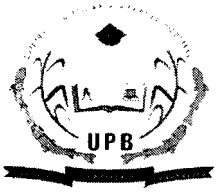
BURKINA FASO
UNITE-PROGRES-JUSTICE

MINISTERE DES ENSEIGNEMENTS SECONDAIRE, SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE POLYTECHNIQUE DE BOBO-DIOULASSO

ECOLE SUPERIEURE D'INFORMATIQUE



E.S.I

MEMOIRE DE FIN DE CYCLE

en vue de l'obtention du

DIPLOME D'INGENIEUR DE CONCEPTION EN INFORMATIQUE

THEME : Gestion intégrée des communautés religieuses et des établissements des Frères des Ecoles Chrétiennes du Burkina/Niger.

Présenté par :

SOME Iyo Ibso Maxence

Maître de stage : Frère Sylvain CONSIMBO

Directeur de mémoire : Dr Loé SANOU

Enseignant-chercheur

JANVIER 2011

N° : .....-2010/CICI3

DEDICACE

A mes parents,

Qui ont veillé sur moi.

REMERCIEMENTS

Ce travail de conception n'aurait pas été envisageable sans l'autorisation et le soutien des Frères des Ecoles Chrétiennes, ma famille religieuse, plus particulièrement du supérieur provincial Janvier DEMBELE.

Il n'aurait pu aboutir sans la disponibilité et la diligente bienveillance de celui qui en fut le superviseur, le docteur Loé SANOU.

Il est en outre redevable aux observations et encouragements du maître de stage, Frère Sylvain CONSIMBO.

A chacun, je voudrais exprimer ici ma vive reconnaissance.

Tables des matières

TABLE DES ILLUSTRATIONS

LISTE DES TABLEAUX

INTRODUCTION

Dans une ère marquée par un progrès technologique de plus en plus rapide, l'exploitation et l'organisation fiables de l'information sont devenues des activités de base sur lesquelles comptent différents organismes pour leurs réussites.

Ayant pris conscience de cette importance, de nombreuses entreprises et organisations ont compris qu'il leur est devenu vital d'adopter les nouvelles technologies pour exploiter au mieux l'information qu'elles détiennent. Mais elles avaient un défi majeur à affronter qui est l'harmonisation de ces technologies avec les systèmes hérités.

Dans cette optique, la congrégation religieuse des Frères des Ecoles Chrétiennes d'Afrique de l'Ouest s'est fixée l'objectif d'informatiser leur système de gestion des ressources humaines. C'est dans ce cadre que s'inscrit notre projet intitulé « Conception d'une application web pour la gestion des ressources humaines des Frères des Ecoles Chrétiennes du Burkina/Niger ».

Le présent rapport décrit notre stage au sein de cette congrégation religieuse. Il comporte deux grandes parties :

La première partie est le contexte général regroupant trois chapitres. Le premier chapitre présente d'abord l'organisme d'accueil, ensuite le deuxième chapitre définit la problématique du stage. Enfin le troisième chapitre propose une méthodologie à suivre pour la conception de l'application.

La deuxième partie est la conception du logiciel qui comporte deux chapitres. Le premier chapitre est la définition des contraintes fonctionnelles. Ce chapitre est appuyé par une étude des contraintes techniques en deuxième chapitre.

Finalement, une conclusion récapitule le travail accompli dans le cadre de ce stage et présente des axes d'amélioration et de poursuite du développement de l'application.

PREMIERE PARTIE : CONTEXTE GENERAL

CHAPITRE I : PRESENTATION DES FRERES DES ECOLES CHRETIENNES

1. PRESENTATION

A Reims, en France, vers 1688, le jeune prêtre Jean-Baptiste De La Salle regroupe quelques maîtres d'école. Il s'associe avec eux sous le nom de « Frères des Ecoles Chrétiennes », leur donne une Règle de vie et des principes éducatifs, les forme à se gouverner eux-mêmes. A sa mort en 1719, ils sont une centaine de Frères en France.

Grâce au personnel venu de France et d'Espagne, des œuvres scolaires ont vu le jour au Burkina Faso depuis 1948, date de leur arrivée:

- Le collège de Toussiana
- Le collège De la Salle à Ouagadougou
- Le collège de Tounouma à Bobo-Dioulasso
- Le collège Charles Lwanga à Nouna
- Le CAPA (Centre d'Apprentissage et de Promotion Artisanale) à Nouna
- Le collège Pierre Kula à Diébougou
- Le CITA (Centre d'Initiation Technique et Artisanale) à Diébougou
- Le lycée professionnel Issa Béri à Niamey
- Le lycée Badenya à Ouagadougou.
- Le centre d'apprentissage au métier de l'agriculture à Bérégadougou
- Le collège de Kongoussi

Cet ensemble est appelé District d'Afrique de l'Ouest et compte 48 Frères.

2. ORGANISATION DES FRERES DES ECOLES CHRETIENNES AU BURKINA FASO /NIGER

Le Chapitre du District est l'instance supérieure qui donne mandat au Frère Supérieur du District pour mettre en application ses orientations.

Le Frère supérieur du District, appelé Frère Visiteur, est assisté par un Conseil de District mandaté également par le Chapitre du District pour l'entourer de conseils.

Le Frère Visiteur est chargé du gouvernement du District c'est-à-dire des Frères, des communautés, des œuvres. Il est le représentant et le garant de toutes les personnes et de toutes les activités du District devant l'Eglise et l'Etat.

Il met en place des services et commissions qui l'assistent sur le plan technique et opérationnel.

L'organigramme, illustré à la figure 1, permet de constater que le Frère supérieur, le Frère Visiteur, exécute les décisions du chapitre de District. Le Frère Visiteur se fait aider par le conseil de District et des consultants pour animer les services qui sont sous son autorité.

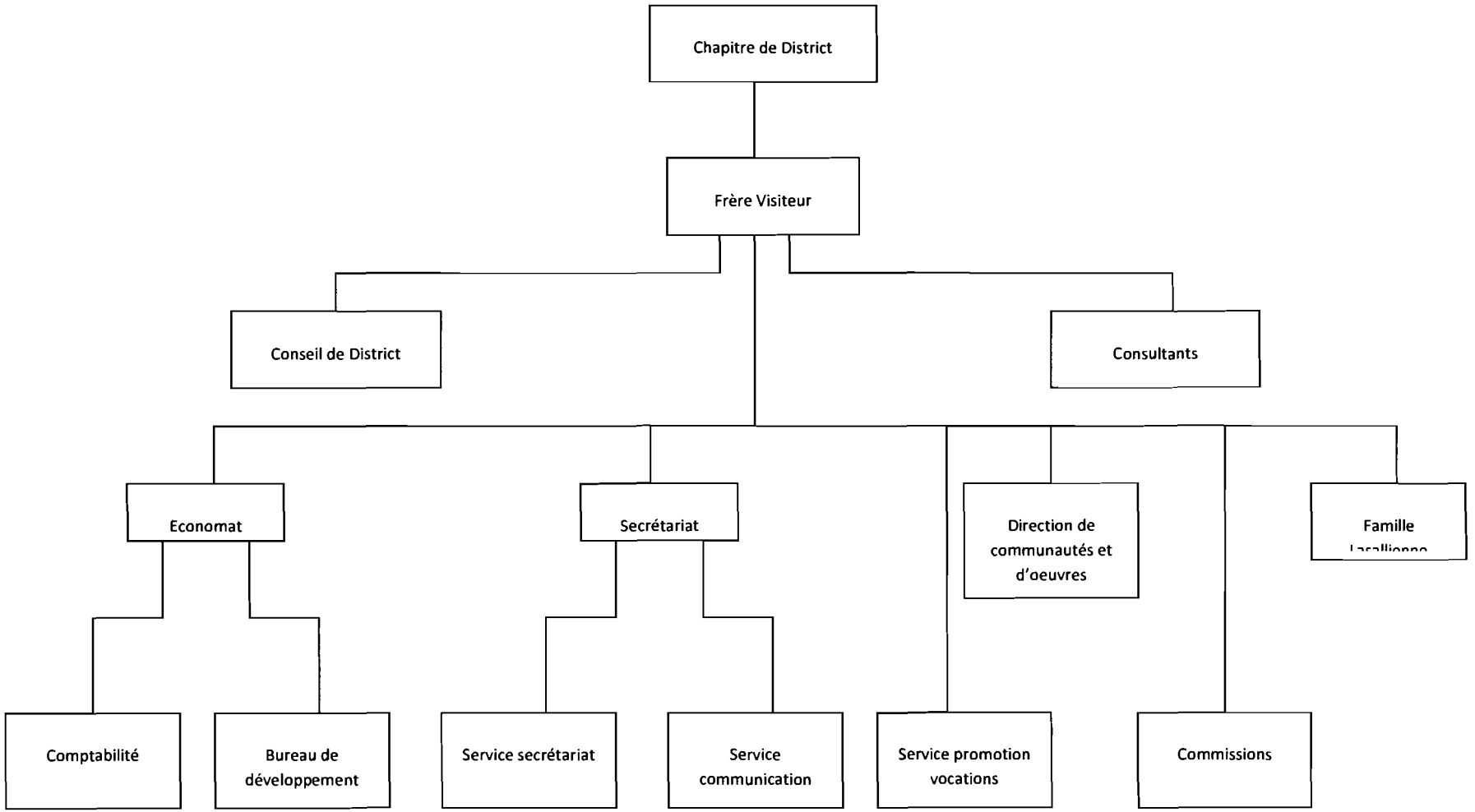


Figure 1: Organigramme du District d'Afrique de l'Ouest

CHAPITRE II : PROBLEMATIQUE

1. PROBLEME A RESOUDRE

Le District d'Afrique de l'Ouest rencontre actuellement des difficultés dues au manque d'ordre dans l'archivage des documents de la congrégation. Il n'y a pas de système d'archivage centralisé de documents. Cela complique les recherches d'informations historiques sur les Frères et leurs œuvres. Ce manque d'informations centralisées concerne aussi bien l'administration du District que les établissements et les communautés religieuses.

Actuellement, au niveau des établissements, les rapports de rentrée scolaire, les rapports de trimestres, les bilans de fins d'année, les rapports d'activités diverses, l'historique (récit de quelques événements journaliers marquants), etc. sont des documents indispensables au travail administratif du centre de la congrégation. Cependant, ces documents restent archivés dans les établissements et parfois introuvables. Ces documents, arrivés à l'administration centrale de la congrégation, sont traités et généralement dupliqués pour archivage.

Au niveau des communautés religieuses, certaines informations doivent être envoyées au gouvernement de la congrégation. Ce sont les projets communautaires, les fiches d'employés, les rapports d'évaluation... Tout ceci est aussi traité au sein de l'administration centrale.

Cependant, entre communautés religieuses et entre établissements scolaires, les flux d'informations circulant accusent de la lenteur due aux distances et à la difficulté de la recherche.

Le problème de gestion des archives et parfois le manque d'informations ont pour conséquences :

- La perte de documents éparpillés dans les communautés,
- La perte des traces d'événements notables,
- Un temps de recherche documentaire très élevé,
- Une impossible centralisation des informations au sein du District.

L'agrandissement prochain du District, qui couvrira le Burkina Faso, le Niger, le Togo, le Bénin, la Côte d'Ivoire et la Guinée (Conacry) ajoute à l'urgence et à l'importance de cet état de fait.

2. RESULTATS ATTENDUS

Tout d'abord, ce projet permettra une meilleure organisation des archives de l'institution. En effet, cette application web sera un appui à l'organisation manuelle des documents. Cela participera de la diminution de la perte de documents.

Ensuite, la permanence du stockage d'informations est une des visées de cette application. Car la volatilité actuelle des informations ajoute aussi à la difficulté d'organisation.

Enfin, un temps de recherche optimisé, fruit de la centralisation, est une raison d'être de cette réorganisation administrative voulue par la congrégation des Frères.

CHAPITRE III : METHODE

1. PRESENTATION DES METHODES

Plusieurs méthodes sont en vogue en ce qui concerne les processus de développement d'applications informatiques. La figure ci-dessous présente un tableau comparatif de quelques méthodes.

Tableau 1: Tableau comparatif de processus de développement

	Description	Points forts	Points faibles
Cascade	• Propose de dérouler les phases projet de manière séquentielle • **Cité pour des raisons historiques**	• Distingue clairement les phases projet	• **Non itératif** • Ne propose pas de modèles de documents
RUP Rational Unified Process	• Promu par Rational. • **Le RUP est à la fois une méthodologie et un outil prêt à l'emploi** (documents types partagés dans un référentiel Web) • Cible des projets de plus de 10 personnes	• Itératif • **Spécifie le dialogue entre les différents intervenants du projet : les livrables, les plannings, les prototypes...** • Propose des modèles de documents, et des canevas pour des projets types	• **Coûteux** à personnaliser : batterie de consultants • Très axé processus, au détriment du développement : peu de place pour le code et la technologie
XP eXtreme Programming	• **Ensemble de « Bests Practices » de développement** (travail en équipes, transfert de compétences ...) • Cible des projets de moins de 10 personnes	• Itératif • **Simple à mettre en œuvre** • Fait une large place aux aspects techniques : prototypes, règles de développement, tests... • **Innovant:** programmation en duo, kick-off matinal debout ..	• Ne couvre pas les phases en amont et en aval au développement : capture des besoins, support, maintenance, tests d'intégration .. • **Elude la phase d'analyse**, si bien qu'on peut dépenser son énergie à faire et défaire • **Assez flou dans sa mise en œuvre:** quels intervenants, quels livrables ?
2TUP Two Track Unified Process	• **S'articule autour de l'architecture** • **Propose un cycle de développement en Y** • Détaillé dans « UML en action » (voir références) • Cible des projets de toutes tailles	• Itératif • **Fait une large place à la technologie et à la gestion du risque** • Définit les profils des intervenants, les livrables, les plannings, les prototypes	• **Plutôt superficiel sur les phases situées en amont et en aval du développement :** capture des besoins, support, maintenance, gestion du changement... • Ne propose pas de documents types

A l'analyse du tableau 1, deux grandes subdivisions se distinguent dans les méthodes de conduite d'un projet informatique : Les méthodes séquentielles et les méthodes itératives.

1.1. LES METHODES SEQUENTIELLES

Elles consistent à découper le projet informatique en phases distinctes sur le principe du non-retour. Les méthodes séquentielles furent développées dans les années 1970 par W. Royce.

La plus connue de ces méthodes est le modèle de développement en cascade.

L'avantage de ces méthodes séquentielles est de proposer au fur et à mesure, une démarche de réduction des risques en minimisant l'impact des incertitudes. Un autre avantage de ce modèle est l'obligation de bonne gestion du temps et des ressources car il y a impossibilité de travail en parallèle. En plus, les méthodes séquentielles sont performantes dans les projets de petites tailles.

Au niveau des inconvénients, on peut noter que ce modèle ne représente pas la réalité. Dans la réalité il est possible de revenir en arrière, car des erreurs peuvent être découvertes. Il y a des risques de ne découvrir des erreurs qu'à la fin d'où la nécessité de refaire toutes les étapes. Leur principal défaut est donc d'exclure très tôt l'utilisateur de l'application dès la phase de conception car conception trop technique. Le contrôle qualité ne survient seulement qu'en fin de projet. Il apparaît donc à l'horizon le risque de refus de la recette utilisateur.

La méthode en cascade contient beaucoup de variantes telles que la méthode en V, la méthode en W, etc.

1.2. LES METHODES ITERATIVES

Au sein des méthodes dites itératives, il est illustré dans le tableau 1 les méthodes RUP, XP et 2TUP.

1.2.1. LES METHODES XP

La méthode xp se fonde sur le principe de la polyvalence et une approche pluridisciplinaire des membres du projet informatique. La méthode tend à minimiser l'impact des évolutions des besoins en cours de projet.

Néanmoins, cette méthode s'appuie le plus souvent sur un solide atelier de génie logiciel[1] pour garantir un passage rapide du concept à la mise en œuvre. C'est le risque des méthodes itératives que de produire un logiciel difficile à maintenir après plusieurs évolutions.

Des variantes viennent enrichir cette méthode. Ce sont : RAD, ASD (Adaptative Software Development), Scrum, Crystal, FDD (Feature Driven Development), AM (Agile Modeling)...

1.2.2. LES METHODES RUP, 2TUP

Ces méthodes consistent en la séparation de l'étude d'architecture de celle de l'étude fonctionnelle afin de paralléliser au maximum les tâches. Elles permettent donc de prendre en compte les problèmes d'architecture dès le début du projet.

Cependant la maintenance, même assouplie par la séparation de l'architecture et du fonctionnel, reste difficile. Aussi, ces méthodes sont caractérisées par une documentation nécessaire et abondante à chaque étape de développement. Cela participe de la facilitation de la maintenance mais risque d'être un grave inconvénient en cas d'insuffisance ou d'absence de ces documents.

[1] Voir glossaire

2. CHOIX DE LA METHODE

2.1. LA METHODE 2TUP

Après description des différentes méthodes faisant ressortir leurs avantages et leurs inconvénients, la méthode 2TUP parait intéressante pour les raisons suivantes :

- Peu exigeante par rapport à la disponibilité du client,
- La séparation des besoins fonctionnels/Architecturaux,
- La possibilité de modélisation graphique (UML),
- Le besoin de documentation de la conception pour faciliter la maintenance future.

Le processus **2TUP** insiste sur la non-corrélation initiale des aspects fonctionnel et technique. « 2 Tracks » signifie littéralement que le processus suit deux chemins. Il s'agit des chemins « fonctionnel » et « d'architecture technique », comme le présente la figure 2.

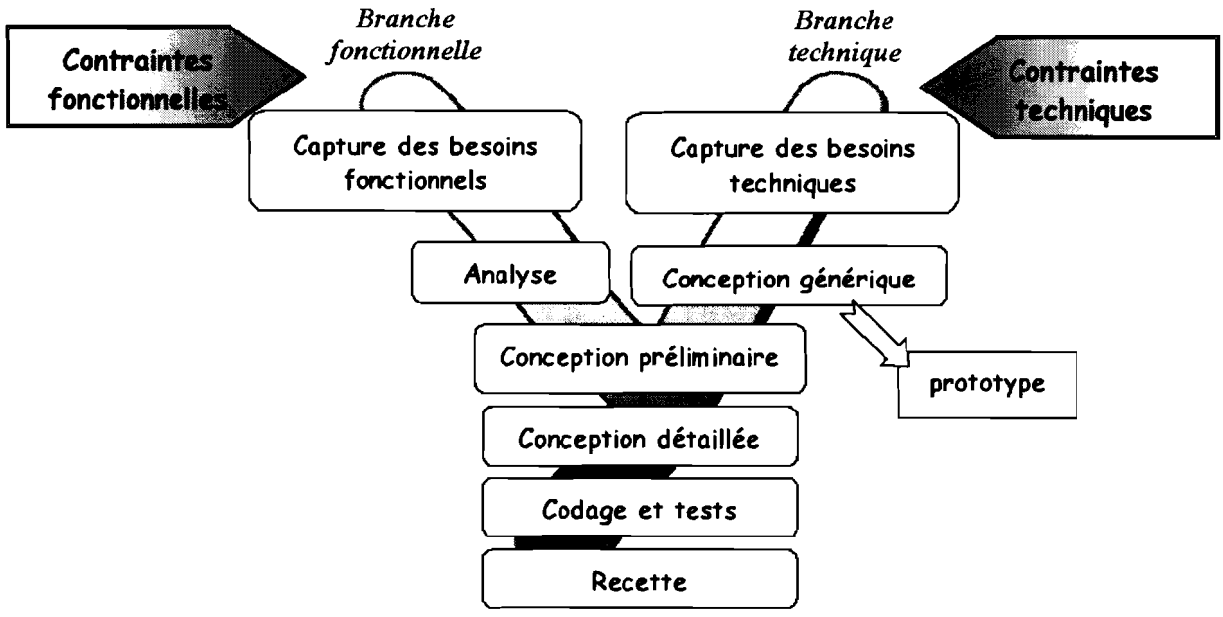


Figure 2: Le processus 2TUP

Les deux branches d'étude fusionnent ensuite pour la conception du système, ce qui donne la forme d'un processus de développement en Y. La séparation initiale permet à la fois de capitaliser la connaissance métier sur la branche gauche et de réutiliser un savoir-faire technique sur la branche droite.

