



IUT Informatique – La Rochelle

Projet PTS2 - Gestionnaire d'emploi du temps

Cahier des charges fonctionnel

Michaël LUCAS
Benjamin VALOGNES
Romain DOUTEAU
Antoine FOUCHE

Tuteur : Mme LASSUS

Responsable : M. MARCHAND

I- Présentation du projet : EDT Project

A- Explication du projet : vue d'ensemble

B- Les fonctionnalités importantes

C- Analyse du besoin : problématique du client

D- Pour qui et pour quoi ?

E- Délimitation du projet

F- État de l'art

II- Architecture & Conception

A- Cas d'utilisations

B- Scénarios

C- Diagramme de conception

D- Diagramme de séquence

E- Wireframe

III- Organisation & Planning

A- Planification du projet : Excel

B- Communication : Slack

IV- Réalisation & Programmation

A- Modification de la conception (modèle du domaine)

B- Ajout de nouvelles fonctionnalités

C- Utilisation du GIT

D- Problèmes rencontrés en programmation

V- Bilan : rendu final

A- Fenêtre de connexion au gestionnaire d'emploi du temps

B- Fenêtre de saisie des groupes

C- Fenêtre de saisie des salles

D- Fenêtre de saisie des cours

E- Fenêtre d'affichage de l'emploi du temps

VI- Extensions possibles

I- Présentation du projet : EDT Project

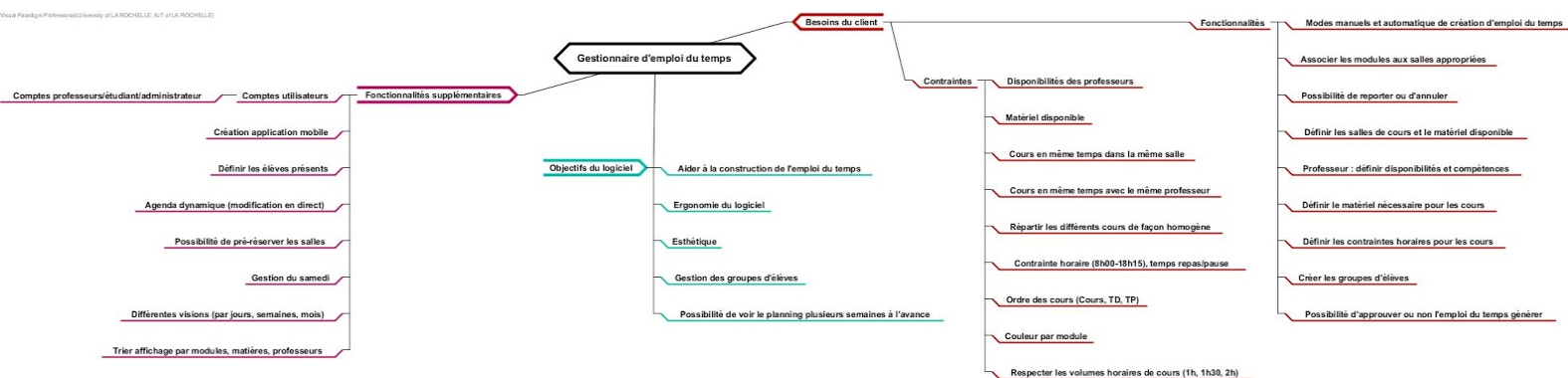
A- Explication du projet : vue d'ensemble

Le projet qui nous a été confié consiste en la création d'un programme permettant au directeur des études de l'IUT de créer l'emploi du temps avec plus de facilité, comparé à la solution logicielle proposée actuellement, qui repose sur plusieurs tableaux Excel et GPU.

B- Les fonctionnalités importantes

Le but est donc de créer un programme facilitant le travail du directeur des études dans la création et la mise en place des différents cours pour constituer l'emploi du temps, sur plusieurs semaines. Nous avons donc pu déterminer différentes fonctionnalités à intégrer au logiciel à concevoir, fonctionnalités que nous détaillerons dans les partie Conception.

