REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI DE TIZI OUZOU
FACULTE DE GENIE ELECTRIQUE ET INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDES

En vue de l'obtention du diplôme de master 2 en informatique.

Thème :

Développement d'une application Web pour la gestion de l'emploi du temps.
Cas :
Département informatique

Proposé et dirigé par :

Promotrice : Mme BOUARAB.F

Jury composé de:

Président(e):Mme AOUGHLIS.F
Examinateurs :Mr HABET .MS
Examinateurs :Mme YESLI.Y

Réaliser par:
Melle AILAM kamelia
&
Melle LOUNIS Fazia

Promotion : 2012/2013

Remerciements

Nous tenons à remercier vivement notre promotrice Mme BOUARAB_DAHMANI FARIDA de nous avoir proposé ce sujet, pour la qualité de son encadrement, et son suivi durant toute la durée du projet.

Nous remercions chaleureusement les membres du jury pour l'honneur qu'ils nous font en acceptant de juger ce modeste travail.

Enfin, nous remercions toutes les personnes ayant contribué de près ou de loin au bon accomplissement de notre travail.

JE DÉDIE CET HUMBLE TRAVAIL À :

LA MÉMOIRE DE MON TRÈS CHER PAPA QUI MA TOUJOURS SOUTENU ET QUI SERA TOUJOURS GRAVER DANS MON CŒUR;

MA CHÈRE MAMAN POUR SON AFFECTION, SES SACRIFICES ET SON AMOUR ;

MES ADORABLES FRÈRES : KAMEL, KAOUSSENE ET KOCEILA ;

MA TENDRE SŒUR : KATIA ;

MON TRÈS CHER EPOU H. AGHILES QUI A TOUJOURS CRU EN MOI,

ET TOUTE SA FAMILLE ;

MA TRÈS CHÈRE BINÔME FAZIA ET TOUTE SA FAMILLE ;

TOUTE LA PROMOTION 2013

KAMELIA

JE DÉDIE CET HUMBLE TRAVAIL À

TOUTE LA FAMILLE

MES AMIS

MA BINÔME KAMELIA , SA FAMILLE

TOUTE LA PROMOTION 2013

FAZIA.

Sommaire

Sommaire

Chapitre III : Réalisation

Sommaire

Liste des figures

Liste des figures

Liste des tableaux

Introduction Générale

Introduction Générale

Qui dans la vie, n'a pas été confronté à la problématique de la gestion du temps. Des organisations (compagnies aériennes, entreprises de production, hôpitaux, établissements éducatifs,…etc) possédant leurs propres normes et critères sont confronté au même souci.

Parmi la vaste famille de problèmes de planification d´horaire, celui de l'élaboration de l´emploi du temps dans les établissements éducatifs, qui exploite des ressources humaines et donc financières. Ce problème est très important. En effet un mauvais emploi du temps influe directement et négativement sur le niveau de l´acquisition des étudiants.

Le problème de l´emploi du temps est ardu, dont la réalisation à la main est une tâche draconienne qui peut mobiliser plusieurs personnes voir plusieurs jours de travail. Sans oublier, que toute modification des données peut complètement remettre en cause la solution trouvée.

D'une manière générale, le problème de l'emploi du temps consiste à définir un certain nombre d'affectations qui permettent d'assigner plusieurs ressources (humaines, matérielles,…etc) sur une période de temps, tout en respectant les contraintes imposées par les entités citées (disponibilité des ressources humaines, matérielles,…etc).

Ces difficultés ont induit l'idée d'assister par ordinateur l'élaboration des emplois du temps en adoptant des outils et applications robustes permettant de faciliter cette tâche.

Dans ce cadre s'inscrit notre projet de fin d'études qui consiste à mettre en place une application web pour la gestion des emplois du temps afin de faciliter la vie des personnes qui s'en chargent, ou tous autres acteurs faisant partie du cercle éducatif, pour bien mener la gestion des séances de cours et ainsi, exploiter au mieux les ressources humaines et matérielles .

Afin de mener à bien notre travail, nous avons adopté la démarche suivante :

- Le premier chapitre intitulé «le problème des emplois du temps», présente quelques notions sur le problème des emplois du temps dans le domaine de la pédagogie.
- Le deuxième chapitre qui s'intitule « Analyse et conception », est consacré à l'analyse et à la conception de l'application proprement dite.
- En fin le troisième chapitre intitulé « Réalisation » porte sur la réalisation et l'implémentation de l'application ainsi que son fonctionnement.

Chapitre 1

I.1 Introduction

Le problème de l´emploi du temps est un problème ardu dont la réalisation à la main est une tâche draconienne qui peut mobiliser plusieurs personnes plusieurs jours de travail. Sans oublier, que toute modification des données du problème peut complètement remettre en cause la solution trouvée.

La construction d'emploi du temps pour les universités est un problème étudié depuis de nombreuses années. Bien que les problèmes semblent similaires d'une université à l'autre, le problème d'une université est unique à cause des contraintes et critères d'évaluation des solutions de chaque institution. De nombreuses approches de résolution ont été utilisées sur ces problèmes variés.

Ce chapitre met en scène la problématique de la planification des horaires dans un contexte général et sa complexité au quotidien dans les organismes. En effet, la question de l'aménagement du temps de travail et de ses enjeux préoccupe toute société ou établissement actif ce qui a incité les chercheurs à proposer des méthodes et des techniques pour aider à gérer au mieux les horaires de travail.

I.2 Les emplois du temps dans le domaine de la pédagogie

La confection d'horaires (ou confection d'emploi du temps) dans les établissements scolaires est un travail très important, difficile à réaliser. Pour fournir une solution, nécessitante d'être capable de s'adapter aux changements dynamiques de l'environnement en tenant compte de la diversité des contraintes telles que l'interdépendance des programmes d'enseignement, la multitude des modules étudiés et les contraintes sur ces modules (cours, TD, TP…), la durée des cours, les contraintes de disponibilité des enseignants, la disponibilité limitée des salles. C'est un problème qui peut être défini comme un problème qui fait assigner quelques évènements dans un nombre limité de périodes. Il peut être divisé en deux catégories principales : la confection d'horaires des cours et la confection d'horaires des examens.
La confection de plannings d'horaires est donc une tâche très difficile et sa solution manuelle peut exiger beaucoup d'effort ce qui a attiré énormément l'attention de la communauté scientifique.

Les problèmes des emplois du temps s'étendent de la construction des emplois du temps semestriels ou annuels dans les universités, écoles ou collèges aux emplois du temps d'examens à la fin de ces périodes. Les premières activités d'emploi du temps ont été effectuées manuellement et un emploi du temps typique, une fois construit est resté statique avec seulement quelques changements nécessaires.

Cependant la nature des enseignements à changé considérablement au cours des années et ainsi les exigences en matière de confection d'emploi du temps sont devenues beaucoup plus compliquées qu'ils ont eu l'habitude de l'être. Par conséquent le besoin de la génération automatisée d'emploi du temps augmente et ainsi le développement d'un système de génération d'emploi du temps qui produit des solutions valables est essentiel. En conséquence, pendant les 30 dernières années, beaucoup d'approches liées à l'automatisation des emplois du temps ont été publiées. De plus, plusieurs applications ont été développés et mises en œuvres avec divers succès.

Dans les dernières décennies, les sujets de résolution du problème d'emploi du temps ont été principalement limités à la (RO) (les techniques employées étaient naturellement mathématiques). Dans la décennie actuelle, la contribution de l'IA a fourni au problème de résolution de l'emploi du temps une heuristique moderne telle que les algorithmes génétiques, le recuit simulé et la recherche tabou.

Par conséquent, les problèmes d'emploi du temps ont attirés l'attention de la communauté scientifique incluant la RO et l'IA pour environ 40 ans et au cours de la dernière décennie, il y a eu un intérêt accru dans ce domaine et plusieurs méthodes ont été décrites dans la littérature comme suit :

I.2.1 Les méthodes séquentielles

Ces méthodes ordonnent les évènements en les assignant séquentiellement dans des périodes de temps valides pour qu'aucun évènement dans une période ne soit en conflit avec un autre dans une période de temps donné. Dans les méthodes séquentielles, les problèmes de confection d'horaires sont généralement représentés par des graphes où les évènements (cours/examens) sont représentés par des nœuds et les conflits entre les évènements sont représentés par les arcs. Par exemple si quelques étudiants doivent suivre deux évènements, il y a un arc entre les nœuds qui représentent le conflit. La construction d'un emploi du temps peut donc être modelée comme un problème de coloration de graphe. Chaque fois la période dans l'emploi du temps correspond à une couleur et les nœuds du graphique sont colorés de telle façon qu'aucun des nœuds adjacents ne soit coloré par la même couleur. Une variété d'heuristiques de coloration de graphe pour la construction des conflits d'emploi du temps est disponible dans la littérature. Ces heuristiques ordonnent les évènements basés sur une évaluation de comment il est difficile de les prévoir. Les heuristiques qui sont souvent employées sont :

- **le plus grand degré d'abord :**

Les évènements qui ont un grand nombre de conflits avec d'autres évènements sont prévus tôt. Le raisonnement est que les évènements avec un grand nombre de conflits sont plus difficiles à prévoir et donc doivent être abordé d'abord.

- **le plus grand degré pondéré :**

C'est une modification du plus grand degré d'abord où le poids de chaque conflit est représenté par le nombre d'étudiants impliqués dans le conflit.

- **le degré de saturation :**

Dans chaque pas de la construction de l'emploi du temps, l'événement qui a un nombre petit de périodes valables disponibles pour la planification dans l'emploi du temps est choisi lointainement.

I.2.2 Les méthodes basées contraintes

Dans ces méthodes, le problème d'emploi du temps est modelé comme un jeu de variables (c à d les évènements), les valeurs (c à d les ressources comme les pièces et les périodes) vont être assignées pour satisfaire un certain nombre de contraintes.
D'habitude quelques règles sont définies pour l'assignation de ressources aux évènements. Quand aucune règle n'est applicable à la solution partielle actuelle un retour arrière est exécutée jusqu'à une solution est trouvée qui satisfait toutes les contraintes.

I.2.3 Les méthodes métaheuristiques

Pendant les deux dernières décennies une variété d'approches métaheuristiques1 comme le recuit simulé, la recherche tabou, les algorithmes génétiques, les colonies de fourmis et les approches hybrides ont été étudiées pour la résolution du problème d'emploi du temps. Les métaheuristiques commencent par une ou plusieurs solutions initiales et emploient les stratégies de recherche qui essayent d'éviter des optimums locaux. Tous ces algorithmes de recherche peuvent produire des solutions de qualité mais ont souvent un coût informatique considérable.

Les algorithmes génétiques sont une classe des algorithmes de recherche stochastiques qui emploient la génétique comme modèle pour la résolution du problème. L'application de la reproduction, la sélection et la mutation peut donner beaucoup d'avantages pour la survie de nouvelles générations. De façon similaire, le recuit simulé a été décrit comme une méthode d'optimisation basée sur une analogie physique. Il a été démontré que le recuit simulé est une bonne technique pour la résolution des problèmes d'optimisation combinatoire dure. La recherche tabou, est une métaheuristique qui guide une procédure de recherche locale pour explorer l'espace de solution au delà de l'optimum local. Les algorithmes de colonies de fourmis s'inspirent du comportement naturel des fourmis où chaque fourmi dépose, le long de son chemin une substance chimique (la phéromone), tous les membres de la colonie perçoivent cette substance et orientent leur marche vers les régions les plus « odorantes », il en résulte la faculté collective de retrouver le plus court chemin. Ces algorithmes sont très indiqués pour les problèmes distribués par nature et les problèmes susceptibles d'évolution dynamique. Les approches hybrides associent souvent une métaheuristique et une méthode locale afin d'affiner la solution. Cette coopération peut prendre la simple forme d'un passage de relais entre la métaheuristique et la technique locale, comme elles peuvent être entremêlées de manière plus complexe.

Parmi les modèles proposés pour confectionner des emplois du temps citons quelques **exemples** : les méthodes évolutives: où l'approche proposée est fondée sur une programmation par contraintes utilisant un algorithme génétique comme moteur d'optimisation. Les méthodes utilisant les colonies de fourmis: où les auteurs de ce projet procèdent à une simplification du problème d'emploi du temps d'université en impliquant trois types de contraintes dures et trois types de contraintes souples. Le système de fourmi « Max-Min Ant System » se sert d'une routine de recherche locale séparée, proposée pour aborder ce problème, une construction de graphe approprié et une représentation de matrice de phéromones sont conçus et les résultats de ce système ont démontré qu'il peut construire des horaires meilleurs qu'un algorithme qui réitère le procédé de recherche locale des solutions.

Les méthodes utilisant la recherche tabou où Schaerf à utilisé cette technique pour résoudre le problème. Il a employé le codage matriciel Mi,j qui contient le nom de la classe du professeur i à la période j. Les voisinages proposés sont :

- Echanger deux cours pour un même professeur.
- Déplacer un cours à une autre période.

Pour des instances de tailles moyennes, il a été démontré que les résultats obtenus ont été encourageants.

1 Une **métaheuristique** est un algorithme d'optimisation visant à résoudre des problèmes d'optimisation difficile (souvent issus des domaines de la recherche opérationnelle, de l'ingénierie ou de l'intelligence artificielle) pour lesquels on ne connaît pas de méthode classique plus efficace.

D'autres méthodes plus spécifiques faisant appel à des modèles de coloration de graphes, de relaxation lagrangienne et de réseaux de neurones.

I.3 Notions fondamentales :

Parmi les objectifs de la théorie de complexité est de classer les problèmes en fonction des ressources (temps de calcul, espace mémoire, etc.…) nécessaires à leur résolution algorithmique. Ceci a montré qu'il existe des problèmes qui ont une solution calculable mais dont toute réalisation effective sur une machine est pratiquement inutilisable parce que le temps de calcul ou la place mémoire nécessaire sont trop importants. L'enjeu est donc de discerner entre les problèmes qui ont une solution réalisables et ceux qui, quelles que soient les améliorations futures des machines, ne peuvent intrinsèquement avoir une telle solution. Quand on prend pour critère le temps d'exécution, on exprime cette frontière en considérant qu'un problème est réalisable si son temps d'exécution est polynomial en fonction de la taille de l'entrée. Si le degré du polynôme est élevé cette notion devient un peu irréalisable, mais la distinction entre temps polynomial et temps non polynomial donne naissance à une classification des problèmes qui est très utile pratiquement.

I.3-1 Notion de complexité :

D'une manière générale, pour résoudre un problème, on est appelé à trouver l'algorithme le plus efficace. Cette notion d'efficacité induit normalement toutes les ressources de calcul nécessaires pour exécuter un algorithme. Or, le temps d'exécution est généralement le facteur dominant pour déterminé si un algorithme est assez efficace pour être utilisé dans la pratique, pour cela on se concentre principalement sur cette ressource.
On appelle complexité en temps d'un algorithme le nombre d'instructions élémentaires mises en œuvre dans cet algorithme afin de résoudre un problème donné.
Une instruction élémentaire sera une affectation, une comparaison, une opération algébrique, la lecture et l'écriture etc.… Mais comme le décompte précis de toutes les instructions d'un programme risque d'être assez pénible, et qu'entre deux exécutions du même algorithme avec un jeu de paramètres différent, le nombre d'instructions exécutées peut changer, on se contentera, en général, d'apprécier un ordre de grandeur de ce nombre d'instructions. C'est ce qu'on désigne sous le nom de complexité de l'algorithme. Donc pour mesurer la complexité temporelle d'un algorithme, on s'intéresse plutôt aux opérations les plus coûteuses :
- Racine carrée, Log, Exp, Addition réelle…
- Comparaisons dans le cas des tris…

I-3-2 Algorithmes polynomiaux et Algorithmes exponentiels :

Un algorithme polynomial est un algorithme dont la complexité est $O(p(n))$ où p est une fonction polynomiale et n dénote la longueur de données. Tout algorithme dont la complexité ne peut être bornée par un tel polynôme d'ordre n, est un algorithme exponentiel (bien que cette définition inclue certaines complexités non polynomiales comme nlogn, qui n'est pas considéré comme fonction exponentielle).

I-3-3 Problèmes combinatoires :

Un problème combinatoire est un problème où il s'agit de trouver la meilleure combinaison possible de solutions. Un tel problème peut être soit un problème de décision, un

problème de recherche ou un problème d'optimisation, selon la question à la quelle on est censé répondre.

I-4-3-1 Problème de décision :

Un problème de décision est un problème où la résolution se limite à la réponse par « oui » ou par « non » à la question de savoir s'il existe une solution au problème. Par conséquent, il n'est pas nécessaire de trouver la solution proprement dite.

I-3-3-2 Problème de recherche :

Dans ce cas précis, la résolution du problème ne s'arrête plus au point de savoir si le problème admet ou non une solution. L'algorithme doit être en mesure de fournir la solution si celle ci existe. Par conséquent, tout problème de décision peut être étendu à problème de recherche.

I-3-3-3 Problème d'optimisation :

Un problème d'optimisation est obtenu à partir d'un problème de recherche en associant à chaque solution une valeur. Il consiste à trouver parmi un ensemble de solutions potentielles celle qui répond le mieux à certains critères décrits sous forme d'une fonction objectif à maximiser ou minimiser c'est à dire on cherchera une solution de valeur optimale, minimale, si on minimise la fonction objective, et maximale, si on la maximise.

I-3-4 La réduction polynomiale:

On dit qu'un problème P1 se réduit polynomialement en un problème P2, s'il existe un algorithme polynomial A construisant, à partir d'une donnée D1 de P1, une donnée D2 de P2 telle que la réponse pour D1 soit oui si et seulement si la réponse pour D2 est oui.

I-3-5 La réduction de Turing:

La réduction polynomiale permet de comparer les problèmes de décision. La réduction de Turing permet de comparer les problèmes de recherche.
On dit qu'un problème de recherche P1 se réduit polynomialement à un problème de recherche P2 par la réduction de Turing s'il existe pour résoudre P1 un algorithme A1 utilisant comme sous–programme un algorithme A2 résolvant P2, de telle sorte que la complexité de A1 est polynomiale, quand on évalue chaque appel de A2 par une constante.

I-3-6 Les différentes classes de complexité :

I-3-6-1 La classe P :

La classe **P** contient tous les problèmes relativement faciles c'est à dire ceux pour lesquels on connaît des algorithmes efficaces. Plus formellement, ce sont les problèmes pour lesquels on

peut construire une machine déterministe (ex : une machine de Turing2) dont le temps d'exécution est de complexité polynomiale (le sigle **P** signifie « **P**olynomial time »).

I-3-6-2 La classe NP :

Les problèmes de la classe **NP** sont ceux pour lesquels on peut construire une machine de Turing non déterministe dont le temps d'exécution est de complexité polynomiale (le sigle **NP** provient de « **N**ondeterministic **P**olynomial time ») (et non de « Non Polynomial).
Contrairement aux machines déterministes qui exécutent une séquence d'instructions bien déterminée, les machines non déterministes ont la remarquable capacité de toujours choisir la meilleure séquence d'instructions qui mène à la bonne réponse lorsque celle-ci existe.
Ce concept abstrait est en fait la base de toute la théorie de la **NP**-complétude.

