Diffusion :

- M. FOFANA
- documents du projet

date de diffusion(s) : ???

Dernière impression le : 02/12/04 14:55

REPUBLIQUE DE GUINEE

Travail justice solidarité

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET
DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE GAMAL ABDEL NASSER DE CONAKRY

CENTRE INFORMATIQUE

Année universitaire : 1999-2000

Option : Filière longue

MEMOIRE DE FIN D'ETUDES SUPERIEURES

Présenté et soutenu le ___/___/___

CANDIDAT : Mamadi Saran FOFANA

REPUBLIQUE DE GUINEE
Travail justice solidarité

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET
DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE GAMAL ABDEL NASSER
DE CONAKRY

CENTRE INFORMATIQUE

Année universitaire 1999-2000
Promotion : 35 ème

Option : Filière longue

PROJET DE FIN D'ETUDES

THEMES
<<Conception d'un site web interactif(d'architecture 3-tiers) interfacé à une base de données>>

Présenté et soutenu le ___/___/___

Candidat
Mamadi Saran FOFANA

Directeur Adjoint
Dr. Ibrahima CAMARA

Directeur du Centre
Dr. Binko Mady TOURE

Vice Recteur
chargé des études
Prof. Amara CISSÉ

Recteur de l'université
Dr. Ousmane SYLLA

Introduction :

L'objectif premier d'un système d'informations est de permettre à plusieurs utilisateurs d'accéder aux mêmes informations. La technologies du web en pleine expansion depuis quelques années répond parfaitement à cette exigence.
C'est dans cette optique que le projet Pêche Ecologique en Guinée (PEG) a décidé la mise en place d'un centre d'informations sur la pêche en guinée. Ce sera un système d'informations qui rassemblera puis restituera les connaissances disponibles sur le secteur halieutique guinéen à travers un site web.

C'est une idée du chef de projet Jean Lefur dont la réalisation a fait appel à une équipe composée d'ingénieurs et de techniciens: Loïc Thibaut (ingénieur qualité) Mamadi Fofana(moi-même) Misako Ito (durant 6 semaines) Salif Sylla(pour la conception de la base données) , Souleymane Diakité(XSL et HTML), Cheick Ba (datamining).

Sa construction débuta début 2001 et à ce jour deux versions avec plusieurs variantes ont déjà été développées, testées puis déployées avec succès.

J'ai été impliqué dès le départ dans l'ensemble des activités de conception et de réalisation du logiciel et dans ce rapport j'expose les grandes étapes de sa construction, de l'analyse des besoins jusqu'au déploiement.

Remerciements :

Je tiens avant tout à adresser mes sincères remerciements à M. Jean Lefur le coordonnateur du projet Pêche Ecologique en Guinée, à M. Loïc Thibaut à qui l'on doit l'architecture du logiciel et qui prit une part active dans l'implémentation, au personnel du CNSHB pour leur accueil et leur disponibilité ainsi qu'a l'ensemble de mes professeurs avec qui j'ai pu acquérir les bases scientifiques et informatiques qui m'ont permis de prendre une part active dans ce travail.

Table des matières

PRÉSENTATION DU CNSHB ET DU PROJET PEG 7

CAHIER DES CHARGES UTILISATEURS 8

ANALYSE DES BESOINS 8
SPÉCIFICATIONS GLOBALES 12
Architecture technique 12
Méthodes, formalismes et outils 13

CAHIER DES CHARGES DE LA REALISATION 19

CONCEPTIONS ARCHITECTURALE ET DETAILLEE 20
La couche Data 21
La couche Business logic 21
La couche Présentation 22
Modelisation UML 26
La base de données du centre d'informations 33

PROGRAMMATION, INTÉGRATION ET TEST 34

JAVA 34
XSL 34

RÉSULTATS 34

LE DÉPLOIEMENT 35

QUELQUES DIFFICULTÉS RENCONTRÉES 36

AMÉLIORATIONS POSSIBLES 37

ANNEXES 38

LES FICHIERS DE FORMATAGE XSL 38
Pour l'affichage d'une Information 39
Pour l'affichage d'une liste d'informations 41
Pour l'affichage des instances d'un descripteur 42
Pour l'affichage d'une liste de descripteurs(disponible seulement dans la version 1.1) 42

Présentation du CNSHB et du projet PEG

Le Centre National des Sciences Halieutiques de Boussoura(CNSHB) est un établissement public à caractère scientifique, placé sous la tutelle du Ministère de la pêche. Sa mission est de contribuer au développement de la pêche maritime en Guinée par une meilleure connaissance et évaluation des ressources halieutiques. Il est en étroite collaboration avec l'Institut de Recherche pour le Développement (IRD) basé à Montpellier en France qui lui apporte son appui technique. C'est dans le cadre de cette coopération qu'a été initié le projet Pêche et Ecologie en Guinée (PEG).

Le projet PEG qui débuta en septembre 2000 et qui est mené par des chercheurs de l'IRD et du CNSHB a pour objectif d'établir les modalités(compétences, méthodes, outils) d'un développement durable de la pêche maritime guinéenne. Ce projet est fondé sur l'usage respectueux des écosystèmes marins afin d'éviter la surexploitation et donc l'extinction des ressources halieutiques guinéennes. Il consiste notamment à préserver les capacités productives de l'écosystème marin et à développer au sein de ce secteur une connaissance sur les effets de l'activité humaine.

Les principaux volets d'activité de ce projet sont l'acquisition de la connaissance sur le secteur de la pêche, la modélisation de l'écosystème marin et de l'exploitation guinéenne, qui sera restituée sous forme d'un logiciel de simulation. Le modèle adopté est un modèle multiagents où chaque information est représentée sous forme d'un agent dans un environnement Java, et cette modélisation sera effectuée à partir des informations qui seront rassemblées dans un centre d'informations qui fait l'objet de mon présent thème de soutenance.

Cahier des charges utilisateurs

Analyse des besoins

Première activité dans un processus de développement elle a pour but d'éviter de développer un logiciel non adéquat. Après entretien avec le chef de projet Jean Le Fur (l'initiateur du centre d'informations) il en est ressortie une description du futur logiciel que voici :

Il s'agit d'un centre d'informations générique **(CI)** qui rassemble les connaissances disponibles sur le secteur des pêches en Guinée, il doit être :

- universel, pour constituer une plate-forme de discussion commune à tous les acteurs
- robuste, c'est à dire susceptible de s'adapter aux évolutions du secteur et des questions posées à son sujet ou, autrement dit, robuste à l'alimentation. On considère que les objectifs seront atteints s'il est encore utilisé 5 ans après le projet.

> Le travail effectué pendant le projet vise plus à concevoir *(i)* la plate-forme générique et *(ii)* une procédure rigoureuse d'alimentation du CI qu'à *(iii)* l'alimentation exhaustive du CI avec les connaissances disponibles. La réalisation des points *(i)* et *(ii)* associée aux autres activités du projet devant permettre de mettre en place l'environnement nécessaire à ce que le CI puissent continuer à être alimenté après le projet.

- Le CI doit pouvoir héberger et restituer des informations de toutes natures mais répondant à la définition de l'information retenue dans le projet :

Information : Assemblage de données *codées* procurant un *renseignement*[1] sur le domaine d'étude.

Notes :

- *le codage*[2] *donne la possibilité de conserver, traiter et communiquer l'information*
- *le domaine d'étude rassemble ici tout ce qui concerne le secteur des pêches en Guinée : environnement, ressource, pêche, commercialisation, consommation, gestion, recherche.*

Le code[3] utilisé doit assurer une restitution générique des informations quelles qu'elles soient. On considèrera plusieurs modalités de codage telles que chiffres, tableaux, graphes, cartes, texte, photos et représentations individu centrées.
De cette façon, une information donnée pourra être codée de plusieurs façons et faire alors l'objet de plusieurs modes de restitution (dans la mesure des formes de restitution disponibles dans le CI pour l'information considérée).

[1] renseignement : indication, information, éclaircissement donnés sur quelqu'un, quelque chose.

[2] codage : action d'appliquer un code pour transformer un message, des données en vue de leur transmission ou de leur traitement

[3] code : système conventionnel, rigoureusement structuré, de symboles et de règles combinatoires, intégré dans le processus de la communication. (ex : *Code gestuel, Code de la langue*)

L'information dans le CI sera donc décrite par un ensemble de descripteurs (qui correspondent aux champs d'une base de données qui lui est associée) : dimension, date, lieu, composants, groupes, mots-clés, sujet, titre, source, niveau de confiance, support.
l'utilisateur du CI a certaines prospectives/perspectives qui l'intéressent plus que d'autres. On propose trois entrées possibles dans le centre d'information pour l'utilisateur :

a. L'utilisateur progresse vers l'information recherchée en affinant sa demande. Cela correspond aussi à :
   - Naviguer dans des chapitres et des sous-chapitre ou encore à :
   - Une recherche dirigée dans les différentes dimensions (économiques, etc.) du secteur (voir module 21.typologie et 22.).
b. navigation hypertexte : l'utilisateur accède à des informations liées à d'autres en fonction d'affinités sur des thèmes communs (composants du secteur, coordonnées spatiales, temporelles, descripteurs clé, support, dimension)
c. recherche par descripteur (utiliser un moteur « freeware »)

Le renseignement fourni par une information donnée appartiendra à une dimension particulière du (selon laquelle on perçoit le) domaine telle que par exemple : Sociale, Ecologique, Economique, Gestion, Institution, Ressource, Technologique, Environnement, Halieutique, Hommes, Information, Production, Spatiale, Usage. Les dimensions traduisent une typologie générique et consensuelle du secteur :

L'utilisation du logiciel qui s'apparente surtout à une navigation dans le domaine de connaissance représenté se fera par le biais d'un browser hypertexte type Navigator ou Internet Explorer. On passe d'une page à une autre en cliquant sur les divers liens présents comme illustré dans cette maquette de quelques cas d'utilisations.

SOMMAIRE

GENERALITES

- Politique sectorielle
- Caractéristiques générales du secteur

LA FILIERE PECHE EN GUINEE

- L'environnement marin
- Les espèces :
  - non exploitées (la faune marine)
  - exploitées (la ressource)
- Les flottilles de pêche (l'exploitation)
- Les débarquements (la production)
- La transformation
- La commercialisation
- La consommation

- Technologie des équipements

7 Informations trouvées sur Dimension Exploitation

- Délimitation des zones fréquentées par les engins de pêche artisanale (cartes des zones de pêche)
- Exemple de stratégie d'accès à l'espace et à la ressource des pêcheurs à la palangre. (savoir et pratique de pêche)
- PA - rendements par préfecture de 1995 à 2000 (en kg par jour de mer)
- PA - rendements par type d'engin de pêche de 1995 à 1999 ( )
- PI - activité des chalutiers en Guinée (taux d'inactivité des chalutiers (période 1995-2000))
- PI - activité des chalutiers en Guinée (taux de présence des chalutiers sur une période de 72 mois (1995-2000))
- PI - activité des chalutiers en Guinée (nombre de traits de chalut réalisés mensuellement par bateau entre 1995 et 2000)

PA - rendements par préfecture de 1995 à 2000
en kg par jour de mer

En pêche artisanale, le rendement moyen par jour de mer est estimé à 150 kg/j. Ces rendements varient considérablement selon les cinq zones statistiques ou préfectures. Ils sont :

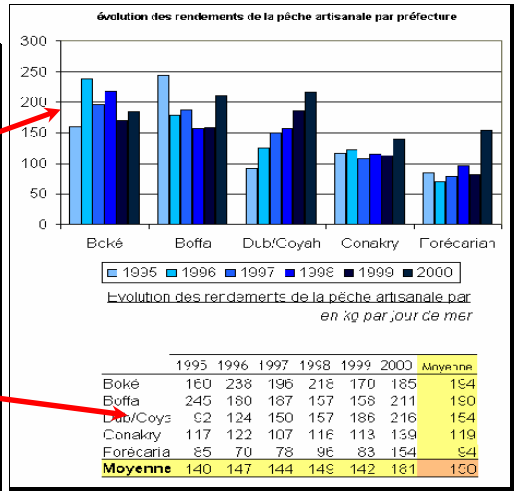
- élevés à Boké et à Boffa variant entre 160 à plus de 200 kg/jour,
- faibles à Forécariah (moins de 100 kg/j);
- stables (relativement) à Conakry entre 115 kg/j
- en hausse à Dubréka, passant de 72 kg en 1995 à 216 kg/j en 1999.

