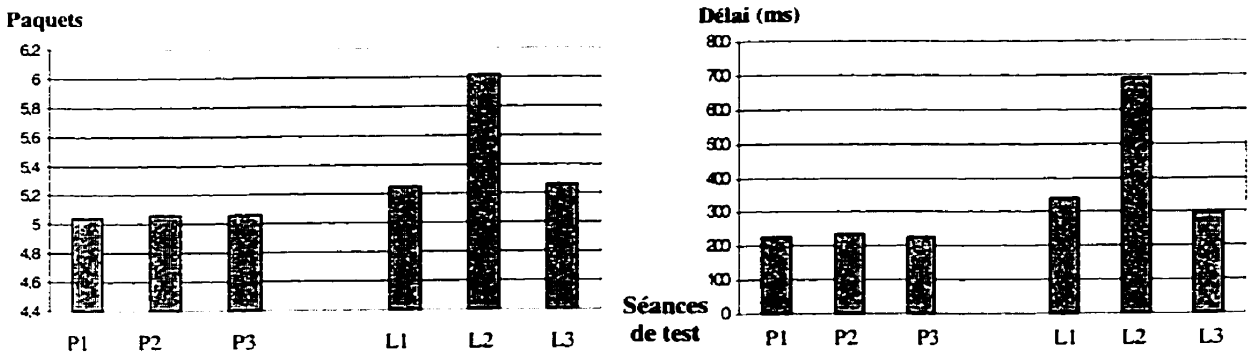


Figure 5.14 Nombre de paquets et délais d'ajout d'un enregistrement

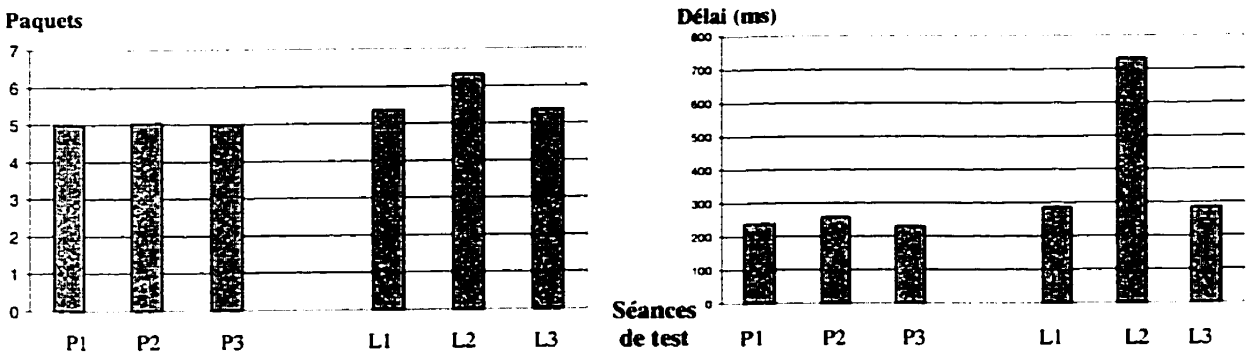


Figure 5.15 Nombre de paquets et délais de modification d'un enregistrement

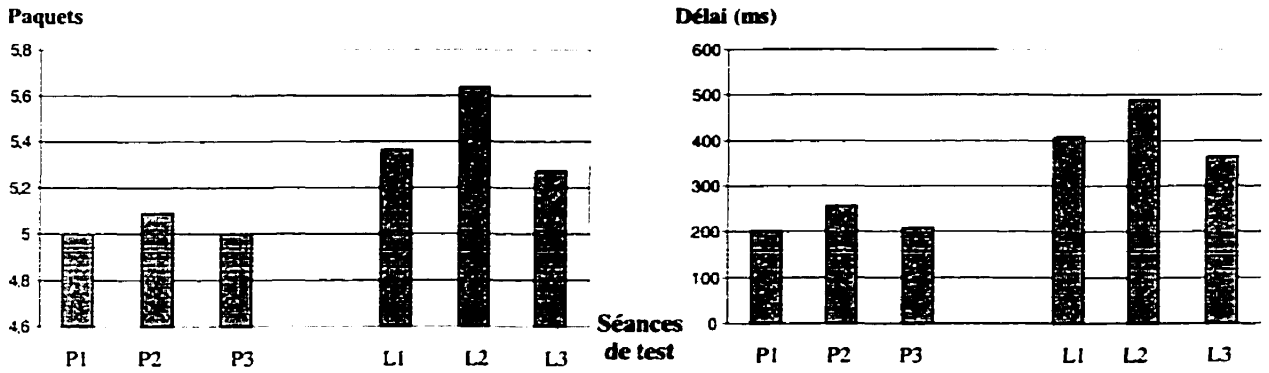


Figure 5.16 Nombre de paquets et délais de suppression d'un enregistrement

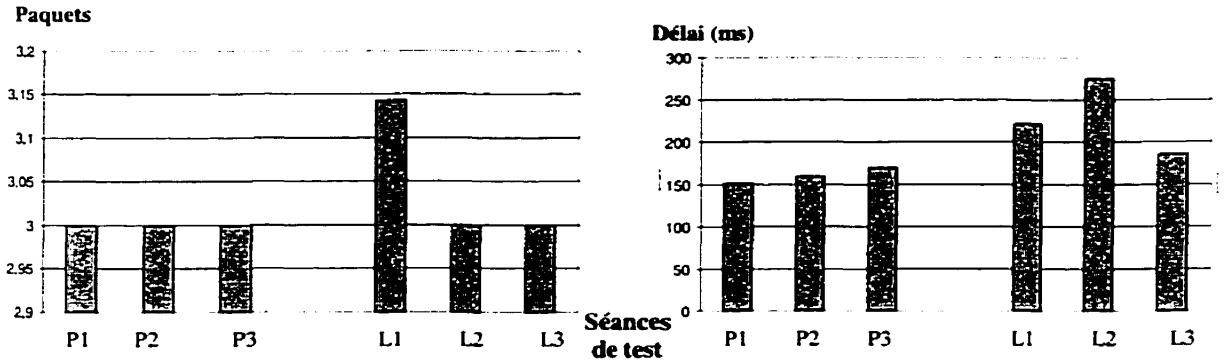


Figure 5.17 Nombre de paquets et délais de rupture de connexion

Le cahier de Laboratoire : Les échanges dynamiques entre un client et le serveur de base de données, installé à l'École Polytechnique pour les tests, font circuler sur les liens des paquets TCP/IP encapsulant des octets utiles ainsi que ceux relatifs au protocole IIOP de CORBA. Chaque fonction sollicitée de la base de données introduit