I-3-6-3 La classe NP-complets :

Parmi l'ensemble des problèmes appartenant à **NP**, il en existe un sous ensemble qui contient les problèmes les plus difficiles : on les appelle les problèmes **NP**complets.
Un problème **NP**-complets possède la priorité que tout problème dans **NP** peut être transformé (réduit) en celui-ci en temps polynomial. C'est à dire qu'un problème est **NP**-complets quand tous les problèmes appartenant à **NP** lui sont réductibles. Si on trouve un algorithme polynomial pour un problème **NP**-complets, on trouve alors automatiquement une résolution polynomiale de tous les problèmes de la classe **NP.**

I-3-6-4 La classe NP-difficiles :

Un problème est **NP**-difficile s'il est plus difficile qu'un problème **NP**-complet, c'est à dire s'il existe un problème **NP**-complet se réduisant à ce problème par la réduction de Turing.
Ceci explique pourquoi, lors de l'étude d'un nouveau problème, on commence par chercher à classer ce problème. Si l'on parvient à montrer qu'il est polynomial, le problème sera résolu. Si par contre, on parvient à montrer qu'il est **NP**-complet, la recherche d'un algorithme exact pour résoudre un tel problème ne sera pas de première priorité, et il sera approprié de se concentrer sur des méthodes heuristiques que la plupart des spécialistes de l'optimisation combinatoire ont orienté leurs recherches pour les développer. Une méthode heuristique est souvent définie comme une procédure exploitant au mieux la structure du problème considéré, dans le but de trouver une solution de qualité raisonnable en un temps de calcul aussi faible que possible.

Remarque :

Le problème de l'emploi du temps s'avèrent être NP difficile et la taille de leurs instances se caractérisent souvent par leur très grande taille.

2 Une machine de Turing est constituée d'un ensemble fini de bandes composées d'un nombre infini de cellules. Chaque cellule, si elle n'est pas vide, contient un symbole représentant une information nécessaire au calcul. Chaque bande ne comporte qu'un nombre fini de cellules non vides. Sur chacune des bandes une tête de lecture-écriture se déplace de cellule en cellule faisant ainsi évoluer les informations contenues dans les bandes et le comportement de la est entièrement décrit par une table finie représentant les actions possibles des têtes de lecture-écriture.

I.4 Description du problème à résoudre

Dans un établissement éducatif, un ensemble d'étudiants groupés sous une structure hiérarchique (spécialité, promotions, sections, groupes,…) sont sensés avoir un ensemble d'enseignements qui se répètent périodiquement, chacun de ces enseignements s'étend sur une durée de temps, dont l'unité élémentaire est la période.

Résoudre le problème de l'emploi du temps revient à affecter à chacun de ces enseignements un nombre de périodes consécutives égal à la durée qu'il exige, un local dont le type et la capacité sont convenables, et un enseignant apte à assurer le module concerné par l'enseignement de façon à prévenir les conflits sur les enseignants, sur les étudiants et sur les locaux..

Le problème de l'emploi du temps se manifeste en plusieurs différentes formes dont chacune des formes est spécifique à l'environnement ou à l'institution qui en a besoin. Dans notre cas, le problème de l'emploi du temps étudié est celui d'université (département d'informatique) où les responsables pédagogiques ont besoin chaque année d'établir une nouvelle planification des différentes promotions en essayant au mieux de satisfaire les contraintes « humaines » des enseignants et des étudiants, les contraintes pédagogiques imposées par la progression des enseignements et en tenant compte des contraintes « physiques » liées aux ressources matérielles (les locaux, les équipements etc.…).

Le département d'informatique regroupe différentes formations qui ont une durée qui varie entre trois ans (Licence) et cinq ans (Master 2).
Le programme pédagogique de chaque formation est connu à priori. Ce programme précise les modules à suivre, leurs volumes horaires et quelques informations pédagogiques (répartition en cours, travaux dirigés, travaux pratiques, etc.…). Selon les besoins pédagogiques et les conditions physiques des ressources, chaque formation est structurée en promotions, en sections, et en groupes, le nombre d'étudiants par groupe en travaux dirigés(TD) est limité à 30 pour préserver un meilleur suivi des étudiants. Les enseignants interviennent selon leur discipline et leur domaine de compétence. Administrativement, les enseignants doivent assurer un nombre minimal d'heures qui est défini dans leur statut. Lorsqu'un un enseignant est chargé d'un enseignement donné, il est tenu d'en respecter le volume horaire prévu par le responsable pédagogique. En cas d'absence, l'enseignant doit prévoir des séances de rattrapage. Il doit donc connaître précisément la disponibilité des ressources de sa séance. Cette organisation garantit que tous les étudiants qui suivent une même formation auront eu le même volt'(ume horaire d'enseignement.

En résumé, les données du problème à résoudre sont constituées par :
1- un ensemble de créneaux horaires étalés sur une semaine de cinq jours, du dimanche au jeudi.
2- Un ensemble de promotions ou groupes d'étudiants.
3- Un ensemble de cours, TD ou TP à programmer dans la semaine.
4- Un ensemble de locaux (salles, amphis et labos).

L'affectation des modules, enseignants, locaux à des périodes est soumise à des contraintes qui diffèrent selon leurs priorités (l'intérêt que recouvre la satisfaction d'une contrainte ou sa violation).

Une contrainte ne revêt pas nécessairement un aspect absolu (soit elle est vérifiée ou violée) mais peut être formulée sous forme d'un objectif qui doit être approché autant que possible, selon ce critère, les contraintes peuvent être réparties en deux grandes classes : les contraintes dures (absolues) et les contraintes de préférences.

I.4.1 Les contraintes dures

Ce type de contraintes doit être obligatoirement satisfait dans toutes les situations car la violation de l'une de ces contraintes rend l'emploi du temps inefficace dans la réalité. On distingue dans notre cas six contraintes dures :

1- un enseignant ne peut pas être affecté à deux séances différentes à la même période.
2- Une salle ne peut pas accueillir deux séances différentes à la même période.
3- Chaque enseignant doit enseigner un module qui entre dans ses compétences.
4- Un module doit respecter le nombre de séances hebdomadaires de ce dernier c'est à dire si un module est enseigné trois fois par semaine, alors il doit apparaître 3 fois dans le chromosome de la promotion.
5- Un emploi du temps doit comporter tous les modules d'une promotion.
6- La charge journalière d'un enseignant ne doit pas être dépassée.

I.4.2 Les contraintes de préférence

Contrairement au type de contraintes précédent, les contraintes de préférences n'exigent pas la vérification stricte, mais d'approcher au maximum de l'objectif voulu. Dans notre cas, on distingue :

1- essayer d'éviter aux (enseignant ou étudiants) des pertes de temps par de trop longs espacements entre deux séances d'une même journée (pas de trous).
2- Eviter que certains jours se trouvent surcharger alors que d'autres le sont moins.
3- Eviter d'affecter une période jugée non convenable à un enseignant, sauf si cela est inévitable
4- Les modules de coefficient minimal ne doivent pas occuper les séances de la matinée d'une journée donnée, au détriment des modules de coefficients élevés.
5- Minimiser les déplacements des étudiants dans l'établissement.
6- Libérer quelques après-midi pour les enseignants Afin de modéliser l'ensemble des contraintes régissant notre problème, on doit d'abord définir les structures de données nécessaires.

I.5 Modélisation du problème des emplois du temps à partir de l'observation des activités relatives à la gestion des emplois du temps

De manière abstraite, le problème des emplois du temps consiste à répartir dans le temps des séances, pendant lesquelles s'effectue une activité pédagogique nécessitant des ressources, en l'occurrence des enseignants, groupes, salles et matériels. Ces séances ont une durée. Le problème des emplois du temps nécessite donc de modéliser les activités, les ressources et le temps.

I.5.1 Modélisation de l'activité pédagogique

L'activité pédagogique est modélisée à l'aide de trois entités : les modules, les enseignements.

Chaque ***module*** est caractérisée par : son nom, sa description pédagogique, son type (cours, travaux dirigés, travaux pratiques, ou autre), une discipline.

Chaque ***enseignement*** est caractérisé par : son nom, ses enseignants, ses groupes, ses matériels par défaut (d'une manière abstraite, ses ressources par défaut), son module, son volume horaire total, sa durée par défaut. De plus pour chaque enseignement est indiqué la liste des enseignements qui doivent avoir été suivis auparavant (pré-requis). D'une manière abstraite, les enseignants, les groupes, les matériels par défaut constituent les ressources par défaut de l'enseignement. Les matériels par défaut sont les matériels qui seront attribués par défaut aux séances de l'enseignement. Les matériels qui sont réellement utilisés dans chaque séance peuvent être différents de ces matériels par défaut. Un enseignement peut être assuré simultanément par plusieurs enseignants. C'est le cas, par exemple de certains TP d'informatique. La durée par défaut est la durée qui sera choisie par défaut pour les séances de l'enseignement. La durée réelle de chaque séance peut être différente de cette durée par défaut.

Les ***modules*** sont des ensembles d'enseignements.

I.5.2 Modélisation des ressources

Les ***ressources*** considérées sont les entités physiques nécessaires à l'élaboration des emplois du temps. Il s'agit des salles, des enseignants, des groupes, des étudiants et des matériels. Afin de prendre en compte la plupart des configurations imaginables, chaque ressource *R* peut être décomposée en un ensemble de sous-ressources désignées par *filles(R)*. De même, *R* peut faire partie de la composition de plusieurs autres ressources. On désigne par *mères(R)* l'ensemble des ressources qui contiennent cette ressource *R*.

Cette organisation hiérarchique permet de considérer les groupes sous différents angles : par exemple, on peut aussi bien parler des étudiants du DEUG-Sciences que des étudiants qui sont inscrits en MIAS, etc. Cette structuration hiérarchisée se retrouve également dans la gestion des matériels. Par exemple une unité de reportage se compose d'une caméra, d'un pied, d'un micro, etc. Lorsque le responsable du matériel doit attribuer le matériel aux séances de TP d'audiovisuel, il lui est plus facile de parler d'unité de reportage plutôt que de faire référence à la liste des matériels qui la compose.

Chaque référence à une ressource R concerne également toutes les filles de R, ainsi que ces "petites filles", etc., jusqu'au niveau le plus bas de la hiérarchie. De plus, dès qu'on place une séance qui concerne R, cela limite les degrés de liberté de toutes ses ressources mères.

Les ressources sont caractérisées par des données abstraites et des données spécifiques. Les données abstraites caractérisent chaque ressource et sont constituées d'un code, qui permet de la différencier des autres ressources, son calendrier qui précise quels sont les jours de disponibilité et d'indisponibilité et sa description. En plus de ces données abstraites, chaque ressource possède des caractéristiques spécifiques qui dépendent du type de la ressource. L'intérêt de distinguer ces deux types de caractéristiques est que l'outil peut très

facilement évoluer pour prendre en compte de nouveaux types de ressources. Dans ce cas, il suffit de développer uniquement les parties spécifiques (que l'on pourrait aussi qualifier de concrètes) de ces nouvelles ressources pour que l'outil les intègre.

Dans la suite nous décrivons les caractéristiques spécifiques des ressources considérées dans l'étude.

➢ Les ressources de type « salle »

Une ***salle*** est un lieu dans lequel sont assurés des enseignements. Chaque salle est caractérisée par : son nom, une capacité qui indique le nombre maximal d'étudiants qu'elle peut recevoir, un type indiquant si la salle est dédiée à des TD ou des TP spécifiques (base de données, système d'exploitation, etc.), si elle permet des cours, etc. Le type d'une salle est une indication sur le type d'enseignement qu'on peut y faire.

➢ Les ressources de type « enseignant »

L'***enseignant*** désigne une personne pouvant assurer des enseignements. Chaque enseignant est caractérisé par : son nom et son prénom, son grade, sa résidence administrative, sa spécialité. La spécialité renseigne sur les modules que peut enseigner un enseignant.

➢ Les ressources de type « groupe »

Un ***groupe*** est un ensemble d'étudiants. Il se compose d'autres groupes qui peuvent être réduits à un seul étudiant. Cette modélisation ne fait pas de distinction entre groupe et étudiant. Ceci permet de traiter de manière homogène les situations dans lesquelles on crée les emplois du temps en fonction des spécialités et les situations dans lesquelles on planifie des enseignements indépendamment des spécialités. Les emplois du temps sont faits pour chaque spécialité sans volonté de partager des enseignements entre plusieurs formations.

Les jours d'indisponibilité des groupes sont scindés en deux ensembles : les jours de congé, les jours d'examen. Par défaut, pendant ces journées, il ne peut pas y avoir de séances d'enseignement.

➢ Les ressources de type « matériel »

Pour qu'un enseignement puisse se dérouler, l'enseignant utilise un matériel pédagogique. Généralement ce matériel est limité à un tableau, des craies, éventuellement un rétroprojecteur et parfois un vidéo projecteur. Pour des TP, l'enseignant a besoin des machines…etc. La disponibilité de ce matériel doit donc être prise en compte pour le bon déroulement de l'enseignement.

En définitive, un matériel se caractérise par : son nom, sa description, sa fréquence de révision et son historique (en terme de maintenance). Ces caractéristiques sont utilisées notamment pour planifier les périodes de révision pendant lesquelles le matériel est indisponible. Cela a donc une conséquence sur les emplois du temps.

I.5.3 Modélisation du temps

Les analyses menées avec les utilisateurs mettent en avant différentes perceptions du temps. Par exemple, lors de la définition des maquettes pédagogiques c'est le temps sous forme de durée qui est considéré : chaque responsable détermine le volume de chaque enseignement. Puis lors de l'élaboration des emplois du temps, le responsable répartit les enseignements dans une semaine "type" et reproduit cette semaine tout au long de l'année.

. La représentation du temps est un problème en soi. Si l'on cherche une représentation la plus générique possible et la plus indépendante possible de l'application, on aboutit à une certaine lourdeur et des performances réduites. Paradoxalement, si l'on souhaite définir un outil suffisamment convivial, celui-ci doit répondre rapidement aux requêtes de son utilisateur d'où la nécessité d'une représentation adaptée qui permette des temps de réponse très courts (de l'ordre de quelques secondes).

Pour représenter le temps, deux notions de base sont nécessaires : l'*instant* et la *durée*. De plus, pour le problème des emplois du temps on rajoute la notion de *fréquence*. Cette notion de fréquence n'est pas indispensable puisqu'elle peut s'exprimer à partir de celles d'instant et de durée. Dans les emplois du temps, on dira qu'un enseignement a lieu tous les 15 jours (*fréquence*) si la durée qui sépare deux séances consécutives de E est d'au moins 15 jours (*durée*). Par exemple, dire « qu'il y a cours de génie logiciel tous les quinze jours » c'est dire que la durée qui sépare deux séances consécutives de ce cours est de 15 jours. Ainsi on distingue la durée de chaque séance de la durée qui sépare les séances.

I.5.3.1 Granularité du temps

La granularité du temps permet de mettre en relation des représentations du temps de différentes finesses. Au niveau le plus bas se trouve la représentation temporelle de base (par exemple le quart d'heure). Aux niveaux plus élevés se trouvent des notions composées du temps (par exemple l'heure, la demi-journée, la journée, la semaine, la quinzaine, le mois, etc.). Il est important de pouvoir appréhender le temps selon différentes granularités, pour pouvoir exprimer facilement des propriétés temporelles.

Dans cette étude, l'objectif n'est pas de déterminer une représentation universelle du temps.

I.5.3.2 Les entités temporelles

Pour modéliser le temps, les entités : ***date***, ***heure***, ***durée***, ***créneau*** et ***calendrier*** sont définies.

Une ***date*** désigne un instant défini par un triplet (jour, mois, année). A partir de ce triplet, on détermine la valeur qui lui est associée sur l'axe des jours. Pour avoir un grain plus fin, la notion d'heure est utilisée. Il s'agit d'un nombre entier compris entre la valeur minimale ***HMin*** et la valeur maximale ***HMax***. Ces deux nombres correspondent à des heures par rapport à une date. Par exemple, dans notre cas, nous avons choisi ***HMin*=8h00** et ***HMax*=17h00** par défaut.

Une ***durée*** est un nombre compris entre ***DMin*** et ***Dmax*** = ***Hmax - HMin***.

DMin représente la plus petite unité temporelle disponible. Nous avons choisi par défaut la valeur ***DMin*** = 1 heure 30 minutes.

Un ***créneau*** horaire désigne un intervalle temporel dans une journée. Ainsi, un créneau est caractérisé par un couple *(H,D)* où *H* représente l'heure de début du créneau et *D* sa durée.

Un ***calendrier*** est un ensemble de dates. Chaque ressource possède son propre calendrier ce qui permet de différencier les disponibilités de toutes les ressources. Par exemple, on peut rendre indisponible une salle du 1er au 15 du mois pour raison de travaux de rénovation ou du 13 au 17 du mois pour cause de conférence. On peut également rendre indisponible tel enseignant en dehors de la période qui s'étend du 1er septembre au 31 décembre par exemple, pour des raisons de détachement. Un calendrier est donc un ensemble de dates auxquelles on associe un état, c'est-à-dire une valeur parmi {**disponible**, **non disponible**}. Pour des raisons pratiques d'autres valeurs ont été utilisées pour rendre l'outil mieux adapté aux pratiques courantes de l'université. Par exemple pour les groupes, ces valeurs sont {*congé*, *examen*}.

I.5.3.3 Les séances et les réservations

Une ***séance*** correspond à une instance temporelle d'un enseignement à une date donnée, pendant un créneau précis. Les caractéristiques d'une séance sont : son enseignement, sa date, son créneau, ses matériels et sa salle. L'ensemble des séances d'une ressource est appelé *planning* de cette ressource. Il s'agit de l'ensemble des séances dans lesquelles apparaît la ressource. Ainsi, il est possible de considérer le planning d'une salle au même titre que celui d'un enseignant. La notion de planning apparaît également au niveau des enseignements : cela permet ainsi de manipuler l'évolution dans le temps d'un enseignement particulier en vue, par exemple, de s'assurer que le volume horaire prévu a bien été réalisé.

Une ***réservation*** correspond à un créneau ajouté au planning d'une ressource. Elle correspond à une option posée sur l'occupation de cette ressource. Par rapport à une séance, une réservation n'est pas associée à un enseignement. Les réservations apportent de la souplesse dans l'utilisation de l'outil notamment lors de la phase de création des emplois du temps par les différents responsables. Par exemple, une salle peut être réservée par un responsable pour un groupe sans savoir précisément quel enseignement y sera fait. Cette réservation est connue des autres responsables qui éviteront ainsi d'occuper cette salle. S'ils le font, le responsable des salles devra envisager une alternative en proposant une autre salle.

I.5.4 Modélisation des spécialités

La notion de spécialité permet de limiter le nombre des ressources à considérer. En effet, le responsable d'une spécialité n'a besoin de connaître que le planning des ressources qu'il utilise. Une spécialité est donc caractérisée par : son nom, son responsable, une liste d'enseignants qui interviennent dans la spécialité, une liste de salles, une liste de groupes, un calendrier de référence, une liste d'enseignements et une liste de modules. Le calendrier de référence est celui qui est partagé par toutes les ressources de la spécialité.

Cependant, certains conflits ne peuvent être détectés que lorsque les différentes spécialités sont rassemblées dans une base de données commune. Il s'agit surtout des conflits d'enseignants et de salles puisqu'il est fréquent qu'un enseignant intervienne dans plusieurs

spécialités ou qu'une salle soit partagée par plusieurs spécialités. Par contre, il est impossible qu'un groupe appartienne à plusieurs spécialités.