(NOTE: Les données de l'année 1999 sont relativement peu fiables).

graphique tableau

- dimension: Exploitation
- mots-clés: 1995-2000, littoral guinéen, pêche artisanale, rendement, jour de mer, préfecture, ,
- thèmes: activité, circonscription, flottille, mesure ou indicateur,

source:CNSHB (1996-2002). Bulletins statistiques des pêches, 1-6, Centr. Nat. Sci. Hal. Boussoura, 1996-2002.
information: n°0191 - mise à disposition: 2002-04-07 - rédacteur de l'information: Youssouf Hawa Camara (CNSHB) - niveau de confiance: moyen



Evolution des rendements de la pêche artisanale par préfecture
en kg par jour de mer

	1995	1996	1997	1998	1999	2000	Moyenne
Boké	160	238	196	218	170	185	194
Boffa	245	180	187	157	158	211	190
Dub/Coya	62	124	150	157	186	216	154
Conakry	117	122	107	116	113	139	119
Forécaria	85	70	78	96	83	154	94
Moyenne	140	147	144	149	142	181	150

Recherche par sommaire
Recherche par date
Recherche par lieu
Recherche par mot-clé
Recherche par type d'information

type de représentation :

- carte
- graphique
- page web
- photo
- schéma
- tableau
- texte

fiabilité des informations:

- 100%
- bon
- moyen
- faible
- inconnu

source des informations
rédacteur des informations

5 Informations trouvées sur support schéma

- Caractérisiques biologiques des crabes consommés sur le littoral guinéen ( )
- Distribution du baliste-cabri (Balistes carolinensis) sur le plateau continental guinéen (invasion de la ZEE par l'espèce dans les années 1980)
- Présentation et inconvénients du fumage sur banda ( )
- Régime alimentaire des requins (place des requins dans la chaîne trophique)
- Stratification des espèces benthiques de la mangrove guinéenne (environs de Conakry)

14 Informations trouvées sur groupe pêche artisanale

- Captures comparées des pêches artisanale et industrielle de 1995 à 2000 (production de la pêche maritime)
- Captures de la pêche maritime par groupe d'espèces de 1995 à 2000 (pélagiques, démersaux et dive poissons)
- Caractéristiques des centres de débarquement de la pêche artisanale (données 1998)
- Construction d'une pirogue en béton armé (débarcadère de Dixinn 3, sept-oct. 2001)
- Délimitation des zones fréquentées par les engins de pêche artisanale (cartes des zones de pêche)
- Eléments de réglementation du secteur de la pêche artisanale ( )
- PA - débarquements mensuels moyens de 1995 à 1999 (pélagiques et démersaux)
- PA - effort et capture de 1995 à 2000 (évolution annuelle)
- PA - relation entre les captures et le nombre de jours de mer (évolution mensuelle moyenne)
- PA - rendements par préfecture de 1995 à 2000 (en kg par jour de mer)
- PA - rendements par type d'engin de pêche de 1995 à 1999 ( )
- Produits de pêche mis sur le marché guinéen de 1997 à 2000 (résultats mensuels pour PA, PI, importations par mer et par terre)
- Profils comparés des captures des pêcheries artisanales et industrielles (par catégorie commerciale)
- Répartition de la pêche artisanale (localisation et enclavement des débarcadères)

Distribution du baliste-cabri (Balistes carolinensis) sur le plateau continental guinéen
invasion de la ZEE par l'espèce dans les années 1980

L'invasion du baliste a commencé dans le golfe de Guinée en 1972 et a atteint le Sénégal en 1978. En Guinée, la population a augmenté graduellement jusqu'en 1985. Le stock représentait alors 80% (estimation de 1984) de la biomasse de toutes les espèces démersales de poissons sur le shelf guinéen.

A cette époque, la répartition bathymétrique observée est irrégulière : le balise vit dans les profondeurs de 10-100 m; Sa plus grande concentration est observée entre 30-60 m.

Il n'est plus actuellement rencontré qu'occasionnellement dans la ZEE guinéenne.

Le baliste cabri est une espèce pélagique grégaire qui se nourrit principalement d'organismes benthiques. Il peut atteindre 60cm de long.

schéma carte

- dimension: Ressource
- mots-clés: 1984, plateau continental, baliste, Balistes carolinensis, répartition, poissons, stock, ,
- thèmes: mesure ou indicateur, organisme marin,

source:Bondar, A. et G. Zouev (1985). Les ressources en poisson de la zone économique exclusive guinéenne et les perspectives de leur exploitation. Paris Nathan, 1985, 78p
information: n°0227 - mise à disposition: 2003-03-10 - rédacteur de l'information: Bangali Kaba (Cerescor) - niveau de confiance: bon

Schéma de quelques cas d'utilisations du CI

Il faut signaler qu'il existait déjà un prototype du centre d'informations qui avait été développé en utilisant des techniques et une démarche simpliste. Ce prototype ne possédait pas toutes les fonctionnalités du futur logiciel à développer mais donnait une idée assez précise de son fonctionnement**.**

Spécifications Globales

Après l'analyse des besoins qui a permis de recenser les aspects pertinents de l'environnement du futur système c'est à dire son rôle et sa future utilisation s'en est suivi l'étape de spécification globale. Le but de cette activité est d'établir une première description technique du futur système. Elle a été principalement effectuée par Loïc Thibaut l'ingénieur qualité de l'équipe de réalisation.
Cette spécification à permis d'identifier l'architecture technique du futur logiciel, les choix technologiques ainsi que les outils logiciels nécessaires.

Architecture technique

Le Centre d'informations est un site web[4] interactif connecté à une base de données et générant son contenu de façon dynamique.
Il présente une interface client au sein d'un navigateur dans laquelle l'utilisateur fait des requêtes et obtient de façon dynamique les résultats en fonction de ses choix. En effet le document HTML correspondant à un lien hypertexte [5]du centre d'informations est construit au moment même où l'on clique sur le lien. Le document est envoyé au client au fur et à mesure de sa construction sans jamais être stocké dans un fichier. Ceci est réalisé à l'aide de *liens exécutables*. Le client indique le nom d'un fichier, toujours à l'aide d'une URL[6], non pour en recevoir le contenu mais pour demander son exécution sur le serveur.

Le centre d'informations est une application repartie tri-partite(3-tiers), qui consistent en une interface utilisateur, une logique applicative et un accès à une base de données.
L'interface utilisateur dans une telle application est souvent basée sur le HTML. Dans certains cas, les applets Java interviennent dans cette tranche. Grâce aux fonctionnalités de réseau garanties par le navigateur, l'interface utilisateur peut communiquer avec la logique applicative intermédiaire. La partie intermédiaire constituée de servlets java et de classes utilitaires s'exécutant dans un serveur d'application java peut ensuite se connecter à la base de données pour accéder aux données et les manipuler. Les trois parties peuvent résider sur des ordinateurs distincts, connectés à un réseau.

[4] Un site web est un espace virtuel situé sur un serveur web contenant des renseignements présentés de diverses façons. Chaque site est composé d'une ou de plusieurs pages HTML reliées les unes aux autres par des liens hypertextes.

[5] Lien hypertexte :Mots ou images d'un document permettant de naviguer vers d'autres documents HTML qui sont rattachés par des liens pré-existants.

[6] URL : Universal Ressources Locator. Adresse d'un service, d'une page sur Internet.
Exemple : http://www.cnshb.org.gn/index.html

Une architecture 3-Tiers se différencie d'un système Client-Serveur par le fait que dans un système Client-serveur l'interface utilisateur et la logique applicative sont confondues.

La mise en place de ce type d'architecture permet une plus grande évolutivité du système. Il est ainsi possible de commencer par déployer les deux serveurs (serveur de base de données et serveur applicatif) sur la même machine, puis de déplacer le serveur applicatif sur une autre machine lorsque la charge devient excessive. Les éléments permettant la réalisation classique d'un système en architecture trois tiers sont les suivants:

- système de gestion de base de données relationnelles (SGBDR) pour le stockage des données (Mysql dans notre cas)
- serveur applicatif(Jakarta-Tomcat dans notre cas) pour la logique applicative
- navigateur web pour la présentation

Il est important de remarquer que l'essentiel du travail de développement sera implanté au niveau du serveur applicatif. Le SGBDR nécessitera un travail d'administration surtout dans le cas d'une quantité de données importantes. Le travail de conception de la base de donnée sera la pierre angulaire du système. En effet l'ensemble du développement s'appuiera sur cette conception. Le navigateur web nécessitera la programmation de code spécifique permettant de gérer l'affichage par ce navigateur. Ce code sera placé sur le serveur applicatif pour permettre une mise à jour sans nécessiter de nouveaux déploiements.

L'avantage principal d'une architecture 3-tiers est la facilité de déploiement. L'application en elle-même n'est déployée que sur la partie serveur (serveur applicatif et serveur de base de données). Le client ne nécessite qu'une installation et une configuration minime. En effet il suffit d'installer un navigateur web compatible avec l'application pour que le client puisse accéder à l'application, ce navigateur étant par ailleurs souvent installé par défaut sur toutes les machines. Cette facilité de déploiement aura pour conséquence non seulement de réduire le coût de déploiement mais aussi de permettre une évolution régulière du système. Cette évolution ne nécessitera que la mise à jour de l'application sur le serveur applicatif. Ceci est très important car cette évolutivité est un des problèmes majeurs de l'informatique. Le troisième avantage est l'amélioration de la sécurité. Dans un système client-serveur tous les clients accédaient à la base de données ce qui la rendait vulnérable. Avec une architecture multi-tiers l'accès à la base n'est effectué que par le serveur applicatif. Ce serveur est le seul à connaître la façon de se connecter à cette base. Il ne partage aucune des informations permettant l'accès aux données, en particulier le login et le password de la base.

Méthodes, formalismes et outils

Les travaux de réalisation du centre d'informations ont été conduits suivant une démarche incrémentale et itérative dans le but d'avoir un cycle de développement court. Dans ce type de démarche un logiciel noyau est tout d'abord développé, puis des incréments sont successivement développés et intégrés.
Le processus de développement de chaque incrément est un des processus classiques.
Les avantages de ce type de modèle sont nombreux :

- Chaque développement est moins complexe
- Les intégrations sont progressives ;
- Il peut y avoir des livraisons et des mises en service après chaque intégration d'incrément.

Dans la mise en application de cette démarche, des méthodes des formalismes et des outils ont été adoptés.

Objets / Agents

La programmation objet, bien plus qu'une méthode de travail, est une véritable façon de penser. Il a fallu approfondi cette manière de penser puisque tout le travail de développement au sein du projet était basé sur cette approche.

La **programmation orientée objet** consiste à modéliser informatiquement un ensemble d'éléments d'une partie du monde réel (que l'on appelle **domaine**) en un ensemble d'entités informatiques. Ces entités informatiques sont appelées **objet**.
La difficulté de cette modélisation consiste à créer une représentation abstraite, sous forme d'objets, d'entités ayant une existence matérielle.
Un objet est caractérisé par plusieurs notions :
• Les attributs : il s'agit des données caractérisant l'objet. Ce sont des variables stockant les informations d'état de l'objet.
• Les méthodes : elles caractérisent le comportement de l'objet, c'est à dire l'ensemble des actions (ou opérations) que l'objet est à même de réaliser.
• L'identité : elle permet de distinguer les objets entre eux, indépendamment de leur état.

La **classe** décrit le domaine de définition d'un ensemble d'objets de même type ou ayant un même comportement. Chaque objet appartient à une classe. Les généralités sont contenues dans les classes et les particularités dans les objets. Les objets informatiques sont construits à partir de la classe, par un processus appelé **instanciation**.
L'approche objet est une idée qui a désormais fait ses preuves. Simula a été le premier langage de programmation à implémenter le concept de classes en 1967. En 1976, Smalltalk implémente les concepts d'encapsulation, d'agrégation, et d'héritage (les principaux concepts de l'approche objet).
D'autre part, de nombreux langages orientés objets ont été mis au point dans un but universitaire (Eiffel, Objective C, Loops…).
Les concepts de l'approche objet sont :

- L'**encapsulation** qui consiste à rassembler les données et les méthodes au sein d'une même structure (l'objet) en cachant l'implémentation des données à l'utilisateur de l'objet. L'encapsulation permet donc de garantir l'intégrité des données contenues dans l'objet.
- L'**héritage** est la transmission des attributs et des méthodes d'une classe vers une sous-classe.
- Le **polymorphisme** caractérise la possibilité de définir plusieurs fonctions de même nom mais possédant des paramètres différents.

Des organismes se sont créés, face à l'émergence du concept objet, dans un but de standardisation des méthodes.
L'**ODMG** (Object Database Management Group) est composé de représentants des vendeurs de SGBDO : Gemstone Systems, IBEX, O2 Technology, Object Design, Objectivity, POET Software, UniSQL et Versant. Ses membres se sont mis d'accord sur un standard. Ils ont proposé un langage de définition de données (ODL), un langage d'interrogation des données (OQL) et une intégration avec les langages objet (OML) C++, Smalltalk et Java. L'ODMG est une émanation de l'OMG.

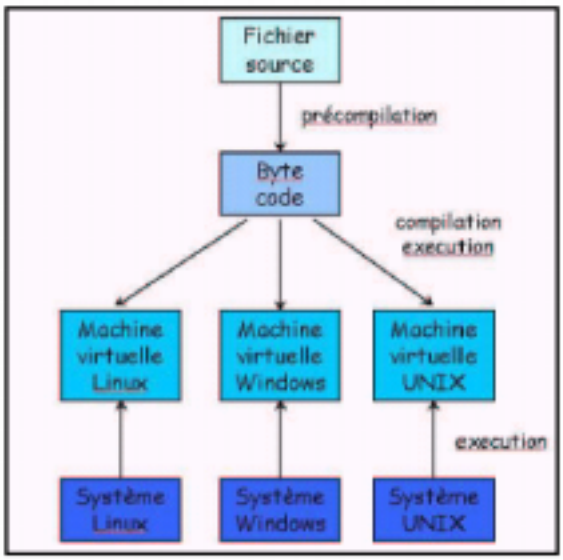
L'**OMG** (Object Management Group) est un consortium international créé en 1989 et regroupant actuellement plus de 850 acteurs du monde informatique : des constructeurs (IBM, Sun), des producteurs de logiciel (Netscape, Inprise ou ex-Borland/Visigenic, IONA Tech.), des utilisateurs (Boeing, Alcatel) et des institutionnels et universités (NASA, INRIA, LIFL). L'objectif de l'OMG est de faire émerger des standards pour l'intégration d'applications distribuées hétérogènes à partir de technologies orientées objet.
Ainsi, les concepts clés mis en avant sont la réutilisabilité, l'interopérabilité et la portabilité de composants logiciels. L'élément clé de la vision de l'OMG est CORBA (Common Object Request Broker Architecture) qui permet à des objets écrits dans des langages de programmation différents de communiquer.

