Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche
researchportal.unamur.be
University of Namur

THESIS / THÈSE

MASTER EN SCIENCES INFORMATIQUES

Valves électroniques et portail Web à l'Institut: analyse, prototype, aspects théoriques

Collet, Jean-Marc

Award date:
2003

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 19. Apr. 2024

Facultés Universitaires Notre-Dame de la Paix, Namur

Institut d'Informatique

Année académique 2002-2003

Valves électroniques et Portail Web à l'Institut. Analyse, Prototype, Aspects théoriques

Jean-Marc COLLET

Mémoire présenté en vue de l'obtention du grade de Licencié en Informatique

JM COLLET
Mémoire

ERRATA

- Page 67, en bas de page

  → lire figure 16 et non figure 18

- Page 102, figure 39

  → lire HTTP et non HTTPS

Résumé

Dans le contexte de mise en œuvre d'un portail Web à l'Institut d'Informatique (Facultés Universitaires Notre Dame de la Paix), celui-ci souhaite s'équiper d'une application de valves électroniques destinées à la communication entre les étudiants et les secrétariats.
Ce travail décrit l'analyse business, fonctionnelle et technique, ainsi que le développement et l'intégration du prototype « *WebfacInfo* » répondant à ces besoins.
Par ailleurs, le concept et les mécanismes de la plate-forme J2EE sont décrits en détail, et la notion de portail Web ainsi que les aspects d'authentification qui s'y rapportent sont abordés.
Enfin compte tenu du grand nombre de systèmes potentiellement accessibles au travers d'un portail Web, les principes d'authentification simplifiée sont également abordés.

Mots clés

- Portail Web
- J2EE
- Prototype « *WebFacInfo* »
- Authentification simplifiée

Abstract

In the context of bringing a Web Portal into play at the Computer Science Department (Facultés Universitaires Notre Dame de la Paix), it is desirable to equip it with an electronic notice board system, in order to improve communication between students and the secretariats.
This topic describes the business, the functional and the technical analysis as well as the implementation and integration of the "*WebFacInfo*" prototype, answering those requirements.
Moreover, the concept and mechanisms of the J2EE platform are described in detail, and the Web Portal features as well as the related authentication problematic are broached.
Finally, regarding the number of potentially reachable systems through a portal, simplified authentication principles are broached too.

Keywords

- Web Portal
- J2EE
- « *WebFacInfo* » prototype
- Simplified authentication

Avant-propos

C'est le printemps, ...

Printemps 97, je m'en souviens comme si c'était hier, je me trouve dans l'appartement d'une amie (je ne sais pas encore que je l'épouserai quelques années plus tard) et pour la première fois de mon existence je me retrouve assis derrière un PC, avec une lettre de dix lignes à écrire. Bilan : 2 heures de galère et une trentaine d'appels au secours à la pauvre qui tente en vain de suivre un film à la télévision !
Je suis fasciné par cet outil et dans les mois qui vont suivre, je n'aurai de cesse de comprendre son fonctionnement.
Très vite, je me demanderai ce qui se cache derrière l'écran, et aussi qu'elles sont les réelles possibilités d'un ordinateur au-delà de l'hébergement d'une simple application bureautique.

Printemps 98, je découvre un encart publicitaire, dans mon journal quotidien, annonçant l'organisation d'une licence en Informatique à horaire décalé dans ma bonne vieille ville de Charleroi, l'occasion est belle, je me laisse tenter ...

Automne 98, année préparatoire, auditoire des 1ere Licences pour 1 cours en tronc commun, le prof s'appelle Olivier .... Bonaventure, tout un programme. Avec quelques condisciples, on se regarde, on ne comprend rien, on se demande ce qu'on fait là, mais on y est et on y reste.

Printemps 99, premier projet de programmation en « Pascal », une montagne, avec du recul, une souris, mais à chaque étape suffit sa peine.

Printemps 00, fin de première Licence, un très mauvais souvenir, un gros passage à vide et une énorme envie de tout plaquer là, j'en viens à bout, mais je ne sais pas trop comment.

Printemps 01, après une énorme poussée d'adrénaline (merci le Simplex), je navigue en eaux calmes, la deuxième Licence se termine paisiblement et je décide de faire mon mémoire rapidement pour en finir au plus vite.

Printemps 02, je termine mon .... année sabbatique, je ne m'en veux même pas de ne pas avoir tenu mes résolutions. Cette fois, je suis décidé à en finir, d'ailleurs je viens de choisir mon sujet de mémoire et je rencontre mon promoteur la semaine prochaine.

Printemps 03, dans une centaine de pages, l'aventure (s'en était une) se termine, elle aura finalement duré 5 ans, elle était belle, elle valait la peine d'être tentée, je ne regrette pas de l'avoir vécue.

C'est le printemps, ...

Remerciements

Avant d'aller plus loin, je souhaite adresser mes plus vifs remerciements à toutes les personnes qui de près ou de loin m'ont aidé, soit tout au long de mes études, soit dans la réalisation de ce travail.

Mes remerciements s'adressent à :

- Dans mon entourage familial :
  - *Mon épouse*, Nadine, qui m'a incité à entreprendre cette formation et qui a dû ensuite assumer
  - *Mes enfants*, Julie, Thomas, Margaux et Léa avec qui j'ai pas mal de temps à rattraper
  - *Ma mère*, Janine, qui a financé mes études
- Dans mon entourage universitaire :
  - Radu Cotet, *mon Promoteur*
  - Philippe Thiran, *Assistant*
  - Anne-Marie Breny, *Secrétaire*
  - Babette Di Guardia, *Secrétaire*
  - Annick Massart, *Secrétaire*
  - Isabelle Daelman, *Secrétaire*
  - Gisèle Henrard, *Secrétaire*
- Dans mon entourage professionnel :
  - Marc Stern, *Security & Network Specialist* (CSC)
  - Jean-Michel Pollet, *System Architect* (CSC)
  - Jacques Van Roey, *Business Architecture* (CSC)
  - Vanessa Fourez, *Oracle Team* (CSC)
  - Philippe Van De Woestyne, *System Engineer* (DIV)

Table des matières

1 GLOSSAIRE 12

2 INTRODUCTION 14

3 ANALYSE DES BESOINS 15

3.1 CONTEXTE 15
3.1.1 Les Facultés Universitaires Notre Dame de la Paix 15
3.1.2 L'Institut d'Informatique 15
3.1.3 Les organes de l'Institut d'Informatique 16
3.1.3.1 Le conseil 16
3.1.3.2 Le conseil restreint 16
3.1.3.3 L'Assemblée Générale 16
3.1.3.4 Le bureau exécutif 16
3.1.3.5 La Commission de contact 17
3.1.3.6 La Commission de l'enseignement 17
3.1.3.7 La Commission de la recherche 17
3.1.4 Les secrétariats 18
3.1.4.1 Le secrétariat de Direction 18
3.1.4.2 Le secrétariat administratif 18
3.1.4.3 Le secrétariat aux études 19
3.1.4.4 Les secrétariats étudiants 19
3.2 LA SITUATION ACTUELLE 20
3.2.1 Le secrétariat de Direction 20
3.2.1.1 Le Calendrier Académique 20
3.2.1.2 Les listes de membres du Conseil et du BEX 20
3.2.1.3 L'envoi de convocations 20
3.2.1.4 La tenue des ordres du jour 20
3.2.1.5 La gestion des procès verbaux 20
3.2.1.6 Annonces diverses au personnel de l'Institut 20
3.2.2 Le secrétariat aux études 21
3.2.2.1 La diffusion de la fiche signalétique 21
3.2.2.2 La transmission des résultats de l'épreuve d'admission 21
3.2.2.3 L'édition et la diffusion de la liste des cours par étudiant 21
3.2.2.4 La réservation de locaux et de matériel 21
3.2.3 Les secrétariats étudiants 21
3.2.3.1 La diffusion du guide de l'étudiant 21
3.2.3.2 L'édition, la mise à jour et la diffusion du trombinoscope 21
3.2.3.3 La gestion de la problématique des syllabus (LIHD uniquement) 21
3.2.3.4 L'édition, la mise à jour et la diffusion des listes d'étudiants 22
3.2.3.5 L'édition, la mise à jour et la diffusion des listes de cours par étudiants 22
3.2.3.6 L'édition et la transmission au CGI des listes d'étudiant 22
3.2.3.7 La gestion des inscriptions aux cours à option 22
3.2.3.8 L'édition, la mise à jour et la diffusion des horaires 23
3.2.3.9 La gestion des (in)disponibilités des professeurs pour les examens 23
3.2.3.10 L'édition, la mise à jour et la diffusion des horaires d'examen 23
3.2.3.11 La gestion des horaires de passage aux examens oraux 23
3.2.3.12 La diffusion des cotes d'examen 23

3.2.3.13 La gestion des inscriptions aux examens 23
3.2.3.14 La diffusion des offres d'emploi 23
3.2.3.15 La diffusion d'avis en tout genre 24
3.2.3.16 La réservation de locaux et de matériel 24
3.3 LES BESOINS 24
3.3.1 Périmètre du projet 24
3.3.1.1 Fonctionnalités comprises dans le projet 24
3.3.1.1.1 La gestion des listes d'étudiants par année d'étude 24
3.3.1.1.2 La gestion des inscriptions aux cours à option 24
3.3.1.1.3 La gestion des dispenses 25
3.3.1.1.4 La gestion des listes de cours par étudiant 25
3.3.1.1.5 La gestion de la problématique des syllabus 25
3.3.1.2 Fonctionnalités exclues du projet 25
3.3.2 Arborescence des buts 26
3.4 LES EXIGENCES 27
3.4.1 Les exigences sur les acteurs 27
3.4.1.1 Les acteurs 27
3.4.1.2 Les contraintes 27
3.4.1.2.1 Contraintes sur la secrétaire 27
3.4.1.2.2 Contraintes sur les étudiants 28
3.4.1.2.3 Contraintes sur les utilisateurs λ 28
3.4.2 Les exigences sur le système 28
3.4.2.1 Les contraintes fonctionnelles 28
3.4.2.2 Les contraintes non fonctionnelles 29

4 ANALYSE FONCTIONNELLE 30

4.1 LA MODELISATION 30
4.2 LES USES CASES 30
4.2.1 Diagramme des uses cases 31
4.2.2 Uses cases 32
4.2.2.1 Consulter liste d'étudiants 32
4.2.2.2 Modifier liste d'étudiants 32
4.2.2.3 Consulter liste de cours d'1 étudiant 33
4.2.2.4 Consulter sa liste de cours 34
4.2.2.5 Choisir ses options 34
4.2.2.6 Accepter nouveau cours à option 35
4.2.2.7 Accepter/refuser une dispense 36
4.2.2.8 Demander une dispense 37
4.2.2.9 Choisir ses syllabus 37
4.2.2.10 Consulter choix de syllabus 38
4.3 SCHEMA DE ROBUSTESSE 38

5 CONCEPTION DE LA DB 40

5.1 VALIDATION 40
5.2 SCHEMA CONCEPTUEL 41
5.3 TRANSFORMATIONS 41
5.4 SCHEMA RELATIONNEL 42
5.5 GENERATION DES SCRIPTS 43

6 ASPECTS THEORIQUES SUR LES ARCHITECTURES MULTI NIVEAUX ET LA PLATE FORME J2EE 44

6.1 LES BESOINS DES ENTREPRISES 44
6.2 LES ARCHITECTURES MULTI-NIVEAUX 45
6.2.1 Architecture à 2 niveaux 45
6.2.2 Architecture à 3 niveaux 45
6.2.3 Architecture multi-niveaux 47
6.3 ARCHITECTURE D'ENTREPRISE 48
6.4 LA PLATE-FORME "JAVA 2 ENTERPRISE EDITION" (J2EE) 48
6.4.1 Généralités 48
6.4.2 La plate-forme J2EE 49
6.4.2.1 L'environnement d'exécution de J2EE 49
6.4.2.2 Les API J2EE 49
6.4.3 L'architecture J2EE 50
6.4.4 les conteneurs 52
6.4.4.1 Les composants applicatifs 52
6.4.4.2 Les descripteurs de déploiement 52
6.4.4.3 Le contrat des composants 52
6.4.4.4 Les API des services des conteneurs 53
6.4.4.5 Les Services déclaratifs 53
6.4.4.6 Les autres services des conteneurs 53
6.4.5 Les technologies J2EE 54
6.4.5.1 Les technologies de composants 54
6.4.5.2 Les technologies de services 55
6.4.5.3 Les technologies de communication 55
6.4.5.4 Le XML 56
6.4.6 Les produits existants 57
6.4.6.1 Les plate-formes concurrentes 57
6.4.6.2 les serveurs d'application J2EE 57

7 ASPECTS THEORIQUES SUR LE CONCEPT DE PORTAIL 58

7.1 INTERNET 58
7.1.1 Généralités 58
7.1.2 Historique 59
7.1.3 Perspectives 59
7.2 PORTAIL, DEFINITION, CLASSIFICATION 60
7.2.1 Définition 60
7.2.2 3 générations de portails 61
7.2.3 Classification 62
7.3 « IPLANET PORTAL » ARCHITECTURE 63
7.3.1 composants de base 63
7.3.1.1 iPlanet Portal Server 64
7.3.1.2 iPlanet Profile Server 64
7.3.1.3 iPlanet Portal Server Gateway 64
7.3.1.4 firewall 64
7.3.2 Architecture à 2 machines 64
7.3.3 architecture multi-machines 66
7.3.4 architecture multi-machines avec load balancing 67
7.3.5 « iPlanetPortal » à l'Institut 67
7.4 « IPLANET PORTAL » ORGANISATION ET STRUCTURE 68
7.4.1 Le niveau « root » 69
7.4.2 Le niveau domaine 69

7.4.3 Le niveau rôle *69*
7.4.4 Le niveau utilisateur *69*
7.5 SECURITE, AUTHENTIFICATION 71
7.5.1 installation *71*
7.5.2 Personnalisation *72*

8 PORTAIL ET AUTHENTIFICATION SIMPLIFIEES 73

8.1 GENERALITES 73
8.2 PASSWORD MANAGEMENT 74
8.2.1 Password management par propagation *75*
8.2.2 Password management par centralisation *76*
8.3 SINGLE SIGN ON (SSO) 77
8.3.1 SSO par usage de scripts *78*
8.3.2 SSO par usage de cookies *79*
8.4 AUTHENTICATION MANAGEMENT INFRASTRUCTURE (AMI) 79
8.5 AVANTAGES ET INCONVENIENTS 80
8.5.1 Avantages *80*
8.5.2 Inconvénients *80*
8.6 EN RESUME 81

9 ANALYSE TECHNIQUE 82

9.1 WEBFACINFO 82
9.2 ARCHITECTURE APPLICATIVE "WEBFACINFO" 82
9.2.1 La vue *83*
9.2.2 Le contrôleur *83*
9.2.3 Le modèle *84*
9.3 COMPOSANTS ACTIFS 84
9.3.1 Les objets du domaine *84*
9.3.2 L'objet MainServlet *85*
9.3.3 Les objets ValueObject *85*
9.3.4 Les objets Command *86*
9.3.5 L'objet VOFactory *86*
9.3.6 Les objets EJB *87*
9.3.7 L'obbjet ViewDispatcher *87*
9.3.8 les objets de persistance *88*
9.4 CLASS DIAGRAM 90
9.5 FLOW DIAGRAM 91

10 PROTOTYPE (IMPLEMENTATION) 92

10.1 DOCUMENTATION TECHNIQUE 92
10.1.1 les packages *92*
10.1.1.1 Le package fundp.jmc.command 92
10.1.1.2 Le package fundp.jmc.configuration 92
10.1.1.3 Le package fundp.jmc.constant 92
10.1.1.4 Le package fundp.jmc.domain 92
10.1.1.5 Les packages fundp.jmc.ejb.*EJB 92
10.1.1.6 Le package fundp.jmc.factory 92
10.1.1.7 Le package fundp.jmc.persistence 92
10.1.1.8 Le package fundp.jmc.servlet 92
10.1.1.9 Le package fundp.jmc.view 92
10.1.1.10 Le package fundp.jmc.vo 93

10.1.2 la structure de l'application *93*
10.1.2.1 le cascading style sheet 93
10.1.2.2 le javascript 93
10.1.2.3 les properties 93
10.1.2.4 les images 93
10.1.2.5 les jsp 93
10.1.2.6 les sources 94
10.1.2.7 les sources compilées 94
10.1.3 le serveur d'application *94*
10.1.3.1 Localisation 94
10.1.3.2 Déploiement WebFacInfo 94
10.1.3.3 Administration 97
10.1.4 accès à la base de données *98*
10.1.5 Accès à l'application *100*
10.2 CODE SOURCE 100

11 DEPLOIEMENT 101

11.1 WEBFACINFO A L'INSTITUT 101
11.2 INFRASTRUCTURE GLOBALE 101
11.3 L'APPLICATION ET L'INFRASTRUCTURE RESEAU 102
11.4 CONFIGURATION SERVEUR 103
11.5 LES RELATIONS PORTAIL – WEBFACINFO 103

12 CONCLUSIONS 105

13 BIBLIOGRAPHIE 106

13.1 LIVRES 106
13.2 SITES WEB 106
13.3 AUTRES RÉFÉRENCES 107

Table des figures

FIGURE 1 : ARBORESCENCE DES BUTS 26
FIGURE 2 : DIAGRAMME DES ACTEURS 27
FIGURE 3 : DIAGRAMME DES USES CASES 31
FIGURE 4 : SCHEMA DE ROBUSTESSE GLOBAL 38
FIGURE 5 : SCHEMA DE ROBUSTESSE DETAIL 39
FIGURE 6 : DB - SCHEMA CONCEPTUEL 41
FIGURE 7 : DB - SCHEMA RELATIONNEL 42
FIGURE 8 : ARCHITECTURE A 2 NIVEAUX 45
FIGURE 9 : ARCHITECTURE A 3 NIVEAUX 46
FIGURE 10 : ARCHITECTURE MULTI-NIVEAUX 47
FIGURE 11 : ARCHITECTURE D'ENTREPRISE 48
FIGURE 12 : ARCHITECTURE J2EE 51
FIGURE 13 : ARCHITECTURE DE CONTENEUR 53
FIGURE 14 : 3 GENERATIONS DE PORTAILS 61
FIGURE 15 : PORTAIL VERTICAL/HORIZONTAL 62
FIGURE 16 : IPS – ARCHITECTURE À 2 MACHINES 65
FIGURE 17 : IPS – ARCHITECTURE MULTI - MACHINES 66
FIGURE 18 : IPS – ARCHITECTURE MULTI – MACHINES AVEC LOAD BALANCING 67
FIGURE 19 : IPP A L'INSTITUT 68
FIGURE 20 : IPP ORGANISATION GENERALE 70
FIGURE 21 : IPP EXEMPLE D'ORGANISATION 70
FIGURE 22 : IPP UN EXEMPLE D'IMPLANTATION 71
FIGURE 23 : PASSWORD MANAGEMENT PAR PROPOAGATION 76
FIGURE 24 : PASSWORD MANAGEMENT PAR CENTRALISATION 77
FIGURE 25 : SSO PAR USAGE DE SCRIPTS 78
FIGURE 26 : SSO PAR USAGE DE COOKIES 79
FIGURE 27 : MIDDLEWARE AMI 80
FIGURE 28 : MIDDLEWARE D'AUTHENFICATION 81
FIGURE 29 : ARCHITECTURE WEBFACINFO 82
FIGURE 30 : ARCHITECTURE WEBFACINFO - DÉTAIL 84
FIGURE 31 : CLASS DIAGRAM 90
FIGURE 32 : FLOW DIAGRAM 91
FIGURE 33 : EJB – FICHIERS ARCHIVES .JAR 96
FIGURE 34 : WEBFACINFO.WAR 97
FIGURE 35 : ACCES A LA DB 98
FIGURE 36 : CHOIX D'ACCES A LA DB 99
FIGURE 37 : ACCES A LA DB – CLASS DIAGRAM 99
FIGURE 38 : INFRASTRUCTURE FUNDP 101
FIGURE 39 : WEBFACINFO ET INFRASTRUCTURE RESEAU 102
FIGURE 40 : CONFIGURATION SERVEUR 103
FIGURE 41 : RELATIONS PORTAIL – WEBFACINFO 104

1 GLOSSAIRE

Termes et acronymes	Définition
AMI	Authentication Management Infrastructure
BEX	Bureau exécutif
CGI	Comité de Gestion des ressources Informatiques
EJB	Enterprise Java Bean
FUNDP	Facultés Universitaires Notre Dame de la Paix
GSM	Global System for Mobile telecomunication
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfert Protocol
iPP	iPlanet Portal
iPS	iPlanet Server
IRC	Internet Relay Chat
IVR	Interactive Voice Response
J2EE	Java 2 Enterprise Edition
JAF	Javabean Activation Framework
JDBC	Java Data Base Connectivity
JDK	Java Development Kit
JMS	Java Message Service
JNDI	Java Naming and Directory Interface
JSP	Java Server Page
JTA	Java Transaction Architecture
LIHD	Licence en Informatique à Horaire Décalé
MVC	Modèle-Vue-Contrôleur (Model-View-Controler)
OTP	One-Time Password
PDA	Personal Digital Assistant
PIN	Personal Identification Number

PKI	Public Key Infrastructure
Portail	Un portail est un point d'entrée unique, permettant de fédérer et de diffuser des informations à des communautés d'utilisateurs
RMI	Remote Method Invocation
RMI-IIOP	RMI sur Inter ORB Protocol
SSL	Secure Socket Layer
SSO	Single Sign On
UML	Unified Modeling Language
Utilisateur λ	Toute personne ayant accès au système qui ne soit ni une secrétaire, ni un étudiant.
Valves	Panneau d'affichage destiné à la communication entre une instance administrative et une communauté qui s'y rapporte.
Valves électroniques	Version informatisée des valves traditionnelles
WAP	Wireless Access Protocol
WebFacInfo	Application de valves électroniques à l'Institut faisant l'objet de ce travail
WLS	Weblogic Server
XML	eXtensible Markup Language

2 INTRODUCTION

Arrivé au terme d'une formation universitaire en Informatique, au cours de laquelle les matières enseignées couvraient un large éventail de domaines (analyse, méthodologie, paradigmes et méthodes de programmation, ingénierie des bases de données, réseaux, sécurité, gestion de projet, ...) il nous semblait intéressant de produire un travail constituant tant que faire se peut une synthèse de l'apprentissage, couvrant un maximum de domaines, permettant la mise en évidence de la compréhension et de l'application des différents concepts, ainsi que notre aptitude à appréhender une situation réelle et à mener à bien un véritable « projet ».

C'est la raison pour laquelle notre choix s'est porté sur l'analyse et le développement d'une application, depuis le stade initial de l'identification des besoins, jusqu'au développement d'un prototype parfaitement fonctionnel.

Nous avons pu de la sorte aborder les principales étapes du cycle de vie d'un logiciel (abstraction faite des différents modèles existants) :

- Analyse business
- Analyse fonctionnelle
- Architecture
- Développement
- Tests
- Déploiement
- Maintenance corrective

Sans oublier la création d'une base de données opérationnelle.

Parallèlement à ce développement, la mise en place d'un portail Web à l'Institut d'Informatique, et la perspective de l'intégration de notre application au sein de celui-ci, nous a permis de nous intéresser à la problématique des portails, en fournissant une approche de ce concept (définitions, classification, architecture, exemples).
La problématique de l'authentification à répétition, liée à la variété du contenu accessible via un portail est également abordée, et des solutions sont envisagées, au travers des technologies d'authentification simplifiées, telles que le « Password management », le « Single Sign On » ou l'« Authentication Management Infrastructure ».

Enfin, notre application est développée en Java, sur la plate forme J2EE, dont nous présentons le concept et les possibilités, de manière détaillée, dans le cadre d'architectures multi-niveaux.

L'ensemble du travail, richement illustré (41 figures), est rédigé dans un style direct et la phraséologie est réduite à sa plus simple expression. Ainsi nous avons favorisé la synthèse sous forme de listes à puces chaque fois que c'était possible.
Chaque concept envisagé est illustré par un schéma simple et aisément compréhensible.

Les différents chapitres théoriques ont été intercalés entre les chapitres d'analyse et de développement, afin d'introduire les notions nécessaires à la compréhension globale des matières abordées dans ceux-ci.

3 ANALYSE DES BESOINS

3.1 Contexte

3.1.1 Les Facultés Universitaires Notre Dame de la Paix

Les Facultés Universitaires Notre Dame de la Paix, implantées à Namur, constituent à ce jour le plus ancien établissement d'enseignement libre de Belgique.[Xavier Rousseaux, 1992]

Leur origine remonte à la création en 1831, par les Jésuites, du collège Notre Dame de la Paix, dont la vocation première était l'enseignement de la Philosophie et des Sciences Mathématiques et Physiques, mais avec un programme peu différencié de celui de l'enseignement secondaire, jusqu'en 1845, date à laquelle l'enseignement s'oriente vers la préparation des étudiants aux Candidatures en Philosophie et Lettres et en Sciences.

Si dès 1845, la loi consacre le caractère universitaire des formations délivrées, il faudra attendre 1890 pour que soient organisés les premiers cycles en Philosophie, en Sciences et en Sciences Médicales, et 1929 pour que les Facultés obtiennent une autonomie complète pour les 3 Candidatures précitées.

A partir de 1945, les Facultés se développent fortement grâce notamment à l'obtention de subsides gouvernementaux, pour assurer de nos jours l'organisation de cycles complets, jusqu'au Doctorat, en Sciences Economiques et Sociales, Mathématiques, Physiques, Chimiques, Biologiques et en Informatique, ainsi que l'organisation des Candidatures en Philosophie, Histoire, Langues et Littératures classiques, romanes et germaniques, Droit, Sciences Politiques, Médicales, Vétérinaires et Pharmaceutiques.

A ce jour, les Facultés comptent plus de 4500 étudiants et plus de 100 chercheurs, ce qui les placent en très bonne position juste derrière les plus grandes universités belges

3.1.2 L'Institut d'Informatique

Créé en 1970 au sein des Facultés Universitaires Notre Dame de la Paix, l'Institut d'Informatique a pour missions l'enseignement et la recherche.

Il assure depuis lors la formation de Licenciés et de Maîtres en Informatique.

Depuis sa création, l'Institut d'Informatique s'est enrichi de Candidatures en Sciences Economiques option Informatique (1976), de Candidatures en Sciences Mathématiques option Informatique (1982), de la Licence en Informatique à horaire décalé (1998) et enfin de Candidatures en Informatique (2002).

3.1.3 Les organes de l'Institut d'Informatique

3.1.3.1 Le conseil

Le conseil de l'Institut est composé:

- De tous les enseignants
- De représentants du personnel:
  - Scientifique
  - Administratif
  - Technique
- De représentants des étudiants
- De représentants de la Faculté des Sciences Economiques, Sociales et de Gestion
- D'un représentant de l'Association des Anciens

Sous réserve des dispositions fixées par le statut organique de l'Université, le Conseil de l'Institut possède, aux fins de sa gestion, tous pouvoirs à l'exception de ceux qui sont expressément confiés à un autre organe.

3.1.3.2 Le conseil restreint

Le conseil restreint est composé uniquement des membres académiques du conseil de l'Institut.

Il traite en particulier les problèmes relatifs au personnel (par exemple: les promotions)

3.1.3.3 L'Assemblée Générale

L'Assemblée Générale réunit, sous la présidence du Directeur, l'**ensemble du personnel** académique, scientifique, administratif et technique de l'institut. Elle est convoquée par le Directeur au moins une fois chaque année.

Organe de contact et d'information, l'Assemblée générale tient un rôle consultatif à l'égard du Conseil de l'institut.

3.1.3.4 Le bureau exécutif

Le bureau exécutif de l'Institut est composé de:

- *Membres de droit:*
  - Le Directeur
  - Le Secrétaire Académique
  - Les responsables des Unités d'Enseignement

- *Membres élus:*

  - Un représentant du personnel académique
  - Un représentant du personnel scientifique permanent
  - Un représentant du personnel administratif et technique
  - Le Secrétaire scientifique

Le bureau exécutif assure la gestion courante de l'Institut. A ce titre, il a notamment les compétences suivantes :

- Il examine les dossiers d'inscription litigieux
- Il est souverain en matière de dispenses d'assistance à un ou plusieurs cours.
- Il fait rapport annuellement au Conseil de l'Institut des activités d'enseignement et de recherche menées à l'Institut.

3.1.3.5 La Commission de contact

La Commission de contact de l'Institut est composée:

- Du Directeur
- Du Secrétaire académique
- D'un représentant du personnel scientifique non définitif
- D'un membre de la Commission d'enseignement
- D'un représentant du secrétariat des étudiants
- D'un représentant étudiant de chaque année d'étude

La Commission de contact est un lieu où s'échange régulièrement l'information entre étudiants et enseignants, afin de rendre aussi efficace que possible l'enseignement organisé par l'institut.

3.1.3.6 La Commission de l'enseignement

La Commission de l'enseignement est composée:

- De trois membres du personnel académique ou scientifique permanent
- Du Secrétaire scientifique
- D'un membre du personnel scientifique non définitif

La Commission de l'Enseignement propose au Conseil de l'institut des mesures destinées à favoriser l'amélioration de la pédagogie; elle est l'organe relais entre l'institut et la Commission de l'Enseignement des Facultés et enfin, elle assure l'organisation de la journées portes ouvertes.

3.1.3.7 La Commission de la recherche

La Commission de la recherche est composée:

- De deux membres du personnel académique
- D'un membre du personnel scientifique permanent
- De deux représentants du personnel scientifique non définitif

La Commission de la recherche est chargée d'examiner les problèmes relatifs aux études de doctorats et de veiller à l'administration de la vie scientifique à l'Institut.

3.1.4 Les secrétariats

Les tâches administratives de l'Institut d'Informatique sont assurées par 6 secrétariats ayant chacun leurs prérogatives:

- *Les 3 secrétariats administratifs*
  - Le secrétariat de Direction
  - Le secrétariat administratif
  - Le secrétariat aux études
- *Les 3 secrétariats étudiants*
  - Le secrétariat pour la LIHD
  - Le secrétariat pour les candidatures du jour
  - Le secrétariat pour les licences du jour

Par soucis de lisibilité, ne seront décrites ci-après que les tâches respectives de ces secrétariats ayant un impact sur la suite du présent travail.

3.1.4.1 Le secrétariat de Direction

Le secrétariat de Direction a, entre autres, pour mission d'assurer le fonctionnement des différents organes de l'Institut d'Informatique.

Il prend en charge:

- L'élaboration et la diffusion du Calendrier académique sur base duquel les activités de l'Institut et de ses organes sont fixées.
- L'édition et la mise à jour de la liste des membres de ces organes
- L'envoi des convocations aux différentes réunions planifiées
- L'édition et la diffusion des ordres du jour pour ces réunions.
- L'édition, la diffusion et l'archivage des procès verbaux consécutifs à ces réunions.
- La diffusion des communications du Directeur (offre de bourses, prix, voyages, colloques, ...)

3.1.4.2 Le secrétariat administratif

Le secrétariat administratif a pour mission d'assurer le bon fonctionnement des U.E. (Unités d'enseignement).

En autres, il prend en charge :

- l'organisation des voyages scientifiques
- l'organisation des réunions prévues par les membres des U.E.
- ...

3.1.4.3 Le secrétariat aux études

Le secrétariat aux études assure la prise en charge des candidats depuis leur premier contact avec l'Institut d'Informatique, jusqu'à leur admission.

La procédure d'admission est complexe et variable en fonction du cycle envisagé, mais aussi en fonction du niveau académique et de l'origine des candidats.
Ainsi, certains candidats seront admis sur base d'un dossier et d'autres, devront présenter un examen d'admission.

