République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid – Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: *Système d'Information et de Connaissances (S.I.C)*

Thème

Un système automatique pour la génération des emplois du temps basé sur le logiciel national gestion de scolarité SEES

Réalisé par :

- M^elle^. BOUZIANE Houria

Présenté le 23 octobre 2012 devant le jury composé de MM.

- BENMAAMAR Bader *(Président)*
- BENAMAR Abdelkrim *(Encadreur)*
- ELMIR Youcef *(co-encadreur)*
- BENZIANE Mohammed Yaghmorasan *(Examinateur)*
- BENAISSA Mohamed *(Examinateur)*

Année universitaire: 2011-2012

Remerciements

Il est de coutume de dire qu'une thèse n'est pas le fruit du seul travail de son auteur, mais le résultat de nombreuses et étroites collaborations; celle-ci ne déroge pas à la règle.

Nous remercions avant tout le Bon Dieu de nous avoir donné la volonté de finir ce mémoire.

Ce travail a pu voir le jour avec énormément d'aide et encouragement des personnes autour de nous. Ce court remerciement ne sera pas suffisant pour récompenser leurs efforts mais tout de même ...

A l'issue de deux agréables années au sein de département d'informatique de l'université ABOU BAKR BELKAID de Tlemcen nous tenons à remercier l'ensemble des enseignants pour leur dévouement sans oublier d'adresser des remerciements particuliers à Monsieur Benamar, le chef du département, pour le dynamisme de ce département d'études.

Ensuite, toutes nos pensées de gratitude se dirige vers notre encadreur Mr.Elmir Youcef et co-encadreur Mr.Benamar, qui en tant qu'un encadreur de mémoire, s'est toujours montré a l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'il a bien voulu nous consacrer, et qui sans son concours ce mémoire n'aurait jamais vu le jour.

Nous tenant aussi à remercier les membres du jury qui ont accepté d'examiner notre mémoire.

Nous adressons nos plus sincères remerciements à tous nos collègues et nos amis qui partagent avec nous les bons moments de l'étude pendant les deux années.

Nous exprimons nos gratitude a tous nos proches qui nous ont toujours soutenue et encouragée au cours de la réalisation de ce mémoire.

Enfin, tous ceux qui ont Contribués, de prés ou de loin à la réalisation de cette thèse et que nous ne pouvons malheureusement citer, trouvent ici l'expression de notre profonde gratitude.

Dédicace

Je dédie ce travail à

Mes très chers parents.

Mes très chères sœurs.

Mes très chers frères.

A tout la famille

Département d'informatique d'ADRAR

Mr.Cheragui Mohamed Amine

Mr.kohili Mohamed

Mr.Boukerraf Mohamed

Mr.Belaoura Abdelwahab

Tous mes amis, ainsi que toute la promotion du Master 2 en informatique SIC 2011/2012 de l'université de Tlemcen.

Mes très chères amies Djihad & ZOHRA & Assia

Table de matière

Table de matière

Liste des figures……iv

Liste d'abréviations……v

Introduction générale……..1

Chapitre I : Etude de l'existant

I. Introduction……4

II. Etude Préalable……...4

II.1. Présentation de l'organisme d'accueil……4

II.1.1. Présentation de l'UABT……4

II.1.2. Mission……...6

II.2. Cadre de l'étude……6

II. 2.1. Infrastructures et moyens humains……7

III. Analyse de l'existant……..7

III.1.Introduction……7

III.2. Etude du système pédagogique……..7

IV. Conclusion……..12

Chapitre II : Méthodes d'Intelligence Artificielle pour la Génération d'emplois du temps

I. Introduction……14

II. Problème d'affectation sous contrainte……14

II. 1. Complexité……15

II. 2. Formulation des problèmes d'affectation sous contrainte……16

III. Méthodes de résolution……...16

III. 1. Approches de construction……..17

III. 2. Approche séquentielle……19

III. 2 .1. La méthode descente……...19

III. 2 .2. Le recuit simulé……...20

III. 2 .3. Les algorithmes d'acceptation avec seuil……22

III. 2 .4. La méthode Tabou……...23

III. 3. La méthode évolutive……..24

III. 4. Les Méthodes hybrides……25

IV. Conclusion……..25

Chapitre III : Conception et mise en œuvre

I. Introduction……27

II. Outils et les langages utilisés……27

II.1. Langage PHP…….27

II.2. Serveur web……...28

III. Présentation de la base……28

IV. Processus de développement logiciel…….28

IV.1. Processus Unifié……..28

IV.2. Modélisation avec UML……..29

IV.2.1.Diagramme de cas d'utilisations……....29

IV.2.2. Diagramme de séquence……30

IV.2.3. Diagramme de classe……30

V. Présentation de l'IHM……..31

VI. Conclusion……..35

Conclusion générale…….37

Références bibliographiques……39

Annexe…….....41

Liste des figures

Figure I.1 : Organigramme de la Faculté des Sciences……...2

Figure II.1 : Classement des méthodes de résolution en RO et IA……13

Figure III.1 : Diagramme des cas d'utilisation…….25

Figure III.2 : Diagramme de séquence de cas « génération d'EDT »……..26

Figure III.3 : Diagramme de Classes ……27

Figure III.4 : interface pour accéder …….28

Figure III.5 : choisie les informations pour générer l'EDT ……..28

Figure III.6 : Choisir le jour pour affecter la séance…….29

Figure III.7 : Choisie la promotion pour afficher……..29

Figure III.8 : EDT d'une promotion…….30

Figure III.9 : Choisie l'enseignant pour afficher EDT……..30

Figure III.10 : Affichage résultat EDT d'un enseignant …….31

Liste des abréviations

HTML	**H**ypertext **M**arkup **L**anguage.
IA	**I**ntelligence **A**rtificielle
IHM	**I**nterface **H**omme/**M**achine
PHP	**P**ersonal **H**ome **P**age
PASC	**P**roblème de **S**atisfaction des **C**ontraintes
POSC	**P**roblème d'**O**ptimisation **S**ous **C**ontraintes
PSCM	**P**roblème de **S**atisfaction des **C**ontraintes
RO	**R**echerche **O**pérationnelle
SGBD	**S**ystème de **G**estion de **B**ase de **D**onnées
SQL	**S**tructural **Q**uery **L**anguage
UML	**U**nified **M**odelling **L**anguage

Introduction générale

Introduction Générale

La construction d'un emploi du temps est un travail long et difficile à mettre en place et qui doit être répétée à intervalle régulier. C'est pour cette raison que l'automatisation d'emplois du temps, est une tâche d'une grande importance car elle permet de gagner beaucoup d'heures de travail pour les employés, les institutions et les entreprises, et de fournir des solutions optimales avec satisfaction de contraintes en quelques minutes, ce qui peut accroître la productivité, la qualité de l'éducation, qualité des services et enfin, la qualité de vie.

La difficulté de cette tache fastidieuse et répétitive, qui fait généralement intervenir de nombreux éléments d'information, est liée à la nature des contraintes qu'il s'agit de satisfaire. A celle-ci s'ajoutent des caractéristiques bien particulières de l'institution ou l'organisation concernée. Parce qu'il y a plusieurs modèles du problème, des contraintes qui changent, et des caractéristiques particulières pour chaque institution ou organisation, il devient de plus en plus difficile de trouver une solution générale pour les problèmes d'emploi du temps, et c'est pour cela que ce domaine a besoin de plus en plus de recherche.

Dans le domaine de l'éducation, le problème connu sous le nom d'emploi du temps consiste à organiser des rencontres entre des enseignants et des groupes d'étudiants en définissant un lieu et une heure de rencontre tout en satisfaisant un ensemble de contraintes. Il existe différentes versions de ce dernier mais la plupart peuvent être classifié en trois grandes catégories qui sont : emploi du temps pour examen et emploi du temps universitaire.

La problématique de génération d'emploi du temps du département d'informatique de l'université de Tlemcen représente un vrai challenge semestriel.

L'existence d'un horaire respectant à la lettre toutes les contraintes n'est pas garantie dans le cas général. En réalité, il s'agit de trouver un horaire aussi bon que possible dans un temps de calcul raisonnable tout en tolérant la présence de conflits, le nombre de ces conflits doit être minimisé pour que l'horaire soit jugé satisfaisant par les différentes parties concernées (direction de l'université, Intervenant, Etudiants,…etc.)

Dans ce mémoire nous allons nous intéresser au problème de génération automatique d'emploi du temps universitaire et plus précisément à celui du département d'informatique de l'université de Tlemcen. Ce problème entre dans le cadre de plusieurs projets recherche tel que Chronos (a pour but de construire une application de génération automatique d'emploi du

temps pour dans le département d'informatique de l'université de Tlemcen basé sur les algorithmes d'optimisation).

Par conséquence, le problème de gestion automatique d'emploi du temps est formulé comme un problème d'optimisation combinatoire.

Il existe en littérature deux classes de metteures de résolution de problème d'optimisation combinatoire à savoir la recherche opérationnelle et l'intelligence artificielle.

D'une manière très générale, les méthodes de résolution suivent quatre approches différentes pour la recherche d'une solution : l'approche de construction, l'approche de relaxation, l'approche de voisinage et l'approche d'évolution.

Ces méthodes font partie de deux groupes de nature différente. Le premier groupe comprend les méthodes exactes d'arborescence qui garantissent la complétude de la résolution : c'est le cas de SEP, A* et CSP.

Le second groupe comprend les méthodes approchées dont le but est de trouver une solution de bonne qualité en un temps de calcul raisonnable sans garantir l'optimalité de la solution obtenue. Les méthodes approchées sont fondées principalement sur diverses heuristiques, souvent spécifiques à un type de problème. Les métaheuristiques constituent une autre partie importante des méthodes approchées et ouvrent des voies très intéressantes en matière de conception de méthodes heuristiques pour l'optimisation combinatoire.

Dans la suite nous nous intéresserons essentiellement aux métaheuristiques : les méthodes de voisinage et les algorithmes évolutifs. Nous présentons également les possibilités de combiner différentes méthodes pour créer des méthodes hybrides.

Le présent rapport est composé de trois chapitres. Son organisation est comme suit :

- Dans le premier chapitre de ce mémoire, nous parlons de la présente une étude préalable de l'existant dans le département d'informatique en faculté des sciences université Abou Baker Belkaid Tlemcen.
- Le chapitre qui suite, concerne la des différentes méthodes utilisées pour résoudre ce genre de problème.
- Le troisième chapitre présentera les conceptions et mise en œuvre qui concerne la présentation de la technologie PHP/MYSQL. Dans cette partie, nous illustrons par des exemples le fonctionnement de cette technologie ainsi détaille la conception en diagrammes de langage UML (Unified Modeling Langage) et applications de réalisation d'emploi du temps.

Chapitre I

I. Introduction

L'analyse de l'existant est une étape importante dans le cycle de vie d'un système, il s'agit de connaître la situation actuelle de l'organisation pour pouvoir porter un jugement juste. Ainsi, l'analyse du système existant doit nous fournir toute l'information nécessaire, afin d'établir une bonne conception et de proposer de bonnes solutions.

Dans ce chapitre, nous allons présenter une étude générale du système pédagogique universitaire (ressources logicielles et matérielles) concerné par notre étude. Par la suite, un intérêt particulier est porté à l'application existante de gestion des emplois du temps du département d'informatique. Dans un premier temps nous présenterons l'Université Abou Bekr Belkaid Tlemcen (UABT).

II. Etude Préalable

II.1. Présentation de l'organisme d'accueil

II.1.1. Présentation de l'UABT

L'UABT est le fruit d'une longue évolution, crée par le décret n° 89 – 138 du 01-08-1989, modifié et complété par le décret exécutif N°95-208 du 05/12/1998.

Le pôle de Chetouane est situé au niveau de la commune de Chetouane «nord-est» de la ville de Tlemcen dans un cadre agréable et à proximité de la zone industrielle, et Il s'étend sur une superficie de 38 hectares. De plus, ce pôle abrite la faculté des sciences et la faculté de technologie pour une capacité théorique de 4000 places.