Visual Paradigm Professional University of LA ROCHELLE, ANT (LA ROCHELLE)



Une des demandes importantes, sur laquelle nous avons été amené à nous concentrer est la partie affichage de l'emploi du temps, saisies des données associées à l'affichage (Cours, Salles, Professeurs, Heures) et l'affichage d'éventuelles contraintes lors du placement d'un cours dans l'emploi du temps.

C- Analyse du besoin : problématique du client

Afin de répondre correctement à la demande du client, il a d'abord fallu analyser la demande de manière précise, afin de définir les différentes fonctionnalités à implémenter au programme.

D- Pour qui et pour quoi ?

Nous avons d'abord commencé par reprendre la demande dans les grandes lignes afin d'en extraire les fonctionnalités importantes, que nous devions impérativement intégrer.

Nous avons donc commencé par définir différents acteurs : le logiciel doit s'adresser à la fois aux étudiants (en consultation), au directeur des études (pour consulter et créer l'emploi du temps) et aux professeurs afin qu'ils puissent indiquer leurs contraintes et consulter leur emploi du temps respectif.

Nous avons dans le même temps définis des fonctionnalités connexes :

- possibilité de filtrer l'affichage de l'emploi du temps par Professeurs, Cours et Salles
- définition de différents affichages : par semaines et par jours

E- Délimitation du projet

La demande du client était assez vaste, car elle suggérait la construction plus ou moins automatique de l'emploi du temps, et, l'aide à la création de celui-ci à travers une interface soignée et plus agréable à utiliser que l'existant, à savoir GPU.

En terme de compréhension du projet, nous étions donc d'abord partis dans l'optique d'une création automatique de l'emploi du temps, concept auquel nous avons rapidement renoncé aux vues des difficultés futures que nous allions rencontrer.

Nous avons dû diminuer nos ambitions en terme de fonctionnalités à implémenter au logiciel et nous sommes concentrés sur la saisie des données (Cours, Salles, Professeurs) ainsi que de l'affichage de l'emploi du temps pour les différents acteurs concernés.

F- État de l'art

Avant de nous lancer dans la conception du programme, nous avons d'abord réalisé des recherches vis à vis de l'existant. Nous avons commencé par analyser les différentes fonctionnalités offertes par GPU du côté étudiant et directeur des études.

The screenshot displays the SATELLYS GPU interface. At the top left is the SATELLYS logo. The main header includes the 'GPU' tab and navigation links for 'Emploi du temps Etudiants', 'Absences de l'étudiant', and 'Notes de l'étudiant'. A user profile for '190592 VALOGNES Benjamin' is shown, along with buttons for 'Planning', 'PDF', 'Export VCal', and 'Légende'. A sidebar on the left lists 'Emplois du temps' with sub-items: '> EDT Etudiants', 'EDT Groupes', 'EDT Modules', and 'EDT Salles'. The main content area shows a weekly timetable for the student, with days of the week (Lundi to Samedi) and dates (19 to 24 June 2017) on the left, and a grid of time slots (7 to 19) on the right. The grid is color-coded by course, with labels such as 'S2_GrC1 TO S2_RN6 D303 D304', 'S2_GrC1 TO AMN2 PCr D207', 'S2_GrC1 TO GPU L3a D302 D303BIS', 'S2_GrC1 PROJET PFS2 D301', 'S2_GrC1 TP P009 B3 D308', 'S2_GrC1 PROJET PFS2 D307', 'S2_GrC1 PROJET PFS2 D304', 'S2_GrC1 PROJET PFS2 D308', 'S2_GrC1 TO AMN2 PCr D305', 'S2_GrC1 TO S2_RN6 D303 D304', 'S2_GrC1 TO EC2 PMa D304', 'S2_GrC1 TP AN22 L3a A304', 'S2_INF01 CS LAN2 CMa Ala PCa', and 'Y00 COM ANA A100a D302'.

Afin d'avoir une vision plus globale du marché, nous avons élargi nos recherches et avons trouvé d'autres solutions alternatives à GPU, certaines payantes et d'autres open-source. Voici donc quelques unes des solutions existantes : *Pronote* (payant), *EDT Soft* (payant), *Free Timetabling Software* (open-source).