Cette procédure d'admission débouche dans tous les cas sur la rédaction d'une fiche signalétique pour chaque candidat, synthétisant les données utiles de chacun.

Le secrétariat aux études est en charge de:

- La diffusion de la fiche signalétique
- La transmission des résultats de l'épreuve d'admission aux candidats ayant participé à celle-ci.
- L'édition et la diffusion de la liste des cours par étudiant.
- La réservation de locaux et de matériel

3.1.4.4 Les secrétariats étudiants

Les prérogatives des 3 secrétariats étudiants sont identiques à quelques détails près. Les tâches spécifiques à l'un ou à l'autre de ces secrétariats feront l'objet de remarques dans le texte.

Ces secrétariats sont en charge de:

- La diffusion du guide de l'étudiant
- L'édition, la mise à jour et la diffusion du trombinoscope
- La gestion de la problématique des syllabus (LIHD uniquement)
- L'édition, la mise à jour et la diffusion des listes d'étudiants
- La mise à jour et la diffusion des listes de cours par étudiant
- L'édition et la transmission au CGI des listes d'étudiants pour création de login.
- La gestion des inscriptions aux cours à option
- L'édition, la mise à jour et la diffusion des horaires
- La gestion des (in)disponibilités des professeurs pour les examens
- L'édition, la mise à jour et la diffusion des horaires d'examen
- La gestion des horaires de passage aux examens oraux
- La diffusion des cotes d'examen
- La gestion des inscriptions aux examens
- La diffusion des offres d'emploi
- La diffusion d'avis en tous genres (séminaires, délibération, repas de fin d'année, etc …) uniquement pour LIHD.
- La gestion des badges d'accès uniquement pour LIHD.
- La réservation de locaux et de matériel

3.2 La situation actuelle

3.2.1 Le secrétariat de Direction

3.2.1.1 Le Calendrier Académique

Le calendrier académique est rédigé manuellement.

Il est à noter que plusieurs personnes participent à son élaboration, entraînant la circulation du document entre ces différentes personnes, jusqu'à l'obtention du document final.

Ce document est finalement transmis par courrier aux membres du Bureau Exécutif et du Conseil pour approbation. Il est ensuite distribué à tout le personnel de l'Institut ainsi qu'aux étudiants.

3.2.1.2 Les listes de membres du Conseil et du BEX

La gestion des listes de membres consiste en la tenue à jour de listes Acces destinées à l'impression d'étiquettes et servant de base au mailing.

3.2.1.3 L'envoi de convocations

Les différentes convocations sont envoyées sur bases des listes tenues au point précédent. En cas de retard lors du postage, une convocation par e-mail est adressée aux membres.

3.2.1.4 La tenue des ordres du jour

Les ordres du jour sont établis par le Directeur 1 semaine avant la réunion. Dès qu'elle en prend connaissance, la secrétaire assure l'envoi de ces ordres du jour et de leurs éventuelles annexes sur base des listes de membres.

Occasionnellement des ajouts pourront intervenir ultérieurement, la secrétaire communiquera les nouveaux points aux membres par e-mail.

3.2.1.5 La gestion des procès verbaux

Le procès verbal est établi par le secrétaire de séance. Il est ensuite transmis à la secrétaire de direction qui en assure l'archivage sur PC et en version papier.

La secrétaire assure également l'envoi de ce PV aux différents membres. Cet envoi est groupé avec la convocation pour la réunion suivante.

3.2.1.6 Annonces diverses au personnel de l'Institut

Le secrétariat diffuse diverses informations, appelées « Communications du Directeur », en provenance d'organismes divers hors FUNDP et/ou du Rectorat des FUNDP (bourses de voyages, colloques, appels à projets, etc.) par courrier électronique.

3.2.2 Le secrétariat aux études

3.2.2.1 La diffusion de la fiche signalétique

La secrétaire utilise la fiche du dossier d'admission qui est archivée dans une farde accessible à toute personne susceptible d'en avoir besoin. La diffusion est assurée par photocopies.

3.2.2.2 La transmission des résultats de l'épreuve d'admission

Les étudiants ayant participé à un examen d'admission sont avertis par courrier du résultat de ce dernier.

Les étudiants LIHD sont de plus avertis du cycle dans lequel ils sont admis.

3.2.2.3 L'édition et la diffusion de la liste des cours par étudiant

Le programme des cours est préétabli pour chaque cycle, cependant en fonction du parcours académique préalable de chacun, des dispenses automatiques peuvent être accordées.

La secrétaire établit pour chaque étudiant la liste des cours qu'il devra suivre et la diffuse à qui de droit soit par mail, soit sur support papier.

3.2.2.4 La réservation de locaux et de matériel

Une application est déjà en service à cet effet.

3.2.3 Les secrétariats étudiants

3.2.3.1 La diffusion du guide de l'étudiant

Le guide de l'étudiant, disponible sur le site Web de l'Institut est remis à l'étudiant en version papier le jour de la rentrée.

3.2.3.2 L'édition, la mise à jour et la diffusion du trombinoscope

Le trombinoscope est un dossier comprenant les photos des étudiants par année d'étude, ainsi que leurs coordonnées privées et professionnelles.

Pour la LIHD, la secrétaire compose ce trombinoscope à partir des fiches signalétiques.

Pour les cours du jour, la secrétaire dispose d'une photo digitalisée pour chaque étudiant.

Ce dossier leur est remis dans les 15 jours de la rentrée.

3.2.3.3 La gestion de la problématique des syllabus (LIHD uniquement)

La secrétaire doit gérer cette problématique sous 2 aspects différents:

- La constitution d'une liste de syllabus personnalisée pour chacun
- Le paiement

En pratique, un tableau (étudiants / syllabus) est affiché pendant un temps prédéterminé aux valves du secrétariat, sur lequel chaque étudiant peut cocher les syllabus qu'il désire recevoir.

Au terme du temps prévu, la secrétaire récupère ce tableau et constitue manuellement les listes de syllabus pour chacun, et sur base de celle-ci, elle peut gérer le problème des paiements.

3.2.3.4 L'édition, la mise à jour et la diffusion des listes d'étudiants

Dès le début de l'année, la secrétaire établit pour chaque année, une liste des étudiants inscrits.

Cette liste est affichée aux valves du secrétariat.

La secrétaire tient cette liste à jour en fonction des éventuels abandons.

3.2.3.5 L'édition, la mise à jour et la diffusion des listes de cours par étudiants

Aux dispenses automatiques accordées lors de l'inscription, s'ajoutent des dispenses qui peuvent être accordées par les professeurs. Celles-ci doivent faire l'objet d'une demande écrite auprès de ces derniers.

En pratique, un formulaire de demande de dispense est complété par l'étudiant et transmis au professeur concerné, lequel examinera l'opportunité d'accorder ladite dispense (éventuellement après avoir consulté des documents faisant foi des aptitudes de l'étudiant) et accordera par écrit son accord.

Sur base des réponses fournies par les professeurs, la secrétaire met à jour la liste préalablement établie par le secrétariat aux études et la diffuse à qui de droit soit par mail, soit sur support papier.

3.2.3.6 L'édition et la transmission au CGI des listes d'étudiant

Sur base des listes d'étudiants, la secrétaire rédige une nouvelle liste sur laquelle les prénoms et noms de chaque étudiant sont remplacés par les initiales du prénom concaténées au nom avec un maximum de 8 caractères. La secrétaire doit veiller à l'unicité de chaque nom ainsi formé. Cette liste est transmise au CGI par mail.

3.2.3.7 La gestion des inscriptions aux cours à option

Pour chaque année d'étude présentant des cours à option, un formulaire est remis aux étudiants, reprenant l'ensemble des cours à option disponibles. Les étudiants doivent en cocher un nombre prédéfini.

Il est à noter que certains cours non proposés au départ peuvent être reconnus comme cours à option. L'étudiant qui souhaite suivre un tel cours doit en faire la demande et cette dernière doit faire l'objet d'une notification par le bureau exécutif.

Sur base de ces formulaires, la secrétaire met à jour la liste des cours par étudiants préalablement établie.

3.2.3.8 L'édition, la mise à jour et la diffusion des horaires

Les horaires sont établis manuellement.

Une application est déjà en service pour consultation.

3.2.3.9 La gestion des (in)disponibilités des professeurs pour les examens

Afin d'établir les horaires d'examens au mieux, chaque professeur est invité à compléter un tableau (prof / dates) en cochant les dates auxquelles il n'est pas disponible pendant la session d'examens.

Il est à noter que ce tableau est commun aux trois secrétariats étudiants.

3.2.3.10 L'édition, la mise à jour et la diffusion des horaires d'examen

Sur base des disponibilités des professeurs (i), l'horaire d'examen est établi et ensuite affiché aux valves du secrétariat et transmis par mail tant aux professeurs qu'aux étudiants.

Si une modification doit intervenir, un horaire modifié est réaffiché et renvoyé par mail aux intéressés.

3.2.3.11 La gestion des horaires de passage aux examens oraux

Pour chaque examen oral, un tableau (plages horaires / dates) est affiché aux valves du secrétariat. Les étudiants sont invités à s'inscrire au jour et à l'heure qui leur conviennent le mieux en fonction des disponibilités.

Après un laps temps prédéterminé, la secrétaire récupère ces tableaux d'inscription et rédige sur base de ceux-ci un horaire définitif lequel est à son tour affiché aux valves du secrétariat et envoyé par mail aux professeurs et aux étudiants.

En cours de jour, ils sont également envoyés aux délégués de cours.

3.2.3.12 La diffusion des cotes d'examen

Un tableau (étudiants / cours) est affiché aux valves du secrétariat avec les cotes obtenues aux examens par chaque étudiant (LIHD uniquement).

3.2.3.13 La gestion des inscriptions aux examens

La secrétaire transmet à chaque étudiant un formulaire d'inscription aux examens que celui-ci doit lui remettre signé.

Les formulaires sont collectés par la secrétaire qui en assure le suivi.

3.2.3.14 La diffusion des offres d'emploi

La secrétaire peut-être amenée à afficher des offres d'emploi aux valves du secrétariat.

3.2.3.15 La diffusion d'avis en tout genre

La secrétaire peut-être amenée à afficher de l'information d'ordre général aux valves du secrétariat.

3.2.3.16 La réservation de locaux et de matériel

Une application est déjà en service à cet effet.

3.3 Les besoins

3.3.1 Périmètre du projet

L'objet de ce paragraphe est de baliser le périmètre du projet, en mettant en évidence les fonctionnalités qui seront supportées, et celles qui ne le seront pas. Le résultat sera de dégager l'ensemble des développements qu'il y aura lieu de réaliser pour atteindre l'objectif du projet, à savoir le déploiement d'un site, mettant à la disposition des étudiants ainsi que du personnel de l'Institut un système de valves électroniques permettant l'échange d'informations entre les étudiants et le secrétariat étudiants. Ce site offrira une visibilité sur l'Internet au travers d'un portail gérant les accès.

La notion de portail, induit la notion de gestion des utilisateurs qui auront des droits au niveau des accès en fonction de leur appartenance à l'un ou l'autre groupe d'utilisateurs. Ainsi, les fonctionnalités décrites ci-après ne seront accessibles en tout ou en partie, qu'à une certaine catégorie d'utilisateurs

3.3.1.1 Fonctionnalités comprises dans le projet

3.3.1.1.1 La gestion des listes d'étudiants par année d'étude

Le système permettra à la secrétaire de consulter et de modifier (ajouter ou supprimer en fonction des nouveaux arrivant ou des abandons) la liste des étudiants inscrits dans chaque année d'étude.

Les étudiants pour leur part, n'auront accès à ces listes que pour consultation.

3.3.1.1.2 La gestion des inscriptions aux cours à option

L'inscription aux cours à option s'adresse aux étudiants uniquement. Ceux-ci devront pouvoir effectuer leur choix en ce qui concerne les cours à option qu'ils souhaitent suivre au cours de l'année académique en cours. Le système devra leur permettre de faire leur choix parmi une liste de cours disponibles ou le cas échéant de proposer un autre cours.

La secrétaire pour sa part devra pouvoir consulter la liste des cours à option qui sont proposés par les étudiants et valider ceux qui sont acceptés.

3.3.1.1.3 La gestion des dispenses

Cette fonctionnalité s'adresse à la secrétaire uniquement. Celle-ci devra pouvoir introduire dans le système les dispenses qui auront été accordées par les professeurs pour les étudiants qui bénéficient de telles dispenses.

3.3.1.1.4 La gestion des listes de cours par étudiant

Le système tiendra à jour la liste des cours pour chaque étudiant sur base de son année d'étude, de ses dispenses, de ses choix d'options, ...

Le système devra permettre à la secrétaire de prendre connaissance de la liste de cours que chaque étudiant doit suivre pour l'année académique en cours.

Les étudiants eux n'auront accès qu'à leur liste de cours personnelle.

3.3.1.1.5 La gestion de la problématique des syllabus

Le système devra mettre à la disposition des étudiants la liste ainsi que le prix des syllabus disponibles pour les cours auxquels ils sont inscrits. Le système devra leur permettre de faire un choix parmi cette liste en fonction des syllabus qu'ils souhaitent recevoir.

Le système permettra à la secrétaire de prendre connaissance du choix de chaque étudiant, ainsi que du montant dû par chacun.

3.3.1.2 Fonctionnalités exclues du projet

Toutes les autres fonctionnalités décrites au point 3.2 sont exclues du projet dans le cadre de ce travail. Elles pourront faire l'objet d'une étude ultérieure et constituer un upgrade du système qui sera mis en place.

3.3.2 Arborescence des buts

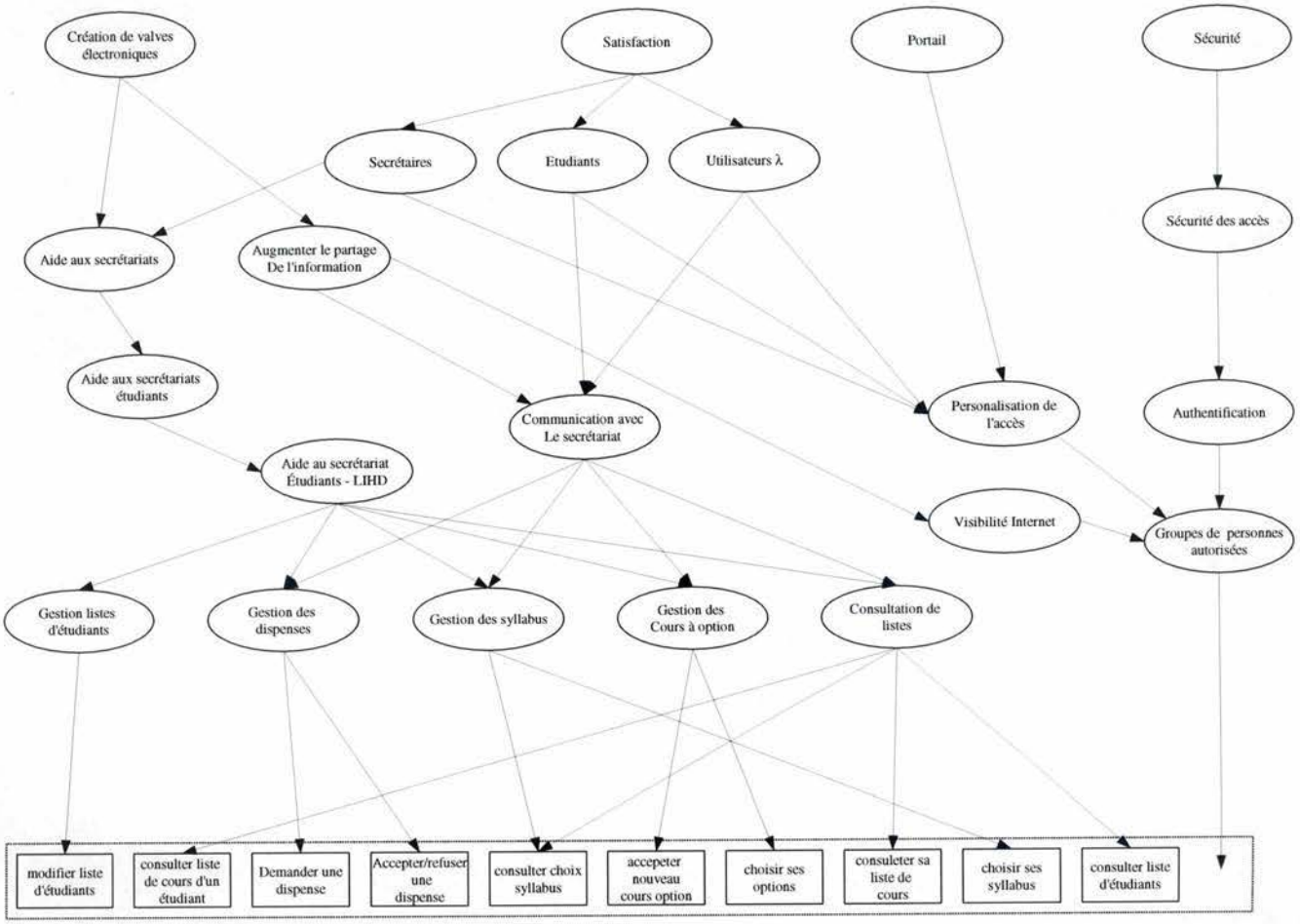


FIGURE 1 : ARBORESCENCE DES BUTS

3.4 Les exigences

3.4.1 Les exigences sur les acteurs

3.4.1.1 Les acteurs

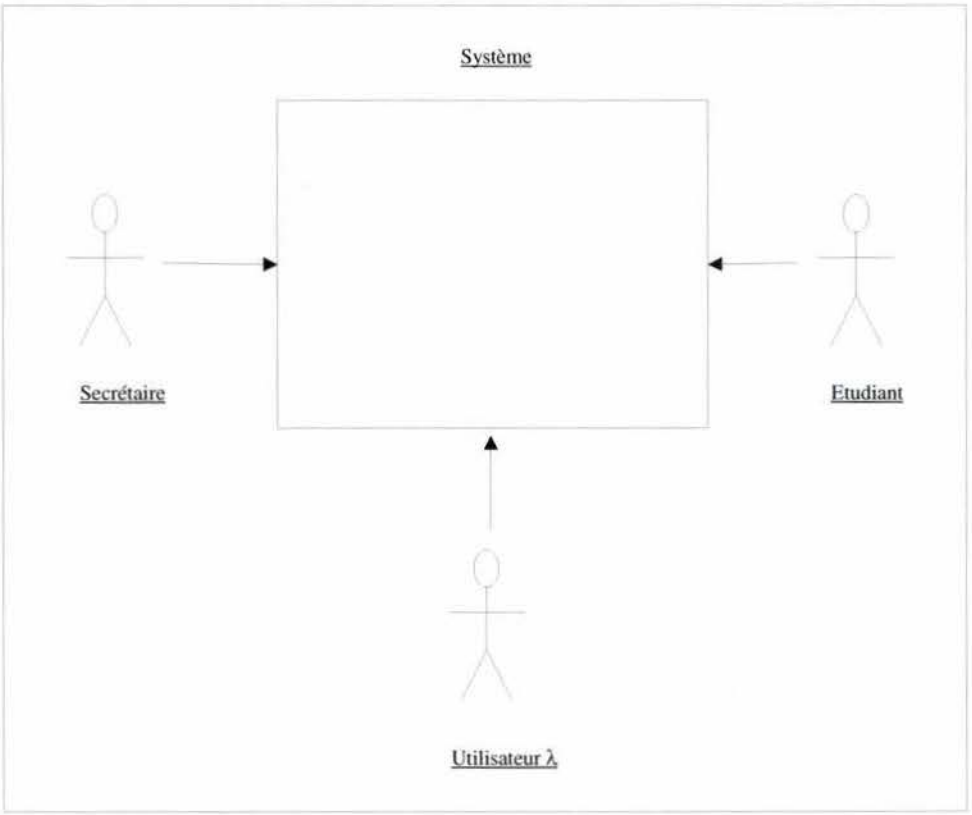


FIGURE 2 : DIAGRAMME DES ACTEURS

3.4.1.2 Les contraintes

3.4.1.2.1 Contraintes sur la secrétaire

- La secrétaire devra suivre la procédure de login afin d'accéder aux fonctionnalités auxquelles elle a accès
- La secrétaire devra signaler au système la fonctionnalité qu'elle souhaite utiliser.
- Dans le cas de la mise à jour d'une liste d'étudiants, la secrétaire devra :
    - Signaler au système l'étudiant qui doit être retiré de la liste dans le cas d'une suppression
    - Signaler au système les données de l'étudiant qui doit être ajouté dans le cas d'un ajout.
- Dans le cas de l'introduction d'une nouvelle dispense pour un étudiant, la secrétaire devra :
    - Signaler au système les données de l'étudiant
    - Signaler au système les coordonnées du cours concerné

- La secrétaire devra signaler au système les cours à option, proposés par les étudiants, qui sont acceptés comme tels

3.4.1.2.2 Contraintes sur les étudiants

- Les étudiants devront suivre la procédure de login afin d'accéder aux fonctionnalités auxquelles ils ont accès.
- Les étudiants devront signaler au système la fonctionnalité qu'ils souhaitent utiliser
- Dans le cas du choix des cours à option, l'étudiant devra signaler au système les cours qu'il souhaite suivre
- Si un cours non prévu est proposé, l'étudiant devra donner les coordonnées exactes de ce cours et fournira éventuellement une brève explication.
- Dans le cas du choix de syllabus, l'étudiant signalera au système quels sont les syllabus qu'il souhaite recevoir.

3.4.1.2.3 Contraintes sur les utilisateurs λ

Sont considérés comme utilisateur λ, tout utilisateur n'appartenant pas aux 2 catégories précédentes et ayant accès au système uniquement pour les fonctionnalités de consultation (professeur, assistant ou toute autre personne autorisée).

- Les utilisateurs λ devront suivre la procédure de login afin d'accéder aux fonctionnalités auxquelles ils ont accès.
- Les utilisateurs λ devront signaler au système la fonctionnalité qu'ils souhaitent utiliser

3.4.2 Les exigences sur le système

3.4.2.1 Les contraintes fonctionnelles

- Le système connaîtra une liste d'étudiants de base (avant d'éventuelles modifications par la secrétaire) pour chaque année d'étude
- Le système connaîtra la liste des cours obligatoires pour chaque année d'étude
- Le système connaîtra la liste de tous les cours à option disponibles pour chaque année d'étude
- Le système connaîtra la liste des syllabus disponibles
- Le système prendra connaissance des modifications qui seront apportées aux listes d'étudiants par la secrétaire
- Le système prendra connaissance des dispenses qui seront accordées par les professeurs
- Le système prendra connaissance du choix des cours à option pour chaque étudiant
- Le système pourra prendre connaissance de cours à option non connus du système et proposés par l'étudiant
- Le système prendra connaissance du choix des syllabus pour chaque étudiant.
- Le système sera en mesure de fournir la liste des étudiants inscrits dans une année d'étude déterminée à la demande de n'importe quel utilisateur
- Le système sera en mesure de fournir à la secrétaire la liste des cours pour n'importe quel étudiant
- Le système sera en mesure de fournir à la secrétaire la liste des dispenses par étudiant

- Le système sera en mesure de fournir à un étudiant sa propre liste de cours
- Le système sera en mesure de fournir à un étudiant le montant dû pour les syllabus qu'il a choisi
- Le système sera en mesure de fournir à la secrétaire la liste des syllabus choisis par un étudiant ; ainsi que le montant dû par ce dernier
- Le système devra disposer de la possibilité de gérer les accès sur base de groupes d'utilisateurs. Chaque groupe ayant des droits d'accès différent au niveau des fonctionnalités disponibles.

3.4.2.2 Les contraintes non fonctionnelles

- Le système devra s'intégrer à l'infrastructure existante à l'Institut, tant en terme de matériel, que de système d'exploitation ou de logiciels déjà en place :
    - Le système sera déployé sur un serveur « *SUN* »
    - Le système d'exploitation sera « *Sun Solaris 8* »(UNIX)
    - Le portail utilisé sera « *I-Planet portal Server 3.0* »
    - Le serveur d'application utilisé sera « *I-Planet* » ou « *BEA Weblogic* »
    - La DB utilisée pour servir de repository sera une DB « *Oracle 9i* »ou « *Interbase 5.5* »
- Le contrôle d'accès se fera sur base des personnes autorisées au niveau de l'OS.
- Un back up de la base de donnée sera mis en place selon les procédures prévues par le SGBD en place.
- La solution proposée sera de type « thin client ». L'application sera accessible à partir d'un navigateur Internet.
- L'application sera développée en Java J2EE, ce qui garanti la stabilité de l'application par rapport à l'OS.
- Le look and feel du site s'intégrera dans la logique du site officiel de l'Institut. Une version anglaise pourra être disponible
- Le site sera accessible 24H / 24H
- En cas de panne, le site devra être de nouveau accessible dans les 6 heures

4 ANALYSE FONCTIONNELLE

4.1 La modélisation

Quelque que soit la complexité d'un système, et compte tenu du fait que l'aptitude humaine à la compréhension est limitée, il y a lieu d'aborder tout problème en divisant le problème initial en une série de sous problèmes de complexité moins grande et donc plus facilement compréhensibles. [Khawar Zaman Ahmed & Cary E. Umrysh, 2001]

La modélisation permet à cet égard de fournir une vue simplifiée et facilement compréhensible des différents aspects d'un système en cernant successivement chacun d'entre eux dans un formalisme standard et explicite.

L'importance de la modélisation est mise en évidence par les 4 objectifs suivants :

- L'aide à la visualisation d'un système tel qu'il est ou tel que l'on veut qu'il soit
- La spécification de la structure ou du comportement du système
- La production d'un modèle aidant la construction du système
- La production d'un document justifiant les choix effectués

Nous nous basons dans ce travail sur la modélisation UML (Unified Modeling Language). [Grady Booch, James Rumbaugh & Ivar Jacobson, 2000]

Dans le cadre de l'analyse fonctionnelle, nous produirons les diagrammes suivants :

- Diagramme des uses case
- Uses case
- Schemas de robustesse

4.2 Les uses cases

Les uses case sont représentés par le diagramme des uses case et par les uses case proprement dits. Il s'agit d'un des moyens mis à notre disposition par UML pour modéliser les aspects dynamiques du système.

D'une manière générale, nous dirons que chaque use case correspond à une fonctionnalité du système.

Les uses case servent aussi bien à spécifier les interactions entre les utilisateurs et le système dans les différents scénarios, que de support pour les différents tests à effectuer.

4.2.1 Diagramme des uses cases

Le diagramme des uses case montre :

- L'ensemble des uses case
- Les acteurs
- Les interactions entre ceux-ci

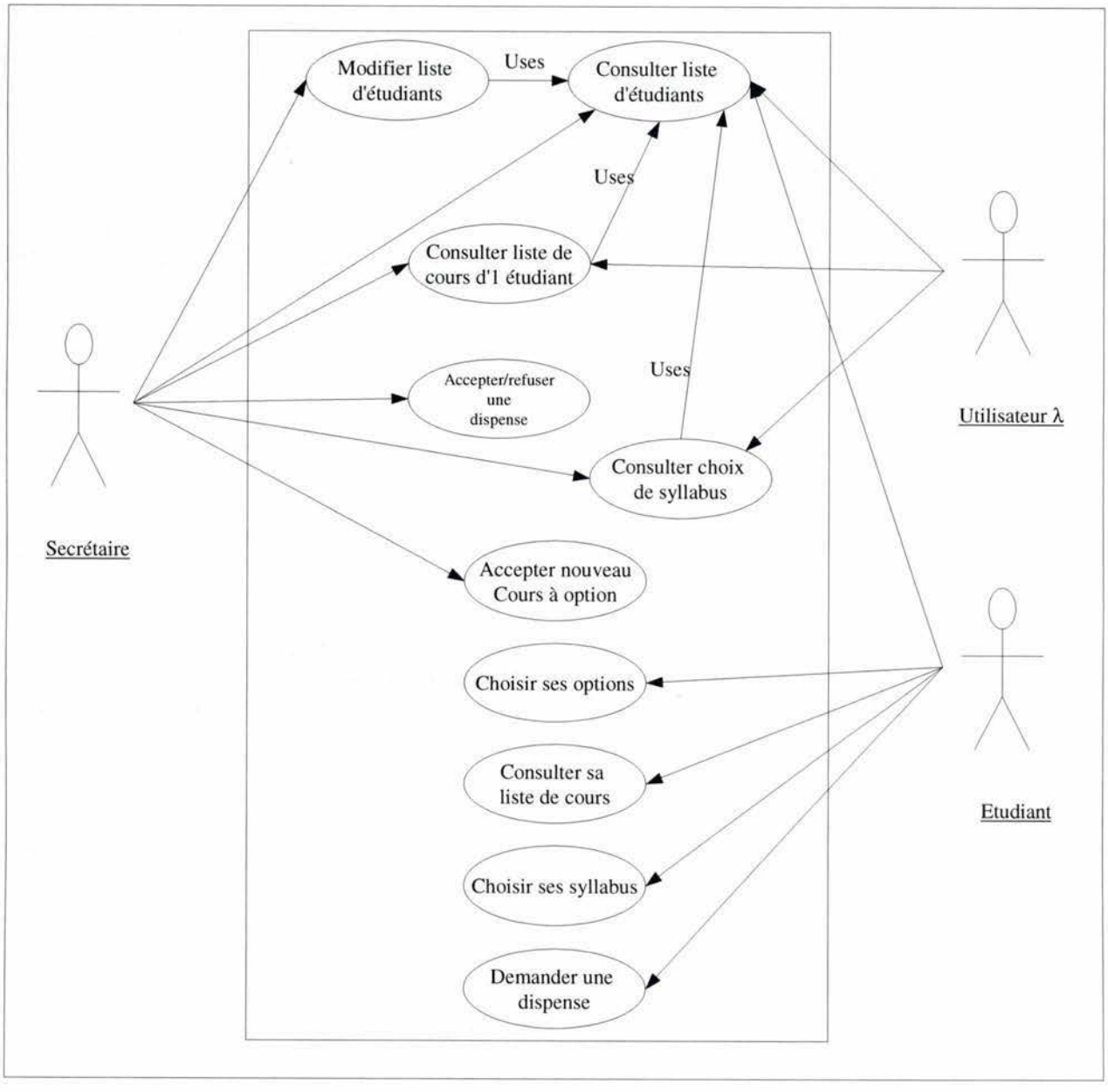


FIGURE 3 : DIAGRAMME DES USES CASES

4.2.2 Uses cases

Pour tous les uses cases qui suivent, on considérera que l'utilisateur a préalablement effectué le choix de la fonctionnalité à laquelle il souhaite accéder dans un menu personnalisé en fonction de son appartenance à l'un ou l'autre groupe.