un nombre de paquets et un délai différents. Ces deux derniers sont légèrement supérieurs pour les séries de mesure L1, L2 et L3 effectuées en réseau étendu, par rapport aux séries P1, P2 et P3 réalisées en réseau local. Cependant, le nombre restreint d'octets de chaque requête fait en sorte que l'influence de la charge du réseau reste minime entre les deux cas. En réseau étendu, des pertes de paquets et des retransmissions sont susceptibles d'avoir lieu, ce qui explique la croissance du nombre de paquets et des délais. Les résultats moyens

de nos tests sont représentés, respectivement, en mode local au Tableau 5.2, et en mode étendu au Tableau 5.3.

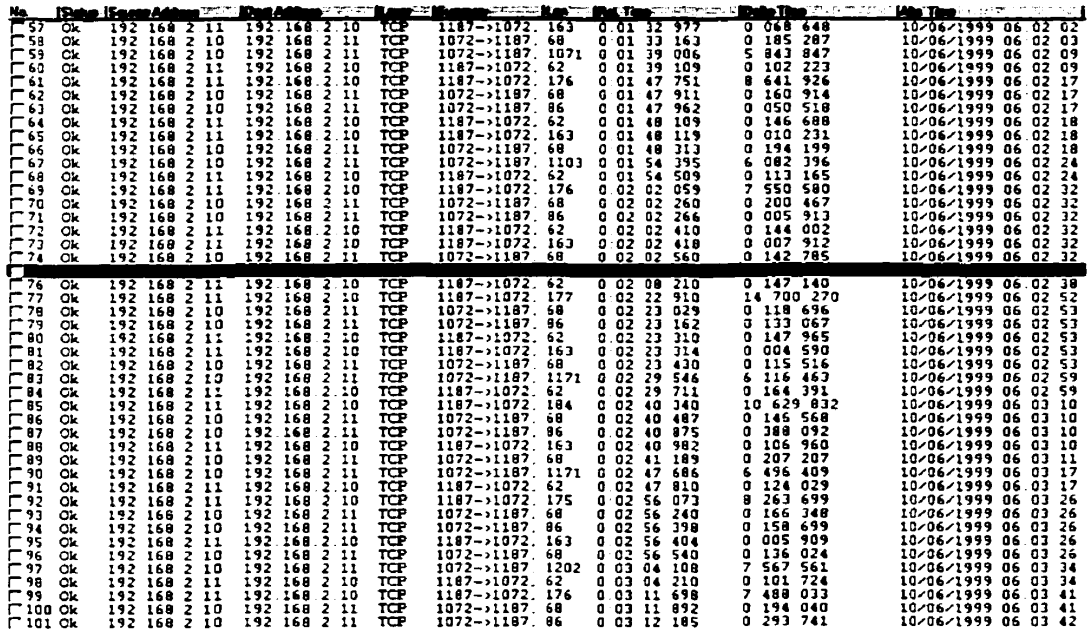
Tableau 5.2 Résultats de manipulation du cahier de laboratoire en mode local avec Internet