Dans le même temps, nous avons recherché différentes bibliothèques Java pouvant permettre la création d'emploi du temps. Nous n'en avons finalement pas intégrées au projet final.

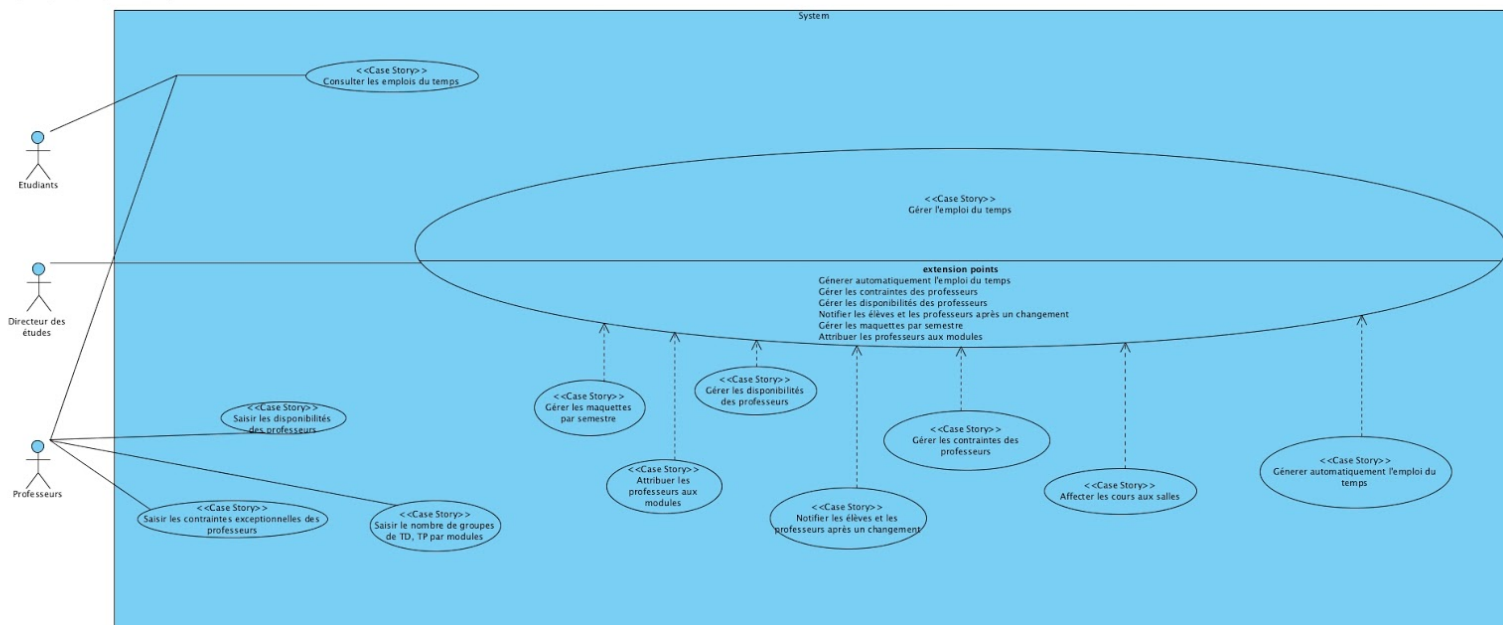
II- Architecture & Conception

Une fois la phase d'analyse du besoin terminée, nous nous sommes concentrés sur la conception du programme, cela impliquait d'avoir une vision claire des limites / périmètre du projet et des fonctionnalités à intégrer.

A- Cas d'utilisations

Comme outil de conception, nous avons utilisé Visual Paradigm, afin de concevoir dans un premier temps des diagrammes de cas d'utilisations. Le diagramme de cas d'utilisation que nous avons conçu regroupe les trois acteurs : Étudiant, Professeur et Directeur des études.

REMIENDIGES Professeur <<Extension>> LA ROCHELLE, IUT DE LA ROCHELLE



A ces trois acteurs nous avons relié les différentes fonctionnalités que chacun peut effectuer au sein du programme.