4.2.2.1 Consulter liste d'étudiants

Objet:	permet à tout utilisateur de consulter une liste d'étudiants pour une année d'étude donnée
Préconditions:	le système a pris connaissance de tous les étudiants inscrits
PostConditions:	l'état du système n'est pas modifié

Secrétaire / Etudiant / λ	**Système**
	1. le système invite l'utilisateur à lui indiquer l'année d'étude pour laquelle il souhaite connaître la liste des étudiants
2. L'utilisateur fourni ce renseignement au système	
	3. le système fourni la liste des étudiants actuellement inscrits dans l'année spécifiée au point 2

Fin du UC

4.2.2.2 Modifier liste d'étudiants

Objet:	permet à la secrétaire de modifier une liste d'étudiants soit en ajoutant, soit en retirant m étudiant(s) de la liste
Préconditions:	le système a pris connaissance de tous les étudiants inscrits. La liste concernée compte n étudiants
PostConditions:	$n' = n \pm m$

Secrétaire	**Système**
	1. le système invite la secrétaire à lui indiquer s'il souhaite procéder à un retrait ou à un ajout d'étudiant
2. la secrétaire fourni cette indication au système	
	3. [use case *"consulter liste d'étudiants"*]
4. la secrétaire choisi dans la liste l'(les) étudiant(s) à supprimer	
	5. le système supprime cet (ces) étudiant(s) de la liste

Fin du UC

Flux d'exception 1: *"ajouter un (des) étudiant(s)"*

Secrétaire	**Système**
	3'. Le système invite la secrétaire à lui indiquer l'année et les données d'un étudiant à ajouter
4'. La secrétaire fourni ces données au système	
	5'. Le système ajoute cet étudiant à la liste de son année et offre la possibilité de recomencer en 3' ou de terminer
6' La secrétaire fait son choix	

Fin du UC

Flux d'exception 1.1: *"le choix est: continuer"*

Secrétaire	**Système**
	7". Reprise du flux d'exception 1 au point 3'

4.2.2.3 Consulter liste de cours d'1 étudiant

Objet:	permet à la secrétaire ou à l'utilisateur lambda de consulter la liste de cours d'un étudiant déterminé
Préconditions:	le système a pris connaissance de l'année d'étude de l'étudiant, des cours obligatoires pour cette année d'études, du choix des cours à option de l'étudiant, des dispenses de cet étudiant
PostConditions:	l'état du système n'est pas modifié

Secrétaire / λ	**Système**
	1. [use case *"consulter liste d'étudiants"*]
2. l'utilisateur choisi dans la liste l'étudiant dont elle veut connaître la liste des cours	
	3. le système fourni la liste des cours pour cet étudiant. Le système fourni aussi de l'information concernant le statut actuel en ce qui concerne le choix de cours à option et l'introduction de dispenses. Enfin , le système offre la possibilité de recommencer en 1 ou de terminer
4. L'utilisateur fait son choix	

Fin du UC

Flux d'exception 1: *"le choix est: continuer"*

Etudiant	**Système**
	5'. Reprise du UC au point 1

4.2.2.4 Consulter sa liste de cours

Objet: permet à un étudiant de consulter sa propre liste de cours
Préconditions: le système a pris connaissance de l'année d'étude de l'étudiant, des cours obligatoires pour cette année d'études, du choix des cours à option de l'étudiant, des dispenses de cet étudiant

PostConditions: l'état du système n'est pas modifié

Etudiant	**Système**
	1. le système fourni la liste des cours pour l'étudiant concerné

Fin du UC

4.2.2.5 Choisir ses options

Objet: permet à un étudiant de choisir ses m cours à option parmi une liste proposée et éventuellement d'en proposer un supplémentaire
Préconditions: le système a pris connaissance des cours à option disponibles pour l'année d'étude de l'étudiant
PostConditions: le système a pris connaissance du choix de l'étudiant pour les cours à option

Etudiant	**Système**
	1. le système fourni à l'étudiant la liste des cours à option disponibles et la possibilité d'en proposer un, non repris dans la liste
2. l'étudiant choisi ses cours dans la liste	
	3. le système vérifie le nombre de cours choisis et prend connaissance du choix de l'étudiant

Fin du UC

Flux d'exception 1: *"le nombre de cours choisis est différent de m"*

Secrétaire	**Système**
2'. L'étudiant choisi un nombre de cours différent de m	
	3'. Le système vérifie le nombre de cours choisis et signale à l'étudiant que le compte n'est pas juste. Reprise du UC au point 1

Flux d'exception 2: *"le nombre de cours choisis est m-1 et un cours supplémentaire est proposé"*

Secrétaire	**Système**
2". L'étudiant choisi m-1 cours et en propose 1	
	3". Le systèmevérifie le nombre de cours choisis , prends connaissance du choix de l'étudiant et signale à ce dernier que le cours proposé est en cours de validation

Fin du UC

Flux d'exception 3: *"le nombre de cours choisis est différent de m-1 et un cours supplémentaire est proposé"*

Secrétaire	**Système**
2'". L'étudiant choisi un nombre de cours, différent de m-1 et en propose un	
	3'". Le système vérifie le nombre de cours choisis et signale à l'étudiant que le compte n'est pas juste. Reprise du UC au point 1

4.2.2.6 Accepter nouveau cours à option

Objet:	permet à la secrétaire de valider m cours à option proposé par un étudiant
Préconditions:	le système connaît n cours à option qui ne sont pas proposés au départ
PostConditions:	n' = n + m

Secrétaire	Système
	1. le système fourni à la secrétaire la liste des élèves pour lesquels un cours à option à été proposé, ainsi que les coordonnées de ce cours
2. la secrétaire choisi le(s) cours qui est (sont) accepté(s) comme cours à option par les autorités académique	
	3. le système prends connaissance de ce(s) cours comme étant un cours à option supplémentaire et l'ajoute à la liste de cours de l'étudiant concerné

4.2.2.7 Accepter/refuser une dispense

Objet: permet à la secrétaire d'accepter ou de refuser une (plusieurs) dispense(s) pour un étudiant déterminé
Préconditions: le système connaît la liste des cours pour cet étudiant
PostConditions: la liste des cours pour cet étudiant ne comprend plus le(s) cours pour lequel la (les) dispense(s) vien(nen)t d'être acceptée(s)

Secrétaire	Système
	1. le système fourni à la secrétaire la liste des étudiants pour lesquels une dispense a été demandée
2. la secrétaire choisi l'étudiant concerné dans la liste	
	3. le système fourni à la secrétaire la liste des dispenses demandées pour cet étudiant
4. la secrétaire choisi la (les) dispense(s) concernée(s) dans la liste et signale au système si celle-ci est acceptée ou refusée	
	5. le système retire le(s) cours concerné(s) de la liste de cours de l'étudiant

Fin du UC

Flux d'exception 1: *"le choix est de refuser la dispense "*

Secrétaire	Système
	5'. Le système ne tient plus compte de cette demande de dispense

Fin du UC

4.2.2.8 Demander une dispense

Objet:	permet à un étudiant de demander une dispense pour un ou plusieurs cours
Préconditions:	le système connaît la liste des cours pour cet étudiant
PostConditions:	le système connaît cet étudiant comme étant en attente d'une réponse pour sa demande de dispense

Etudiant	**Système**
	1. [use case *"Consulter sa liste de cours"]*
2. l'étudiant choisi le(s) cours concerné(s) dans la liste	
	3. le système prend connaissance du choix de l'étudiant et marque celui-ci comme ayant une demande de dispense en cours d'acceptation

Fin du UC

4.2.2.9 Choisir ses syllabus

Objet:	permet à un étudiant de choisir ses syllabus et d'en connaître le prix
Préconditions:	le système connaît la liste des syllabus disponible et le prix de ceux-ci
PostConditions:	le système a pris connaissance du choix des syllabus pour l'étudiant concerné, ainsi que le montant total dû par cet étudiant

Etudiant	**Système**
	1. le système fourni à l'étudiant la liste des syllabus disponibles pour son année d'étude, ainsi que le prix de chacun d'entre-eux
2. l'étudiant choisi ses syllabus	
	3. le système prend connaissance du choix de l'étudiant et fourni à l'étudiant le montant global de sa commande

Fin du UC

4.2.2.10 Consulter choix de syllabus

Objet:	permet à la secrétaire de consulter le choix des étudiants pour leurs syllabus ainsi que le montant dû pour chacun
Préconditions:	le système a pris connaissance du choix des étudiants en matière de syllabus
PostConditions:	l'état du système n'est pas modifié

Secrétaire / λ	**Système**
	1. [use case *"consulter liste d'étudiants"*]
2. l'utilisateur choisi l'(les) étudiant(s) pour le(s)quel(s) il souhaite connaître la commande	
	3. le système fourni à la secrétaire les commandes et le montant global pour chacune d'entre elles pour l'(les) étudiant(s) choisi(s)

Fin du UC

4.3 Schéma de Robustesse

Les schémas de robustesse permettent de modéliser de manière statique les principaux éléments du système en mettant en évidence leur rôle fonctionnel.

D'une manière générale, le système peut être illustré par un schéma de robustesse unique.

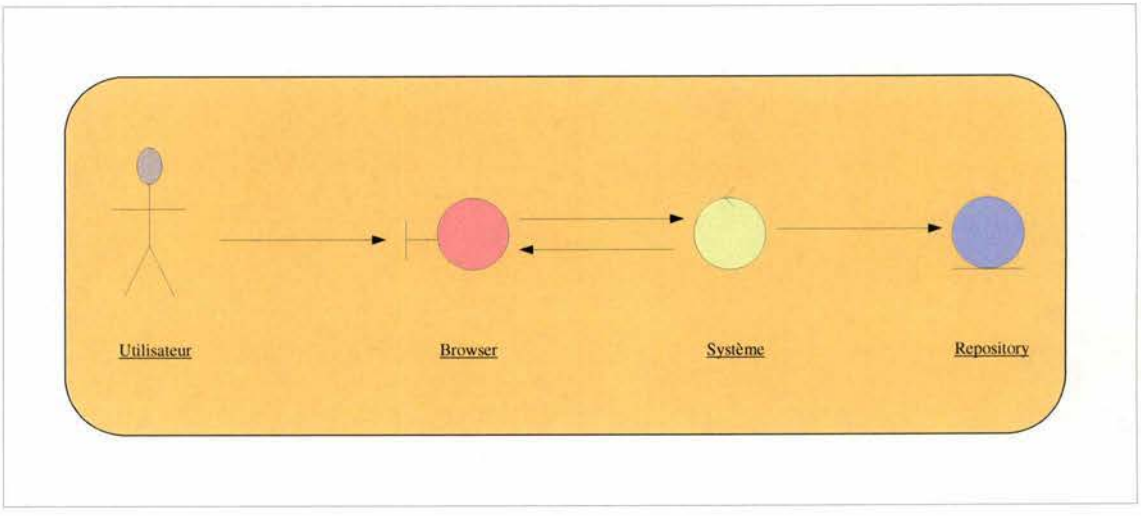


FIGURE 4 : SCHEMA DE ROBUSTESSE GLOBAL

Dans tous les cas, un utilisateur connecté sur le site au moyen d'un *"Browser"* Internet accède à de l'information contenue dans le *"Repository"* , via le *"Système"* qui assure le traitement et la présentation des données.

D'une manière plus détaillée, à chaque fonctionnalité (choisir ses syllabus, encoder une dispense, etc ...), correspond une succession de vues (de pages) correspondant chacune à l'instanciation du modèle souhaité en fonction de la requête transmise par la vue précédente.

Le schéma de robustesse correspondant à l'ensemble du système peut donc être représenté comme suit:

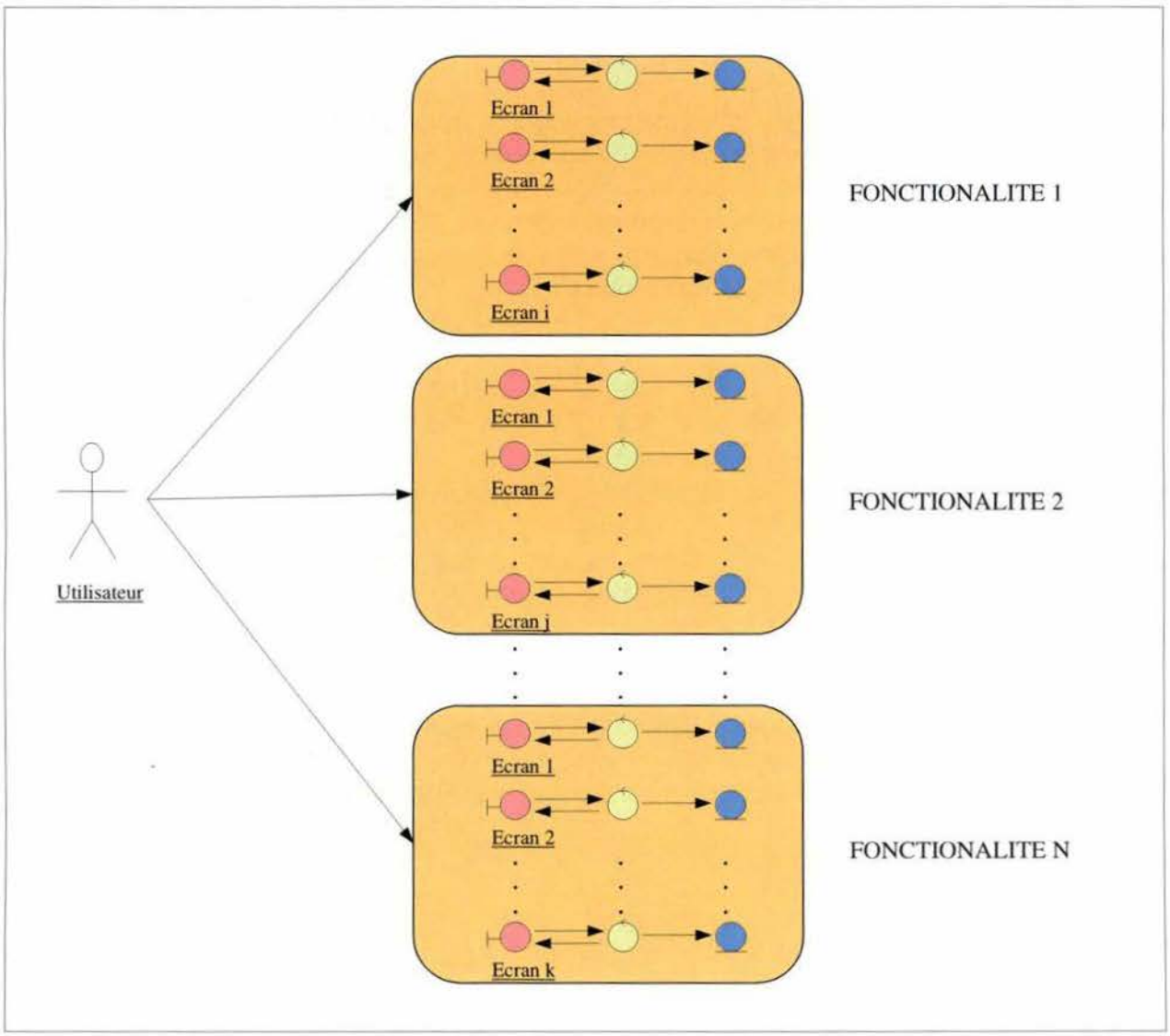


FIGURE 5 : SCHEMA DE ROBUSTESSE DETAIL

Les composants *"Système"* et *"Repository"* seront présentés plus en détail ultérieurement.

5 CONCEPTION DE LA DB

Cette partie est consacrée à la description approfondie du composant *"Repository"* décrit au point 4.3.

Comme nous l'avons vu précédemment, la persistance du système est assurée par une base de données. Nous avons choisi ici de concevoir une petite base de données toute simple, reprenant les données nécessaires au fonctionnement du système, sans prévoir d'intelligence ni de complexité au niveau de cette DB. Cette dernière étant développée spécifiquement pour notre application, nous gérerons les différents aspects de cohérence au niveau applicatif.

5.1 validation

L' enseignement à l'Institut d'Informatique est réparti sur plusieurs années d'études différentes.
Chaque année a un numéro d'identification unique et un nom.
Un étudiant est inscrit dans une ou plusieurs années d'étude et possède un numéro d'identification unique, un nom, un prénom et un login.
Les cours dispensés sont soit obligatoires, soit à option et possèdent un numéro d'identification unique, un intitulé et un titulaire.
A une année d'étude, correspond une liste de cours obligatoires et pour certaines années une liste de cours à option.
Un étudiant suit tous les cours obligatoires pour l'année dans laquelle il est inscrit, et si des cours à option sont prévus pour cette année, il en choisi m (selon l'année dans laquelle il est inscrit) parmi les cours à option disponibles.
Un étudiant inscrit dans une année à laquelle correspondent des cours à option à également la possibilité de n'en choisir que m-1 parmi les cours à option disponibles et de proposer le choix d'un cours d'une autre faculté. Si ce dernier cours est accepté par les autorités académiques, il constituera alors le m ^{ième} cours à option de cet étudiant et fera partie des cours à option disponible pour les autres étudiants de la même année.
Chaque cours peut être illustré par un ou plusieurs syllabus ayant chacun un titre, un prix et un auteur.
Chaque étudiant peut choisir les syllabus qu'il désire acheter.
Enfin, un étudiant peut demander une dispense pour un cours obligatoire. Cette dernière sera acceptée ou refusée. Si elle est acceptée, l'étudiant concerné ne suivra pas ce cours.

5.2 Schéma conceptuel

Du texte de validation énoncé au point 5.1, nous dérivons le schéma conceptuel suivant :

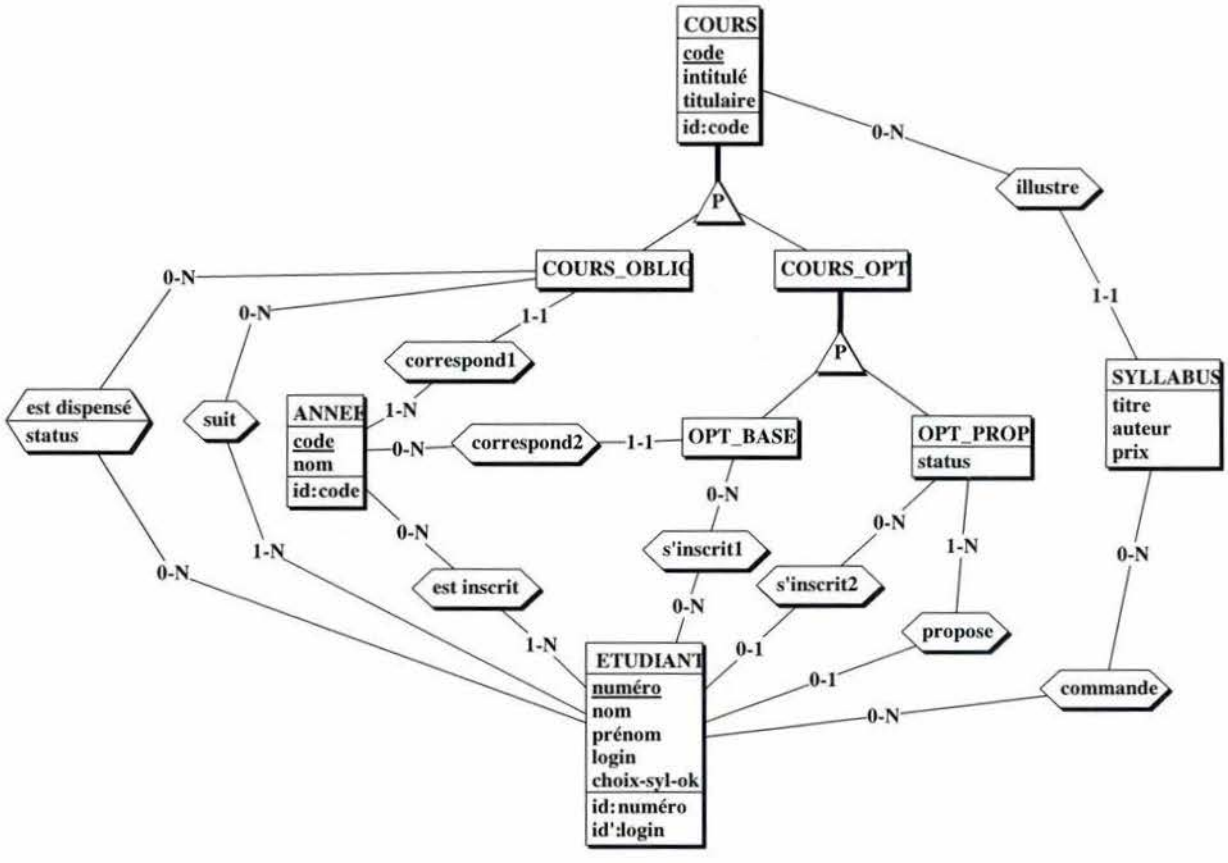


FIGURE 6 : DB - SCHEMA CONCEPTUEL

5.3 Transformations

Les transformations suivantes ont été appliquées au schéma conceptuel produit au point 5.2 afin d'en déduire un schéma relationnel :

- Transformation des relations ISA
- Transformation des types d'associations complexes
- Transformation des types d'associations 1-N et 1-1
- Ajout d'identifiants techniques
- Transformation des types d'associations résiduels

5.4 Schéma relationnel

Après transformations listées au point 5.3, le schéma conceptuel produit le schéma relationnel suivant :

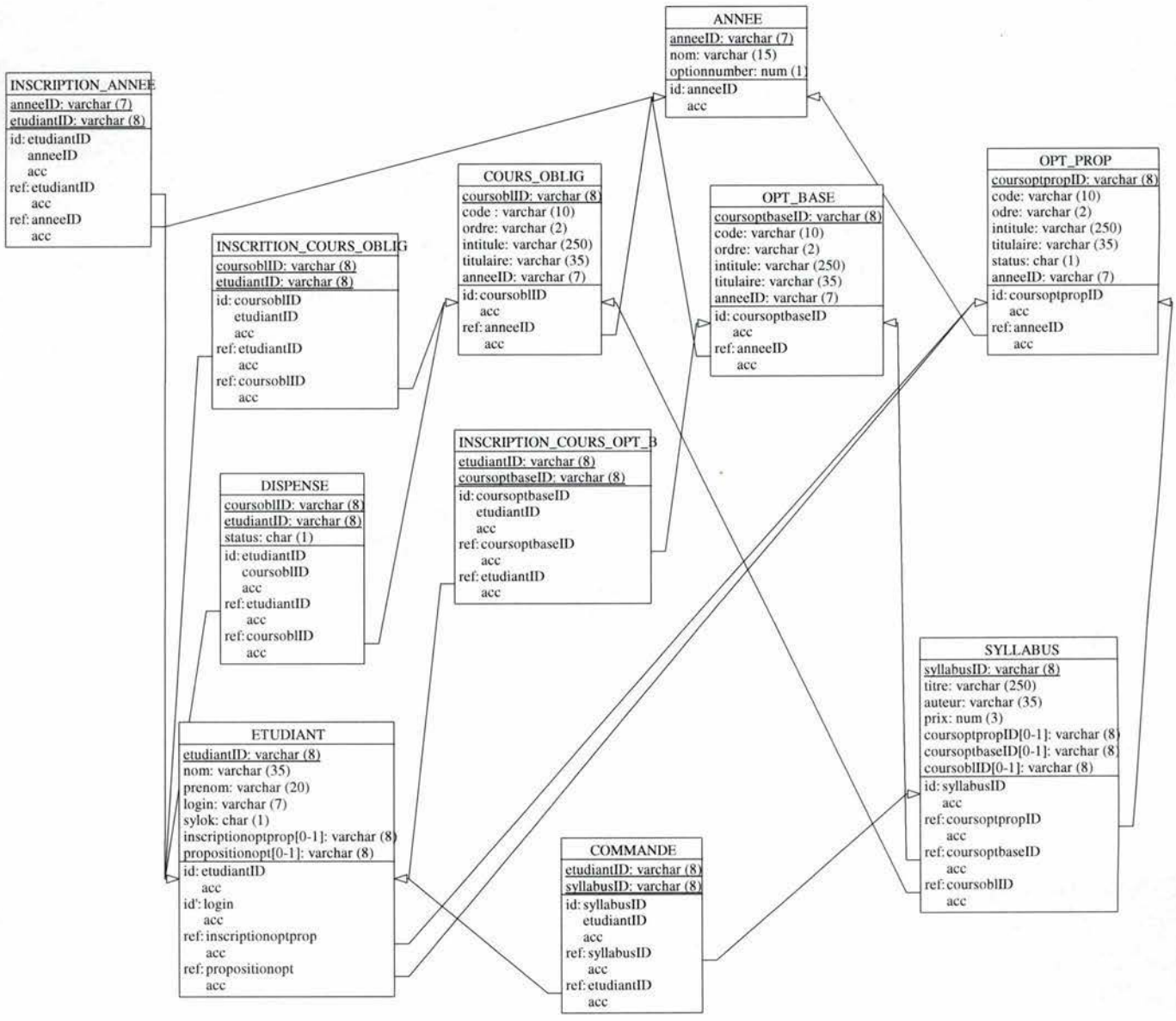


FIGURE 7 : DB - SCHEMA RELATIONNEL

5.5 Génération des scripts

Sur base du schéma relationnel produit au point 5.4, et avec l'aide du logiciel DB-Main, les différents scripts de création de la DB ont été générés.

Ces scripts sont disponibles sur le CD-Rom fourni en annexe.

6 ASPECTS THEORIQUES SUR LES ARCHITECTURES MULTI NIVEAUX ET LA PLATE FORME J2EE

[Subrahmanyam allamaraju & Cie , 2001]
[Paco Gomez & Peter Zadrozny, 2000]

6.1 Les besoins des entreprises

Les besoins des entreprises sont fortement influencés ces dernières années par l'émergence du nouveau créneau commercial apparu avec le développement de l'Internet et du commerce électronique.
La plupart des principes commerciaux de ces entreprises reposent maintenant sur une économie de l'information nécessitant la mise en œuvre de moyens techniques permettant non seulement le partage de cette information avec leurs différents partenaires commerciaux, mais aussi et surtout l'accès à des sources d'information issues de technologies héritées du passé.

Parallèlement à cette augmentation des besoins, le contexte économique est tel que le temps et les budgets consacrés à ces adaptations sont considérablement réduits.

Un dernier paramètre à prendre en compte est la perpétuelle évolution des moyens mis en œuvre, et la nécessité qui en découle d'avoir une grande faculté d'adaptation.

Ainsi, nous résumerons les besoins actuels des entreprises de la manière suivante :

- **Capacité de réaction**: essentielle en terme de concurrence afin de rester compétitif.
- **Productivité élevée en programmation**: la qualité de la mise en œuvre est bien entendu importante, en terme de :
    - Maîtrise des nouvelles technologies
    - Intégration avec d'autres technologies
    - Développements efficaces et rapides
    - Aptitude à maintenir un niveau de compétence élevé et à l'acquisition de nouvelles compétences
- **Fiabilité et disponibilité** : il est important de garantir la fiabilité des transactions commerciales (exécution entière et correcte), mais aussi la disponibilité du système. Un site Internet à vocation commerciale se doit d'être disponible 24H/24H.
- **Sécurité** : tant le nombre potentiel d'utilisateurs, que la valeur sans cesse croissante des informations échangées nécessitent des garanties d'authentification, de confidentialité, ...
- **Evolutivité** : une application actuelle, potentiellement accessible par des milliers d'utilisateurs, doit pouvoir faire face à une montée en charge rapide pour s'adapter aux nécessités et au succès commercial de l'entreprise. Elle doit également être en mesure d'utiliser de manière optimale les ressources du système.
- **Intégration** : l'optimisation de l'utilisation de l'information contenue dans les systèmes hérités du passé, est une clef fondamentale du succès d'un nouveau développement.

6.2 Les architectures multi-niveaux

6.2.1 Architecture à 2 niveaux

L'architecture à 2 niveaux, encore appelée architecture 2-tiers, est une architecture client-serveur classique assurant la séparation entre :

- Les données
- La logique de présentation et la logique applicative

Dans ce type d'architecture, l'application réside entièrement sur la machine cliente, tandis que les données sont hébergées sur un serveur centralisé.

Le schéma suivant illustre cette architecture :

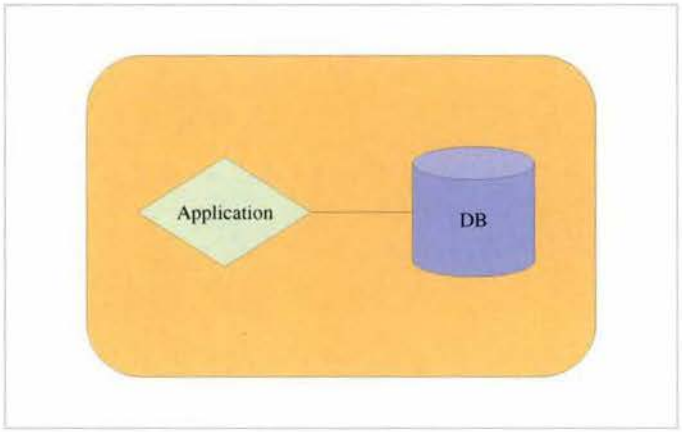


FIGURE 8 : ARCHITECTURE A 2 NIVEAUX

Ce type d'architecture permet le partage de l'information au sein d'une entreprise, mais présente néanmoins un certain nombre d'inconvénients :

- **Performances limitées** : du fait que l'application réside entièrement sur le PC client, les performances du système sont influencées par celle de la machine
- **Augmentation de la charge réseau** : les données étant entièrement traitées côté client, le nombre de requêtes à la base de données est élevé, engendrant une grande consommation de bande passante.
- **Difficultés de maintenance** : la moindre modification sur l'application devra être répétée sur tous les postes clients. Par ailleurs, dans une organisation conséquente, certains départements pourraient ne pas être prêts à investir dans une réorganisation du SI et donc cela pourrait conduire à la cohabitation de plusieurs versions de l'application, augmentant d'autant le problème de maintenance.

6.2.2 Architecture à 3 niveaux

La réponse aux problèmes posés par l'architecture à 2 niveaux est fournie par l'architecture à 3 niveaux, encore appelée architecture 3-tiers.

Dans une telle architecture, on sépare la présentation du traitement des données.

Ainsi, le niveau présentation offre une interface utilisateur graphique, le niveau intermédiaire est un niveau métier, comprenant la logique applicative, et enfin un dernier niveau contient les données nécessaires à l'application.

Une telle architecture est illustrée dans le schéma suivant :

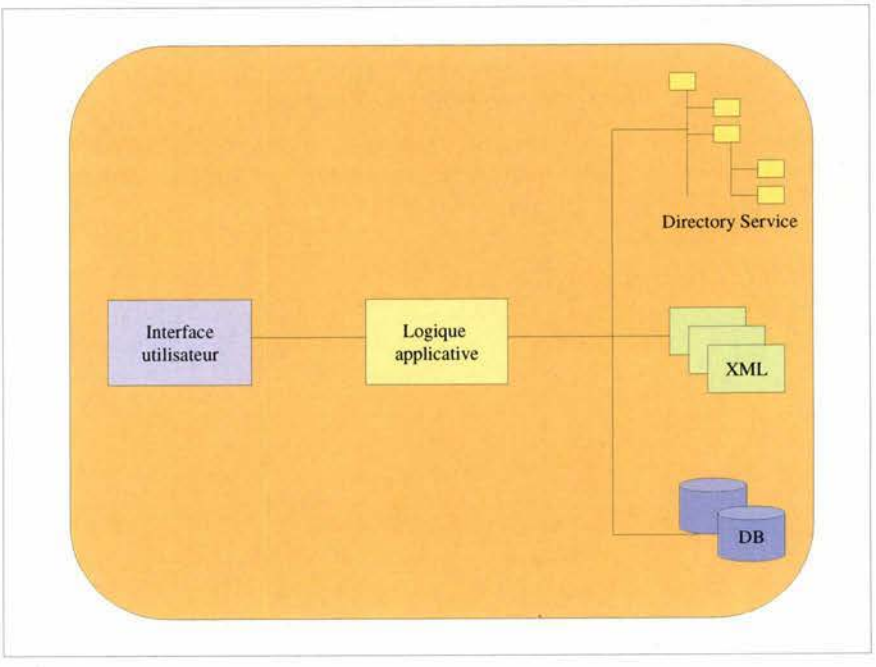


FIGURE 9 : ARCHITECTURE A 3 NIVEAUX

Dans ce modèle, chaque niveau logique représenté est pourvu d'un ensemble d'interfaces spécifiques leur permettant de communiquer entre eux.