Valeur moyenne	**Nombre de paquets**	**Délais (ms)**
Ouverture du cahier de laboratoire	18	360
Ajout d'un enregistrement	5	220
Mise à jour d'un enregistrement	5	230
Suppression d'un enregistrement	5	220
Fermeture du cahier de laboratoire	3	160

Tableau 5.3 Résultats de manipulation du cahier de laboratoire en mode étendu avec Internet

Valeur moyenne	**Nombre de paquets**	**Délais (ms)**
Ouverture du cahier de laboratoire	19	560
Ajout d'un enregistrement	5.5	430
Mise à jour d'un enregistrement	5.6	430
Suppression d'un enregistrement	5.4	420
Fermeture du cahier de laboratoire	3.04	220

Reste finalement à décrire les résultats obtenus en testant la plate-forme de télécommunications sur le lien direct ADSL liant une machine de l'École Polytechnique au serveur HTTP du LICEF installé, dans ce cas, sur la même machine que le serveur de base de données. La Figure 5.18 présente un échantillon des statistiques réalisées sur des requêtes échangées.



No.	Type	Source Address	Dest Address	Protocol	Description	Len	Rel. Time	Delta Time	Abs. Time
57	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 163	0	01 32 977	0 068 648	10/06/1999 06.02.02
58	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	01 33 163	0 185 287	10/06/1999 06.02.03
59	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 1071	0	01 39 006	5 843 847	10/06/1999 06.02.09
60	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	01 39 109	0 102 223	10/06/1999 06.02.09
61	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 176	0	01 47 751	8 641 926	10/06/1999 06.02.17
62	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	01 47 911	0 160 914	10/06/1999 06.02.17
63	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 86	0	01 47 962	0 050 518	10/06/1999 06.02.17
64	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	01 48 109	0 146 688	10/06/1999 06.02.18
65	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 163	0	01 48 119	0 010 231	10/06/1999 06.02.18
66	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	01 48 313	0 194 199	10/06/1999 06.02.18
67	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 1103	0	01 54 395	6 082 196	10/06/1999 06.02.24
68	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	01 54 509	0 113 165	10/06/1999 06.02.24
69	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 176	0	02 02 059	7 550 580	10/06/1999 06.02.32
70	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 02 260	0 200 467	10/06/1999 06.02.32
71	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 86	0	02 02 266	0 005 913	10/06/1999 06.02.32
72	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	02 02 410	0 144 002	10/06/1999 06.02.32
73	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 163	0	02 02 418	0 007 912	10/06/1999 06.02.32
74	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 02 560	0 142 785	10/06/1999 06.02.32
76	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	02 08 210	0 147 140	10/06/1999 06.02.38
77	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 177	0	02 22 910	14 700 270	10/06/1999 06.02.52
78	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 23 029	0 118 696	10/06/1999 06.02.53
79	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 86	0	02 23 162	0 133 067	10/06/1999 06.02.53
80	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	02 23 310	0 147 965	10/06/1999 06.02.53
81	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 163	0	02 23 314	0 004 590	10/06/1999 06.02.53
82	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 23 430	0 115 516	10/06/1999 06.02.53
83	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 1171	0	02 29 546	6 116 463	10/06/1999 06.02.59
84	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	02 29 711	0 164 391	10/06/1999 06.02.59
85	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 184	0	02 40 340	10 629 832	10/06/1999 06.03.10
86	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 40 487	0 146 568	10/06/1999 06.03.10
87	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 86	0	02 40 875	0 388 092	10/06/1999 06.03.10
88	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 163	0	02 40 982	0 106 960	10/06/1999 06.03.10
89	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 41 189	0 207 207	10/06/1999 06.03.11
90	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 1171	0	02 47 686	6 496 409	10/06/1999 06.03.17
91	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	02 47 810	0 124 029	10/06/1999 06.03.17
92	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 175	0	02 56 073	8 263 699	10/06/1999 06.03.26
93	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 56 240	0 166 348	10/06/1999 06.03.26
94	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 86	0	02 56 398	0 158 699	10/06/1999 06.03.26
95	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 163	0	02 56 404	0 005 909	10/06/1999 06.03.26
96	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	02 56 540	0 136 024	10/06/1999 06.03.26
97	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 1202	0	03 04 108	7 567 561	10/06/1999 06.03.34
98	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 62	0	03 04 210	0 101 724	10/06/1999 06.03.34
99	Ok	192.168.2.11	192.168.2.10	TCP	1187->1072. 176	0	03 11 698	7 488 033	10/06/1999 06.03.41
100	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 68	0	03 11 892	0 194 040	10/06/1999 06.03.41
101	Ok	192.168.2.10	192.168.2.11	TCP	1072->1187. 86	0	03 12 185	0 293 741	10/06/1999 06.03.42

Figure 5.18 Requêtes échangées entre client et serveur sur le lien ADSL

Évidemment, la période de test dans ce cas particulier n'a pas d'importance, vu qu'on utilise un circuit fermé auquel aucun utilisateur ne peut se brancher. Les résultats précédents sont groupés dans les Tableaux 5.4 et 5.5.