La branche fonctionnelle comporte :

- la capture des besoins fonctionnels, qui produit un modèle des besoins focalisé sur le métier des utilisateurs. Cette capture des besoins permet d'éviter au plus tôt le risque de produire un système inadapté aux utilisateurs.
- l'analyse, qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en termes de métier. Les résultats de l'analyse ne dépendent d'aucune technologie particulière.

La branche technique comporte :

- la capture des besoins techniques, qui recense toutes les contraintes et les choix dimensionnant la conception du système. Les outils et les matériels sélectionnés ainsi que la prise en compte de contraintes d'intégration avec l'existant conditionnent généralement cette capture ;
- la conception générique, qui définit ensuite les composants nécessaires à la construction de l'architecture technique. Cette conception est complètement indépendante des aspects fonctionnels. Elle a pour objectif d'uniformiser et de réutiliser les mêmes mécanismes pour tout un système. L'architecture technique construit le squelette du système informatique. L'importance de sa réussite est telle qu'il est conseillé de réaliser un prototype pour assurer sa validité.

La branche du milieu comporte :

- la conception préliminaire, qui représente une étape délicate, car elle intègre le modèle d'analyse dans l'architecture technique de manière à tracer la cartographie des composants du système à développer ;
- la conception détaillée, qui étudie ensuite comment réaliser chaque composant ;
- l'étape de codage, qui produit ces composants et teste au fur et à mesure les unités de code réalisées ;
- l'étape de recette, qui consiste enfin à valider les fonctions du système développé.

2.2. CHOIX DE LA MODELISATION AVEC UML

Il est difficile d'envisager le processus 2TUP sans recourir à UML comme support. Le recours à la modélisation est depuis longtemps une pratique indispensable au développement, car un modèle est prévu pour anticiper les résultats du développement.

Un modèle est, en effet, une abstraction du résultat, dont le but est de collecter ou d'estimer les informations d'un système.

UML s'articule autour de 13 types de diagrammes différents, chacun d'eux étant dédié à la représentation des concepts particuliers d'un système logiciel. Par ailleurs, UML modélise le système suivant deux modes de représentation : l'un concerne la structure du système pris « au repos », l'autre concerne sa dynamique de fonctionnement. Les deux représentations sont nécessaires et complémentaires pour schématiser la façon dont est composé le système et comment ses composantes fonctionnent entre elles. Le mode de représentation statique s'appuie sur les six diagrammes ci-après.

• Le diagramme de classes : Sur la branche fonctionnelle, ce diagramme est prévu pour développer la structure des entités utilisées par les utilisateurs. En conception, le diagramme de classes représente les modules du langage de développement.

• Le diagramme d'objets sert à illustrer des structures de classes compliquées. Ce diagramme est utilisé en analyse pour vérifier l'adéquation d'un diagramme de classes à différents cas possibles.

• Le diagramme de composants représente en premier lieu les concepts connus de l'exploitant du système pour installer et dépanner le système. Il s'agit dans ce cas de déterminer la structure des composants d'exploitation que sont les librairies dynamiques, les instances de bases de données, les applications, etc. Le diagramme de composants représente en second lieu les concepts de configuration logicielle.

• Le diagramme de déploiement correspond à la fois à la structure du réseau informatique qui prend en charge le système logiciel, et la façon dont les composants d'exploitation y sont installés.

• Le diagramme de paquetage regroupe les hiérarchies de classes. Il permet d'exprimer la dépendance entre paquetage et donne une vision de la structure du logiciel.

• Le diagramme de structure composite qui présente la structure d'une classe complexe. Il permet de mettre en évidence les différentes parties d'une classe lors de l'exécution.

Le mode de représentation dynamique ou comportemental s'appuie sur les sept diagrammes ci-après.

• Le diagramme de cas d'utilisation représente la structure des fonctionnalités nécessaires aux utilisateurs du système. Il est utilisé dans les deux étapes de capture des besoins fonctionnels et techniques.

• Le diagramme d'états représente le cycle de vie commun aux objets d'une même classe. Ce diagramme complète la connaissance des classes en analyse et en conception.

• Le diagramme d'activités représente les règles d'enchaînement des activités dans le système. Il permet de consolider la spécification d'un cas d'utilisation.

- Le diagramme de séquence qui est une représentation séquentielle du déroulement des traitements et des interactions entre les éléments du système et de ses acteurs. Il représente les échanges de messages dans le cadre d'un fonctionnement particulier du système.

- Le diagramme de communication qui est une représentation simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets.

- Diagramme global d'interaction permet de décrire les enchaînements possibles entre les scénarios préalablement identifiés sous forme de diagrammes de séquences. C'est une variante du diagramme d'activité.
- Diagramme de temps permet de décrire les variations d'une donnée au cours du temps.

2.3. ENVIRONNEMENT DE TRAVAIL

L'administration centrale de la congrégation des Frères des Ecoles Chrétiennes, sise à Bobo-Dioulasso, est constituée de cinq services que sont la direction, le secrétariat, l'accueil, l'économat et la comptabilité. Tous ces services sont connectés à internet.

Au sein de cette administration travaillent un directeur, appelé Frère Visiteur, un secrétaire, un économe général, deux comptables et un technicien de surface. Cette administration est appelée « Maison Provinciale ».

L'environnement matériel mis à notre disposition se présente comme suit :

1. Un ordinateur portable TOSHIBA (Windows 7) AMD Athon™ X2 Dual-CoreQL-65 2.5Ghz de 4 Go de RAM
2. Un ordinateur portable DELL (Windows XP) Intel® Pentium ® M 1.70Ghz de 248 de RAM

DEUXIEME PARTIE : CONCEPTION DU LOGICIEL

CHAPITRE IV : LES CONTRAINTES FONCTIONNELLES

Pour plus de lisibilité du document, il sera traité d'abord, en première partie, la branche gauche du processus en Y qui définit les contraintes fonctionnelles. Et en deuxième partie, la branche droite du 2TUP qui définit les contraintes techniques

1. ETUDE PRELIMINAIRE

Cette étape consiste à cadrer le projet, en particulier à identifier toutes les entités externes qui vont interagir avec le système, et à définir la nature de cette interaction. Elle prépare les activités plus formelles de capture des besoins fonctionnels.

1.1. PRESENTATION DU PROJET

Le logiciel à réaliser est une application web de gestion intégrée des communautés religieuses et des établissements de la congrégation des Frères des Ecoles Chrétiennes au Burkina/Niger dénommée District d'Afrique de l'Ouest.

1.2. RECUEIL DES BESOINS FONCTIONNELS

Nous avons effectué des recherches documentaires au sein de la maison provinciale qui est l'organe administratif du District d'Afrique de l'Ouest. Nous avons aussi appuyé cette recherche par des interviews du Frère Visiteur, du Frère Econome, du Frère Secrétaire, des directeurs d'œuvres. Cela a permis de dégager le cahier des charges descriptif préliminaire suivant :

- Administration du district

L'administration est essentiellement subdivisée en deux éléments : d'une part, la direction composée du Frère Visiteur et de son secrétaire, d'autre part l'économat composé du Frère Econome et des comptables.

Le Frère Visiteur est le premier responsable des Frères. Il a en charge de valider les projets soumis à lui par les Frères, par les établissements scolaires ou centres de

formation, et par les communautés. La communauté est un groupe de Frères vivant ensemble et souvent chargée de la gestion d'un établissement scolaire ou d'un centre de formation.

Le procès verbal de réunions est rédigé par le secrétaire. Il est ensuite transmis sur support papier aux autres religieux. Aussi, les rapports d'activités des établissements scolaires et autres centres de formation sont envoyés au secrétaire. Le secrétaire a également en sa charge la conservation des archives de la congrégation.

Sous l'autorité du Frère Visiteur, qui est l'ordonnateur et le contrôleur, l'économe est chargé de gérer, contrôler, conseiller et s'informer sur les ressources matérielles et financières du district. L'économe est soutenu par un service de comptabilité dont il est le responsable.

- Les œuvres

Il existe des établissements d'enseignement général et des centres d'initiation à divers métiers. Ces établissements et centres sont appelés « œuvres ». Les œuvres sont dirigées par un directeur et un économe tous nommés par le Frère Visiteur.

A chacune des œuvres est associée une communauté de Frères qui apporte une animation « Lasallienne » à l'œuvre. La communauté et l'œuvre ont des employés distincts. La communauté religieuse a un projet communautaire qui court sur un an. Le Directeur de l'œuvre est souvent le directeur de la communauté religieuse. L'économe de l'œuvre peut ne pas être l'économe de la communauté religieuse. Les Frères sont envoyés dans les communautés par le Frère Visiteur suivant les années scolaires. Les Frères sont souvent enseignants dans les établissements qu'ils animent ou parfois libérés pour d'autres fonctions ailleurs que dans l'établissement. Dans les œuvres, il existe des enseignants qui ne sont pas religieux.

- Les religieux

Le postulat est la période où le jeune désirant être Frère vit en communauté avec les Frères durant un an. Tout postulant a une communauté et un accompagnateur.

Le Noviciat est la période durant laquelle un jeune désirant être Frère entre dans la maison de formation pour y apprendre les fondements de la vie religieuse. On appelle « promotion » les jeunes qui entrent au noviciat à la même date. La formation dure 2 ans.

Le scolasticat est la période de deux à trois ans durant laquelle le jeune religieux se forme à la pédagogie et à la théologie.

1.3. IDENTIFICATION DES ACTEURS

UML n'emploie pas le terme *d'utilisateur* mais d'*acteur*. Les acteurs d'un système sont les entités externes à ce système qui interagissent (saisie de données, réception d'informations, ...) avec lui. Les acteurs sont donc à l'extérieur du système et dialoguent avec lui. Ces acteurs permettent de cerner l'interface que le système va devoir offrir à son environnement. Les acteurs du système identifiés dans un premier temps sont :

- *Le religieux* : un religieux peut consulter un profil, un cursus, consulter les différentes années communautaires.

- *Le directeur* : le directeur établit les programmes d'activités scolaires, il crée les rapports de début et de fin d'année. Il introduit les projets communautaires. Il introduit les informations concernant les employés et les enseignants.
- *Secrétaire* : le secrétaire crée les communautés, introduit les profils et cursus des religieux, introduit les procès verbaux des conseils de District et de commissions, les historiques.
- *Administrateur* : crée les profils d'utilisateurs et attribue les droits d'accès.

1.4. IDENTIFICATION DES MESSAGES

Un *message* représente la spécification d'une communication unidirectionnelle entre les objets avec intention de déclencher une activité chez le récepteur.

Le système émet les messages suivants :

- Création de procès verbal
- Création de contrats
- Création de bilans annuels
- Création de projets communautaires
- Création de cursus d'un religieux.
- Création d'un enseignant
- Création d'un employé
- Création d'un religieux.

Le système reçoit les messages suivants :

- Les créations, modifications, suppressions de profils enseignants/religieux/employés.
- Les créations, modifications, suppression de comunnautés/oeuvres.
- L'affectation des enseignants/religieux/employés à une communauté/oeuvre.
- Les créations, modifications, suppressions des Procès verbaux/contrats/bilans/....
- Les créations, modifications, suppressions de profils d'utilisateurs.

1.5. MODELISATION DU CONTEXTE

A partir des informations obtenues lors des précédentes étapes, une modélisation du contexte de l'application est effectuée. Cela permet de définir le rôle de chaque acteur à travers le tableau 2. Ce tableau 2 définit les utilisateurs et leur associe leurs besoins fonctionnels.

Tableau 2: tableau de modélisation de contexte

Utilisateurs	Description des besoins fonctionnels
Le religieux	L'application doit permettre au religieux de : • S'authentifier • Consulter un profil • Consulter un cursus (parcours) • Consulter une communauté • Consulter une œuvre
Le directeur	L'application doit permettre au directeur de : • S'authentifier • Enregistrer les rapports • Enregistrer les projets communautaires • Enregistrer les contrats de travail • Créer/modifier les fiches de profil des enseignants
Le secrétaire	L'application doit permettre au secrétaire de : • S'authentifier • Créer les communautés • Créer les œuvres • Affecter les religieux dans les communautés • Affecter les religieux dans les œuvres • Créer/modifier les profils des religieux • Créer/modifier les données des religieux • Consulter le profil des enseignants • Consulter les rapports d'œuvres/communautés • Enregistrer les documents administratifs
Administrateur	L'application doit permettre à l'administrateur de : • S'authentifier • Créer les profils utilisateurs • Donner des droits d'accès • Configurer l'application

2. CAPTURE DES BESOINS FONCTIONNELS

Cette phase représente un point de vue « fonctionnel » de l'architecture système. Par le biais des cas d'utilisation, le contact sera permanent avec les acteurs du système en vue de définir les frontières de celui-ci, et ainsi éviter de s'éloigner des besoins réels de l'utilisateur final.

2.1. DÉTERMINATION DES CAS D'UTILISATION

Un *cas d'utilisation* représente un ensemble de séquences d'actions réalisées par le système et produisant un résultat observable intéressant pour un acteur particulier.

Un cas d'utilisation modélise un service rendu par le système.

L'identification des cas d'utilisation donne un aperçu des fonctionnalités futures que doit implémenter le système.

L'identification des cas d'utilisation est une activité itérative. Certains cas d'utilisation sont remodelés au fur et à mesure de l'approfondissement de l'analyse des besoins fonctionnels.

Pour constituer les cas d'utilisation, il faut considérer l'intention fonctionnelle de l'acteur par rapport au système dans le cadre de l'émission ou de la réception de chaque message. En regroupant les intentions fonctionnelles en unités cohérentes, on obtient les cas d'utilisations.

Tableau 3: tableau des cas d'utilisation

Cas d'utilisation	Acteur principal, acteurs secondaires	Messages émis/reçus par les acteurs
Gérer les fichiers	**Le directeur** **Le secrétaire**	Emet : Créer/modifier les rapports, Créer/modifier les projets communautaires, l'historique C2, C3, faire un bilan de rentrée, de fin d'année, rapport de conseil, programme du Visiteur, définit les droits de lecture. Reçoit : liste des membres/profil de la communauté, identification de l'œuvre, identification de la communauté, consulter les rapports des œuvres, consulter les rapports des communautés (C2, C3)
Gérer les employés	**Le directeur**	Emet : Créer/modifier les profils des enseignants, les employés. Enregistrer les contrats de travail
Gérer les Frères	**Le secrétaire**	Emet : Identifier les frères, donner des code d'accès aux frères, affecter les frères dans les communautés, nommer les frères dans les commissions.
Gérer les institutions	**Secrétaire**	Emet : créer l'œuvre, créer les communautés, sélectionner les communautés/frères (affectation), Créer les commissions. Reçoit : Consulter la fiche d'un enseignant/employé, la liste des

		enseignants/employés, les contrats, les rapports.
Consulter le profil	**Religieux/Directeur/ secrétaire**	Reçoit : liste des enseignants, des frères.
Consulter les institutions	**Religieux/directeur /secrétaire**	reçoit : liste des institutions, fichiers d'institutions
Gérer les profils utilisateurs	**Administrateur**	Emet : Créer/modifier les contrôles d'accès, Créer/modifier la configuration générale, vérifier/sauvegarder la base de données, faire les statistiques, historier les commandes, envoyer message aux utilisateurs.

Maintenant que les cas d'utilisation et leurs acteurs ont été identifiés, le diagramme de cas d'utilisations permet de visualiser graphiquement les acteurs et leurs activités.

Pour représenter les cas d'utilisation nous utiliserons le formalisme de la figure 3 :

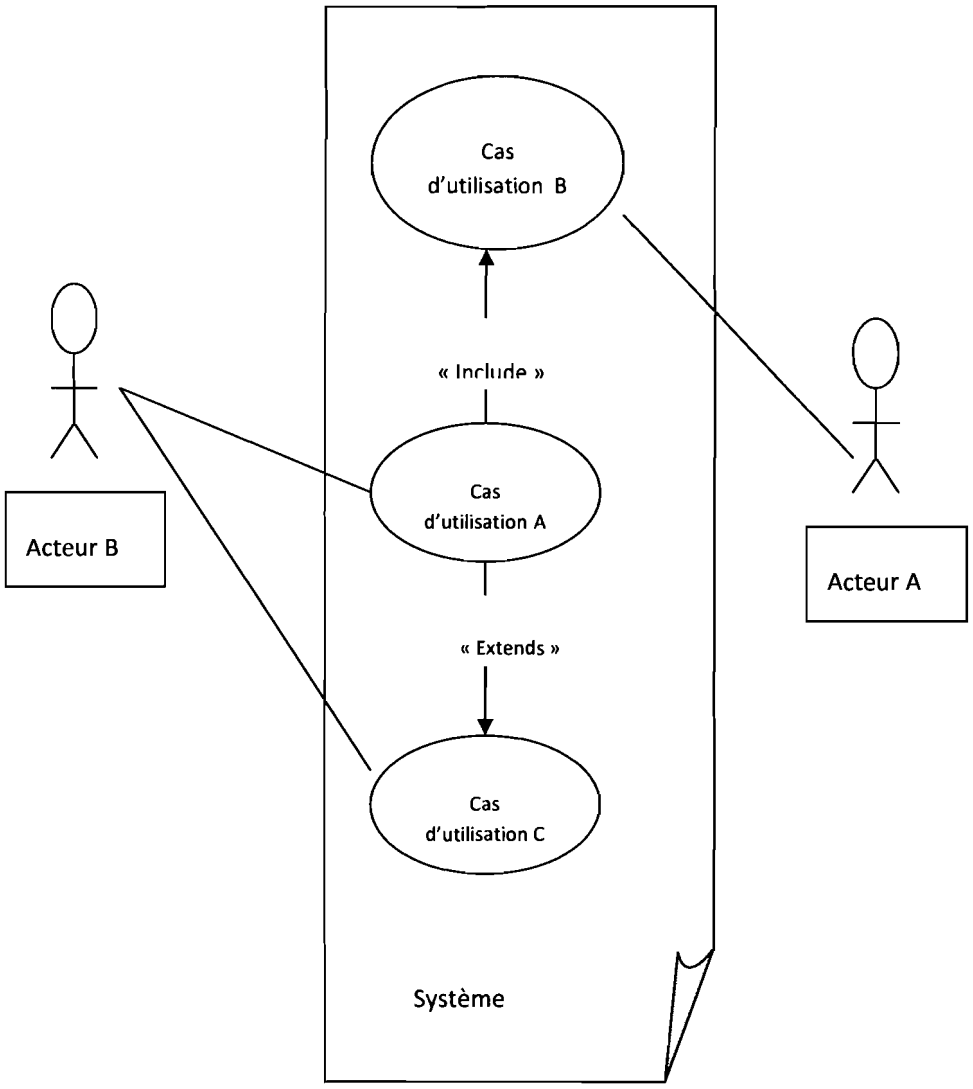


Figure 3: formalisme des cas d'utilisation

1. La relation "include" est une relation entre 2 instances de cas d'utilisation telle que la réalisation du cas d'utilisation à l'origine de la flèche nécessite la réalisation de la cible.

2. La relation extend est une relation entre 2 instances de cas d'utilisation telle que A extend B signifie que le comportement de B peut être complété par le comportement de A. La relation extend indique une **possibilité**, un complément possible.