Le rôle de chaque niveau est défini comme suit :

- **Interface utilisateur** :
  - Appels à la logique applicative
  - Réception des résultats et présentation de ceux-ci
- **Logique applicative** :
  - Traitement de données
- **Niveau données** : ce niveau contient les données nécessaires, sur n'importe quel support :
  - DB relationnelle classique
  - Documents XML
  - Service d'annuaire
  - .....

dans une telle architecture, le déploiement de nombreuses interfaces utilisateurs peut se faire sans modifier la logique applicative , pour autant que celle-ci définisse clairement une interface avec le niveau présentation.

6.2.3 Architecture multi-niveaux

L'architecture multi-niveaux, ou encore n-tiers, constitue une évolution de l'architecture à 3 niveaux. Il existe de multiples configurations d'une telle architecture, mais c'est essentiellement la couche applicative qui bénéficie d'évolutions selon sa fonction spécifique.

Les principaux modules rencontrés dans une telle architecture sont :

- **L'interface utilisateur** : simple navigateur ou application plus sophistiquée
- **La logique de présentation** : prend en charge :
  - Ce que doit afficher l'interface
  - La manière de traiter les requêtes utilisateur
- **La logique métier** : modélise les règles métiers et traite les données
- **Les services d'infrastructure** : fournissent des services supplémentaires aux composants de l'application (service de messagerie, support transactionnel, ...)
- **Le niveau de données** : renferme les données accédées par le système.

Le schéma suivant illustre ces différents modules :

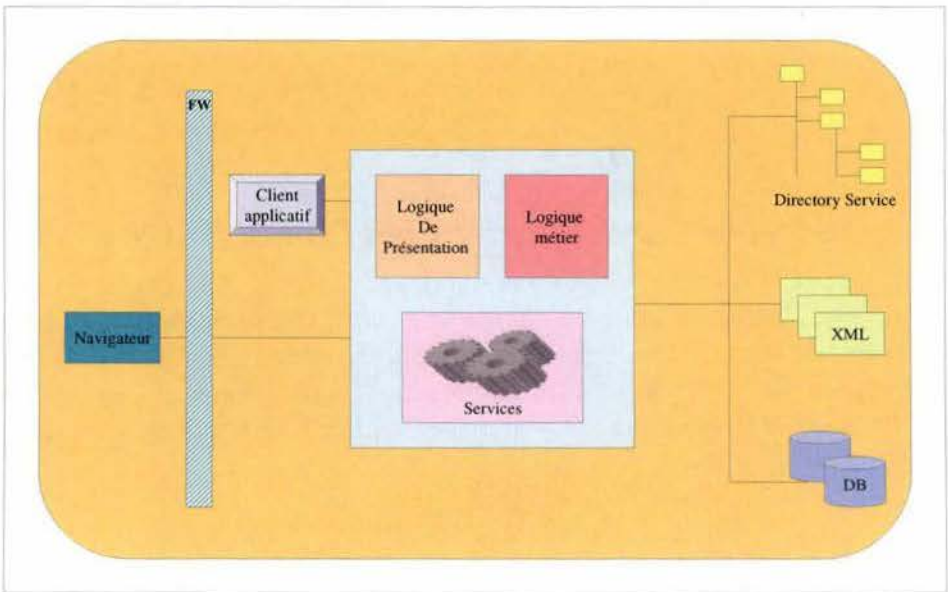


FIGURE 10 : ARCHITECTURE MULTI-NIVEAUX

Une telle architecture est encore appelée architecture Modèle / Vue / Controleur (MVC).

6.3 Architecture d'entreprise

Une architecture d'entreprise étant faite de collaboration entre applications différentes, il suffit d'imaginer que la couche intermédiaire est étendue à plusieurs applications dotées d'une interface leur permettant de communiquer entre elles.

Un middleware de base de données ou autre peut également être ajouté entre la couche intermédiaire et la couche de données.

Une telle architecture est illustrée ci-après :

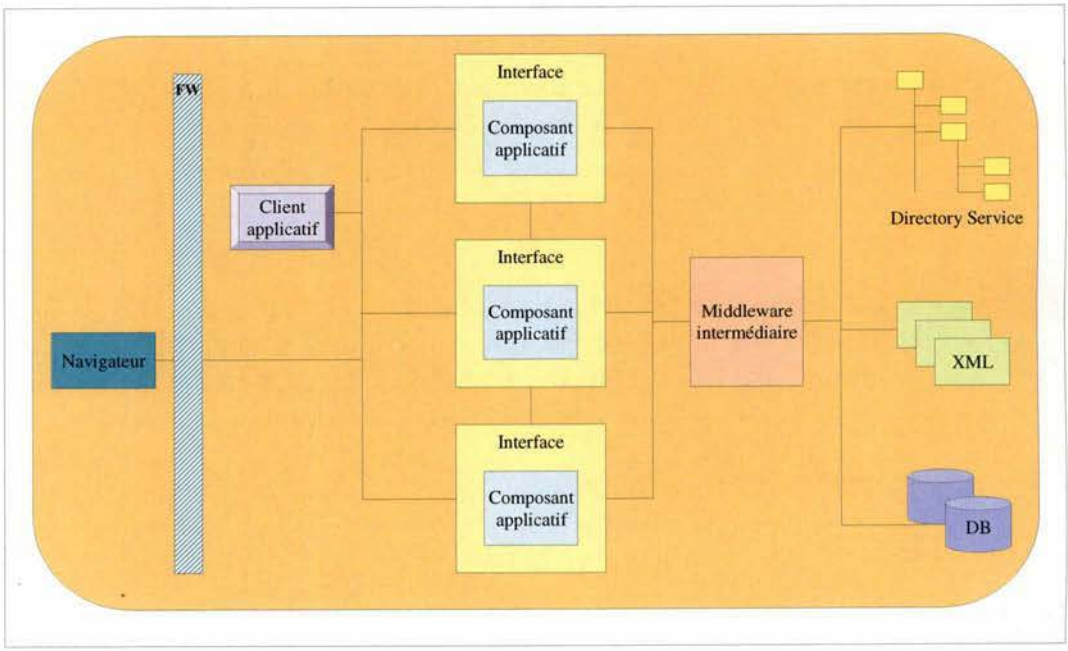


FIGURE 11 : ARCHITECTURE D'ENTREPRISE

6.4 La plate-forme "Java 2 Enterprise Edition" (J2EE)

Les besoins des entreprises précédemment exprimés, sont couverts par un ensemble de technologies solutionnant chacune un ou plusieurs problèmes présentés plus haut, cependant il serait intéressant de bénéficier d'une solution globale, offrant une plate-forme pourvue d'une infrastructure complète et de possibilités architecturales permettant une mise en œuvre fiable et rapide ; **la plate-forme « Java 2 Enterprise Edition »,** développée par SUN, **présente ces caractéristiques**.

6.4.1 Généralités

La plate-forme J2EE, basée comme son nom l'indique sur le langage OO Java, présente les avantages suivants :

- **Indépendance de la plate-forme** : comme chacun le sait, Java est un langage portable, son code étant exécuté par une machine virtuelle spécifique à chaque plate-forme. Il s'agit ici d'un avantage important dans le contexte d'une entreprise

présentant des environnements hétéroclites, et permet l'économie de mécanismes de translations compliqués et peu efficaces.

- **Réutilisabilité** : un objectif commun à tout développeur est la réutilisation de code. Cet objectif est atteint par :
  - La séparation des exigences métier, propres à une application, en composants distincts
  - L'utilisation de l'OO pour encapsuler les fonctionnalités partagées

- **Modularité** : les différents composants de J2EE, (servlets,JavaServerPages [JSP], Enterprise JavaBeans[EJB]) permettent de modulariser l'application en la découpant en niveaux et en tâches, chaque module exécutant une action spécifique.

6.4.2 La plate-forme J2EE

La plate-forme J2EE est un environnement serveur d'applications distribuées, capable de présenter aussi bien des modules de gestion de pages Web, que des module de gestion de transactions de bases de données. Pour ce faire, J2EE offre deux grandes catégories d'outils :

- Un environnement d'exécution hébergeant les applications
- Un ensemble d'API Java pour la conception de ces applications

6.4.2.1 L'environnement d'exécution de J2EE

J2EE présente la caractéristique de faire abstraction de l'infrastructure d'exécution.

J2EE :

- NE SPECIFIE PAS la façon dont un environnement d'exécution devrait être implémenté
- SPECIFIE les rôles et les interfaces pour les applications
- SPECIFIE l'environnement d'exécution dans lequel les applications doivent être déployées.

J2EE permet au développeur de se concentrer sur la programmation de la logique métier, en le déchargeant des tâches de développement propres à l'environnement d'exécution (pool de connexion, connectivité aux bases de données, gestion de transactions, ...) . C'est l'implémentation du serveur d'application fournie par les différents vendeurs qui assure cet aspect des choses.

J2EE introduit la notion de conteneur d'application

J2EE élabore un « contrat » entre le conteneur et les applications, et rempli ce contrat via ses différentes API.

6.4.2.2 Les API J2EE

Les services distribués auxquels les applications distribuées doivent classiquement faire appel, sont :

- Le traitement des transactions
- L'accès aux bases de données
- La messagerie asynchrone
- L'appel synchrone de méthodes
- ....

J2EE standardise ces API et permet aux applications Java d'y accéder par l'intermédiaire du conteneur.

Typiquement, un serveur d'applications **J2EE** comprend un ou plusieurs conteneurs et fourni l'implémentation des API suivantes :

- **Java DataBase Connectivity 2.0 (JDBC)**: accès aux bases de données, pooling de connexions, transactions distribuées, ...
- **Remote Method Invocation over the Internet Inter-ORB 1.0 (RMI-IIOP)** : cet API permet:
  - La mise en oeuvre de l'API RMI classique de Java
  - La connectivité entre applications RMI et CORBA
- **Enterprise Java Beans 1.1 (EJB)** : composants Java pour infrastructure multi niveaux, bénéficiant d'un vaste infrastructure d'exécution et d'hébergement côté serveur
- **Java Servlets 2.2** : couche d'abstraction OO pour la conception d'applications Web
- **Java ServerPages 1.1 (JSP)** : permet la construction de pages Web dynamiques
- **Java Message Service 1.0 (JMS)** : API d'accès à un service de messagerie asynchrone. Permet notamment :
  - Queuing
  - Publish subscribe
- **Java Naming and Directory Interface 1.2 (JNDI)**: permet l'accès aux différents service de nommage et d'annuaires disponibles
- **Java Transaction Architecture 1.0 (JTA)** : permet la mise en œuvre d'applications distribuées transactionnelles
- **JavaMail 1.1** : permet la conception d'applications de courrier électronique basées sur Java.

6.4.3 L'architecture J2EE

Un serveur d'application J2EE comprend un ou plusieurs conteneurs, fournissant :

- Un environnement d'exécution aux composants applicatifs qu'il contient
- Un accès aux API J2EE

Une architecture J2EE classique est représentée dans le schéma ci-dessous :

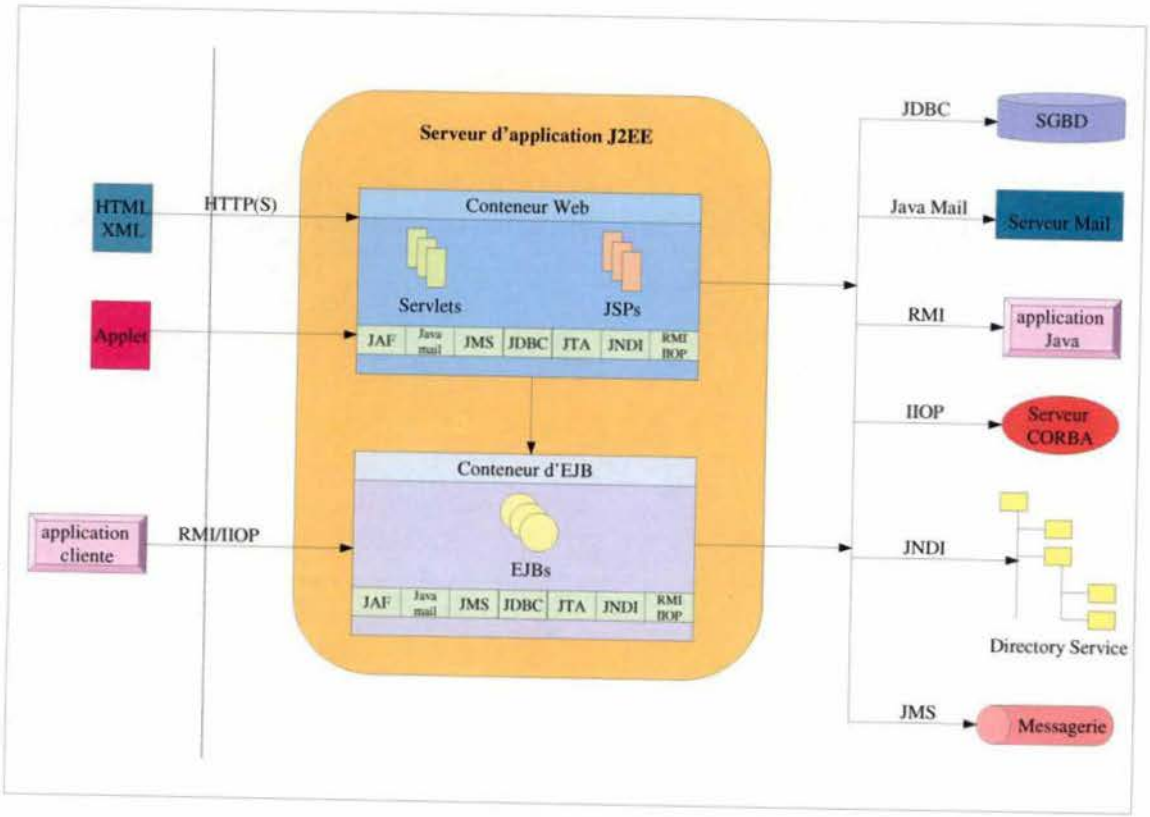


FIGURE 12 : ARCHITECTURE J2EE

Ce schéma représente :

- Les différents types de clients :
  - Navigateur exécutant une page HTML ou une applet
  - Application cliente
- Les protocoles de communication des clients
- Les 2 conteneurs principaux :
  - Le conteneur Web (servlets, JSPs)
  - Le conteneur d'EJBs
- Les API J2EE disponibles pour les 2 conteneurs
- Les API spécifiques (servlets, JSPs, EJBs)
- Les clients du conteneur Web :
  - Navigateur
- Les clients du conteneur d'EJB :
  - Les applications clientes

- Les composants actifs du conteneur Web

- Les différentes sources de données accessibles via les différentes API

Ajoutons enfin pour être complet, que la plateforme J2EE présente en plus des conteneurs déjà décrits, 2 conteneurs supplémentaires qui sont :

- **Un conteneur d'applet,** qui permet d'exécuter des applets Java
- **Un conteneur d'applications clientes,** qui permet d'exécuter des applications Java classiques côté client.

6.4.4 les conteneurs

l'architecture d'un conteneur (Web ou EJB) est la suivante :

- les éléments fournis par le développeur :
  - les composants applicatifs
  - les descripteurs de déploiement
- les éléments structurels du conteneur :
  - le contrat des composants
  - les API des services du conteneur
  - les services déclaratifs
  - les autres services du conteneur

6.4.4.1 Les composants applicatifs

Les composants applicatifs comprennent les servlets, les JSPs et les EJBs

6.4.4.2 Les descripteurs de déploiement

Un descripteur de déploiement est un fichier XML associé à un composant applicatif, ses rôles sont :

- La description du composant applicatif auquel il est associé
- Fournir au conteneur des informations sur la manière de gérer efficacement le composant applicatif auquel il est associé.

6.4.4.3 Le contrat des composants

Le contrat des composants est un ensemble d'API spécifiées par le conteneur que les composants applicatifs qu'il contient sont tenus d'étendre ou de mettre en œuvre. Par exemple, une servlet devra étendre le classe *javax.servlet.http.HttpServlet* et mettre en œuvre les méthodes *doGet() et doPost()* de cette classe, tandis qu'un EJB devra étendre les interfaces *javax.ejb.EJBHome* et *javax.ejb.EJBObject* pour ses interfaces et implémenter l'interface *javax.ejb .SessionBean* ou *javax.ejb.EntityBean* pour le bean proprement dit.

L'architecture globale d'un conteneur est illustrée par le schéma suivant :

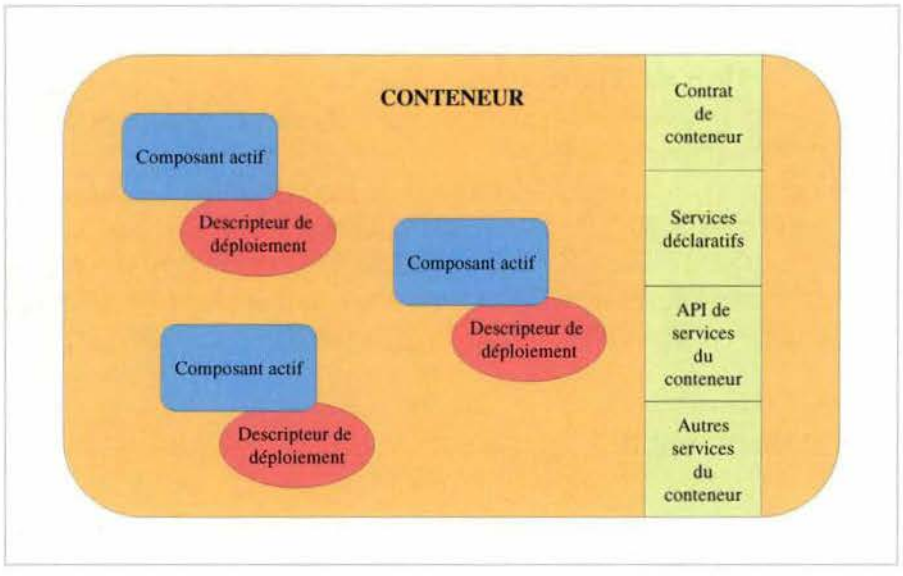


FIGURE 13 : ARCHITECTURE DE CONTENEUR

6.4.4.4 Les API des services des conteneurs

Les différentes API déjà décrites plus haut permettent aux composants applicatifs contenus dans les conteneurs de communiquer d'une manière uniforme avec les autres composants du système. Rappelons que ces API, permettent d'accéder à un service indépendamment de son implémentation.

6.4.4.5 Les Services déclaratifs

L'architecture J2EE permet de configurer ses différents composants de manière dynamique, via les **descripteurs de déploiement**. Ce dernier permet de définir le contrat passé entre le conteneur et le composant

L'information contenue dans les descripteurs de déploiement varie en fonction du type de composant auquel il est associé, mais d'une manière générale, sont configurés dans ces descripteurs des aspects tels que :

- La sécurité
- Le comportement transactionnel
- ....

6.4.4.6 Les autres services des conteneurs

En plus de tous les aspects déjà évoqués, le conteneur offre des services supplémentaires tels que :

- **La gestion du cycle de vie des composants** : incluant la création et la destruction des instances en fonction des besoins

- **Le pooling des ressources** : prise en charge du pooling d'objets et du pooling de connexions
- **Le peuplement de l'espace de noms JNDI sur base des noms de déploiement associés aux EJB**
- **Le peuplement de l'espace de noms JNDI avec les objets nécessaires à l'utilisation des API des services des conteneurs** : il s'agit ici des objets d'accès aux sources de données, des objets *ConnectionFactory* pour l'obtention de connexions JMS et des objets de contrôle transactionnel.
- **Le clustering** : les conteneurs J2EE sont distribuables, c'est à dire, que les composants applicatifs peuvent être répartis sur plusieurs machines virtuelles et cela sur plusieurs machines hôtes. Selon la stratégie de « load balancing » adoptée, la charge sera répartie sur les différents conteneurs concernés. Le clustering est un facteur essentiel de l'extensibilité et de la disponibilité des applications J2EE.

6.4.5 Les technologies J2EE

Nous l'avons vu, les systèmes distribués reposant sur une architecture J2EE nécessitent la mise en œuvre de nombreuses technologies pour assurer leur conception.
Ces technologies peuvent se classifier de la manière suivante :

- Les technologies de composants
- Les technologies de services
- Les technologie de communication

Il s'agit ici d'une classification des différentes API déjà présentée, mais cette classification permet de bien mettre en valeur la nature de chacune d'entre-elle.

6.4.5.1 Les technologies de composants

Les technologies de composants renferment la logique métier, il s'agit des servlets, des JSPs et des EJBs.
Comme nous avons vu qu'il existe 2 types de conteneurs et que chacun d'entre eux renferme des composants spécifiques, nous dirons que les composants se répartissent en :

- **Composants Web** : qui sont d'une manière générale des composants capables de recevoir une requête http, de la traiter et de renvoyer une réponse vers le client.

  - Les servlets sont des programmes côté serveur qui permettent d'associer de la logique applicative au processus request/response, et d'injecter de manière dynamique de l'information dans des documents HTML ou XML.
  - Les JSPs sont une extension du modèle de programmation des servlets, mais elle contiennent du code HTML et du code Java. Lors de sa première exécution, une page JSP est compilée par le conteneur et transformée en servlet. Cette servlet est dès lors appelable par le serveur Web qui renverra le résultat (une page HTML) au navigateur client. Lors des exécutions suivantes, la phase de compilation n'est plus nécessaire.
    Les pages JSP n'étant compilées qu'à l'exécution, et contenant une partie de code Java et une partie de code HTML, elles permettent la séparation de la logique applicative et la logique de présentation.

- **Composants EJB** : la technologie EJB est un modèle de composants distribués :

  - Sûrs
  - Evolutifs
  - Transactionnels
  - Multiutilisateurs

Les EJB permettent de séparer la logique applicative des services de niveau système. Il existe 2 types de composants EJB :

- **Les beans de session**

  - ✓ Les beans de session « statefull » sont des objets transitoires conservant un état entre les différentes requêtes d'un client. La durée de vie d'un bean de session statefull est la durée d'une session client.
  - ✓ Les bean de session « stateless » sont des objets transitoires qui ne conservent pas d'état entre les requêtes du client.

- **Les beans entité** : sont des objets persistants, qui modélisent les objets contenus dans la source de donnée en les encapsulant dans un objet. A l'inverse des beans de session, les beans entité peuvent être partagés par plusieurs clients. L'état persistant du bean peut être géré :

  - ✓ Par le conteneur
  - ✓ Par le développeur

6.4.5.2 Les technologies de services

Les technologies de service fournissent aux composants des services connexes leur permettant de fonctionner en toute efficacité.

Citons les API JDBC, JNDI et JTA déjà décrites précédemment.

6.4.5.3 Les technologies de communication

Les technologies de communication mettent en œuvre les mécanismes de communication entre les différentes parties de l'application tant locales que distantes. On distingue 2 types de protocoles de communication :

- Les protocoles Internet

  - **HTTP**: pour Hypertext Transfert Protocol. Il s'agit d'un protocole fonctionnant sur un modèle request/response. Le client envoie au serveur une requête comprenant :

    - ✓ Une méthode de requête
    - ✓ Une URI
    - ✓ Une version de protocole
    - ✓ Un message de type MIME

Le serveur renvoie à son tour :

- ✓ Une ligne d'état
- ✓ Un message de type MIME contenant la réponse du serveur
- ✓ Les méta-informations de l'entité
- ✓ Le corps de l'entité

- **TCP/IP** : pour Transmission Control Protocol. Il s'agit en fait de l'association de deux protocoles qui permettent l'acheminement des données sur Internet. [Andrew Tanenbaum, 1999]

  - ✓ IP garanti que les données sont acheminées entièrement et convenablement vers leur destinataire
  - ✓ TCP garanti que les données séparées en paquets empruntant des chemins différents sont bien reconstituées avant d'être délivrées à leur destinataire.

- **SSL** : pour Secure Socket Layer. Protocole de sécurité permettant:

  - ✓ L'authentification des intervenants
  - ✓ L'utilisation de la cryptographie pour chiffrer les informations

- Les protocoles appliqués aux objets distants

  - **RMI et RMI/IIOP** : pour Remote Method Invocation. RMI permet l'utilisation d'interfaces pour définir des objets distants. Ces interfaces permettent d'invoquer les méthodes de ces objets distants comme si il s'agissait d'objets locaux.
    RMI-IIOP est une extension de RMI sur IIOP (Inter ORB Protocol) permettant la définition d'une interface avec n'importe quel objet distant dans n'importe quel langage pour autant que cet objet supporte le mapping OMG et ORB
  - **JavaIDL** : permet d'invoquer des appels de méthode sur un objet CORBA (Common Object Request Broker Architecture).cet objet peut être développé dans n'importe quel langage pour autant que celui-ci définisse une interface IDL (Interface Definition Langage) .
  - **JMS** : pour Java Message Sending. Il s'agit d'un middleware orienté messages (MOM) permettant l'envoi et la réception de message de manière asynchrone.
  - **JavaMail** : présente une alternative de messagerie asynchrone, mais plus orientée utilisateurs.

6.4.5.4 Le XML

Le langage Java fourni l'API JAXP d'analyse grammaticale XML (eXtensible Markup Language). Cette API n'est pas (pas encore ...) reprise dans la spécification J2EE.

Cependant, il y a lieu de signaler le rôle non négligeable joué par XML dans les applications J2EE.

Notons au passage l'utilisation faite de XML :

- J2EE permet de fournir des services durant l'exécution au moyen d'un mécanisme déclaratif défini dans un descripteur de déploiement ; ce descripteur de déploiement est un fichier XML
- XML fourni un moyen d'intégration des applications J2EE dans les systèmes hérités en fournissant un support de communication indépendant des plate-formes et des technologies.
- XML peut être utilisé en lieu et place de HTML pour présenter des données applicatives au client.

6.4.6 Les produits existants

6.4.6.1 Les plate-formes concurrentes

Enfin, pour être complet, signalons que J2EE n'occupe pas une place unique sur le marché, en effet, le concept est également repris par d'autres vendeurs. Citons ici et de manière non exhaustive :

- Microsoft DNA (.net)
- Oracle 8i Internet platform
- ...

6.4.6.2 les serveurs d'application J2EE

citons également de manière non exhaustive :

- serveurs d'applications J2EE
  - BEA Weblogic
  - IBM Webphere
  - SunONE Application Server
  - ...
- Conteneurs de servlet
  - Jakarta Tomcat
  - ....
- Conteneurs d'EJB
  - Jboss
  - ....

7 ASPECTS THEORIQUES SUR LE CONCEPT DE PORTAIL

7.1 Internet

7.1.1 Généralités

Internet, e-mail, world wide web, commerce électronique, voilà toute une série de termes désormais devenus familiers, alors qu'il y a une quinzaine d'années, ils étaient réservés à une poignée d'initiés.

Mais que signifie en réalité ces termes du langage courant ?

Sans entrer dans la complexité technique, nous pouvons dire qu'Internet est un ensemble de réseaux interconnectés, utilisant le protocole TCP/IP. Les réseaux ainsi interconnectés, proviennent de tous les horizons possibles et imaginables de par le monde :

- Universités
- Administration
- Entreprises
- Centres de recherche
- ...

Un nombre sans cesse croissant d'ordinateurs sont ainsi interconnectés. Certains de ces ordinateurs offrent un service sur le réseau, accessible aux autres ordinateurs, on les appelle alors des « host server »

Les différents services disponibles sur Internet sont :

- **e-mail** qui permet à tout utilisateur d'Internet d'envoyer ou de recevoir des messages de manière individuelle, avec la possibilité d'associer toutes de sortes de fichiers au message texte initial.
- **Newsgroup,** basé sur l'e-mail, mais permettant l'échange de messages à l'échelle publique, accessible à un nombre illimité d'utilisateurs et regroupés en plusieurs milliers de thèmes différents
- **world wide web ,** permettant de publier des documents contenant entre autre des éléments graphiques, du texte et des hyperliens permettant la navigation d'un document à un autre
- **transfert de fichiers** , qui grâce au protocole FTP permet d'accéder à des milliers de serveurs proposant un nombre quasi illimité de fichier en libre service.
- **IRC** ou Internet Relay Chat, permettant à un nombre illimité d'utilisateurs de dialoguer en temps réel
- **Téléconférence** permettant l'échange d'images et de sons, cette technologie peut s'appliquer aussi bien à la téléphonie, qu'à une utilisation privée ou industrielle de la vidéo conférence
- **E-commerce et e- business**, basés sur l'émergence des techniques de cryptographie et du langage XML comme support de transaction, ces services permettent l'utilisation d'Internet pour tous types de transactions.

7.1.2 Historique

Nous nous contenterons de rappeler ici les grandes dates qui ont jalonné l'évolution d'Internet depuis la naissance des premiers concepts jusqu'à nos jours :

[Eric Larcher, 1998]
[Les Freed & Frank J. Derfler Jr, 1994]

- **Début de années 60** : concepts techniques fondamentaux
- **1969** : projet ARPANET (Advanced Research Projects Agency Network) mis sur pied par le DoD (department of Defense) dont le but est de mettre sur pied un réseau efficace, capable de fonctionner même en cas de panne de certains de ses éléments
- **1972** : présentation d'ARPANET au grand public et mise au point du courrier électronique. Parallèlement, d'autres réseaux sont développés par la NASA (organisme gouvernemental) et par NSF (National Science Foundation, organisme académique et industriel)
- **1973** : bases du protocole TCP/IP et apparition des adresses IP
- **1976** : apparition d'un système permettant de donner des noms aux machines afin de les identifier plus aisément
- **1980** : TCP/IP est adopté comme standard par le ministère de la défense américain
- **1983** : migration d'ARPANET vers TCP/IP et scission d'ARPANET en 2 parties :
  - la partie militaire (MILNET)
  - la partie consacrée à la recherche (ARPANET)
- **1985-86** : la NSF adopte TCP/IP pour son propre réseau NFSNET qui attire de plus en plus d'utilisateurs, conduisant en moins de dix ans au démantèlement d'ARPANET
- **1990** : naissance de l'Internet d'aujourd'hui.
- **1991** : ouverture de l'Internet, jusque là limité au courrier électronique, à l'échange de fichiers et aux groupes de discussions, aux activités commerciales
- **1992** : naissance du World Wide Web, sous l'impulsion du CERN (Centre Européen de Recherches Nucléaires)
- **de nos jours** : aux premières applications se sont ajoutés des services de communication en temps réel, les entreprises exploitent de plus en plus le potentiel commercial de l'Internet, ...

7.1.3 Perspectives

L'évolution de l'Internet est loin d'être terminée, d'autant plus que les possibilités semblent quasi illimitées, les évolutions étant livrées à l'imagination galopante de nos chercheurs et industriels. [Dominique Liard, 2000]

Il semble que l'évolution soit fortement liée aux possibilités en matière de lignes à haut débit.

Ainsi, à terme devrait-on évoluer vers la mise en place d'un réseau permettant le transport de grandes quantités de données à très haute vitesse.

Une telle infrastructure devrait permettre de faire converger tous les modes de communication sur un seul réseau et faire bénéficier entre autres :

- La télévision
- La radio
- Le téléphone
- Le Web
- La vidéoconférence
- ...

Tous ces services étant en principe accessibles avec un seul appareil, en l'occurrence, ... un micro-ordinateur.