Tableau 5.4 Résultats de téléchargement du laboratoire de physique avec ADSL

	Pages HTML	Vidéo 1	Vidéo 2	Information textuelle
Nombre de paquets	445	1408	5459	1545
Nombre d'octets	424728	1411097	5595294	1561197
Délai (s)	10.068	21.937	65.533	39.992

Tableau 5.5 Résultats de manipulation du cahier de laboratoire avec ADSL

Valeur moyenne	**Nombre de paquets**	**Délais (s)**
Ouverture du cahier de laboratoire	37	29.119
Ajout d'un enregistrement	8.5	10.761
Mise à jour d'un enregistrement	8.5	11.278
Suppression d'un enregistrement	8	8.751
Fermeture du cahier de laboratoire	3	0.164

La performance du lien ADSL est en fait loin de nos attentes et elle est comparable à celle d'un réseau Internet fortement chargé. Bien que notre lien fonctionne à une vitesse de 600 à 880 Kbps, la cause principale de la croissance des délais par rapport au cas d'Internet revient en fait à l'émission, de la part du serveur, d'un paquet, de taille dépassant souvent un kilo octets, accompagnant chaque requête sauf celle de rupture de connexion. Ce genre de paquet est perceptible à la ligne marquée du tableau de la Figure 5.18, et se manifeste par la croissance du nombre de paquets, donc d'octets, échangés lors de la manipulation du cahier de laboratoire. Quant aux différents flots de média du laboratoire de physique, les nombres de paquets transmis varient de façon raisonnable par rapport au cas précédent, mais le nombre d'octets échangés est beaucoup plus grand et peut atteindre parfois le triple. Ceci confirme la présence sur le lien ADSL de paquets de très grande taille dépassant parfois le kilo octets comme le montre la Figure 5.18.

CHAPITRE VI
CONCLUSION

6.1 Synthèse des travaux et contributions

Dans ce mémoire, nous avons conçu une plate-forme de télécommunications destinée à supporter des laboratoires virtuels distribués. Pour ce faire, il a fallu d'abord établir un modèle conceptuel capable d'intégrer les moyens de télécommunications à un environnement virtuel de laboratoires scientifiques. Pour y parvenir, nous avons proposé une architecture à trois couches : la plus basse tient compte des aspects d'accessibilité aux laboratoires et d'interopérabilité entre les réseaux hétérogènes permettant aux utilisateurs d'avoir accès à l'environnement, la seconde fournit un ensemble d'outils et de fonctionnalités génériques partageables entre plusieurs laboratoires spécifiques, finalement la couche la plus haute assure l'adaptation de ces outils dits de base, aux outils spécifiques existant dans le contexte particulier de chaque laboratoire.

Notre grand souci était de concevoir une plate-forme ouverte, dans le sens où différentes catégories de technologies peuvent coexister et évoluer perpétuellement. Le résultat est un système distribué dans lequel des éléments interopèrent en invoquant des outils et des environnements informatiques tout à fait variés. Ainsi, le site Web des laboratoires virtuels comporte, outre les renseignements pédagogiques d'ordre public, des accès privés aux simulations et expérimentations virtuelles des différents laboratoires spécifiques. Tout au long de sa séance de laboratoire virtuel, l'étudiant aura droit à l'usage de quelques outils génériques activés à partir d'une console disponible sur la page Web donnant accès aux manipulations de ce laboratoire. Pour le moment, on a implémenté trois outils de base : un chronomètre, une calculatrice et un cahier de

laboratoire. Tous les trois font appel au langage de programmation Java, sauf que les deux premiers sont remis au client via l'interface HTTP classique et le troisième utilise l'intergiciel CORBA permettant à un utilisateur d'effectuer les opérations désirées sur une base de données distante constituant réellement un cahier de laboratoire. La configuration actuelle met en œuvre plusieurs serveurs répartis sur des sites géographiques éventuellement distincts. On en reconnaît présentement un serveur HTTP, un serveur pour la gestion des cahiers de laboratoire, ainsi que de multiples serveurs gérant les laboratoires spécifiques. Certaines configurations préalables sont nécessaires sur le site client en vue de pouvoir se servir de la totalité des modules informatiques implantés. Des liens d'accès de nature différente permettent de connecter un client à l'ensemble des serveurs de la plate-forme de télécommunications.