I.5.5 Les contraintes sur les données

L'analyse sur le terrain montre que les données gérées doivent vérifier certaines contraintes pour garantir leur cohérence. De manière abstraite, les contraintes à respecter peuvent être classées en deux groupes : les *contraintes physiques* et les *contraintes pédagogiques*.

➢ Les contraintes physiques

Ces contraintes ne doivent pas être violées sinon cela conduirait à des situations conflictuelles. On dira qu'il y a un « conflit physique de ressource » entre deux séances *s1* et *s2* si ces deux séances ont une ressource en commun pendant une durée non nulle.

Voici les contraintes physiques:

- une ressource ne peut pas être occupée en même temps dans deux séances différentes ;
- dès qu'une ressource *R* est occupée par une séance *s*, toutes ses ressources sont également occupées par la même séance et cela récursivement ;
- on ne peut pas mettre plus d'étudiants qu'il n'y a de places dans une salle
- le volume horaire total des séances d'un enseignement ne peut pas dépasser le volume prévu ;
- les calendriers sont respectés pour toutes les ressources.

➢ Les contraintes pédagogiques

Les contraintes pédagogiques se différencient des contraintes physiques par le fait qu'elles peuvent éventuellement être violées : dans ce cas on obtient des emplois du temps de moins bonne qualité d'un point de vue pédagogique. Typiquement ces contraintes sont utilisées pour exprimer ce que doit être un « bon » emploi du temps. Voici quelques exemples de ces contraintes :

- homogénéiser la durée des séances d'un enseignement,
- éviter les « trous » dans l'emploi du temps,
- éviter les cours en fin de journée,
- éviter de placer plusieurs séances d'un même enseignement dans une journée,
- répartir la charge d'enseignement des enseignants et des étudiants, si possible, sur toute la période d'enseignement. L'objectif est d'éviter d'avoir des périodes de surcharges. Pour répartir cette charge, on peut considérer différents niveaux de granularité du temps. Par exemple on peut distinguer la journée, la semaine, le mois, le semestre et l'année.
- accorder plus d'importance à la qualité de l'emploi du temps du plus grand nombre. Par exemple, si l'on doit choisir entre planifier une séance de 2 heures pour un groupe de 10 étudiants ou pour un groupe de 100 étudiants, on préfèrera placer le groupe de 100 étudiants.
- éviter de finir après 17h,
- éviter de placer des cours l'après midi,
- favoriser la régularité dans les emplois du temps.

On peut remarquer que dans l'expression de ces contraintes pédagogiques les termes utilisés ne sont pas précis (éviter, devraient, si possible …). Ces contraintes sont plus difficiles à formaliser que les contraintes physiques et leur traitement est plus délicat.

Il peut y avoir des degrés d'importance entre les contraintes et une même contrainte peut avoir différents « degrés » d'importance. Par exemple, la contrainte « éviter les trous dans l'emploi du temps » représente l'ensemble de toutes les contraintes « éviter les trous d'une durée D dans l'emploi du temps », où D ∈ [30min, 1h, 1h30, …]. Mais il est moins important d'éviter les trous de 30 minutes que d'éviter les trous de 10 heures !

Pour traiter ces contraintes, on les représente par un ensemble de contraintes auxquelles on associe un degré d'importance. Soit C une contrainte représentée par l'ensemble {c(1), c(2), …, c(max)} où dans chaque c(i), i est le degré d'importance accordé à la contrainte c. Si c(i) est vérifiée alors toutes les c(j>i) sont implicitement respectées.

Certaines de ces contraintes sont prises en compte de la manière suivante. Lorsqu'on cherche les séances qu'il est possible de placer sur un créneau précis, ces séances sont classées en fonction de critères liés à la qualité de l'emploi du temps. En tête de cette liste, se trouve la séance qui respecte le mieux les contraintes. Par exemple, si la semaine précédente une séance est déjà placée, on propose une séance du même enseignement et de même durée.

I.6 Conclusion

Le problème de l'emploi du temps est un processus complexe, c'est un problème d'optimisation combinatoire très difficile à résoudre. Car une solution au problème est représentable par un ensemble de propriétés. Le but est d'atteindre la meilleure combinaison de ces propriétés. Une autre cause de complexité du problème est le volume du problème. Aussi l'estimation du degré de satisfaction des contraintes de préférence est souvent difficile à formuler. En plus du fait qu'elles peuvent être parfois contradictoires.

Malgré l'éventail de logiciels qui ont essayé de traiter le problème de l'emploi du temps et la multitude d'approches utilisées, le problème reste toujours posé, car le problème lui même n'est pas standard. Aucune modélisation standard, qui englobe toutes les variantes du problème, n'a été formulée.

Il s'agit donc de développer des outils de planification d'horaires, basés sur des techniques efficaces d'optimisation de ressources qui permettent de construire des programmes de travail, respectant la réglementation du travail et garantissant une bonne couverture de charge tout en limitant les coûts.

Chapitre 2

II.1 Introduction

Dans le but d'une meilleure organisation et une bonne maitrise du travail, tout processus de développement d'applications ou système informatique doit suivre une méthode ou démarche bien définie.

Dans ce chapitre, nous allons entamer le processus par une analyse qui mettra en évidence les différents acteurs intervenant dans le système cible ainsi que leurs besoins. La phase conception, s'appuyant sur les résultats de la phase analyse donnera la modélisation des objectifs à atteindre. Pour ce faire, notre démarche va s'appuyer sur le langage UML, conçu pour la visualisation, la spécification et la construction des systèmes logiciels.

II.2 Définition d'UML

UML (Unified Modeling Language) est un langage unifié pour la modélisation dans le cadre de la conception orienté objet. Il s'agit d'un langage graphique de modélisation objet permettant de spécifier, de construire, de visualiser et de décrire les détails d'un système logiciel.

II.3 Objectif du projet

L'objectif de notre projet est de pourvoir mettre en place une application web pour la gestion des emplois du temps au sein du département informatique, et cela en se basant sur l'affectation semi-automatique.

II.4 Contexte du projet

Notre projet va être développé pour un établissement universitaire, et plus précisément pour le département informatique de l'université Mouloud Mammeri.

II.5 Etude d'opportunité

Le choix de ce type de système a été motivé par le besoin pressant des établissements universitaire, à travers cette solution, nous offrons à l'agent administratif la possibilité de générer des emplois du temps. Donc elle offre beaucoup de chose soit du coté de l'établissement universitaire, du coté professeur et du coté étudiant. On l'appel valeur ajouté de la solution :

Du coté établissement universitaire

• Gain en temps
• Epargniez à l'agent administratif de chercher les données contenu dans des fichiers ou documents
• Facilité la tache de gestion
• Possibilité de consulter les données à n'importe quel moment

Du coté professeur

• Gain en temps
• Possibilité de consulter son emploi du temps
• Possibilité de consulter sa disponibilité et la modifier à n'importe quel moment
• Possibilité de consulter l'emploi du temps de ces collègues

Du coté étudiant :

• Gain en temps
• Possibilité de consulter les emplois du temps à n'importe quel moment

II.6 Etude de faisabilité :

Pour une telle application des moyens matériels (serveurs, base de données, machines …) et humains (agent administratif, enseignant, étudiant) doivent être disponible au sein de l'établissement scolaire.

Coté établissement universitaire

• Accéder à la page web d'accueil de l'application pour :
La gestion des utilisateurs
La gestion des spécialités
La gestion des modules
La gestion des salles
La gestion des groupes
La gestion des semestres
La gestion des sections
La gestion des professeurs
La gestion des créneaux horaire
La gestion des emplois du temps (cours)

Coté professeur

• Accéder à la page web d'accueil de l'application.
Consulter son emploi du temps (cours)
Consulter et modifier sa disponibilité
Consulter l'emploi du temps de ces collègues

Coté étudiant

• Accéder à la page web d'accueil de l'application.
Consulter l'emploi du temps des cours

II.7 Analyse

Cette partie à pour objectif la spécification de manière claire de l'application. Pour ce faire, il est nécessaire de déterminer globalement ce qui se trouve dans le champ de l'application. De ce fait, on s'intéressera dans cette phase à l'identification des acteurs du système, leurs espaces et le contexte de l'application.

II.7.1 Identification des acteurs de l'application

Pendant l'étude qu'on a effectuée, nous avons procédé à l'identification des principaux acteurs qui seront les futurs utilisateurs de l'application, ces acteurs sont :

- **Administrateur**
  - ✓ Gestion des utilisateurs
  - ✓ Gestion des spécialités
  - ✓ Gestion des modules
  - ✓ Gestion des salles
  - ✓ Gestion des groupes
  - ✓ Gestion des semestres
  - ✓ Gestion des créneaux horaire
  - ✓ Gestion des professeurs
    - • Professeur
    - • Disponibilité
    - • Module enseigné
  - ✓ Gestion des emplois du temps (cours)

- **Collaborateur**
  - ✓ Gestion des spécialités
  - ✓ Gestion des modules
  - ✓ Gestion des salles
  - ✓ Gestion des groupes
  - ✓ Gestion des semestres
  - ✓ Gestion des créneaux horaire
  - ✓ Gestion des professeurs
    - • Professeur
    - • Disponibilité

• Module enseigné
- ✓ Gestion des emplois du temps

Professeur

- ✓ Consulter son emploi du temps
- ✓ Consulter et modifier sa disponibilité
- ✓ Consulter emploi du temps collègue
- ✓ Modifier mot de passe

Etudiant

- ✓ Consulter l'emploi du temps
- ✓ Modifier mot de passe

II.7.2 Identification des espaces

A chaque acteur est attribué un espace qui regroupe toutes les tâches qu'il peut effectuer. Pour notre cas nous avons identifié quatre espaces :

- ➢ Espace administrateur
- ➢ Espace collaborateur
- ➢ Espace professeur
- ➢ Espace étudiant

Remarque : les activités de chaque acteur dans son espace seront définies de manières détaillées dans les diagrammes de cas d'utilisation.

II.8 Conception

C'est la phase la plus complexe du projet. Elle vise principalement à préciser le modèle de telle sorte qu'il puisse être implémenté avec les composantes de l'architecture, pour ce faire nous avons adopté une démarche pour une bonne conception.

II.8.1 Diagramme de contexte

a_ conception globale :

Le diagramme de contexte est un modèle conceptuel de flux qui permet d'avoir une vision globale des interactions entre le système et les liens avec l'environnement extérieur. Il permet aussi de bien délimiter le champ de l'étude. Pour notre cas le contexte est donné par la figure suivante :

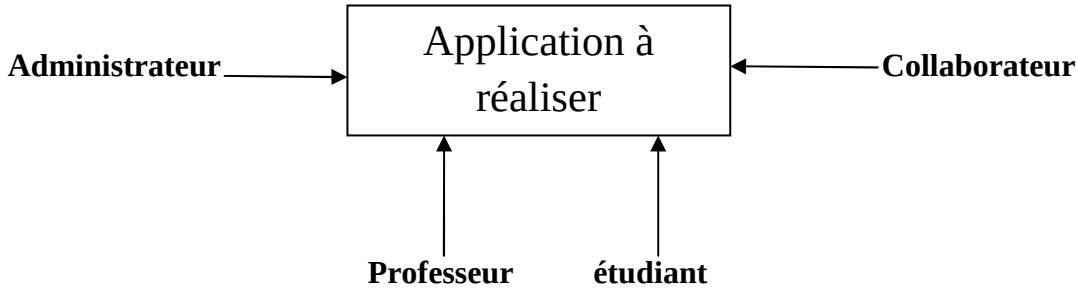

Figure II.1 : Diagramme de contexte (conception globale)

b_ conception détaillé :

La conception détaillée permet d'avoir une vision précise du fonctionnement de l'application.

Et nous allons le montré par le diagramme suivant :

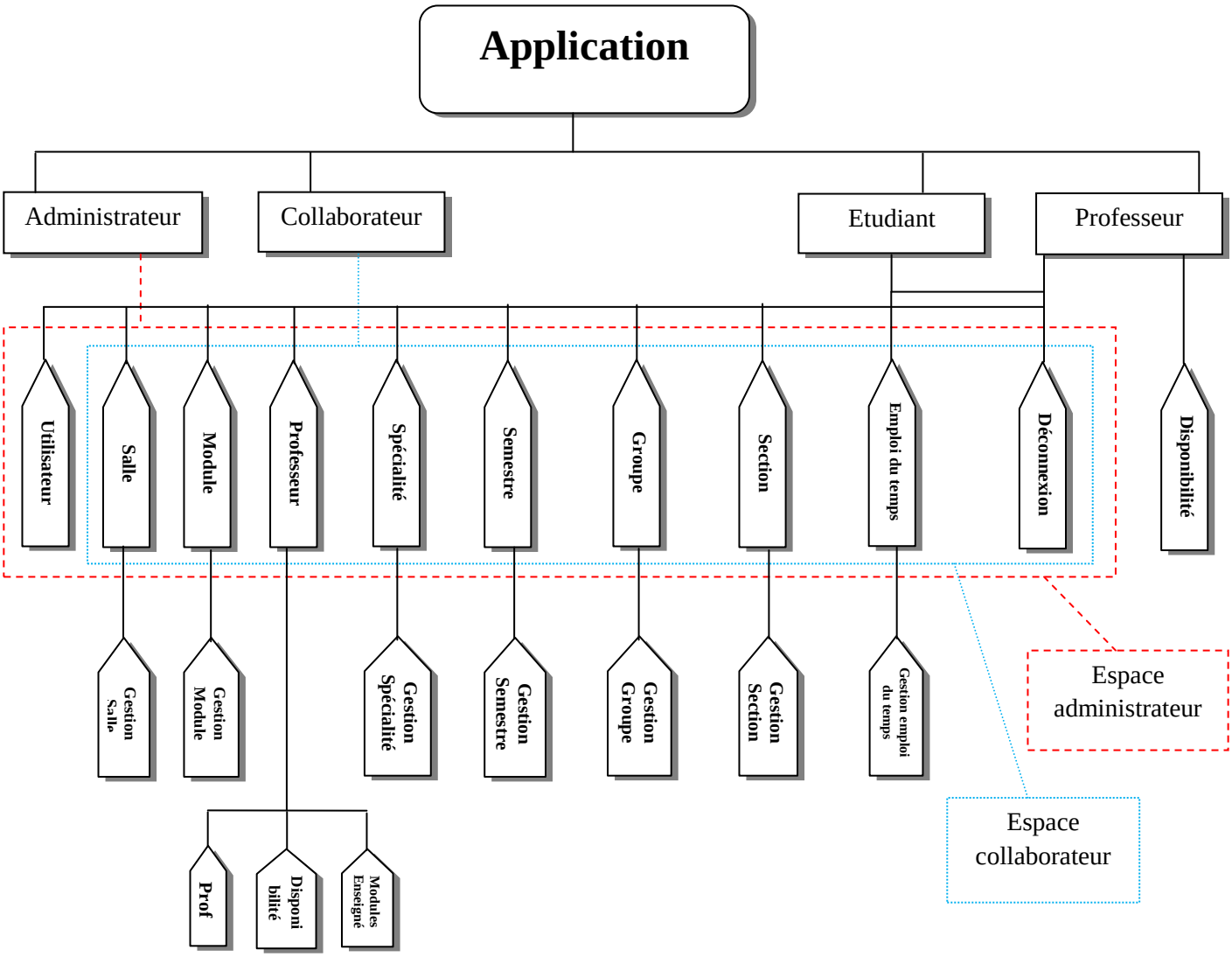

Figure II.2 : Diagramme de conception détaillée

II.8.2 La démarche de conception de l'application

Le processus de conception de notre projet se caractérise par deux niveaux : le niveau applicatif et le niveau donné.

• Le niveau applicatif :

S'appuie essentiellement sur quelques diagrammes du langage de modélisation UML. Donc, après avoir identifié les principaux acteurs ainsi que leurs besoins, à travers notre étude, chose

qui nous a permis d'identifier les différentes fonctionnalités du système à concevoir, nous avons opté pour la démarche suivante :

- Mettre en évidence les cas d'utilisation mis en œuvre par les utilisateurs futurs du système. Les diagrammes de cas d'utilisation détaillés sont élaborés.
- A l'aide du diagramme de séquence et d'activité, on formalise graphiquement les scénarios qui décrivent chaque cas d'utilisation.
- Les classes sont définies par synthèse des diagrammes de séquence et d'activité. Une fois les classes manipulées sont identifiées, on passe à l'élaboration du diagramme de classe.

• Le niveau données

Ce niveau concerne l'organisation conceptuelle, logique et physique des données manipulées. Durant la partie analyse nous avons pu identifier les données nécessaires et indispensables au bon fonctionnement de l'application et à travers la conception du niveau applicatif nous allons dégager les classes significatives. Dés lors on pourra élaborer la conception de la base de données. La démarche que nous avons adoptée pour la conception de l'application s'appuie sur cinq éléments :

1 : identification des acteurs et des besoins

2 : identification et représentation des cas d'utilisation

3 : élaboration des diagrammes de séquences

4 : élaboration des diagrammes d'activités

5 : élaboration du diagramme de classe

La figure ci-dessous donne la représentation graphique de la démarche de modélisation choisie pour concevoir notre application :

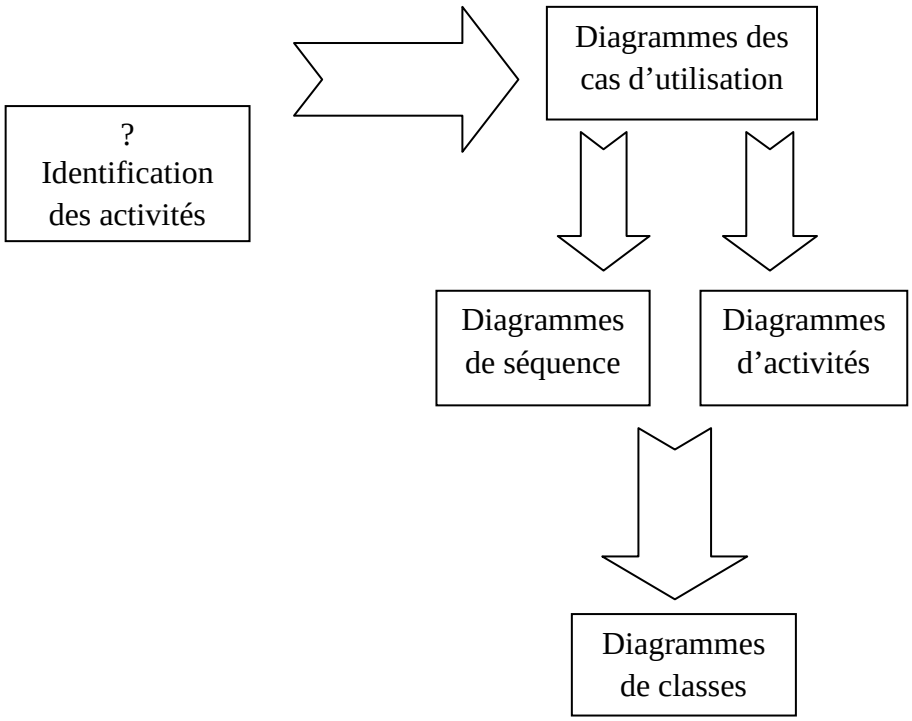

Figure II.3 : Cycle de modélisation de l'application.