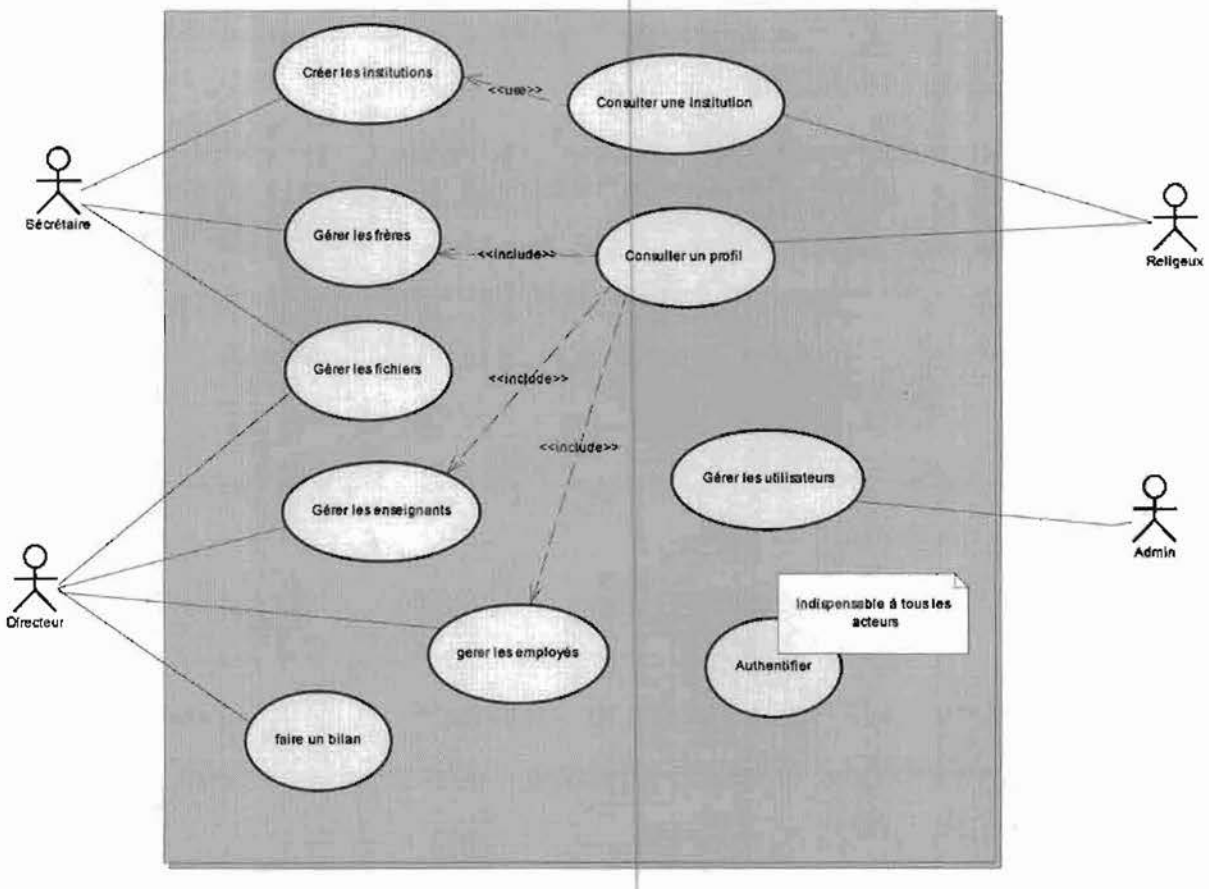


Figure 4: Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation de la figure 4 présente les cas d'utilisation représentés dans le tableau 3. Le cas d'utilisation « authentifier », indispensable à tous les acteurs, n'interagit avec aucun d'eux. Cela est voulu pour améliorer la clarté du diagramme de cas d'utilisation. La description détaillée de quelques cas d'utilisations figure en annexe.

Après avoir illustré le diagramme de cas d'utilisation, le regroupement en packages permet de trouver de grands ensembles de cas d'utilisation selon des critères définis ci-après.

2.2. STRUCTURATION DES CAS D'UTILISATION EN PACKAGES

Pour définir la stratégie de regroupement des cas d'utilisation pour le projet, trois critères de regroupement sont souvent utilisés :

• par domaine d'expertise métier : le plus intuitif et souvent le plus efficace.

Il facilite la spécialisation des analystes et permet d'organiser la disponibilité des différents experts ;

• par acteur : simple à mettre en œuvre uniquement si chaque cas d'utilisation est relié à un et un seul acteur, sinon il s'apparente souvent au critère précédent ;

• par lot de livraison : dans le cadre d'un développement itératif et incrémental, il est intéressant de regrouper dans un même package les cas d'utilisation qui seront livrés ensemble au client.

Le mécanisme générique de regroupement d'éléments en UML s'appelle *package*. Le regroupement par domaine d'expertise métier sera utilisé. Il permet aux groupes de réalisation de l'application de mieux s'organiser.

Tableau 4: package des cas d'utilisation

Cas d'utilisation	Acteur principal, acteurs secondaires	Package
Gérer les fichiers	**Le directeur** **Le secrétaire**	GESTION DE FICHIERS
Gérer les employés **Gérer les frères** **Gérer les institutions**	**Le directeur** **Le secrétaire**	GESTION DES DOMAINES
Consulter les profils **Consulter les institutions**	**Le secrétaire** **Les Frères** **Le directeur**	GESTION DES LECTURES
Gérer les profils utilisateurs	**Administrateur**	GESTION DES CONFIGURATIONS

Le module de *gestion des fichiers* (tableau 4) demande une expertise en manipulation de fichiers de tout genre (lecture de fichiers d'extension txt, chargement de fichier word, excel..). *La gestion des domaines* demande une expertise en manipulation de bases de donées (lecture, écriture, ...). Le module de la *gestion des lectures*, qui est une partie de la manipulation de base de données, se spécialise uniquement dans les droits de lecture des utilisateurs. Enfin, le module de *gestion des configurations* demande une bonne analyse de la configuration générale de l'application.

2.3. IDENTIFICATION DES CLASSES CANDIDATES

A cette étape, l'utilisation des différents diagrammes de cas d'utilisation de la phase de capture des besoins donnera lieu à un ensemble de classes qui permettront de matérialiser de manière informatique chacune des entités identifiées durant la capture des besoins. Chaque cas d'utilisation contient un vocabulaire et des mécanismes, lesquels doivent maintenant se projeter à travers des entités informatiques. Ces diagrammes préliminaires obtenus à partir des cas d'utilisation sont appelés « *diagramme des classes participantes* ». Une classe est un moyen pour regrouper des éléments qui présentent une structure commune et des comportements communs.

1. Cas d'utilisation 'GERER LES FICHIERS'

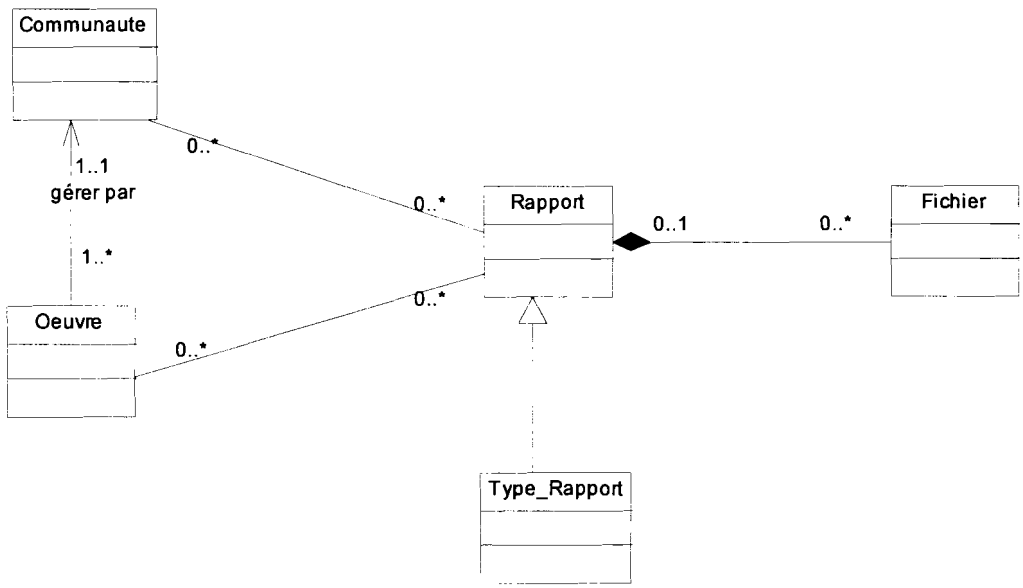


Figure 5: Diagramme des classes participantes de 'GERER LES FICHIERS'

2. Cas d'utilisation 'GERER LES EMPLOYES'

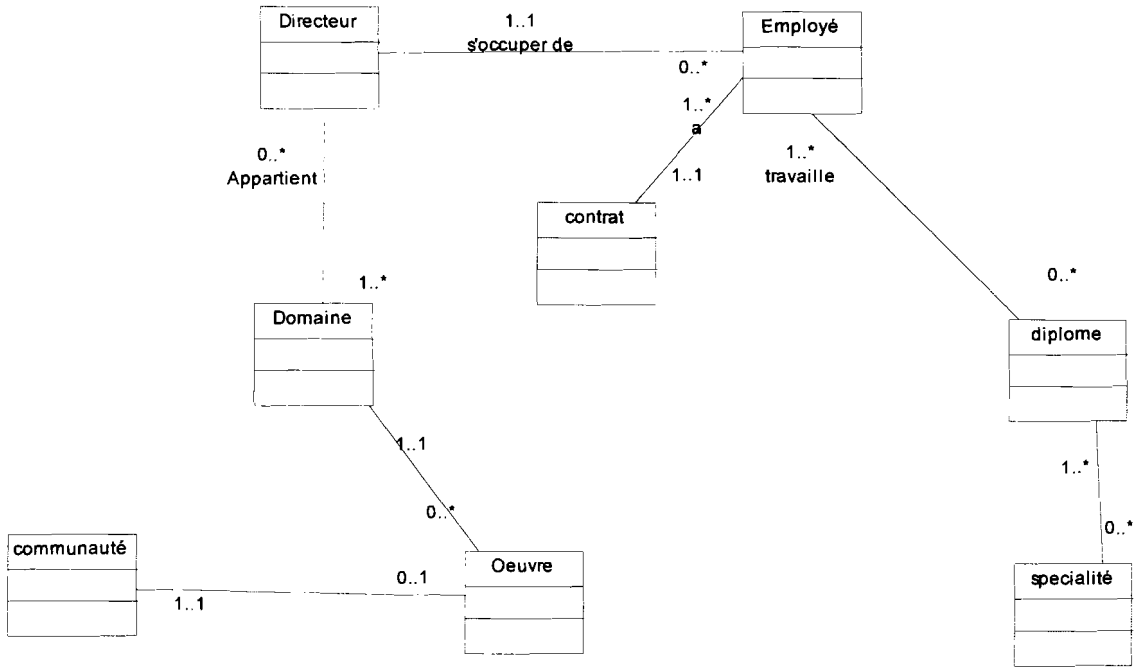


Figure 6: Diagramme des classes participantes de 'GERER LES EMPLOYES'

3. Cas d'utilisation : 'GERER LES FRERES'

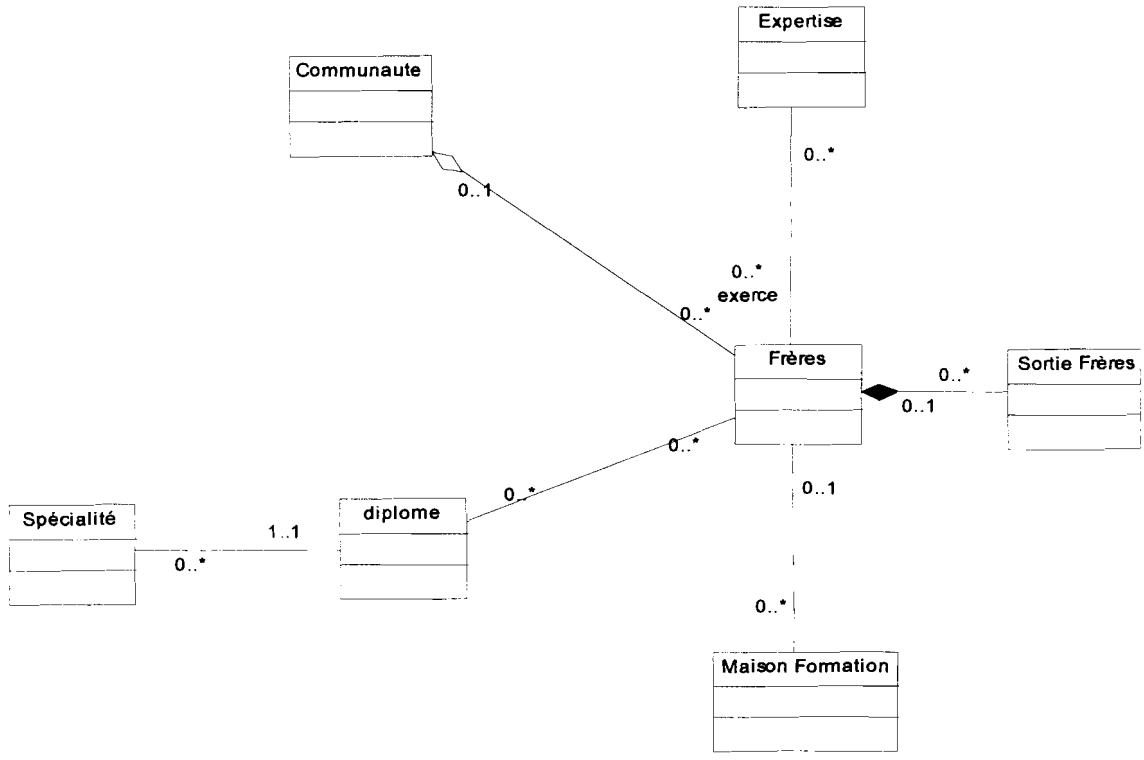


Figure 7: Diagramme des classes participantes de 'GERER LES FRERES'

4. Cas d'utilisation 'CONSULTER UNE INSTITUTION'

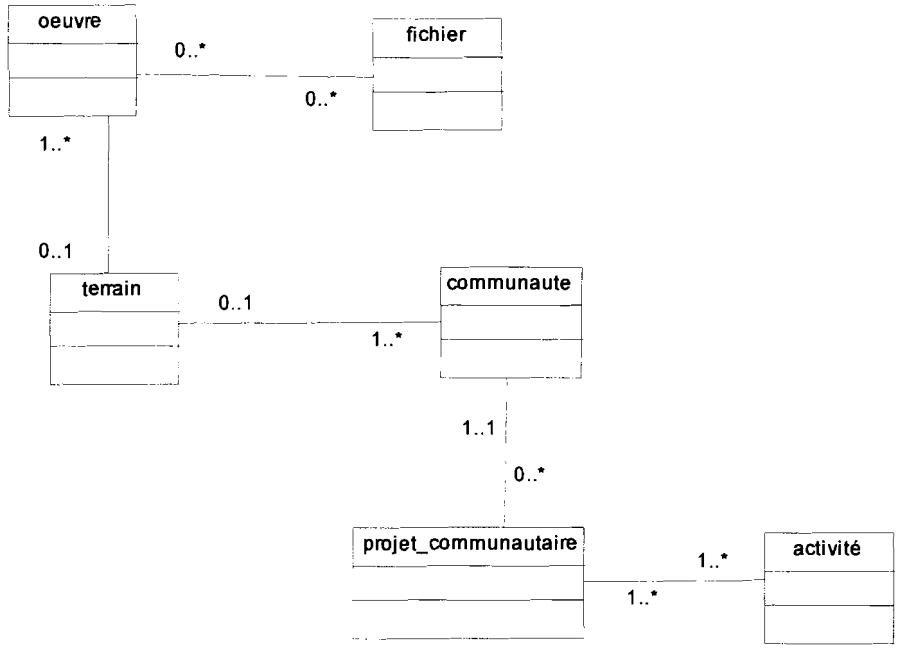


Figure 8: Diagramme des classes participantes de 'CONSULTER UNE INSTITUTION'

5. Cas d'utilisation 'CREER UNE INSTITUTION'

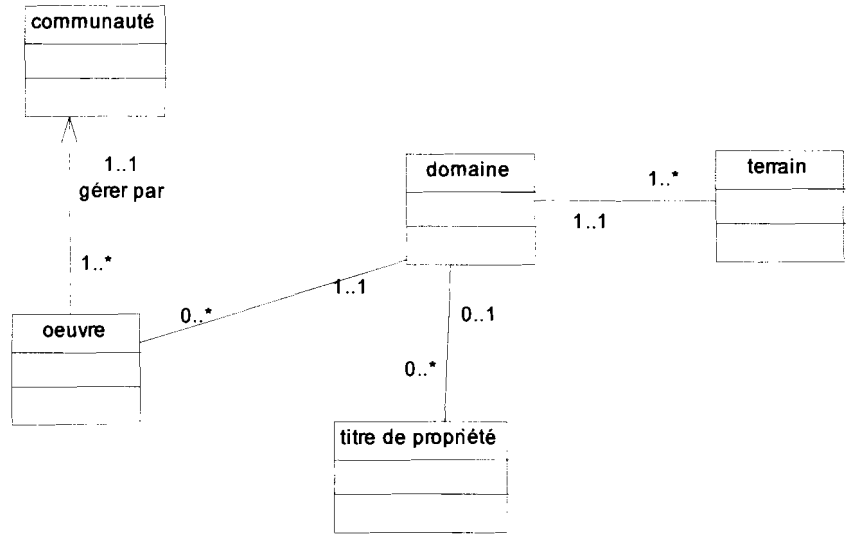


Figure 9: Diagramme des classes participantes de 'CREER UNE INSTITUTION'

Les classes participantes découvertes dans la capture des besoins fonctionnels permettent de faire apparaître les classes potentielles pour l'analyse.

3. ANALYSE

3.1. DECOUPAGE EN CATEGORIE DE CLASSES

Quelques règles guident l'attribution des objets aux paquetages ou « catégories ». Ces règles consistent à regrouper au sein d'un même paquetage les classes reliées par une association d'agrégation, celles qui répondent à la même attente vis-à-vis des utilisateurs et enfin, les classes simples dont l'une peut être interprétée comme une caractéristique de l'autre et qui sont fortement connexes.

Le découpage en catégories constitue la première activité de l'étape d'analyse et elle va s'affiner de manière itérative au cours du développement du projet.

Parmi les classes candidates nous éliminons: les classes redondantes ou vagues.