Afin de valider notre implémentation de la plate-forme de télécommunications, nous avons effectué une série de tests interactifs, à des heures différentes du jour, impliquant des liens d'accès différents. Des scénarios d'expérimentation ont été élaborés dans le but d'évaluer la performance des communications lors de l'exploitation des outils génériques et des laboratoires spécifiques. Pour cela, nous avons considéré l'aspect de transfert statique de documents HTML et de classes Java, ainsi que de l'échange dynamique prenant place lors de l'utilisation du cahier de laboratoire.

L'analyse des résultats révèle l'existence d'un trafic énorme généré lors du téléchargement des pages HTML et surtout des séquences vidéo. Par la suite, les échanges dynamiques ne font apparaître qu'un nombre bien déterminé de paquets TCP/IP. La vitesse de transmission diffère d'un type de liens à un autre. Avec une infrastructure Internet, ceci se traduit par des délais variables, mais qui restent plus petits que ceux observés sur le lien spécialisé ADSL. Dans ce dernier cas, des paquets très longs émis d'une façon incontrôlable par le serveur ont masqué en grande partie la performance prévue avec ADSL. D'autre part, la congestion du réseau et le contrat de

qualité de service d'un client peuvent aussi constituer un facteur déterminant pour l'efficacité globale des accès à distance aux laboratoires virtuels.

Tout compte fait, en dépit de l'écart somme toute inévitable entre le modèle conceptuel proposé et sa réalisation, ce mémoire constitue une preuve de faisabilité d'une plate-forme de télécommunications donnant un accès par divers réseaux et supports à des environnements d'apprentissage distribués. De façon plus précise, nos principales contributions sont au nombre de trois que nous résumerons comme suit :

- L'élaboration et la mise au point d'un modèle conceptuel rendant réalisables des activités d'apprentissage par l'utilisation à distance de laboratoires virtuels fonctionnellement comparables à des laboratoires réels ;
- La conception et la réalisation d'un laboratoire générique offrant un partage de fonctionnalités et d'outils de base utilisables dans tous les laboratoires spécifiques et exploitant des structures de stockage et de traitement informatiques complexes incluant une base de données ;
- Le développement d'un environnement interopérable fournissant l'accès à des laboratoires virtuels spécifiques constitués d'éléments hétérogènes distribués interagissant par l'intermédiaire d'une multitude d'interfaces de communications.

6.2 Limitations actuelles et recherches futures

Même si les prototypes que nous avons mis en œuvre se sont avérés satisfaisants, ils ne jouissent que d'un statut temporaire. En effet, les environnements informatiques avec lesquels ont été développés les laboratoires de physique et de génie électrique, respectivement *Director* et *Labview*, nous ont posé quelques contraintes. Par conséquent, nous n'avons réussi qu'une intégration limitée pour les outils de base, sans possibilité d'interaction directe avec les outils spécifiques. En outre, l'exploitation des manipulations virtuelles se fait pour le moment sur des plates-formes particulières, notamment *Windows 95/98/NT* et *Mac*. Afin d'assurer plus de portabilité, il serait

intéressant de pouvoir se servir d'autres plates-formes, notamment celles ayant UNIX et LINUX comme systèmes d'exploitation.

Par ailleurs, la présence seulement de deux prototypes de laboratoires spécifiques possédant des catégories différentes et limitées d'instruments de mesure et d'appareillage expérimental ne nous a pas offert un grand choix pour l'implantation d'outils génériques partagés. D'autre part, nous n'avons pas eu la chance d'utiliser une grande variété de réseaux de communications. Dans un contexte aussi riche en multimédia, il serait donc opportun de tester le comportement de notre implémentation dans le cas de réseaux publics à haut débit genre ATM.

Les recherches futures devraient s'orienter vers l'élaboration d'une méthodologie pour la conception de laboratoires virtuels distribués. Les aspects à prendre en compte sont variés et incluent les considérations ergonomiques pour concevoir des environnements d'apprentissage basés sur le Web (Azuma, 1999). Évidemment, l'interface utilisateur joue un rôle fondamental dans ce secteur puisque seule la facilité d'utilisation de nouvelles technologies peut pousser les responsables des institutions académiques à encourager l'expansion des nouvelles technologies d'apprentissage. Le dialogue entre étudiants et enseignants d'une part, et la convivialité de la présentation du matériel pédagogique d'autre part, sont aussi des aspects relativement importants à prendre en considération (Ausserhofer, 1999).

En ce qui a trait à l'infrastructure de télécommunications qui doit supporter ces laboratoires distribués, une liste non exhaustive d'enjeux a été établie par Collis (1999) ; elle comprend : la sécurité des connexions, l'accès asynchrone, les communications en temps réel, la qualité de service, la connectivité mobile, etc. La prise en compte – fut-ce même partiellement – de l'ensemble de ces facteurs devrait se traduire par une variété de modèles et d'outils logiciels qui viendraient enrichir le cadre méthodologique nécessaire à la conception de tels environnements distribués d'apprentissage.