II.8.3 Le niveau applicatif

II.8.3.1 Les cas d'utilisation

Un cas d'utilisation représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produit un résultat observable intéressant pour un acteur particulier. Il permet de décrire ce que le système devra faire, sans spécifier comment le faire. Dans notre cas nous distinguons les cas d'utilisation suivant :

II.8.3.1.1 Cas d'utilisation relatif à l'administrateur : nous avons recensés les suivants :

- Gestion des utilisateurs :

•Ajouter utilisateur
• Modifier utilisateur
• Supprimer utilisateur
• Consulter utilisateur

- Gestion des spécialités :

• Ajouter spécialité
• Modifier spécialité
• Supprimer spécialité
• Consulter spécialité

- Gestion des modules :

• Ajouter module
• Modifier module
• Supprimer module
• Consulter module

- Gestion des salles :

• Ajouter salle
• Modifier salle
• Supprimer salle
• Consulter salle

- Gestion des professeurs :
  - professeurs

• Ajouter professeur
• Modifier professeur
• Supprimer professeur
• Consulter professeur

  - disponibilité

• Ajouter disponibilité
• Modifier disponibilité
• Consulter disponibilité

  - modules enseignés

• Ajouter module
• Supprimer module
• Consulter module

- Gestion des groupes :

• Ajouter groupe
• Modifier groupe
• Supprimer groupe
• Consulter groupe

- Gestion des semestres :

• Ajouter semestre
• Modifier semestre
• Supprimer semestre
• Consulter semestre

- Gestion des créneaux horaire

• Ajouter le créneau horaire
• Modifier le créneau horaire

- Gestion des emplois du temps:

• Créer emploi du temps
• Modifier emploi du temps
• Supprimer emploi du temps
• Consulter emploi du temps

II.8.3.1.2 Cas d'utilisation relatif au collaborateur : nous avons recensés les suivants :

- Gestion des spécialités :

• Ajouter spécialité
• Modifier spécialité
• Supprimer spécialité
• Consulter spécialité

- Gestion des modules :

• Ajouter module
• Modifier module
• Supprimer module
• Consulter module

- Gestion des salles :

• Ajouter salle
• Modifier salle
• Supprimer salle
• Consulter salle

- Gestion des professeurs :
  - professeurs

• Ajouter professeur
• Modifier professeur
• Supprimer professeur
• Consulter professeur

  - disponibilité

• Ajouter disponibilité
• Modifier disponibilité
• Consulter disponibilité

- modules enseignés

• Ajouter module
• Supprimer module
• Consulter module

- Gestion des créneaux horaire

• Ajouter le créneau horaire
• Modifier le créneau horaire

- Gestion des groupes :

• Ajouter groupe
• Modifier groupe
• Supprimer groupe
• Consulter groupe

- Gestion des semestres :

• Ajouter semestre
• Modifier semestre
• Supprimer semestre
• Consulter semestre

- Gestion des emplois du temps:

• Créer emploi du temps
• Modifier emploi du temps
• Supprimer emploi du temps
• Consulter emploi du temps

II.8.3.1.3 Cas d'utilisation relatif au professeur : nous avons recensés les suivants :

- S'authentifie
- Consulter son emploi du temps
- Consulter sa disponibilité
- Modifier sa disponibilité
- Modifier mot de passe

II.8.3.1.4 Cas d'utilisation relatif à l'étudiant : nous avons recensés les suivants :

- S'authentifie
- Consulter l'emploi du temps
- Modifier mot de passe

II.9 Diagrammes des cas d'utilisation

Le diagramme de cas d'utilisation montre les interactions fonctionnelles entre les acteurs et le système à l'étude.

II.9.1 Diagrammes des cas d'utilisation détaillés :

Les cas d'utilisation décris ci-dessus sont représenté dans les diagrammes détaillés suivant:

- **Etudiant**

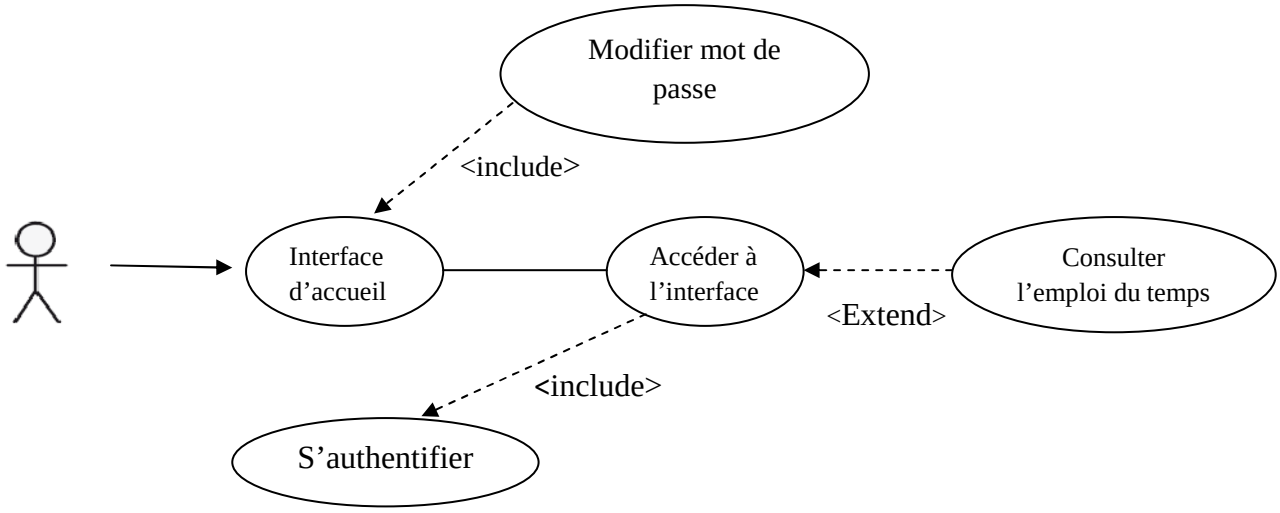

Figure II.3 : Diagramme de cas d'utilisation détaillés relatif à l'étudiant.

- **Professeur**

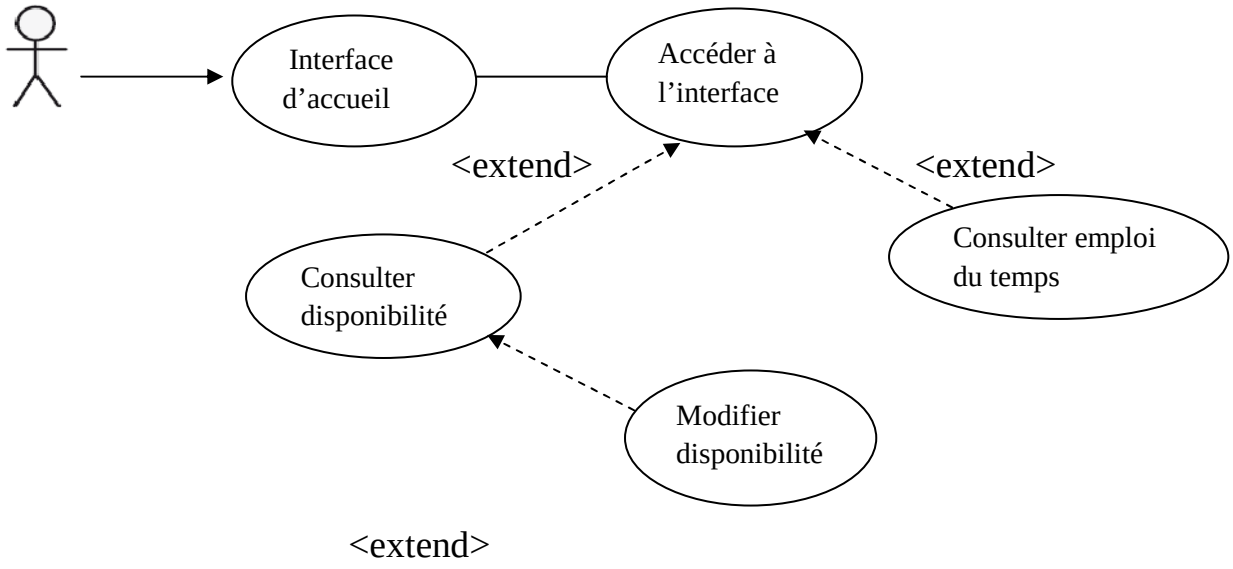

Figure II.4 : Diagramme de cas d'utilisation détaillés relatif au professeur

-
-
-

- **Administrateur**

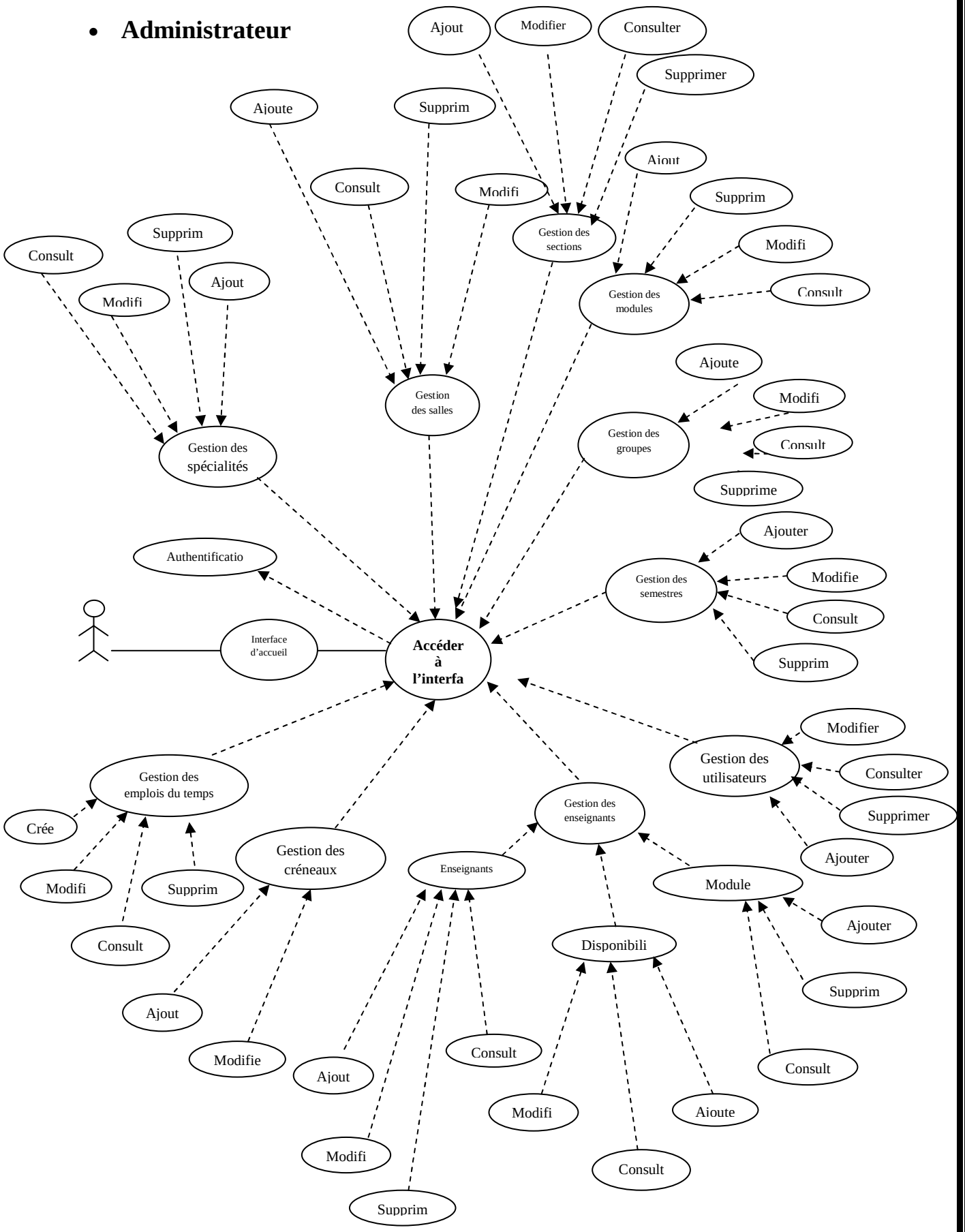

Figure II.5 : Diagramme de cas d'utilisation détaillés relatif à l'administrateur

- **Collaborateur**

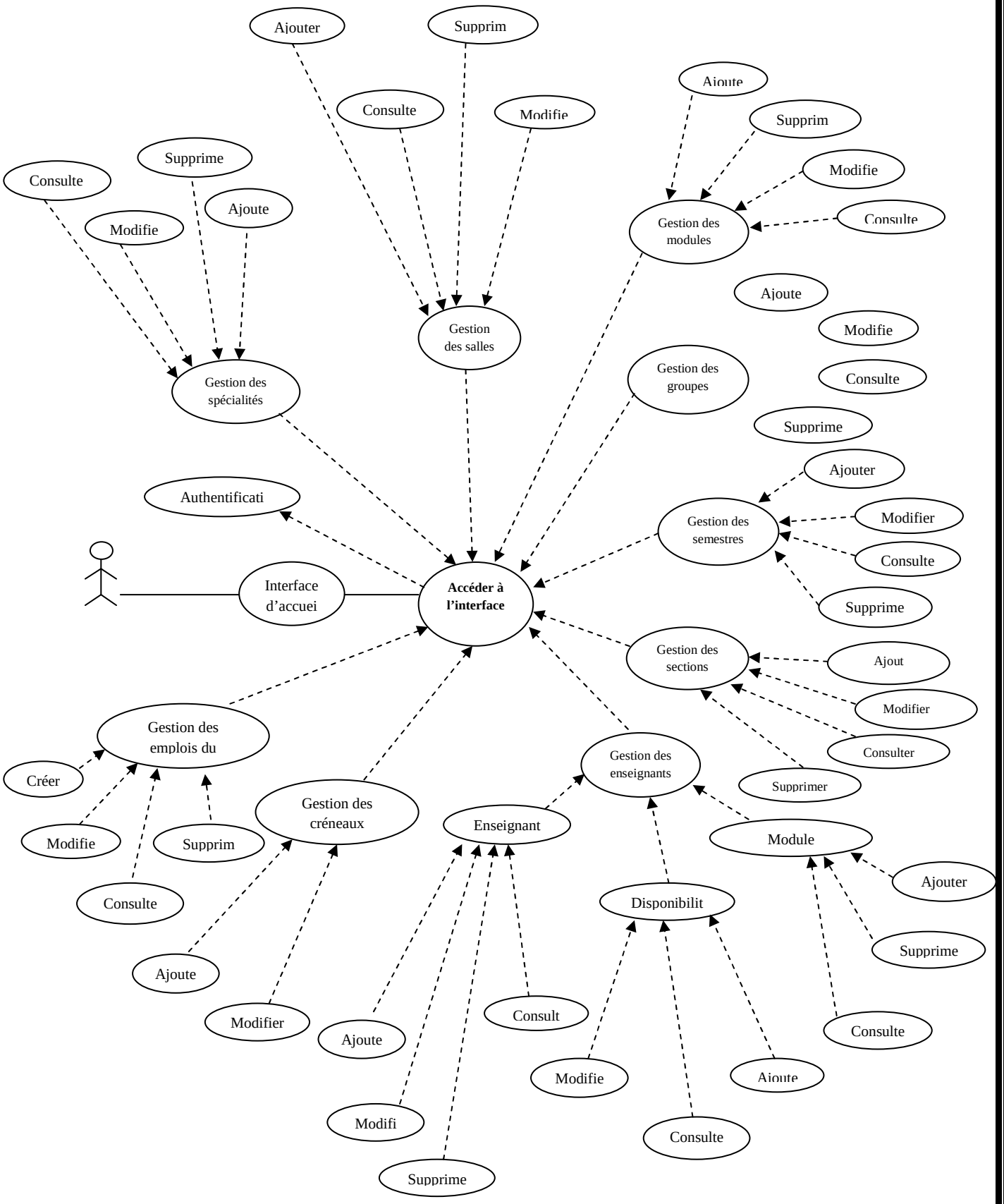

Figure II.6 : Diagramme de cas d'utilisation détaillés relatif au collaborateur

II.9.2 Diagramme de séquence

Le diagramme de séquence représente des échanges de messages entre éléments, dans le cadre d'un fonctionnement particulier du système. Vu le nombre important de cas d'utilisation qu'on a recensé, nous allons décrire que quatre exemples de cas d'utilisation :

- Ajouter Spécialité
- Consulter emploi du temps (administrateur / collaborateur)
- Créer emploi du temps
- Supprimer utilisateur

II.9.2.1 Ajouter spécialité

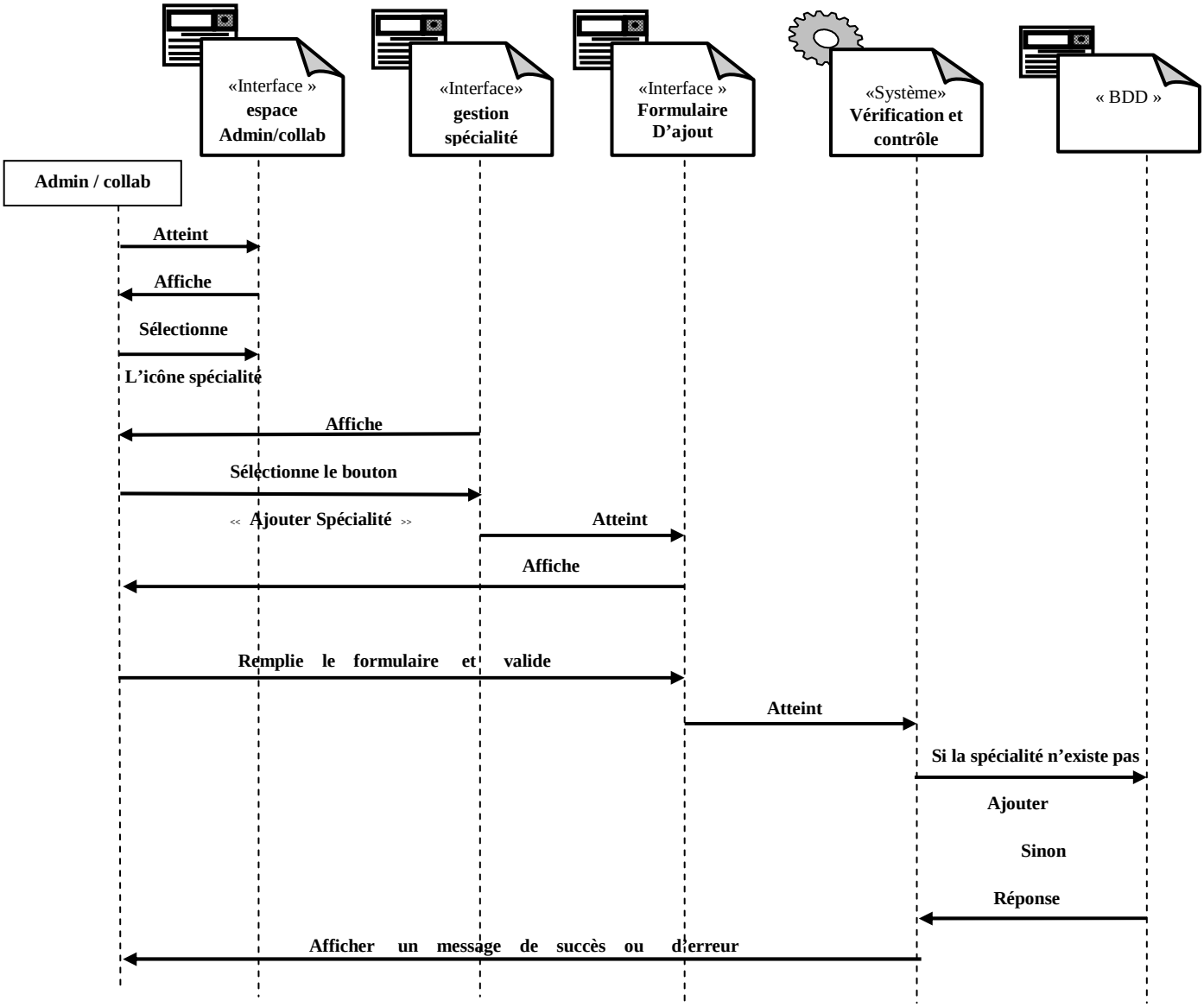

Figure II.7 : Diagramme de séquence pour « ajouter spécialité ».