B- Scénarios

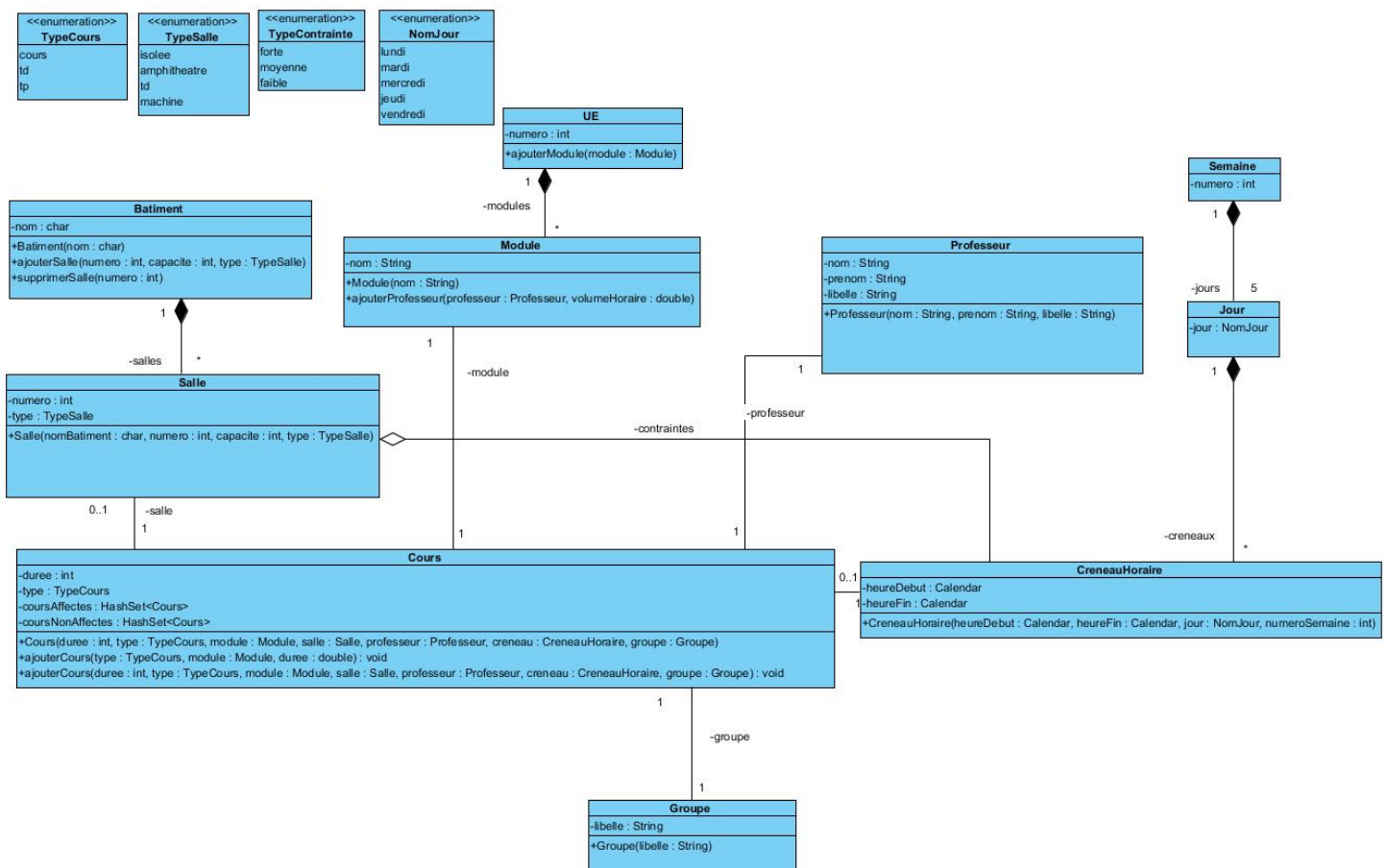
A partir des cas d'utilisations définis précédemment, nous avons choisi de les détailler à l'aide de scénario. Les scénarios nous ont permis de visualiser les fonctionnalités à implémenter et à entrevoir un début de code (objets, acteurs, actions).