La faculté des sciences comporte les quatre départements suivants: informatique, mathématiques, chimie et physique. Elle est dotée de six blocs suivants :

1- Bloc administratif (Doyenné)
2- Bloc pédagogique
3- Bloc des amphis (trois amphis à 100 places)
4- Bibliothèque
5- Hall des technologies
6- Bloc des laboratoires (informatique, physique et chimie)

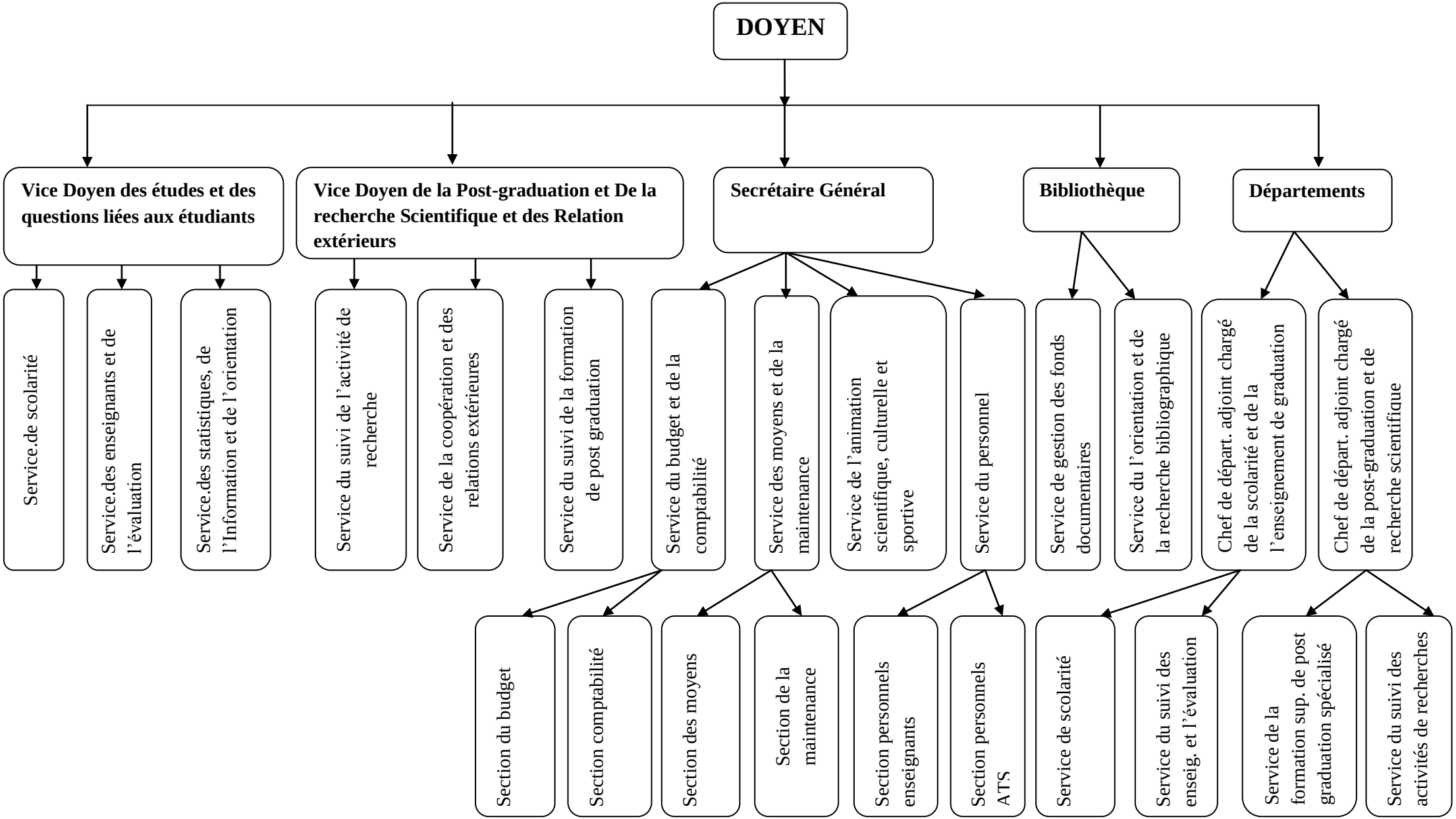

Figure I.1: **Organigramme de la Faculté des Sciences**

Le département d'informatique de l'UABT avait assuré depuis sa création en 1998 jusqu'à 2005, la formation du système classique des deux cycles de la graduation à savoir le cycle court en vue de l'obtention du diplôme de DEUA en informatique et le cycle long en vue de l'obtention du diplôme d'ingénieur d'état en informatique (options : système d'information avancé et informatique industrielle). A partir de l'année universitaire (2005/2006) le département d'informatique a commencé à prendre en charge la formation du système L.M.D (Licence, Master et Doctorat).

II.1.2. Mission

De par les missions qui lui ont été conférées, l'UABT s'est fixé comme objectif de former des éléments compétents capables de relever les défis du futur et de contribuer au développement national. Ses principales missions se résument comme suit :

- Les formations en 1er cycle (Licence: Informatique et Réseaux et Développement Web) et 2eme cycle (Master: Système d'Information et de Connaissances, Modèles Intelligents et Décision, et Réseaux et systèmes Distribués)).
- La formation du 3eme cycle (Doctorat: Intelligence Artificielle et Ingénierie de Connaissances).
- L'assistance du secteur industriel et socioprofessionnel en matière d'informatisation.
- Le développement et la promotion de la recherche scientifique dans les domaines de technologies de pointe notamment les technologies de l'information et de la communication, en coopération avec des centres de recherches et des universités nationaux et internationaux.
- La formation continue pour le perfectionnement de cadres d'entreprises et l'assistance du secteur industriel.
- L'orientation des nouveaux bacheliers vers les établissements universitaires au niveau national.

II.2. Cadre de l'étude

Chaque année, les responsables pédagogiques de la direction des études ont pour mission de concevoir les emplois du temps des différentes filières 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 salles, les équipements, etc.).

Ainsi, l'idée de mettre en place un système de gestion des emplois du temps est née, dont l'objectif est:

- La génération automatique des emplois du temps.
- Consultation et suivi des emplois du temps.

Par conséquent, notre étude concerne la direction des études et en particulier le service du suivi des enseignements et de l'évaluation.

II. 2.1. Infrastructures et moyens humains

Pour son bon fonctionnement, la direction des études dispose d'infrastructures et de moyens conséquents, à savoir :

- 08 salles de cours et travaux dirigés.
- 04 salles de travaux pratiques dotées de matériels performants.
- 40 machines reliées et connectées au réseau local.

III. 1. Analyse de l'existant

III. 1. 1. Introduction

Dans un premier temps, il a été nécessaire d'analyser le système (dans notre cas : l'environnement pédagogique universitaire) dans lequel doit s'intégrer l'outil de la gestion des emplois du temps visé. Cette analyse a été effectuée à partir d'interviews, d'observations et de lectures de supports écrits à l'activité de responsables pédagogiques, de responsables de salles ou de responsables de matériels ; ces analyses ont été effectuées sur leur lieu de travail au département d'informatique.

III. 1. 2. Etude du système pédagogique

Il s'avère que ce système ne possède pas de particularité pédagogique ciblée et possède de nombreuses similarités avec le fonctionnement d'autres universités. Ainsi, l'UABT est une université qui regroupe différentes formations du LMD. Elles ont une durée qui varie entre trois à huit ans. Les étudiants s'inscrivent en début de chaque année universitaire, c'est-à-dire généralement au début de septembre. Mais cela peut varier d'une formation à l'autre. Le programme pédagogique de chaque formation est connu. Ce programme précise les matières à suivre, leurs volumes horaires et quelques informations pédagogiques (répartition en cours, TD et TP). Selon les besoins pédagogiques et les conditions physiques des ressources, chaque formation est structurée en sections qui peuvent eux-mêmes être scindés en groupes..

Administrativement, les enseignants doivent assurer un nombre minimal d'heures qui est défini dans leur statut. Selon le type d'enseignement, on pondère le nombre d'heures par un coefficient. Ainsi, par exemple, une heure de cours est comptabilisée comme 1,5 heure de travaux dirigés. Lorsqu'un enseignant accepte la responsabilité d'un enseignement, il est tenu d'en respecter le volume horaire prévu par le responsable pédagogique. En cas d'absence, un 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 volume horaire d'enseignement.

Chaque formation est gérée par son responsable pédagogique qui crée les emplois du temps en concertation avec les enseignants mais aussi, moins souvent, avec les responsables des autres formations. Les enseignants peuvent assurer leurs enseignements dans différentes formations. Lors de la création des emplois du temps, les responsables pédagogiques ne tiennent généralement pas compte explicitement des salles disponibles. Dans la réalité, les responsables pédagogiques s'appuient sur l'expérience des années précédentes pour planifier les enseignements.

Une fois que les emplois du temps sont créés, l'attribution des salles est assurée par différents responsables qui peuvent être des personnels administratifs ou techniques ou même des enseignants selon le type des salles de cours, TD ou TP. L'emploi du temps sert également aux personnels administratifs pour suivre le travail des intervenants en vue de leur rémunération, pour prévoir l'achat de nouveaux équipements ou de leur mise en maintenance, pour argumenter la demande d'acquisition de nouveaux locaux, etc. Sur le plan pédagogique, l'étude des emplois du temps sert à mettre en évidence certains problèmes tels que les surcharges de travail des intervenants, les mauvaises répartitions des enseignements, et elle permet ainsi de corriger et améliorer le planning les années suivantes. Le nombre des ressources disponibles croît, malheureusement, bien moins rapidement que le nombre des formations ou que leur diversification. Cela oblige donc les responsables à « optimiser » la planification des ressources et rend plus délicate leur gestion. Le besoin d'un outil d'aide devient donc essentiel pour cette gestion efficace des ressources disponibles. L'objectif n'étant pas de remplacer les acteurs actuels mais de les aider, il est donc important de considérer les différents profils des futurs utilisateurs, en particulier par rapport à leur maîtrise des outils informatiques. Pour satisfaire un large éventail de profils d'utilisateurs et faciliter l'acceptation de l'outil visé, il convient de soigner en particulier la robustesse de celui-ci vis-à-vis des erreurs de manipulation, la rapidité d'exécution des commandes, la puissance des fonctions assurées.

III. 1. 2. 1. L'activité pédagogique

L'activité pédagogique est modélisée à l'aide de trois entités : les matières, les enseignements et les modules. D'une manière générale, les enseignants, les groupes, les matériels constituent les ressources de l'enseignement. Les matériels sont les ressources qui seront attribués aux séances de l'enseignement. Un enseignement peut être assuré simultanément par plusieurs enseignants. C'est le cas, par exemple de certains TP d'informatique. Les modules sont des ensembles d'enseignements.

III. 1. 2. 2. Les 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. 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 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. 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 matières 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 (un binôme ou monôme). Cette modélisation ne fait pas de distinction entre groupe et étudiant. En général les étudiant d'une filière sont décomposer en section et chaque section est décomposé en plusieurs groupes.

- **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 enseignements particuliers, l'enseignant peut avoir besoin de matériel spécifique.

III. 1. 2. 3. La notion du temps

Les analyses menées avec les utilisateurs mettent en avant différentes perceptions du temps. 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 l'enseignement est d'au moins 15 jours (*durée*). Par exemple, dire « qu'il y a cours de mathématiques tous les quinze jours » c'est-à- dire que la durée qui sépare deux séances consécutives de ce cours est de 15 jours. Ainsi on distingue la durée de chaque séance et de la durée qui sépare les séances.

- **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*=8h30 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.

- **Les séances et les réservations**

Une *séance* correspond à une instance temporelle d'un enseignement à une date donnée, pendant un créneau précis. 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.

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.

III. 1. 2. 4. Les filières

La notion de filière permet de limiter le nombre des ressources à considérer. En effet, le responsable d'une filière n'a besoin de connaître que le planning des ressources qu'il utilise. Cependant, certains conflits ne peuvent être détectés que lorsque les différentes filières 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 filières ou qu'une salle soit partagée par plusieurs filières.

III. 1. 2. 5. Les contraintes

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:

1- une ressource ne peut pas être occupée en même temps dans deux séances différentes ;
2- dès qu'une ressource r est occupée par une séance s, toutes ses ressources filles rf sont également occupées par la même séance et cela récursivement (il s'agit d'une contrainte liée à la hiérarchisation) ;
3- on ne peut pas mettre plus d'étudiants qu'il n'y a de places dans une salle
4- le volume horaire total des séances d'un enseignement ne peut pas dépasser le volume prévu ;
5- les calendriers sont respectés pour toutes les ressources.