Il existe une grande similitude entre le concept *d'Objet* et celui *d'Agent* : En effet un *agent* peut être défini comme un objet informatique(au sens des langages Objets) qui dispose en plus d'une autonomie de décision et est mu par un ensemble d'objectifs. On implémente le plus souvent les agents sous formes d'objets pour des raisons pratiques.
La première spécification du CI prévoyait d'implémenter les objets sous forme d'agents pour permettre l'intégration du CI au simulateur. Ce choix comportant de nombreuses difficultés d'implémentation a été finalement abandonné.

Deux formalismes liés à ce concept ont été largement utilisés dans le CI :

Java

Le langage de programmation Java, développé par Sun Microsystems est conçu pour être un langage de programmation indépendant de la machine, à la fois suffisamment sûr pour traverser le réseau et assez puissant pour remplacer du code exécutable natif.
Java est un langage dérivé du C++ qui présente la particularité d'être gratuit et portable (voir figure ci-dessous). Il est portable car il s'agit d'un langage interprété et pré compilé. La pré compilation génère un format particulier appelé Byte Code qui peut être exécuté sur n'importe quelle machine pourvu qu'elle ait ce qu'on appelle la machine virtuelle java. La machine virtuelle java est intégrée dans les navigateurs(Netscape Navigator, Internet Explorer), ceci permet à de petits programmes java (appelés dans ce cas applets) d'être intégrés dans des pages au format Internet (HTML : Hyper Text Markup Language) et donc de pouvoir s'exécuter sur la machine de la personne qui consulte le site Internet. C'est cette possibilité d'intégrer des "applet" java dans des pages au format HTML qui a fait l'énorme succès de java ces dernières années.



La portabilité du langage Java.

Java est devenu une plate-forme importante pour les applications coté serveur, grâce à l'interface *servlet*, et pour les applications d'entreprises qui exploitent les technologies telles que *Entreprise JavaBeans*. De plus, java est la plate-forme de prédilection des applications reparties modernes.

La version Java utilisée pour le CI est le jdk1.3.1 et l'IDE (Environnement de développement intégré) forte for Java tous deux gratuits.

La Modélisation UML

La modélisation objet consiste à créer une représentation informatique des éléments du monde réel auxquels on s'intéresse, sans se préoccuper de l'implémentation, ce qui signifie indépendamment d'un langage de programmation. Il s'agit donc de déterminer les objets présents et d'isoler leurs données et les fonctions qui les utilisent. Pour cela, des méthodes ont été mises au point. Entre 1970 et 1990, de nombreux analystes ont mis au point des approches orientées objets, si bien qu'en 1994, il existait plus de 50 méthodes objets. Toutefois, seules 3 méthodes ont vraiment émergé :

· La méthode OMT de Rumbaugh
· La méthode BOOCH'93 de Booch
· La méthode OOSE de Jacobson

A partir de 1994, Rumbaugh et Booch (rejoints en 1995 par Jacobson) ont unis leurs efforts pour mettre au point la méthode UML (Unified Modeling Language), qui permet de définir une notation unifiée standard en incorporant les avantages de chacune des méthodes précédentes (ainsi que celles d'autres analystes). La première version d'UML a été soumise à l'OMG en 1997. Elle a été acceptée à l'unanimité.
Cette étape de modélisation doit permettre de faciliter la compréhension de la structure du centre d'informations et de simuler son comportement, soit ses cas d'utilisations par les futurs utilisateurs.
UML utilise des diagrammes qui permettent de visualiser et de manipuler les éléments de modélisation.

Il s'agit de diagrammes structurels et comportementaux qui présentent respectivement les vues statiques et dynamiques d'un système
Les **diagrammes structurels** sont au nombre de 4.
Les *diagrammes de classes* représentent la structure statique en terme de classes et relations; les *diagrammes d'objets* représentent les objets et leurs liens et correspondent à des diagrammes de collaboration simplifiés, sans représentation des envois de messages; les *diagrammes de déploiement* représentent le déploiement des composants sur les dispositifs matériels; *les diagrammes de composants* représentent les composants physiques d'une application.
Les **diagrammes comportementaux** sont au nombre de 5.
Les *diagrammes d'activités* représentent le comportement d'une méthode, d'un cas d'utilisation, ou d'un processus métier; *les diagrammes d'états-transitions* représentent le comportement d'un classificateur ou d'une méthode en terme d'états; les *diagrammes de cas d'utilisation* représentent les fonctions du système du point de vue des utilisateurs; les *diagrammes de séquence* sont une représentation temporelle des objets et de leur interaction; les *diagrammes de collaboration* sont une représentation spatiale des objets, des liens et des interactions.

L'outil de modélisation UML retenu pour la conception du CI est Rational Rose.

Web

XML

Le langage XML (Extensible Markup language) issu du SGML (Structured Generalized Markup Language) est un métalangage de définition de types de documents. Dans ce sens, il permet de fixer les règles de syntaxe de tout nouveau type de document tout comme le SGML qui a défini le type de document HTML.

Le XML a l'avantage d'abolir les limites du HTML tout en évitant la trop grande complexité du SGML qui a freiné son développement. En effet, alors que le texte d'un document HTML repose sur des données non structurées qui ne peuvent être traitées par un programme, un document XML présente une structure intermédiaire entre celle rigoureuse d'une base de données et celle inexistante d'un texte. Il fournit ainsi des données aptes à être traitées.

Cette extensibilité du XML s'explique par l'introduction d'un nouveau type de balises appelées des balises sémantiques. Celles-ci ne donnent ni indication de structure ni celle de formatage mais sont accessibles aux scripts et aux programmes. Elles permettent ainsi l'application d'autres technologies telles que le HTML dynamique ou les feuilles de style. De cette manière, le XML accroît les possibilités de créer des descriptions adaptées aux besoins et de générer des pages plus interactives.

XSL

L'idée d'écarter les balises de formatage dans un document XML a consolidé son association avec le langage XSL (Extensible Stylesheet Language) qui permet de gérer l'affichage des données. Le rôle du langage XSLT (XSL Transforms) est alors de transformer des documents en appliquant les feuilles de style XSL. Dans notre cas le programme XSLT contiendra des instructions de transformation d'un document XML en un document HTML.

Le document XML généré à partir de la base de données et contenant l'information sera le résultat d'un programme. Celui-ci sera envoyé à un processeur XSLT afin d'être converti en document HTML selon les instructions d'un document XSLT associé. Lorsque le document HTML est reçu par le navigateur, celui-ci ne sait pas que le document original est un document XML. Dans ce modèle, la transformation se fait du côté serveur et le navigateur affiche simplement le document HTML.

On aurait pu envoyer le document XML accompagné du document XSLT au navigateur, ce qui lui présente l'avantage de posséder en mémoire les deux documents XML et HTML et donc de pouvoir transformer le document XML plusieurs fois. Toutefois tous les navigateurs ne sont pas forcément capables de réaliser la transformation et ne possèdent pas un processeur XSLT. Notre modèle utilisera donc juste la fonction principale du navigateur qui est d'interpréter HTML.

L'outil utilisé pour la manipulation des documents xml et xsl est XmlSpy.

HTML

Signifie Langage de Balise HyperTexte (HyperText Markup Language), HTML est le langage de programmation utilisé pour créer des documents hypertexte. HTML utilise une liste déterminée de balises qui décrivent la structure générale d'une variété de documents reliés les uns aux autres sur Web. HTML est un sous-ensemble plus convivial du Standard Generalized Markup Language (SGML). Le code HTML est en constante évolution et s'enrichit régulièrement de nouvelles fonctions(Exemple du dynamic HTML)
De nombreux logiciels ont fait leur apparition pour créer des pages HTML de manière moins contraignante qu'en entrant manuellement des lignes de commandes (FrontPage, Dreamweaver, Golive,).
Le HTML est le mécanisme préféré de représentation de l'interface utilisateur lorsque les impératifs de portabilité priment.

Base de données

SQL92

(Anglais : Structured Query Language). est un langage évolué destiné aux systèmes de bases de données relationnelles. Développé à l'origine par le laboratoire de recherche d'IBM à San Jose en Californie à la fin des années 70, SQL a été adopté pour de nombreux systèmes de gestion de bases de données relationnelles et adapté à ceux-ci. Il a été reconnu en tant que norme officielle de langage de requête relationnelle par l'institut ANSI (American National Standards Institute) et par l'organisme ISO (International Standards Organization). Transact-SQL est compatible avec IBM SQL et avec la plupart des produits SQL commercialisés, mais il offre d'importantes fonctions et fonctionnalités supplémentaires.

Bien que SQL soit un langage de requêtes, il ne comporte pas uniquement des commandes de requête, c'est-à-dire d'extraction de données d'une base de données, mais offre aussi des commandes de création de bases de données et d'objets de bases de données, d'ajout de données, de modification de données existantes et beaucoup d'autres fonctions.

Le langage SQL est utilisé par les programmes java pour accéder à la base de données du centre d'informations via la technologie JDBC.

Deux SGBDR ont été choisis pour stocker les informations du centre d'informations :

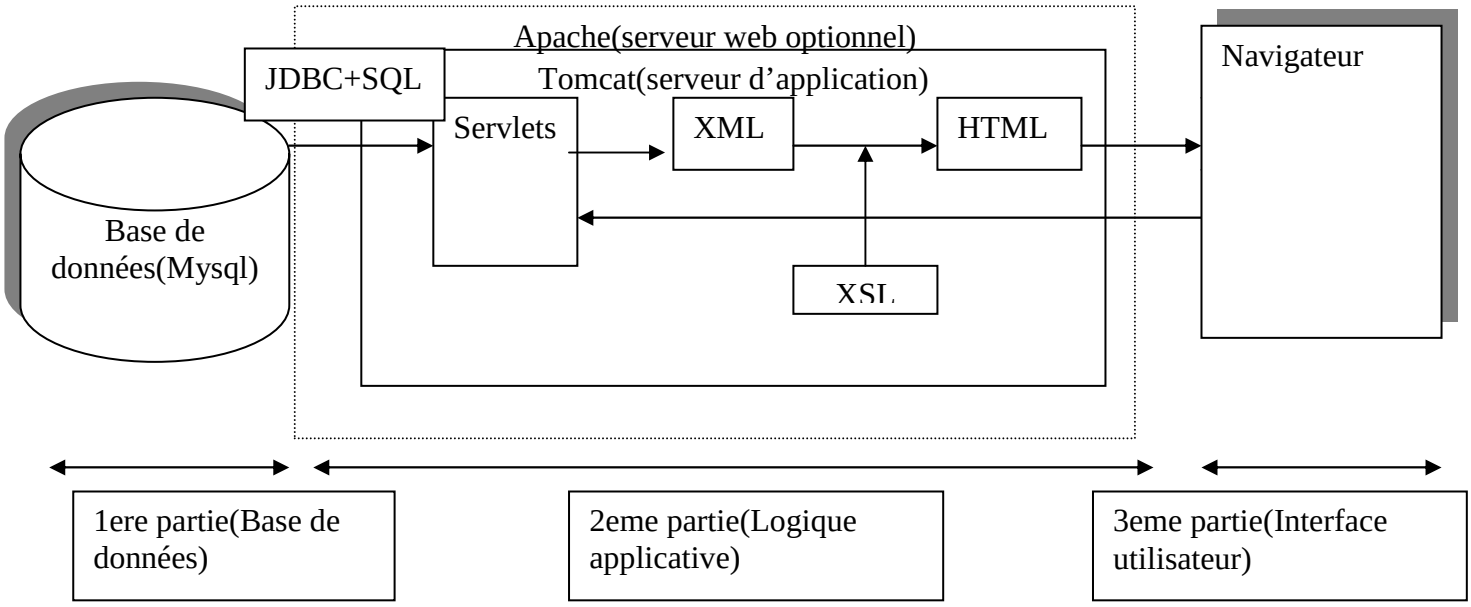
Mysql 3.23.C'est un serveur de base de données très rapide et la licence est gratuite dans le cas de son utilisation dans le centre d'informations.

MsAccess :Inclus dans la suite logicielle Microsoft Office , il constitue la plate forme standard de l'alimentation du CI(saisie de données). Il peut être utilisé pour une utilisation personnelle du centre d'informations sous windows.

Tableau récapitulatif des Concepts ,formalismes et outils

Concepts	Formalismes	Outils
Agents/Objets	UML	Rational Rose
	Java	Jdk1.3.1, Forte,Tomcat, api Servlets, JDBC, XML
Web	Xml/Xsl	XmlSpy
	HTML	FrontPage
Base de données(modèle Relationnel et datamining)	SQL	Mysql, MsAccess

Ceci peut se représenter dans le schéma ci-dessous :



Architecture technique du centre d'informations

Cahier des charges de la réalisation

Conceptions architecturale et détaillée

L'activité de conception consiste à enrichir la description du logiciel de détails d'implémentation afin d'aboutir à une description proche d'un programme.
Cette activité a été menée à trois avec Loïc Thibaut(ingénieur qualité du projet PEG), Misako Ito (étudiante de 2 eme année au ENSEEIHT de Toulouse filière électronique) qui effectuait un stage de 6 semaines au CNSHB et moi.