7.2 Portail, définition, classification

7.2.1 Définition

« *Un portail est un point d'entrée unique, permettant de fédérer et de diffuser des informations à des communautés d'utilisateurs* » [Bernard Truffaut, 2001]

plus concrètement,

- Un portail est un site Web relatif à une communauté, renfermant un certain nombre de données relatives à différents sujets et incluant l'accès à :
  - Des nouvelles
  - Des informations financières
  - Des applications
  - Des services
  - ...
- Un portail fourni des possibilités de personnalisation, en terme de :
  - présentation du contenu
  - nature du contenu initial proposé
- un portail d'entreprise est un site Web personnalisé, rassemblant les données utiles de cette entreprise pour ses collaborateurs, ces derniers pouvant être :
  - des employés
  - des vendeurs
  - des partenaires commerciaux
  - des clients
  - des agents de marketing
  - ...

  Un tel portail d'entreprises doit fournir un environnement e-business sécurisé permettant des transactions complètes telles que la vente aux clients, l'achat aux fournisseurs, ...
- Enfin, un portail fourni un point d'accès sécurisé en matière de
  - Sécurité des données par encryptage (SSL)

- Authentification des utilisateurs

7.2.2 3 générations de portails

Les premiers portails sont apparus en 1998, et depuis, leur évolution a été fulgurante, pour en être aujourd'hui à la troisième génération.

[Gene Phifer, 2001a]
[Gene Phifer, 2001b]

- **La première génération (1998)** se caractérise essentiellement par :
  - L'agrégation de contenu Web
  - Des structures de navigation
  - Des fonctionnalités de recherche
  - La personnalisation
- **La deuxième génération (2000)** se distingue de la première par l'apparition
  - Du concept de portlet permettant l'accès à des applications
  - De l'accès à une couche d'abstraction supplémentaire, le serveur d'application
- Enfin, **la troisième génération (2002)** , appellée « portal application server » regroupe le portail et le serveur d'application en un seul produit, ainsi en plus des fonctionnalités de regroupement, d'accès à des applications et des fonctionnalités de personnalisation, un portail devient un véritable fournisseur de contenu.

Le schéma suivant illustre cette évolution :

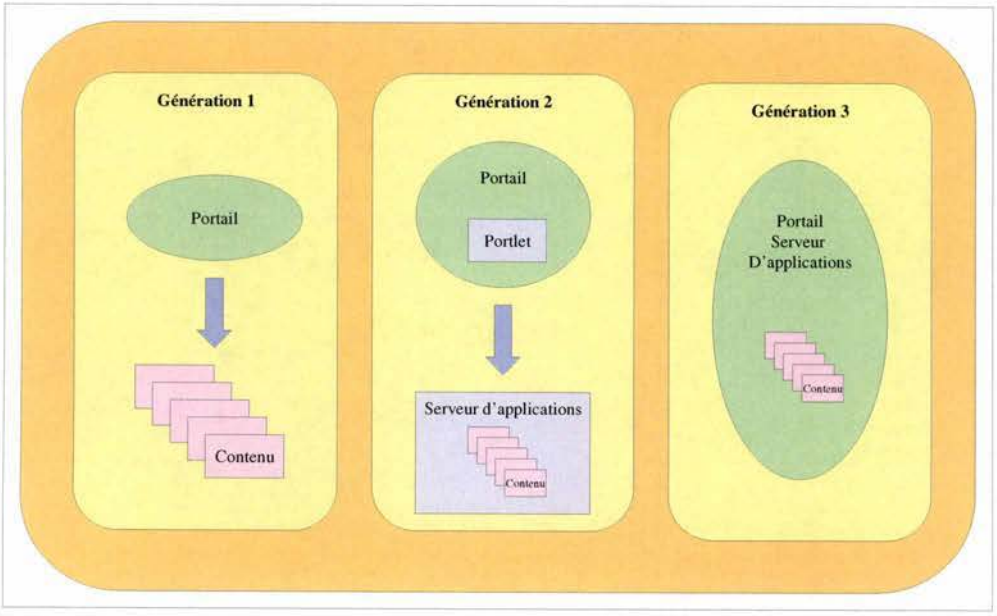


FIGURE 14 : 3 GENERATIONS DE PORTAILS

7.2.3 Classification

[Gene Phifer, 2002]

D'une manière générale, les portails se différencient en :

- Portail d'entreprise (sociétés, B2E)
- Portail B2B
- Portail B2C
- Portail décisionnel
- Portail collaboratif (forum, chat, échanges,...)
- Portail grand public (Yahoo, ...)

En fonction des besoins en matière de :

- Profil des utilisateurs
- Analyse des fonctionnalités et des services à rendre
- Etude des sources de données accessibles
- Niveau de sécurité

Un concept transversal à tous les types de portails décrits ici est la notion de :

- Portail vertical
- Portail horizontal

Ainsi, un portail horizontal représentera l'entreprise au travers de différents secteurs à un même niveau de la chaîne de production, tandis qu'un portail vertical donnera accès à un produit tout au long de sa chaîne de production. [Gerhard Schiefer & Anne Catharina Kreuder, Date inconnue]

Le schéma suivant illustre cette situation :

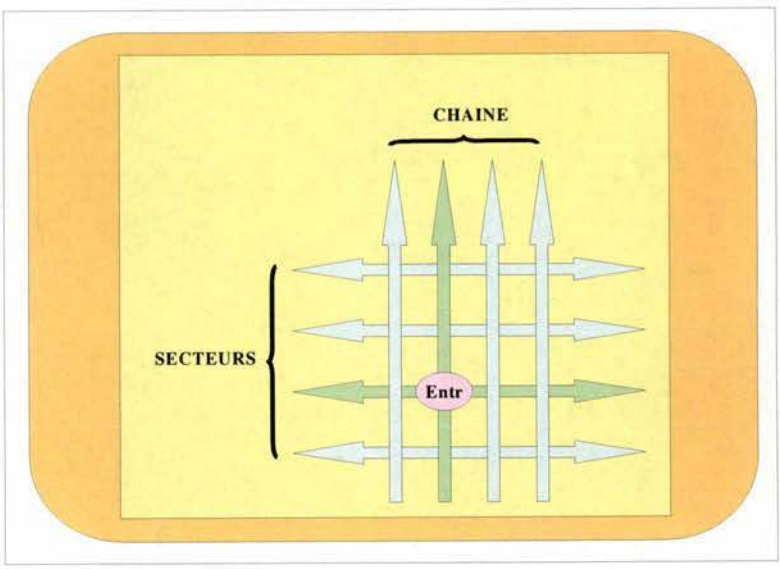


FIGURE 15 : PORTAIL VERTICAL/HORIZONTAL

On retiendra à titre d'exemple, dans le domaine agricole :

- Portail horizontal :
  - Commerce des machines agricoles
  - Production agricole
  - Commerce des denrées agricoles
  - Production de viande
  - ...
- portail vertical :
  - producteurs de viande
  - consommateurs
  - restaurants
  - inspection sanitaire
  - traçabilité

Ces notions sont valables dans le contexte de portail d'entreprise, d'une manière plus générale, un portail horizontal regroupera plusieurs sujets, tandis qu'un portail vertical, se concentrera sur un sujet particulier (sport, musique, cinéma, ...) tout en offrant la gamme habituelle de services (e-mail, chat, news, forums, moteur de recherche, ...)

Dans tous les cas, le choix du type de portail reposera sur les besoins et les cibles de l'entreprise.

On notera enfin, compte tenu de l'émergence des technologies sans fil, l'existence de portails dits « portails mobiles » adaptés à ces technologies (WAP, PDA, GSM, ...). Les concepts énoncés plus haut restent d'application pour ce type de portail.

7.3 « iPlanet Portal » architecture

[Sun Microsystems, 2000]

7.3.1 composants de base

L'architecture de « iPlanet Portal Server 3.0 » (iPS) , installé à l'Institut d'Informatique répond aux exigences précédemment exprimées.

iPS repose sur les protocoles HTTP et HTTPS offrant ainsi la possibilité de communications sécurisées ou non sécurisée.

Les principaux composant de iPS sont :

- iPlanet Portal Server
- iPlanet Profile Server
- iPlanet Portal Server Gateway
- firewall

7.3.1.1 iPlanet Portal Server

les 2 rôles du composant « portal server » sont :

- « portal server » proprement dit
- serveur d'application

en temps que « portal server », ce composant gère toutes les autorisations, les règles et les profils d'accès à la plate forme, tandis qu'en temps que serveur d'application ce composant fourni par exemple le mail service ou un utilitaire de gestion de fichiers.

7.3.1.2 iPlanet Profile Server

le composant « profile server » contient tous les profils associés aux objets iPS tels que domaine, rôle et utilisateurs (voir plus bas)

7.3.1.3 iPlanet Portal Server Gateway

le composant « gateway » sert d'interface et de barrière de sécurité entre les sessions clientes venant de l'Internet et l'Intranet d'entreprise.

Il a 2 fonctions principales :

- la fourniture d'un service d'authentification, permettant l'accès ou l'interdiction d'accès au portail
- la fourniture d'un service de correspondance et de réécriture permettant l'accès à des liens fournissant du contenu Intranet

7.3.1.4 firewall

Les composants « firewall », s'ils ne sont pas indispensables au fonctionnement du portail apportent une plus grande sécurité dans le cadre d'applications orientées Web.

7.3.2 Architecture à 2 machines

Tous ces composants peuvent en théorie être déployés sur une seule et même machine, cependant dans un contexte de portail d'entreprise, on en trouvera plus couramment une version distribuée, la configuration pouvant être plus ou moins différente d'un environnement à un autre, selon la complexité de l'architecture

Le schéma ci-dessous, propose un exemple de configuration répartie sur deux machines.

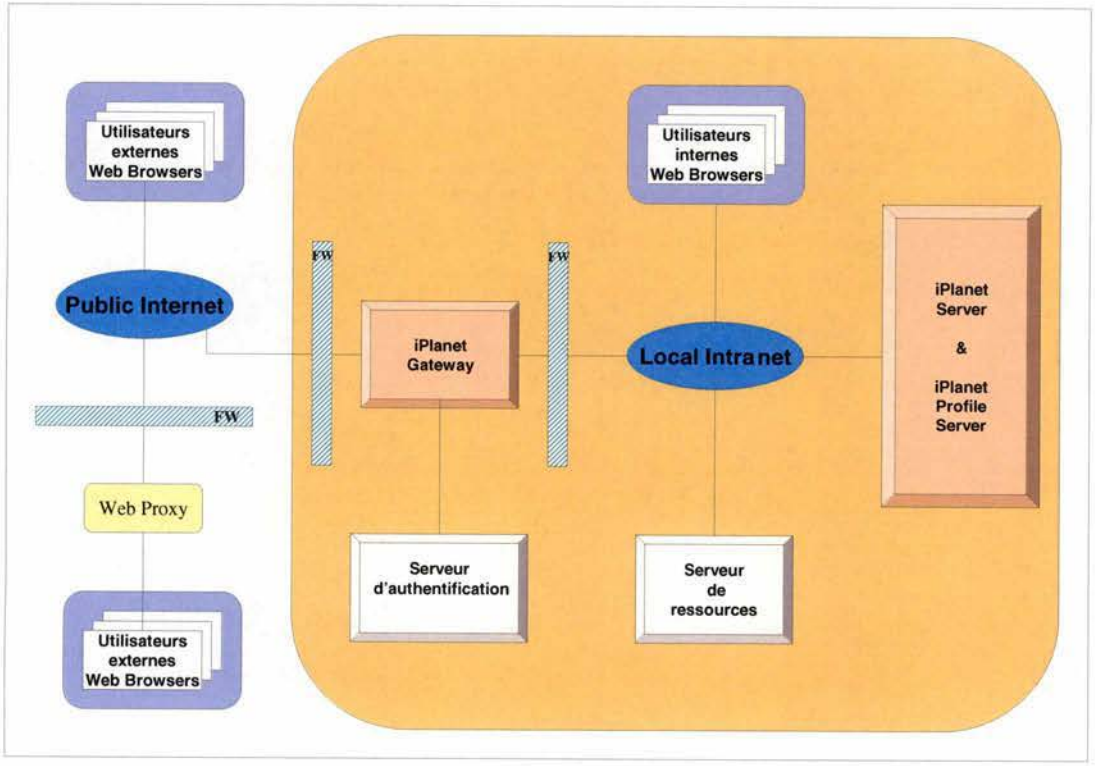


FIGURE 16 : IPS – ARCHITECTURE À 2 MACHINES

Dans cet exemple, iPS se réparti comme suit en terme de machines:

- le gateway et éventuellement l'infrastructure firewall. Ce gateway peut accéder de manière interne à :
  - un serveur d'authentification (LDAP, ….)
- le portal server et le profile server accèdant via l'Intranet aux :
  - serveurs de ressources hébergeant les applications à fournir (WebFacInfo par exemple)

7.3.3 architecture multi-machines

dans une configuration plus complexe on retrouve un portail organisé sur base de 2 gateways et 3 serveurs comme illustré ci-dessous :

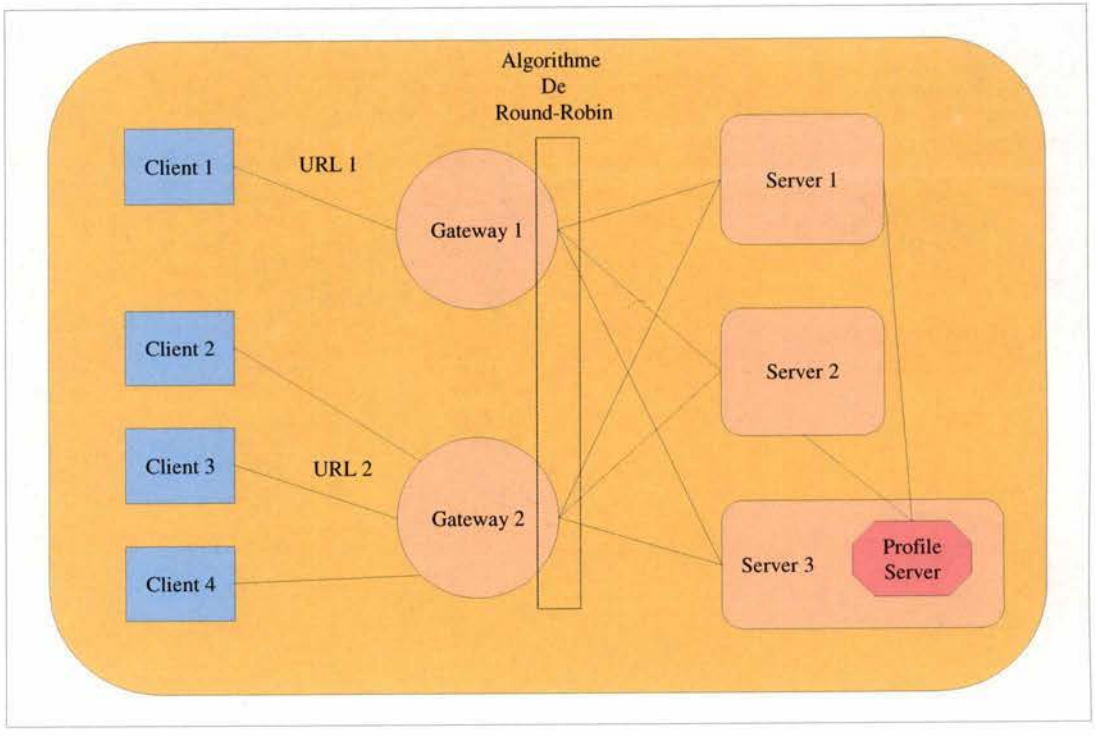


FIGURE 17 : IPS – ARCHITECTURE MULTI - MACHINES

ce schéma montre de gauche à droite :

- les clients initiant chacun une session sur une URL déterminée
- les 2 gateways sur 2 machines différentes
- l'assignation des sessions clientes aux serveurs distribués selon l'algorithme de « Round-Robin »[1]
- les 3 serveurs sur 3 machines différentes
    - le serveur 3 avec le « profile server » installé
    - les serveurs 1 et 2 configurés pour résoudre toutes les requêtes de profil vers le serveur 3

[1] L'algorithme de « Round-Robin », aussi appelé algorithme du « tourniquet », est un algorithme de répartition des requêtes sur plusieurs serveurs. Ces serveurs sont organisés en une liste circulaire. L'algorithme de « Round-Robin » attribue les requêtes, dans l'ordre où ceux-ci se présentent dans la liste avec retour au premier en fin de liste [Bruce Powel Douglass, 2002]. Dans un autre contexte, une variante de cet algorithme est aussi utilisée dans les systèmes d'exploitation pour l'ordonnancement des processus avec préemption [Dominique Revuz, 1998].

7.3.4 architecture multi-machines avec load balancing

Une version avec load balancing de l'architecture permet de faire apparaître les multiples gateways comme un gateway unique, le logiciel de load balancing mis en place assurant la répartition des sessions clientes sur les différents gateway en fontion de la charge respective de ceux-ci. Comme dans les architectures précédentes, les sessions clientes sont réparties sur plusieurs serveurs, l'un d'entre eux étant installé en cohabitation avec le « profile server » et les autres étant configurés pour s'y référer pour toute requête de profil.

Cette situation est illustrée dans le schéma suivant :

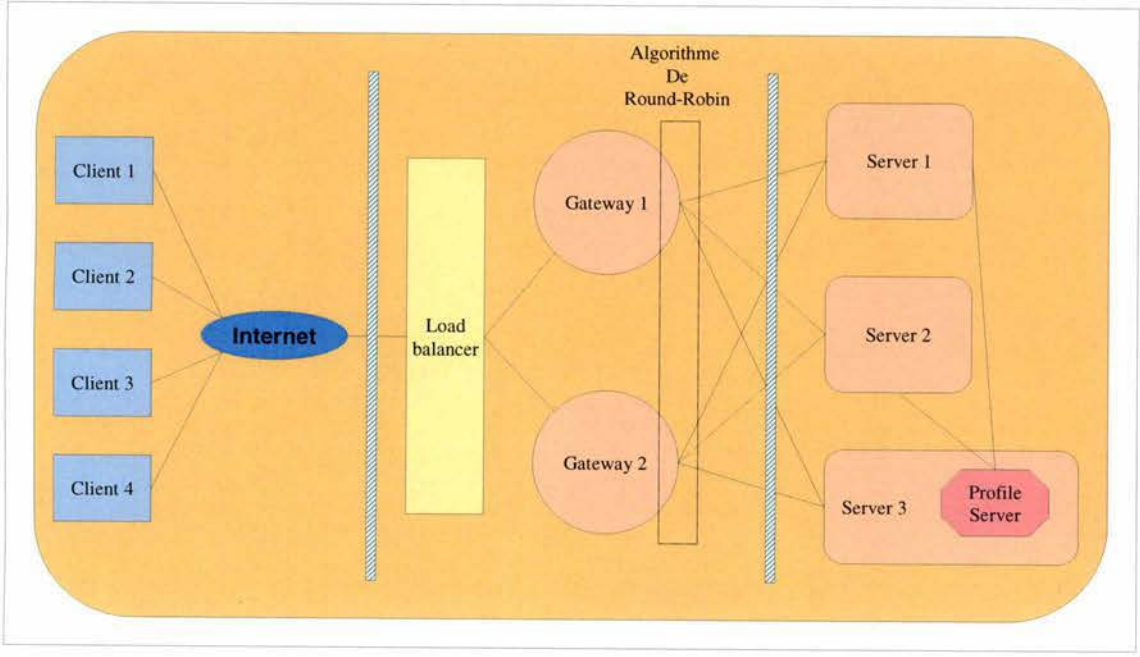


FIGURE 18 : IPS – ARCHITECTURE MULTI – MACHINES AVEC LOAD BALANCING

7.3.5 « iPlanetPortal » à l'Institut

L'installation de iPP à l'Institut est réalisée dans sa version la plus simple, sur une seule machine UNIX, le serveur « Backus ».
iPP bénéficie de la sorte du système d'authentification UNIX.

Il est prévu d'installer l'application WebFacInfo sur cette même machine.

Ainsi, les différents composants du système représentés à la figure 18, sont-ils tous rassemblés sur le serveur « Backus ».

Cette configuration est représentée dans la figure 19 ci-dessous :

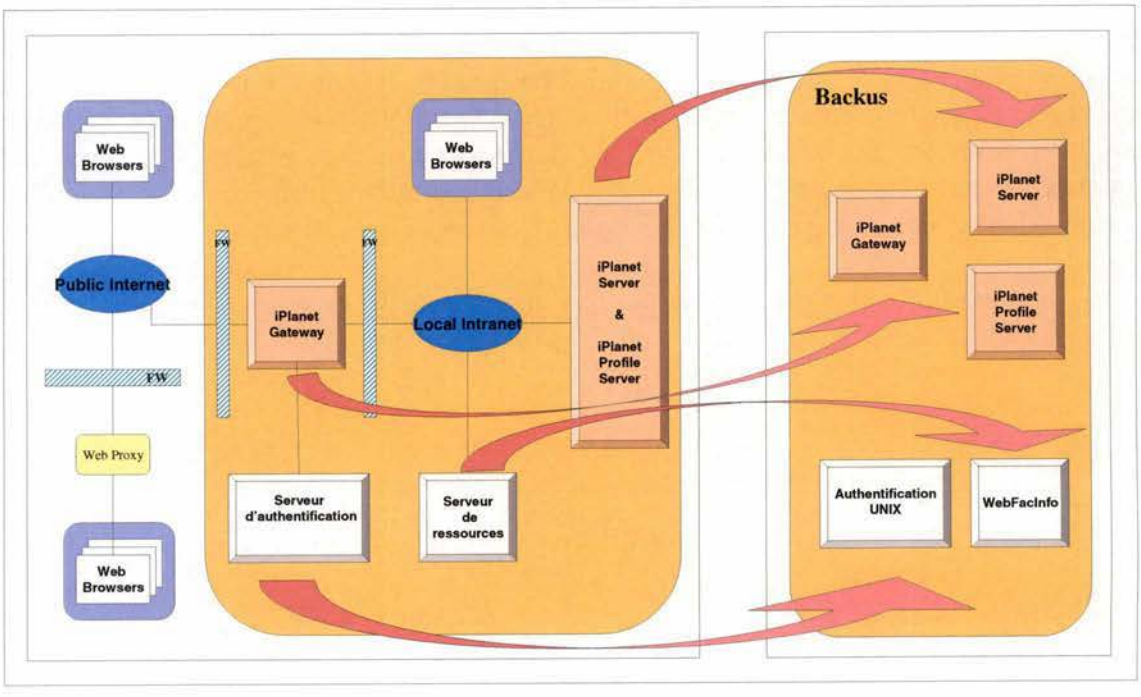


FIGURE 19 : IPP A L'INSTITUT

Ce schéma ne montre pas l'intégration du serveur « Backus » dans l'infrastructure des FUNDP et ne symbolise pas les accès via l'Intranet ou l'Internet. Le lecteur se reportera à la figure 35 illustrant cette infrastructure.

7.4 « iPlanet Portal » organisation et structure

l'organisation des utilisateurs et des applications au sein de iPP est assurée par une hiérarchie appellée « *the role tree* ». Cette hiérarchie permet de désigner un administrateur dont les rôles sont :

- la gestion des gateways et des serveurs
- la gestion des contrôles d'accès au site
- la gestion du design des pages du portail
- la gestion du « *look and feel* » des bureaux des utilisateurs.

La hiérarchie de iPP est organisée sur 4 niveaux :

- Root
- Domaine
- Rôle
- Utilisateur

Cette hiérarchie est établie sur le modèle parent - enfants , permettant l'héritage (des attributs) entre les différents niveaux. Chaque niveau se voit attribué un administrateur ayant des privilèges correspondant à son niveau hiérarchique.

Chaque enfant peut hériter des attributs spécifiés au niveau de son parent, cependant à chaque niveau, on peut attribuer des règles et des attributs plus restrictifs ou plus permissifs selon les circonstances.

IPP possède 2 types d'attributs différents :

- **Les attributs globaux**, applicables à l'ensemble de la plate forme et configurables uniquement par le super utilisateur (paramètres gateways et serveurs)
- **Les attributs modifiables par l'utilisateur** ou plus exactement par l'administrateur du niveau correspondant. Ainsi l'administrateur de domaine pourra modifier les attributs du niveau domaine et ceux-ci seront valables pour les niveaux inférieurs. De même, certains attributs peuvent être personnalisés au niveau utilisateur, et seront donc modifiés par chacun d'entres eux pour son compte propre.

7.4.1 Le niveau « root »

Le niveau « root » est le sommet du « role tree » , il est administré uniquement par le super administrateur. Ce dernier à tous les droits d'écriture et de lecture pour tous les attributs du portail.

7.4.2 Le niveau domaine

Le niveau domaine est utilisé pour administrer une grande quantité de ressources et d'utilisateurs. Ainsi, une grande entreprise avec plusieurs divisions pourra consacrer un domaine à chacune d'entre elles.

7.4.3 Le niveau rôle

Le niveau rôle défini des groupes d'utilisateurs qui sont membres de ce rôle. C'est à ce niveau que l'on définit un ensemble d'attributs et de règles qui régissent le contenu des utilisateurs

A noter que lors de l'installation, iPP crée automatiquement un rôle administrateur et un rôle par défaut. Le rôle administrateur ne peut pas être supprimé ni renommé, par contre le rôle par défaut peut être modifié et étendu selon les besoins.

Lors d'une tentative de login, un utilisateur identifié par le gateway est adressé à l'un ou l'autre des rôles définis selon son identité. Si un nouvel utilisateur non affecté à un rôle particulier est identifié, il est adressé au rôle par défaut.

7.4.4 Le niveau utilisateur

Le niveau utilisateur renferme toutes les personnes autorisées à utiliser le portail. Chaque utilisateur possède un ensemble unique d'attributs constitué:

- Des attributs hérités des niveaux supérieurs
- D'attributs uniques, personnalisés, qui spécifient une manière personnelle d'exécuter le portail.

Cette organisation hiérarchique est illustrée de manière générique dans la figure ci-dessous :

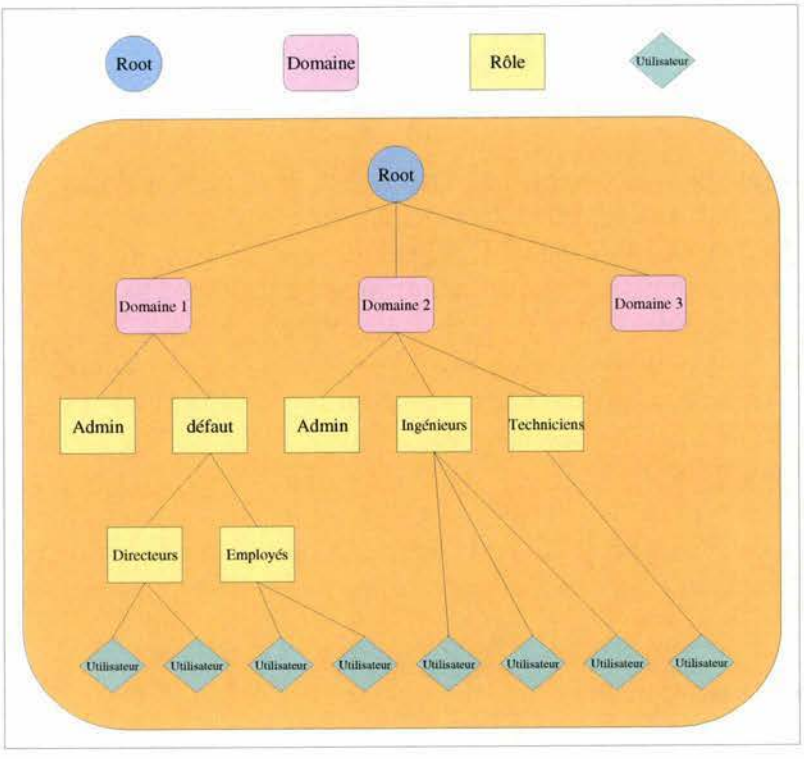


FIGURE 20 : IPP ORGANISATION GENERALE

Et de manière plus spécifique dans la suivante :

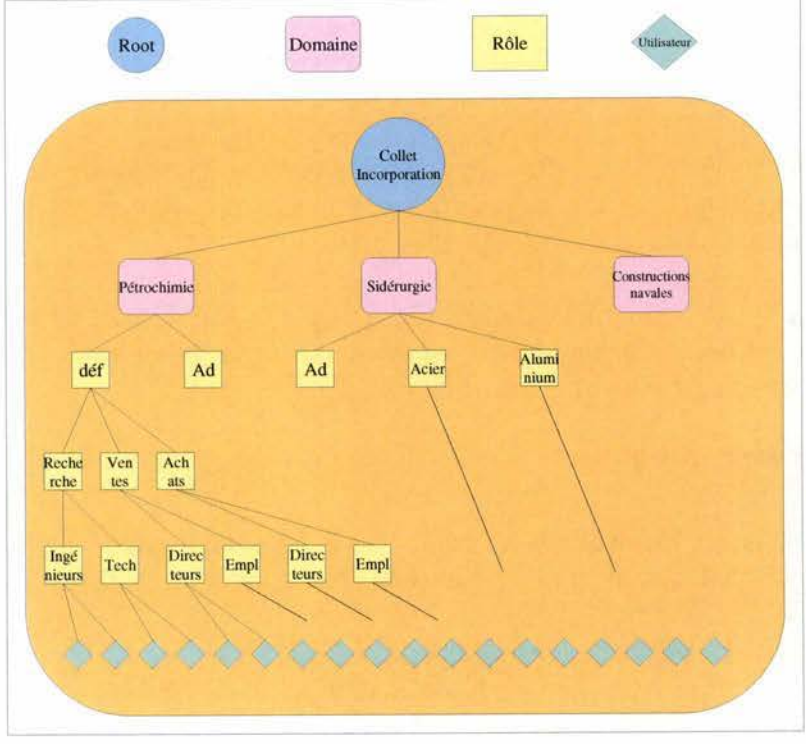


FIGURE 21 : IPP EXEMPLE D'ORGANISATION

Enfin, la figure suivante illustre une possibilité d'organisation d'iPP dans le cadre de l'Institut d'informatique et de l'intégration de WebFacInfo. Cette organisation est donnée ici à titre d'exemple et n'augure en rien de l'implantation réelle :

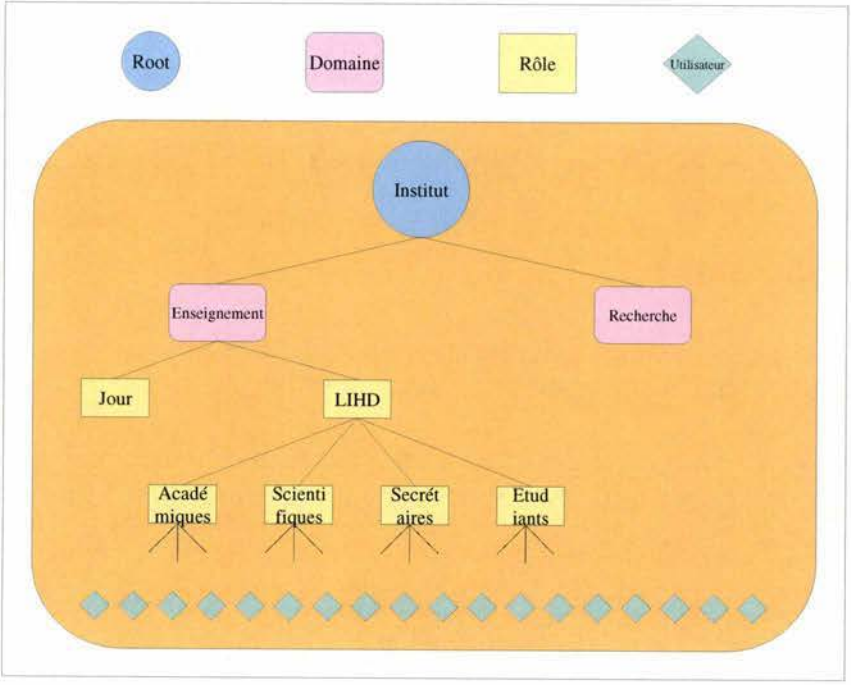


FIGURE 22 : IPP UN EXEMPLE D'IMPLANTATION

7.5 Sécurité, authentification

Le but de ce paragraphe n'est pas de décrire en détail l'administration de la sécurité au niveau d'iPP, mais bien de fournir un aperçu des possibilités du portail en matière de sécurité et d'authentification. Pour une information détaillée à ce sujet, le lecteur se reportera à l' « Administration Guide » de IPP.