- **Les contraintes pédagogiques**

Lors des interviews, des contraintes pédagogiques ont pu être identifiées. Elles 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:

1- homogénéiser la durée des séances d'un enseignement,
2- éviter les séances creuses dans l'emploi du temps,
3- éviter de placer plusieurs séances d'un même enseignement dans une journée,
4- éviter de finir après 18h30,
5- éviter de placer des cours l'après midi,
6- 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 séances creuses dans l'emploi du temps » représente l'ensemble de toutes les contraintes « éviter les séances creuses d'une durée D dans l'emploi du temps », où D $\in$ [30min, 1h, 1h30, …]. Mais il est moins important d'éviter les séances creuses de 60 minutes que d'éviter les trous de 10 h.

IV. Conclusion

Durant l'analyse de l'existant nous avons pu recenser toutes les informations nécessaires et indispensables à notre projet à savoir la conception et la réalisation d'un système de génération automatique et gestion des emplois du temps pour le département d'informatique.

Ces informations tirées entre autre à partir de l'étude du système pédagogique universitaire nous ont permis de cerner les principaux objectifs et de choisir techniques à adopter.

Chapitre II

I. Introduction

Un grand nombre de problèmes de planification sont des problèmes d'optimisation combinatoire et d'affectation sous contrainte, voici quelques domaines d'application :

- Les systèmes industriels de production: résolution conjointe des problèmes d'ordonnancement et de planification des moyens de transformation et de transport.
- Les systèmes informatiques: parallélisation de programmes informatiques, gestion des applications temps réel.
- Les systèmes administratifs: planification de soins dans le domaine de la santé.
- Les systèmes de transport: problèmes de tournées, d'habillage dans les réseaux de transport en commun.

Etant donné un ensemble d'objets et de ressources il s'agit d'affecter chaque objet à une ressource en satisfaisant un ensemble de contraintes et en optimisant une fonction objectif donné en mesurant la qualité de l'affectation. La gestion des emplois du temps consiste à affecter un ensemble d'objet à un ensemble de ressource dans l'espace de temps en satisfaisant un ensemble de contraintes. Dans ce chapitre nous traiterons le problème d'affectation d'un point de vue **R**echerche **O**pérationnelle (RO) et **I**ntelligence **A**rtificielle (IA), en détaillant quelques méthodes de résolution et en comparant entre les deux approches.

II. Problème d'affectation sous contrainte

La définition générale de l'optimisation ne précise ni la forme des solutions de *S*, ni la façon de générer ces solutions (admissibles ou non admissibles). Nous nous intéressons en particulier à une classe importante de problèmes dont une solution peut être décrite explicitement par une *affectation* de valeurs à l'ensemble des variables du problème. Etant donné un ensemble fini V = {V1,..., Vn} de variables et un ensemble D = {D1,..., Dn} de domaines finis associés, une solution potentielle du problème (affectation) consiste à choisir pour chaque variable Vi ($1 \leq i \leq n$) une valeur choisie dans son domaine Di. L'ensemble S des solutions potentielles est donc représenté par le produit cartésien $D1 \times D2... \times Dn$ des domaines. On dispose en outre d'un ensemble C = {C1,..., Cp} de contraintes, où chaque contrainte Cj ($1 \leq j \leq p$) est une relation sur un sous ensemble V'j de V qui spécifie quelles combinaisons de valeurs sont compatibles pour les variables de V'j. **[BEN 08]**

Par la suite, nous utilisons le terme informel de « problèmes d'affectation sous contraintes » (PASC) pour qualifier la classe des problèmes d'affectation utilisant la notion de Contrainte. Cette classe de problèmes inclut notamment les problèmes de satisfaction de

contraintes, les problèmes de satisfaction partielles (maximales) (PSCM) et les problèmes d'optimisation sous contraintes (POSC). Etant donné un triplet <V, D, C>, le problème CSP consiste à trouver une assignation qui satisfait toutes les contraintes et le problème MCSP une assignation qui satisfait un nombre maximum de contraintes. Etant donné un quadruplé <V, D, C, f>, le problème POSC consiste à trouver une assignation qui satisfait toutes les contraintes et qui minimise la fonction f de S vers R. **[BEN 08]**

Les problèmes d'affectation sous contraintes possèdent de très nombreuses applications pratiques concernant l'affectation de ressources, le groupement, la classification, la planification, l'emploi du temps et l'ordonnancement, dans des domaines très variés. Les PASC permettent également de modéliser facilement des problèmes de référence comme par exemple la k-coloration et la satisfiabilité.

II. 1. Complexité

Un problème est dit polynomial s'il existe un algorithme permettant de trouver une solution optimale pour toutes ses instances en un temps polynomial par rapport à la taille de l'instance. Un tel algorithme est dit efficace pour le problème en question. C'est notamment le cas de certains problèmes de plus court chemin dans un graphe valué, du recouvrement d'un graphe valué par un arbre de poids minimum, et des problèmes classiques de flots.

Cependant, pour la majorité des problèmes d'optimisation combinatoire, aucun algorithme polynomial n'est connu actuellement **[RM10]**. La difficulté intrinsèque de ces problèmes est bien caractérisée par la théorie de la NP-complétude. De nombreux problèmes d'optimisation combinatoire (la plupart de ceux qui sont vraiment intéressants dans les applications) ont été prouvés NP-difficiles. Cette difficulté n'est pas seulement théorique et se confirme hélas dans la pratique. Il arrive que des algorithmes exacts de complexité exponentielle se comportent efficacement face à de très grosses instances et pour certains problèmes et certaines classes d'instances. Mais c'est très souvent l'inverse qui se produit ; pour de nombreux problèmes, les meilleures méthodes exactes peuvent être mises en échec par des instances de taille modeste, parfois à partir de quelques dizaines de variables seulement. A titre d'exemple, on ne connaît aucune méthode exacte qui soit capable de colorier de façon optimale un graphe aléatoire de densité 1/2 lorsque le nombre de sommets dépasse 90. Néanmoins pour le problème d'affectation de fréquences dans le domaine des réseaux radiomobiles qui est une extension du problème de coloration, nous avons eu à traiter des instances comportant plus d'un millier de variables. Pour traiter les grosses instances de ce

type de problèmes, on se contente de solutions approchées obtenues avec les méthodes heuristiques. **[BEN 08]**

II. 2. Formulation des problèmes d'affectation sous contrainte

Etant donné un ensemble d'objets et de ressources, le problème consiste à affecter les objets aux ressources tout en satisfaisant un ensemble de contraintes et en minimisant une fonction objective donnée. Le formulation de ce problème est présentée comme suit:

Min F(s)
sous les contraintes: **C1, C2,…, Cn**

III. Méthodes de résolution

Un très grand nombre de méthodes de résolution existent en **RO** et en **IA** pour l'optimisation combinatoire et l'affectation sous contraintes. La figure met en parallèle les méthodes représentatives développées en **RO** et en **IA**, avec à titre indicatif la date approximative d'apparition de chaque méthode.

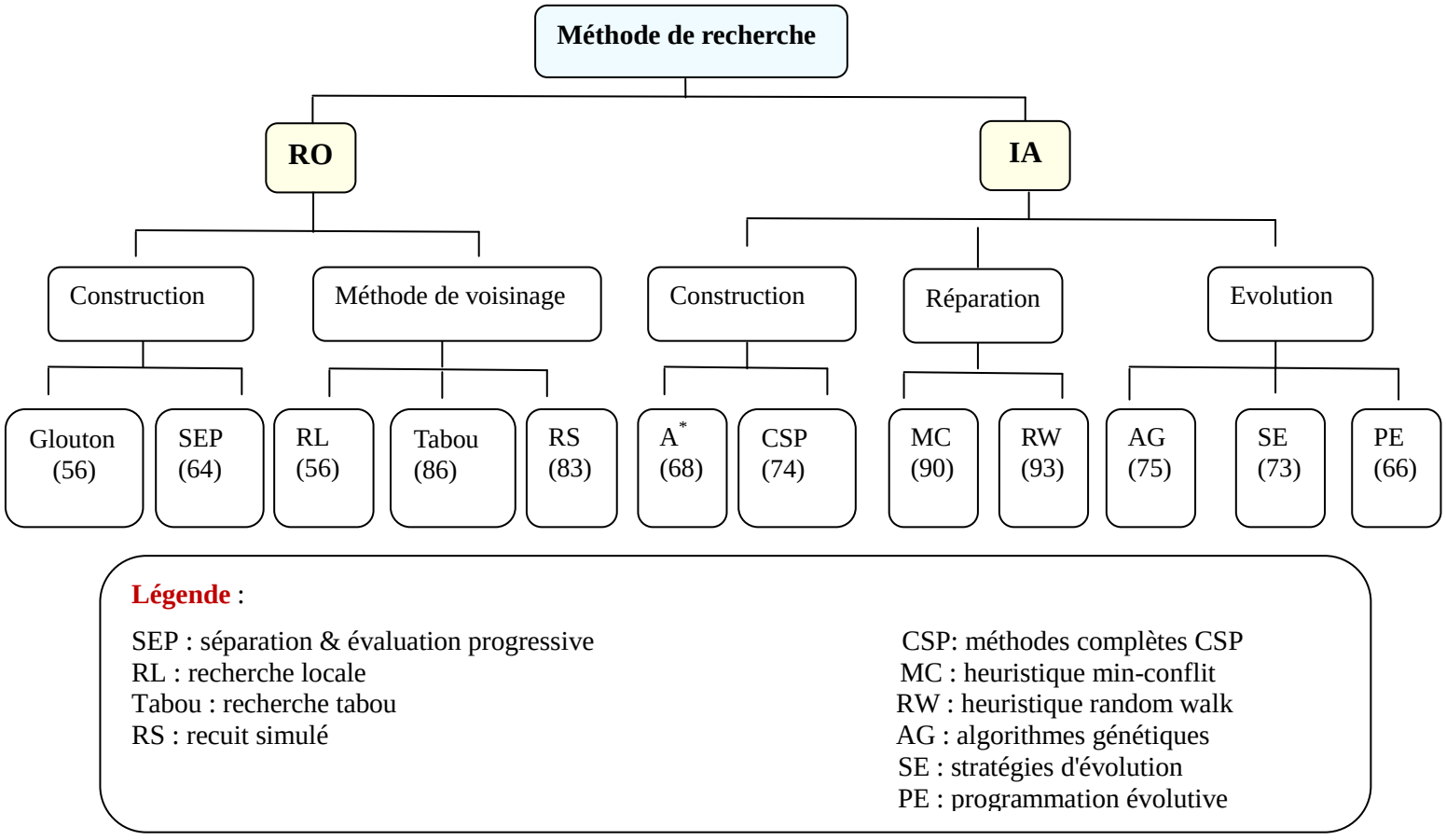

Figure II.1: Classement des méthodes de résolution en RO et IA

De manière générale, les méthodes de résolution suivent quatre approches différentes pour la recherche d'une solution : l'approche de construction, l'approche séquentielles, l'approche de voisinage et l'approche d'évolution. Ces méthodes font partie de deux groupes de nature différente.

- Le premier groupe comprend les méthodes exactes d'arborescence qui garantissent la complétude de la résolution : c'est le cas de SEP, A* et CSP. Le temps de calcul nécessaire d'une telle méthode augmente en général exponentiellement avec la taille du problème à résoudre (dans le pire des cas). Pour améliorer l'efficacité de la recherche, on utilise des techniques variées pour calculer des bornes permettant d'élaguer le plus tôt possible des branches conduisant à un échec. Parmi ces techniques, on peut citer les trois types de relaxations : la relaxation de base en programmation linéaire, la relaxation lagrangienne, la relaxation agrégée et la décomposition lagrangienne. De plus, on emploie des heuristiques pour guider les choix de variables et de valeurs durant l'exploration de l'arborescence.
- Le second groupe comprend les méthodes approchées dont le but est de trouver une solution de bonne qualité en un temps de calcul raisonnable sans garantir l'optimalité de la solution obtenue. Les méthodes approchées sont fondées principalement sur diverses heuristiques, souvent spécifiques à un type de problème.