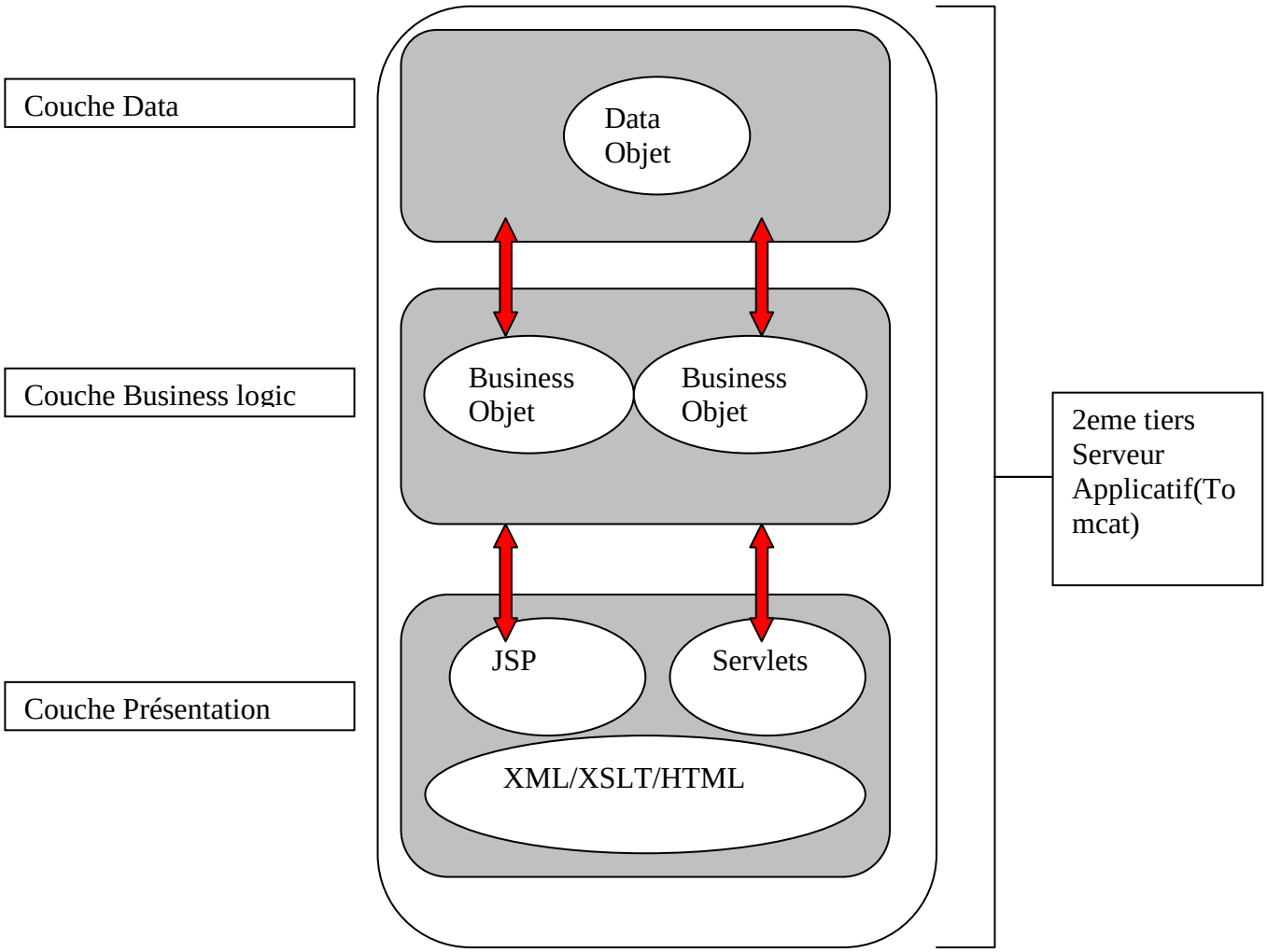
Dans l'architecture 3-Tiers retenue le seul tiers nécessitant une phase de programmation importante est le serveur d'application. Notre objectif lors de la programmation du serveur applicatif était de rester suffisamment proche de l'architecture 3-tiers pour faciliter la maintenance du système. Nous avons donc cherché à conserver la ``logique trois-tiers" dans la structure de cette partie. Le codage de cette partie nécessite la réalisation de plusieurs tâches directement en lien avec chacun des trois tiers:

- l'accès aux données (JDBC)
- la logique applicative ou traitements sur les données (peu importante ici)
- la génération et la diffusion du code nécessaire à la présentation (JSP, Servlets, XML/XSLT)

L' évolutivité du système sera directement liée à la séparation nette de ces différentes parties. En effet chacune de ces parties peut-être modifiée séparément du reste du système. On peut par exemple décider de modifier la page HTML(fichiers xsl) de présentation sans modifier le reste. La séparation nette entre la présentation et le reste permet d'effectuer les modifications sans être perdu dans le code applicatif et l'accès à la base de données mais également sans risquer d'effectuer de modification inopportune sur ce code. En pratique on constate un lien très fort entre la logique applicative et les données. En effet l'objectif est ici d'obtenir des objets correspondant au mapping des tables(Informations, Descripteurs etc.) de la base de données et directement utilisables pour la présentation (Servlet, XML) et la couche Data.

L'application a donc été structurée en trois couches abstraites associées chacune à un package java. Chaque couche étant constituée d'un ensemble de classes java ayant des fonctions apparentées.

Figure 1 Modélisation du Tiers Business logic



La couche Data

C'est dans cette couche que s'effectue la connexion avec la base de données et l'extraction de l'information requêtée sous un format générique.

Les classes et interfaces utilisées dans cette couche sont :

DataFactory : Interface possédant des méthodes pour la récupération des Business objets.

MyDataFactory : Classe servant à la récupération des données dans une base de données. Cette classe utilise l'API JDBC(Java DataBase Connectivity) pour l'accès aux bases de données.

La couche Business logic

La couche Business logic regroupe les objets encapsulant les résultats des requêtes utilisateurs. Les objets de cette classe sont manipulés par les objets des deux autres couches.

-**Information** qui comporte des variables d'instances privées représentant tous les descripteurs de l'information et des méthodes publiques permettant d'y accéder.

-**ListeInformations** qui comporte une collection d'informations ayant en commun les paramètres du descripteur et qui possède un itérateur permettant d'accéder à cette liste d'informations.

-**ListeInformationsFiltre** Classe héritée de ListeInformations qui contient en plus un filtre qui permet de faire une sélection sur l'ensemble des informations*(version 1.1 du CI).*

-**Filtre** Classe encapsulant les informations nécessaires au filtrage d'une requête utilisateur*(version 1.1 du CI).*

-**Descripteur** : Classe abstraite implémentée par les descripteurs simples et complexes.

-**DescripteurSimple** et **DescripteurComplexe** qui implémentent l'interface Descripteur. Un descripteur est dit simple lorsqu'à un type du descripteur on peut associer un contenu unique (ex : support, dimension, date_info…) et il est qualifié de composite lorsqu'à un type du descripteur on peut associer plusieurs contenus (ex : mots_cles, composants).
Un objet descripteur comporte donc en attributs privés une chaîne de caractères représentant le type du descripteur et une chaîne de caractères ou une collection représentant le ou les contenus du descripteur selon qu'il soit simple ou composite.

-**DescripteurSimpleSupport:** Descripteur ayant une valeur unique et possédant en plus un champ fichier et un champ logo. Cette classe a été insérée pour traiter spécialement le cas du descripteur support.

-**InformationFactory** : Classe faisant office d'interface entre la couche Business et la couche supérieure pour la récupération des business objets.
Elle utilise une instance de la classe DataFactory à cet effet.

- **InstancesDescripteur** : Classe encapsulant la liste des instances d'un descripteurs du CI.

- **InstanceDescripteur :** Classe représentant une instance d'un descripteur

- **ListeDescripteurs :** Classe contenant un ensemble de descripteurs liés à un autre descripteur*(version 1.1 du CI).*

La couche Présentation

Elle représente l'interface avec l'utilisateur. Elle est composée de deux sous-couches :
La première permet la création des documents XML à partir des objets générés par la couche Business logic.

Il existe une seule classe dans cette sous-couche :

XMLDocumentFactory : Classe servant à construire un document xml

à partir d'un Business objet.

Les modèles des documents XML que la classe XMLDocumentFactory doit générer sont les suivants :

- pour un document XML présentant une information

```
<information id="0001">
<titre>Pêche au Tété Yèlè</titre>
<sujet>pêcheur portant un filet à juvéniles de poisson</sujet>
<referencesource>(crédit photo)</referencesource>
<commentaire>La pêche au filet tété yèlè est une pêche à pied pratiquée
majoritairement
par des femmes sur les côtes de Guinée.
La plupart des pêcheurs utilise des moustiquaires pour la confection des filets; les
mailles utilisées pour ces filets vont de 3,5 à 5mm.
Les produits vendus sont les kouppè et gbiti (fritures) et le Samataran (petites
crevettes).</commentaire>
<niveauconfiance type="niveauconfiance" elem="100%25">100%</niveauconfiance>
<lieu type="lieu" elem="Kaporo+mer">Kaporo mer</lieu>
<source type="source" elem="Le+Fur%2C+J.%2C+IRD%2C+2000">Le Fur, J., IRD,
2000</source>
<date type="date" elem="2000">2000</date>
<dimension type="dimension" elem="Technologie">Technologie</dimension>
<auteurinfo type="auteurinfo" elem="Jean+Le+Fur+%28IRD%29">Jean Le Fur
(IRD)</auteurinfo>
<datesaisie type="datesaisie" elem="2000-11-14">2000-11-14</datesaisie>
<groupes>
<groupe type="groupe" elem="filet+t%E9t%E9+y%E8l%E8">filet tété yèlè</groupe>
</groupes>
<composants>
<composant type="composant" elem="engin+de+p%EAche">engin de
pêche</composant>
<composant type="composant" elem="p%EAcheur">pêcheur</composant>
</composants>
<supports>
<support type="support" elem="photo" fichier="0001.jpg"
logo="bphoto.gif">photo</support>
<support type="support" elem="photo" fichier="0001.gif"
logo="bphoto.gif">photo</support>
</supports>
</information>
```

Note : Les attributs « elem » et « type » des différentes balises xml ont été encodées afin d'être insérés dans des URLs qui ne doivent pas contenir des caractères accentués.

On définit ainsi sous XML une information par un élément « information » comportant des attributs et des sous-éléments qui sont les descripteurs. L'attribut « id » permet d'identifier de manière unique l'information et « fichier » indique le nom du fichier contenant l'information. Parmi les sous-éléments qui composent l'information, on peut distinguer par la structure du document XML, les descripteurs simples et les descripteurs composites. Les descripteurs composites sont représentés sous XML par des éléments comportant des sous-éléments dont

le nom est identique à celui de l'élément père au singulier et dont le nombre correspond au nombre de contenus.
Cette structure du document XML permet de référencer chaque composant d'une information pour y faciliter l'accès.

-pour un document XML présentant une liste d'informations

```
<liste type="lieu" contenu="zee" nombre="3">
<information id="0158">
<sujet>répartition et composition des fonds marins de 0 à 200m</sujet>
<titre>Carte des sédiments de la ZEE guinéenne</titre>
</information>
<information id="0197">
<sujet>coordonnées et tracé officiel</sujet>
<titre>Délimitation de la ZEE (zone économique exclusive) guinéenne</titre>
</information><information id="0248">
<sujet>taux d'inactivité des chalutiers (période 1995-2000)</sujet>
<titre>PI - activité des chalutiers en Guinée</titre>
----------suite---------- ----------
</liste>
```

Une liste d'informations est définie par ses attributs et rassemble toutes les informations qui ont le même descripteur c'est à dire même « contenu » pour un même « type » du descripteur. Chaque information faisant partie de cette liste possède un attribut « id » et deux élément-fils « sujet » et titre.

- pour un document présentant une liste d'informations filtrée

```
<liste type="lieu" contenu="zee" nombre="3" filtreType="support"
filtreContenu="graphique">
<information id="0158">
<sujet>répartition et composition des fonds marins de 0 à 200m</sujet>
<titre>Carte des sédiments de la ZEE guinéenne</titre>
</information>
<information id="0197">
<sujet>coordonnées et tracé officiel</sujet>
<titre>Délimitation de la ZEE (zone économique exclusive) guinéenne</titre>
</information><information id="0248">
<sujet>taux d'inactivité des chalutiers (période 1995-2000)</sujet>
<titre>PI - activité des chalutiers en Guinée</titre>
</liste>
```

Une liste d'informations filtrée est une liste d'informations limitée aux informations satisfaisant aux conditions du filtre. Exemple pour le cas ci-dessus on affichera les informations appartenant au lieu « zee » et ayant en plus comme support le type grahique.