1. L'admin/collab atteint son espace.
2. L'admin/collab sélectionne l'icone formation
3. L'espace gestion des formations est affiché.
4. L'admin/collab sélectionne le bouton « ajouter formation ».
5. L'admin/collab atteint le formulaire d'ajout.
6. L'admin/collab remplie le formulaire et valide.
7. Le système vérifie la saisie et exécute la requête.
8. Si la formation n'existe pas, elle sera ajouter et le système lui affiche un message de sucées.
9. Sinon il lui affiche un message d'erreur.

II.9.2.2 Consulter emploi du temps

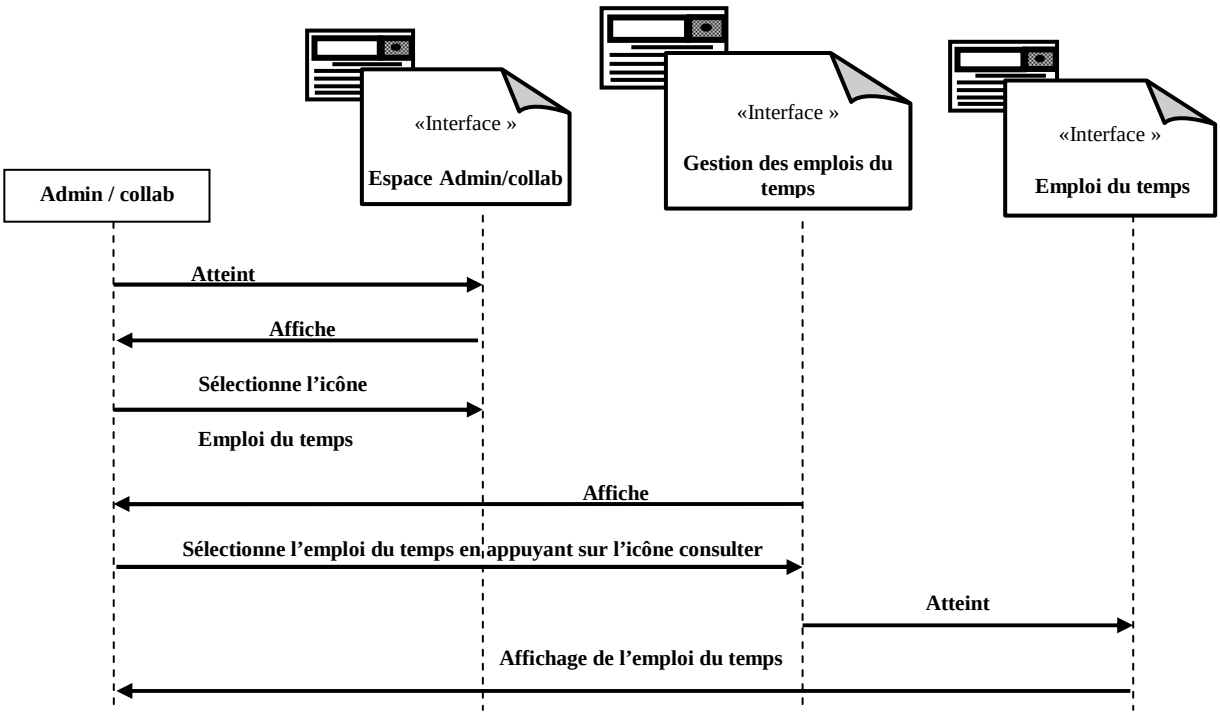

Figure II.8 Diagramme de séquence pour « consulter emploi du temps (admin / collab)».

1. L'admin / collab atteint son espace.
2. Son espace est affiché
3. L'admin / collab sélectionne l'icone emploi du temps.
4. L'admin / collab atteint l'interface gestion des l'emploi du temps.
5. L'admin / collab Sélectionne l'emploi du temps en appuyons sur l'icone consulter emploi du temps
6. L'admin / collab atteint l'interface emploi du temps
7. L'emploi du temps est affiché.

II.9.2.3 Créer emploi du temps

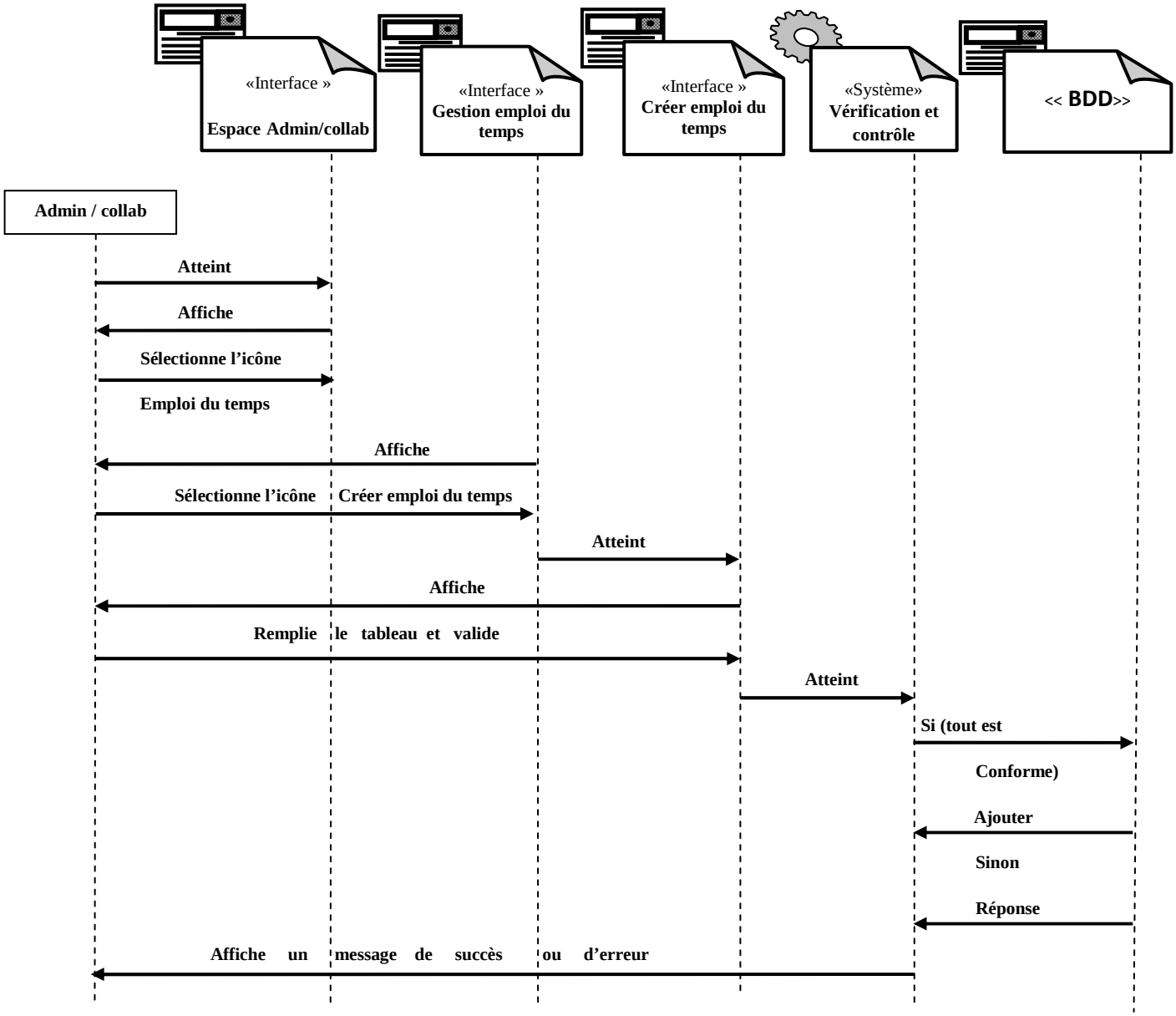

Figure II.9 : Diagramme de séquence pour « créer emploi du temps ».

1. L'admin/collab atteint son espace.
2. L'admin/collab sélectionne l'icone emploi du temps
3. L'espace gestion des emplois du temps est affiché.
4. L'admin/collab sélectionne l'icone créé emploi du temps.
5. L'admin/collab atteint l'interface d'ajout.
6. L'admin/collab remplie le tableau et valide.
7. Le système vérifie et exécute la requête.
8. Si tout est conforme, il sera ajouter et le système lui affiche un message de sucées.
9. Sinon il lui affiche un message d'erreur.

II.9.2.4 Supprimer utilisateur

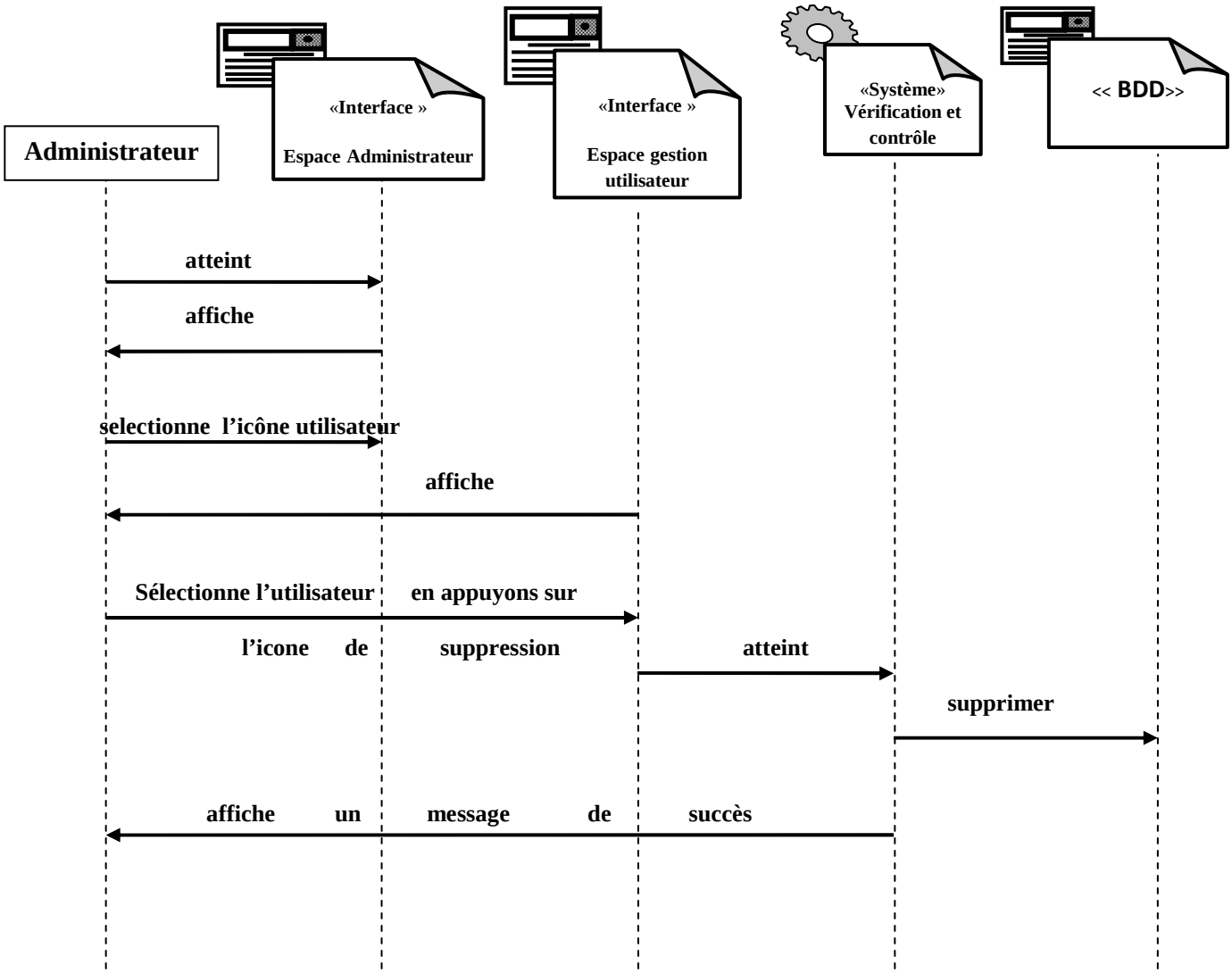

Figure II.10 : Diagramme de séquence pour « supprimer utilisateur».

1. L'administrateur atteint son espace.
2. L'administrateur sélectionne l'icone utilisateur.
3. L'espace gestion des utilisateurs est affiché à l'administrateur.
4. L'administrateur sélectionne l'utilisateur à supprimer en appuyons sur l'icone de suppression
5. Le système supprime l'utilisateur de la base de données système

II.9.3 Diagramme d'activités

Le diagramme d'activité représente les règles d'enchaînement des actions et décisions au sein d'une activité. Dans notre cas on va présenter le diagramme d'activité pour le cas de la création d'un emploi du temps :

II.9.3.1 Créer emploi du temps :

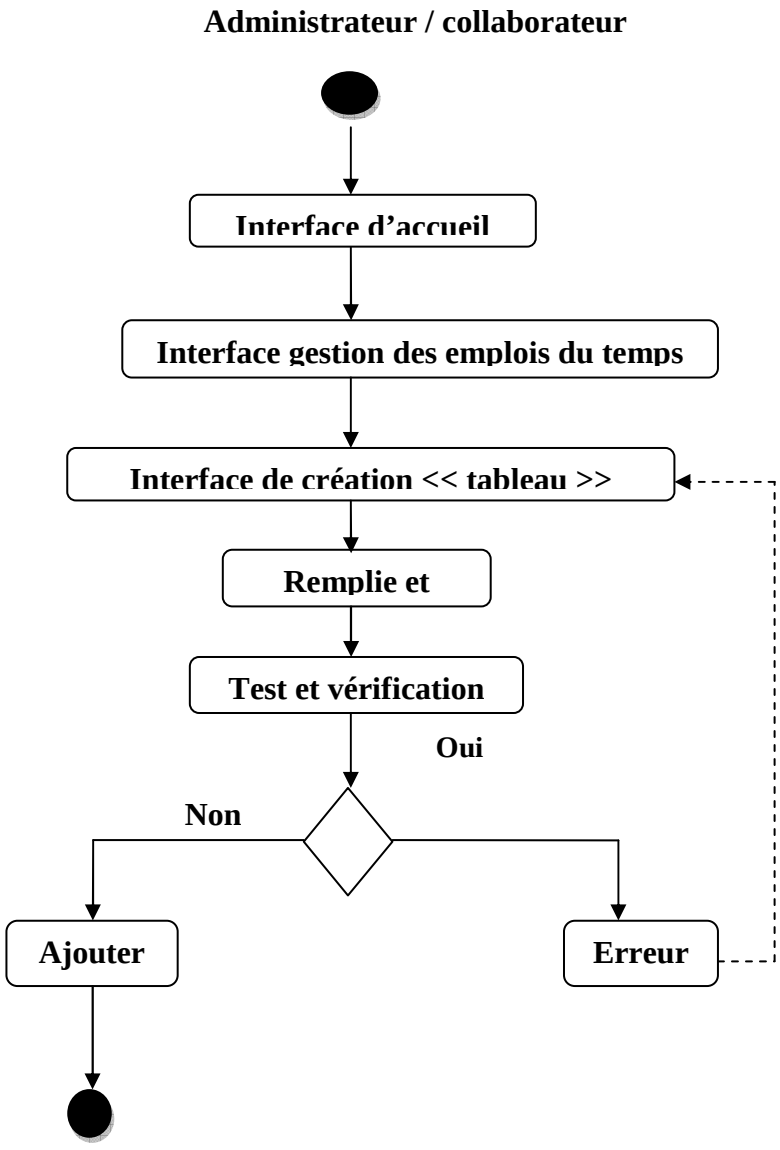

Figure II.11 : Diagramme d'activité de cas d'utilisation « Créer emploi du temps »

II.9.3.2 Ajouter spécialité

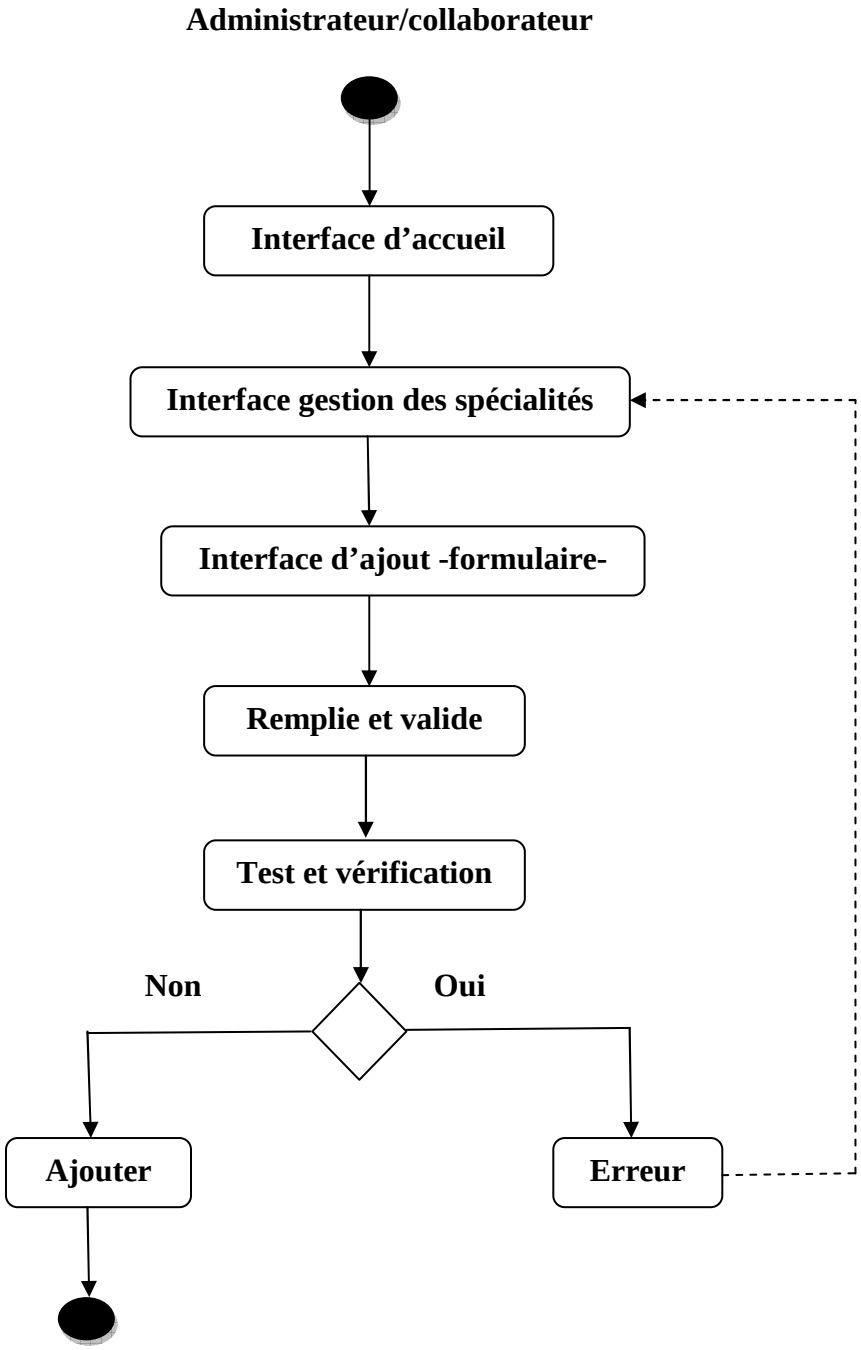

Figure II.12: Diagramme d'activité de cas d'utilisation « Ajouter spécialité »