Les techniques de relaxation permettent également de fournir des solutions approchées. Les métaheuristiques constituent une autre partie importante des méthodes approchées et ouvrent des voies très intéressantes en matière de conception de méthodes heuristiques pour l'optimisation combinatoire. Dans la suite on va citez quelques approches et méthodes de résolutions des problèmes d'optimisation combinatoire.

III. 1. Approches de construction

L'approche de construction est probablement la plus ancienne et occupe traditionnellement une place très importante en optimisation combinatoire et en intelligence artificielle. Une méthode de construction construit pas à pas une solution de la forme : **s = (<V1, v1> <V2, v2>...<Vn, vn>).** Partant d'une solution partielle initialement vide s = (), elle cherche à étendre à chaque étape la solution partielle **s = (<V1, v1>...<Vi-1, vi-1>) (i ≤ n)** de l'étape précédente. Pour cela, elle détermine la prochaine variable Vi, choisit une valeur vi dans **Di** et ajoute **<Vi, vi>** dans s pour obtenir une nouvelle solution partielle : **s = (<V1, v1>...<Vi-1, vi-1> <Vi, vi>).** Ce processus se répète jusqu'à ce que l'on obtienne une solution complète.

Trois types de construction sont proposés en littérature:

1- Un premier type de méthodes de construction est représenté par les méthodes gloutonnes. Une méthode gloutonne consiste à fixer à chaque étape la valeur d'une variable sans remettre en cause les choix effectués précédemment. Par exemple, une heuristique gloutonne bien connue pour la coloration introduite par Brélaz **[D.E89]** est la suivante : pour choisir le nœud suivant à colorier, prendre celui dont les nœuds adjacents sont déjà coloriés avec le plus grand nombre de couleurs différentes et lui assigner la couleur autorisée de plus petit rang possible. Les méthodes gloutonnes sont généralement rapides, mais fournissent le plus souvent des solutions de qualité médiocre. Elles ne garantissent l'optimum que dans des cas particuliers, par exemple la présence d'une structure de matroïde.

2- Un deuxième type de méthode de construction est représenté par les méthodes avec retour arrière. Une méthode de retour arrière avec une stratégie de recherche en profondeur d'abord consiste à fixer à chaque étape la valeur d'une variable. Aussitôt qu'un échec est détecté, un retour arrière est effectué, i.e., une ou plusieurs instanciations déjà effectuées sont annulées et de nouvelles valeurs recherchées. Par exemple, un algorithme typique avec retour arrière pour la résolution d'un problème de satisfaction de contraintes cherche à prolonger à chaque étape l'assignation courante de manière consistante. En cas d'échec, un retour arrière est effectué sur la dernière variable instanciée possédant encore des valeurs non essayées. Les méthodes avec retour arrière sont en général complètes et de complexité exponentielle. Pour réduire le nombre de retour arrière (et le temps de recherche), on utilise des techniques de filtrage afin d'anticiper le plus tôt possible les échecs.

3- Un troisième type de méthode de construction concerne de nombreux algorithmes basés sur le principe de séparation et évaluation progressive. Un exemple typique est l'algorithme A* avec une stratégie « meilleur d'abord » pour la recherche d'un plus court chemin dans un graphe valué. Cet algorithme débute avec un nœud initial et prolonge ensuite parmi l'ensemble de chemins partiels développés le chemin dont la longueur est la plus faible, en utilisant une estimation de la longueur restante pour calculer la borne inférieure. Lorsqu'un chemin complet est trouvé, les chemins partiels dont la longueur est supérieure à celle du chemin complet sont éliminés. De manière générale, on utilise des techniques de relaxations pour obtenir des bornes aussi serrées que possible (un exemple simple consiste à relâcher la contrainte d'intégrité pour appliquer un algorithme de simplexe).

III. 2. Approche séquentielle

Les méthodes séquentielles (de voisinage) sont des algorithmes de recherche itératifs qui explorent l'espace **S** en se déplaçant pas à pas d'une solution à une autre. Deux solution S et S'sont dites voisines si l'on peut obtenir s'en modifiant **S** selon une règle bien définit. Nous dirons que le passage de **S** a **S'** se fait par mouvement. Le voisinage **V(s)** d'une solution s appartient à **S** est l'ensemble des solutions voisines de **s**.

Voici quelques méthodes séquentielles :

III. 2 .1. La méthode descente

La recherche locale, appelée aussi la descente ou l'amélioration itérative, représente une classe de méthodes heuristiques très anciennes. Traditionnellement, la recherche locale constitue une arme redoutable pour attaquer des problèmes réputés très difficiles tels que le voyageur de commerce et la partition de graphes. Contrairement à l'approche de construction, la recherche locale manipule des configurations complètes durant la recherche. Une méthode de recherche locale est un processus itératif fondé sur deux éléments essentiels : un voisinage $N : X \rightarrow 2X$ et une procédure exploitant le voisinage. Plus précisément, elle consiste à :

1- débuter avec une configuration quelconque ***s*** de ***X***,
2- choisir un voisin ***s'*** de *s* tel que $f(s') < f(s)$ et remplacer ***s*** par ***s'***
3- répéter (2) pour tout voisin ***s'*** de ***s***, jusqu'à ce que $f(s') \geq f(s)$.

Cette procédure fait intervenir à chaque itération le choix d'un voisin qui améliore la configuration courante. Plusieurs possibilités peuvent être envisagées pour effectuer ce choix. Il est possible d'énumérer les voisins jusqu'à ce qu'on en découvre un qui améliore strictement (première amélioration). On peut également rechercher le meilleur voisin (meilleure amélioration). Cette dernière solution peut sembler plus coûteuse, mais le voisin découvert sera en général de meilleure qualité. De plus, l'utilisation d'une structure de données appropriée peut souvent permettre de trouver directement ce meilleur voisin. Comme l'espace des solutions ***X*** est fini, cette procédure de descente s'arrête toujours, et la dernière configuration trouvée ne possède pas de voisin strictement meilleur qu'elle-même. Autrement dit, la recherche locale retourne toujours un optimum local. L'avantage principal de cette méthode réside dans sa grande simplicité et sa rapidité. Mais les solutions produites sont souvent de qualité médiocre et de coût très supérieur au coût optimal. Pour remédier à ce

problème, la solution la plus simple est la *méthode de relance aléatoire* qui consiste à générer une nouvelle configuration de départ de façon aléatoire et à recommencer une descente.

Avantages et inconvénients

En général, l'efficacité des méthodes de recherche locale simples (descente, ou plus grande descente) est très peu satisfaisante. D'abord, par définition, la recherche s'arrête au premier minimum local rencontré, c'est là leur principal défaut. Pour améliorer les résultats, on peut lancer plusieurs fois l'algorithme en partant d'un jeu de solutions initiales différentes, mais la performance de cette technique décroît rapidement.

En revanche, autoriser de temps à autre une certaine dégradation des solutions trouvées, afin de mieux explorer tout l'espace des configurations, a conduit au développement des deux méthodes que nous explorons aux paragraphes suivants, à savoir le recuit simulé et la méthode Tabou.

Le principal avantage de la recherche locale simple est évidemment sa grande simplicité de mise en œuvre: la plupart du temps, elle ne fait que calculer f(s+i)-f(s), où i correspond à un déplacement élémentaire, et si cette expression peut se simplifier algébriquement, alors on pourra évaluer très rapidement cette différence.

Il est important de remarquer également l'importance du choix de la fonction de voisinage N : un minimum local pour une certaine structure de voisinage ne l'est pas forcément pour une autre. C'est d'ailleurs ce constat qui est à l'origine de la méthode dite de recherche par voisinage variable, qui repose sur la construction de solutions s parmi plusieurs voisinages Ni, plutôt que dans un seul.

III. 2 .2. Le recuit simulé

La méthode de recuit simulé **(RS)** s'inspire du processus de recuit physique. Les origines du recuit simulé remontent aux expériences réalisées dans les années 50 par Metropolis et al. **[JPM 99]** pour simuler l'évolution d'un tel processus de recuit physique. Metropolis et al ont utilisé une méthode stochastique pour générer une suite d'états successifs du système en partant d'un état initial donné. Tout nouvel état est obtenu en faisant subir un déplacement (une perturbation) aléatoire à un atome quelconque. Soit ΔE la différence d'énergie occasionnée par une telle perturbation. Le nouvel état est accepté si l'énergie du système diminue **($\Delta E \leq 0$), s**inon, il est accepté avec une probabilité définie par :

$p(\Delta E, T) = \exp(-\Delta E / (Cb \times T))$ où T est la température du système et **Cb** une constante physique connue sous le nom de *constante de Boltzmann*.

A chaque étape, l'acceptation ou non d'un nouvel état dont l'énergie est supérieure à celle de l'état courant est déterminée de manière probabiliste : un réel θ est tiré aléatoirement de l'intervalle [0, 1] et il est ensuite comparé avec **$p(\Delta E, T)$.** Si $\theta \leq p(\Delta E, T)$ alors le nouvel état est accepté pour remplacer l'état courant, sinon l'état courant est maintenu.

Après un grand nombre de perturbations, un tel processus fait évoluer le système vers un état d'équilibre thermodynamique selon la *distribution de Boltzmann* qui est définie par la probabilité de se trouver dans un état d'énergie **E** suivante:

$Pr(E) = c(T) \times \exp(-E/(Cb \times T))$ où **c(T)** est un facteur de normalisation.

Un tel processus du RS est appelé pour résoudre des problèmes d'optimisation combinatoire. Le RS peut être vu comme une version étendue de la méthode de descente. Le processus du recuit simulé répète une procédure itérative qui cherche des configurations de coût plus faible tout en acceptant de manière contrôlée des configurations qui dégradent la fonction de coût. A chaque nouvelle itération, un voisin $s' \in N(s)$ de la configuration courante *s* est généré de manière aléatoire. Selon les cas, ce voisin sera soit retenu pour remplacer celle-ci, soit rejeté. Si ce voisin est de performance supérieure ou égale à celle de la configuration courante, *i.e.*, $f(s') \leq f(s)$, il est systématiquement retenu. Dans le cas contraire, ***s'*** est pris avec une probabilité **$p(\Delta f, T)$** qui dépend de deux facteurs : d'une part l'importance de la dégradation $\Delta f = f(s') - f(s)$ (les dégradations plus faibles sont plus facilement acceptées), et d'autre part un paramètre de contrôle **T**, la température (une température élevée correspond à une probabilité plus grande d'accepter des dégradations). La température est contrôlée par une fonction décroissante qui définit un schéma de refroidissement. Les deux paramètres de la méthode définissent la longueur des paliers et la fonction permettant de calculer la suite décroissante des températures. En pratique, l'algorithme s'arrête et retourne la meilleure configuration trouvée lorsqu'aucune configuration voisine n'a été acceptée pendant un certain nombre d'itérations à une température ou lorsque la température atteint la valeur zéro.

Avantages et inconvénients

Le recuit simulé présente l'avantage d'offrir des solutions de bonne qualité, tout en restant simple à programmer et à paramétrer. Il offre autant de souplesse d'emploi que

l'algorithme de recherche local classique : on peut inclure facilement des contraintes dans le corps du programme. Aarts, Korst et Laarhoven (1997) ont par ailleurs démontré que, sous certaines conditions de décroissance de la température, l'algorithme du recuit simulé converge en probabilité vers un optimum global lorsque le nombre d'itérations tend vers l'infini.

L'un des inconvénients du recuit simulé est qu'une fois l'algorithme piégé à basse température dans un minimum local, il lui est impossible de s'en sortir tout seul.

Plusieurs solutions ont été proposées pour tenter de résoudre ce problème, par exemple en acceptant une brusque remontée de la température, de temps en temps, pour relancer la recherche sur d'autres régions plus éloignées. Il est également possible d'empêcher la température de descendre trop bas : on lui donne une valeur minimale au delà de laquelle on ne change plus de palier de température. Mais si cette valeur est trop grande, l'algorithme passera son temps à augmenter et diminuer son énergie car il acceptera trop de perturbations dégradantes et il n'arrivera pas à explorer à fond une vallée.

Ainsi, il est fort possible que l'algorithme arrive à "trouver" la vallée dans laquelle se cache un minimum global, mais il aura beaucoup de mal à l'explorer et donc risque de s'en éloigner sans avoir décelé la solution au problème…

C'est là le principal problème du recuit : le paramétrage de la température peut être rebutant et très empirique. On constate également qu'il faut une diminution de la température « suffisamment lente », donc un certain nombre d'itérations, pour voir la solution s'améliorer. Corrélativement, le temps de calcul devient parfois excessif avec certaines applications, et l'algorithme exige en particulier le calcul d'une exponentielle.