Nous avons donc créé plusieurs User Story, un pour chaque scénario. Nous avons choisi de décrire le scénario nominal, mais avons aussi pensé à quelques scénarios alternatifs, ce qui a fait émerger des parties de codes (notamment pour les structures conditionnelles).

C- Diagramme de conception

Une fois les cas d'utilisations en place, nous avons démarré la conception du programme via la mise en place d'un diagramme de conception.

View: [creneauHoraire](#) (University of LA ROCHELLE, IUT of LA ROCHELLE)



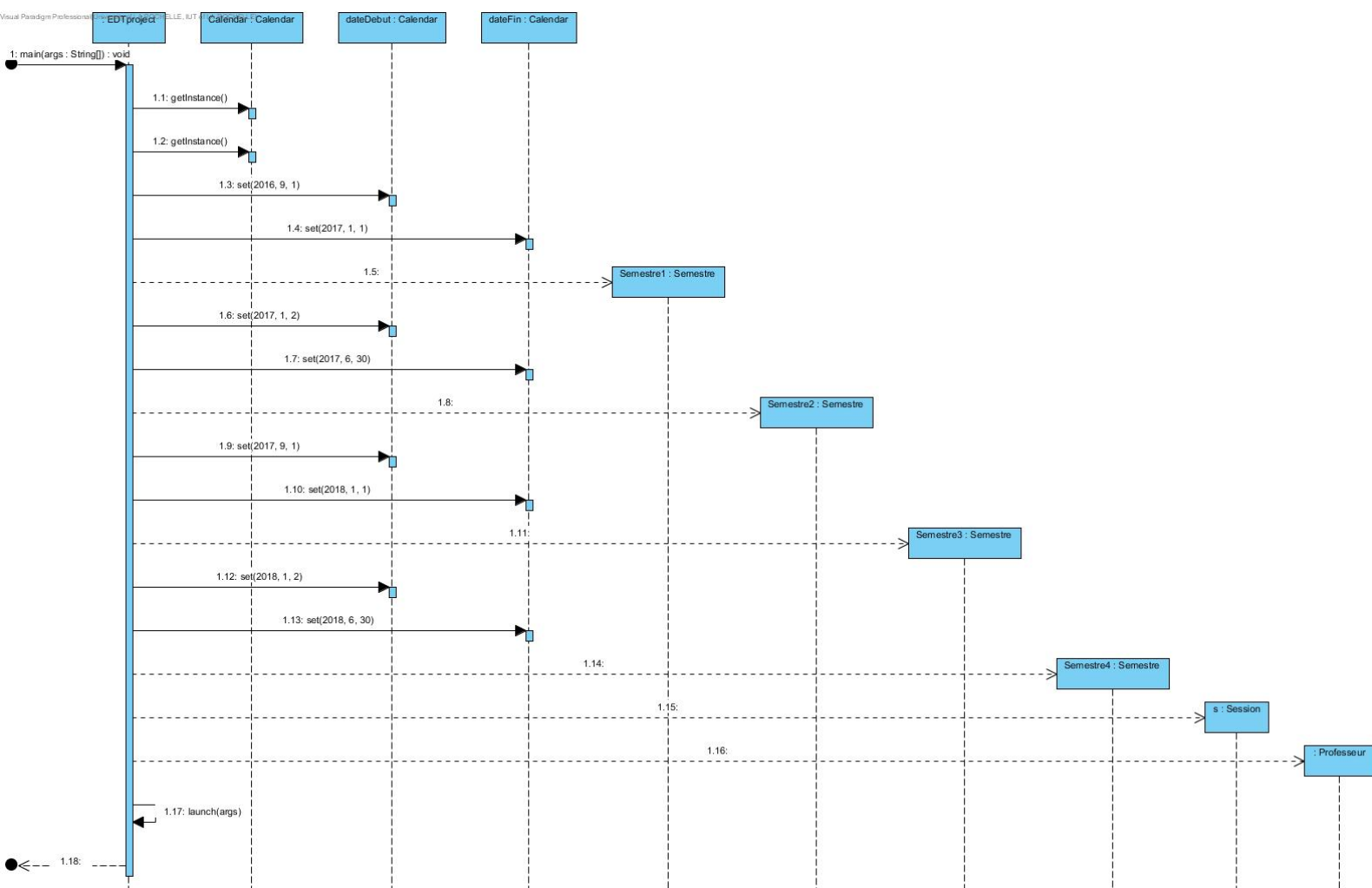
Nous avons réalisé une première version du diagramme de conception, incluant toutes les classes et objets nécessaires à la création complète de l'emploi du temps (Semestres, Jours, Professeurs...).