II.9.4 Diagramme de classes

Le diagramme de classes est le plus important dans la modélisation orientée objet. Il représente un ensemble de classes, d'interface et de collaboration ainsi que leurs relations, il a pour objet de décrire la structure des entités manipulées par les utilisateurs. Dans notre cas on va représenter le diagramme de classes pour quelque cas d'utilisation :

II.9.4.1.Créer emploi du temps

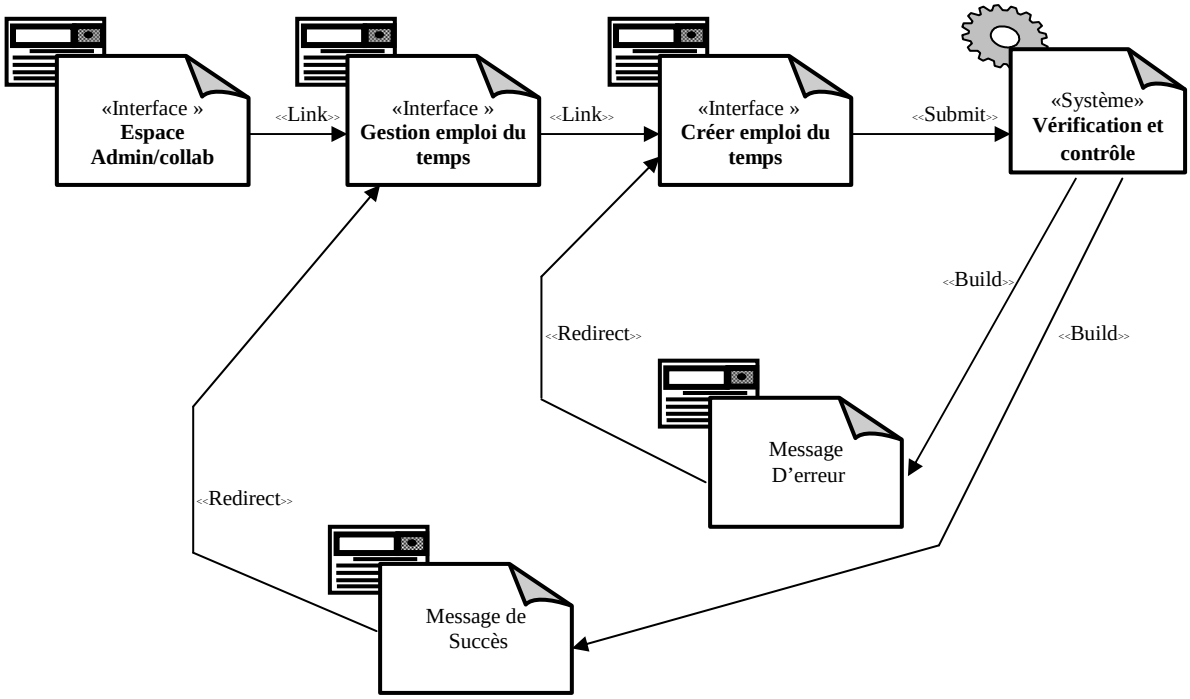

Figure II.13 : Diagramme de classes du cas d'utilisation « créer emploi du temps ».

II.9.4.2 Ajouter spécialité

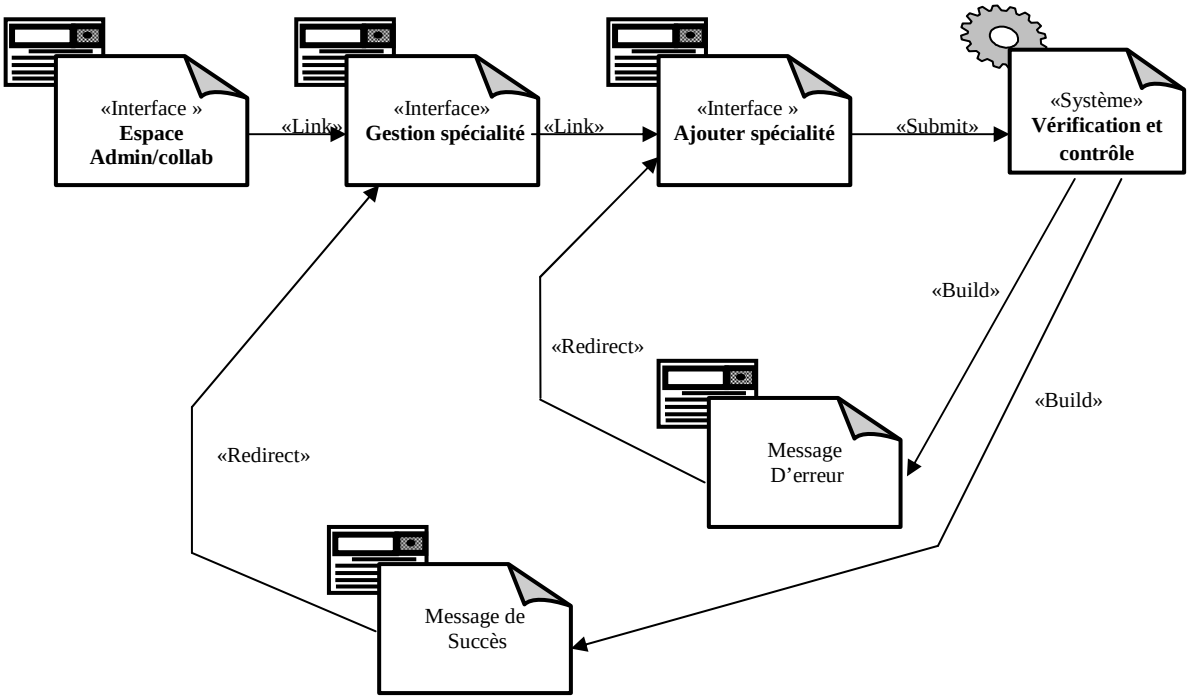

Figure II.14 : Diagramme de classes du cas d'utilisation « Ajouter spécialité ».

II.9.4.3 Consulter emploi du temps

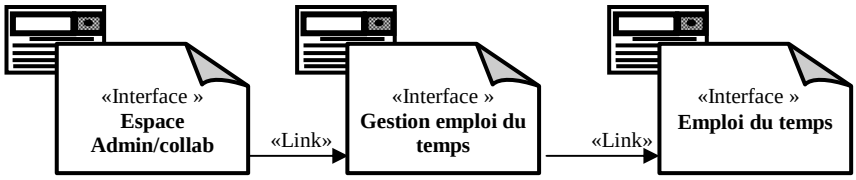

Figure II.15 : Diagramme de classes du cas d'utilisation « supprimer utilisateur»

II.9.5 Le niveau données

Dans ce niveau, le travail consiste en premier lieu à définir un modèle conceptuel de manière à concevoir la structure de la base de données, en se basant sur modèle conceptuelle de données (MCD).

II.9.5.1 Modèle conceptuelle de données (MCD)

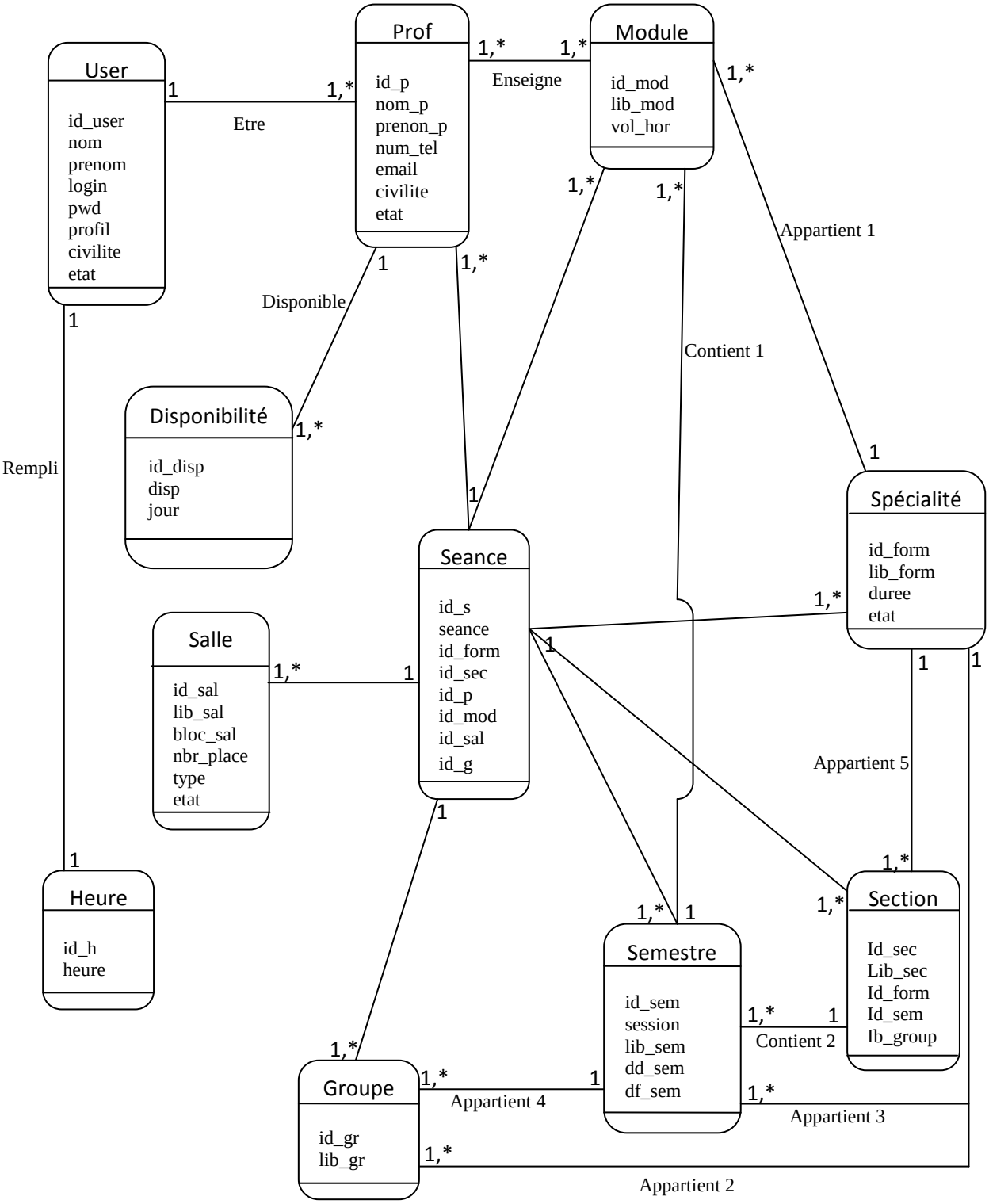

Figure II.16 : Modèle conceptuelle de données (MCD)

II.9.5.2 Le modèle physique de données (MPD)

Après l'élaboration du modèle conceptuelle de données (MCD) la conception de la base de données est simple, chaque classe du diagramme représente une table et les colonnes de la table sont les attributs de la classe. Le modèle physique de données (MPD) nous donne la représentation physique de l'ensemble des tables de la base de données du système étudié :

Utilisateur

Champ	Désignation	Type	Taille	Observation
id_user	Identifiant de l'utilisateur	N		Auto incrémente
Nom	Nom de l'utilisateur	A	30	
Prenom	Prenom de l'utilisateur	A	30	
Login	Login de l'utilisateur	AN	15	
Pwd	Passord de l'utilisateur	AN	15	
Profil	Profil de l'utilisateur	A	15	
Civilité	Civilité de l'utilisateur	A	5	
Etat	Etat de l'utilisateur	A	10	Compte Activé/désactivé
Id_p*	Identificateur du proffesseur	N	10	

Tableau II.1: Structure de la table utilisateur.

Professeur :

Champ	Désignation	Type	Taille	Observation
id_p	Identifiant du professeur	N		Auto incrémente
Nom	Nom du professeur	A	30	
Prenom	Prenom du professeur	A	30	
Num_tel	Numero de telephone du professeur	N	10	
Email	Email du professeur	AN	40	
Civilité	Civilité de l'utilisateur	A	5	
Etat	Etat du professeur	A	10	Compte Activé/Désactivé

Tableau II.2: Structure de la table professeur.

Groupe

Champ	Désignation	Type	Taille	Observation
id_gr	Identifiant du groupe	N		Auto incrémente
lib_gr	Libelle du groupe	AN	15	
id_spec*	Identifiant de la spécialité	AN	5	
id_sem*	Identifiant du semestre	N	5	

Tableau II.3: Structure de la table groupe.

Spécialité

Champ	Désignation	Type	Taille	Observation
id_spec	Identifiant de la spécialité	N		Auto incrémente
lib_spec	Libelle de la spécialité	AN	40	
Duree	Durée de la spécialité	N	4	
Etat	Etat de la spécialité	A	10	Activé/Désactivé

Tableau II.4: Structure de la table spécialité.

Module

Champ	Désignation	Type	Taille	Observation
id_mod	Identifiant du module	N		Auto incrémente
lib_mod	Libelle du module	AN	30	
vol_hor	Volume horaire du module	N	30	
id_spec*	Identifiant de la spécialité	N	10	

Tableau II.5: Structure de la table module.

Salle

Champ	Désignation	Type	Taille	Observation
id_sal	Identifiant de la salle	N		Auto incrémente
lib_sal	Libelle de la salle	AN	15	
bloc_sal	Bloc de la salle	AN	5	
nbr_place	Nombre de places de la salle	N	3	
Type	Type de la salle	A	10	Cours/Machine
Etat	Etat de la salle	A	15	Utilisable/Inutilisable

Tableau II.6: Structure de la table salle.

Semestre

Champ	Désignation	Type	Taille	Observation
id_sem	Identifiant du semestre	N		Auto incrémente
Session	Session du semestre	N	15	
lib_sem	Libelle du semestre	AN	15	
dd_sem	Date début du semestre	Date		
df_sem	Date fin du semestre	Date		
id_spec*	Identifiant de la spécialité	N	10	

Tableau II.7: Structure de la table semestre.

Disponibilité

Champ	Désignation	Type	Taille	Observation
id_disp	Identifiant da la disponibilité	N		Auto incrémente
Disp	Disponibilité	AN	4	
Jour	Jour	A	10	
id_p*	Identifiant du professeur	N	5	

Tableau II.8: Structure de la table disponibilité.

Enseigne

Champ	Désignation	Type	Taille	Observation
id_p	Identifiant du professeur	N	5	
id_mod	Identifiant du module	N	5	

Tableau II.9: Structure de la table enseigne.

Séance

Champ	Désignation	Type	Taille	Observation
id_spec	Identifiant de la spécialité	N	5	
id_sem	Identifiant du semestre	N	5	
id_gr	Identifiant du groupe	N	5	
id_p	Identifiant du professeur	N	5	
id_mod	Identifiant du module	N	5	
id_sal	Identifiant da la salle	N	5	
Seance	Seance	AN	5	

Tableau II.10: Structure de la table séance.

Heure

Champ	Désignation	Type	Taille	Observation
id_h	Identifiant du créneau horaire	N		Auto incrémente
Heure	Créneau horaire	AN	10	

Tableau II.11: Structure de la table heure.

Section

Champ	Désignation	Type	Taille	Observation
id_sec	Identifiant de la section	N		Auto incrémente
lib_sec	Libelle de la section	AN	15	
id_spec*	Identifiant de la spécialité	AN	5	
id_sem*	Identifiant du semestre	N	5	
id_gr*	Identifiant du groupe	N	5	

Tableau II.12: Structure de la table section.

Remarque :

• La table heure est remplie une seule fois par l'administrateur/collaborateur avec les créneaux horaires voulus.

• Codification :

- A : Alphabétique
- AN : Alphanumérique
- * : clé étrangère
- ——— : clé primaire

II.10 Conclusion

Afin de modéliser notre application, nous avons introduit l'un des langages de modélisation « UML », dans le but de spécifier les cas d'utilisation et concevoir les diagrammes de séquence puis élaborer les diagrammes de classes. Ensuite, nous avons élaboré le modèle conceptuel de données. Enfin, on a cité et défini toutes les tables utilisées dans notre base de données.

Le chapitre qui suit sera consacré à l'implémentation et la réalisation de notre application.

Chapitre 3

III.1 Introduction

Pour tout développement d'application, il est nécessaire de choisir les technologies et outils adéquats pour faciliter la réalisation. Dans ce chapitre nous allons présenter les technologies et différents outils utilisés, puis nous passerons à l'architecture du système et les outils nécessaires pour le déploiement de l'application, enfin nous allons expliquer ses fonctionnalités en présentant quelques interfaces illustratives.

III.2. Performance du système :

La machine qui a servi au développement de notre application est un ordinateur portable muni de la configuration suivante :

* Microsoft Windows 7 Edition integrale, Version 2009, Service Pack 1.
* Intel(R) Core(TM) 2 Duo CPU.
* T6670 @ 2.20 GHz.
*2.20 GHz, 3 Go de RAM.

Sur laquelle les logiciels suivant ont été installé :

• JDK-6u18-Windows-i586

• Apache / Tomcat 6.0.20

• MySQL Server 5.4

• Mozilla Firefox

III.3 Technologie et outils

III.3.1 Technologies

III.3.1.1 Java

Java est un langage de programmation informatique orienté objet pour des applications monotones, des applications client/serveur. Créé par James Gosling et Patrick Naughton employés de Sun Microsystems avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au *SunWorld* . Ce dernier a la particularité principale que les logiciels écrits avec ce dernier sont très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Microsoft Windows, Mac OS ou GNU/Linux.

Java permet de développer des applications client-serveur. Côté client, les applets sont à l'origine de la notoriété du langage. C'est surtout côté serveur que Java s'est imposé dans le milieu de l'entreprise grâce aux servlets, et plus récemment les JSP (Java Server page) :

- **Les servlet**

Ou « un peu d'HTML dans beaucoup de Java... » (Terme formé de la réunion de serv, début du mot serveur et de let, par analogie avec applet). Une servlet est un programme à part entière qui se doit de générer la totalité de la ressource demandée par l'internaute. Elle reçoit une requête du client, elle effectue des traitements et renvoie le résultat. Cette dernière est compilée comme un programme Java et déposé sur le serveur. Elle peut être invoquée plusieurs fois pour rependre aux requêtes simultanées.