III. 2 .3. Les algorithmes d'acceptation avec seuil

Ces algorithmes sont des variantes du recuit simulé, la différence entre eux se situe au niveau de l'acceptation de dégradation à chaque étape. Dans le RS, cette décision est prise selon le critère de Metropolis. Dans un algorithme d'acceptation avec seuil, une telle décision est prise de manière déterministe. A chaque itération ***k***, l'acceptation d'un voisin $s' \in N(s)$ se base uniquement sur une fonction auxiliaire **r(s',s)** et un seuil **Tk.** A cet effet, ***s'*** est accepté si **r(s',s) <Tk**. La fonction **r(s',s)** et le seuil **Tk** peuvent être définis de nombreuses manières. Ainsi, on peut définir la fonction **r(s',s)** par $\Delta f = f(s') - f(s)$, et le paramètre de seuil **Tk** de manière analogue à la température **T** du RS. En effet, le seuil **Tk** est initialisé à une valeur élevée puis décroît progressivement à chaque fois qu'un nombre prédéterminé d'itérations est effectué. Les seuils ainsi générés correspondent à une suite de valeurs positives décroissantes

T1 ≥T2 ≥ ... ≥ Tk-1 ≥ Tk ≥ 0 et donc **Tk** tends vers 0. L'idée est de diminuer petit à petit la chance d'accepter des configurations qui dégradent la fonction de coût. Quand **Tk** tend vers 0, l'algorithme réalise une recherche de descente aléatoire.

Ces algorithmes ont été utilisés pour résoudre des problèmes d'optimisation et ont obtenu des résultats intéressants. La difficulté essentielle de cette approche se situe au niveau de la détermination des seuils pour une application donnée.

III. 2 .4. La méthode Tabou

La méthode Tabou a été développée par Glover **[BEN 08]** et indépendamment par Hansen **[D.E89].** Cette méthode fait appel à un ensemble de règles et de mécanismes généraux pour guider la recherche de manière intelligente à travers l'espace des solutions. Contrairement au recuit simulé qui génère aléatoirement une seule solution voisine $s' \in N(s)$ à chaque itération, la méthode Tabou examine un échantillon de solutions de $N(s)$ et retient la meilleure s', même si s' est plus mauvaise que s.

La recherche Tabou ne s'arrête donc pas au premier optimum trouvé, mais elle peut entraîner des cycles, par exemple un cycle de longueur 2 : $s \rightarrow s' \rightarrow s \rightarrow s' ...$ etc. Pour empêcher ce type de cycle, on mémorise les *k* dernières configurations visitées dans une mémoire à court terme et on interdit tout mouvement qui conduit à une de ces configurations. Cette mémoire est appelée *la liste tabou* qui est une des composantes essentielles de cette méthode. Elle permet d'éviter tous les cycles de longueur inférieure ou égale à **k**. La valeur de **k** dépend du problème à résoudre et peut éventuellement évoluer au cours de la recherche.

Avantages et inconvénients

La méthode Tabou est une méthode de recherche locale, et la structure de son algorithme de base est finalement assez proche de celle du recuit simulé, donc on passera donc de l'un à l'autre facilement, avec l'avantage, par rapport au recuit simulé, d'avoir un paramétrage simplifié : dans un première temps, le paramétrage consistera d'abord à trouver une valeur indicative t d'itérations pendant lesquelles les mouvements sont interdits. Il faudra également décider d'une stratégie de mémorisation à long terme – sur la qualité des solutions, sur leur récence, ou sur leur qualité…

En revanche, la méthode Tabou exige une gestion de la mémoire de plus en plus lourde à mesure que l'on voudra raffiner le procédé en mettant en place des stratégies de mémorisation

complexe.

L'efficacité de la méthode Tabou fait qu'elle est largement employée dans les problèmes d'optimisation combinatoire : elle a été testée avec succès sur les grands problèmes classiques (voyageur de commerce, ordonnancement d'ateliers) et elle est fréquemment appliquée sur les problèmes de constitution de planning, de routage, d'exploration géologique, etc.

III. 3. La méthode évolutive

Le terme « algorithmes évolutifs » englobe une autre classe assez large de métaheuristiques. Ces algorithmes sont basés sur le principe du processus d'évolution naturelle. Les algorithmes évolutifs doivent leur nom à l'analogie avec les mécanismes d'évolution des espèces vivantes. Un algorithme évolutif typique est composé de trois éléments essentiels :

1- *une population* constituée de plusieurs individus représentant des solutions potentielles (configurations) du problème donné;
2- *un mécanisme d'évaluation de l'adaptation* de chaque individu de la population à l'égard de son environnement extérieur;
3- *un mécanisme d'évolution* composé d'opérateurs permettant d'éliminer certains individus et de produire de nouveaux individus à partir des individus sélectionnés.

Du point de vue opérationnel, un algorithme évolutif typique débute avec une population initiale souvent générée aléatoirement et répète ensuite un cycle d'évolution composé de trois étapes séquentielles :

1- *mesurer l'adaptation* (la qualité) de chaque individu de la population par le mécanisme d'évaluation,
2- *sélectionner une partie des individus*,
3- *produire de nouveaux individus par des recombinaisons d'individus sélectionnés*. Ce processus se termine quand la condition d'arrêt est vérifiée, par exemple, quand un nombre maximum de cycles (générations) ou un nombre maximum d'évaluations est atteint. Par analogie à l'évolution naturelle, la qualité des individus de la population devrait tendre à s'améliorer au fur et à mesure du processus.

Les algorithmes évolutifs ont été appliqués à de très nombreux problèmes complexes et difficiles, y compris des problèmes d'optimisation (continue ou combinatoire).

D'une manière générale, on peut distinguer trois grandes familles d'algorithmes évolutifs: les algorithmes génétiques, la programmation évolutive et les stratégies d'évolution. Ces méthodes se différencient par leur manière de représenter les données et par leur façon de faire évoluer la population d'une génération à l'autre. **[JP99]**

III. 4. Les Méthodes hybrides

Le mode d'hybridation qui semble le plus fécond concerne la combinaison entre les méthodes de voisinage et l'approche d'évolution. L'idée essentielle de cette hybridation consiste à exploiter pleinement la puissance de recherche de méthodes de voisinage et de recombinaison des algorithmes évolutifs sur une population de solutions. Un tel algorithme utilise une ou plusieurs méthodes de voisinage sur les individus de la population pendant un certain nombre d'itérations ou jusqu'à la découverte d'un ensemble d'optimum locaux et invoque ensuite un mécanisme de recombinaison pour créer de nouveaux individus. La recombinaison doit impérativement être adaptée au problème traité.

Les algorithmes hybrides sont sans doute parmi les méthodes les plus puissantes. Malheureusement, le temps de calcul nécessaire peut devenir prohibitif à cause du nombre d'individus manipulés dans la population. A cet effet, une solution pour ce problème est la parallélisation de ces algorithmes sur des machines parallèles ou sur des systèmes distribués.
Nous venons d'évoquer l'hybridation d'un algorithme évolutionniste avec des méthodes de voisinage. Il est également possible d'hybrider d'autres types d'approches, par exemple une heuristique gloutonne et une méthode de voisinage ou un algorithme évolutif.

IV. Conclusion

Les méthodes approchées constituent une classe des méthodes adaptable à très grand nombre de problèmes d'optimisation combinatoire et d'affectation sous contraintes. Elles permettent de trouver des solutions de bonne qualité en un temps d'exécution limité pour des problèmes réel de grande taille. Pour cette raison l'étude de ces méthodes est actuellement en plein de développement. Mais ces méthodes sont basé sur un algorithme qui ne laisse pas l'opportunité pour la programmation orienté intelligence distribué. Les méthodes développées par la recherche opérationnelle sont basé sur un algorithme, ce dernier ne change pas de stratégie durant la résolution, cela ne permet en aucun doute l'évolution de la méthode face aux changements d'environnement.

Chapitre III

I. Introduction

Dans les projets de gestion d'emploi du temps, l'optimisation de solution est un réel problème du fait que la recherche d'une solution est complexe car il s'agit d'un problème de résolution de contraintes.

L'objectif principal de notre étude est de mettre en œuvre un logiciel qui rendra le processus de génération des emplois du temps plus efficace en termes d'implication d'utilisateurs, et qui est capable d'agir sur les changements des contraintes.

Dans ce chapitre nous avons fait la modélisation de l'application par création des diagrammes UML selon un ordre définie par le processus unifié et terminons par la présentation de l'application développée.

II. Outils et les langages utilisés

La réalisation d'une application génération d'emploi du temps se base sur l'utilisation de langages. Pour cet objectif nous avons utilisé PHP/MySQL.

II.1. Langage PHP

PHP est un langage de programmation comme le C ou Java, mais il est :

- beaucoup plus simple à apprendre et à utiliser (en particulier, pas de types),
- très adapté à la programmation Web (production de texte) et
- contient énormément de fonctions et de scripts prêts à l'emploi (disponibles sur le Web)
- très utilisé pour la réalisation de sites web.

On utilise PHP pour ces raisons suivantes :

- La gratuité et la disponibilité du code source (PHP est distribué sous licence GNU GPL).
- Sa richesse fonctionnelle : PHP comporte plus de 1000 fonctions.
- La simplicité d'écriture des scripts (?).
- La disponibilité sur le web de nombreux scripts PHP prêts à l'emploi.
- La possibilité d'inclure le script PHP au sein d'une page Html.
- La simplicité de liaison avec des bases de données.

II.2. Serveur web

Un serveur Web est un outil qui permet la publication d'information sur un réseau Internet et stocker les fichiers. Il permet aussi l'accès à une base de données pour fournir des informations issues à la demande de l'utilisateur. Nous avons utilisé le serveur Apache car il est intégrer dans Wamp.

III. Présentation de la base

Notre base de données est présentée sous Microsoft Access, elle est interrogée par SQL intégré dans le serveur WAMP.

La base regroupe les tables essentielles pour la gestion des emplois du temps, comme salles, cours, enseignants … etc.

IV. Processus de développement logiciel

Un processus définit une séquence d'étapes, en parties ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. [Rocques, 2007]. Nous avons utilisé le processus unifié (UP) qui est présenté par la suite :

IV.1. Processus Unifié

Le Processus Unifié (PU ou UP en anglais pour **Unified Processus**) est un processus de développement logiciel, il regroupe les activités à mener pour transformer les besoins d'un utilisateur en système logiciel. Les principales caractéristiques du processus unifié sont :

a) **Conduit par les cas d'utilisations**, ce qui veut dire que la conception est réalisée conformément aux attentes des utilisateurs du produit.

b) **Piloté par les risques,** dans le cadre, où les différentes sources d'échec du produit doivent être ecratées.une source importante d'échec d'un produit logiciel est son inadéquation aux attentes de l'utilisateur.

   La gestion du processus est organisée suivant les quartes phases suivantes : initialisation, élaboration, construction, transition.

c) **Centré sur l'architecture**, car il impose le respect des décisions d'architecture a chaque état de instruction du modèle.

d) **itératif et incrémentale,** au sens que la réalisation du produit se fait en plusieurs itérations où chaque itération aboutit à livrer une partie de produit (un module) et

l'ajoute aux différentes parties déjà construit (incrémentation) en gardant un produit homogène.