Cependant, après révision, nous nous sommes rendus compte que notre diagramme de conception allait au-delà des limites que nous avons

préalablement fixé. Nous avons donc revu ce diagramme et n'avons traité, en programmation, qu'une partie des classes nécessaires à la réalisation des fonctionnalités définies.

D- Diagramme de séquence

Toujours en conception, une fois le diagramme de conception finalisé, nous avons généré quelques diagrammes de séquences afin de mieux visualiser les différents appels de méthodes.



E- Wireframe

Le projet reposant en grande partie sur une IHM, nous avons prévu de multiples wireframe pour prévisualiser l'IHM à concevoir. En voici quelques captures:

Mr Nohé

Comptes utilisateurs

Gérer emploi du temps

Déconnexion

Professeurs

Disponibilités

Contraintes

Emploi du temps

Déconnexion

Eleves

Disponibilités

Contraintes

Emploi du temps

Déconnexion

Valider

Mr Martin

Disponibilités

Contraintes

Emploi du temps

Déconnexion

Saisie des disponibilités durant l'année

	Lundi	Mardi	Mercredi	Jeudi	Vendredi
Matin	Disponible	Disponible	Disponible	Disponible	Occupé
Après-midi	Disponible	Disponible	Disponible	Disponible	Disponible

Demi-journées à libérer: 1

Valider

III- Organisation & Planning

A- Planification du projet : Excel

Afin d'organiser le travail de programmation, nous avons mis en place un tableau Excel, sur lequel sont inscrites les différentes tâches de chacun, que les membres de l'équipe sont amenés à effectuer dans une période prédéfinie (plus ou moins longue en fonction de la difficulté de la tâche).

Chaque tâche à réaliser est donc inscrite dans ce fichier, avec son membre attribué ainsi que la durée prévisionnelle de réalisation de la tâche. Une section spéciale est prévue pour suivre l'avancement de la tâche, en pourcentage.

	A	B	C	D	E	F	G	H	I	J	K
1	Tâche	Lucas Michaël	Douteau Romain	Valognes Benjamin	Fouche Antoine	Période Début	Dépendance(ligne tableau)	Stade d'avancement (en%)	Priorité	Date fin de tâche réelle	
2	Mise en place d'un filtre pour trier par semestre	X				01/06 - 15/06	0	70	4		
3	Mise en place d'un DatePicker pour afficher l'emploi du temps	X	X			01/06 - 06/06	0	±00	4	11/06	
4	Terminer le contrôleur permettant d'afficher l'emploi du temps pour un professeur	X				06/06 - 08/06	0	±00	5	11/06	
5	Bug fixe de la fonctionnalité "consulter emploi du temps par un professeur"	X				08/06 - 11/06	8-9-10	±00	5		
6	Création d'un bloc FXML qui va contenir uniquement la fonctionnalité de l'affichage de l'emploi du temps	X				08/06 - 11/06	4	±00	3		
7											
8	Nouveau bloc FXML pour la création d'un cours par le directeur des études			X		01/06 - 11/06	0	±00	5	11/06	
9	Mise en place d'un fichier modèle avec variables, constructeurs, getter et setter				*	01/06 - 08/06	0	0	5		
10	Mise en place du lien entre le contrôleur et le fichier modèle			*		11/06 - 18/06	0-9	0	5		
11											
12	Bloc FXML racine	X				01/06 - 06/06	0	70	2		
13											
14	Mise en place de la liaison java-BDD		*	*		01/06 - 7	0	0	2		
15	Bloc FXML pour la gestion des sessions			X		01/06 - 11/06	0	±00	1		

Le fichier Excel se trouvant sur un Drive commun à tous les membres du projet, chacun peut visualiser l'avancement de chacun des membres et ainsi pouvoir proposer de l'aide ou des modifications dans l'ordre des tâches à réaliser.