- pour un document XML présentant une liste des instances d'un descripteur

```
<liste type="source" nombre="3">
<instance>
<champ1 type="source" contenu="Anonyme+%281996%29">Anonyme
(1996)</champ1>
<champ2 type="source"
contenu="Arr%EAt%E9+n%B096%2F7009%2FMPE%2FSGG+portant+approbation+du+
plan+de+p%EAche+1997%3B+sign%E9+le+24%2F12%2F1996">Arrêté
n°96/7009/MPE/SGG portant approbation du plan de pêche 1997; signé le
24/12/1996</champ2>
</instance>
<instance>
<champ1 type="source" contenu="Bamy%2C+I.L.%2C+CNSHB%2C+2001">Bamy, I.L.,
CNSHB, 2001</champ1>
<champ2 type="source" contenu="%28cr%E9dit+photo%29">(crédit photo)</champ2>
</instance>
<instance>
<champ1 type="source" contenu="Bazzo%2C+D.+%282000%29">Bazzo, D.
(2000)</champ1>
<champ2 type="source"
contenu="Reconstitution+virtuelle+du+relief+de+la+Guin%E9e+Maritime.+Observatoir
e+de+la+Mangrove%2C+CNSHB%2C+R%E9p.+de+Guin%E9e%2C+2000.">Reconstitut
ion virtuelle du relief de la Guinée Maritime. Observatoire de la Mangrove, CNSHB, Rép.
de Guinée, 2000.</champ2>
</instance>
</liste>
```

Note : Les attributs « contenu » et « type » des différentes balises xml ont été encodées afin d'être insérés dans des URLs qui ne doivent pas contenir des caractères accentués.

Une liste des instances d'un descripteur est identifiée seulement par un attribut contenant le type de l'instance(ex source, dimension) et un attribut indiquant le nombre d'instance . Chaque instance faisant partie de cette liste possède un ou deux éléments-fils suivant le descripteur.

- pour un document XML présentant une liste de descripteurs associée à un autre

```
<?xml version="1.0" encoding="UTF-8"?>
<liste type="composants" nombre="18">
   <instance>bateau</instance>
   <instance>chalut</instance>
   <instance>pêcheur</instance>
   <instance>espèce</instance>
   <instance>poisson</instance>
   <instance>bâtiment</instance>
   <instance>débarcadère</instance>
   <instance>capture</instance>
   <instance>effort</instance>
   <instance>pêche artisanale</instance>
   <instance>pêche industrielle</instance>
   <instance>pêche maritime</instance>
   <instance>poisson séché</instance>
   <instance>filet tété yèlè</instance>
   <instance>pêcheurs</instance>
   <instance>sédiment</instance>
   <instance>fond marin</instance>
</liste>
```

Une liste de descripteurs est identifiée par un attribut contenant le type de l'instance(ex composant) et un attribut indiquant le nombre d'instances. Les instances ici ne possèdent aucun élément-fils.

La seconde sous-couche permet la transformation d'un document XML en HTML en appliquant une feuille de style xsl et l'envoi au navigateur des documents HTML ainsi générés à l'aide des *Servlets*.
*Un **Servlet** est un programme Java exécuté sur un serveur web et qui génère automatiquement du contenu dynamique.*

On définit dans cette couche cinq(5) Servlets : un qui sera exécuté lorsque l'utilisateur cliquera sur un lien vers une information ,un autre pour un lien vers une liste d'informations le troisième vers les instances d'un descripteur le quatrième vers les descripteurs associés à un autre descripteur(comme les groupes associés à un composant donné) et le cinquième pour ajouter ou retirer un filtre à une liste d'informations.

Les classes utilisées dans cette sous-couche sont :

HttpGetServletInstancesDescripteur : Servlet traitant les requêtes portant sur l'affichage de la liste des instances d'un descripteur.

HttpGetServletListeInformations : Servlet traitant les requêtes portant sur l'affichage

d'une liste d'informations.

HttpGetServletInformation : Servlet traitant les requêtes portant sur l'affichage d'une information.

HTMLDocumentFactory : Contient une méthode qui permet de transformer un fichier xml

en un fichier html en utilisant une feuille de style xsl.

Les différentes feuilles de style(xsl) utilisées pour la transformation sont en annexes du rapport.

HttpGetServletListeDescripteurs : Servlet traitant les requêtes portant sur l'affichage sur la liste des descripteurs associés à un autre descripteur(version 1.1 du CI).

HttpGetServletFiltre : Servlet traitant les requêtes portant sur le filtrage d'une liste d'informations. Ce servlet fait appel à la fonctionnalité suivi de session pour identifier chaque utilisateur qui applique ou annule un filtre(version 1.1 du CI).

Les classes de ces trois couches utilisent la Classe **Parametres**(qui n'est dans aucune de ces couches) pour accéder aux paramètres de l'application tels que les informations de connexion à la base de données à utiliser .

Modélisation UML

Les diagrammes de classes

Un diagramme de classes est une collection d'éléments de modélisation statiques (classes, paquetages...), qui montre la structure d'un modèle. Il fait abstraction des aspects dynamiques et temporels.

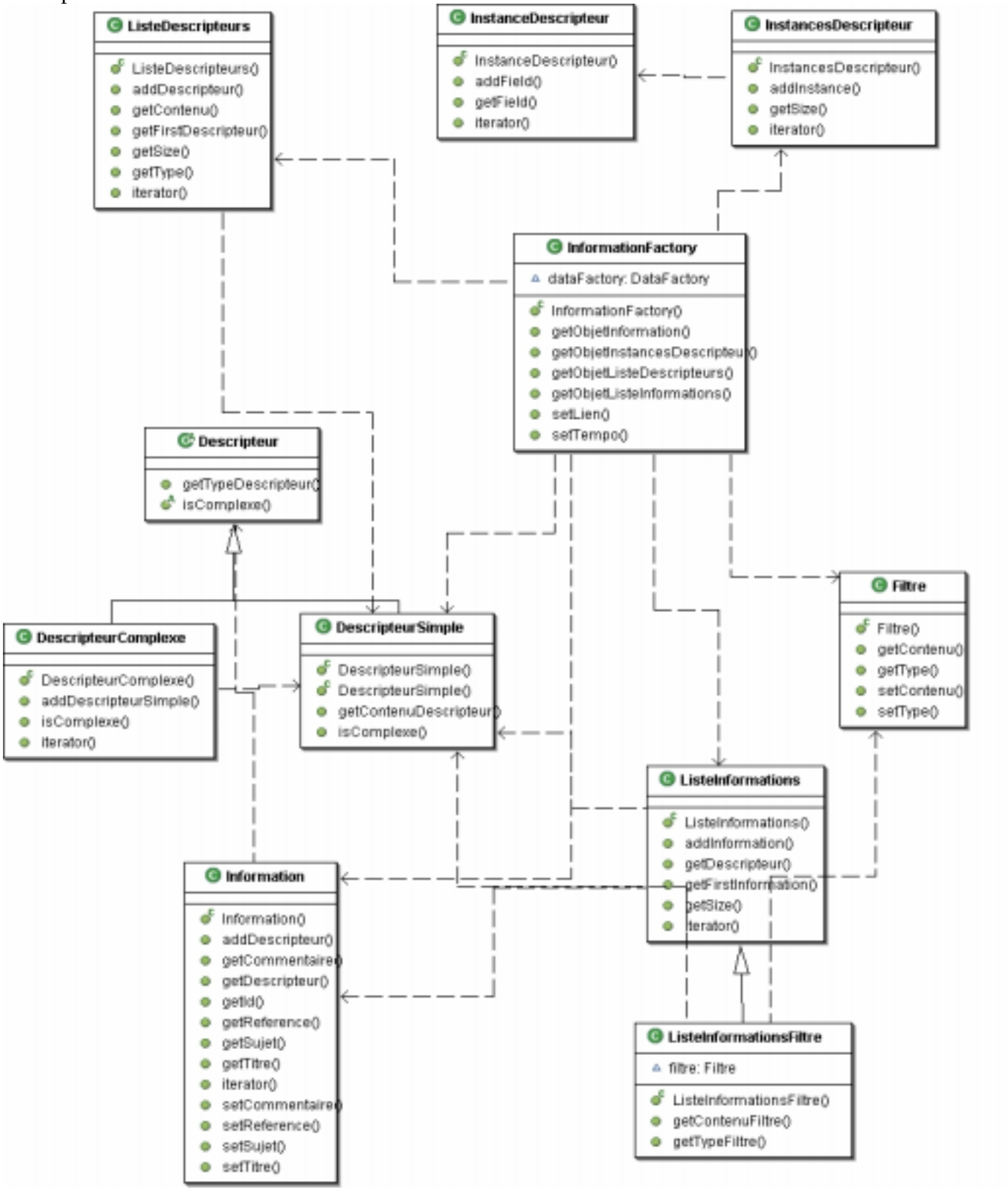


Diagramme de classe de la couche Business logic

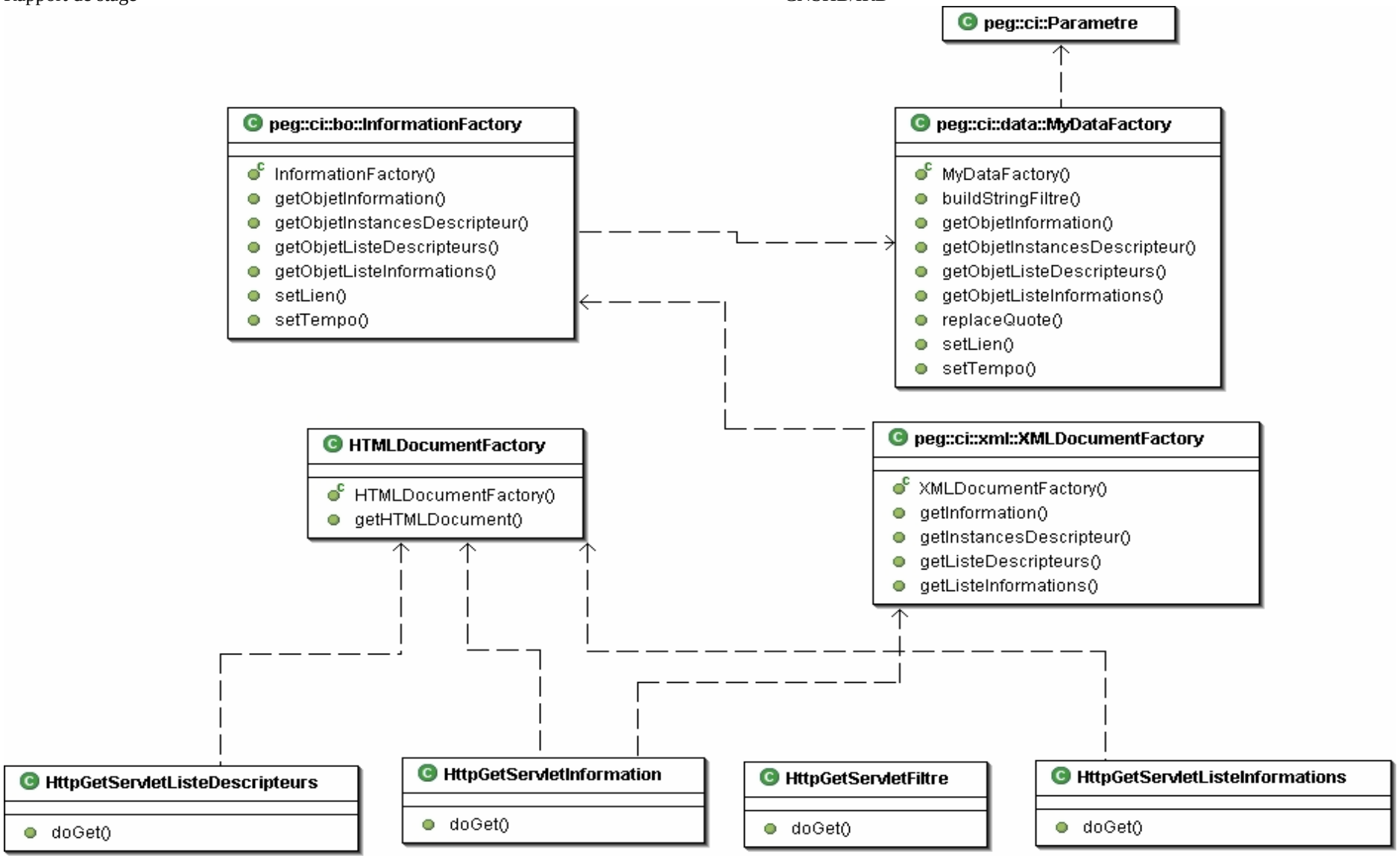


Diagramme de classes simplifié du CI

Les diagrammes de collaboration

Les diagrammes de collaboration montrent des interactions entre objets(instances de classes et acteurs). Ils permettent de représenter le contexte d'une interaction, car on peut y préciser les états des objets qui interagissent.
Trois diagrammes de collaboration permettent de comprendre le fonctionnement du centre d'informations.