Le découpage en catégories du présent projet a donné le résultat de la figure 10 :

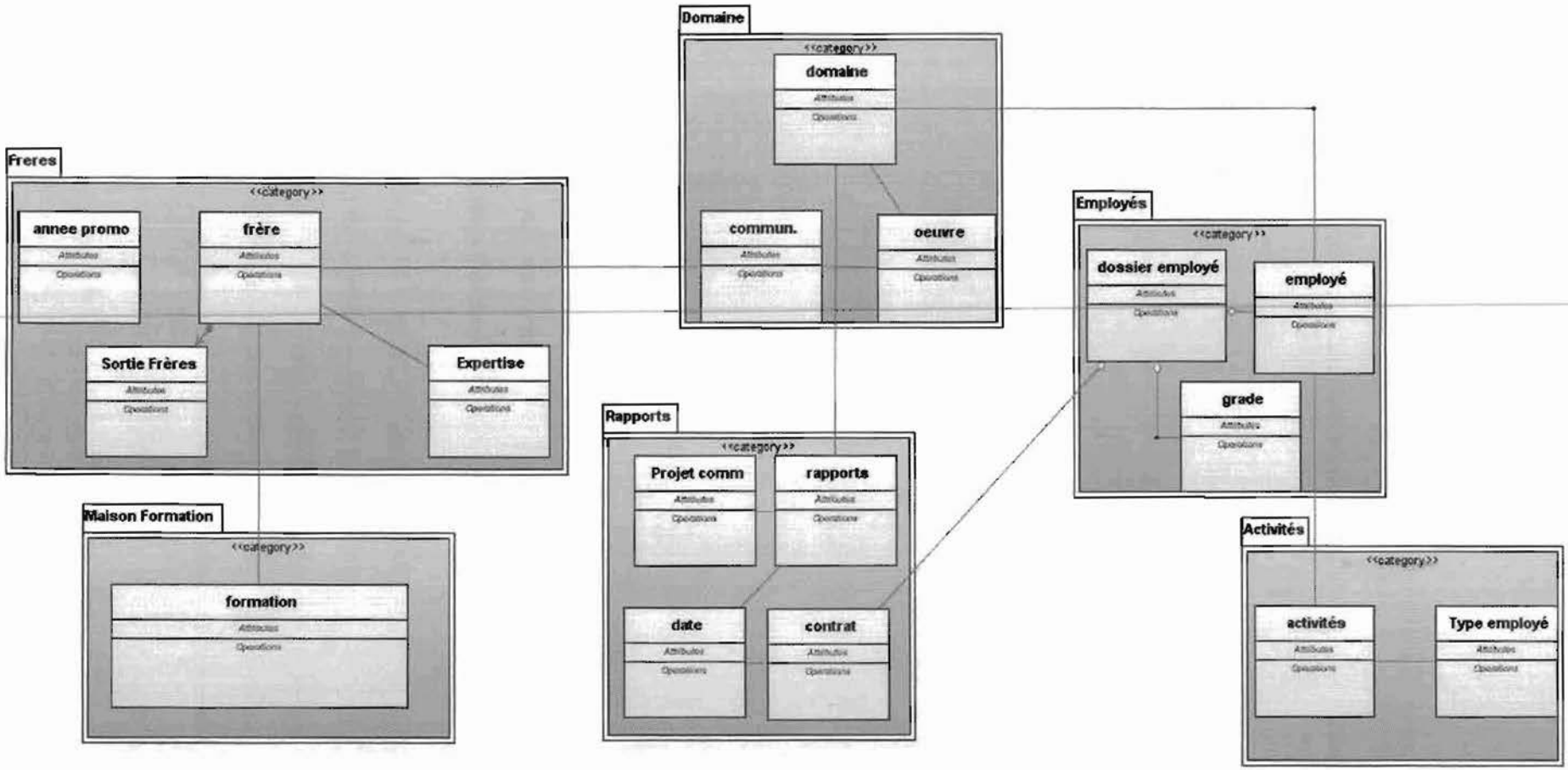


Figure 10: diagramme de paquetages

3.2. DEVELOPPEMENT DU MODELE STATIQUE

Après la définition des catégories et des dépendances qui règnent entre elles, le développement du modèle statique permettra de fixer les classes principales du diagramme de classes.

Les diagrammes de classes établis sommairement dans les *diagrammes de classes participantes*, vont être rassemblés en un seul diagramme de classes (figure 11).

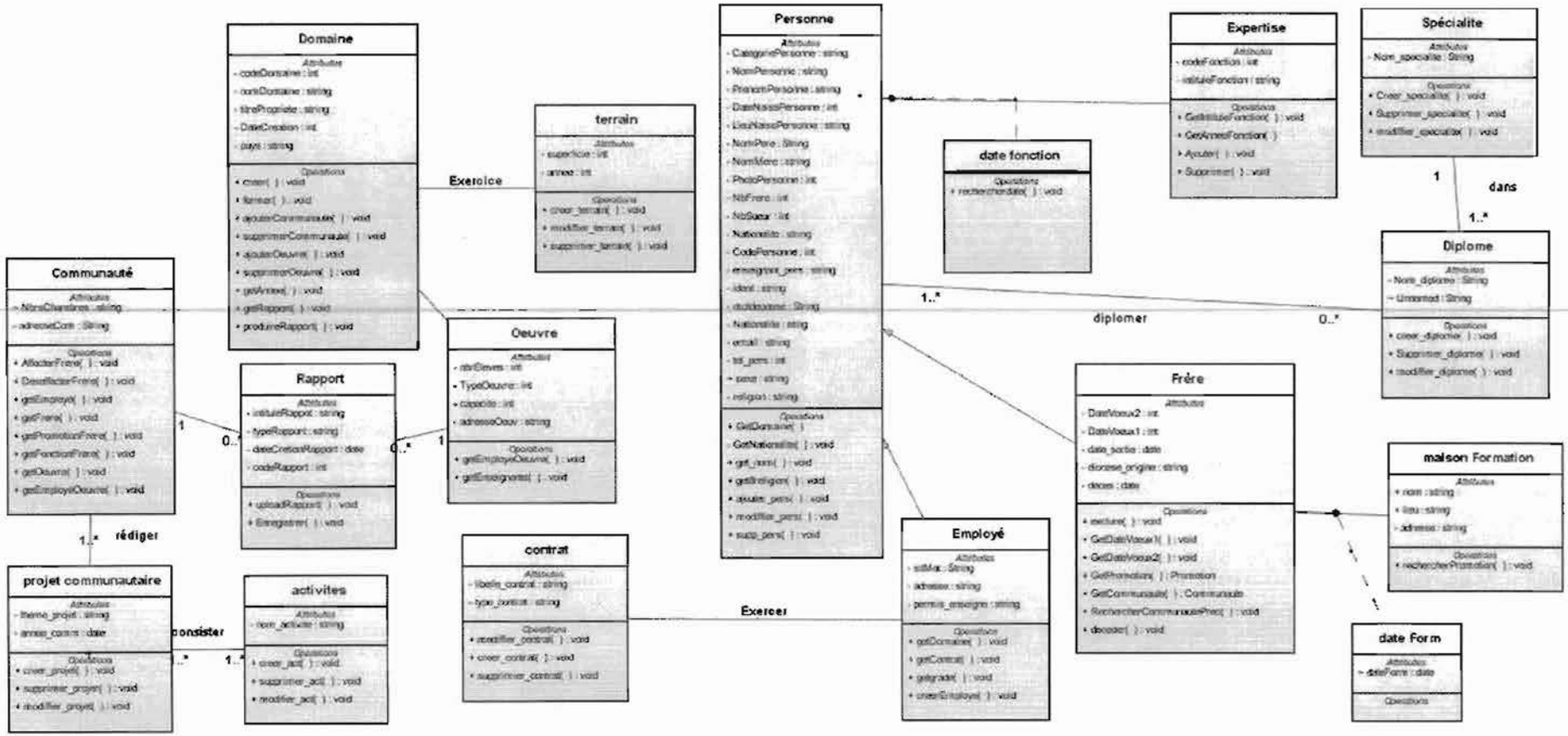


Figure 11: Le diagramme de classes

3.3. DEVELOPPEMENT DU MODELE DYNAMIQUE

Le développement du modèle dynamique constitue la dernière activité de notre analyse. Il s'agit d'une activité itérative, fortement couplée avec le diagramme de classes.

Pour développer le modèle dynamique, nous utiliserons les diagrammes de séquences.

3.3.1. FORMALISME DU DIAGRAMME DE SEQUENCE

Le diagramme de séquence montre une interaction présentée en séquence dans le temps. En particulier, il montre les instances qui participent à l'interaction par leur « ligne de vie » et les messages qu'ils échangent, présentés en séquence dans le temps. Une interaction est un ensemble de messages échangés entre les objets participants dans l'interaction. Son formalisme est représenté à la figure 12:

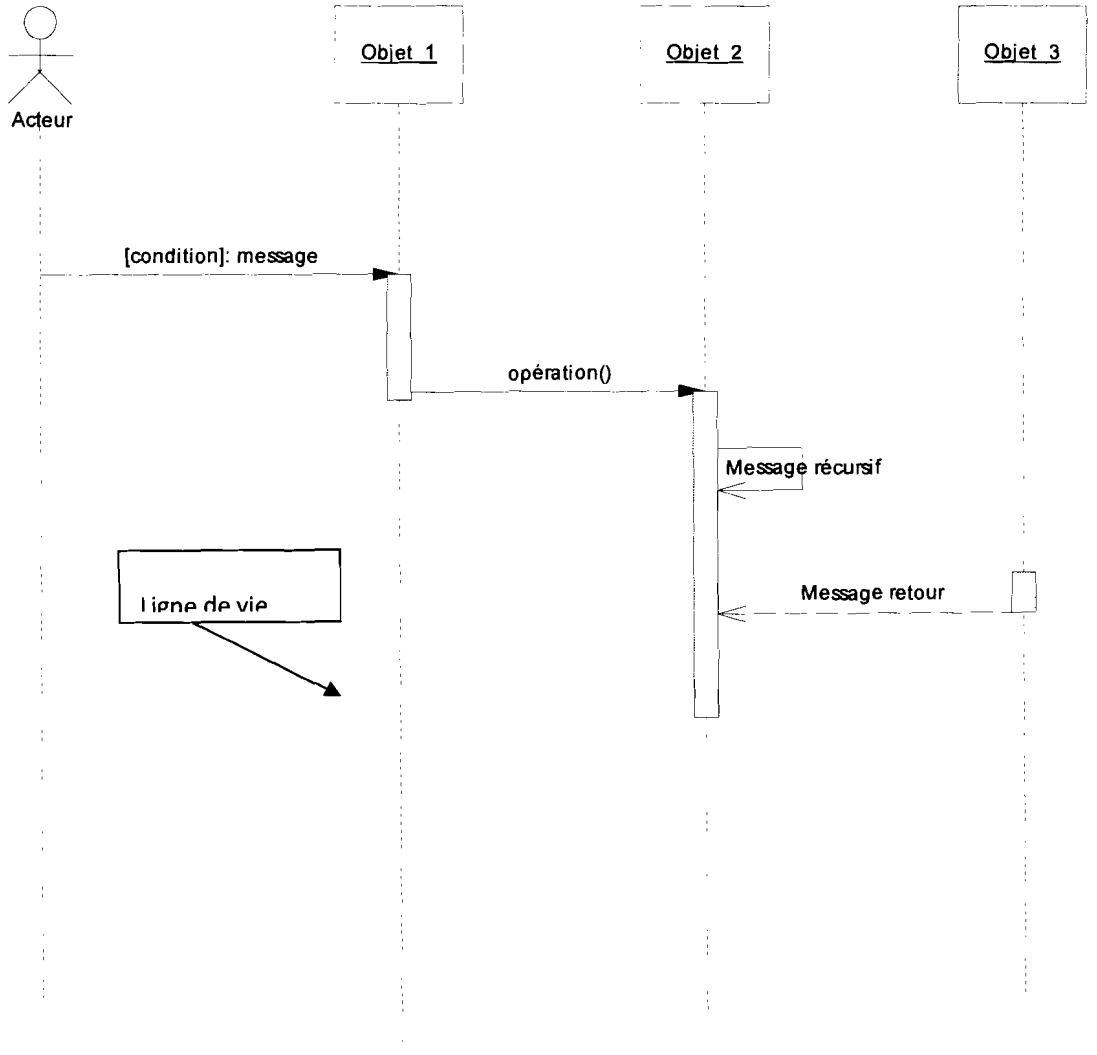


Figure 12: Formalisme du diagramme de séquence

Message asynchrone

Demande dans laquelle l'objet émetteur n'attend pas un résultat et peut donc faire autre chose en parallèle.

Appel de procédure

La séquence est terminée avant que la séquence suivante ne commence. L'émetteur doit attendre une réponse ou la fin de l'activation.

Message retour Message retour

Généralement associé à un Appel de procédure. La flèche Retour peut être omise puisqu'elle est implicite à la fin d'une activation

Les objets : Ils peuvent être de diverses sortes :

- dialogue : c'est une classe modélisant la communication entre le système et son environnement. C'est une classe qui se trouve à la périphérie du système tout en étant à l'intérieur du système. Elle interagit avec les acteurs et les objets.

- Contrôle : Il contrôle les interactions d'une collection d'objets. C'est généralement l'expression d'un objet intangible.

- Entité : C'est une classe généralement persistante modélisant une information. C'est une classe passive qui peut participer à la réalisation de plusieurs cas d'utilisation.

3.3.2. LES SCÉNARIOS

Un scénario décrit une exécution particulière d'un cas d'utilisation du début à la fin. Il correspond à une sélection d'enchainements du cas d'utilisation. Un cas d'utilisation comporte une multitude de scénarios. Il sera développé quelques scénarios de cas d'utilisations importants.

On peut distinguer plusieurs types de scénarios permettant d'illustrer un diagramme de séquences :

- **Scénarios nominaux** : ils réalisent les *post conditions* du cas d'utilisation, d'une façon naturelle et fréquente ;

- **Scénarios alternatifs** : ils remplissent les *post conditions* du cas d'utilisation, mais en empruntant des voies détournées ou rares ;

- **Scénarios limites** : ils réalisent les *post conditions* du cas d'utilisation, mais modifient le système de telle sorte que la prochaine exécution du cas d'utilisation provoquera une erreur ;

- **Scénarios d'erreur** : ne réalisent pas les *post conditions* du cas d'utilisation.

1. Scénario de 'GERER LES FICHIERS'

Parmi tous les scénarios possibles pour le cas d'utilisation « Gérer les fichiers » (GF) nous avons choisi les suivants :

Scénarios nominaux

GF_N1 : Créer un dossier

GF_N2 : Importer un nouveau fichier

Scénarios alternatifs :

GF_A1 : transférer un fichier dans un répertoire

GF_A2 : Supprimer un rapport

Scénarios d'exception :

GF_E1 : non validation d'importation de rapport pour cause de nom inexistant.

Description détaillée

Scénario nominal : GF_N2 'Importer un nouveau fichier'

L'utilisateur donne un nom/mot de passe d'identification.

L'utilisateur choisit d'importer un document

Le système demande le type de fichier à importer

L'utilisateur sélectionne le type de fichier

Le système demande le module concerné (Directeur, secrétariat, employé, enseignants...)

L'utilisateur sélectionne le module concerné

Le système demande le chemin du fichier à charger

L'utilisateur choisit le document

L'utilisateur valide son choix

Le système vérifie le choix (exception GF_E1)

Le système enregistre la date d'importation

Le système enregistre le document

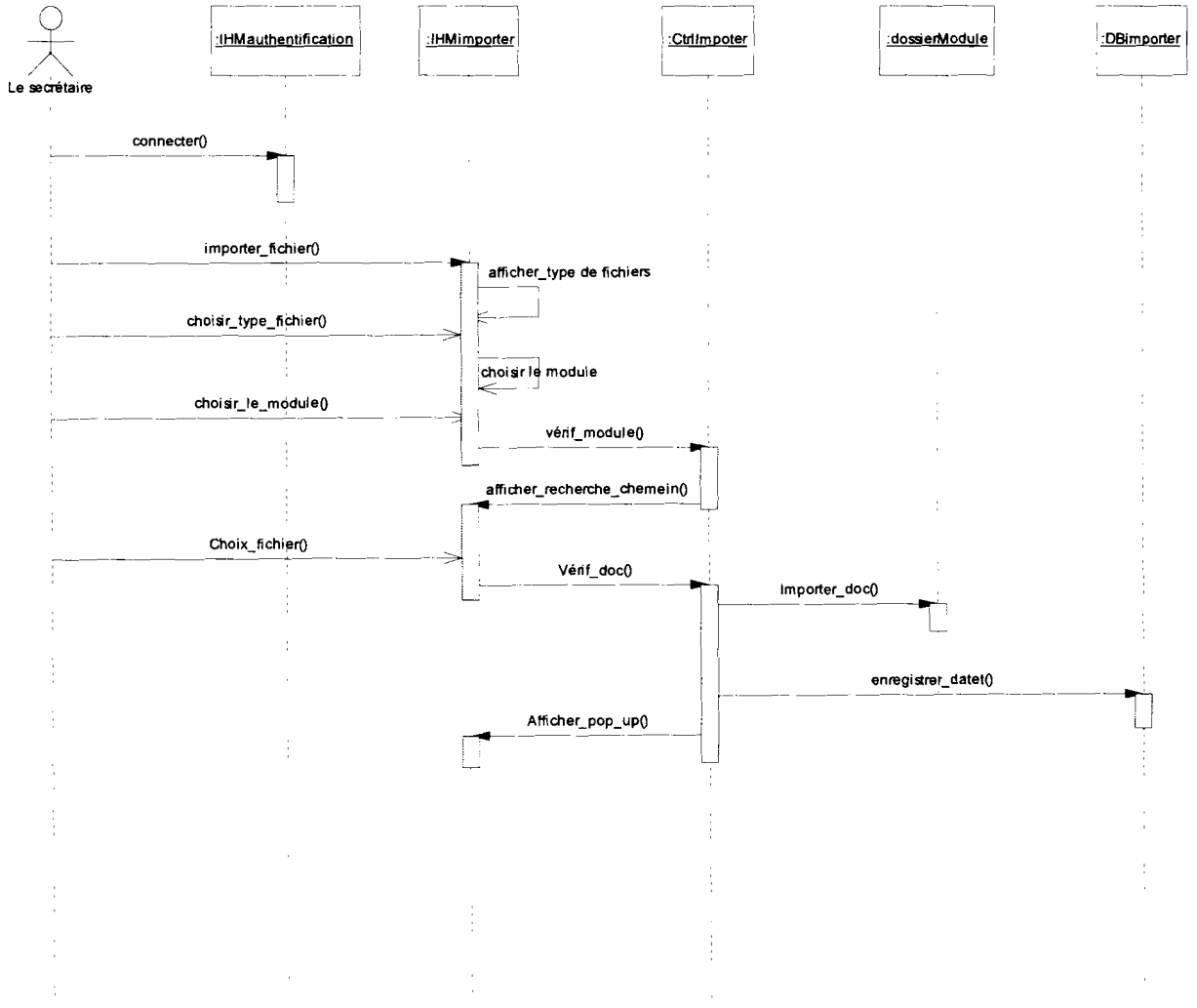


Figure 13: diagramme de séquence de : IMPORTER UN FICHIER

Scénario nominal : GF_N1 'Créer un dossier'

L'utilisateur donne un nom/mot de passe d'identification.

L'utilisateur choisit de créer un nouveau répertoire

Le système affiche le formulaire de création du répertoire

L'utilisateur saisit le nom du nouveau répertoire

Le système vérifie la validité du nom

Le système crée le répertoire dans la racine du site

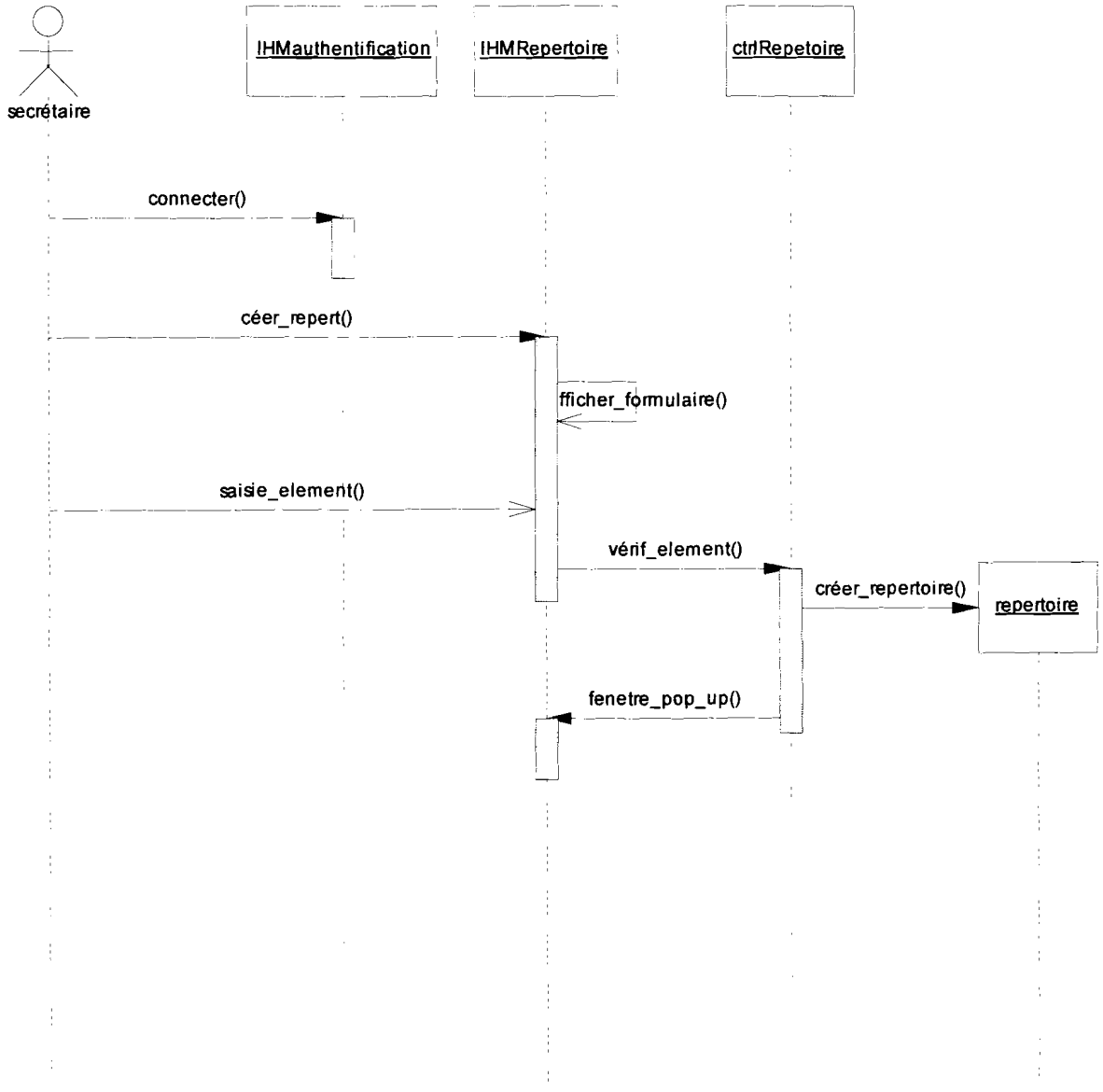


Figure 14: Diagramme de séquence : CREER DOSSIER

L'utilisateur est capable de créer ses propres dossiers au sein de l'application pour pouvoir ranger ces fichiers d'importations. La création de dossier par l'utilisateur participe de l'organisation voulue et de la consultation possible des documents.