B- Communication : Slack

Comme moyen de communication supplémentaire, nous avons aussi utilisé la plate-forme Slack. Une manière simple de pouvoir échanger entre les différents membres de l'équipe.

The screenshot shows a Slack interface for a channel named 'groupe_c1_3'. On the left is a sidebar with a dark purple background containing the workspace name 'IUT LR', a list of users (benjamin), a section for 'All Threads', a 'CHANNELS' section with '# general' and '# groupe_c1_3' (the latter is highlighted), a '# random' channel, and a 'DIRECT MESSAGES' section with 'slackbot' and 'benjamin (you)'. Below these are several other direct message contacts and an 'Invite people' button. The main area shows the channel header with the name 'groupe_c1_3', a star icon, a member count of 4, and a topic 'Add a topic'. The message history is divided by dates: 'April 5th' with a message '=> réfléchir les enjeux => les objectifs => les fonctionnalités pour chacun des 3 acteurs', and 'May 3rd' with a message from 'romain.douteau' at 9:32 PM asking for a meeting. Another date separator 'May 4th' follows. Then, a message from 'michael' at 1:50 PM is shown, followed by a message from 'annick' at 3:42 PM discussing project progress and a meeting proposal. At the bottom is a message input field with a plus icon on the left and a smiley face icon on the right.

groupe_c1_3 April 5th

=> réfléchir les enjeux => les objectifs => les fonctionnalités pour chacun des 3 acteurs

May 3rd new messages

romain.douteau 9:32 PM
Bonjour, nous sommes actuellement bloqué en ce qui concerne le diagramme de classe de conception du projet et nous aimerions quelques éclaircissements. Serait-il possible de mettre un rendez-vous avec vous demain ou vendredi si vous êtes disponible ces jours ? Merci de votre compréhension

May 4th

michael 1:50 PM
Bonjour, ce serait aussi pour savoir comment la répartition des heures est effectué sur le programme national. Est-ce que quand le nombre d'heures de chaque module est défini les différents types de cours sont précisés (exemple : 5h en amphi, 15h en TD et 7h en TP) ou non (exemple: 27h de IHM2)

annick 3:42 PM
je n'ai pas vu le résultat de votre travail sur la partie fonctionnalité pouvez vous me la transmettre avec les maquettes que vous envisagez le diagramme de conception du projet me parait ambitieux commencez par travailler pour un cas d'utilisation avez vous fait le choix javafx pour le développement, il faut avoir le choix de langage pour faire le diagramme de classe de "conception" par contre il y a également le diagramme de classes participantes pour préparer la conception de la BD, là aussi c'est à faire par cas d'utilisation Donc je ne peux pas vous aider sans voir l'état actuel de vos fonctionnalités
Merci de me les transmettre, ensuite je vous propose un point mercredi 10 à 13h30 à mon bureau

+ Message groupe_c1_3

IV- Réalisation & Programmation

A- Modification de la conception (modèle du domaine)

La phase de conception terminée et approuvée par toutes les parties prenantes, nous avons pu commencer la phase de réalisation, qui se traduit par l'écriture du modèle de conception en langage de programmation Java.

Comme dit précédemment, nous n'avons pas repris l'entièreté du modèle de conception, au vu du fait que celui-ci était en dehors du nouveau périmètre défini pour le projet. Nous avons donc dans un premier temps été fidèles au modèle de conception (aux classes concernées et leur structure) puis avons, petit à petit, été amenés à modifier ou ajouter certaines fonctionnalités, pour faire face à différents problèmes.

Exemple de modifications : Problème d'objets de tests

Dans la classe EDTproject.java, nous initialisons des objets afin de tester le programme. Un problème est survenu lors de l'exécution de classes non jointes à la classe principale.