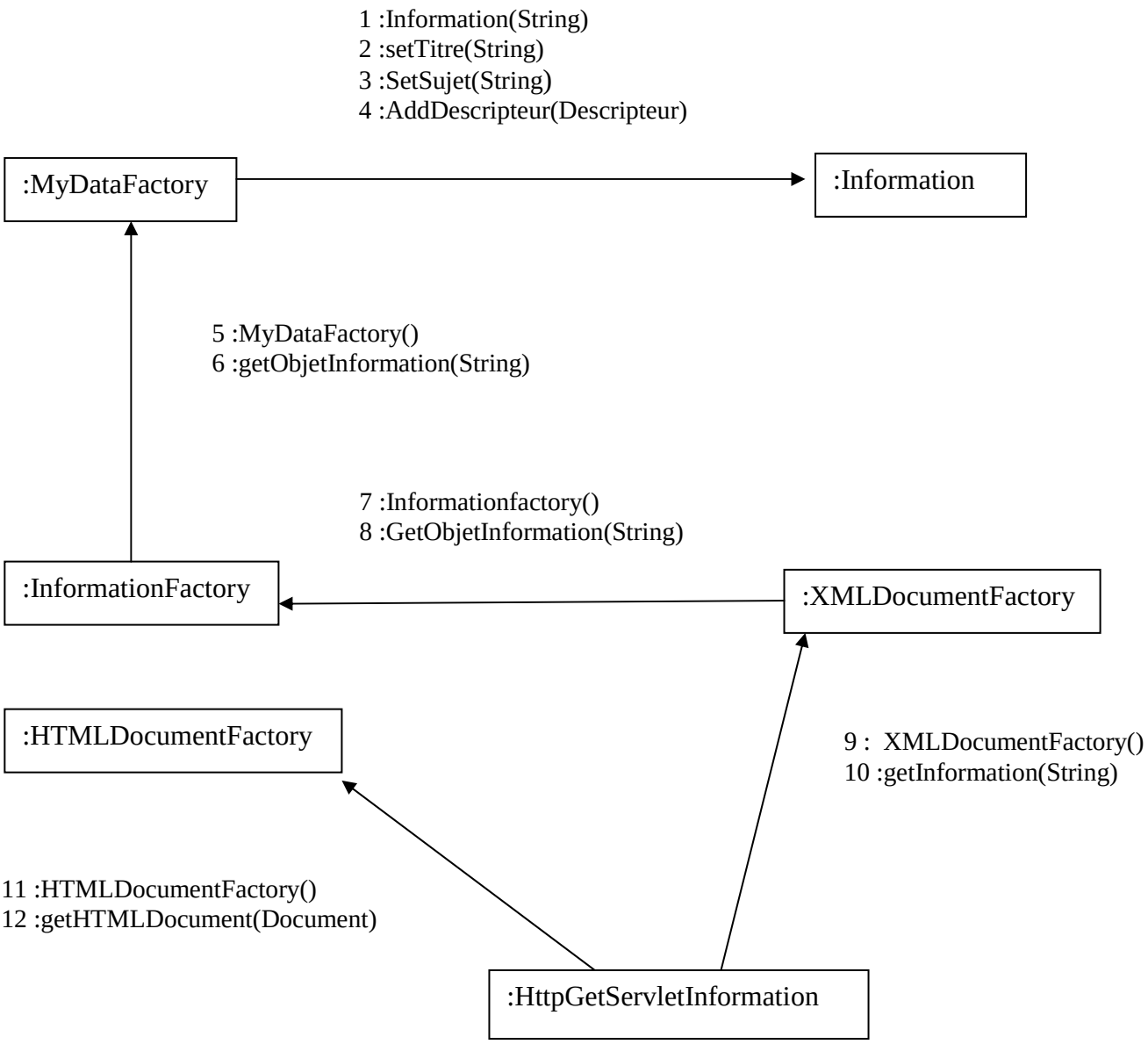


Diagramme d'interaction pour l'affichage d'une information

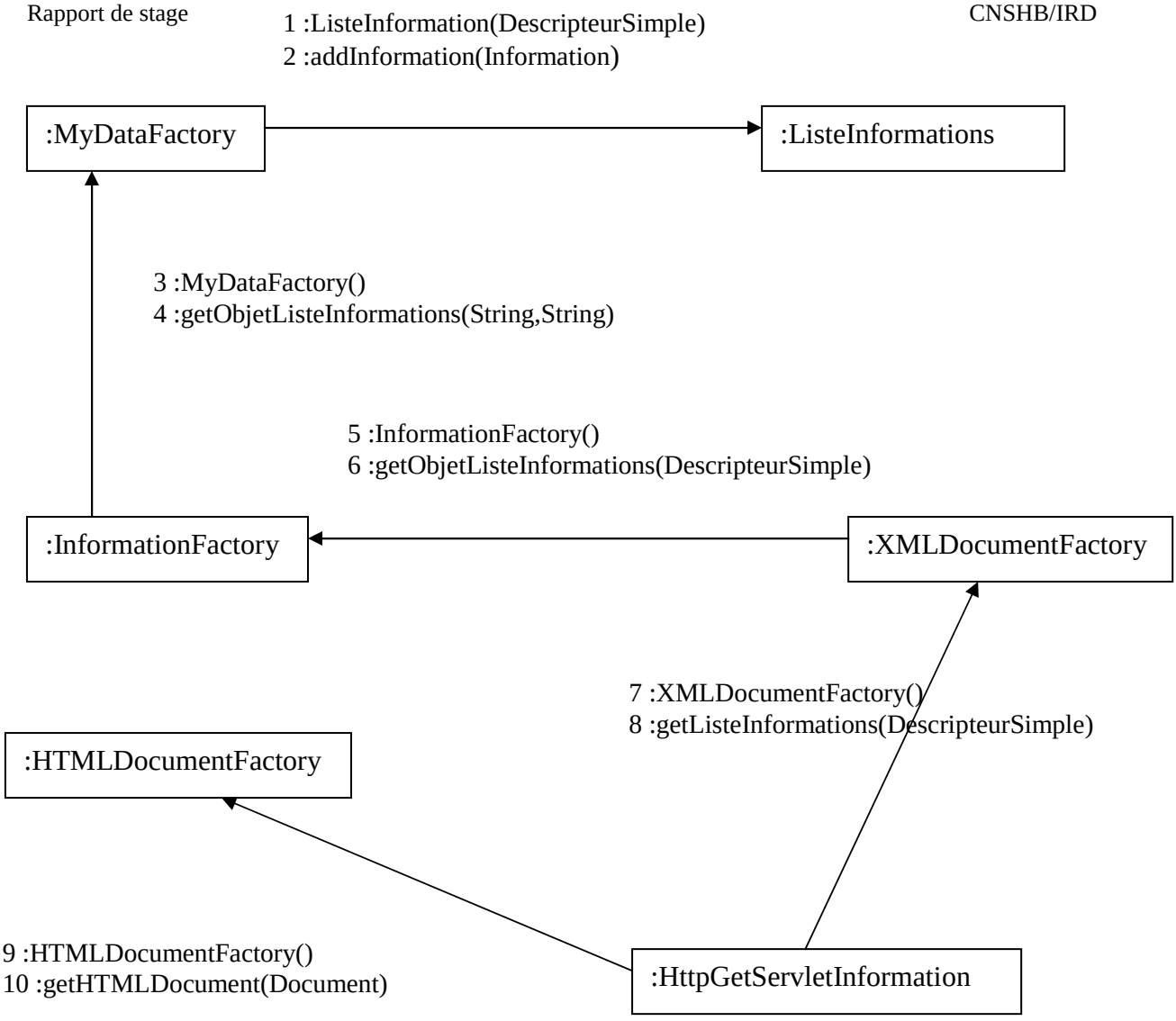


Diagramme d'interaction pour l'affichage d'une liste d'informations

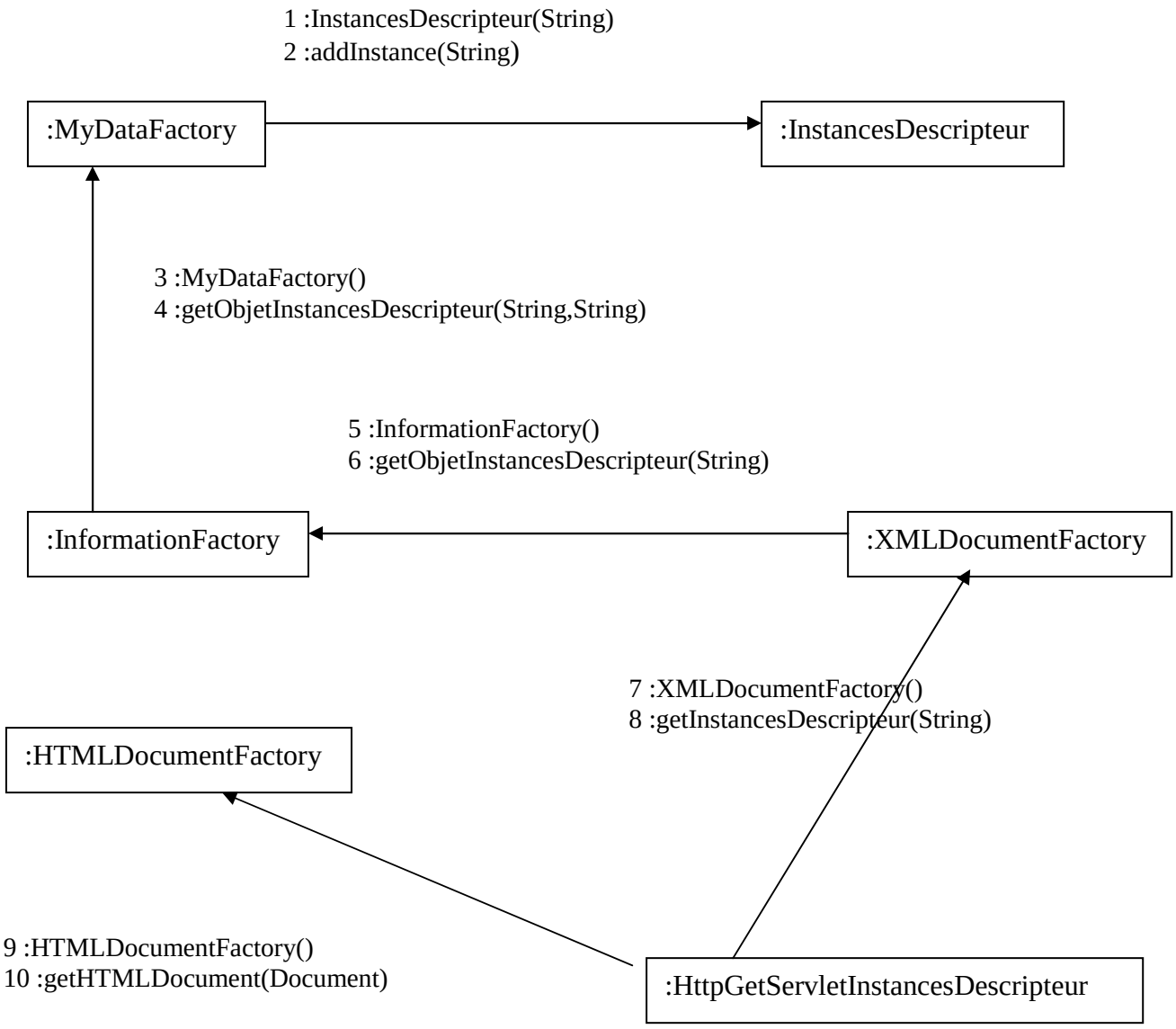
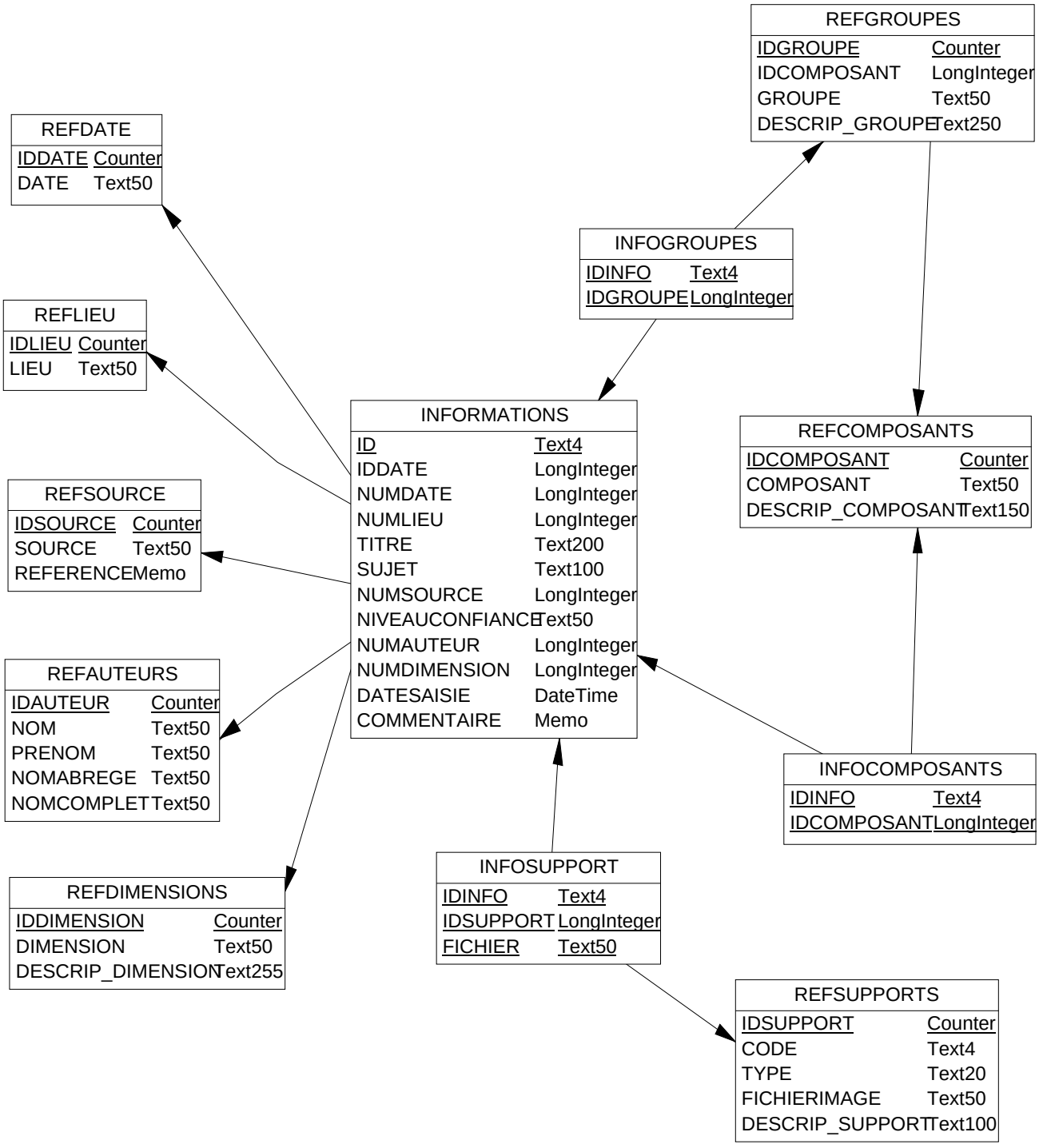


Diagramme d'interaction pour l'affichage des instances d'un descripteur

Note :Le diagramme d'interaction pour l'affichage des descripteurs associés à un autre n'est pas représenté ici :En effet il pourrait se déduire à partir des trois autres diagrammes d'interaction.