7.5.1 installation

IPP utilise l'authentification pour vérifier l'identité des utilisateurs qui essayent de se connecter au portail.

Par défaut, lors de l'installation de iPP, les modules suivants d'authentification sont disponibles :

- S/Key
- SecurID
- SafeWord
- UNIX authentication
- Windows NT
- LDAP
- Personal Digital Certificates
- RADIUS

- Membership

Les rôles de l'administrateur lors de cette installation sont :

- Paramétrer les modules qui nécessitent de l'information (ex : l'adresse d'un serveur LDAP)
- Activer/inactiver les différents modules selon les besoins
- Personnaliser les écrans de loggin et les messages de rejet
- Personnaliser les comportements du portail par rapport à l'authentification effectuée pour charger les sites Web et les applications en rapport avec cette authentification.

7.5.2 Personnalisation

En fonction des besoins, le super administrateur ou les administrateurs de domaine peuvent personnaliser les modules d'authentification disponibles au niveau rôle et au niveau utilisateur.
Ainsi chaque groupe d'utilisateur pourra accéder à un menu différent, proposant les différentes méthodes d'authentification en fonction des spécificités des sites et des applications disponibles dans leur rôle.

Toutes ces tâches de personnalisation et de paramétrisation se font via la console d'administration accessible par les administrateurs.

8 PORTAIL ET AUTHENTIFICATION SIMPLIFIEES

[Allan Ant, 2003]

8.1 Généralités

Nous l'avons vu, le principe d'un portail et la notion de profil d'utilisateur repose essentiellement sur l'authentification des utilisateurs qui tentent de se connecter sur celui-ci. Une fois ce contrôle d'identité initial franchi, un portail donne accès à une grande quantité de ressources sécurisées qui peuvent être :

- Des sites Web
- Des applications

Dans les 2 cas, ces ressources peuvent nécessiter une authentification complémentaire, spécifique à la ressource accédée.

Il est évident qu'un utilisateur accédant à plusieurs ressources successivement va se trouver confronter à une succession de procédures d'authentification au cours de sa navigation.

Ces procédures peuvent être de natures différentes :

- **UserID et mot de passe** qui constitue le moyen d'authentification le plus faible tant un mot de passe peut être aisément cassé.
- **Authentification forte** de natures différentes

    - One-time password (OTP) tokens, associant un PIN avec un token
    - Smart cards, incluant une infrastructure PKI
    - Moyens biométriques, utilisant une carctéristique unique :

        - ✓ Physique (empreinte digitale, iris, rétine, ...)
        - ✓ Comportementale (voix, démarche, ...)

Dans cette optique, il est utile d'envisager une solution visant à réduire le nombre de ces procédures d'authentification, surtout en raison du nombre croissant de ressources sécurisées accessibles.

Cette solution n'est cependant pas triviale, en raison :

- Des règles en matière de longueur et format de mot de passe d'un système à un autre
- Des temps de vie différents pour les mots de passe d'un système à un autre
- Des problèmes liés à l'authentification forte :

    - L'incompatibilité de certains produits d'authentification forte avec certains systèmes
    - Les besoins différents d'authentification selon les utilisateurs
    - Les besoins d'installation des terminaux de lecture (scanner biométrique) ou de diffusion des accessoires (securID) ne doivent pas priver un utilisateur de ses accès

Cette problématique de procédures d'authentification à répétition a un impact sur :

- Les utilisateurs
  - Plusieurs mots de passe à retenir
  - Plusieurs accessoires à gérer (SecurID, smartCard , ...)
  - Certaines techniques biométriques sont invasives (rétine)
  - Aspect répétitif
  - Perte de temps
- Les « helpdesk »
  - Nombreux appels pour des oublis de mot de passe (+ de 25% des appels)
- La sécurité
  - Un employé « helpdesk » qui doit réinitialiser un mot de passe doit avoir des droits d'administrateur
  - Risque que les utilisateurs gardent des traces écrites de leurs mots de passe
  - Risque non négligeable de vol d'identité

Dans le but de réduire au maximum le nombre et la complexité des procédures d'authentification, il existe 3 types de middleware d'authentification simplifiée:

- Password management
- Single-sign-on (SSO)
- Authentication management infrastructure (AMI)

8.2 Password management

Un middleware de password management assure au moins une des fonctions suivantes :

- **Synchronisation automatique des mots de passe**, de telle sorte qu'un changement de mot de passe sur un système soit propagé aux autres systèmes sur le réseau
- **Réinitialisation personnelle des mots de passe**, permettant à l'utilisateur de réinitialiser son mot de passe au travers de différents systèmes
- **Réinitialisation des mots de passe par un tiers**, permettant à l'opérateur du helpdesk de réinitialiser un mot de passe utilisateur au travers de différents systèmes

La réinitialisation personnelle des mots de passe est généralement assurée via :

- Une interface Web
- Une interface IVR (Interactive Voice Response)

Dans ces deux cas, tant en cas d'oubli de mot de passe, qu'en cas de changement de celui-ci, une méthode alternative d'authentification, basée sur un jeu de questions / réponses est utilisée.

La réinitialisation des mots de passe par un tiers est assurée par une interface permettant à l'opérateur helpdesk de modifier le mot de passe sur un ou plusieurs système après s'être assuré de l'identité de son correspondant par un jeu de questions / réponses. Afin d'augmenter la sécurité, ce jeu de QR peut être différent de celui utilisé en cas de réinitialisation personnelle.
A noter que si l'accès à cette interface d'administration des mots de passe est sécurisé, le problème lié à l'attribution de droits d'administrateur à l'employé helpdesk est réglé.

La synchronisation automatique des mots de passe, indépendamment du type d'interface et du type d'intervenant (utilisateur ou helpdesk) est assurée par 2 types de middleware de password management:

- Password management par propagation
- Password management par centralisation

8.2.1 Password management par propagation

Dans un système de password management par propagation, tout changement de mot de passe sur un système est détecté par le middleware de password management et répercuté sur les autres systèmes.

Un tel système de propagation des mots de passe suppose que :

- Le middleware de password management est capable de détecter un changement de mot de passe sur un système. A cet égard, chaque produit possède ses propres listeners vers un nombre limité de système
- Les règles en matière de mot de passe sont les mêmes sur les systèmes vers lesquel le changement est propagé que sur le système sur lequel le changement est initié. Cela suppose la tenue à jour d'une liste des systèmes compatibles.

On le voit ici, cette technique est limitative en raison de ces deux conditions.

Un tel système de password management par propagation est illustré dans la figure ci-dessous :

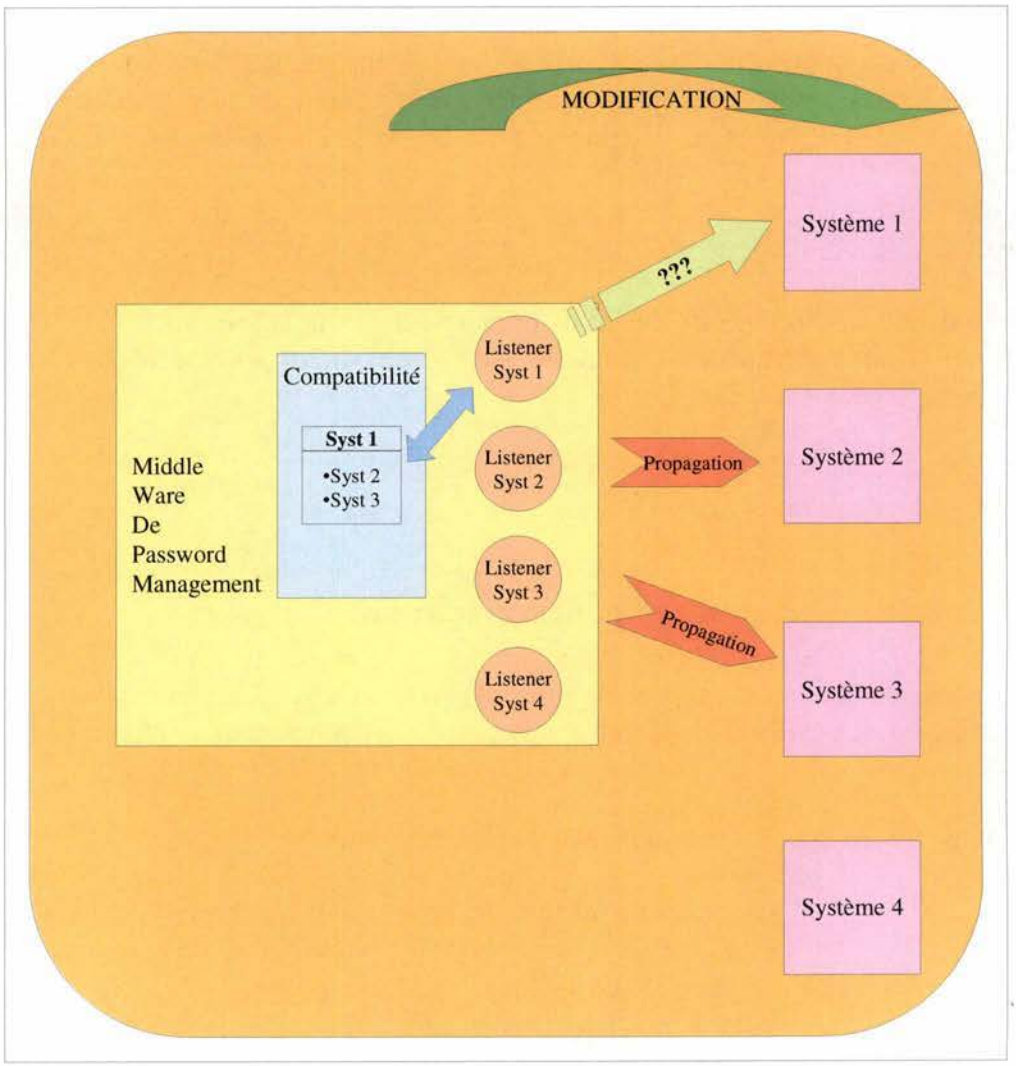


FIGURE 23 : PASSWORD MANAGEMENT PAR PROPOAGATION

8.2.2 Password management par centralisation

Dans un système de password management par centralisation, les requêtes d'authentification sont toutes dirigées vers un serveur central qui autorise ou refuse l'accès vers le système cible. Si un changement est effectué au niveau d'un système cible, il est aussitôt répercuté au niveau du serveur central.

Un tel système de password management par centralisation est illustré dans la figure ci-dessous :

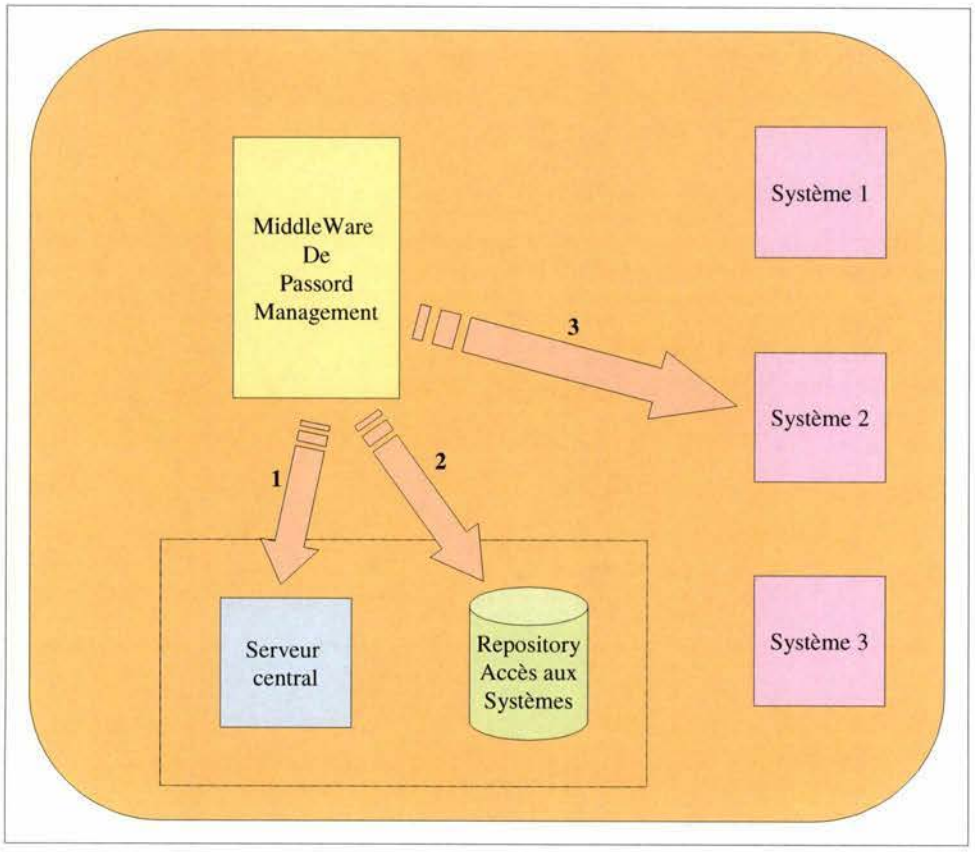


FIGURE 24 : PASSWORD MANAGEMENT PAR CENTRALISATION

8.3 Single Sign On (SSO)

Un middleware SSO permet à un utilisateur d'accéder plusieurs plates-formes ou applications après s'être authentifié une seule fois

Il existe 2 modèles SSO :

- Le « *single network credential model* »
- Le « *multiple network credential model* »

Le « ***single network credential model*** » repose sur la génération d'un « *network credential* » qui est retourné à l'utilisateur. Ce « *network credential* » est ensuite utilisé pour l'authentification sur les différents systèmes.

Le « ***multiple network credential model*** », plus adapté aux environnements hétérogènes actuels, repose sur une authentification effectuée au niveau du middleware SSO, lequel contacte sa DB pour retrouver les login respectififs de chaque système, afin d'y donner accès. Dans un tel système, l'authentification effectuée en arrière plan est accomplie avec un « *credential* » différent pour chaque système.

Dans le cadre du « *multiple network credential model* », on distingue encore 2 types de mécanismes SSO :

- SSO par usage de scripts
- SSO par usage de cookies

8.3.1 SSO par usage de scripts

SSO par usage de script est illustré dans le schéma ci-dessous :

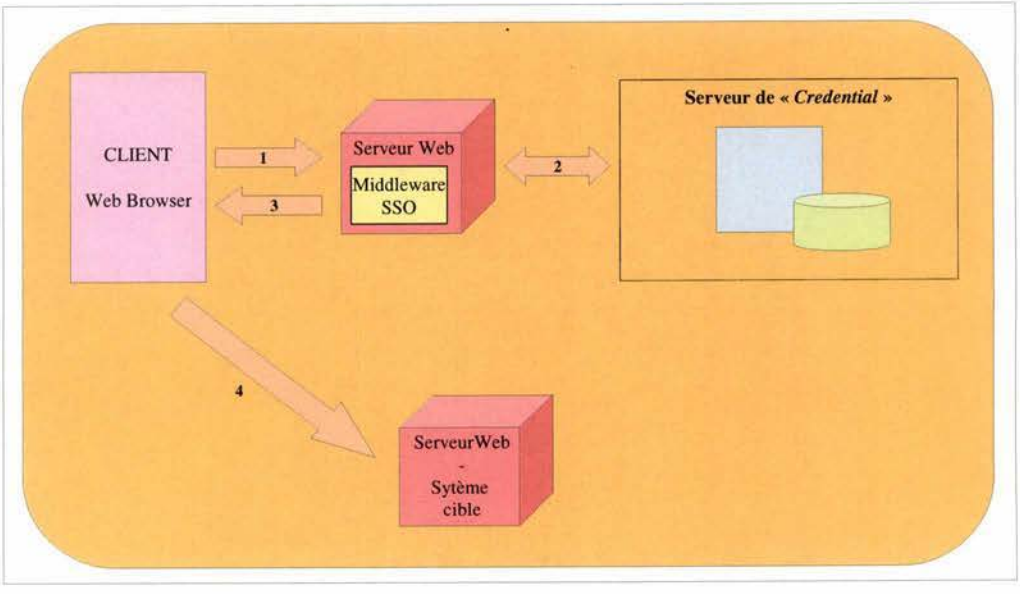


FIGURE 25 : SSO PAR USAGE DE SCRIPTS

Les différentes étapes illustrées par la figure 25 sont :

1. l'utilisateur effectue une authentification initiale sur le serveur SSO
2. lors de l'accès vers un système cible, le système SSO récupère le script et le « *credential* » correspondant à ce système dans son repository
3. le système SSO fourni ces indications au client
4. le client utilise ces informations pour accéder au système cible

8.3.2 SSO par usage de cookies

SSO par usage de cookies est illustré dans le schéma ci-dessous :

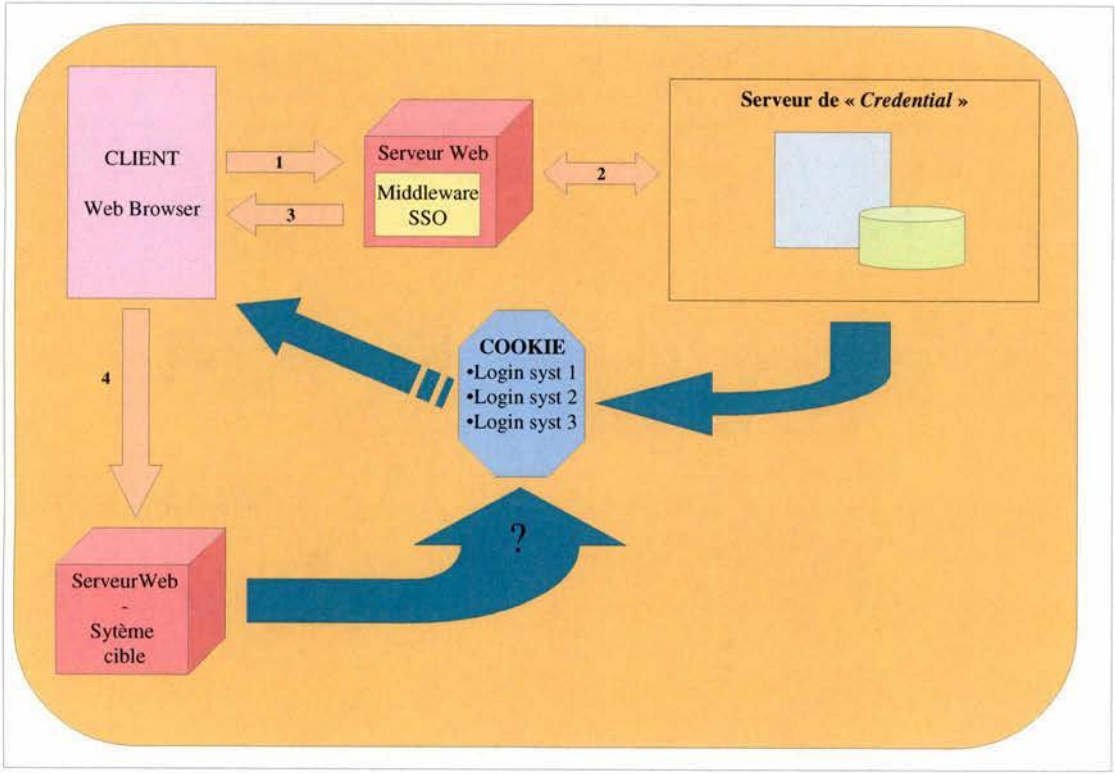


FIGURE 26 : SSO PAR USAGE DE COOKIES

Les différentes étapes illustrées par la figure 26 sont :

1. l'utilisateur fourni son identité
2. le middleware SSO authentifie l'utilisateur et récupère les différents « *credentials* » sur son «*serveur de credentials* »
3. le middleware SSO dépose un cookie au niveau du client.
4. lors d'un accès vers un système cible, celui-ci récupère le « *credential* » le concernant dans le cookie

8.4 Authentication Management Infrastructure (AMI)

Un middleware AMI permet à un utilisateur d'accéder à plusieurs systèmes après s'être authentifié auprès d'un serveur unique pouvant utiliser plusieurs modes d'authentification.

Ce type de middleware permet d'augmenter la sécurité par l'utilisation de méthode d'authentification plus fortes que les traditionnels mots de passe utilisés sur les systèmes hérités et fourni un point d'administration unique pour la gestion de la politique d'authentification. C'est également un moyen de migration « en douceur » vers des techniques d'authentification fortes

Dans un tel middleware, les règles d'authentification peuvent être définies pour des utilisateurs uniques ou des groupes utilisateurs, et ces règles peuvent porter sur une ou plusieurs techniques d'authentification.

Ainsi , la figure suivante illustre t-elle un système AMI où tous les utilisateurs partagent un service d'authentification unique, mais où chaque groupe d'utilisateurs utilise un moyen ou une combinaison de moyens d'authentification différent :

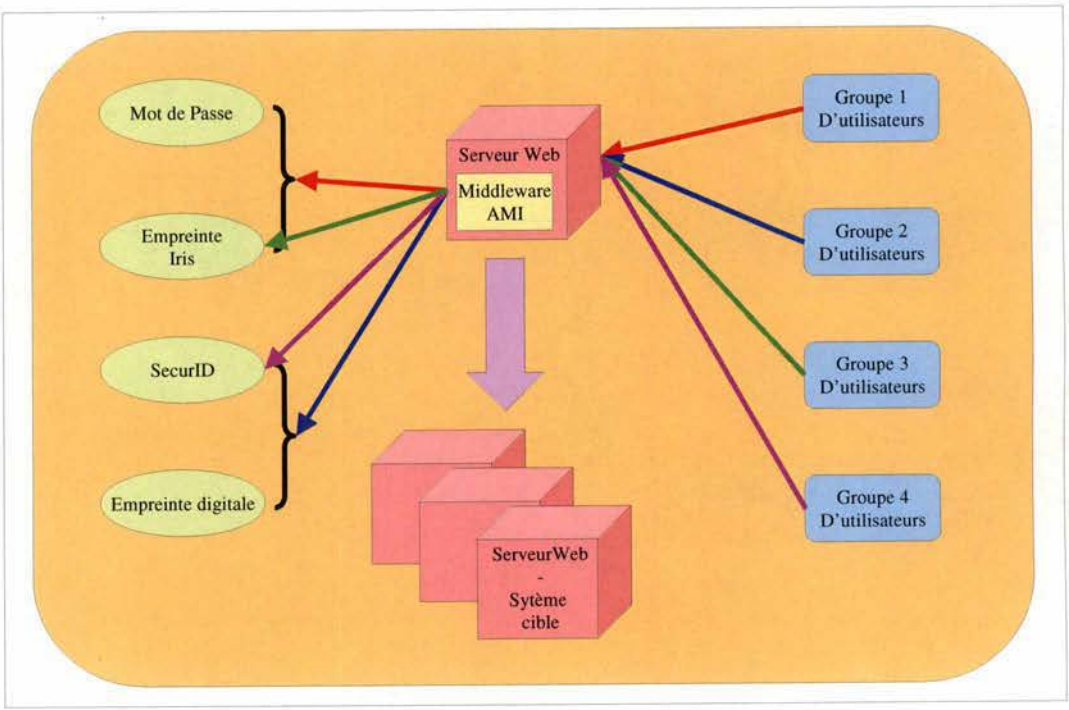


FIGURE 27 : MIDDLEWARE AMI

8.5 Avantages et inconvénients

8.5.1 Avantages

- Augmentation de la facilité d'utilisation
- Augmentation de la productivité
- Diminution de la surcharge administrative
- Augmentation de la sécurité par réduction du nombre de mot de passe
- Augmentation de la sécurité par utilisation de méthodes d'authentification forte

8.5.2 Inconvénients

- Diminution de la sécurité par le fait qu'il n'y a plus qu'un seul mot de passe
- Diminution des possibilités de choix en matière de méthodes d'authentification pour le futur
- Risques au développement (besoins des utilisateurs en matière de sécurité, possibilités des systèmes cible → nécessité d'une solide analyse préalable)

8.6 En résumé

Le password management et SSO ont comme objectifs communs :

- la facilité d'utilisation en réduisant le nombre de mot de passe à l'unité
- la gestion cohérente des mots de passe au sein d'une entreprise et un accès unique pour la réinitialisation des mots de passe par les opérateurs de helpdesk ou par l'utilisateur

tandis que la différence entre SSO et AMI est peu claire, en ce sens que :

- un certain nombre de middleware SSO supportent des moyens d'authentification forte
- de nomreux middleware AMI présentent des fonctionnalités proches du SSO

Ces 2 derniers types de middleware d'authentification convergent fortement.

Des besoins d'une entreprise en matière de sécurité ou de facilité d'emploi dépendra le choix du middleware d'authentification le plus approprié, comme illustré sur la figure suivante :

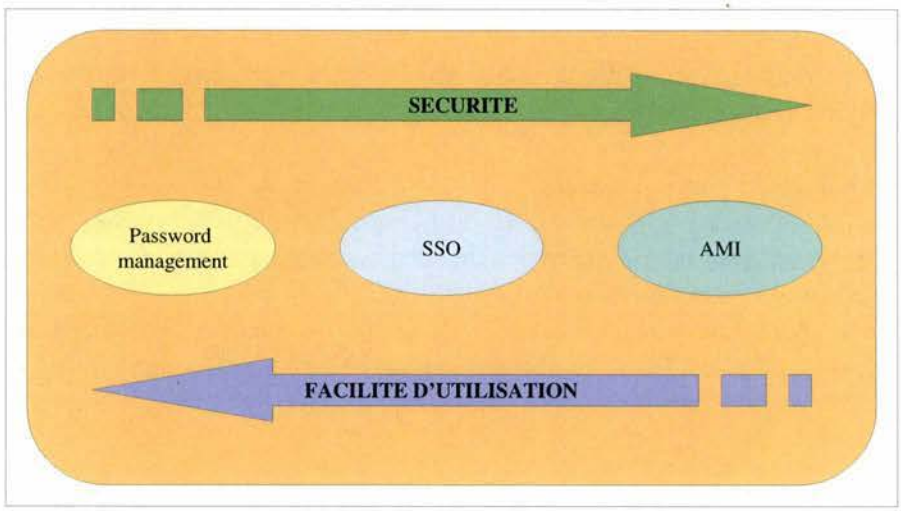


FIGURE 28 : MIDDLEWARE D'AUTHENFICATION

9 ANALYSE TECHNIQUE

Cette partie est consacrée à la description approfondie du composant *"Système"* décrit au point 4.3.

Selon les contraintes non fonctionnelles exprimées précédemment, l'application est développée en Java J2EE.

9.1 WebFacInfo

Le prototype développé sur base de la présente analyse à été baptisé « WebFacInfo », en rapport avec la volonté d'ouverture de la **fac**ulté d'**Info**rmatique sur l'Internet (**web**).

9.2 Architecture applicative "WebFacInfo"

L'architecture de l'application WebFacInfo est une architecture Model-View-Controler (MVC) classique, dans laquelle les requêtes du client sont traduites en terme de ressources applicatives.

La vue permet au client d'introduire des requêtes qui sont interceptées par le contrôleur qui se charge d'instancier le modèle adéquat.

La **vue** est composée de pages statiques Html et de pages Jsp.
Le **controleur** est représenté par une servlet qui centralise toutes les requêtes des clients. Une autre classe (le view dispacher) est chargée de choisir la vue appropriée en fonction de la requête et des résultats de celle-ci.
Le **modèle** est implémenté sous forme d'EJB de session « stateless », ceux-ci accèdent à la base de données.

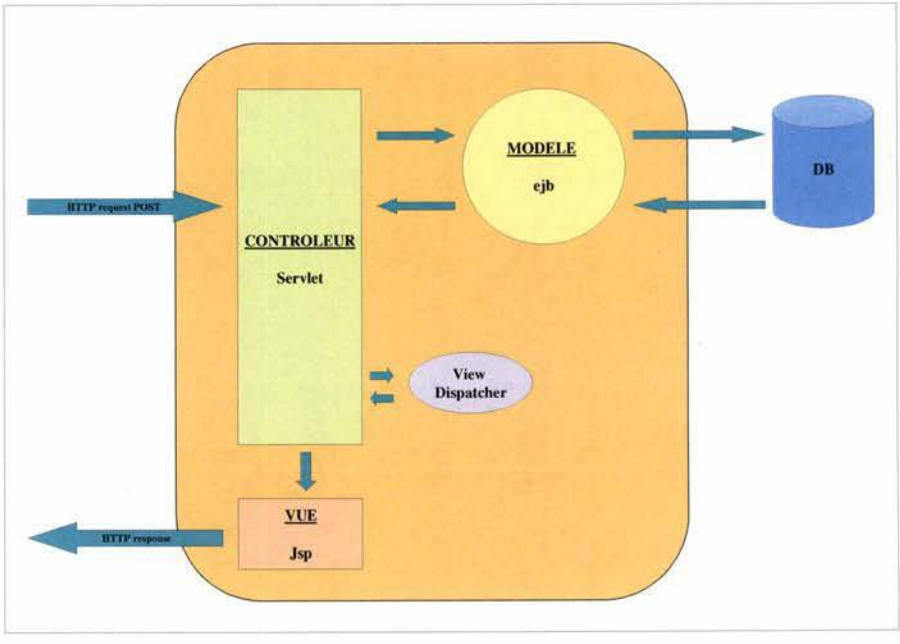


FIGURE 29 : ARCHITECTURE WEBFACINFO

Les données sont véhiculées entre les différents modules dans des Value-Objets qui encapsulent toutes les données d'une requête-réponse.

Enfin, les informations persistantes du client telle que l'identité de l'utilisateur ou le groupe auquel il appartient sont stockées dans l'objet HttpSession. L'objet HttpSession est un objet associé de façon permanente à un client durant toute sa visite sur le site.

9.2.1 La vue

La vue est composée principalement de pages Jsp.

Des pages statiques Html sont utilisées lorsque aucun traitement spécifique n'est requis. C'est le cas par exemple de la page de menu personnalisé.

Il est à noter que la vue ne contient AUCUN mapping vers des ressources coté modèle. Cela signifie que lorsque l'utilisateur clique sur un bouton 'Submit', il n'indique pas les ressources nécessaires au traitement de la requête mais bien la requête elle-même !

La vue doit permettre à l'utilisateur de surfer aisément sur le site, d'offrir un look-and-feel agréable et consistant ainsi que de permettre à l'utilisateur d'envoyer des requêtes puis d'afficher les résultats de celles-ci.

9.2.2 Le contrôleur

Toutes les requêtes sont redirigées automatiquement vers une servlet centrale qui se charge de traduire ces requêtes en ressources applicatives.
Cette redirection est obtenue par une configuration à cet effet du serveur d'application.

Un contrôle de l'identité est effectué en ce point central de l'application.

La servlet crée alors un objet command dont l'unique méthode "execute()" permet d'extraire les informations de l'HttpServletRequest qui sont alors envoyées à un EJB qui accède à la base de données afin de compléter son information avant de procéder au traitement spécifique de celle-ci. L'EJB créé ensuite un Value-Objet dont les variables d'instance sont garnies par les valeurs résultant des traitements.