Scénario alternatif : GF_A1 'transférer un fichier dans un répertoire'

Le directeur donne un nom/mot de passe d'identification.

Le directeur choisit de transférer un fichier dans un répertoire déjà créé

Le système propose la recherche du fichier

Le système propose la liste de nom de répertoire

Le directeur sélectionne le nom du répertoire

Le système transfère le fichier.

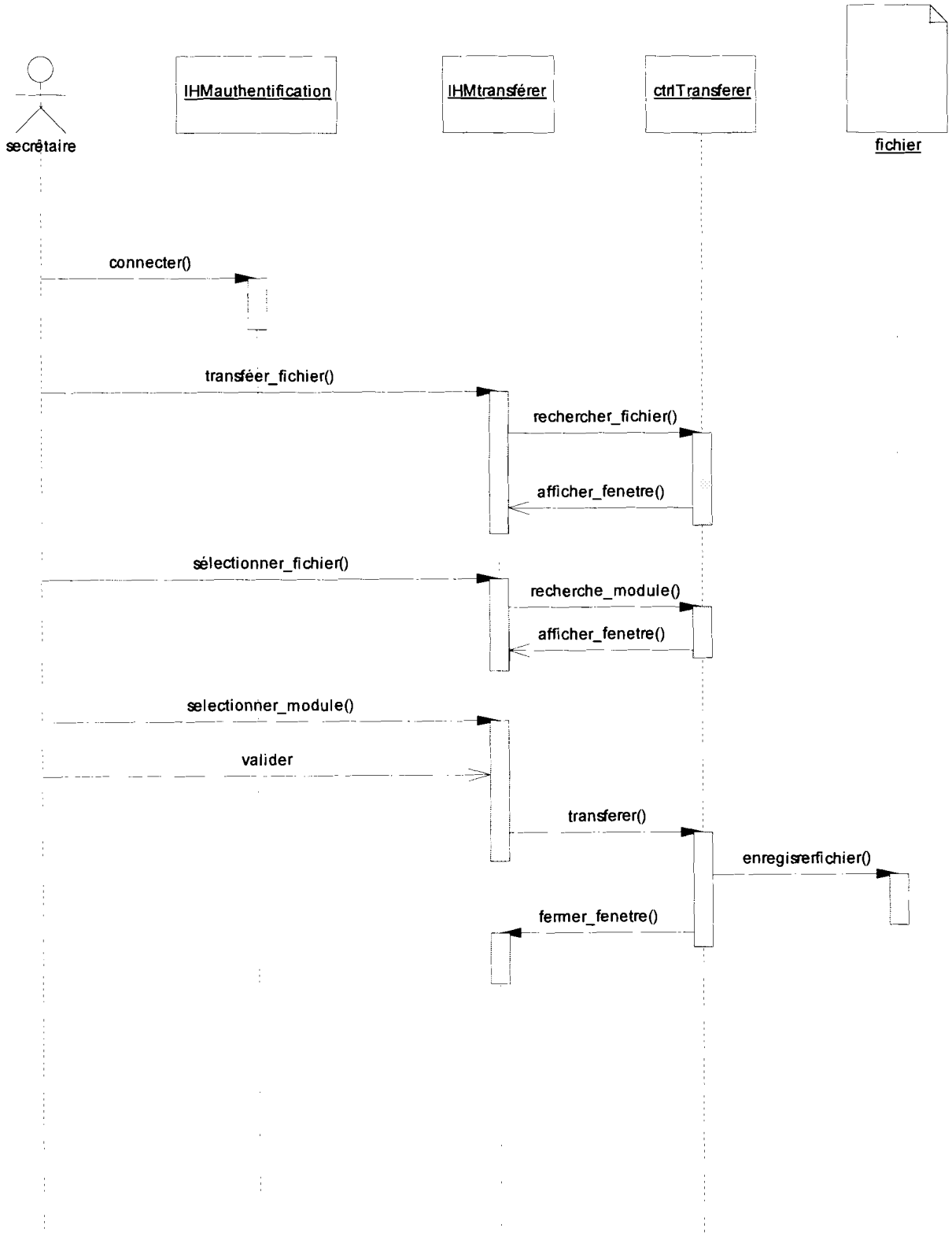


Figure 15: Diagramme de séquence de transfert de fichier

Scénario alternatif : GF_A2_'Supprimer un fichier'

Le directeur donne un nom/mot de passe d'identification.

Le directeur choisit de rechercher un document

Le directeur choisit de supprimer le fichier

Le directeur valide la suppression

Le système détruit les références du fichier concerné

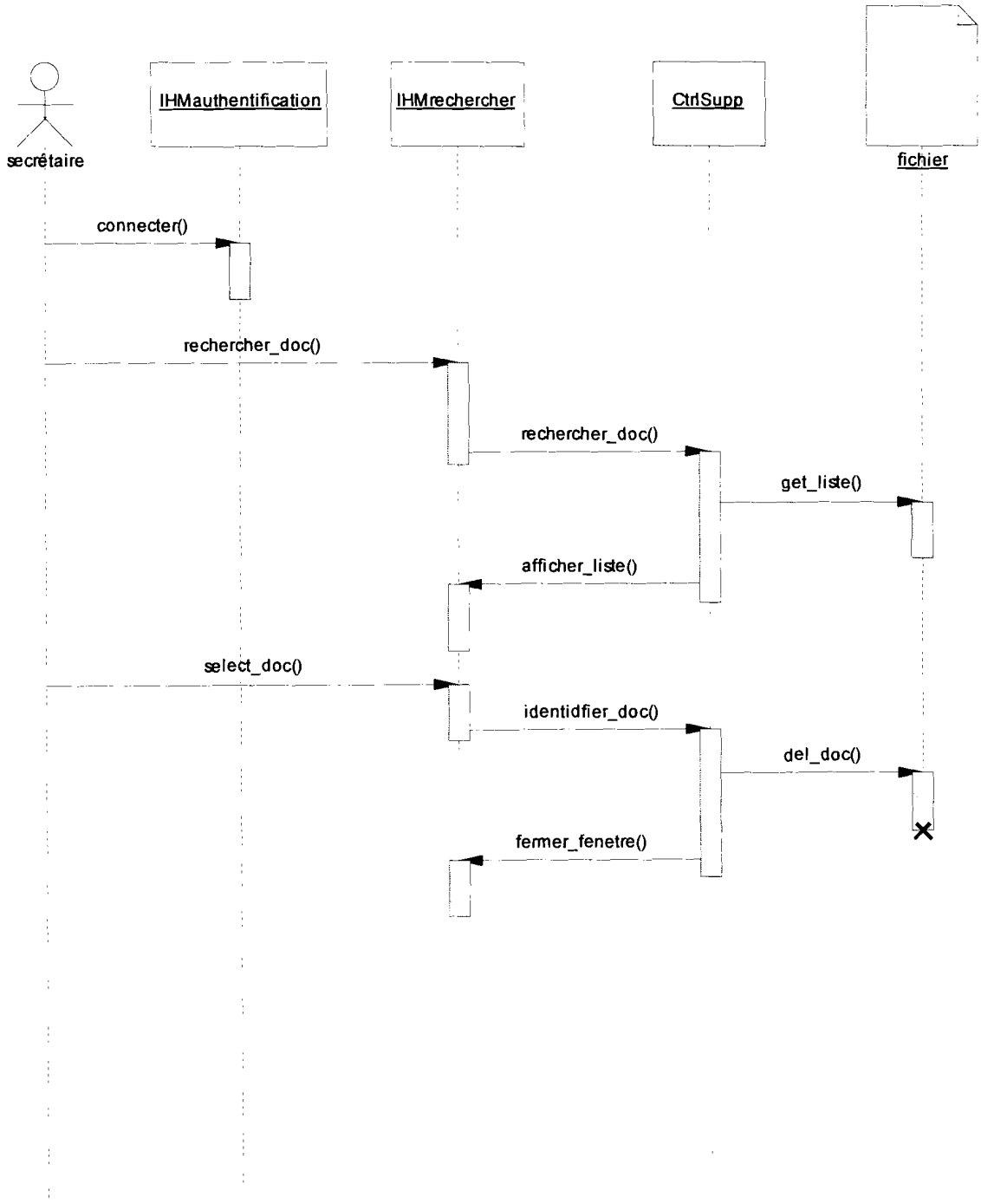


Figure 16: Diagramme de séquence de suppression de fichier

Scénario d'exception GR_E1 'non validation d'importation de fichier pour cause de nom inexistant.'

Le directeur enregistre le fichier

Le système vérifie la validité du nom fichier

Si le nom est inexistant, le système envoie un message d'erreur

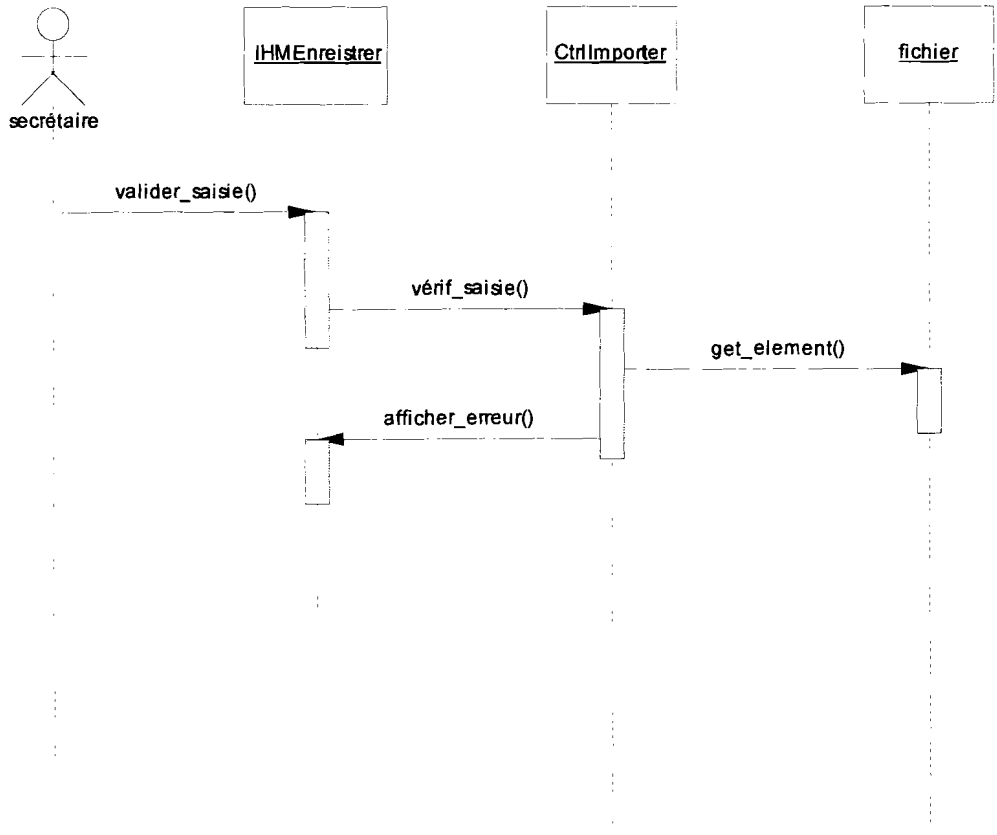


Figure 17: diagramme de séquence de non validation d'importation

Quand l'utilisateur choisit le nom du fichier, il pourrait choisir un fichier au format incorrect (fichier word, excel ou autres). L'application doit être capable de réfuter cette demande.

En conclusion, le développement du modèle dynamique à travers les diagrammes de séquences met fin à cette étape d'étude des contraintes fonctionnelles de la conception de l'application. Cette étude des contraintes fonctionnelles correspond à la branche gauche du processus du modèle en Y. En résumé, cette étude des contraintes fonctionnelles a consisté d'abord en une étude préliminaire qui circonscrit le projet, ensuite en une capture des besoins fonctionnels. La capture des besoins fonctionnels identifie les fonctions nécessaires à la conception de l'application. Enfin, l'analyse de ces fonctions apporte la découpe informatique de ces fonctions.

Le deuxième chapitre de cette partie, intitulée « conception du logiciel », consistera à étudier les contraintes techniques de cette conception d'application web, c'est-à-dire l'étude de la configuration physique et logicielle du projet soumis à notre réflexion.

CHAPITRE V : CONTRAINTES TECHNIQUES

Ce chapitre comprendra deux grandes parties suivant le modèle 2TUP (figure 2) : La capture des besoins techniques et la conception générique. Ensuite la politique de sécurité se penchera, en troisième partie, sur la sécurisation de l'application en conception.

1. CAPTURE DES BESOINS TECHNIQUES

La spécification technique est une activité de la branche droite du Y du modèle 2TUP; elle est primordiale pour la conception d'architecture. Cette étape a lieu lorsqu'on a obtenu suffisamment d'informations sur les pré-requis techniques. Le matériel, à priori, doit être connu, à savoir les machines, les réseaux et les outils retenus pour le développement. En cours d'élaboration, viendront s'ajouter les contraintes non fonctionnelles identifiées dans les cas d'utilisation de la branche gauche. Les besoins techniques s'expriment suivant les deux points de vue que sont la spécification logicielle et la structure du matériel à exploiter.

1.1. SPECIFICATION LOGICIELLE

1.1.1. CAPTURE DES BESOINS

Les besoins techniques ont été recueillis en ce qui concerne la spécification logicielle. Ils se résument en ceci :

1. Les utilisateurs ne doivent pas être obligés d'avoir la dernière version d'un navigateur. L'application doit pouvoir supporter tous les navigateurs.
2. Chaque utilisateur doit disposer d'une aide contextuelle qui l'aide à exploiter le système de la manière la plus efficace.
3. L'utilisateur doit se connecter et être reconnu du système pour pouvoir y travailler.

   L'authentification est le mécanisme qui protège le système des intrusions externes.
4. L'administrateur doit pouvoir gérer les erreurs d'utilisation du système

5. L'administrateur doit pouvoir configurer tout le système.

Cette capture de besoins de la spécification logicielle conduit à un choix d'outils de développement approprié. Ces outils de développement sont choisis suivant leur disponibilité et leur performance relative.

1.1.2. OUTILS DE DEVELOPPEMENT

Dans le cadre de ce travail, il a été utilisé différents outils informatiques pour modéliser l'application. Chaque outil ayant ses spécificités, l'utilisation s'est faite selon les besoins. Ce sont :

NetBeans 6.5: c'est un plugin[2] intégré à l'EDI[3] NetBeans, il est très puissant avec de grandes possibilités de personnalisation, mais assez lourd pour les ordinateurs peu puissants. Cependant, la gratuité de ses plugins fait de netbeans un outil de développement accessible comparativement à « élipse », son concurrent. C'est aussi à partir de la version 6.5 de netbeans que le support PHP a été intégré.

PowerAMC : PowerAMC permet de réaliser tous les types de modèles informatiques. Il reste un des seuls qui permettent de travailler avec la méthode Merise[4]. Cela permet d'améliorer la modélisation, les processus, le coût et la production d'application. PowerAMC a été choisit car nous l'avions déjà utilisé et il était déjà présent dans notre environnement de travail. Il est payant.

1.1.3. PRESENTATION DES LANGAGES ET LOGICIELS DE CONFIGURATION

Trois logiciels seront utilisés dans la construction de l'architecture technique. Apache, PHP, MySQL. Ses logiciels ont chacun des rôles différents.

---

[2] voir glossaire

[3] voir glossaire

[4] voir glossaire

Apache est le SERVEUR WEB. Son rôle est d'écouter les requêtes émises par les navigateurs qui demandent des pages web, comme firefox, internet explorer et autres, de chercher la page demandée et de la renvoyer.

PHP (PHP Hypertext Preprocessor) est un *langage de script*. Il permet, de décrire dans une page web, un affichage dynamique d'information, c'est-à-dire que le texte affiché peut être contenu dans des variables. Les instructions PHP sont généralement contenues dans des fichiers d'extension php. Ces fichiers peuvent contenir du HTML[5], entremêlé avec le code PHP. PHP est un open-source[6].

Les principaux concurrents de PHP sont Perl, ASP.NET, Ruby, Java Server Pages (JSP) et ColdFusion.

Par rapport à tous ces produits, PHP possède plusieurs avantages :

- La performance : la rapidité de traitement des requêtes.
- Des interfaces vers différents Systèmes de gestion de bases de données : MySQL, PostgreSQL, Oracle, InterBase, SyBase, SQLite... peuvent être directement connectés à PHP.
- Des bibliothèques intégrées : La génération d'images, la gestion de courriers électroniques, la production de documents PDF avec seulement quelques lignes de codes.
- Le Coût : PHP est gratuit.
- Portabilité : Un code PHP peut généralement être utilisé sans modification sur différents systèmes d'exploitation (Unix, Windows).
- Disponibilité de la documentation : Zend Technologies, à l'adresse www.zend.com, fournit un grand nombre de ressources et d'informations complètes sur le langage.

[5] voir glossaire

[6] Voir glossaire

JavaScript sera aussi utilisé. Le JavaScript est un langage important parce qu'il s'agit du langage des navigateurs web. Le JavaScript est incontournable pour une solide application web car il se trouve dans tous les navigateurs et il est léger.

EASYPHP : Il s'agit d'une plateforme de développement Web, permettant de faire fonctionner localement (sans se connecter à un serveur externe) des scripts PHP. EasyPHP n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (un serveur web Apache et un serveur de bases de données MySQL), un interpréteur de script[7] PHP, ainsi qu'une administration SQL phpMyAdmin. Il permet donc d'installer en une seule fois tout le nécessaire au développement local du PHP.

MySQL est un SYSTEME DE GESTION DE BASE DE DONNEES. Son rôle est de stocker les données, sous forme de tables, et de permettre la manipulation de ces données à travers le langage de requête SQL. Toutes solutions ayant ses avantages et ses inconvénients, la solution MysQL a pour avantages :

- Solution très courante en hébergement public,
- Très bonne intégration dans l'environnement Apache/PHP,
- Open Source,
- Facilité de déploiement et de prise en main,
- PHP est capable de passer à MySQL des requêtes à travers des fonctions. PHP possède également des fonctions pour dialoguer avec d'autres systèmes de gestion de base de données. C'est une raison du succès du couple « PHP+MySQL » dans la mise en place d'applications web.