- **Les JSP**

Ou « Un peu de Java dans beaucoup d'HTML … » Une JSP « Java Server Page » s'apparente à une page HTML simple dans laquelle du code Java a été incorporé. La page est alors interprétée par le serveur qui génère une servlet par un moteur inclus dans le serveur d'applications (Catalina dans notre cas puisque nous avons utilisé Tomcat comme serveur) lors de leur premier appel. Les JSP sont analogues aux pages PHP, à la seule différence près que les JSP sont compilée une fois pour toute par le serveur alors que les pages PHP sont interprétées à chaque appel de la page.

III.3.1.2 HTML

« Hyper Text Markup language » est le format de données conçu pour représenter les pages web. Il permet notamment d'implanter de l'hypertexte dans le contenu des pages. Il repose sur un langage de balises. Il permet aussi d'inclure des ressources multimédias dont des images, des formulaires de saisie….etc.

III.3.1.3 SQL

Le langage SQL (Structured Query Language) peut être considéré comme un langage d'accès normalisé aux bases de données. Il est aujourd'hui supporté par la plupart des produits commerciaux que ce soit par les systèmes de gestion de bases de données micro tel qu'Access ou par les produits plus professionnels tels qu'Oracle ou Sybase. Il a fait l'objet de plusieurs normes ANSI/ISO dont la plus répandue aujourd'hui est la norme SQL2 qui a été définie en 1992.

Le succès du langage SQL est dû essentiellement à sa simplicité et au fait qu'il s'appuie sur le schéma conceptuel pour énoncer des requêtes en laissant le SGBD responsable de la stratégie d'exécution. Le langage SQL propose un langage de requêtes ensembliste. Néanmoins, ce dernier ne possède pas la puissance d'un langage de programmation : entrées/sorties, instructions conditionnelles, boucles et affectations. Donc on a assurée ces traitements grâce au langage java (servlets/JSP) qui contient des requêtes SQL et les traitements sur ces dernières.

III.3.1.3 Java Script

Java script est un langage de script incorporé dans un document HTML. C'est un langage qui permet d'apporter des améliorations au langage HTML en permettant d'exécuter des commandes du côté client. Il permet d'effectuer des contrôles de saisis pour valider les champs d'un formulaire, d'utiliser les boite de dialogue (alerte, confirmation, …).

III.3.1.4 CSS (Cascading Style Sheets)

Les feuilles de style sont un élément essentiel à la création de site Web. Elles permettent une gestion normalisée, uniformisée la mise en page d'éléments dans un ensemble de pages Web. Elles permettent d'alléger les pages HTML en les débarrassant de toutes les balises de mise en forme et de maîtriser parfaitement la publication d'un ensemble de pages dont les caractéristiques sont identiques.

III.3.2 Outils

III.3.2.1 Netbeans 6.8

NetBeans fut développé à l'origine par une équipe d'étudiants à Prague, racheté ensuite par Sun Microsystems. en 2002, Sun a décidé de rendre NetBeans open-source. En plus d'être un IDE moderne et libre, La plateforme NetBeans est un outil très puissant pour la réalisation d'applications java et permet d'implémenter, documenter le code avec une grande facilité ainsi que de tester des Servlets sans configurer explicitement un serveur HTTP adapté.

La figure suivante nous montre une page de NetBeans 6.8 :

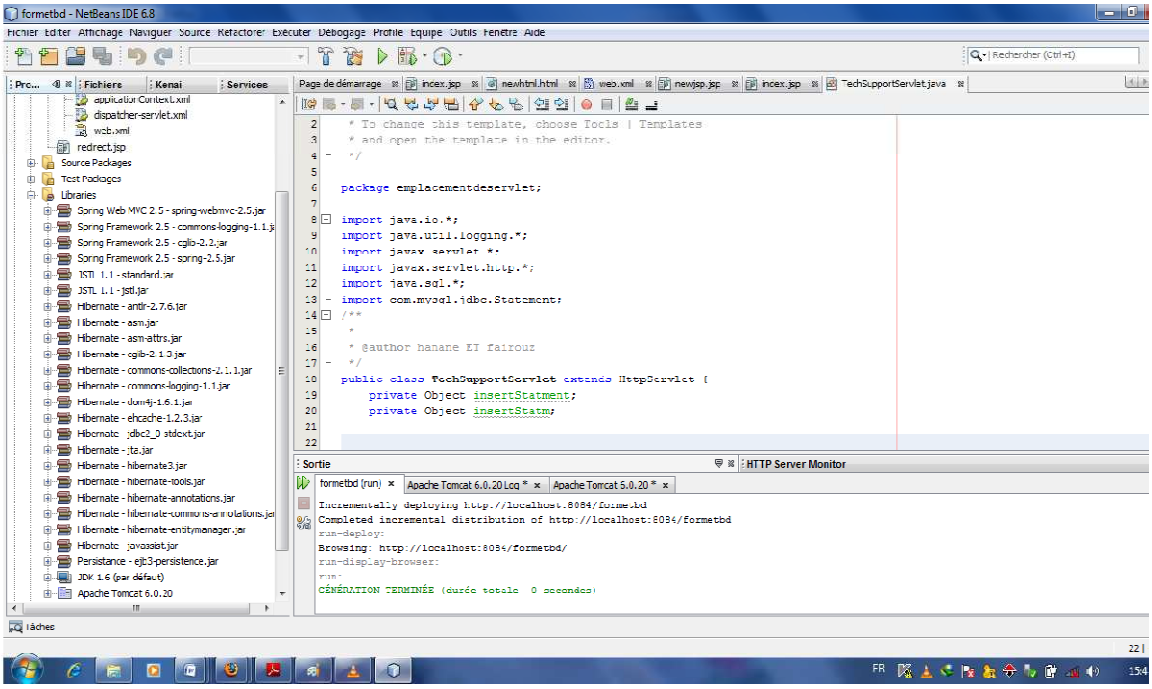

Figure III.1 : Plate-forme NetBeans 6.8.

III.3.2.2 Le serveur apache

Le serveur Apache ou Apache http Serveur est un serveur http produit par Apache Software Fondation en avril 1995, gratuit, libre et ouvert, disponible aussi bien sous forme binaire exécutable que sous forme de source ce qui lui permit de dominer le marché avec plus de 60% d'activité sur internet. Parmi les avantages d'Apache c'est qu'il est conçu pour prendre en charge de nombreux modules lui donnant des fonctionnalités supplémentaires : interprétation du langage perl, PHP, Python, CGI, Servlets et JSP java, réécriture d'URL, négociation de contenu…, ainsi qu'il peut fonctionner plusieurs systèmes d'exploitation UNIX, MacOs X,BSD, Linux et Windows. Dans notre cas nous l'avons utilisé avec le module Tomcat comme conteneur de servlet.

III.3.2.3 Le module Tomcat

Quand le serveur web reçoit une requête dont la réponse doit être construite par un processus, il confie en général l'exécution de ce processus à un module extérieur. Si le processus est une servlet ou une page JSP, ce module est appelé conteneur de servlet ou encore moteur de servlet. Il existe plusieurs conteneurs de servlet dans certains sont gratuit, parmi ces derniers, nous avons choisi d'utiliser tomcat, celui-ci n'est peut être pas le plus performant, mais c'est le plus répondu et seul le couple Apache/Tomcat a été testé tomcat peut être utilisé comme serveur indépendant, il joue alors les deux rôles serveur et conteneur, mais il ne fournit pas toutes les possibilités d'Apache en matière de sécurité notamment, c'est pour quoi nous l'utilisons comme conteneur de servlet derrière le serveur web Apache (Apache Tomcat 6.0.20).

III.3.2.4 Le serveur de données : (MySQL Server 5.4)

En plus de sa simplicité d'utilisation, le SGBD MySQL est un serveur de base de données SQL très rapide et flexible, multithread, multiutilisateur et robuste. Le serveur MySQL est destiné aux missions stratégiques et aux systèmes de production à forte charge, ainsi qu'à l'intégration dans des logiciels déployés à grande échelle. MySQL est utilisé depuis 1996 dans des environnements de plus de 40 bases de données, il est devenu le plus populaire et le plus utilisé au monde. Beaucoup des sociétés les plus importantes et à forte croissance telle que Google, l'autre point positif de serveur MySQL, c'est qu'il fonctionne sur plus de 20 plateformes dont on a : Linux, Windows, OS/X, HP-UX, AIX, Netware.

III.3.2.5 Le middleware JAVA Data Base Connectivity (JDBC):

III.3.2.5.1 Définition de JDBC

Pour assurer la compatibilité de JAVA avec diverses bases de données, les applications java utilisent les mêmes instructions pour s'adresser au pilote JDBC, qui est un ensemble de classes et d'interfaces qui prennent en charge les spécificités du serveur de base de données,

ainsi, il permit à un programme java d'accéder via des requêtes SQL à un moteur de bases de données et facilite le changement d'éditeur de base de données.

III.3.2.5 .2 Utilisation de JDBC

Quand un programme java souhaite accéder à une base de données, il commence par demander le chargement du pilote en mémoire ensuite l'utiliser dans une servlet ou JSP pour établir la connexion et effectuer les requêtes souhaitées.

III.4.Présentation de l'application :

III.4.1. Connexion :

Figure III.2. Page d'authentification.

Pour accéder à l'interface de l'application, l'utilisateur (Administrateur, collaborateur, enseignant, étudiant) doit s'authentifier.

III.4.2.Interface d'accueil administrateur :

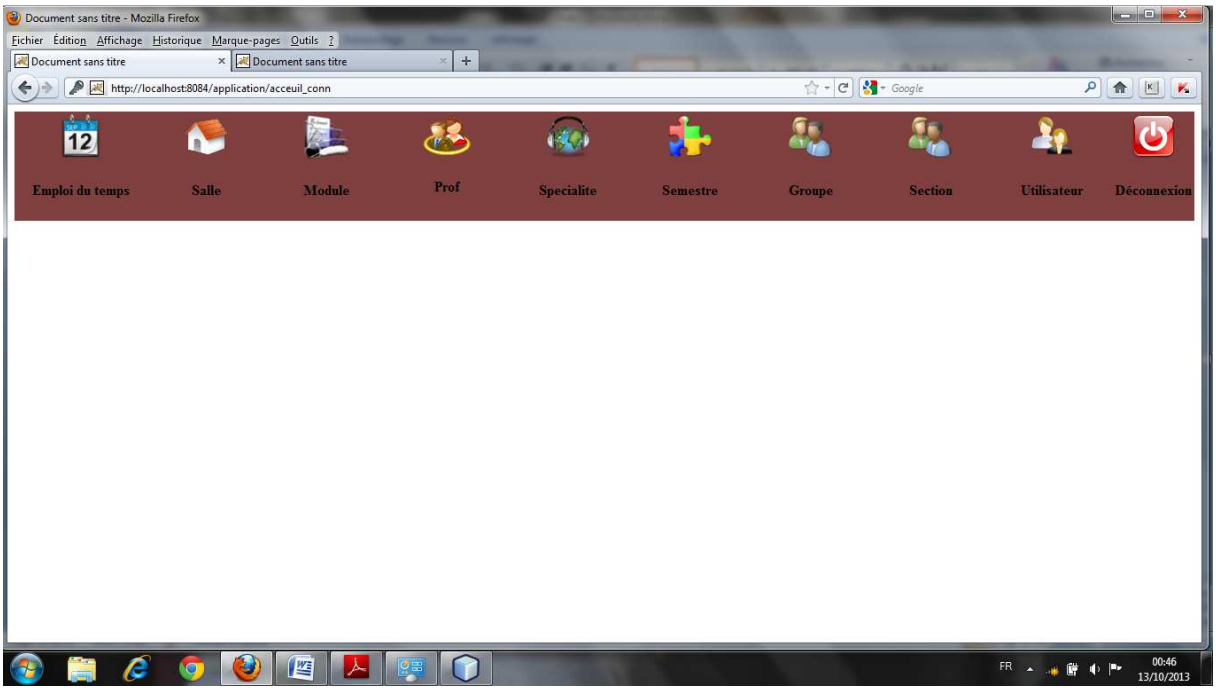

Figure III.3.Interface d'accueil de l'administrateur

Cette interface, illustre l'interface d'accueil de l'administrateur, il dispose d'un menu d'icones nécessaires à la gestion de l'emploi du temps (salle, prof, spécialité,….), ainsi que l'icone de gestion des utilisateurs et de Déconnexion.

III.4.3.Interface d'accueil collaborateur :

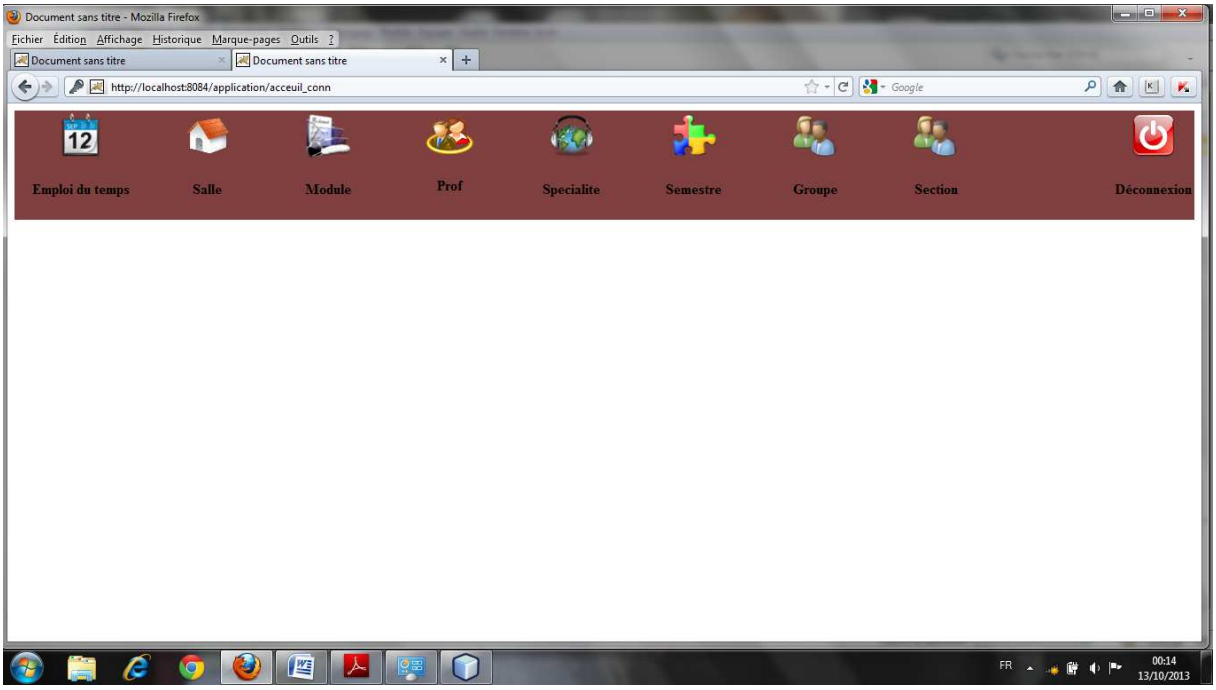

Figure III.4.Interface d'accueil du collaborateur

Cette interface, illustre l'interface d'accueil du collaborateur, il dispose d'un menu d'icones nécessaires à la gestion de l'emploi du temps (salle, prof, spécialité….), ainsi que l'icone de Déconnexion, mais il ne peut pas gérer les utilisateurs contrairement a l'administrateur,

III.4.4.Prof

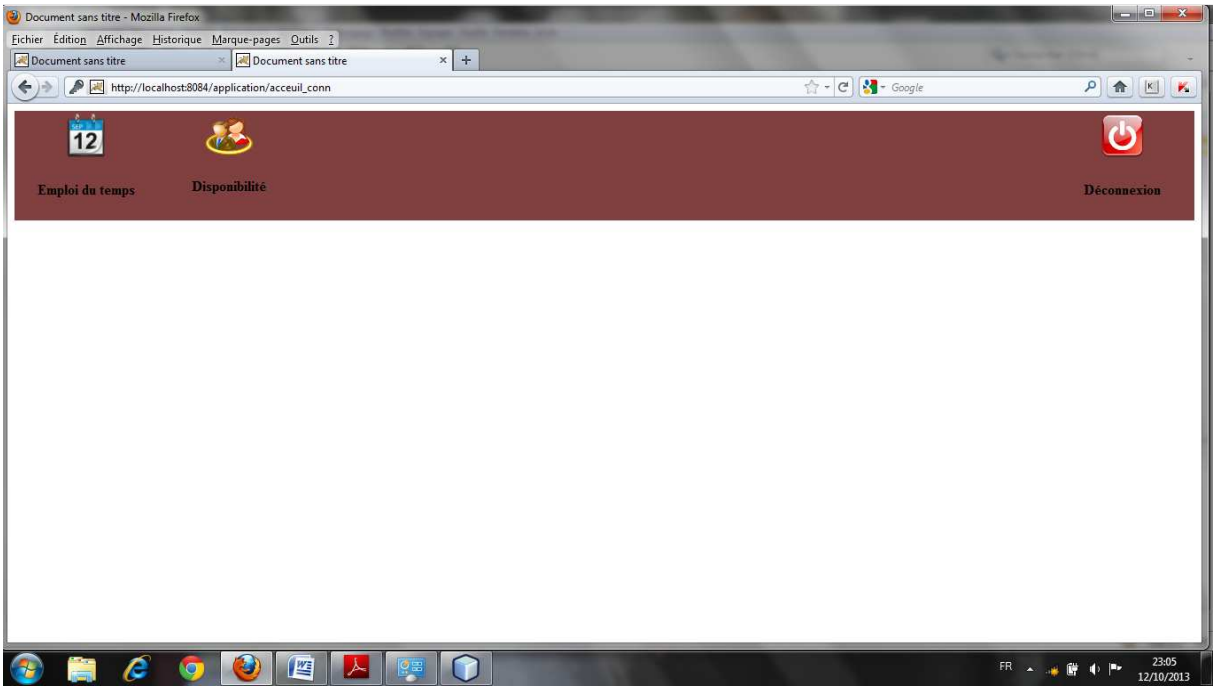

Figure III.5.Interface d'accueil du prof

Cette interface, illustre l'interface d'accueil du prof, il dispose d'un menu de trois icones. Une icone pour consulter son emploi du temps, une autre pour consulter sa disponibilité ainsi que l'icone de Déconnexion.