Le Value-Objet est retourné à la servlet centrale qui délègue alors le choix de la vue à un ViewDispatcher. Cet objet inspecte le Value-Objet pour décider de la vue appropriée. Finalement, le Value-Objet est attachée à la requête de l'utilisateur de sorte que la vue puisse afficher toutes les données contenues dans le Value-Objet.

Le controleur agit comme un intermédiaire entre la vue et le modèle. C'est à ce niveau que la navigation du site est spécifiée.

9.2.3 Le modèle

Le modèle est composé d'EJB. Un EJB accède à la base de données et effectue des traitements sur les données. Il crée le Value-Object et garnit celui-ci des données à renvoyer.

Les EJB sont créés à partir d'une factory. Cette factory présente l'intérêt de faire du caching des home-object, ce qui accroît les performances du système.

Le modèle extrait les informations pertinentes de la base de données, les encapsule dans un Value-objet et retourne ce Value-Objet au controleur.

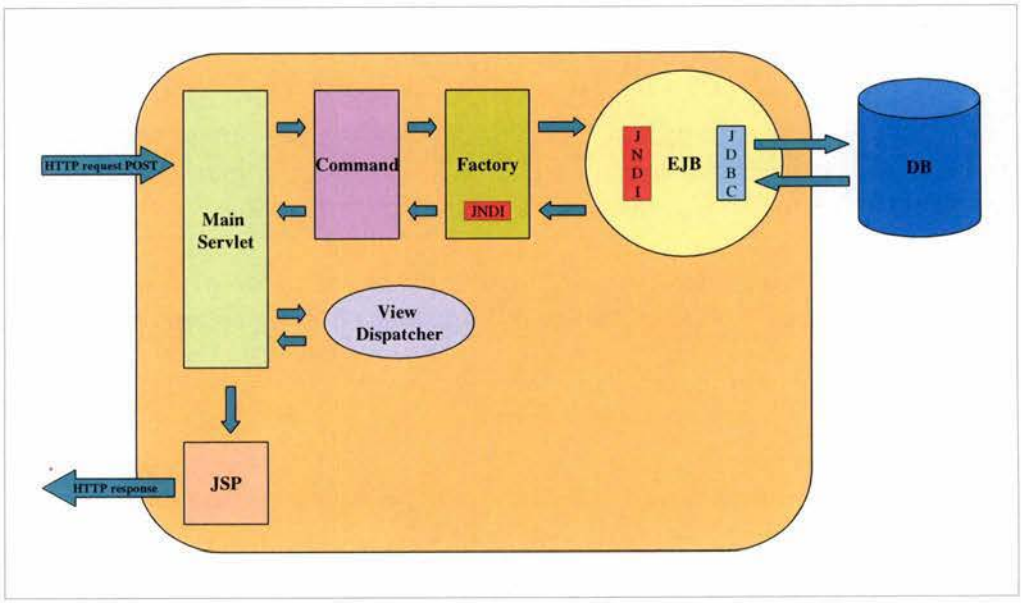
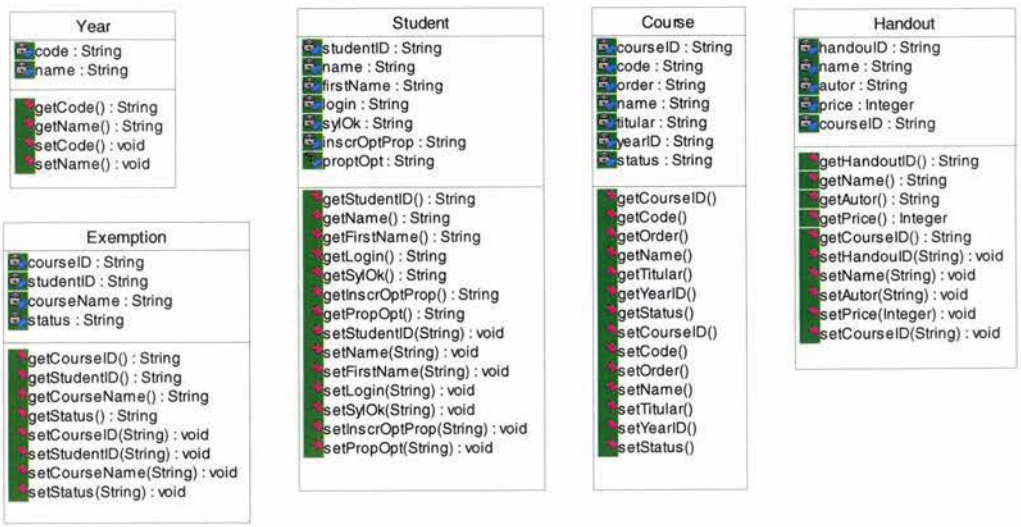


FIGURE 30 : ARCHITECTURE WEBFACINFO - DÉTAIL

9.3 Composants actifs

9.3.1 Les objets du domaine

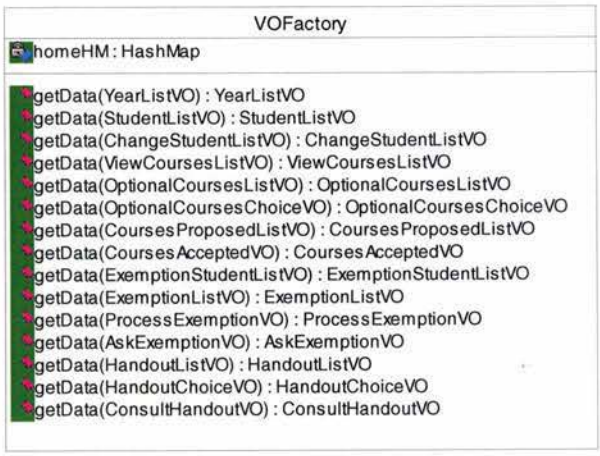


9.3.2 L'objet MainServlet

9.3.3 Les objets ValueObject

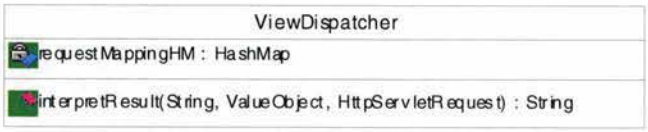
9.3.4 Les objets Command

9.3.5 L'objet VOFactory



9.3.6 Les objets EJB

9.3.7 L'obbjet ViewDispatcher



9.3.8 les objets de persistance

WebfacinfoDBAccess
con : Connection
dataSource : String
ps : PreparedStatement
sql : String
sql1 : String
sql2 : String
sql3 : String
sql4 : String
sql5 : String
sql6 : String
rs : ResultSet
hm : HashMap
getStudentList(yearCode : String) : Student[ ]
insertNewStudent(student : Student, year : String, yearBis : String) : void
deleteStudents(studentToSuppres : String[ ]) : void
getStudentByLogin(login : String) : Student
getYearList() : Year[ ]
getOptionNumber(studentID : String) : Integer
getInscriptOptPropNumber(studentID : String) : Integer
getCountInscriptOptBase(studentID : String) : Integer
getCountExemptionAsked(studentID : String) : Integer
getFullCoursesList(studentID : String) : Course[ ]
getCoursesObligList(studentID : String) : Course[ ]
setExemptionRequest(exemptionToAsk : String[ ], studentID : String) : void
getAskedExemptions(studentID : String) : Exemption[ ]
getHandoutList(login : String) : Handout[ ]
getOptionalCoursesChoice(login : String) : Course[ ]
getOrder(login : String) : String[ ]
setHandoutChoice(handoutID : String, argname : String) : void
getTotalPrice(studentID : String) : Integer
deleteHandout(studentID : String) : void
getHandoutChoice(studentID : String) : Handout[ ]
getSylStatus(studentID : String) : String
setSylStatus(studentID : String, sylOk : String) : void
getExemptionStudentList() : Student[ ]
getExemptionList(studentID : String) : Exemption[ ]
setExemptionStatus(courseID : String, studentID : String, status : String) : void
updateCoursesList(courseID : String, studentID : String) : void
getOptionalCoursesList(login : String) : Course[ ]
setOptionalCoursesChoice(studentID : String, courseID : String) : void
setOptionalCourseProposed(login : String, courseID : String, courseName : String, courseTit : String, yearID : String) : void
getYearByLogin(login : String) : Year
deleteOptionalCoursesChoice(studentID : String) : void
getCourseProposed(login : String) : Course
getStudentListWithProposal() : Student[ ]
getCourseProposedByID(courseID : String) : Course
setCourseProposedStatus(courseID : String, status : String) : void
updateStudentCourseProposedRefused(studentID : String) : void
updateStudentCourseProposedAccepted(courseID : String, studentID : String) : void
getPreviousProposal(courseID : String) : Course
isProposedCourse(courseID : String) : Boolean
setOptionalProposedCoursesChoice(courseID : String, studentID : String) : void

9.4 Class diagram

le class diagram est le diagramme de base de la modélisation OO. Il s'agit d'une modélisation de la statique du système. [Khawar Zaman Ahmed & Cary E. Umrish, 2001]

Le class diagram représente :

- Un ensemble de classes et d'interfaces
- Les relations et collaborations entre ces éléments

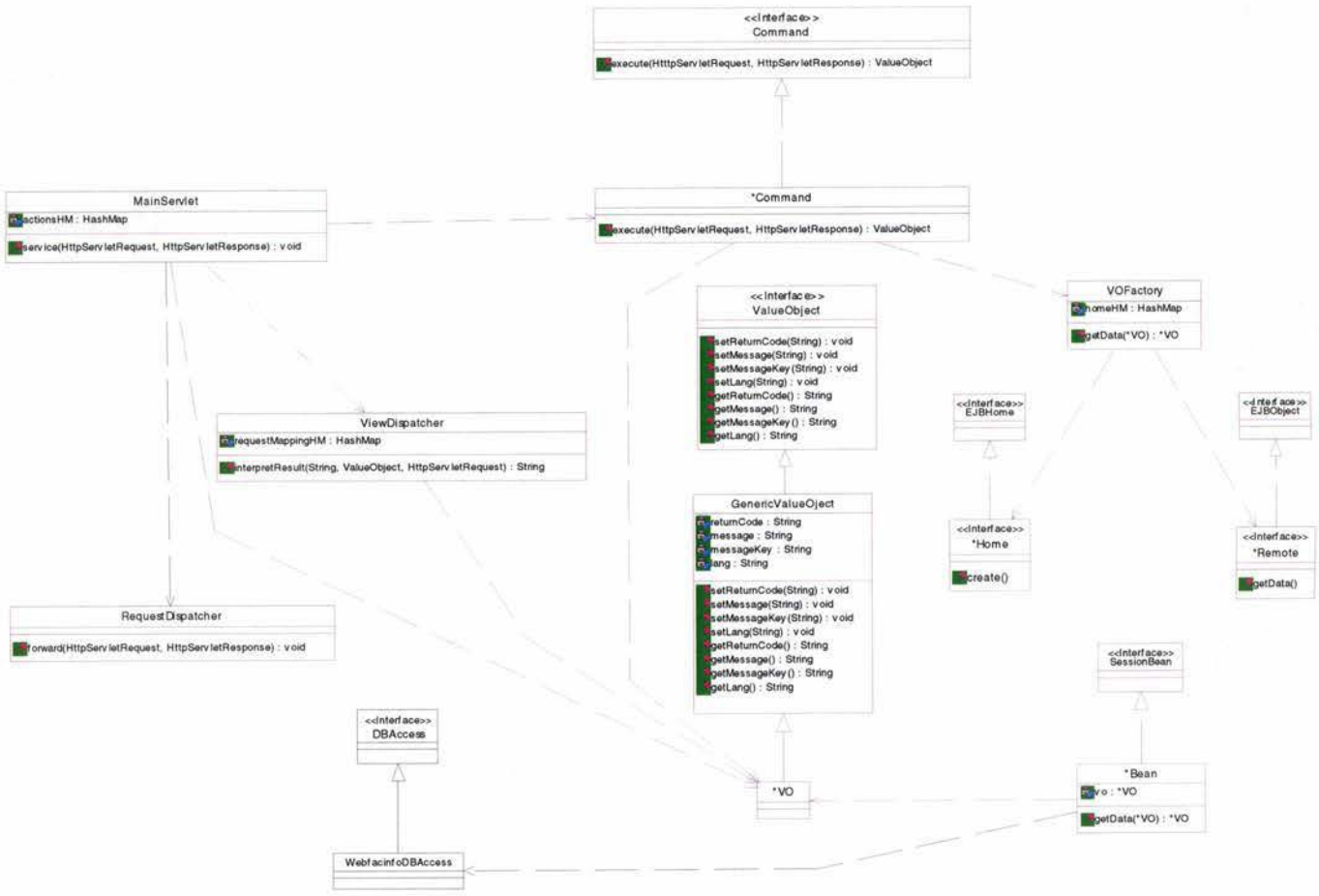


FIGURE 31 : CLASS DIAGRAM

9.5 Flow diagram

Le flow diagram permet de modéliser l'ordonnancement dans le temps des messages échangés entre les différents objets du système.
Ce diagramme permet également de visualiser le temps d'interaction de chaque objet, symbolisé par les rectangles verticaux situé sur la ligne du temps de chacun d'entre-eux. [Khawar Zaman Ahmed & Cary E. Umrish, 2001]

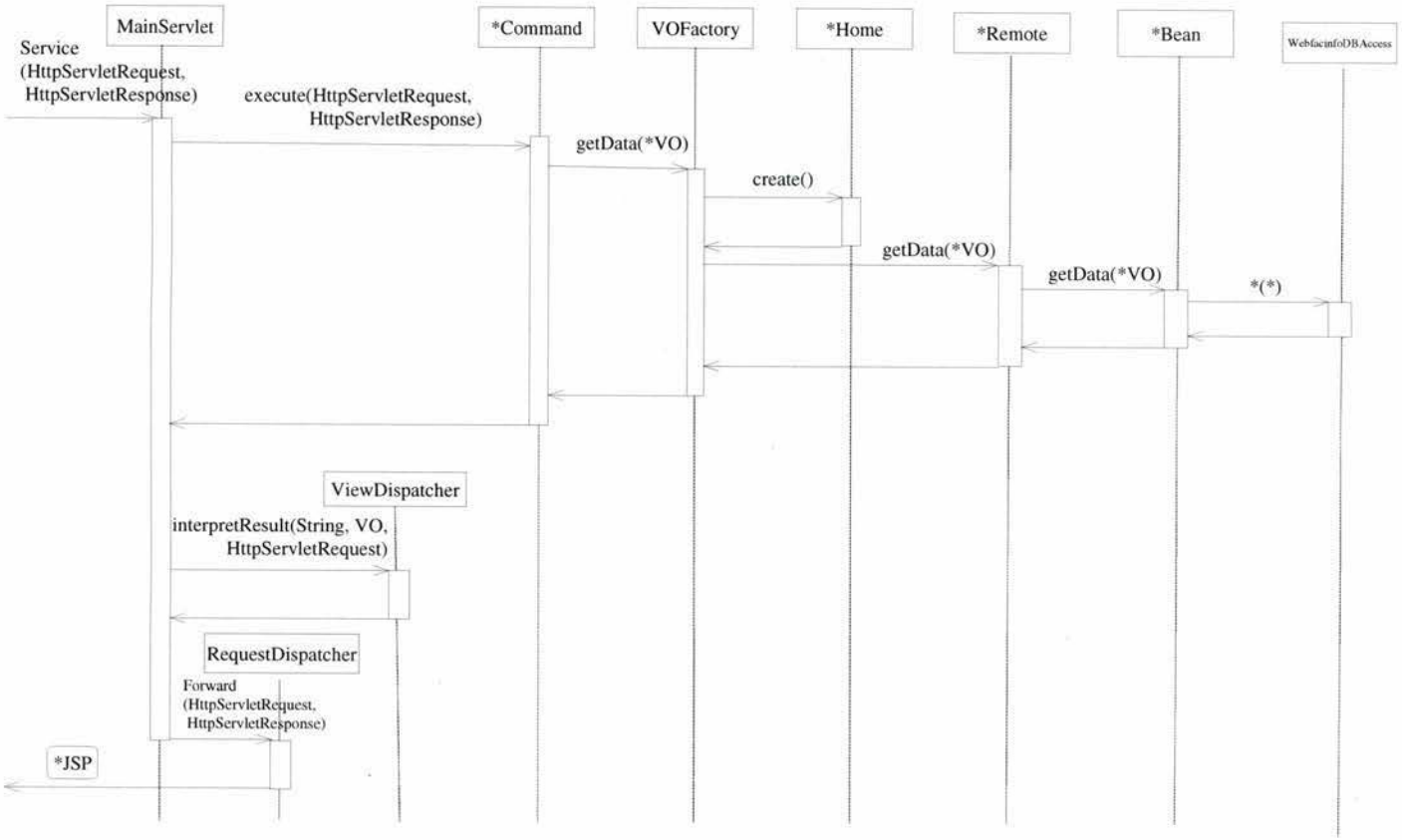


FIGURE 32 : FLOW DIAGRAM

10 PROTOTYPE (implementation)

10.1 Documentation technique

10.1.1 les packages

La découpe de l'application en package regroupe les différents objets en respectant leurs fonctions respectives au sein de l'application.

10.1.1.1 Le package fundp.jmc.command

Ce package contient les objets command.

10.1.1.2 Le package fundp.jmc.configuration

Ce package contient les objets permettant d'accéder aux fichiers de configuration.

10.1.1.3 Le package fundp.jmc.constant

Ce package contient la classe Constant regroupant les différentes constantes utilisées dans l'application.

10.1.1.4 Le package fundp.jmc.domain

Ce package contient les objets du domaine.

*10.1.1.5 Les packages fundp.jmc.ejb.*EJB*

Ces packages contiennent chacun les 3 classes (home, remote, bean) correspondant à un EJB, où * représente le nom spécifique de chaque bean.

10.1.1.6 Le package fundp.jmc.factory

Ce package contient les classes de type factory (VOFactory et DBAccessFacytory)

10.1.1.7 Le package fundp.jmc.persistence

Ce package contient les classes permettant l'accès à la base de donnée

10.1.1.8 Le package fundp.jmc.servlet

Ce package contient la servlet centrale
10.1.1.9 Le package fundp.jmc.view

Ce package contient l'objet ViewDispatcher

10.1.1.10 Le package fundp.jmc.vo

Ce package contient les value objects

10.1.2 la structure de l'application

10.1.2.1 le cascading style sheet

le cascading style sheet utilisé pour le layout de l'interface est localisé sous:

c:\webfacinfo\css\webfacinfo.css

10.1.2.2 le javascript

le javascript utilisé dans les pages jsp est localisé sous :

c:\webfacinfo\js\common.js
\test.js

10.1.2.3 les properties

le système est paramétrable à partir de 3 fichiers properties :

- un fichier de configuration :
  - c:\webfacinfo\properties\webfacinfo.properties
- 2 fichiers contenant des messages d'alerte applicatifs dans les 2 langues :
  - c:\webfacinfo\properties\fr.properties
    \en.properties

10.1.2.4 les images

les images utilisées pour l'interface utilisateur sont localisées sous :

c:\webfacinfo\images

10.1.2.5 les jsp

les pages jsp en français utilisées sont localisées sous :

c:\webfacinfo\ressource\fr*.jsp

les pages jsp en anglais utilisées sont localisées sous :

c:\webfacinfo\ressource\en*.jsp

10.1.2.6 les sources

les code sources sont localisés sous :

c:\webfacinfo\src\fundp\jmc\command
\configuration
\constant
\domain
\ejb*EJB
\factory
\persistence
\servlet
\view
\vo

10.1.2.7 les sources compilées

les sources compilées (.class) sont localisées sous :

c:\webfacinfo\fundp\jmc\command
\configuration
\constant
\domain
\ejb*EJB
\factory
\persistence
\servlet
\view
\vo

10.1.3 le serveur d'application

10.1.3.1 Localisation

Le serveur d'application BEA Weblogic est installé sous :

c:\bea\wlserver6.1

par défaut celui-ci intègre un JDK (Java Development Kit) sous :

c:\bea\jdk131

10.1.3.2 Déploiement WebFacInfo

[Joe Zuffoletto, 2002]

Le déploiement de l'application au sein du serveur d'applications, nécessite les opérations suivantes :

- La mise en place des descripteurs de déploiement

  - Pour les composants web
  - Pour les EJB

- La génération d'un fichier archive .jar pour chaque EJB
- La génération d'un fichier archive .war pour l'application proprement dite

Descripteurs de déploiement pour les composants web

Dans weblogic, la configuration d'une application web se fait par deux descripteurs de déploiement :

- Le fichier web.xml qui fait partie de la spécification J2EE et qui permet de paramétrer les composants web
- Le fichier weblogic .xml qui est optionnel et qui est utilisé pour configurer des fonctionalités spécifiques de weblogic

Ces deux fichiers sont localisés sous :

c:\webfacinfo\WEB-INF

Descripteurs de déploiement pour les EJB

De la même manière, les descripteurs de déploiement pour chaque EJB sont au nombre de deux :

- Ejb-jar.xml
- Weblogic-ejb-jar.xml

Ces fichiers sont localisé sous :

c:\webfacinfo\META-INF*EJB

Génération des fichiers *Bean.jar

Pour chaque EJB, un fichier *bean.jar est généré à partir des classes du bean et de ses descripteurs de déploiement. Une fois généré, ce fichier est déposé sous :

c:\bea\wlserver6.1\config\jmc\applications

au démarrage du serveur, les fichiers *bean.jar sont décompressés et automatiquement déployés par le serveur d'application.

Le schéma suivant illustre la manière dont ces fichiers sont générés :

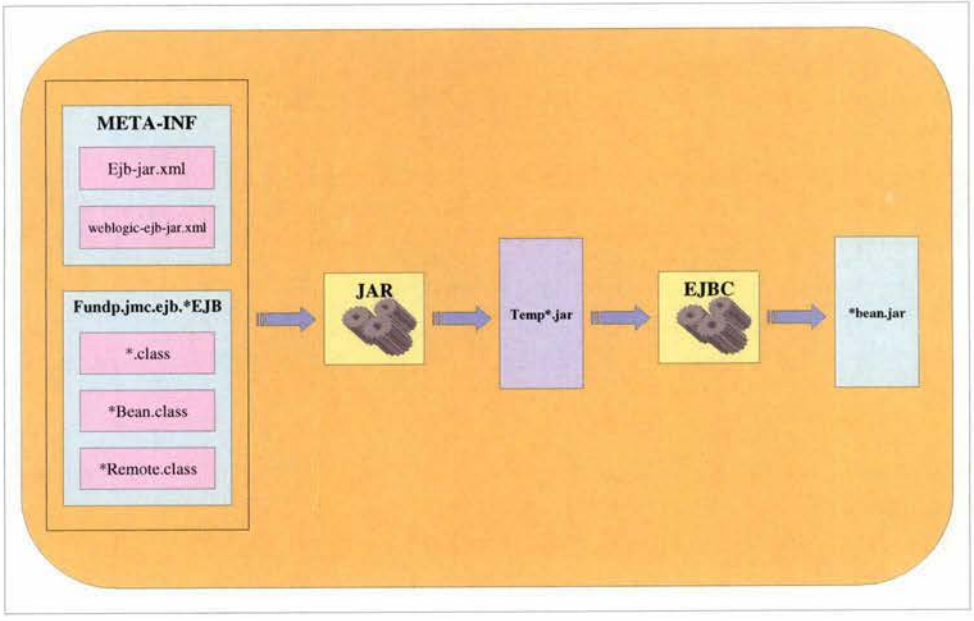


FIGURE 33 : EJB – FICHIERS ARCHIVES .JAR

ce schéma montre de gauche à droite :

- Les éléments nécessaires à la constitution de l'archive
- L'utilitaire JAR
- Un fichier .jar temporaire qui sert d'imput pour l'utilitaire EJBC
- L'utilitaire EJBC
- Le fichier *Bean.jar

Génération du fichier webfacinfo.war

Les différents éléments de l'application qui doivent être déployé dans le serveur d'application sont rassemblés dans un fichier archive webfacinfo.war. Ce fichier est généré à partir des répertoires de l'applications situés sous c:\webfacinfo

Une fois généré, ce fichier est déposé sous :

c:\bea\wlserver6.1\config\jmc\applications

au démarrage du serveur, le fichier webfacinfo.war est décompressé et automatiquement déployé par le serveur d'application.

Le schéma suivant illustre la manière dont ce fichier est généré :

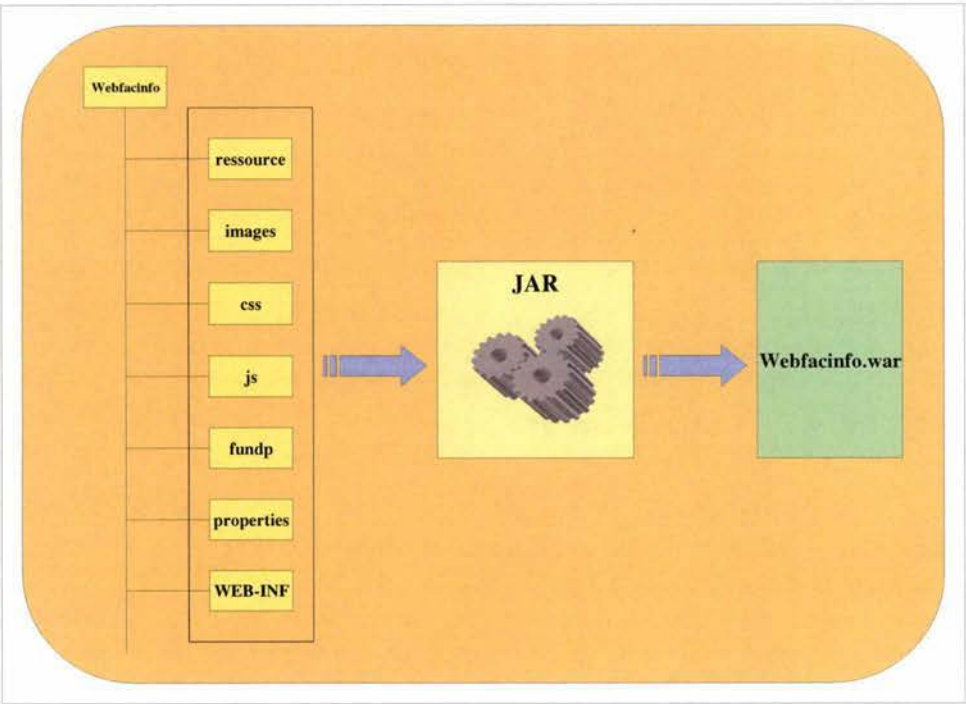


FIGURE 34 : WEBFACINFO.WAR

ce schéma montre de gauche à droite :

- Les répertoires à inclure dans l'archive :
  - Ressource qui contient les JSP
  - Images qui contient les images
  - Css qui contient les cascading style sheet
  - Js qui contient les javascripts
  - Fundp qui contient les .class
  - Properties qui contient les fichiers de configuration
  - WEB-INF qui contient natement les descripteurs de déploiement de l'application
- L'utilitaire JAR
- Le fichier wefacinfo.war

10.1.3.3 Administration

[Joe Zuffoletto, 2002]

sans entrer dans le détail, signalons enfin que l'administration de weblogic est assurée par une console accessible avec un navigateur web via l'URL :

http://serverName/console

cette console permet entre autre :

- La configuration d'une nouvelle application web
- L'ajout d'EJB
- L'ajout de servlets
- La paramétrisation du serveur web
- La paramétrisation de la sécurité
- Le tuning du serveur d'application
- La configuration du clustering en cas de montée en charge
- ...

Cette liste est loin d'être complète, pour de plus amples informations, le lecteur se reportera à l'ouvrage cité en référence.

10.1.4 accès à la base de données

Nous avons vu précédemment que la base de données développée dans le cadre de ce travail est un modèle tout à fait basique, assurant juste un modèle de données nécessaire et suffisant pour un fonctionnement correct de l'application.

Dans l'avenir, on pourrait très bien imaginer que le système repose sur des données issues d'une base de données différentes, voire de plusieurs DB réparties, ou de toute autre source de données.
L'idée est de fournir une architecture permettant une évolution vers un accès à d'autres sources de données à moindre prix. Cet aspect est illustré par le schéma ci-dessous.

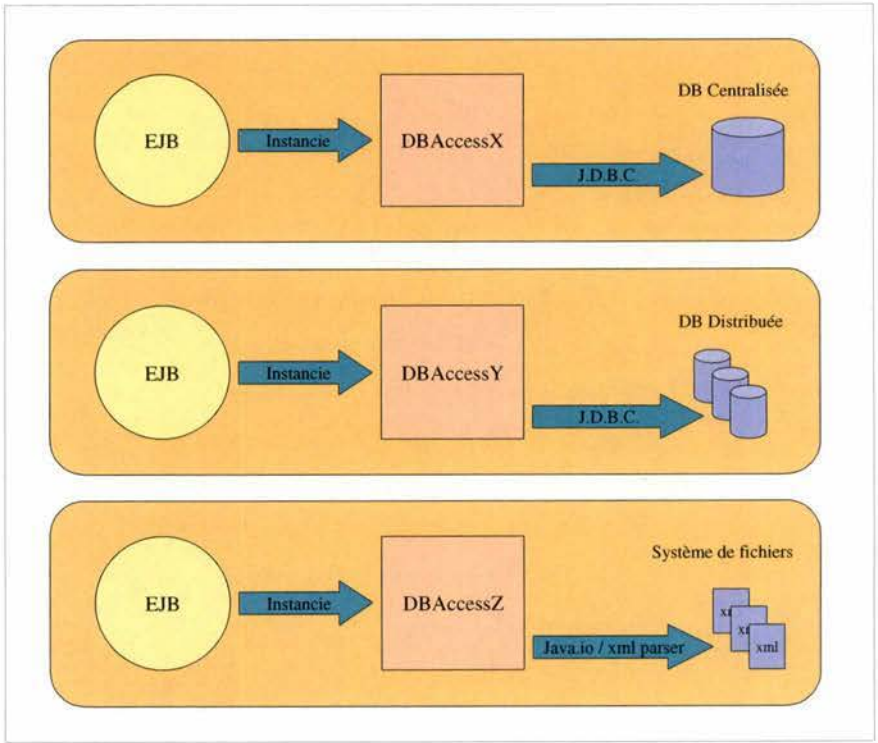


FIGURE 35 : ACCES A LA DB

dans cette optique, le schéma montre clairement que les EJB devrait instancier une classe spécifique d'accès aux données pour chaque cas.

A cette fin l'architecture mise en place fait intervenir l'interface DBAccess dont l'implémentation spécifique à chaque source de donnée est indiquée dans le fichier properties de l'application. Pour ce faire, on utilise une DBAccessFactory qui fourni l'instanciation de la classe correspondante, comme le montre le schéma suivant :

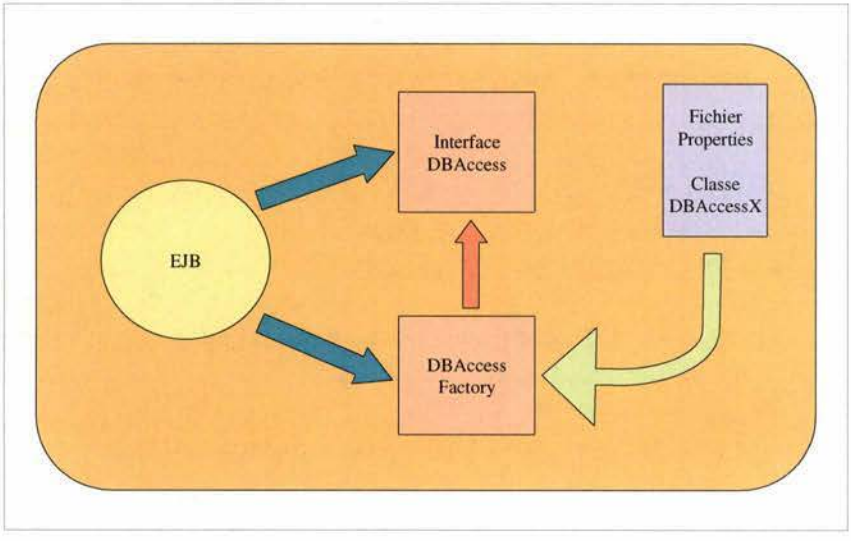


FIGURE 36 : CHOIX D'ACCES A LA DB

Ainsi, en cas de changement de source de données, il suffirait de :

- développer une nouvelle classe implémentant l'interface DBAccess
- fournir le nom de cette classe au niveau du fichier properties de l'application.