[7] Voir glossaire

1.1.4. CHOIX DE L'ARCHITECTURE DU SYSTEME

Le MVC (Modèle-View-Controller) est une architecture qui permet d'organiser une application en 3 couches distinctes. Le MVC participe de l'organisation de l'application mais aussi de la sécurisation du code. Les trois couches du MVC sont :

- le modèle (model), qui contient la logique métier;
- la vue (view), qui regroupe tout ce qui a trait à la présentation des données ou des interactions utilisateurs;
- le contrôleur (controller), qui répond à des interactions utilisateurs en provenance de la vue, en appelant des traitements mis à disposition sous forme de méthode par le modèle. Cependant un sens de lecture est nécessaire.

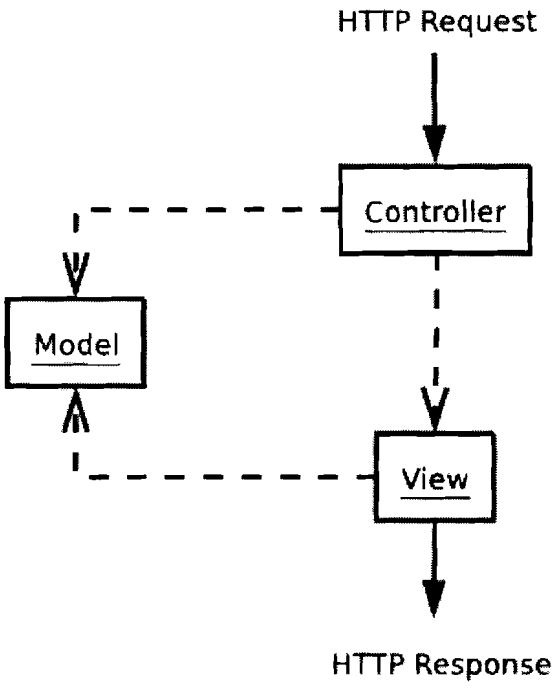


Figure 18: Modèle MVC Global

La figure 18 montre qu'il est très important que le modèle ne connaisse ni les contrôleurs, ni les vues. Les liens vont des contrôleurs vers le modèle, de temps en temps, des vues vers le modèle, mais jamais dans le sens inverse. Le modèle étant

totalement indépendant du reste, il pourra faire l'objet d'une réflexion et d'un développement à part.

Les données manipulées par le modèle doivent persister entre toutes les requêtes. C'est pourquoi un support de stockage MySql est utilisé.

La méthode MVC conduit à une spécification architecturale des fichiers de l'application.

Dans la situation présente, le fichier **index_admin.php**, fichier d'authentification qui est dans le dossier vue, fait appel au fichier **verifLogin2.php** du dossier contrôleur. Ce fichier verifLogin2.php exécute les opérations suivantes:

- Demande de connexion à la base de données

```
<?php
session_start();
error_reporting(0);
//nettoyage des sessions
$_SESSION=array();
session_unset();
include_once("connec.php");
.....
```

- demande d'incrémentation des tentatives de connexions dans un fichier compteur_acces.txt

```
include_once("./librairie_php/lib_statistique.php");
count_saisie("./data/compteur/compteur_acces.txt","visited","360
0","compteur_acces.time");
```

```
.....
```

- demande de récupération des données de sessions

```
//recupération des données de sessions

$nomsPostVar=array('user','connecte','login','identifiant','password'
,'pwd');

include_once("./librairie_php/db_triade.php");

$hashPostVar=hashPostVar($nomsPostVar);

.....
```

- demande de récupération du nom du navigateur web utilisé par l'utilisateur connecté

```
$nav_info=$_POST["info_nav"];

if ((eregi("microsoft",$nav_info)) || (eregi("internet
explorer",$nav_info))) {

    $navigateur="IE";

}else {

    $navigateur="NONIE";

}

.....
```

- demande d'identification de l'utilisateur par définition des variables de sessions

```
$nom=trim(ucwords($data['nom_pers']));

 $prenom=trim(ucwords($data['prenom_pers']));

$NomDOeuvre=trim(ucwords($selectNomDOeuvre['nom_oeuvre']))
;
```

```
$user="menudirecteurEtbl" ;

$id_pers=$data['id_pers'];

$_SESSION["nom"]=$nom;

$_SESSION["prenom"]=$prenom;

$_SESSION["nom_oeuvre"]=$NomDOeuvre;

   $_SESSION["user"]="menudirecteurEtbl";

   $_SESSION["id_pers"]=$id_pers;

   $_SESSION["nav"]=$nav;

   $_SESSION["os"]=$os;

   $_SESSION["ip"]=$ip;

   $_SESSION["id_session"]=$id_session;

   setcookie("nom","$nom");

   setcookie("prenom","$prenom");

   setcookie("id_pers","$id_pers");

.....
```

- Appel de la page désirée

```
header("Location: acces_admin.php?id_pers");
```

En conclusion, la spécification logicielle permet d'appréhender les besoins en logiciels de travail et de préciser la méthode de travail. Cette spécification logicielle, couplée à la spécification matérielle dans les paragraphes suivants, circonscrit davantage les besoins techniques.

1.2. SPÉCIFICATION MATÉRIELLE

1.2.1. CAPTURE DES BESOINS

La spécification matérielle s'exprime en ses besoins suivants :

1. Les utilisateurs doivent pouvoir se connecter de n'importe où, sur n'importe quel ordinateur.
2. L'application doit pouvoir fonctionner en intranet.
3. Plusieurs utilisateurs peuvent travailler en parallèle.
4. L'administrateur, qui s'occupe des réglages de l'application Web doit pouvoir administrer sans forcément se connecter sur Internet mais via un intranet au cas où il serait en manque de connexion.
5. Il n'est pas prévu d'interface système avec des serveurs de données autres que ceux mis en place.
6. L'application devra être centralisée sur un serveur web performant afin d'héberger le serveur d'application ainsi que la base de données (celle-ci pourra être par la suite séparée du serveur d'application si besoin). Une sauvegarde journalière de la base de données sera nécessaire.
7. La sécurité de l'application sera fonction du choix de l'hébergeur.

Ce recueil de besoins techniques matériels permet d'ébaucher un environnement de développement et une configuration du déploiement futur du système.

1.2.2. ARCHITECTURE 3-TIERS

La configuration que l'on met ici en place est souvent nommée "3-tiers" car on peut décomposer fonctionnellement notre application en tiers couches distinctes.

- COUCHE PRESENTATION: c'est la partie de l'application visible par les utilisateurs (on parle d'interface utilisateur). Dans ce cas-ci, elle se présentera sous la forme de pages HTML, composée de formulaires, boutons, etc.

- COUCHE METIER: c'est la "logique" de l'application elle-même, c'est-à-dire les algorithmes implémentés pour remplir les fonctions spécifiées.
- COUCHE ACCES AUX DONNEES: c'est la partie qui gère les données. Cette couche est souvent implantée avec un système de gestion de base de données. L'interface proposée est SQL. L'architecture 3-tiers est représentée sur la figure 19 :

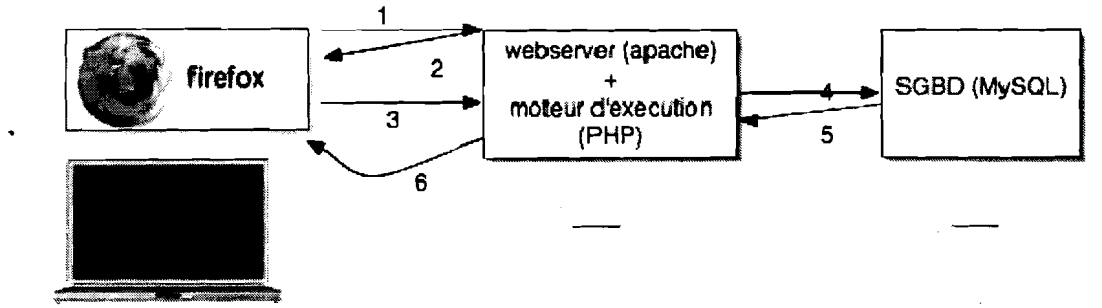


Figure 19: Architecture 3-tiers avec Apache+PHP+MySQL

1.2.3. CONFIGURATION MATERIELLE

La spécification technique du point de vue matériel exprime les contraintes relatives à la configuration du réseau matériel. Elles sont de nature géographique, organisationnelle et technique. Pour visualiser la configuration matérielle, nous utiliserons le diagramme de déploiement.

Le diagramme de déploiement correspond à la fois à la structure du réseau informatique qui prend en charge le système logiciel, et la façon dont les composants d'exploitation y sont installés. Un composant d'exploitation est une partie du système logiciel qui doit être connue, installée, déclarée et manipulée par les exploitants du système. Un composant d'exploitation doit être interchangeable entre différentes versions et peut être arrêté ou démarré séparément.

La configuration géographique actuelle de l'environnement de déploiement fait pencher vers un développement d'une solution à niveau local. Cette configuration connaîtra un redéploiement ultérieur régional, redéploiement dû à la reconfiguration géographique ultérieure du District d'Afrique de l'Ouest. La configuration matérielle est schématisée dans le diagramme suivant (figure 20) :

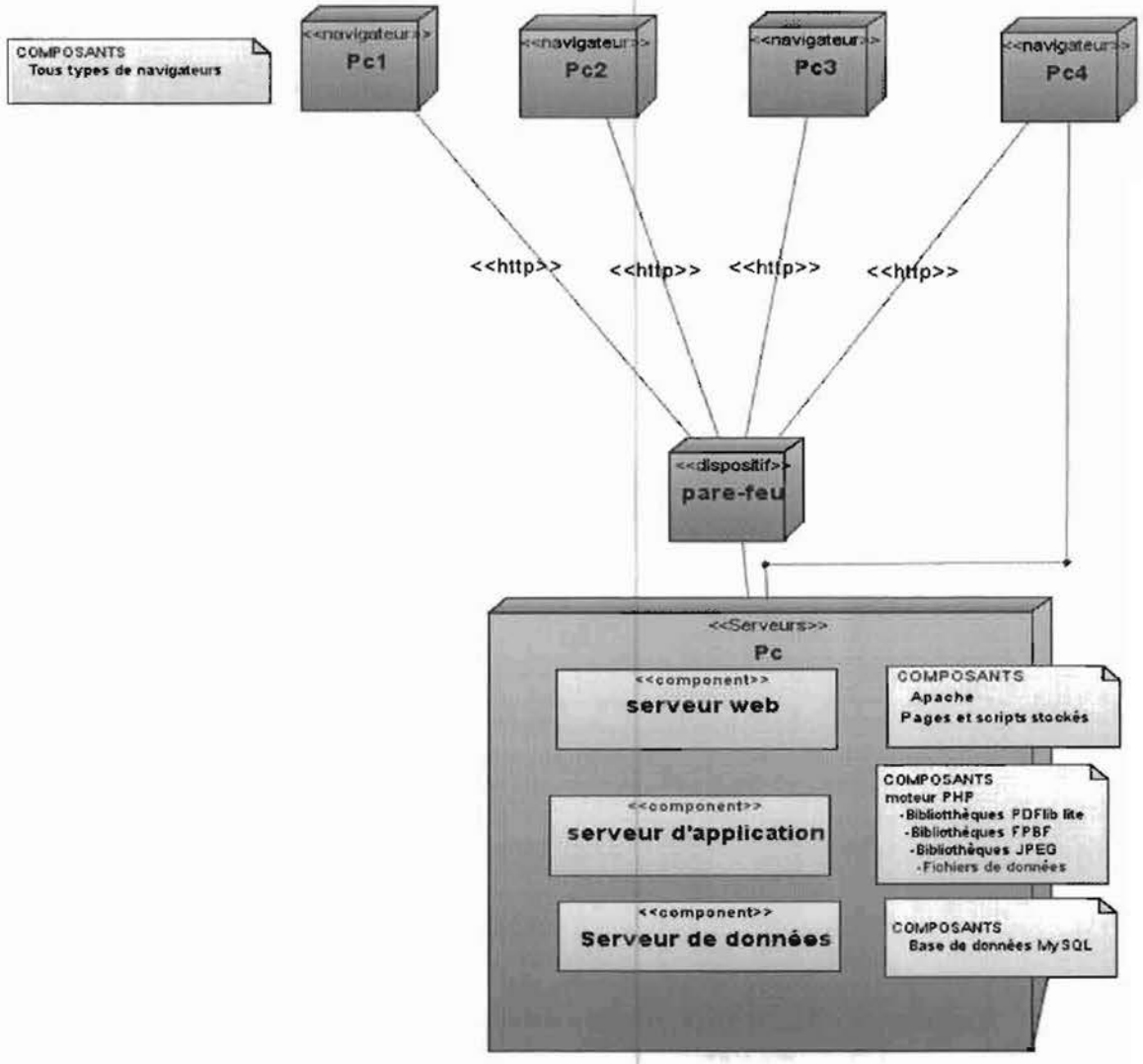


Figure 20: diagramme de déploiement

NB : PC : « Personal Computer »

A la fin de cette capture des besoins techniques, qui a consisté en la capture des besoins logiciels et en la capture des spécifications matérielles, la conception générique est la première mise en œuvre visuelle de l'application. C'est la deuxième activité de la branche droite du 2TUP. La conception générique est essentielle avant la conception préliminaire car elle offre le premier contact visuel entre l'utilisateur et son application.

2. LA CONCEPTION GENERIQUE : LE PROTOTYPAGE

Le prototypage est la clé de voûte du développement itératif. Les prototypes se différencient selon leur degré de réalisme. Il est effectué un prototypage horizontal et vertical.

2.1. PROTOTYPE HORIZONTAL OU MAQUETTE

Le prototype horizontal correspond à la couche de surface du logiciel. Il s'agit de l'interface homme-machine seule. Les composants de l'interface (fenêtres, boutons, menus, etc.) s'affichent, mais les commandes du logiciel ne fonctionnent pas. Ce premier prototype permet de réaliser un test de perception.

Un **test de perception** a été effectué. Il permet d'évaluer la compréhension de l'interface. Il consiste à montrer à l'utilisateur les principales fenêtres de l'application. Sans que l'utilisateur se serve de la souris, le concepteur demande d'expliquer comment il interprète les informations affichées à l'écran et de décrire le comportement qu'il attend du logiciel.

Cette première étape permet de vérifier le comportement local de l'interface. Elle sert aussi à identifier les points critiques où des problèmes d'utilisation sont susceptibles d'apparaître.

Après le test de perception, il en résulte que l'allure générale de l'application web devra respecter les règles suivantes, à savoir :

- Aller à l'essentiel,

- Structurer le contenu de la page,
- Homogénéiser la presentation,
- Faciliter la lecture,
- Satisfaire l'utilisateur en un clic c'est-à-dire gérer la profondeur de l'application,
- Optimiser le chargement des pages.

Ci-après quelques interfaces illustrant le prototypage horizontal de l'application :

La page d'identification de l'utilisateur présente une fenêtre de connexion (figure 21). Cette fenêtre possède deux entrées : l'identifiant de l'utilisateur et son mot de passe :

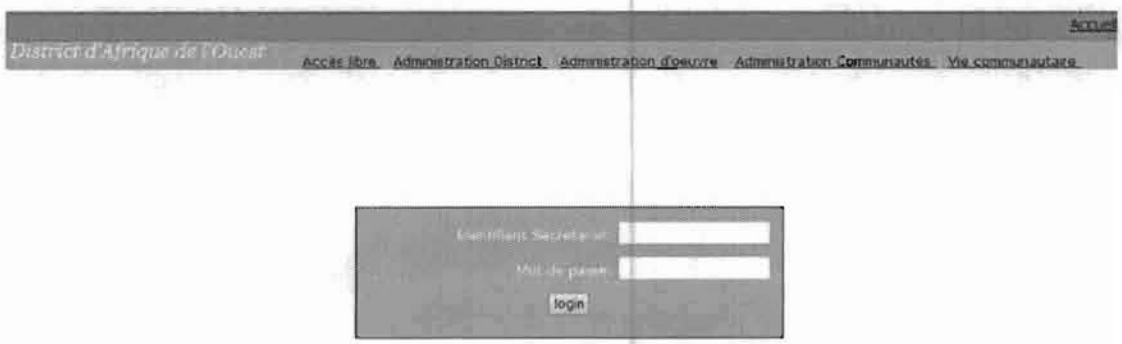


Figure 21: page d'identification de l'utilisateur

Après identification, l'utilisateur accède à la page des menus (figure 22). Cette page de menus est composée d'une barre horizontale contenant le lien vers la page d'administration « admin », le lien de fermeture de l'application « quitter » et quelques modules additionnels « aide », « forum »...

Les menus verticaux de gauche sont des menus de création et de modification des données. Et ceux de droite sont des menus de consultation des données. A chaque utilisateur correspond une page de menus selon ses droits de lecture ou de modification des données.

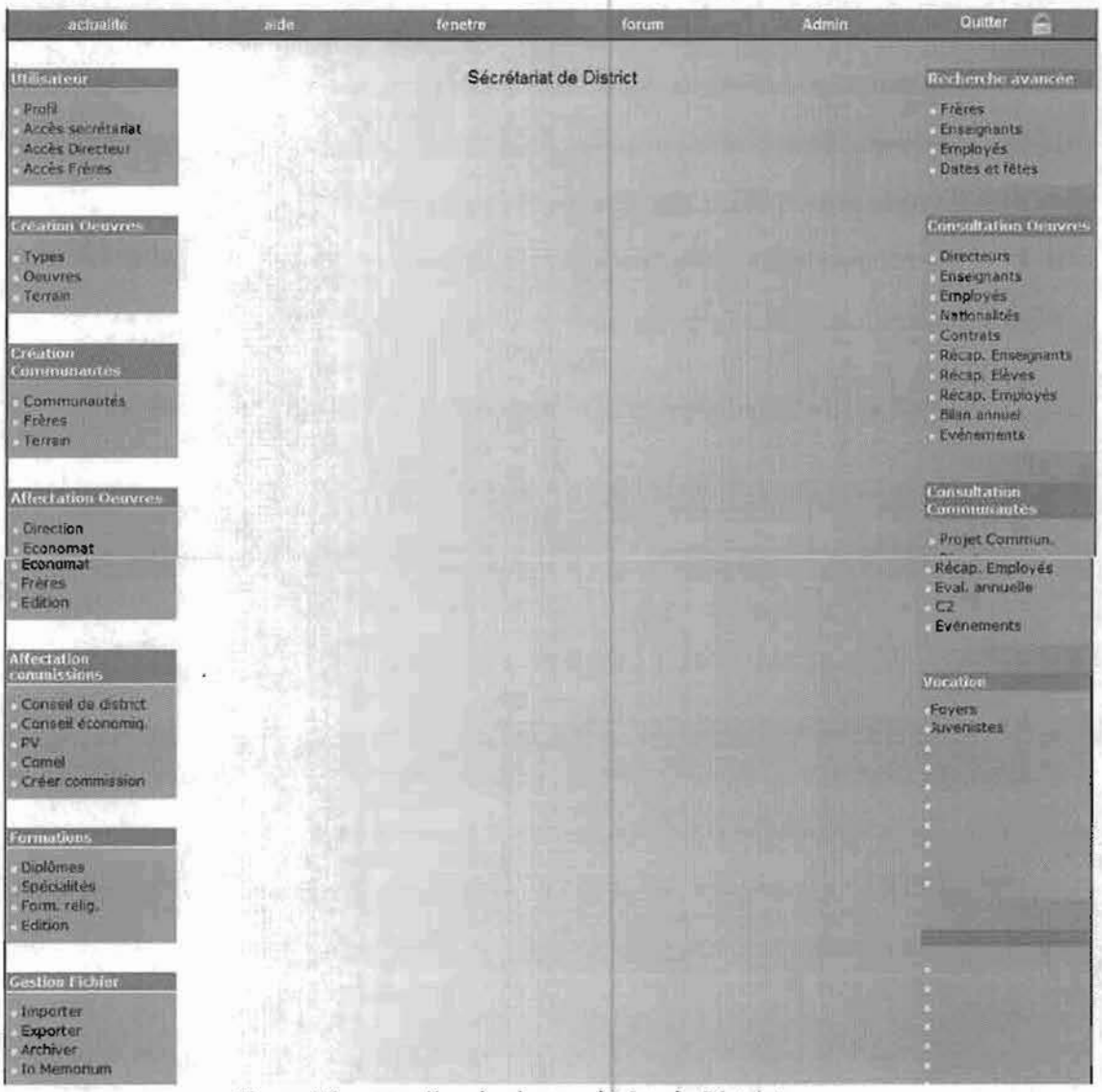


Figure 22: page d'accès du secrétaire de District

La figure suivante présente la gestion des mots de passe par le secrétaire (figure 23). Dans le menu gauche intitulé « utilisateur », un clic sur accès secrétariat permet de gérer les données de connexion : identifiant, mot de passe...