La base de données du centre d'informations

C'est dans cette base de données que sont stockées toutes les informations utilisées dans le centre d'informations. Cette base est implantée dans le SGBD Access ou MySQL.
Le choix de la base de données dépend de la variante du logiciel :pour la variante standalone on préférera Access tandis que pour les versions destinées à tourner en réseau on optera pour MySQL.
Le modèle physique de cette base de données est le suivant :



Programmation, intégration et test

Java

La programmation des classes Java modélisées a été principalement faite par Misako Ito et moi avec les conseils de Loïc Thibaut. Je me suis chargé au départ du codage de la couche Business logic et Data tandis que Misako avait en charge le codage de la couche présentation. Après le départ de celle-ci, j'ai eu à apporter de nombreuses modifications aux classes des différentes couches.
En ce qui concerne la conception et la réalisation de la base de données du centre d'informations, elles ont été principalement faites par le chef de projet Jean Le fur et Salif Sylla qui est aussi membre du projet PEG avec mon aide.

XSL

Au départ la création des feuilles de style XSL qui permettent la mise en forme des résultats des requêtes utilisateurs a été faite par Misako puis après par Jean Le Fur, Souleymane Diakité et moi. Quatre(4) feuilles de style ont été créées à cet effet, une pour la mise en page d'une information, une pour celle d'une liste d'informations la troisième pour l'affichage des instances d'un descripteur et la dernière pour l'affichage d'une liste de descripteurs.
Les sources de ces documents sont disponibles en Annexes.

Résultats

Après la phase de programmation et au terme de nombreuses mises à jour le centre d'informations se présente à ce jour en deux versions:

- La version 1.0
- La version 1.1.

Ces deux versions existent en deux variantes :

- Une variante monoposte (sous Windows) destinée à un usage local du CI
- Une variante réseau(Windows ou linux) qui permettra l'utilisation du CI par le réseau(Client-Serveur).

La version 1.1 diffère de la version 1.0 du fait que dans la version 1.1 en plus des traitements des requêtes sur l'affichage d'une information, d'une liste d'informations et des instances d'un descripteur, il existe une quatrième fonctionnalité qui permet d'afficher les descripteurs associés à un descripteur donné(ex les groupes associés à un composant). De plus dans la version 1.1 le descripteur « motscles » à été retiré et le lien entre les descripteurs groupes et composants a été pris en compte dans le code.
Enfin la version 1.1 est destinée à recevoir de nouvelles fonctionnalités à venir.
A ce jour seule la version 1.0 a été déployée en réseau sur un serveur. Les deux versions ayant été auparavant testées en utilisation monoposte avec succès.
Des informations complémentaires sur le déploiement du CI sont disponibles dans la section déploiement plus loin dans ce chapitre.

Par ailleurs une autre variante du CI intégrant des techniques de *Datamining* [7]et ayant fait l'objet du thème de DEA de Cheick Bâ (Etudiant sénégalais à Montpellier) a été aussi développée puis installée en réseau. Cette variante développée par Cheick Bâ à partir de la version 1.0 ajoute au CI une procédure d'apprentissage qui lui permettra à terme de connaître entre autres les informations les plus demandées, les relations entre informations et descripteurs pour ensuite guider l'utilisateur dans sa navigation en lui faisant des suggestions.

Le déploiement

Le déploiement du CI varie selon la variante.
Le déploiement de la variante monoposte destinée à marcher sur une seule machine(Windows) a été presque entièrement automatisé grâce notamment à la création d'un exécutable d'installation avec le logiciel *InstallMaker*. Avec cet exécutable le seul préalable à l'installation du CI est la présence de Microsoft Access(97 ou 2000), tous les autres logiciels nécessaires(java, tomcat, fichiers jar java, code binaire du CI) ainsi que certaines configurations(source de données ODBC/JDBC, raccourcis) sont ajoutés pendant l'installation.
Quant au déploiement de la variante réseau c'est une opération longue et complexe. En effet il s'agit ici d'installer puis de configurer un certain nombre de logiciels de différents constructeurs afin de les faire marcher ensemble. Une documentation technique est disponible dans laquelle de larges informations sur le déploiement du CI en réseau sont détaillées.

En outre un fichier xml (Parametres.xml) permet de paramétrer le CI. Dans ce fichier on peut préciser la base de données à utiliser, les fichiers de formatage ainsi que tout autre paramètre futur de l'application.
Exemple de fichier Parametre.xml

```
    <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Parametres SYSTEM "parametres.dtd">
<!--Ce fichier contient les parametres du centre d'informations-->
<Parametres>
    <!--Chemin vers la source de données utilisée par le centre d'informations-->
    <BD>jdbc:odbc:ci11</BD>
    <!--<BD>jdbc:mysql://nserver/ci</BD>-->
    <!--Pilote JDBC ou JDBC-ODBC de la base de données utilisée par le centre d'informations-->
    <!--<PiloteJDBC>org.gjt.mm.mysql.Driver</PiloteJDBC>-->
    <PiloteJDBC>sun.jdbc.odbc.JdbcOdbcDriver</PiloteJDBC>
    <User>root</User>
    <!-- la chaine "null" si absence de mot de passe-->
    <Passwd>null</Passwd>
</Parametres>
```

Diagramme de déploiement du CI(variante réseau)

[7] *(exploitation des données)*
Se dit de techniques logicielles permettant de faire apparaître de nouvelles relations utiles entre des données extraites des gigantesques bases de données .

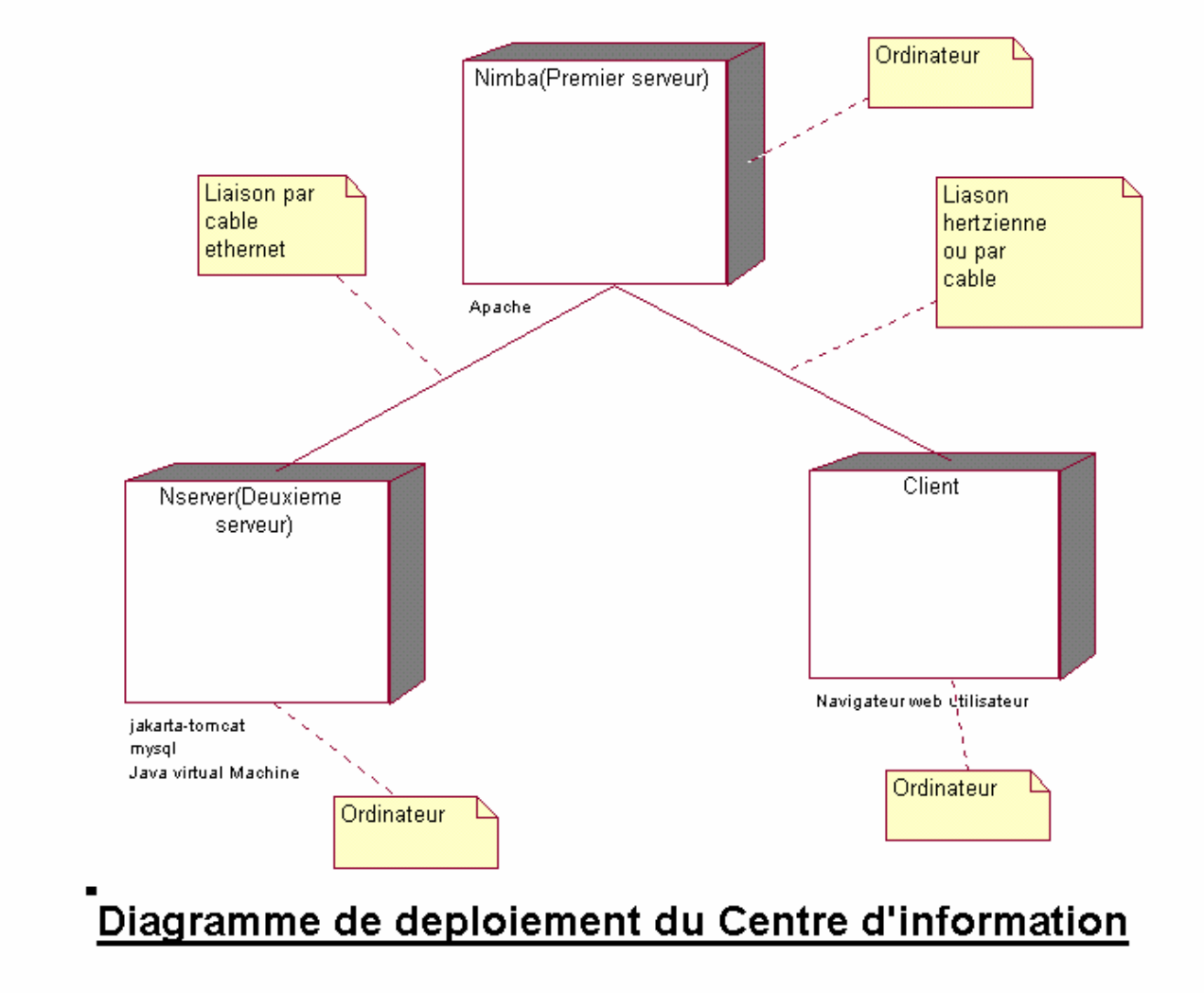


Diagramme de deploiement du Centre d'information

Quelques difficultés rencontrées

Au nombre des difficultés rencontrées trois particulièrement seront exposées ici.

- Encodage des URL : Tomcat utilisé seul (sans apache) permet de gérer de façon transparente les url comportant des caractères accentués. Dans la variante réseau du CI tomcat a été associé à apache(le serveur web le plus populaire sous Linux) et ce pour deux raisons :

D'une part apache gère mieux le contenu statique que tomcat et d'autre part la combinaison de tomcat et d'apache permet de greffer le CI au site web(statique) du CNSHB.
Cependant Apache ne gérant pas automatiquement les urls avec des caractères accentués, il a fallu encoder les informations extraites de la base de données. L'encodage en lui-même ne pose pas de problème en java c'est l'utilisation de xml avec java qui en pose. En effet il a fallu encoder des données xml ce qui implique de créer pour chaque élément xml son équivalent encodé. Cette méthode permet de résoudre le problème d'url mais doit figurer en bonne place dans les améliorations possibles du CI car alourdit énormément les documents xml générés.

- Le déploiement du CI sur un serveur est une tâche longue et compliquée pour laquelle une alternative est donnée dans la section améliorations possibles plus loin dans ce chapitre.

- L'exportation des tables de MS Access vers Mysql :Etant donné que la saisie des données du CI se fait dans Access, il a fallu après trouver une méthode d'exportation des tables de Access vers Mysql. La plupart des méthodes classiques d'exportation montraient des insuffisances ce qui nous a amené à créer une routine en Visual Basic permettant l'exportation de façon efficace.

Améliorations possibles

Comme tout logiciel le Centre d'informations est loin d'être parfait ! Des améliorations doivent permanemment y être apportées si besoin. Parmi ces améliorations on peut citer:

- L'amélioration de la gestion des exceptions :Les exceptions sont des erreurs qui surviennent pendant l'exécution d'un programme. Les routines mises en place pour gérer les exceptions du CI doivent être améliorées afin d'aboutir à une gestion efficace des erreurs.

- Le déploiement du CI pourra être considérablement amélioré avec l'utilisation d'un serveur d'application commerciale comme Weblogic de Bea WebSphere de IBM Oracle AS d'Oracle. Ces produits intègrent toute l'api du j2ee(java 2 enterprise édition)(servlet, xml, ejb, javamail etc.), un serveur de base de données, des drivers JDBC, un serveur web etc. De plus ces produits intègrent d'autres fonctionnalités avancées telles la gestion de la sécurité et un environnement intégré et homogène qui permet d'améliorer la vitesse d'exécution des applications ainsi que leur déploiement.

- Une solution aussi doit être trouvée au niveau de l'utilisation de SQL avec les dates. En effet l'impossibilité de trouver une expression standard SQL pour extraire des dates dans des tables Access nous a amené à faire deux codes distincts pour la base Access et la base Mysql. Il serait préférable dans l'avenir de trouver une expression standard utilisable par les deux bases de données.