IV.2. Modélisation avec UML

UML (Unified Modeling Language) ce définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier, concevoir des solutions et communiquer des points de vue. UML unifie à la fois les notations est les concepts orientés objet. UML propose un ensemble de diagrammes, les plus importants seront présentés :

IV.2.1.Diagramme de cas d'utilisations

Un diagramme de cas d'utilisation permet de structurer les besoins des utilisateurs et les objectifs correspondant d'un système.il permet aussi d'identifier les possibilités d'interactions entre le système et les acteurs (intervenants extérieurs au système). La figure suivante montre notre diagramme de cas d'utilisation

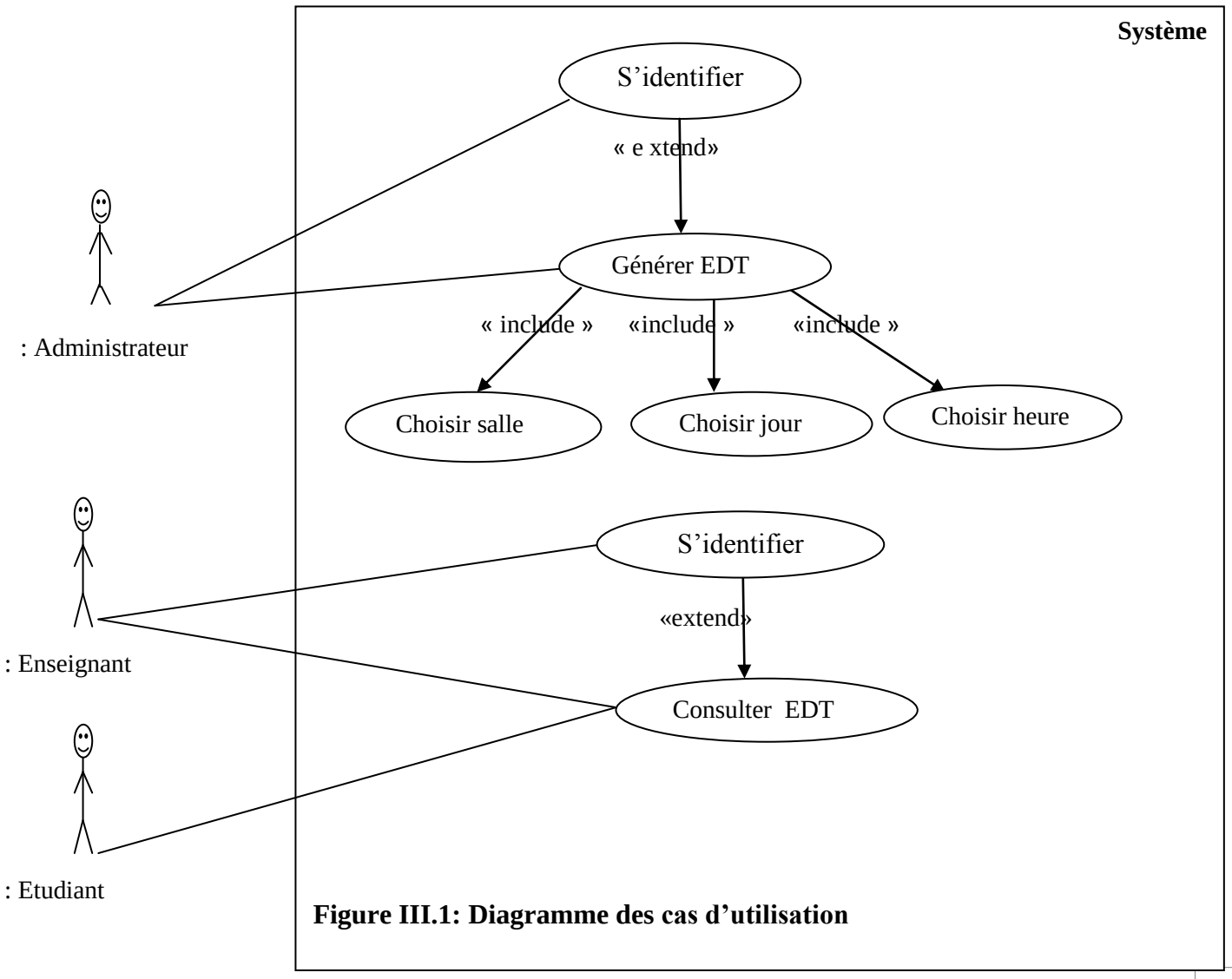

Figure III.1: Diagramme des cas d'utilisation

IV.2.2. Diagramme de séquence

Un diagramme de séquence permet de représenter des collaborations entre objets selon un point de vue temporel. Les objets communiquent en échangeant des messages représentés sous forme de flèches. Il peut servir à illustré un cas d'utilisation. Le diagramme de séquence de cas « génération EDT » est représenté dans la figure.

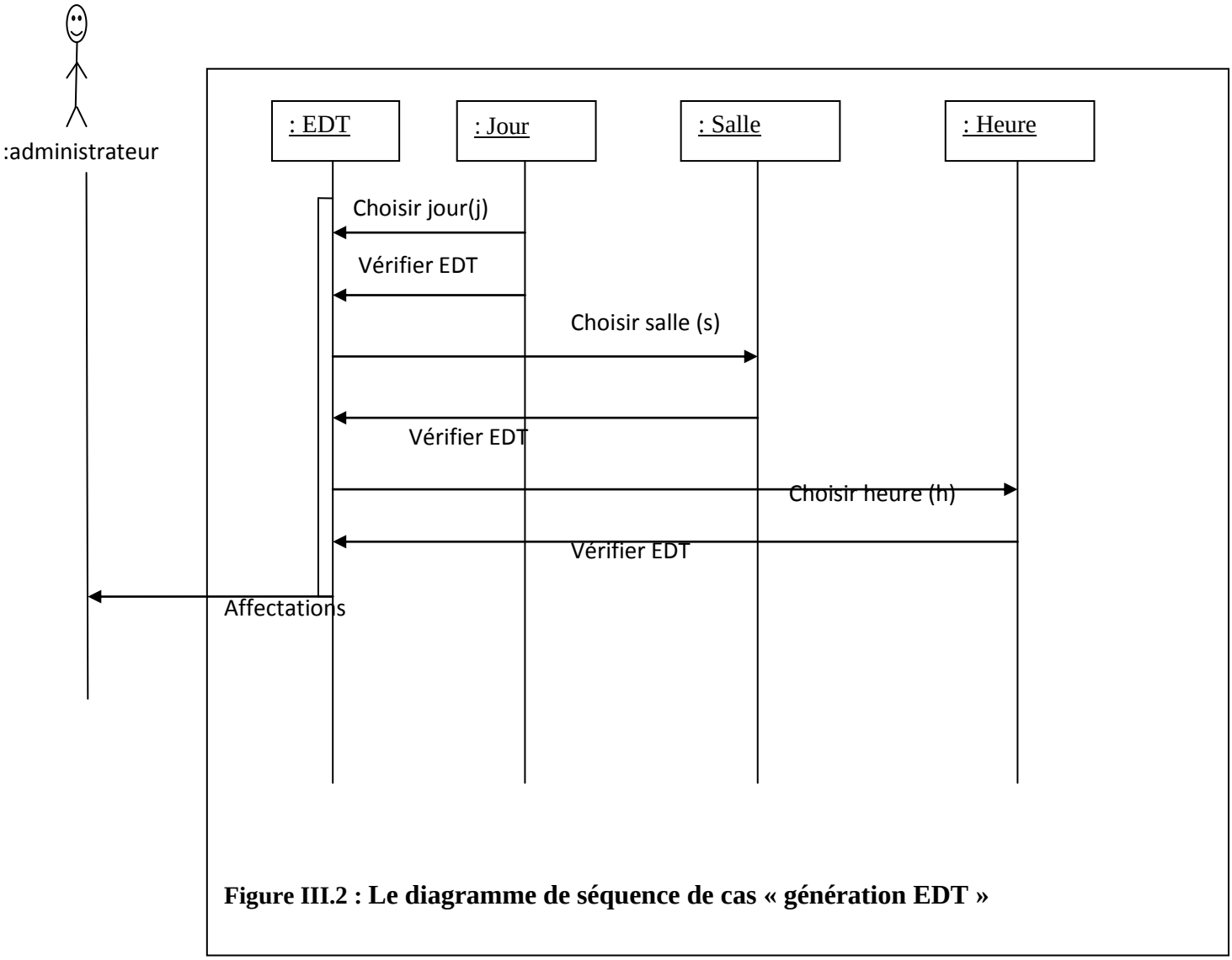

Figure III.2 : Le diagramme de séquence de cas « génération EDT »

IV.2.3. Diagramme de classes

Un diagramme de classes permet de représenter les classes intervenant dans le système.il constitue un élément très important de la modélisation et permettre de définir quelle seront les composantes du système final. Il permet de structuré le travail de développement de manière très efficace. La figure suivant représente le diagramme de classes de notre application.

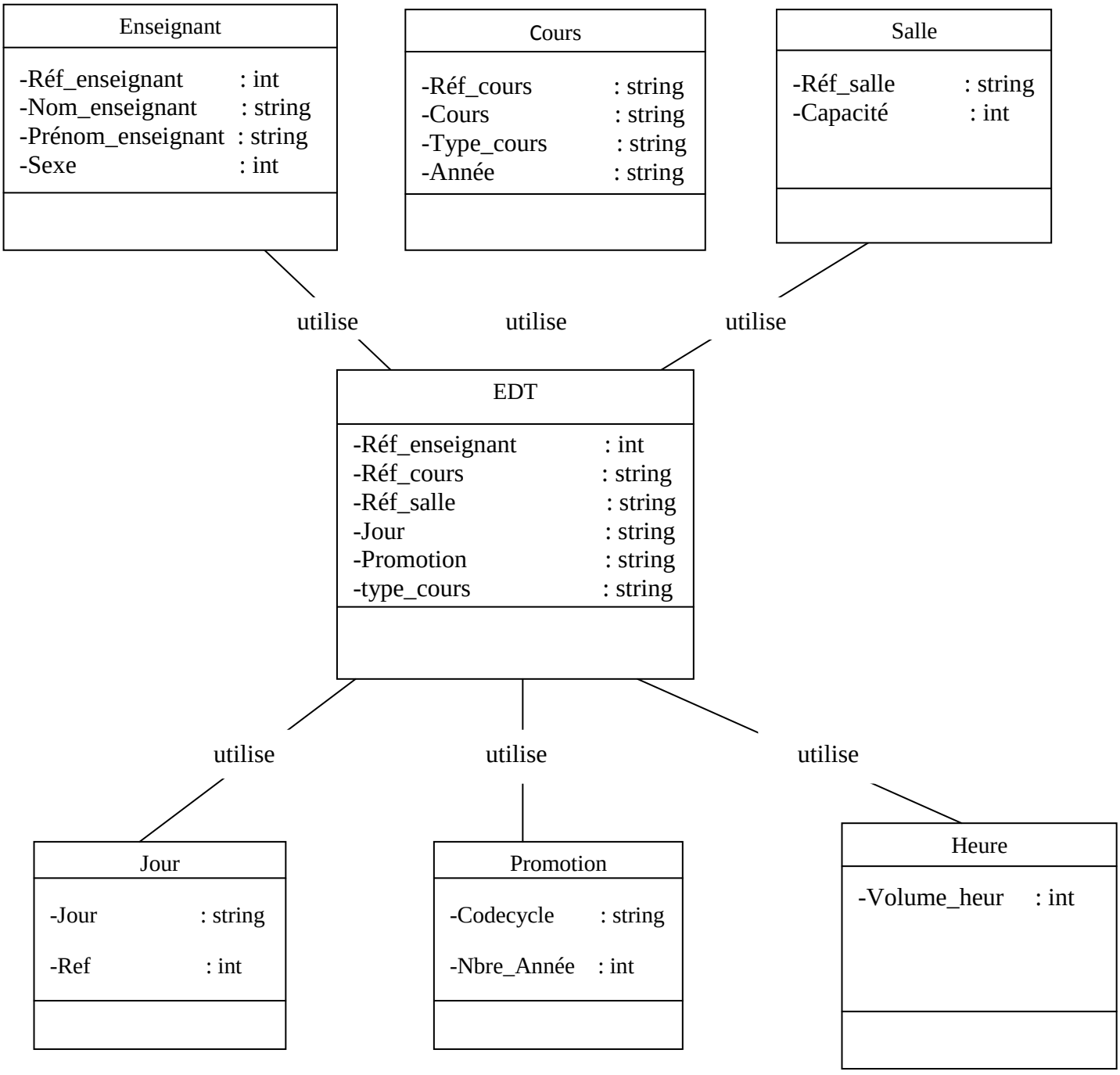

Figure III.3 : Diagramme de Classe

V. Présentation de l'IHM

L'interface homme/machine représente l'élément clé dans l'utilisation de tout système informatique.les interfaces de notre système sont conçues de manière à être simples, naturelles, compréhensible et d'utilisation faciles.

Fenêtre principale La fenêtre principale constitue une fenêtre d'un mot de passe pour accéder le programme principal :

- mot de passe pour l'administrateur qui accéder et il a privilège de générer l'EDT d'une promotion et afficher l'EDT d'un enseignant.