RÉFÉRENCES

ANUFF, E. (1996). The Java Sourcebook. Wiley Computer Publishing, 5-28.

AUSSERHOFER, A. (1999). Web-Based Teaching and Learning: A Panacea ?. IEEE Communications Magazine, 37, 3, 92-96.

AZUMA, J. (1999). Creating Educational Web Sites. IEEE Communications Magazine, 37, 3, 109-113.

BARDOUT, Y., HAUW, L-H., PAVON, J. et TOMAS, J. (1998). CORBA for Network and Service Management in the TINA Framework. IEEE Communications Magazine, 36, 3, 72-79.

BARTON, S., EYKHOLT, J., FAULKNER, R., KLEIMAN, S., SHIVALINGIAH, A., SMITH, M., STEIN, D., VOLL, J., WEEKS, M., et WILLIAMS, D. (1992). Beyond multiprocessing ... multithreading the SunOS Kernel. Proceedings of the Summer USENIX Conference, San Antonio, Texas, 11-18.

BERNERS-LEE, T., FIELDING, R., et FRYSTYK, H. (1996). Hypertext Transfer Protocol -- HTTP/1.0. RFC 1945.

BOSTICA, B., CALLEGATI, F., CASONI, M., et RAFFAELLI, C. (1999). Packet Optical Networks for High-Speed TCP-IP Backbones. IEEE Communications Magazine, 37, 1, 124-129.

COLLIS, B. (1999). Applications of Computer Communications in Education: An Overview. IEEE Communications Magazine, 37, 3, 82-86.

COLLIS, B. (1996). Tele-Learning in a Digital World: The Future of Distance Learning. International Thomson.

KASSOUF, M., PIERRE, S., LEVERT, C., et CONAN, J. (1999). Modeling a Telecommunication Platform for Remote Access to Virtual Laboratories. 1999 IEEE Canadian Conference on Electrical and Computer Engineering, Edmonton, Alberta, Canada, 127-132.

DALGIC, I., HAMDI, M., HUBAUX, J-P., VERSCHEURE, O., et WANG, P. (1999). Voice Service Interworking for PSTN and IP Networks. IEEE Communications Magazine, 37, 5, 104-111.

HAGGERTY, P. et SEETHARAMAN, K. (1998). The Benefits of CORBA-Based Network Management. Communications Of The ACM, 41, 10, 73-79.

HARKEY, D. et ORFALI, R. (1998). Client/Server Programming with JAVA and CORBA. Second Edition. Wiley Computer Publishing, 3-379.

HENNING, M. (1998). Binding, Migration, and Scalability in CORBA. Communications Of The ACM, 41, 10, 62-71.

PRNJAT, O., et SACKS, L.E. (1999). Integrity Methodology for Interoperable Environments. IEEE Communications Magazine, 37, 5, 126-132.

REDLICH, J-P., SUZUKI, M. et WEINSTEIN, S. (1998). Distributed Object Technology for Networking. IEEE Communications Magazine, 36, 10, 100-111.

RESCORLA, E., et SCHIFFMAN, A. (1996). The Secure HyperText Transfer Protocol. Internet Draft.

SCHMIDT, D.C. (1998). Evaluating Architecture for Multithreaded Object Request Brokers. Communications Of The ACM, 41, 10, 54-60.

SEETHARAMAN, K. (1998). The CORBA Connection. Communications Of The ACM, 41, 10, 34-36.

SIEGEL, J. (1998). OMG Overview : CORBA and the OMA in Enterprise Computing. Communications Of The ACM, 41, 10, 37-43.

TJANDRA, D.A. et WONG, S.T.C. (1999). A Digital Library for Biomedical Imaging on the Internet. IEEE Communications Magazine, 37, 1, 84-91.

VINOSKI, S. (1998). New Features for CORBA 3.0. Communications Of The ACM, 41, 10, 44-52.

ANNEXE I
CONFIGURATION PRÉALABLE DU SITE CLIENT POUR ÉTABLIR DES CONNEXIONS CORBA

La démarche présentée ci-dessous permet à une Applet Java, chargée par un serveur HTTP, de se connecter au serveur CORBA à partir de la machine client. Il s'agit d'affecter à la source du code Java des permissions spéciales afin de désactiver les barrières de sécurité limitant la liberté d'actions d'un code mobile transmis sur réseau. La source en question est en fait un répertoire bien déterminé du serveur HTTP. Quant à l'outil utilisé pour configurer la politique de sécurité du fureteur, il est fourni dans l'environnement JRE du JDK 1.2 installé auparavant sur le site client, et lancé à l'aide de la commande "*policytool*". La fenêtre représentée à la Figure AI.1 s'ouvre sur l'écran. Le champ "*Keystore*" permet d'ajouter des étiquettes d'identification des auteurs de code, ce qui en renforce certes le contrôle mais dont on peut s'abstenir présentement.