En effet, selon le schéma suivant, toute classe implémentant DBAccess peut aisément remplacer la classe actuelle WebFacInfoDBAccess .

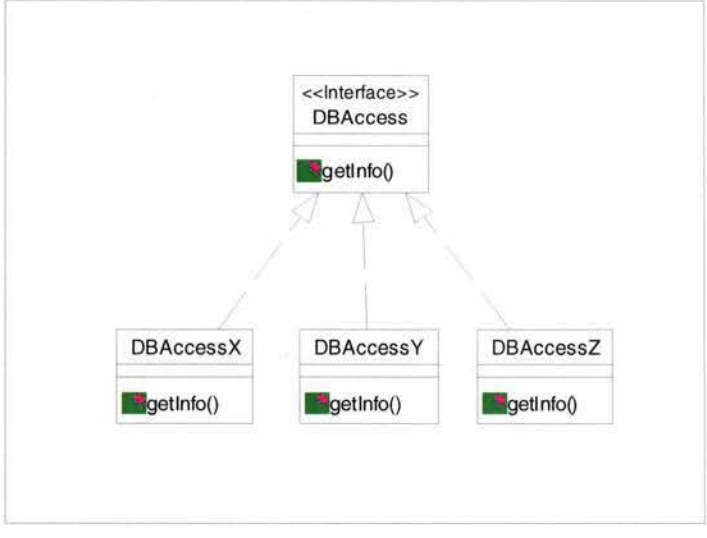


FIGURE 37 : ACCES A LA DB – CLASS DIAGRAM

10.1.5 Accès à l'application

L'application est accessible via un menu général avec l'URL :

http://MyServer/webfacinfo/user

ou via des menus personnalisés en fonction du type d'utilisateur (étudiant, secrétaire, λ) par le passage de paramètres :

http://MyServer/webfacinfo/user?User=stu
http://MyServer/webfacinfo/user?User=sec
http://MyServer/webfacinfo/user?User=other

dans le cas où l'utilisateur est un étudiant, le système peut prendre connaissance de son identité avec un paramètre supplémentaire :

http://MyServer/webfacinfo/user?User=stu&Login=jmcolle

il est à noter que le paramètre « user » correspond à la notion de groupe en terme de portail, et que le paramètre « login » correspond au UserName utilisé lors de la procédure d'authentification. Ce sont donc 2 données aisément récupérables au niveau du portail et donc qui peuvent être passées à l'application dans ce mode d'utilisation. Ainsi, un étudiant étant authentifié comme tel lors de son accession au portail (groupe « étudiant » et userName « jmcolle »), et qui souhaiterait consulter sa liste de cours personnelle, ne devrait plus s'authentifier au niveau de l'application, puisque celle-ci aurait pris connaissance de ces données lors du passage de l'URL.

10.2 Code source

Le code source, une version exécutable de l'application ainsi que les environnements nécessaires sont disponibles sur le CD-Rom fourni en annexe.

11 DEPLOIEMENT

11.1 WebFacInfo à l'Institut

L'intégration de l'application WebFacInfo au sein de l'infrastructure existante à l'Institut d'Informatique, dans le contexte de Portail Web est prévue. A ce stade, le prototype de cette application est terminé, fonctionne parfaitement et une démonstration de celui-ci est disponible en version locale sur l'environnement de développement. Le déploiement effectif dans l'environnement de production n'est pas encore effectué, et se fera ultérieurement.

Outre la mise en production effective du prototype développé, le principal attrait de ce déploiement est la démonstration et l'expérimentation de la portabilité d'applications développées en Java. En effet, le développement s'est effectué sur une plateforme Windows 2000, alors que l'infrastructure de production est constituée d'un serveur Solaris (Unix). Le serveur d'application BEA Weblogic et la DB Interbase existant tous deux en version windows et unix, l'essentiel du travail est l'installation et la configuration de ces deux environnements dans le monde unix, et ensuite, le déploiement et la configuration de l'application à proprement parlé dans ces environnements, ce qui nécessiterait vraisemblablement l'adaptation de nombreuses choses telles les sripts de démarrage de weblogic, la configuration des path et classpath, probablement une attention particulière au niveau du code (par exemple, les séparateurs dans les paths qui sont différents sous Windows et sous Unix), etc ....

Dans le cadre de ce futur déploiement, il nous semble nécessaire de présenter ici les principaux éléments du système afin d'assurer au lecteur une compréhension globale de l'infrastructure en rapport avec les besoins précédemment exprimés.

11.2 Infrastructure globale

Une vue globale de la connectivité du système est représentée par le schéma ci-dessous :

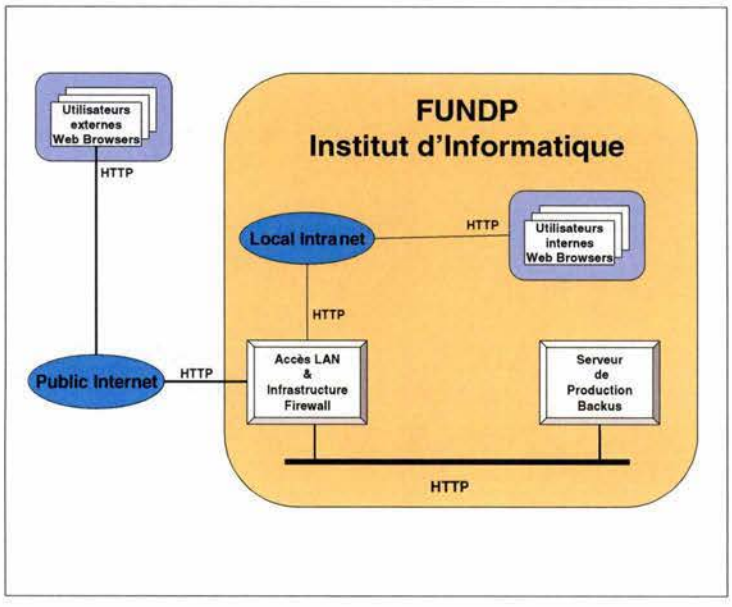


FIGURE 38 : INFRASTRUCTURE FUNDP

Ce schéma montre les principaux composants de système WebfacInfo :

- L'infrastructure du site WebFacInfo hébergé à l'Institut
- La connexion de WebFacInfo avec l'Internet public au travers de l'infrastructure firewall de l'Institut.
- Les utilisateurs externes communiquant avec WebFacInfo via l'Internet public
- Les utilisateurs internes communiquant avec WebFacInfo via l'Intranet de l'Institut

11.3 L'application et l'infrastructure réseau

La structure générale de l'application est représentée dans le schéma ci-dessous. Les principaux composants de l'application sont représentés selon les conventions du modèle TCP/IP utilisées pour la représentation des empilements de couches réseau. [Andrew Tanenbaum, 1999]

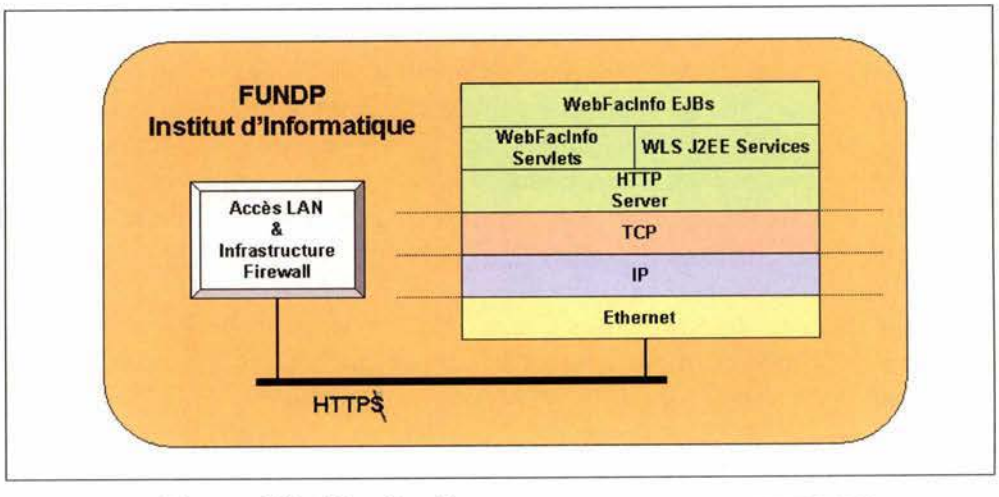


FIGURE 39 : WEBFACINFO ET INFRASTRUCTURE RESEAU

De gauche à droite et de haut en bas, le schéma montre :

- Le firewall et l'accès à l'Intranet à travers lesquels toutes les communications sont routées
- La couche physique assurée par l'infrastructure réseau de l'Institut
- La couche Ethernet
- La couche IP
- La couche TCP
- La couche application
    - Le serveur HTTP qui reçoit toutes les requêtes entrantes
    - La servlet principale qui interface le serveur HTTP
    - Les services J2EE fournis par le serveur BEA Weblogic
    - Les Enterprise Java Beans (EJB)

11.4 Configuration serveur

Les différents éléments software constitutifs du système, installés sur le serveur UNIX sont représentés dans le schéma ci-dessous :

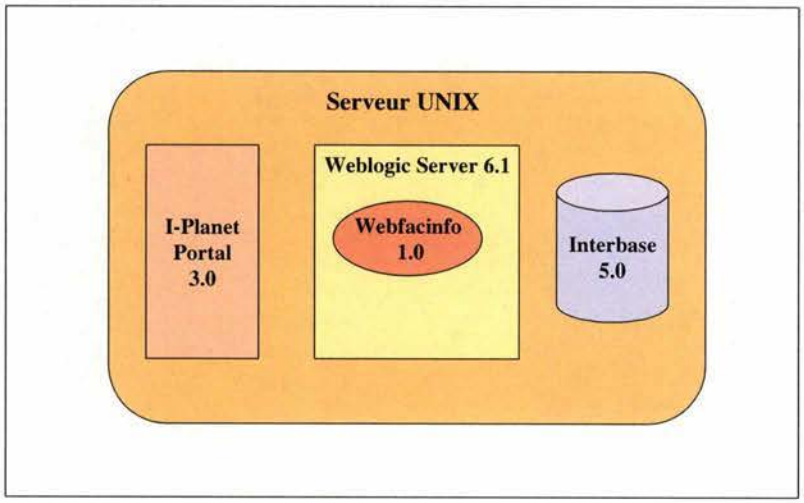


FIGURE 40 : CONFIGURATION SERVEUR

Les différents éléments représentés ici sont :

- I-Planet Portal 3.0
- Welogic Server 6.1
- WebFacInfo 1.0 (déployée dans WLS)
- Interbase 5.0

11.5 Les relations portail – WebFacInfo

Nous avons vu précédemment, que le portail s'adresse à un serveur de ressources pour délivrer le contenu à fournir à l'utilisateur sur base de sa requête. Dans le cas qui nous occupe, la ressource est l'application WebFacInfo.

Nous avons vu également que l'application WebFacInfo pouvait avoir 3 points d'entrée différents sur base de paramètres passés dans l'URL.

Il nous paraît utile d'expliquer brièvement quels sont les mécanismes mis en place pour assurer la liaison portail – WebFacInfo.

Le schéma suivant met en évidence ces mécanismes :

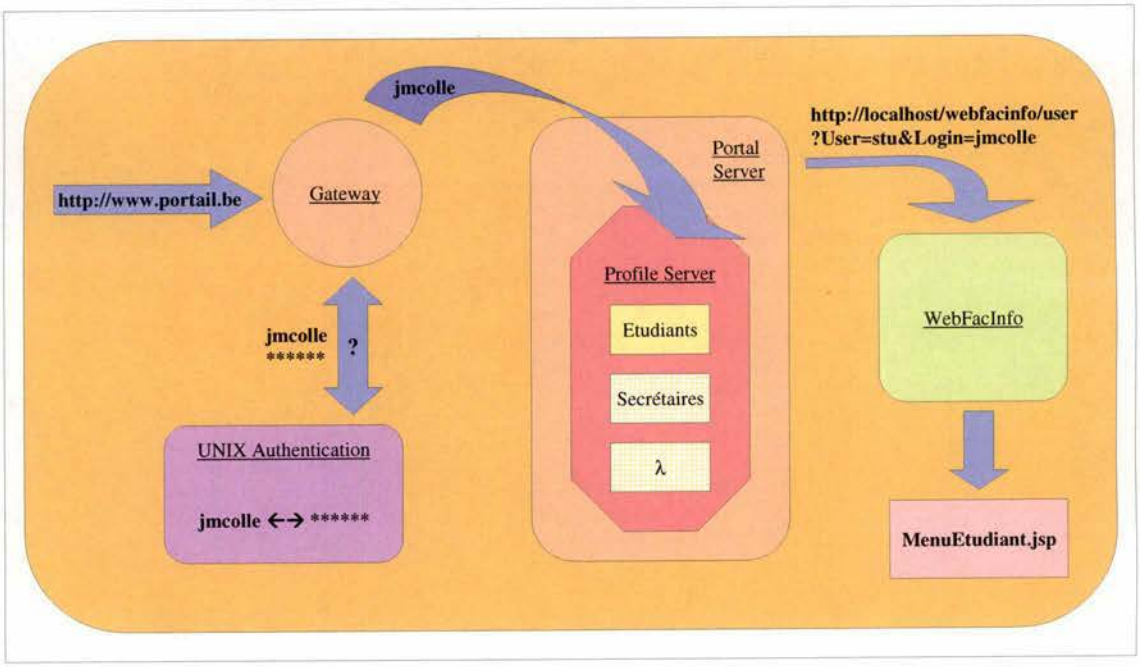


FIGURE 41 : RELATIONS PORTAIL – WEBFACINFO

Ce schéma met en évidence :

- La requête cliente, via l'URL générale du portail
- L'authentification par le gateway auprès du serveur d'authentification
- L'accès au serveur de profil sur base du login récupéré lors de la procédure d'authentification
- La redirection d'URL vers l'application WebFacInfo, avec les paramètres nécessaires.
- La page d'accueil correspondante servie par l'application

12 CONCLUSIONS

Voici venu le moment de faire le bilan de ce travail, et de voir si tous les objectifs énoncés ont été atteints.

Concrètement, nous avons :

- Une analyse des besoins mettant en évidence un grand nombre de fonctionnalités à mettre en place
- Une analyse fonctionnelle mettant en évidence des scénarios répondant à certaines de ces fonctionnalités
- Une architecture permettant de répondre à ces fonctionnalités, mais surtout, permettant d'implémenter rapidement de nouvelles fonctionnalités, en se basant sur le même modèle, autorisant de la sorte le développement modulaire du système.
- Une base de données, représentative du modèle de données, et opérationnelle.
- Un prototype opérationnel, implémentant les fonctionnalités retenues, et la documentation technique s'y rapportant.

Sur le plan théorique, nous avons :

- Une présentation détaillée de la plate forme J2EE
- Une approche du concept de Portail Web
- Une étude des solutions permettant l'authentification simplifiée
- Une présentation des perspectives de déploiement, et de l'intégration du prototype au sein de l'infrastructure existante

Sur le plan personnel, nous avons réalisé tous nos objectifs, en menant à bien ce projet, nous avons appliqué de manière concrète les différents concepts théoriques enseignés, nous avons appliqué une démarche scientifique pour aborder et présenter des concepts nouveaux, et nous sommes parvenus à transformer une vague idée subjective en un système opérationnel, documenté et argumenté.

En ce qui concerne les perspectives, on peut prévoir entre autre :

- Le déploiement effectif de « WebFacInfo » sur le serveur « backus »
- L'intégration effective de « WebFacInfo » au Portail de l'Institut
- L'augmentation du nombre de fonctionnalités accessibles
- La migration vers d'autres sources de données
- La mise en place de clustering au niveau de weblogic (en cas de montée en charge importante)

Mais, tout cela devra faire l'objet d'un autre travail, par un autre étudiant …

13 BIBLIOGRAPHIE

13.1 Livres

- Paco Gomez & Peter Zadrozny, *Java 2 Enterprise Edition with BEA Weblogic Server*, Wrox, Birmingham, United Kingdom, 2000.
- Xavier Rousseaux, *Les jésuites belges 1542-1992 450 ans de la Compagnie de Jésus dans les Provinces belges*, aesm, Bruxelles, Belgique,1992
- Bruce Powel Douglass, *Real-time design patterns: robust scalable architecture for real-time systems*, Addison Wesley, Boston, USA, 2002.
- Les Freed and Frank J. Derfler Jr, *Building the information highway*, Emeryville, USA, 1994.
- Andrew Tanenbaum, *Réseaux (3ième édition)*, Dunod, Paris, France, 1999.
- Subrahmanyam Allamaraju, Karl Avedal,Richard Browett, Jason Diamond, John Griffin, MacHolden, Andrew Hoskinson, Rod Johnson, Tracie Karsjens, Larry Kim, Andrew Longshaw, Tom Myers, Alexander Nakimovsky, Daniel O'Connor, Sameer Tyagi, Geert Van Damme, Gordan Van Huizen, Mark Wilcox & Stefan Zeiger, *Programmation avec J2EE*, Eyrolles, Paris, France, 2001.
- Khawar Zaman Ahmed, Cary E. Umrysh, *Developing Enterprise Java Applications with J2EE and UML*, Addison-Wesley, Boston, USA, 2001
- Grady Booch, James Rumbaugh, Ivar Jacobson, *The unified Modeling Language user guide,* Addison-Wesley, Boston, USA, 2000
- Joe Zuffoletto, *BEA Weblogic Server, Bible,* Hungry minds, New York, USA, 2002

13.2 Sites Web

- Dominique Revuz, Round Robin (tourniquet), http://www-igm.univ-mlv.fr/~dr/NCS/node97.html (last modified February 2, 1998) (Date of access 28/4/2003)
- Dominique Liard, Internet - historique, http://www.infini-fr.com/Sciences/Informatique/Reseau/Internet/historique.html (last modified September 11, 2000) (Date of access 23/4/2003)
- Eric Larcher, Internet, historique et utilisation (3ième édition), http://www.larcher.com/eric/guides/ihu/ (last modified December, 1998) (Date of access 23/4/2003)
- Bernard Truffaut, Les portails, http://heitml.megadyne.fr/documents/Bernard_TRUFFAUT.ppt (last modified April, 2001) (Date of access 9/5/2003)
- Allan Ant, Password managament, Single sign-on and authentication management infrastructure, www.gartner.com (January 7, 2003) Date of access 15/3/2003)
- Gene. Phifer, Next for Portal products: the portal apllication server, www.gartner.com (October 16, 2001) (Date of access 15/3/2003)
- Gene. Phifer, Portals in 2002: a year of major change, www.gartner.com (December 12, 2001) (Date of access 15/3/2003)
- Gene Phifer, A vertical look at portals, www.gartner.com (January 17, 2002) (Date of access 15/3/2003)

- Gerhard Schiefer, Anne Catharina Kreuder, Vertical and horizontal information portals: cooperation models for sector and chain information services, http://www.ifama.org/conferences/2001Conference/Papers/Area%20III/Schiefer_Gerhard.PDF (last modified unknown) (Date of access 11/5/2003)

13.3 Autres références

- Sun Microsystems, iPlanet Portal Server 3.0, Administration Guide, 2000

Valves électroniques et Portail Web à l'Institut, analyse, prototype, aspects théoriques.

Jean-Marc COLLET
LIHD 2
Défense de mémoire - juin 2003

Plan de la défense

- Objectifs
- Contenu du mémoire
- Prototype *WebFacInfo*
- Conclusions
- Q & R
- Démonstration prototype

Objectifs

- Synthèse de l'apprentissage
  - *Un maximum de concepts*
- Aptitude à conduire un "projet" tout au long de son cycle de vie
  - *Un maximum de rôles*
- Prototype
- Approche théorique de concepts neufs

Contenu du mémoire

- Analyse et développement du prototype
  - *Analyse des besoins*
  - *Analyse fonctionnelle*
  - *Analyse technique*
  - *Conception DB*
  - *Implémentation & tests*
  - *Déploiement (vue théorique)*
- Aspects théoriques
  - *La plate-forme J2EE*
  - *Portails – iPlanetPortal*
  - *Authentification simplifiée*

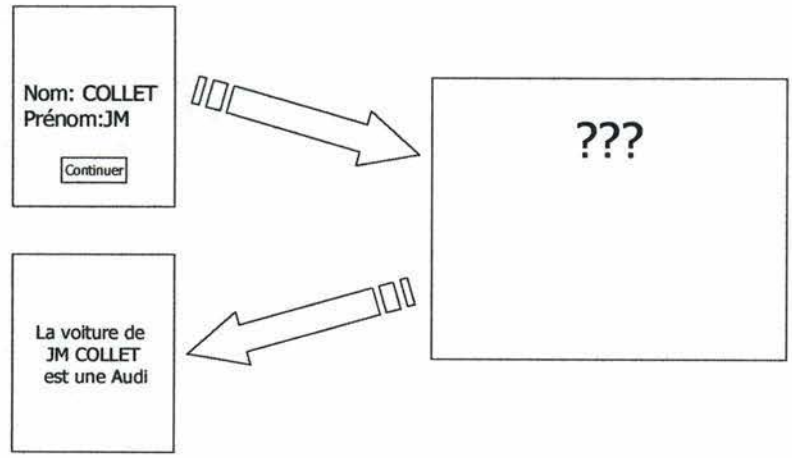
WebFacInfo analyse des besoins

- Analyse de l'existant
  - *Interviews*
- Périmètre du projet
  - *Les buts*
- Exigences
  - *Sur le système*
  - *Sur les acteurs*
  - *Exigences non fonctionnelles*

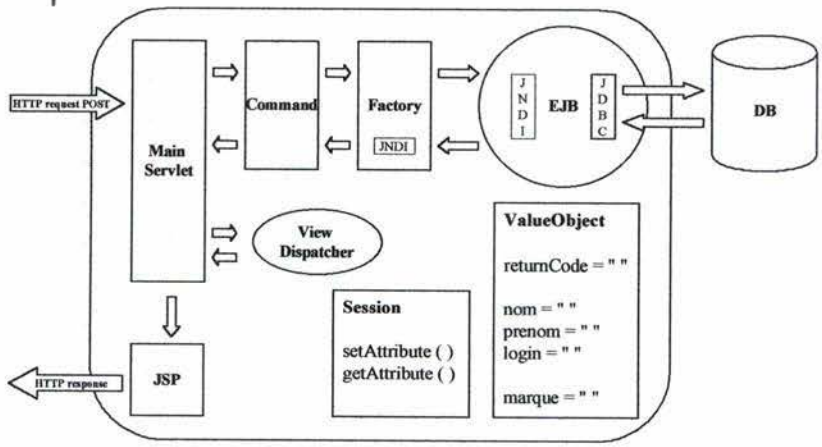
WebFacInfo analyse fonctionnelle

- Use case
  - *Elaboration de scénarios*
- Schémas de robustesse
  - *Organisation en modules fonctionnels*

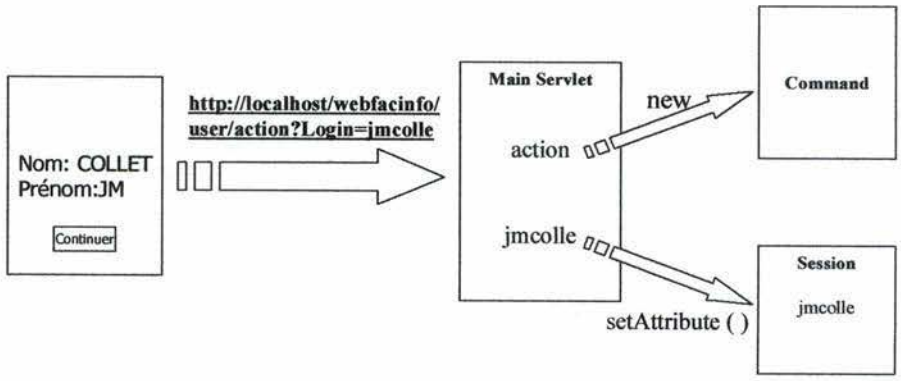
WebFacInfo
analyse technique (I)



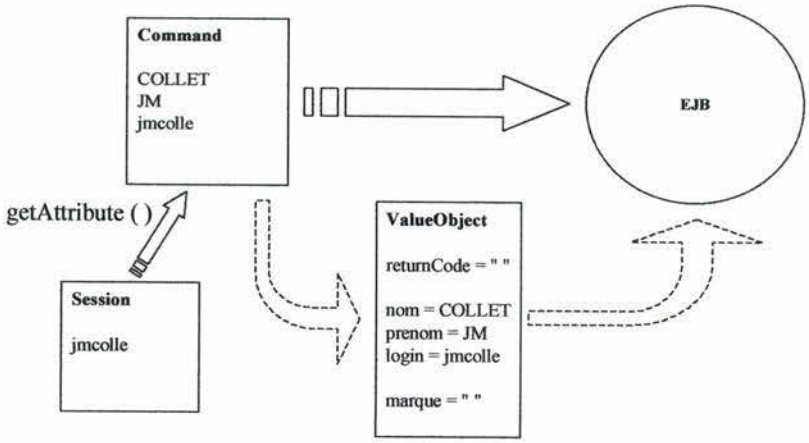
WebFacInfo
analyse technique (II)



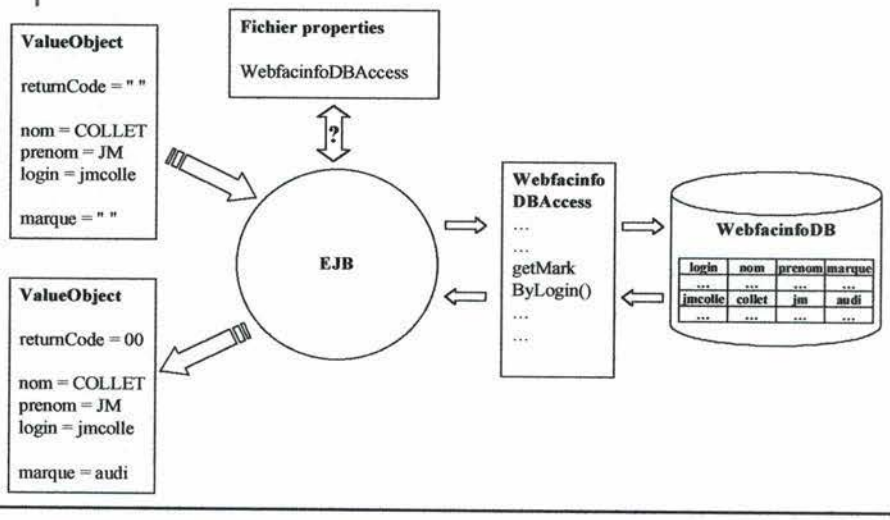
WebFacInfo
analyse technique (III)



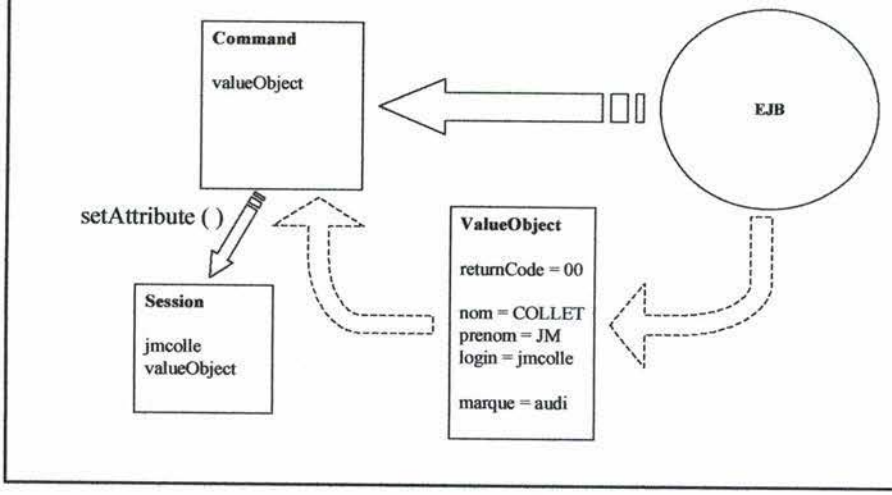
WebFacInfo
analyse technique (IV)



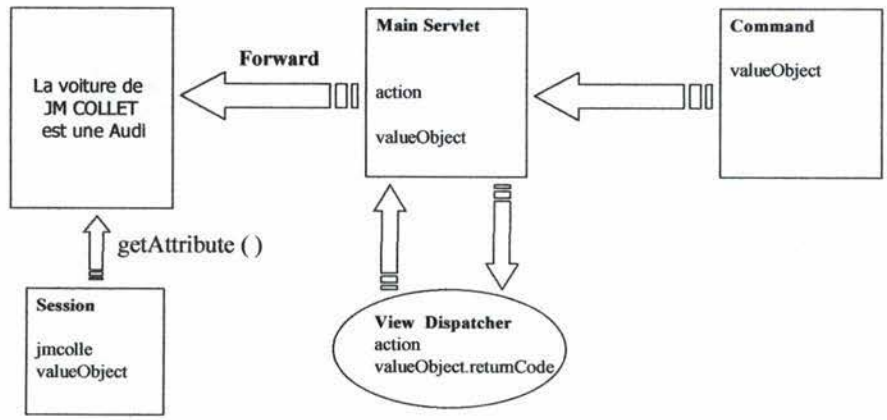
WebFacInfo
analyse technique (V)



WebFacInfo
analyse technique (VI)



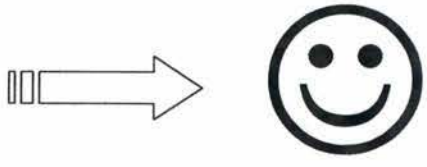
WebFacInfo analyse technique (VII)



Conclusions

- Tous les objectifs énoncés sont atteints
- Portabilité du système
  - ≠ *plateformes (java)*
  - ≠ *serveurs d'application*
    - Standards J2EE
    - Limitations (EJB)
- Extensibilité du système
  - *Fonctionnalités supplémentaires*
  - *Montée en charge*

Q & R

??? 

Démonstration *WebFacInfo*

- Secrétaire
  - *Consulter liste d'étudiants*
  - *Ajouter étudiant*
- Etudiant
  - *Consulter sa liste de cours*
  - *Demander une dispense*
- Secrétaire
  - *Accepter une dispense*
- Etudiant
  - *Consulter sa liste de cours*
  - *Commander ses syllabi*
- Secrétaire
  - *Gestion des syllabi*

******

- Mise en évidence de:
  - *Interactions du système*
  - *Messages informatifs à destination de l'utilisateur*
  - *Changement de langue*
  - *Rapidité & Facilité d'utilisation*
  - *Look & feel compatible avec celui de l'Institut*