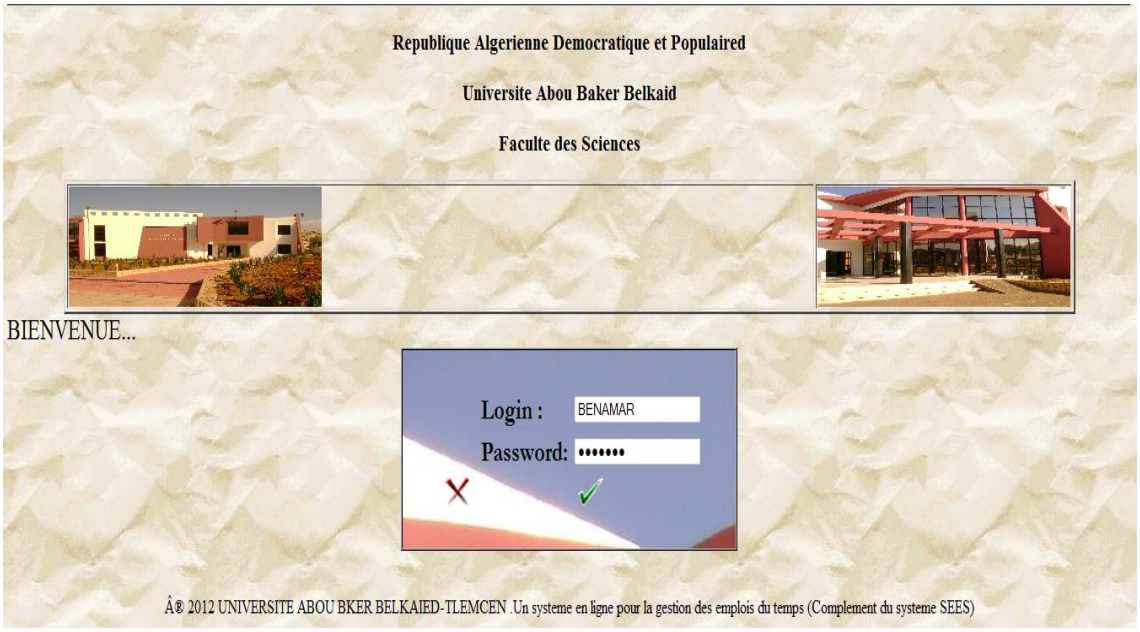

Figure III.4 : interface pour accéder

L'interface pour génér d'un EDT .

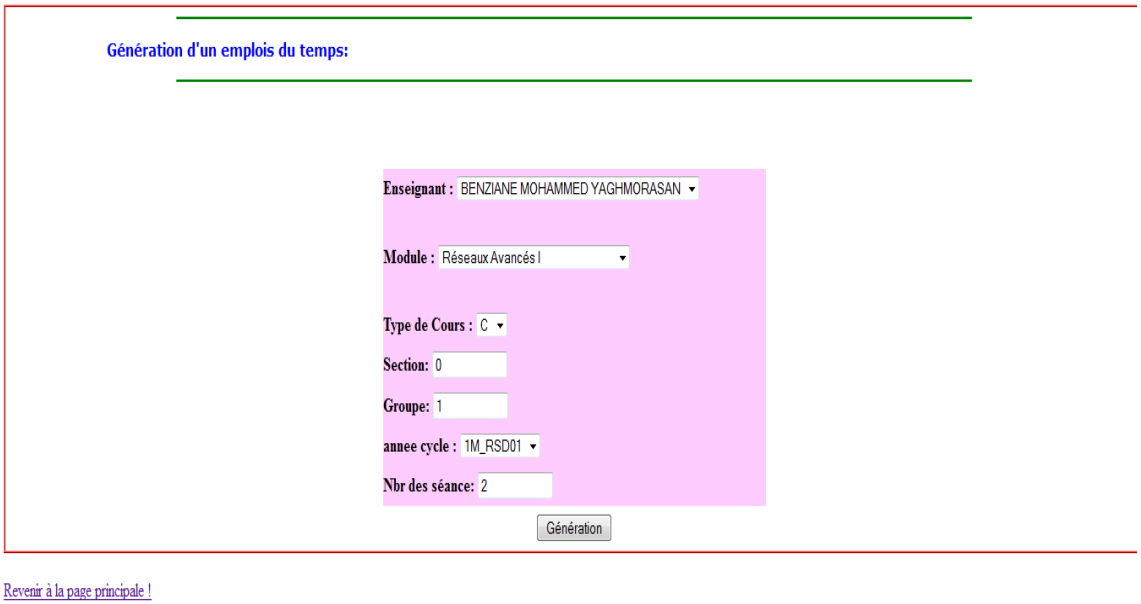

Figure III.5 : choisie les information pour générer l'EDT

La résultat d'affichage

l'emploi du temps : BENZIANE MOHAMMED YAGHMORASAN

	Dimanche	Lundi	Mardi	Mercredi	Jeudi
8h00					
9h30					
11h00					
12h30					
14h00		Réseaux Avancés I Cours 1M_RSD01 14h00 - 15h30		Réseaux Avancés I Cours 1M_RSD01 14h00 - 15h30	
15h30					
17h00					
18h30					

Figure III.6 : choisir le jour pour affecté la séance

L'interface pour afficher d'EDT d'un promotion

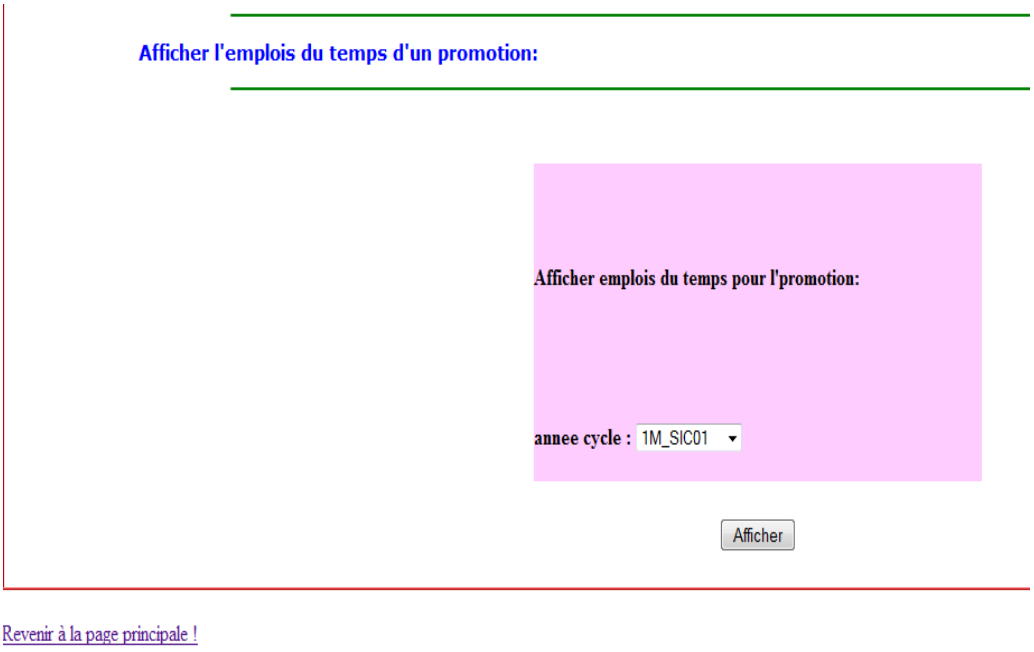

Figure III.7 : choisie la promotion pour afficher

Résultat d'affichage d'un promotion

1ere Master SIC

	Dimanche	Lundi	Mardi	Mercredi	Jeudi	Vendredi
8h00				Alg 1 S23 1M_SIC01 BELABED C 8h00 - 9h30		
9h30	M 173 S2 1M_SIC01 BELABED C 9h30 - 11h00	M202 S23 1M_SIC01 BELHABI D 9h30 - 11h00				
11h00		M 173 S18 1M_SIC01 BELABED D 11h00 - 12h00	M 173 S23 1M_SIC01 BELABED D 11h00 - 12h00			
12h30						
14h00			M 172 S18 1M_SIC01 BELABED C 14h00 - 15h30	M 173 S20 1M_SIC01 BELABED D 14h00 - 15h30		
15h30	Alg 1 S5 1M_SIC01 BELABED C 15h30 - 17h00	M 172 S1 1M_SIC01 BELABED C 15h30 - 17h00		M 172 S2 1M_SIC01 BELABED C 15h30 - 17h00		
17h00						

Figure III.8: EDT d'une promotion

- Mot de passe d'un enseignant qui accéder pour afficher l'EDT.

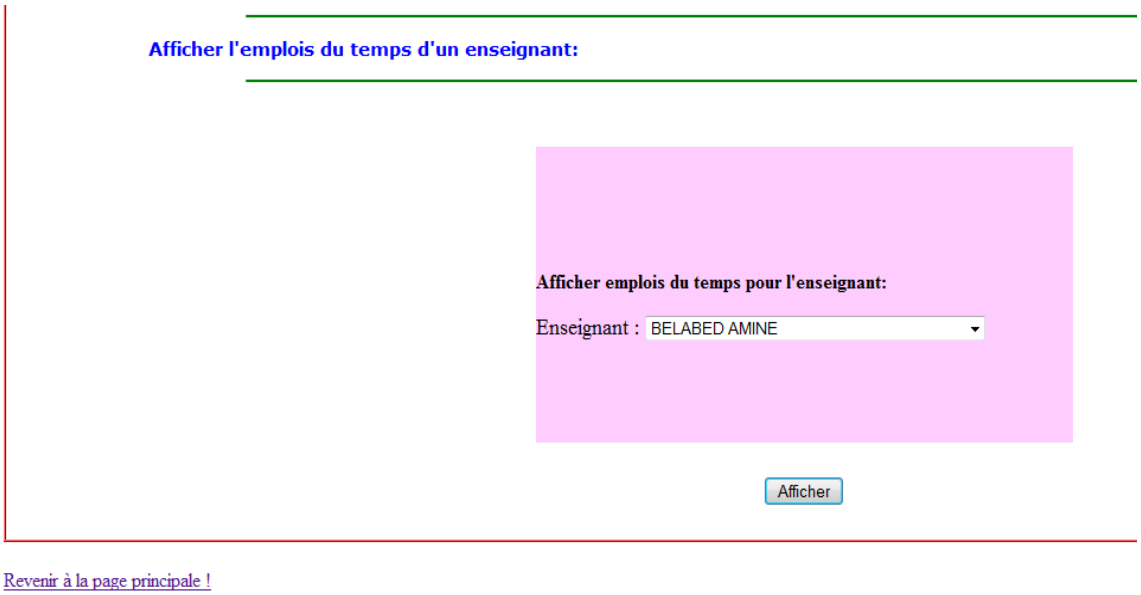

Figure III.9 : choisie l'enseignant pour afficher EDT

Le résultat d'EDT d'enseignant

Emploi du temps d'enseignant: BELABED AMINE

	Dimanche	Lundi	Mardi	Mercredi	Jeudi
8h00		Alg 1 S16 C 1M_MID01 8h00 - 9h30	Alg 1 S19 C 1M_MID01 8h00 - 9h30	Alg 1 S23 C 1M_SIC01 8h00 - 9h30	M 202 S18 C 2M_SIC014 8h00 - 9h30
9h30	M 173 S2 C 1M_SIC01 9h30 - 11h00	M 183 Amph.3 D 1SIC011 9h30 - 11h00		M 202 S19 C 2M_SIC014 9h30 - 11h00	
11h00		M 173 S18 D 1M_SIC01 11h00 - 12h00	M 173 S23 D 1M_SIC01 11h00 - 12h00		
12h30					
14h00			M 172 S18 C 1M_SIC01 14h00 - 15h30	M 173 S20 D 1M_SIC01 14h00 - 15h30	M 183 S23 D 1SIC011 14h00 - 15h30
15h30	Alg 1 S5 C 1M_SIC01 15h30 - 17h00	M 172 S1 C 1M_SIC01 15h30 - 17h00		M 172 S2 C 1M_SIC01 15h30 - 17h00	
17h00					
18h30					

Figure III.10 : Affichage résultat EDT d'un enseignant

VI. Conclusion

A la fin de la réalisation, le résultat sera une application qui fait la génération des emplois du temps des différentes promotions et enseignants sans aucun conflit au niveau des séances. Cette application a permis de répondre aux besoins des administratifs par la résolution des problèmes de la gestion des emplois du temps ainsi que les conflits qui peuvent exister.