La première ligne indique le nom du fichier, s'il existe, où la politique de sécurité du fureteur est sauvegardée. Par défaut, ce fichier porte le nom de "*.java.policy*", et est situé dans le répertoire courant de l'utilisateur, soit "C:\WINDOWS" sur une plate-forme Windows 95/98. Dans la fenêtre précédente, il faut ajouter une source de code en appuyant sur le bouton "*Add Policy Entry*". La fenêtre, représentée à la Figure AI.2, s'ouvre et permet à l'utilisateur de spécifier directement l'emplacement ou bien l'URL de la source de code à la ligne "*CodeBase*". Par exemple, cet emplacement peut être "http://www.licef.teluq.uquebec.ca/lvest/physique/dswMedia/-". Le tiret à la fin de la ligne accorde la permission a tout code provenant du répertoire spécifié, ainsi que de n'importe lequel de ses sous-répertoires. La signature, identifiant l'auteur du code, n'est pas obligatoire et le champs "*SignedBy*" peut rester vide.

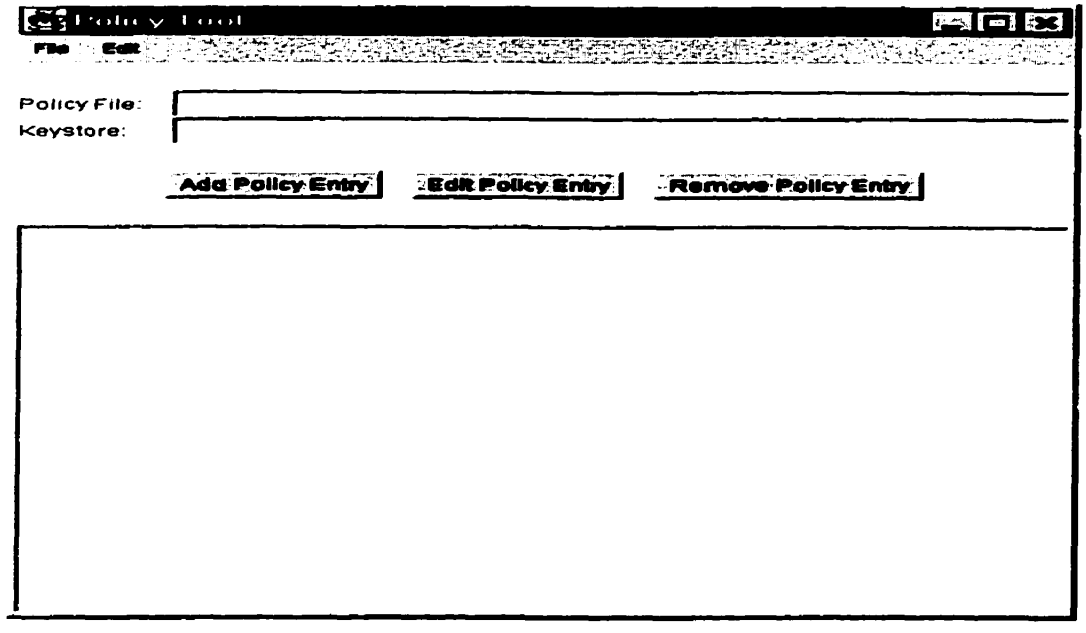


Figure AI.1 Fenêtre de mise à jour de la politique de sécurité

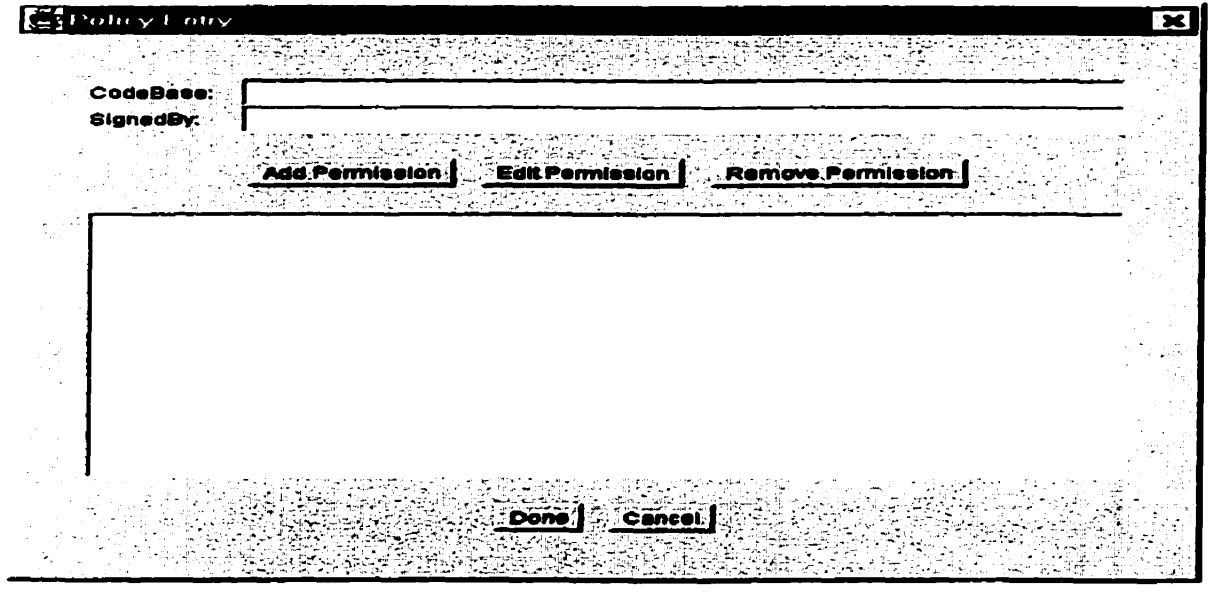


Figure AI.2 Fenêtre d'attribution des permissions à une source de code

Une fois l'emplacement de la source introduit, il faut sélectionner le bouton "*Add Permission*" afin d'associer les permissions convenables. Ceci conduit à une troisième fenêtre comme l'illustre la Figure AI.3.

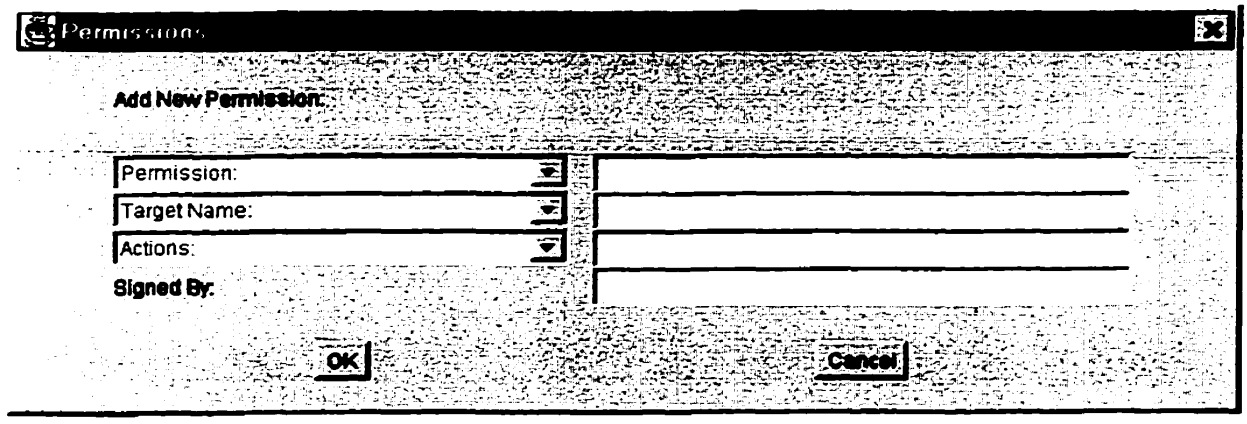