III.4.5. Etudiant

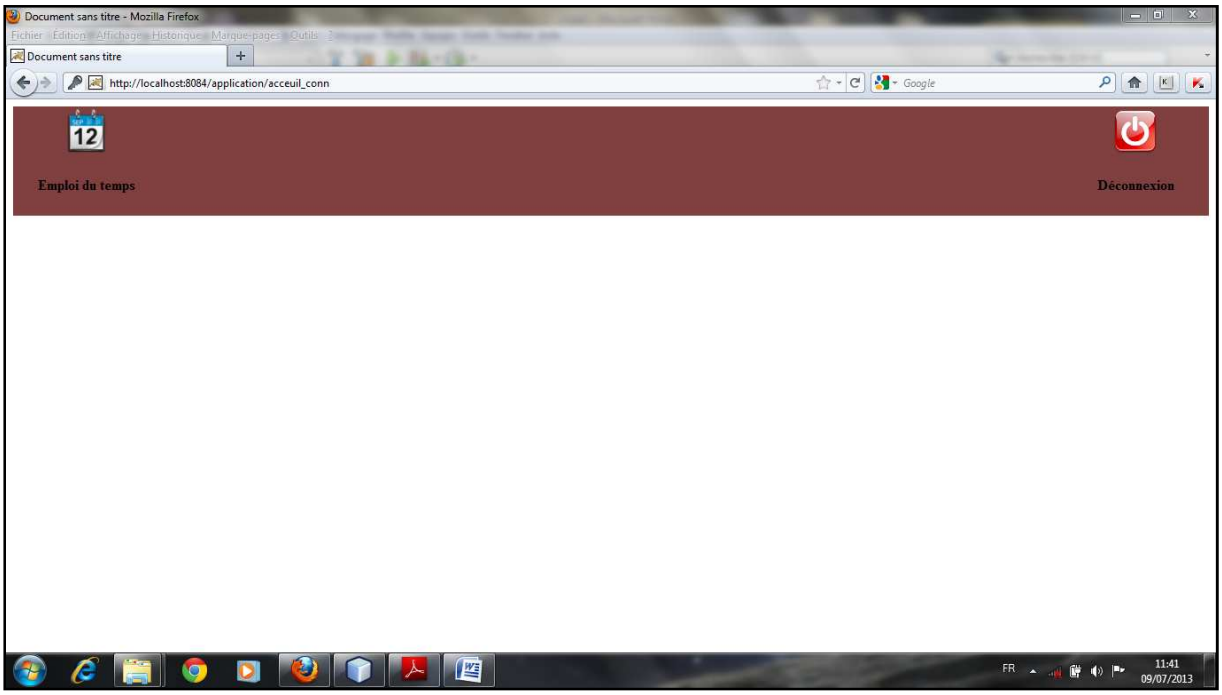

Figure III.6.Interface d'accueil de l'étudiant

Cette interface, illustre l'interface d'accueil de l'étudiant, il dispose d'un menu de deux icones. Une icone pour consulter son emploi du temps, ainsi que l'icone de Déconnexion.

III.4.6.Quelque interface de l'application :

III.4.6.1.Interface gestion des salles :

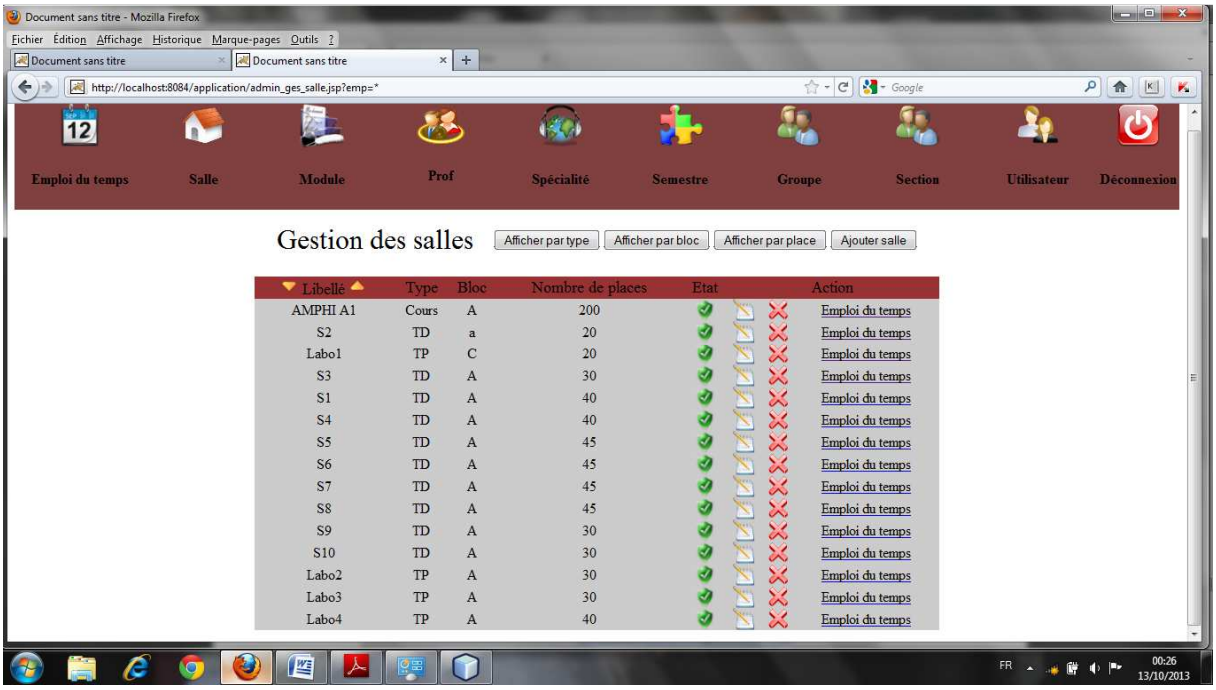

Figure III.7.Interface gestion des salles

III.4.6.2.Interface gestion des modules :

Figure III.8.Interface gestion des modules

III.4.6.3.Disponibilité enseignant :

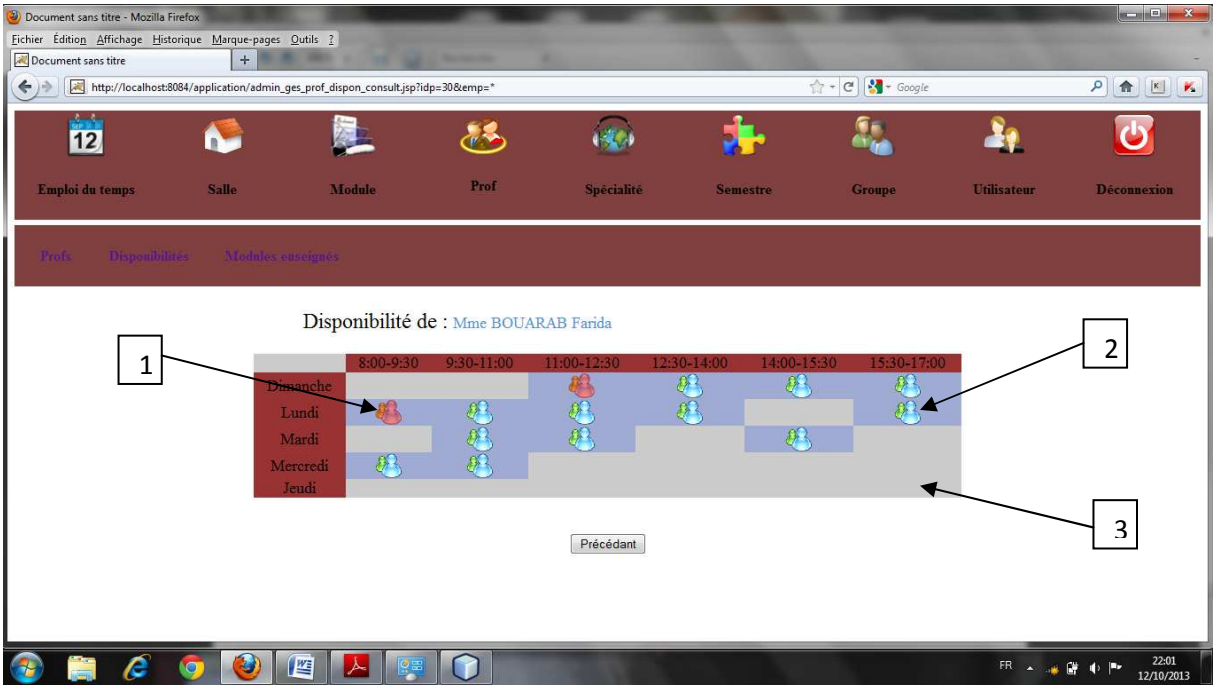

Figure III.9.Interface disponibilité enseignant

- Le numéro (1) sur la figure, veut dire que le prof est programmé pour une séance de cours.
- Le numéro (2) sur la figure, veut dire que le prof n'est pas programmé pour une séance de cours.
- Le numéro (3) sur la figure, veut dire que le prof n'est pas disponible.

III.4.6.4.Gestion des emplois du temps :

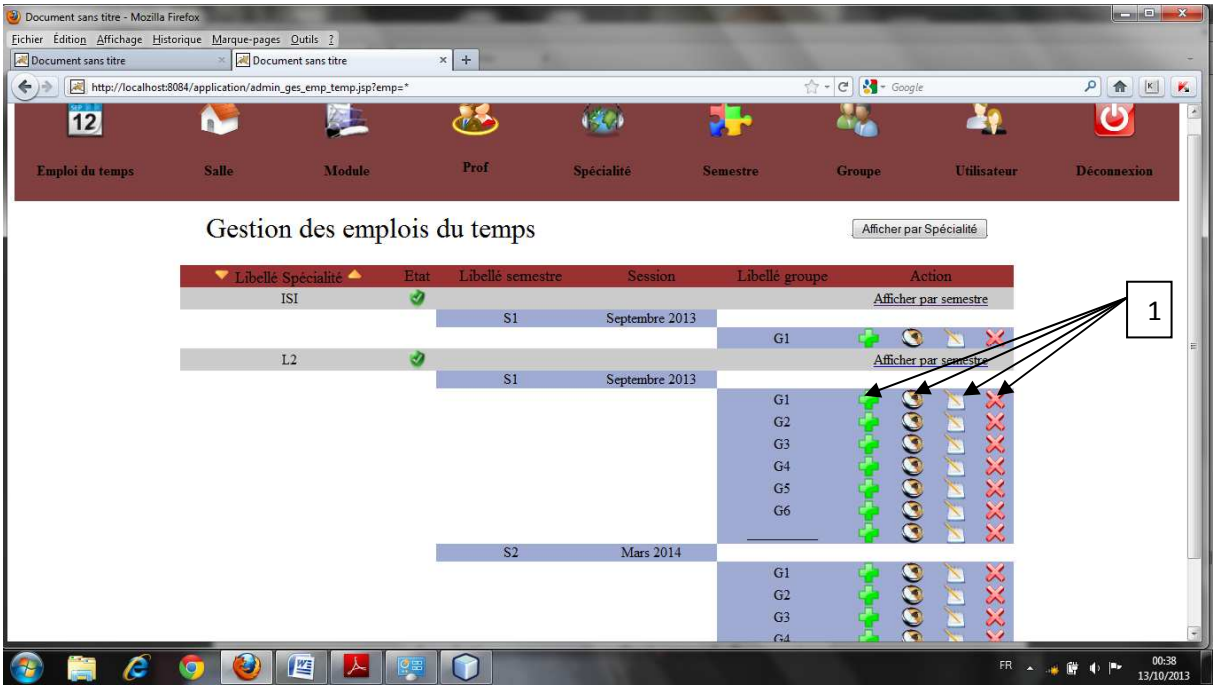

Figure III.10.Interface gestion des emplois du temps

Des icones de (création, consultation, modification et suppression) de chaque emploi du temps (pour un groupe, dans un semestre, pour une spécialité) désigné sur la figure par le numéro (1).

• Si l'admin/collab veut créer un emploi du temps, il n'a qu'a appuyer sur la première icone en parton de la gauche (✚), l'interface de création va apparaitre, illustrer par la figure suivante:

III.4.6.5.Créer emploi du temps section :

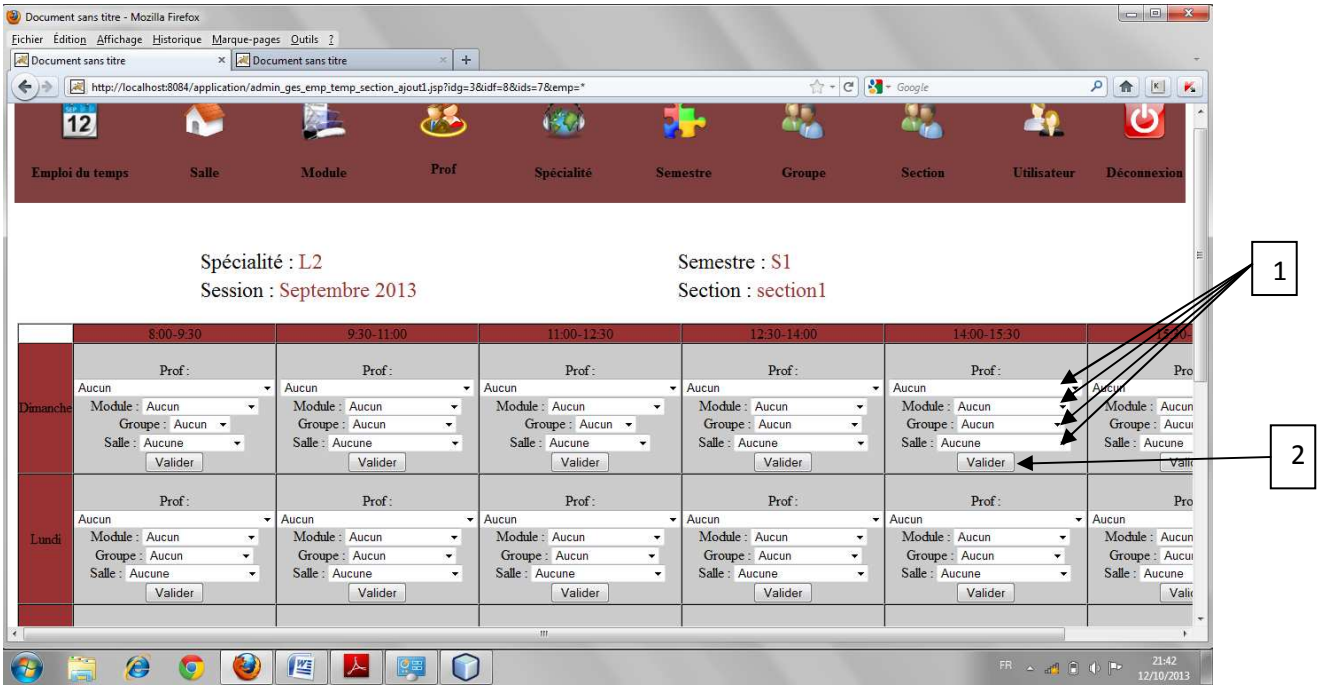

Figure III.11.Interface « Créer emploi du temps section »

L'admin/collab va choisir le prof, le module le groupe et la salle dans la liste déroulante désigné par le numéro (1) et il valide (2).

- Si l'admin/collab veut consulter un emploi du temps, il n'a qu'a appuyer sur la deuxième icone en parton de la gauche ( ), l'interface de consultation va apparaitre, illustrer par la figure suivante :

III.4.6.6 .Consulter emploi du temps

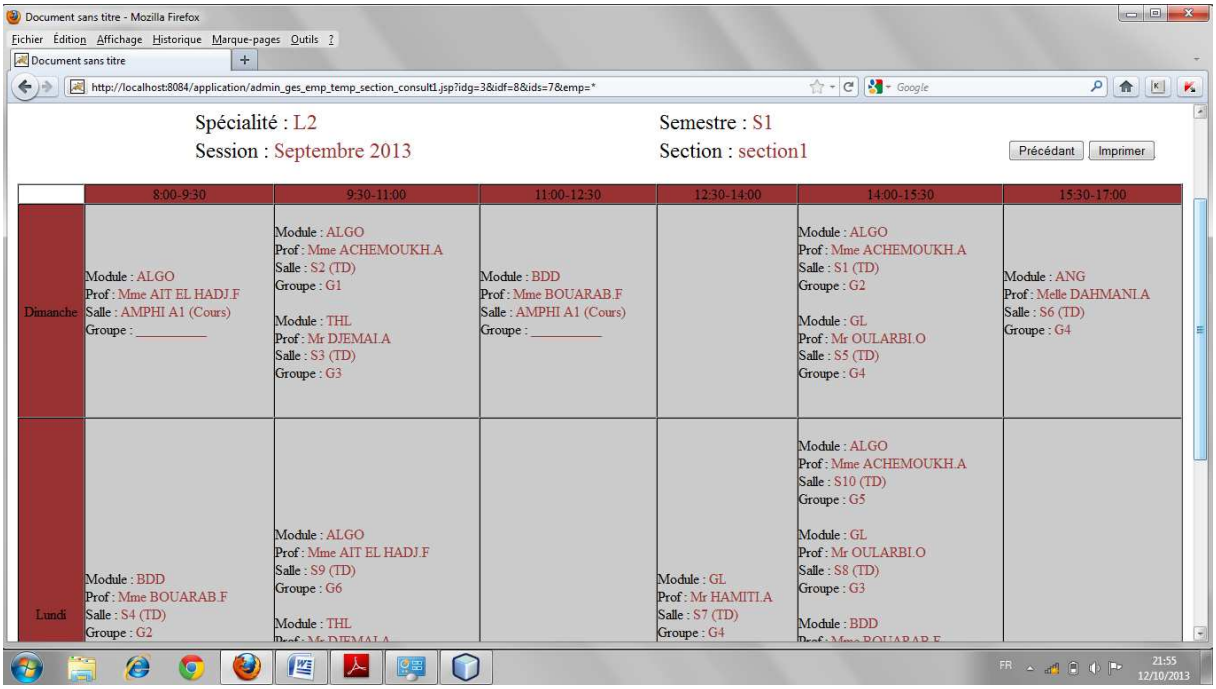

Figure III.12.Interface « Consulter emploi du temps section »

III.5 Conclusion :

Dans ce chapitre, nous avons abordés la réalisation de notre application d'une manière générale. Nous avons insisté sur les outils de travail et la présentation des interfaces.

Conclusion Générale

Conclusion Générale

L’objectif de notre travail était de développer une application web pour la gestion des emplois du temps dans un établissement universitaire, afin de facilité la vie de toutes personnes participant a la mise en place de ce dernier, et ainsi exploiter au mieux les ressources humaines et matérielles.

La réalisation de ce travail nous a permis de bénéficier de plusieurs avantages. Dans un premier temps, nous avons pu approfondir nos connaissances théoriques et pratiques en rapport avec le Web et les techniques de programmation et dans un second temps, nous nous sommes familiarisées avec un certain nombre d’outils et de logiciels de développement Web tels que l’environnement NetBeans et le système de gestion de base de données MySQL.

Un volume important de travail reste à faire. Actuellement les affectations se font de manière semi automatique c.à.d. que le système laisse le choix à l’opérateur l’affectation manuelle d’un module à un enseignant en surveillant la disponibilité des ressources et des enseignants. Il serait intéressant de passer dans le futur à l’affectation automatique.

Enfin, on espère que notre travail sera d’une utilité à toute personne intéressée par ce sujet et que la gestion du temps prenne une place importante dans la vie quotidienne des personnes.

Bibliographie

[01] Olivier Sigaud , « Introduction à la modélisation orientée objets avec UML » Edition 2005-2006

[02] Gilles Roy, « Conception de bases de données avec UML » Edition 2009

[03] Claude Delannoy, « Programmer en Java » 5éme édition

[04] Jerome Moliére, « les Cahiers du Programmeur J2EE »

[05] Van Lancker Luc, « Apprendre le langage Html »

[06] Luc VAN LANCKER, « Des CSS au DHTML - JavaScript appliqué aux feuilles de style »

[07] Mathieu Nebra, « Reussir Son Site Web XHTML-CSS » 2e tirage 2007

[08] Christian Soutou, « Apprendre SQL avec MySQL » Edition Eyrolles, 2006

[09] Mme Troudi Fatiha « Résolution du problème de l'emploi du temps : Proposition d'un algorithme évolutionnaire multi objectif », Université Mentouri – Constantine, 2005-2006.

[10] «Analyse et conception d'un outil interactif d'aide à la gestion des emplois du temps basé sur les points de vue», **S. Piechowiak, C. Kolski,** Université de
Valenciennes et du Hainaut-Cambrésis, 59313 Valenciennes Cédex 9– France.