- La résolution du problème d'encodage exposé plus haut dans la section Difficultés rencontrées allégera le code et la compréhension des documents XML générés.

Conclusion :

La réalisation du centre d'informations a fait appel à des technologies d'apparition récente et en pleine expansion : Java , xml et à une méthodologie de programmation orientée objet ; ce qui est un gage de sa pérennité. D'un autre coté cela m'a permis d'avoir une certaine connaissance dans l'utilisation de ces nouvelles technologies et des outils associés(serveur web, serveur d'application, etc.) .

Des mises à jour sont constamment apportées au centre d'informations pour le rendre suffisamment fiable et robuste afin d'en faire un logiciel générique qui pourrait être adapté à d'autres domaines de connaissances.

Annexes

Les fichiers de formatage XSL

Pour l'affichage d'une Information

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <xsl:output method="html" indent="yes" doctype-public="-//W3C//DTD HTML 4.0//EN"/>
    <xsl:template match="/">
        <!--balise correspondant à la racine du document -->
        <html>
            <head>
                <title>CNSHB - IRD - Centre d'information (CI)</title>
                <!--Le titre du document -->
            </head>
            <body background="icones/FIB14.gif" bgcolor="#E1E1FF">


                <xsl:apply-templates/>
                <!--boucle implicant toutes les balises templates-->
                <p></p><p></p><table border="0" width="100%">
            <tr>
                <td width="50%">
                    <a href="index.htm" target="_top" alt="retour à la page d'accueil du CI">
                        <img border="1" src="icones/bci.gif" width="40" height="40"/>
                    </a>
                    <img src="icones/pttEspaceBlanc.gif"></img>
                    <a href="note_sur_le_logiciel.1a.htm" alt="note sur le logiciel">
                        <img border="1" src="icones/bmanuel2.gif" width="40" height="40"/>
                    </a>

                </td>
                <td width="50%">
                    <p align="right">
                     <img src="icones/pttEspaceBlanc.gif"></img>
                    <img src="icones/pttEspaceBlanc.gif"></img>
                    <img src="icones/pttEspaceBlanc.gif"></img>
                    <img src="icones/pttEspaceBlanc.gif"></img>
                    <A href="http://www.cnshb.org.gn/" target="_top">
                            <img border="1" height="40" src="icones/bcnshb2.gif" width="57"/>
                        </A>
                <img src="icones/pttEspaceBlanc.gif"></img>
                            <A href="http://www.ird.fr/" target="_top">
                                <img border="1" height="40" src="icones/bird.gif" width="57"/>
                            </A>
            <img src="icones/pttEspaceBlanc.gif"></img>
                            <A href="http://www.ambafrance-gn.org/Cooperation/scac.html" target="_top">
                                <img border="1" height="40" src="icones/brf.gif" width="57"/>
                            </A>
            <img src="icones/pttEspaceBlanc.gif"></img>
                            <A href="http://europa.eu.int/comm/development/sector/environment/b7-
6200budgetline/index.htm" target="_top">
                                <img border="1" height="40" src="icones/bCE.gif" width="57"/>
                            </A>
                    </p>
                </td>
            </tr>
        </table>
        </body>
        </html>
    </xsl:template>

    <xsl:template match="information">
        <!--trouve toutes balises "information" contenues dans la racine-->
        <div align="center">
            <center>
                <table border="2" height="1" cellpadding="20" cellspacing="8">
                    <!--création d'une table pour le titre-->
                    <tr>
                        <td valign="middle" align="left" bgcolor="#BFBFFF" height="1">
                            <p align="center">
                                <font face="Abadi MT Condensed Light" size="4">
                                    <b>
                                        <xsl:value-of select="titre"/>
                                    </b>
                                    <br/>
```

```
                                    <!--récupère les données des balises "titre",  "sujet" les centre dans la 2e colonne de la table-->
                                    <xsl:value-of select="sujet"/>
                                </font>
                            </p>
                        </td>
                    </tr>
                </table>
            </center>
        </div>
        <font size="3">
            <pre>
                <xsl:value-of select="commentaire"/>
            </pre>
        </font>
        <!--il récupère toutes les balises "support" contenues dans "information"-->
            <p/>
            <p/>
        <xsl:apply-templates select="supports"/>
        <!--<br><font size="1">   (note: cliquer sur l'icône ouvre dans le cadre, cliquer sur le texte ouvre en pleine page)</font></br>-->
        <hr/>
        <font face="Arial Narrow">- dimension:
            <a href="GetList.html?type=Dimension&amp;contenu={dimension}" target="_self">
                <xsl:value-of select="dimension"/>
            </a>
            <br>
                <!--création d'un lien sur la donnée de la balise "dimension"-->
             - mots-clés: <xsl:apply-templates select="date"/>,
            <!--recherche la balise "date" contenue dans la balise "information"-->
                <xsl:apply-templates select="lieu"/>,
                <!--recherche la balise "composants" contenue dans la balise "information"-->
                <xsl:apply-templates select="groupes"/>,</br>
            <!--recherche la balise "groupes" contenue dans la balise "information"-->
             - thèmes: <xsl:apply-templates select="composants"/>
            <!--recherche la balise "mots_clés" contenue dans la balise "information"-->
            <!--recherche la balise "niveau_de confiance" contenue dans la balise "information"-->
            <!--<xsl:apply-templates select="reference"/>-->
            <!--recherche la balise "source" contenue dans la balise "information"-->
            <hr/>
            <u>source:</u>
            <xsl:apply-templates select="source"/>. <xsl:value-of select="referencesource"/>
            <br>
                <u>information:</u> n°<xsl:value-of select="@id"/>
                <!--récupère la donnée de l'attribut id de l'information-->
                - mise à disposition: <xsl:apply-templates select="datesaisie"/>
                - rédacteur de l'information: <xsl:apply-templates select="auteurinfo"/> - niveau de confiance: <xsl:apply-templates select="niveauconfiance"/>
            </br>
        </font>

        </xsl:template>
    <xsl:template match="date">
        <a href="GetList.html?type=date&amp;contenu={@elem}">
            <!--selection et création d'un lien sur la donnée de la balise "date"-->
            <xsl:value-of select="."/>
        </a>
    </xsl:template>
    <xsl:template match="lieu">
        <a href="GetList.html?type=lieu&amp;contenu={@elem}">
            <!--selection et création d'un lien sur la donnée de la balise "lieu"-->
            <xsl:value-of select="."/>
        </a>
    </xsl:template>
    <xsl:template match="niveauconfiance">
        <a href="GetList.html?type=niveauconfiance&amp;contenu={@elem}">
            <!--selection et création d'un lien sur la donnée de la balise "niveau_de confiance"-->
            <xsl:value-of select="."/>
        </a>
    </xsl:template>
    <xsl:template match="auteurinfo">
        <a href="GetList.html?type=auteurinfo&amp;contenu={@elem}">
            <!--selection et création d'un lien sur la donnée de la balise "niveau_de confiance"-->
            <xsl:value-of select="."/>
        </a>
```

```
    </xsl:template>
    <xsl:template match="datesaisie">
        <a href="GetList.html?type=datesaisie&amp;contenu={@elem}">
            <!--selection et création d'un lien sur la donnée de la balise "niveau_de confiance"-->
            <xsl:value-of select="."/>
        </a>
    </xsl:template>
    <xsl:template match="source">
        <a href="GetList.html?type=source&amp;contenu={@elem}">
            <!--selection et création d'un lien sur la donnée de la balise "source"-->
            <xsl:value-of select="."/>
        </a>
    </xsl:template>
    <xsl:template match="composants">
        <xsl:for-each select="composant">
            <a href="GetListdesc.html?type=composant&amp;contenu={@elem}">
                <!--selection et création d'un lien sur la donnée dechaque balise balise "composant"-->
                <xsl:value-of select="."/>
            </a>, 
         </xsl:for-each>
    </xsl:template>
    <xsl:template match="motscles">
        <xsl:for-each select="motcle">
            <a href="GetList.html?type={@type}&amp;contenu={@elem}">
                <!--selection et création d'un lien sur la donnée de chaque balise "mots_clés"-->
                <xsl:value-of select="."/>
            </a>, 
        </xsl:for-each>
    </xsl:template>
    <xsl:template match="groupes">
        <xsl:for-each select="groupe">
            <a href="GetList.html?type=groupe&amp;contenu={@elem}">
                <!--selection et création d'un lien sur la donnée de chaque balise "groupe"-->
                <xsl:value-of select="."/>
            </a>, 
        </xsl:for-each>
    </xsl:template>
    <xsl:template match="supports">
        <td valign="middle" align="center" height="1">
            <!--placement des données de la 1ère colonne de la table-->
            <xsl:for-each select="support">
                <!--récupère et crée des liens sur les fichiers et logos de chaque balise "support"  contenues dans
"information"-->
                <a href="infos/{@fichier} " target="_self">
                    <img border="2" src="icones/{@logo}" width="59" height="58" align="texttop"/>
                </a>
                <a href="infos/{@fichier} " target="_top">
                    <xsl:value-of select="."/>
                </a> 
            </xsl:for-each>
        </td>
    </xsl:template>
</xsl:stylesheet>
```

Pour l'affichage d'une liste d'informations

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:fo="http://www.w3.org/1999/XSL/Format">

    <xsl:output method="html" indent="yes" doctype-public="-//W3C//DTD HTML 4.0//EN"/>

    <xsl:template match="/">
        <html>
            <head>
                <title>CNSHB - IRD: Centre d'Information (CI) sur les pêches en Guinée </title>
            </head>
            <body background="icones/FIB14.gif" bgcolor="#E1E1FF">

    <!--Option d'affichage-->
        <xsl:choose>
```

```
        <!--Cas afichage de toutes les informations-->
      <xsl:when test="liste/@type ='tous'"><h2 align="center"><xsl:value-of select="liste/@nombre"/>  Informations
présentes en tout  dans le Centre d'informations</h2>
 <font size="4"><ul>
<xsl:for-each select="liste/information">
 <li><xsl:value-of select="@id"/> <a href='GetInfo.html?id={@id}'><xsl:value-of select="titre"/></a> (<xsl:value-of
select="sujet"/>)</li>
</xsl:for-each></ul></font>
</xsl:when>

<!--Cas afichage des informations appartenant à un descripteur donné-->
<xsl:otherwise>
           <h2 align="center"><xsl:value-of select="liste/@nombre"/>  Informations trouvées sur <xsl:value-of
select="liste/@type"/> <xsl:value-of select ="liste/ @contenu"/></h2>
                    <font size="4">
                              <ul><xsl:apply-templates select='liste/information'/></ul>

              </font></xsl:otherwise></xsl:choose>
              </body>
       </html>
    </xsl:template>

           <xsl:template match='information'>

                  <li><a href='GetInfo.html?id={@id}'><xsl:value-of select="titre"/></a>
                   (<xsl:value-of select="sujet"/>)
                  </li>

          </xsl:template>

</xsl:stylesheet>
```

Pour l'affichage des instances d'un descripteur

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <xsl:output method="html" indent="yes" doctype-public="-//W3C//DTD HTML 4.0//EN"/>
    <xsl:template match="liste">
        <html>
            <head></head>
            <body background="icones/FIB14.gif" bgcolor="#E1E1FF">
                <h2 align="center"><xsl:value-of select="@nombre"/>  Instances de  <xsl:value-of
select="@type"/></h2>
                <blockquote>
                    <ul>
                        <xsl:for-each select="instance">
                            <li>
                                <a href="GetList.html?type={champ1/@type}&amp;contenu={champ1/@contenu}"
target="principal">
                                <xsl:value-of select="champ1"/>
                                </a> 
                                <xsl:if test="champ1/@type='source'">
                                    <xsl:value-of select="champ2"/>
                                </xsl:if>
                            </li>
                        </xsl:for-each>
                    </ul>
                </blockquote>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

Pour l'affichage d'une liste de descripteurs(disponible seulement dans la version 1.1)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
```

```
<xsl:output method="html" indent="yes" doctype-public="-//W3C//DTD HTML 4.0//EN"/>

    <xsl:template match="/">
        <html>
            <head>
                <title>CNSHB - IRD: Centre d'Information (CI) sur les pêche en Guinée </title>
            </head>
            <body background="icones/FIB14.gif" bgcolor="#E1E1FF">

                <h2 align="center"><xsl:value-of select="liste/@nombre"/>  Groupe(s) trouvé(s) sur <xsl:value-of
select="liste/@type"/> <xsl:value-of
                        select ="liste/ @contenu"/></h2>
                    <ul><xsl:apply-templates select='liste/descripteur'/></ul>

            </body>
        </html>
    </xsl:template>
        <xsl:template match='descripteur'>
            <li><a href='GetList.html?type={@type}&amp;contenu={@contenu}'><xsl:value-of
select="contenu"/></a>
            </li>
        </xsl:template>
</xsl:stylesheet>
```

Bibliographies et References

Precis de Genie logiciel M.C Gaudel/B.Marre/F.Schlienger/G.Bernot
Edition Masson

Rapport de cedric Babault disponible sur le net à l'adresse :www.cedric.babault.free.fr/rapport

Rapport de Guerin(DEA) disponible sur le net à l'adresse : http://www.irisa.fr/dea-genomique/promo2000/Guerin.pdf

Rapport de Misako Ito(Membre de l'équipe de développement)