Figure AI.3 Fenêtre de sélection de permissions

En sélectionnant la liste "*Permission*" de droite, un menu apparaît sur l'écran avec plusieurs options. Il s'avère convenable d'en choisir l'option "*AllPermission*". Avec ce choix, les options "*Target Name*" et "*Actions*" seront désactivées automatiquement. Il suffit ensuite d'appuyer sur le bouton "*OK*" pour que la fenêtre AI.3 se ferme. La fenêtre AI.2 se fermera à son tour une fois la touche "*Done*" est appuyée. Pour la fenêtre AI.1, il faut sauvegarder ce fichier dans le répertoire courant de l'utilisateur. Ceci est faisable en choisissant l'option "*Save As*" du menu "*File*". Le nom du fichier est bien "*.java.policy*". Une fois le fichier sauvegardé, il suffit de choisir l'option "*Exit*" du menu "*File*" pour quitter.

La démarche décrite ci-dessus correspond en fait à une première manipulation de politique de sécurité d'un fureteur. Pour les manipulations ultérieures, le fichier "*.java.policy*" sera accessible par défaut et les sources privilégiées seront facilement modifiées ou supprimées, respectivement à l'aide des options "*Edit Policy Entry*" et "*Remove Policy Entry*" de la fenêtre AI.1. Les permissions d'une source existante seront

aussi manipulables à l'aide des options "*Edit Permission*" et "*Remove Permission*" de la fenêtre AI.2. Les permissions et les fonctionnalités de l'outil "*policytool*" du JDK 1.2 sont expliquées en détail sur le site http://java.sun.com.