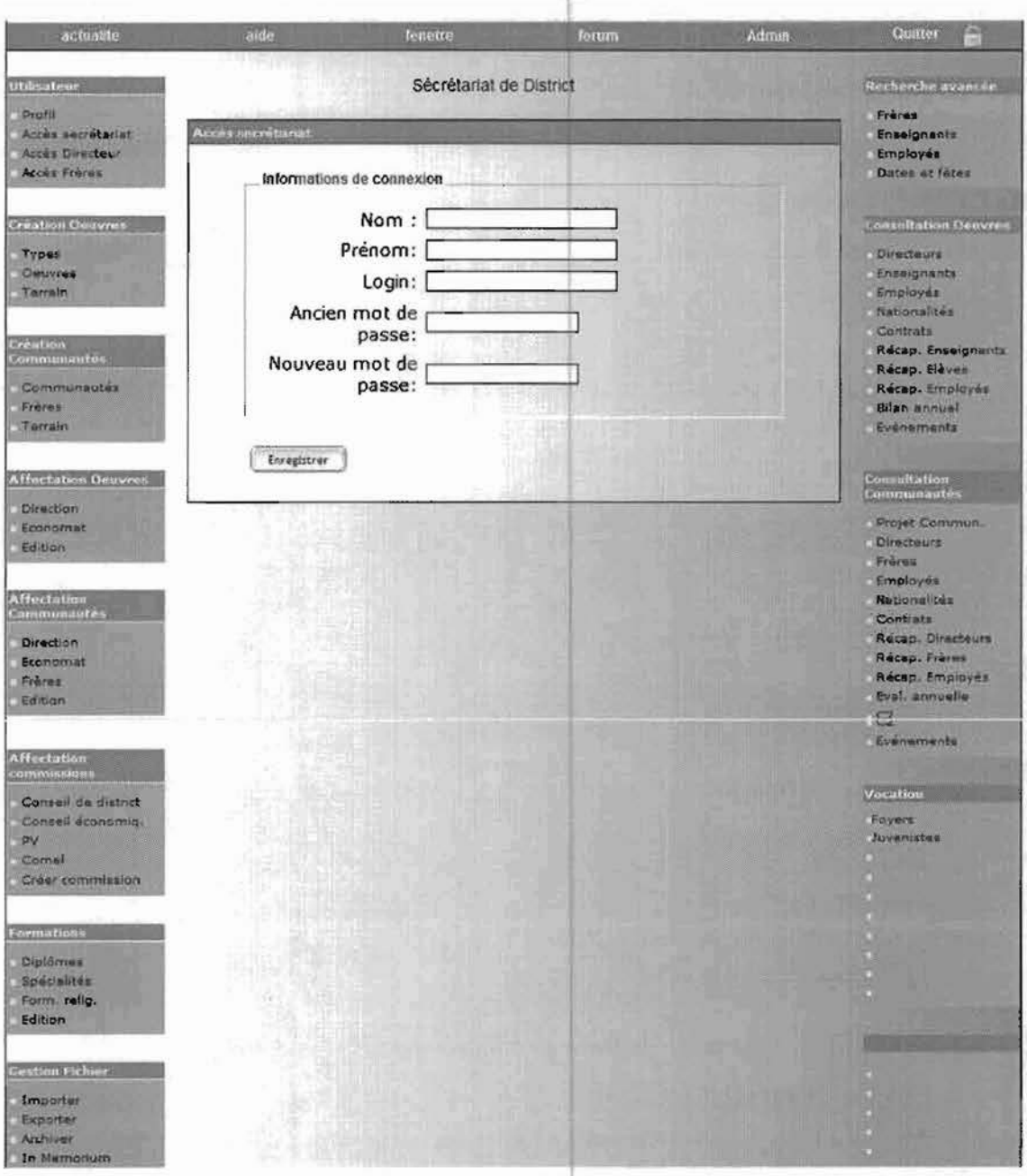


Figure 23: fenêtre de gestion de mot de passe

Ensuite, le secrétaire a le droit d'accès aux créations, modifications et suppressions des œuvres, communautés et Frères (figure 24). L'apparition des boutons « modifier œuvres », « imprimer », « supprimer »..., permet la gestion de ces modules. La figure suivante présente la page de création d'une œuvre grâce au menu gauche « création œuvres ».

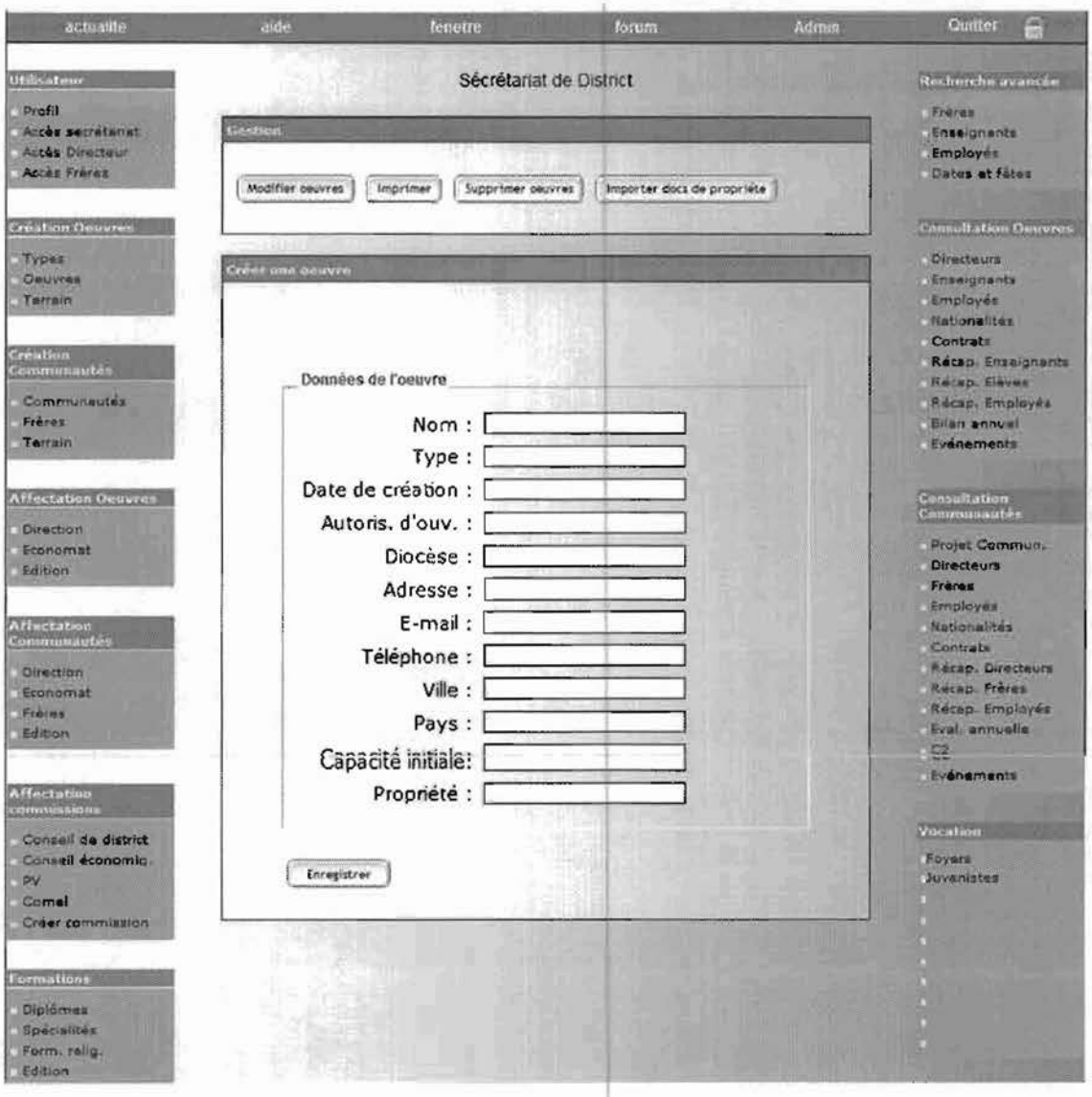


Figure 24: fenêtre de création d'une œuvre

Ensuite, grâce au menu gauche « création communautés », le secrétaire a la possibilité d'enregistrer un Frère (figure 25). L'insertion d'un calendrier permet d'harmoniser les formats des dates.

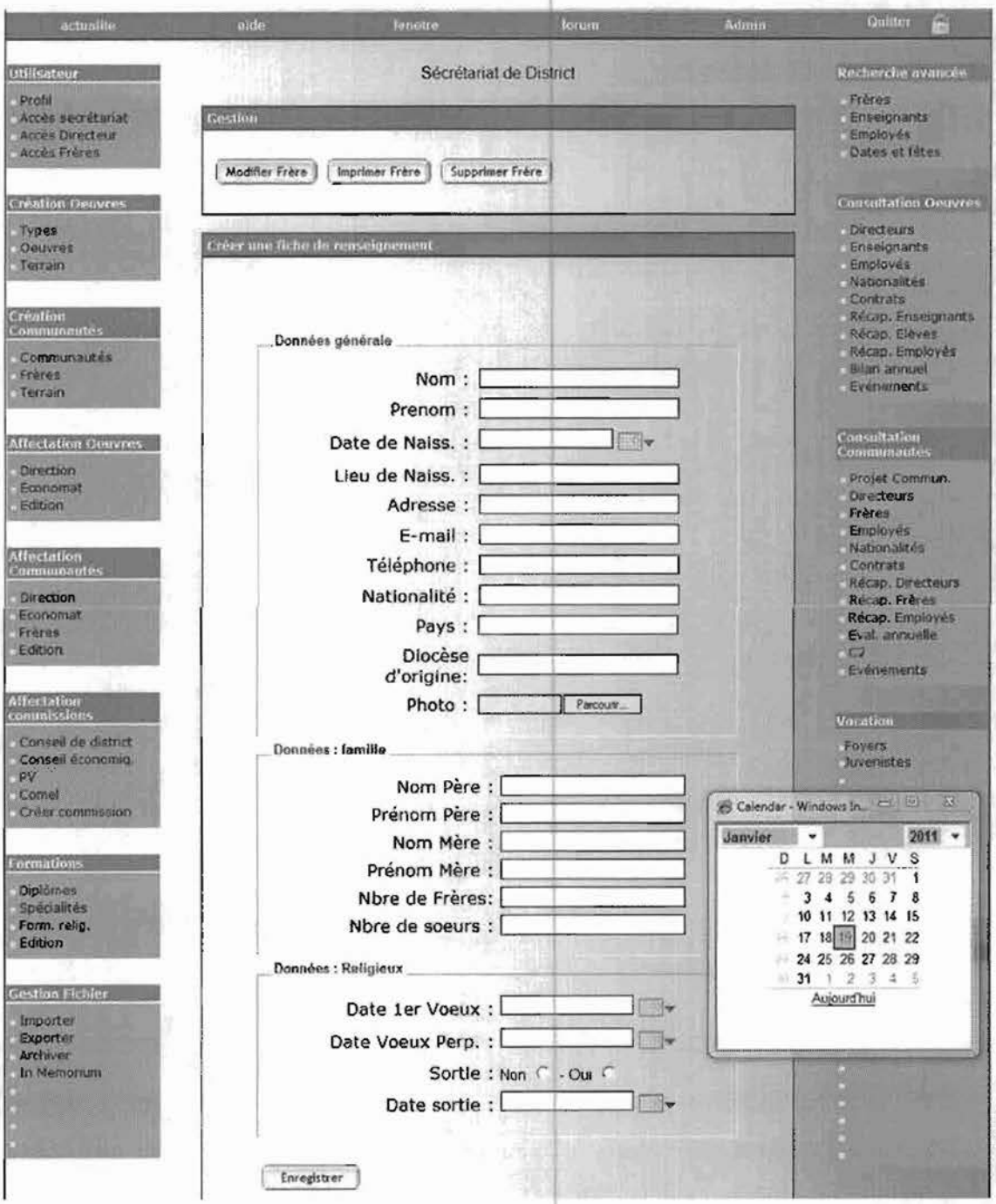


Figure 25: fenêtre d'enregistrement d'un Frère

2.2. PROTOTYPE VERTICAL

Un prototype vertical est ensuite développé. Il correspond à la mise en œuvre d'un ensemble cohérent d'une ou de plusieurs fonctionnalités. Sur un prototype vertical, l'utilisateur peut dérouler complètement une tâche significative du logiciel.

Le **test d'utilisabilité avec l'utilisateur** permet d'identifier les problèmes et d'analyser leurs causes. Des solutions ont été élaborées et mises en œuvre et ainsi de suite jusqu'à ce que les problèmes soient corrigés.

3. LA POLITIQUE DE SECURITE

La sécurité proposée prend en compte l'installation de l'application en réseau local, étant donné que le choix de l'hébergeur n'est pas encore définit.

3.1. SÉCURISATION DU CODE

La sécurisation n'est pas une activité indépendante de la programmation. Sécuriser son code c'est écrire de telle sorte que le code soit sûr c'est-à-dire qu'il puisse se défendre contre les attaques. Les besoins de sécurisation de code sont :

- Créer un fichier .htaccess

Les principales raisons d'utilisation des fichiers .htaccess sont :

  - Gérer l'accès à certains fichiers.
  - Protéger l'accès à un répertoire par un mot de passe.
  - Protéger l'accès à un fichier par un mot de passe.
  - Définir des pages d'erreurs personnalisées.

- Filtrer toutes les données fournies par l'utilisateur.
  - Vérifier les valeurs attendues
  - Vérifier les types des valeurs
- Sécuriser les chaînes pour SQL
  - Utiliser les fonctions de sécurisations des entrées : mysql_escape_string,...

- Protéger les sorties pour que le navigateur web du client ne se serve des valeurs que pour les afficher et non les modifier.

3.2. SECURISATION DU SERVEUR WEB ET PHP

Outre la sécurisation du code, l'installation et la configuration du serveur web et php sont également des éléments importants d'une politique de sécurité. Quelques moyens simples sont mis en œuvre pour sécuriser le serveur :

- Garder les logiciels à jour c'est-à-dire utiliser la dernière et la plus sécurisée des versions de php et d'apache.

- Configurer le fichier de configuration « php.ini » de php car une configuration par défaut est installée.

- Sauvegarder le fichier php.ini.

- Configurer le fichier de configuration httpd:conf du serveur apache car une configuration par défaut est fournie.

3.3. SECURISATION DU SERVEUR DE BASE DE DONNEES

Outre les mises à jour du logiciel, quelques opérations s'effectuent pour sécuriser la base de données.

- Sécuriser le compte administrateur de la base de données par mot de passe.

- Supprimer les comptes par défaut qui sont créés automatiquement à l'installation.

- Gérer les permissions d'accès aux tables.

- Créer des utilisateurs avec le moins de permissions possibles.

- Configurer le fichier my.ini de MySQL pour qu'il soit en adéquation avec le serveur Apache.

3.4. PROTECTION DU RÉSEAU

Tout comme l'on doit filtrer les données qui parviennent à l'application PHP, le trafic qui arrive sur le réseau doit être également filtré. Pour ce faire l'installation d'un pare-feu permettra d'éliminer le trafic indésirable et bloquer l'accès aux parties d'un réseau que l'on veut isoler.

3.5. SECURISATION DES ORDINATEURS ET DU SYSTEME D'EXPLOITATION

La politique de sécurisation des ordinateurs passe par une mise à jour régulière du système d'exploitation. Cela permet d'appliquer les correctifs de sécurité. En outre, l'installation d'un antivirus est obligatoire.

Cependant, il est essentiel que les serveurs qui exécutent l'application web se trouvent dans un environnement sécurisé. Vu la configuration du lieu possible d'installation, il sera inévitable de trouver une politique de lutte contre la poussière.

En conclusion, la sécurité n'est pas une activité indépendante. La politique de sécurité traverse le projet depuis la conception jusqu'à la réalisation de l'application.

CONCLUSION

La congrégation des Frères de Ecoles Chrétiennes en Afrique de l'Ouest, désirant améliorer sa gestion administrative, s'est engagée dans la gestion informatique de ses données. Cela a bien des avantages tels que la permanence du stockage, la rapidité de la recherche et la disponibilité de l'information en tout lieu pourvu d'une connexion internet.

Pour ce faire, le processus 2TUP nous a permis de capturer les besoins fonctionnels et les besoins techniques nécessaires à la réalisation de l'application. Ce processus nous a permis d'analyser les besoins fonctionnels et techniques de l'application et d'en établir une conception générique. En effet, grâce à la modélisation UML, nous avons tenté une analyse globale du système dans la branche gauche du 2TUP. Ensuite, le recueil des besoins techniques, dans la branche droite du 2TUP, a abouti à la proposition d'un prototype. Enfin, la jonction entre les deux branches de ce processus en Y est une étape cruciale de la conception, étape à laquelle nous nous attelions en fin de stage. Cette application est en cours de réalisation. Le module de l'administrateur et le module de la gestion des fichiers sont deux parties importantes du logiciel qui n'ont pas été assez approfondies durant ce temps de stage.

Ce stage nous a permis de nous appliquer à suivre, aussi rigoureusement que possible, une démarche de modélisation. Il nous a fait découvrir un processus de modélisation très itératif dont la compréhension et la bonne utilisation conduit lentement mais sans nul doute sûrement vers une réalisation méthodique d'application.

Ce stage nous a aussi fait apprécier le langage PHP. Les différentes recherches nous ont introduits dans les subtilités de ce riche langage surtout dans sa nouvelle version, php5.

Enfin, le désir de la congrégation des Frères des Ecoles Chrétiennes d'avoir cet outil informatique d'administration, et notre soif d'approfondissement né de ce temps de stage, font de ce document non pas un aboutissement mais un départ vers de nouvelles découvertes.

GLOSSAIRE

Atelier de génie logiciel : Un atelier de génie logiciel est un ensemble de programmes informatiques permettant eux-mêmes de produire des programmes de manière industrielle.

HTML : L'*Hypertext Markup Language*, généralement abrégé HTML, est le format de données conçu pour représenter les pages web. C'est un langage de balisage qui permet d'écrire de l'hypertexte, d'où son nom.

IDE : Un environnement de développement intégré (*EDI* ou *IDE* en anglais pour *Integrated Development Environment*) est un programme regroupant un ensemble d'outils pour le développement de logiciels.

Interpréteur de script : C'est un outil informatique ayant pour tâche d'analyser, de traduire et d'exécuter un programme écrit dans un langage informatique.

Merise : La méthode Merise est une méthode d'analyse, de conception et de réalisation de systèmes d'informations informatisés.

Open source : La désignation *open source* s'applique aux logiciels dont la licence respecte des critères précisément établis par l'*Open Source Initiative*, c'est-à-dire la possibilité de libre redistribution, d'accès au code source et de travaux dérivés.

Pare feu : Un pare-feu (appelé aussi *coupe-feu*, *garde-barrière* ou firewall en anglais), est un système permettant de protéger un ordinateur ou un réseau d'ordinateurs des intrusions provenant d'un réseau tiers, notamment internet.

Plugin : un plugin est un logiciel qui complète un logiciel hôte pour lui apporter de nouvelles fonctionnalités.