Conclusion générale

Conclusion générale

Durant les derniers trois mois, nous avons analysé et réalisé un projet de fin d'étude relatif à l'information de la génération d'emplois du temps au sein du département MI de l'université Abou Bekr Belkaid Tlemcen (UABT).

Dans ce projet, nous avant exploité nos efforts et nos connaissances pour pouvoir bien analyser le sujet de la génération d'emplois du temps. Nous espérons que ce travail peut ramener un plus au département MI.

Ainsi, nous avons appris les différentes étapes de conception d'un système d'information en utilisant la méthode UML. D'une autre coté, la programmation web en utilisant les langages HTML et PHP, ainsi que la programmation des base des données Access avec SQL.

Perspectives

Nous souhaitons qu'on ait résolu une grande partie du problème de la génération d'emploi du temps, en désirants que nous aurons l'occasion de le développer et l'enrichir pour plus d'efficacité, selon les perspectives suivantes :

- L'extension du system aux autres départements et facultés;
- Utiliser les méthodes d'optimisations pour résolutions les problèmes génération d'EDT comme AG, Tabou, RS…etc.
- Injecter un mécanisme qui permet le traitement automatique et intelligent pour la génération de l'emploi du temps;
- Améliorer l'interface du système;
- Héberger le système dans le serveur web de l'université Abou Bekr Belkaid Tlemcen (UABT).

Références bibliographique

Référence

[BEN 08] Ben-delhoum soheib Hadji brahim **« Gestion des emplois du temps par les Systèmes Multi-Agents»**, Mémoire d'ingénieur, ESI (Ex INI) Alger, Promotion 2007/2008.

[CCM 06] «***Comment Ça Marche*** », Encyclopédie informatique libre, consulter le 07/09/2012 URL : http://www.commentcamarche.net/forum/index.php3

[COMM 04] URL: http://www.commentcamarche.net/cs/cs3tier.php3

[ZA&AA10] zouambia adel et amazouz assil « **Conception et Réalisation d'un système interactif d'aide à La gestion des emplois du temps »,** Mémoire d'ingénieur, ESI (Ex INI) Alger, Promotion 2009/2010.

[RM10] Rédha MILI « **Conception et implémentation d'un algorithme d'optimisation pour la Génération automatique d'emploi du temps Universitaire»,** Mémoire de Magister, Université 20 Août 1955 Skikda, Promotion 2010.

[D.E89] D.E.Goldberg « **Algorithmes Génétiques : Exploration, optimisation et apprentissage automatique** », Addison-Wesley France, 1989

[UML 02] Pascal Roques et Frank Vallée **« UML 2 EN ACTION»**, Quatrième tirage, 2002, EYROLLES.

[UML 03] UML 2 EN ACTION, Pascal Roques et Frank Vallée Premier tirage Juin 2004, EYROLLES JOUVE.

[JPM 99] Jin-Kao Hao, Philippe Galinier, Michel Habib **« Méthaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes»**, 1999.

[JP99] J. K. Hao, P. Galinier et M. Habib, « **Méthaheuristiques pour l'Optimisation Combinatoire et l'Affectation Sous Contraintes** », Revue d'Intelligence, vol. 13, no2, pp. 283-324, 1999

[SC04] S. Piechowiak, C. Kolski « **Analyse et conception d'un outil interactif d'aide à la gestion des emplois du temps basé sur les points de vue** », LAMIH - UMR CNRS 8530 – Université de Valenciennes et du Hainaut-Cambrésis 59313 Valenciennes Cédex 9– France, Revue d'Interaction Homme Machine (RIHM), volume 5, n° 2, pp. 1-30, 2004.

Annexes

Annexe 1: Logiciel S.E.E.S V2 (Suivi des Etudiants et des Enseignements Supérieurs)

1 – Présentation générale du système S.E.E.S

S.E.E.S : Suivi des Etudiants et des Enseignements Supérieurs en exploitation dans les établissements universitaires à l'échelle nationale depuis l'année universitaire 2007/2008.

Cette seconde version est apparue à la suite des changements de règles de gestion apportés par les arrêtés N°136 et N°137 du 20 juin 2009 abrogeant l'arrêté de référence du 23/01/2005 du MESRS.

Des perfections, des améliorations et des extensions ont été aussi apportées par rapport à la première version.

Son objectif principal est d'assurer la gestion et le suivi d'une année universitaire par

- Les inscriptions administratives des étudiants,

- L'articulation des unités d'enseignements et des matières par semestre et par année,

- Les différents traitements (saisie de notes, calcul de résultats, délibérations, etc…),

- L'édition de plus de soixante dix (70) états différents,

- Des éditions statistiques,

- Des analyses de synthèse à diverses échelles (département, faculté, établissement, conférences régionales ou conférence nationale)

- L'orientation intra et extra département,

- Le passage automatique à l'année universitaire suivante,

Le système S.E.E.S déborde sur les aspects purement pédagogiques en offrant des possibilités permettant une gestion optimale des ressources humaines, des prévisions financières et des locaux pédagogiques.

Le système S.E.E.S est totalement bilingue et il est présenté avec deux interfaces : en langue arabe pour les départements utilisant exclusivement la langue arabe et en langue française pour le département utilisant quotidiennement la langue française. Dès le lancement du système, une option est offerte à l'utilisateur de choisir son interface (arabe ou française).

Le système S.E.E.S peut être utilisé dans sa version monoposte ou en version réseau.

2 – Objectif de la formation

La formation a concerné deux groupes :

L'équipe de pilotage du système LMD constitué d'un responsable de la cellule LMD, d'un responsable administratif chargé de la scolarité et de la pédagogie, et d'un chef de département adjoint chargé de la pédagogie ayant été déjà confrontés aux difficultés quotidiennes du système LMD. L'objectif de la formation de ce groupe est d'abord l'initiation au système S.E.E.S mais surtout l'uniformisation de la compréhension et de l'interprétation du système LMD à l'échelle national. L'équipe de pilotage constituera la cellule de consultation pour toute question de compréhension et d'interprétation du système LMD.

L'équipe de formateurs constituée de trois chefs de département adjoints chargés de la pédagogie ayant été déjà confrontés à la gestion quotidienne du système LMD. L'objectif de la formation est la maîtrise du système S.E.E.S et d'assurer la formation des futurs utilisateurs au niveau de leurs établissements.

Annexe 2: PHP

Voici une liste des principales fonctions PHP :

Echo ( ): Elle écrit sur la sortie standard, laquelle est directement transmise au navigateur par le serveur web.

Date ( ) : affichage de la date courante récupère la date courante et la met en forme selon un format donné.

Empty ( ) : vérifie que les données ne sont pas vides.

isSet ( ) : utiliser pour tester l'existence d'une variable.

Include ( ) : instructions pour inclure du code dans un fichier père.

Require ( ) : joue le même rôle que Include ( ).

phpInfo () : Affiche des informations sur l'interpréteur PHP.

Print ( ) : Imprime une chaîne sur la sortie standard. Identique à echo ( ).

Print_r ( ) : Affiche tous les éléments d'un tableau.

session_start ( ) : Initialise les informations de session.

session_id ( ) : Renvoie l'identifiant de la session.

Annexe 3 : Les principales fonctions MySQL en PHP

odbc_close() : ferme la connexion Odbc.

odbc_connect() : ouvre une connexion à un serveur Odbc.

odbc_error() : renvoie l'erreur Odbc rencontrée.

odbc_fetch_array() : retourne une ligne de résultat sous la forme d'un tableau associatif.

odbc_fetch_assoc() : lit une ligne de résultat dans un tableau associatif.

odbc_fetch_row() : découpe une ligne de résultat en un tableau.

odbc_free_result() : libère la mémoire du résultat de la dernière requête.

odbc_num_rows() : renvoie le nombre d'enregistrements résultants d'une requête de type SELECT.

odbc_query() : envoie une requête SQL.

odbc_select_db() : sélectionne une base de données Odbc.

Annexe 4 : HTML

Voici une liste des principales balises HTML :

Mise en forme des caractères

<B>…<B>	Texte en gras.
<I>…</I>	Texte en italique.
<FONT color= "#xxxxx"> …</FONT>	Texte en couleur ou xxxxx est une valeur hexadécimale.
<FONT size=x>…</FONT>	Taille des caractéres ou x est une valeur de 1à7.
<PRE>…</PRE>	Texte préformante.

Mise en forme du texte

< !--…-- >	Commentaire ignoré par le navigateur.
<CENTER>…</CENTRE>	Centre tout éléments compris dans le tag.
<TR>…</TR>	Ligne du tableau.
<TD>…<TD>	Cellule du tableau.

<TD bgcolor="#xxxxxx">	Couleur d'une cellule de tableau.
<marquee> …</marque>	mobiliser un titre.

Fichier HTML

<HTML>…</HTML>	Début et fin de fichier HTML.
<HEAD>…</HEAD>	Zone d'en-tete d'un fichier HTML.
<TITLE>…</TITLE>	Titre affiché par le browser (élément de HEAD).
<BODY>…</BODY>	Début et fin du corps du fichier HTML.
<BODY bgcolor="#xxxxxx">	Couleur d'arrière-plan (en hexadécimal).
<BODY background="#xyz.gif">	Image d'arriére plan.

Listes

<UL>	Liste non numérotée (dite à puce).
<LI>…</LI>	Elément de liste.
<OL>…</OL>	Liste numérotée.

Ligne de séparation

<HR>	Trait horizontal (centré par défaut).
<HR width=x>	Largeur du trait en pixels.
<HR align=centre>	Trait centré (défaut).

Hyperliens

<A href=http: //...>…</A>	Lien vers une page web.

Images

<IMG src=" xyz.gif ">	Insertion d'une image en format Gif.
<IMG src="xyz.jpg">	Insertion d'une image en format JPG.
<IMG…border=x>	Définition de la bordure d'une image avec lien.
<IMG…align=bottom>	Aligne l'image en bas.
<IMG…align=middle>	Aligne l'image au milieu.
<IMG…awidth=x height=y>	Mise à l'échelle de l'image en pixels.

Tableau

<TABLE>…</TABLE>	Définition d'un tableau.
<TABLE width= "x%">	Largueur du tableau en %.
<TABLE width=x>	Largueur du tableau en pixels.
<TABLE border=x>	Largueur de la bordure.
<TABLE cellpadding=x>	Espace entre la bordure et le texte.

Résumé :

La génération automatique des emplois du temps universitaires consiste à allouer les ressources (enseignants, étudiants, créneaux horaires, salles) aux événements (séances de cours, TD et TP), tout en essayant de satisfaire le maximum de contraintes.

Les objectifs principaux atteints par le système que nous avons réalisé sont : la génération automatique des emplois du temps, et la gestion des différents conflits d'utilisation de ressources.

Mots Clés *:* emploi du temps, contraints, optimisation combinatoire, méta-heuristique.

Abstract:

Automatic generation of timetables is to allocate university resources (teachers, students, time slots, rooms) to events (courses sessions, TD and TP), while trying to satisfy as many constraints. The main objectives achieved by the system that we realized are: the automatic generation of timetables, and management of various conflicts of resource use.

Key words: timetables, constraints, combinatorial optimization, meta-heuristics.

الملخص

تعتبر جدولة الأحداث داخل أي هيئة من أهم تطبيقات البرمجة في الإعلام الآلي، خاصة إذا كانت الهيئة تتميز بكثرة الأحداث وتنوعها مثل الجامعة. ويتمحور هذا البحث على معالجة جدولة مواعيد الجامعة المختلفة، وهذا باعتبار هذه العملية كعملية تخصيص للموارد(الوقت، الغرف....) للأحداث و هي الدروس مع التقييد بمجموعة من القيود.

الأهداف الرئيسية التي حققها النظام الذي أدركنا هي: إنشاء أو تعديل أو توليد التلقائي للجداول زمنية، وإدارة الصراعات المختلفة في استخدام الموارد.

الكلمات المفتاحية: تخصيص الموارد، القيود، التحسين الاندماجي، الأحداث