SGBD : Un Système de Gestion de Base de Données est un ensemble de logiciels qui sert à la manipulation des bases de données. Il sert à effectuer des opérations ordinaires telles que consulter, modifier, construire, organiser, transformer, copier, sauvegarder ou restaurer des bases de données.

BIBLIOGRAPHIE

[1] Douglas Crockford, JavaScript *Gardez le meilleur !*, Pearson, paris, 2008.

[2] Isabelle Canivet, *Bien rédiger pour le web*, Eyrolles, Paris, 2010.

[3] Kazi, A. B. & Rostane, Z., *Suivie des enseignements du LMD par application de la méthode 2TUP*, Mémoire d'ingéniorat, Université Abou Bekr Belkaid de Tlemcen, Algérie, 2007.

[4] Luke Welling et Laura Thomson, *Php et MySQL (4e édition),* Pearson, Paris, 2009.

[5] *Manuel de procédures du District d'Afrique de l'Ouest*, fec, 2010

[6] Michel Divas, *Java et la programmation objet*, dunod, Paris, 2001.

[7] N. Kettani et al., *De Merise à UML*, Eyrolles, Paris, 2001.

[8] Philip J. Pratt, *Initiation à SQL,* Eyrolles, Paris, 2001.

Sitographie

Ces références ont été consultées durant la période de stage, de Septembre à décembre 2010.

[1]Apache foundation, Configuration de serveur apache http version 2.3, Apache Software Foundation, 2010
httpd.apache.org/docs/trunk/fr/howto/htaccess.html
httpd.apache.org/docs/trunk/fr/sections.html

[2] Diane GAMACHE, Cours de conception informatique, professeure au département d'informatique, cégep de Drummondville, Québec, canada, 1999.
www.cdrummond.qc.ca/cegep/informat/Professeurs/diane/Cours/420-CS1/420-CS1.html

[3] Encyclopédie en ligne, fr.wikipedia.org

[4]Jean-François NOGIER, Ergonomie du logiciel et design web, le manuel des interfaces utilisateurs, 2007, www.usabilis.com/livre/table.htm

[5] M. Chaouki et B. Abdorrahman, Mise en oeuvre d'un système d'information Pour la gestion des ressources humaines, Mémoire de fin de cycle, ENSIAS, Maroc, 2007, www.ensias.net/files/Rapport%20-%20SI%20RH.pdf

[6] Panorama des méthodologies utilisées pour le développement d'applications web, www.dsi.cnrs.fr/methodes/gestion-projet/methodologie/etat-art.htm

[7] triade, Transparence et rapidité de l'Informatique au Service de l'Enseignement, 2009, www.triade-educ.com/accueil/index1.php

ANNEXES

A. DESCRIPTION DETAILLEE DES CAS D'UTILISATION

La fiche de description textuelle d'un cas d'utilisation n'est pas normalisée par UML. Ces fiches servent à fixer les idées et n'ont pas pour but de spécifier un fonctionnement complet et irréversible. Les descriptions vont être organisées de la façon suivante :

- Les diagrammes (*optionnelle*): plusieurs diagrammes vont apparaitre (mais pas nécessairement) pour apporter une compréhension additive au cas d'utilisation.
- Un sommaire d'identification : va résumer les propriétés du cas d'utilisation dans une description sommaire d'intentions et d'actions.
- Une description détaillée : des Préconditions au déclenchement du cas d'utilisation doivent être spécifiées, un scénario nominal décrivant celui-ci additionné à des scénarios alternatifs et d'exceptions.

Il sera présenté, dans les pages suivantes, quelques cas d'utilisation significatifs.

CAS D'UTILISATION 1 : GERER LES FICHIERS

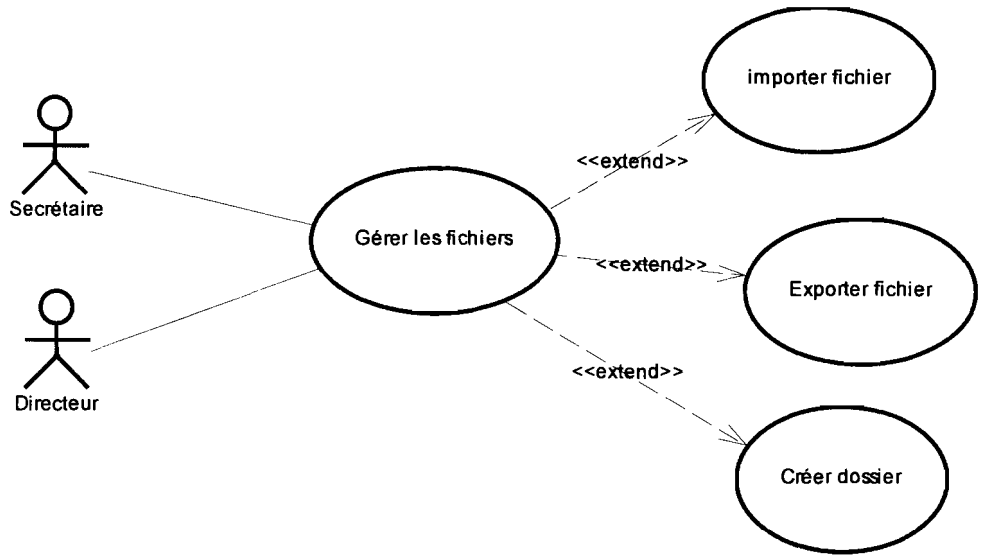


Figure 26: Raffinement de cas d'utilisation : GERER LES FICHIERS

Sommaire d'identification

Titre : ***Gérer les fichiers***
Intention: *pouvoir gérer tous types de rapports*
Résumé : *créer un type de rapport*
Acteur : *Le directeur, le secrétaire*

Préconditions :

Les domaines (scolarité/année et communauté/année) existent déjà.

Le directeur est authentifié dans le domaine désiré.

Scénario nominal :

Ce cas d'utilisation commence lorsque le directeur demande au système de créer un type de rapport.

Enchaînement nominal

1. Dans le domaine choisi, il donne un nom au type de rapport.
2. Le directeur valide la création du type de rapport.

Si le type existe déjà, déclencher **[Exeption 1 : TypeDeRapportExistant]**

Exceptions

[Exeption 1 : TypeDeRapportExistant] :

3. Le système renvoie un message d'erreur tant que le nom du type de rapport n'est pas valide.

Post conditions

Un dossier est créé portant le nom choisi.

Titre : *Gérer les fichiers*

Intention: *pouvoir gérer tous types de rapports*

Résumé : *importer un document*

Acteur : *Le directeur, le secrétaire*

Préconditions :

1. Les domaines (scolarité/année et communauté/année) existent déjà.
2. Le directeur est authentifié dans le domaine désiré.

Scénario nominal

1. L'utilisateur spécifie le chemin du fichier à importer.
2. On enregistre dans la base de données les informations récupérées du fichier.

Exception2 : [NomDeFichierInvalide] : Le fichier spécifié n'est pas valide

Exception3 : [NomDeFichierInexistant] : Le fichier spécifié n'existe pas.

Description :

- Un message d'erreur est affiché.
- La base de données n'est pas changé.

- **Titre :** ***Gérer les fichiers***
- **Intention:** *pouvoir gérer tous types de rapports*
- **Résumé :** *Exporter un document*
- **Acteur :** *Le directeur, le secrétaire*

Préconditions :

1. Les domaines (scolarité/année et communauté/année) existent déjà.
2. Le directeur est authentifié dans le domaine désiré.

Scénario nominal :

1. Récupération de l'ensemble des données.
2. Un fichier est créé avec l'ensemble des paramètres de l'application qui comprend les données relatives au district.

Post-condition :

Un fichier XML valide est généré.

CAS D'UTILISATION 2 : GERER LES EMPLOYES

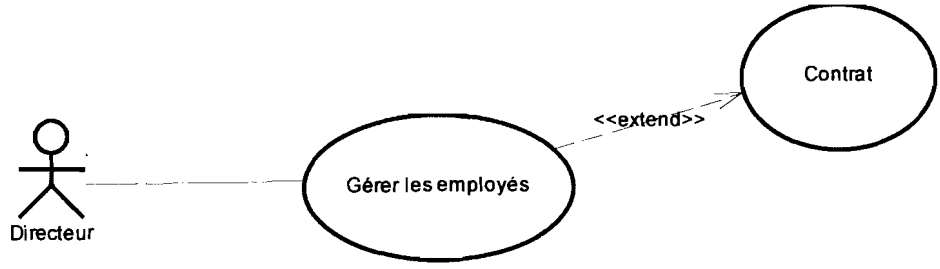


Figure 27: Raffinement de cas d'utilisation : GERER LES EMPLOYES

Titre : *Gérer les fiches des employés*

Intention: *Créer un dossier d'employé.*

Résumé : *créer un nouvel employé, modifier un dossier existant. Un religieux enseignant dans l'établissement est employé du directeur d'établissement.*

Acteur : *Le directeur*

Préconditions :

Les domaines (scolarité/année et communauté/année) existent déjà.

Le directeur est authentifié dans le domaine désiré.

Les grades possibles sont créés.

Scénario nominal :

Ce cas d'utilisation commence lorsque le directeur demande au système de créer un nouveau dossier d'employé.

Enchainement a : enregistrement de l'employé

1. Le directeur demande la création d'un dossier d'employé.
2. Le directeur enregistre les informations de l'employé.
3. Il valide le dossier de l'employé en construction.

3.1. Si les champs obligatoires ne sont pas remplis **[Exception 4 : ChampObligatoireNonRempli]**

3.2. Si l'employé existe déjà **[Exception 5 : EmployeExistant]**

Enchaînement b : enregistrement du contrat

1. Le directeur demande l'enregistrement d'un contrat.
2. Le directeur fournit les données de contrat.
3. Si les dates ne sont pas mentionnées, exécuter **[Exception 6 : DateManquantes]**

Enchaînements alternatifs

a) Modifier un dossier d'employé existant ou en construction.

Le directeur demande modification d'un dossier.

Le directeur valide les modifications du dossier.

b) Supprimer un employé existant

Pré-condition : l'employé existe déjà

Le directeur demande suppression du dossier de l'employé.

Le directeur confirme la suppression du dossier de l'employé.

Ce cas d'utilisation se termine lorsque le directeur a validé la création du dossier de l'employé.

Exceptions

[Exception 4 : ChampObligatoireNonRempli] :

Le système renvoie un message d'erreur tant que les champs obligatoires ne sont pas remplis.

[Exception 5 : EmployeExistant] :

Le système renvoie un message d'erreur tant que le matricule existe déjà.

[Exception 6 : DateManquantes]

Le système renvoie un message d'erreur de dates manquantes.

Post-condition

Mise à jour de la base de données.

Besoins d'affichage

L'utilisateur doit pouvoir :

a. Lister les employés.

b. Lister leur grade

CAS D'UTILISATION 3 : GERER LES FRERES

Titre : ***Gérer les frères***

Intention: *gestion des frères.*

Résumé : *enregistrer un frère*

Acteurs : *Le secrétaire*

Pré-conditions:

1- Le secrétaire est authentifié.

2- Les maisons de formations sont créées.

3- Les grades possibles sont créés.

Scénario nominal :

Ce cas d'utilisation commence lorsque le secrétaire demande au système d'enregistrer un frère.

Enchaînement a : enregistrement d'identification

1. Le secrétaire demande l'enregistrement d'un frère.
2. Il enregistre les données d'état civil, de religieux du frère.

Enchaînement b : enregistrer cursus de formation

1. L'utilisateur sélectionne une maison de formation.
2. Il introduit les dates de passage du frère dans cette maison.
3. Si le Frère n'a pas été identifié, exécuter **[Exception 7 : FrereInexistant]**
4. Si le grade n'a pas été identifié, exécuter **[Exception 8: GradeInexistant]**

Enchaînements alternatifs

a) Modifier l'enregistrement d'un frère en construction ou déjà enregistré.

Le secrétaire peut modifier un enregistrement en construction ou déjà enregistré.

b) Supprimer un enregistrement

Le secrétaire peut supprimer un enregistrement.

Exception

[Exception 7 : FrereInexistant] :

Le système renvoie un message d'erreur tant qu'il y a doublon ou inexistence.

[Exception 8 : GradeInexistant] :

Le système renvoie un message d'erreur tant que le grade n'est pas trouvé.

Ce cas d'utilisation se termine lorsque le secrétaire a validé une promotion.

Besoin d'affichage

Le système doit pouvoir

1. Lister les frères
2. Lister les maisons de formation
3. Lister les grades

CAS D'UTILISATION 4 : GERER LES INSTITUTIONS

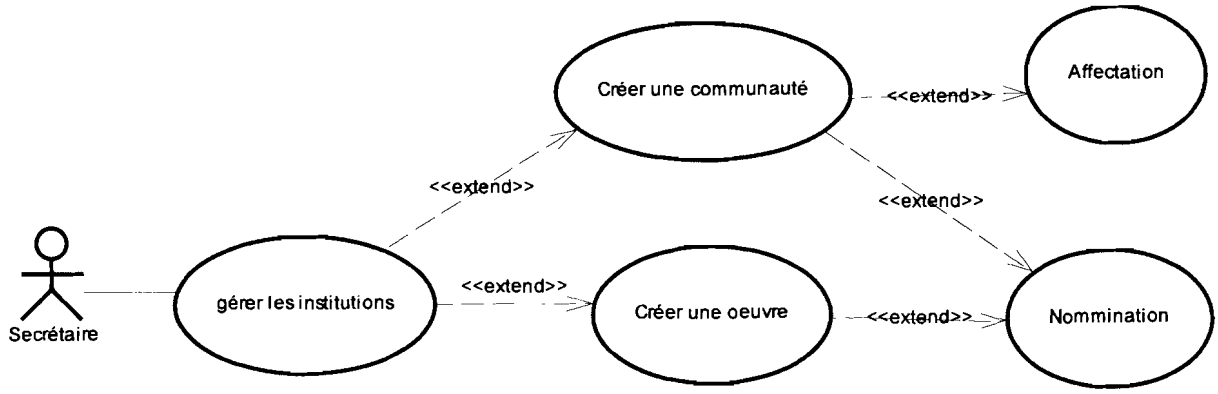


Figure 28: Raffinement de cas d'utilisation : GERER LES INSTITUTIONS

Titre : *Gérer les institutions*

Intention: permettre d'enregistrer les données des œuvres et des communautés

Résumé : *créer une communauté, une œuvre ; affecter les Frères dans les communautés, dans les œuvres*

Acteur : *Le secrétaire*

Pré-conditions :

1. Le secrétaire est authentifié.
2. Tout Frère se trouve dans une communauté.

Scénario nominal

Ce cas d'utilisation commence lorsque le secrétaire demande création d'institution.

Enchaînement a : créer une communauté.

1. Le secrétaire demande la création d'une communauté.
2. Le secrétaire remplit nom/adresse, date de création de la communauté.
3. Le secrétaire valide la création de la communauté.

Si la communauté existe déjà, déclencher **[Exception 9 : CommunauteExistante]**

Enchaînement b: créer une œuvre.

1. Le secrétaire demande la création d'une œuvre.
2. Le secrétaire remplit nom/adresse, date de création de l'œuvre.
3. Le secrétaire valide la création de l'œuvre.

Si l'œuvre existe déjà, déclencher **[Exception 9 : CommunauteExistante]**

Enchaînement c : affecter à une communauté

1. L'utilisateur sélectionne la communauté
2. Il introduit les dates d'affectation
3. Si le Frère n'a pas été identifié, exécuter **[Exception 7 : FrereInexistant]**
4. Si un membre appartient à deux communautés, déclencher **[Exception 10 : MembreDejaAffecte]**

Enchaînement d : nommer à une fonction

1. L'utilisateur sélectionne la function.
2. Il introduit les dates de nomination.
3. Si le Frère n'a pas été identifié, exécuter **[Exception 7 : FrereInexistant]**

Enchaînements alternatifs

a) Enregistrer un Frère hors du District d'Afrique de l'Ouest.

Cette alternative rejoint le scénario nominal.

b) Modifier une communauté en construction ou déjà créée.

Le secrétaire peut modifier les références d'une communauté.

c) Enregistrer un décès ou démission

Le secrétaire peut enregistrer le décès d'un membre de la communauté en y introduisant sa date de décès ou de démission.

d) Suppression d'enregistrement

L'utilisateur peut supprimer un enregistrement de Frère sans supprimer les communautés ni les œuvres ni les fonctions exercées.

Exceptions

[Exception 9 : CommunauteExistante] :

Le système renvoie un message d'erreur d'une communauté non enregistrée.

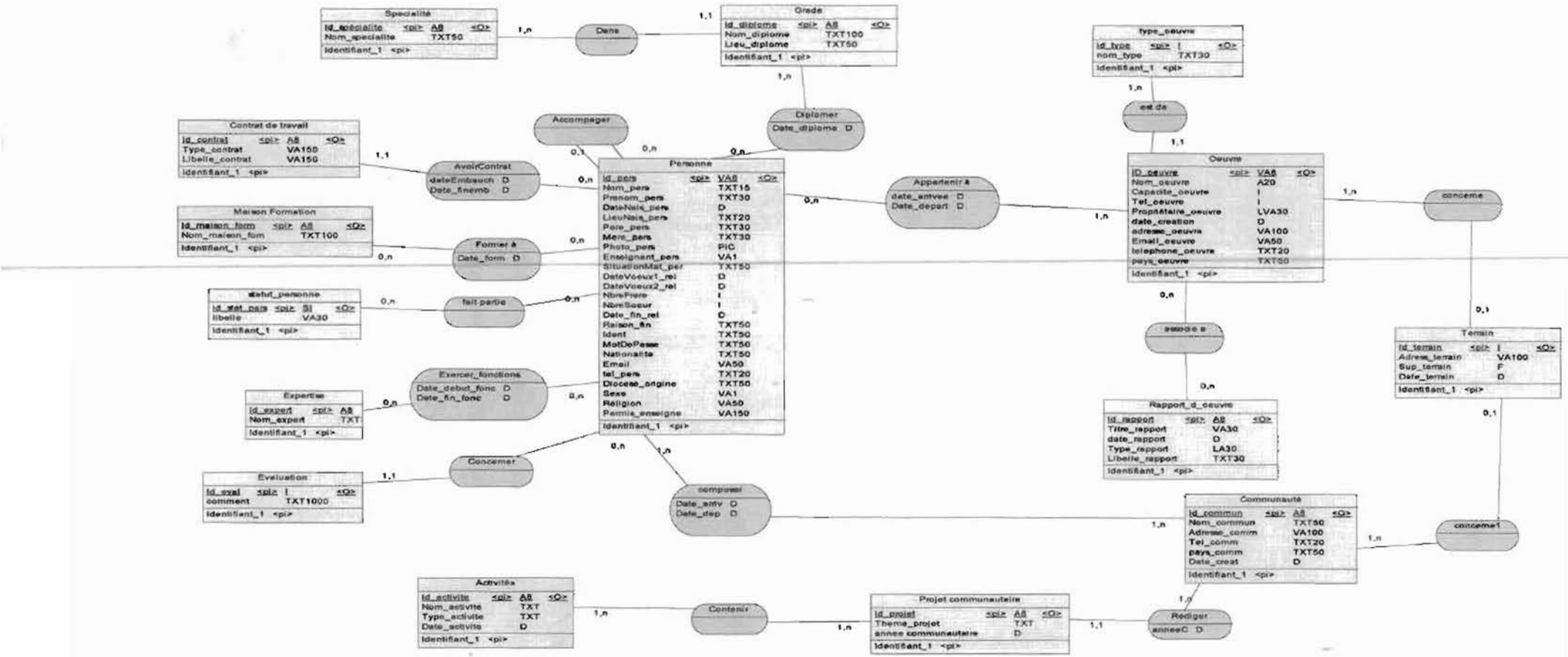
[Exception 10 : MembreDejaAffecte] :

Le système affiche un message d'erreur tant que le membre n'est pas réaffecté.

Ce cas d'utilisation se termine lorsqu'un enregistrement de Frère a été validé.

Besoin d'affichage

4. Lister les Frères
5. Lister les communautés
6. Lister les fonctions



B. Modèle conceptuel de données