La solution que nous avons apporté pour résoudre ce problème est la création de tableau statiques dans la classe EDTproject et la modification des classes Professeur, Salle, Session avec l'ajout dans leur constructeur respectif d'une ligne de code permettant l'ajout automatique au tableau de chaque type (cités précédemment).

Nous avons donc ajouté des tableaux Professeurs, Salles, Sessions qui regroupent automatiquement tous les objets créés, nous permettant ainsi d'appeler ces différents tableaux dans d'autres classes, sans devoir recopier le code de création des objets.

B- Ajout de nouvelles fonctionnalités

Nous avons globalement respecté le périmètre du projet. Nous n'avons donc pas été amené à ajouter de nouvelles fonctionnalités.

Pour être certain d'avoir un livrable, nous avons procédé par "méthode agile".

C'est à dire que nous avons d'abord concentré nos efforts sur l'élaboration de la vue de l'emploi du temps (vue professeur, puis étudiant et directeur des études).

Ainsi, nous avons procédé par ajout : une fois les étapes clés prévues terminées, nous avons progressivement ajouté des fonctionnalités supplémentaires, incluses dans le périmètre mais optionnelles. Après la mise en place des différentes vues, nous avons ajouté les fonctionnalités d'ajout des salles, d'ajout des professeurs, d'ajout des sessions utilisateur puis le placement des salles de la réserve (cours créés mais non placés sur l'emploi du temps) sur l'emploi du temps.

C- Utilisation du GIT

Dans le cadre du projet, nous avons été amené utiliser GIT, afin de pouvoir travailler en équipe et de manière simultanée sur le code Java. Nous avons eu quelques problèmes d'utilisations du GIT (comme le montre les plus de cent push sur le dépôt), notamment au niveau des fonctionnalités Merge et Rebase.

pts2C1-3 Recherche

Aperçu Activité **Dépôt** Configuration

pts2c132017 @ master Statistiques | Branche: master | Révision:

Nom	Taille	Révision	Âge	Auteur	Commentaire
EDTproject		350c1c08	environ 22 heures	Michaël Lucas (BuildTools)	Rajout des semestres

Dernières révisions

#	Date	Auteur	Commentaire
350c1c08	18/06/2017 19:31	Michaël Lucas (BuildTools)	Rajout des semestres
280c28ed	16/06/2017 19:35	Michaël Lucas (BuildTools)	Merge origin/master Conflicts: EDTproject/src/edtproject/CodePrincipal/Cours.java EDTproject/src/edtproject/CodePrincipal/CoursList.java EDTproject/src/edtproject/EDTproject.java
0ac8a0ff	16/06/2017 19:28	Michaël Lucas (BuildTools)	Correction sur les filtres de l'emploi du temps, ça fonctionne !
b193b5d8	16/06/2017 11:58	Romain Douteau	Ajout d'un attribut semaine à la classe cours et modification des fichiers en conséquence. Résolution des conflits. Suppression brutale des fichiers css.
edb2ad03	16/06/2017 10:55	Romain Douteau	Merge origin/master Conflicts: EDTproject/src/edtproject/view/MenuPrincipal.fxml
c18552f9	16/06/2017 10:54	Romain Douteau	Correction des problèmes dans le contrôleur MenuPrincipalController. Lien entre les fichiers de sauvegarde et le programme.
d5ed64ca	16/06/2017 10:50	Michaël Lucas (BuildTools)	Merge origin/master
10df89fd	16/06/2017 10:50	Michaël Lucas (BuildTools)	Correction des failles de sécurité de l'interface de connexion
5fc5f435	16/06/2017 10:22	VALOGNES	Merge origin/master
4719e8a2	16/06/2017 10:21	VALOGNES	Modification du FXML de l'interface Connexion.

Voir les différences

Nous avons, au fil de l'utilisation du GIT et de la confrontation à ces problèmes, appris à les résoudre sans créer d'incompatibilités entre les versions des uns des autres.

Nous avons donc organisé de façon plus précises les modifications que nous allions apporter au programme de manière à ne pas modifier chacun le même fichier, et bien travailler sur des parties différentes du programme.

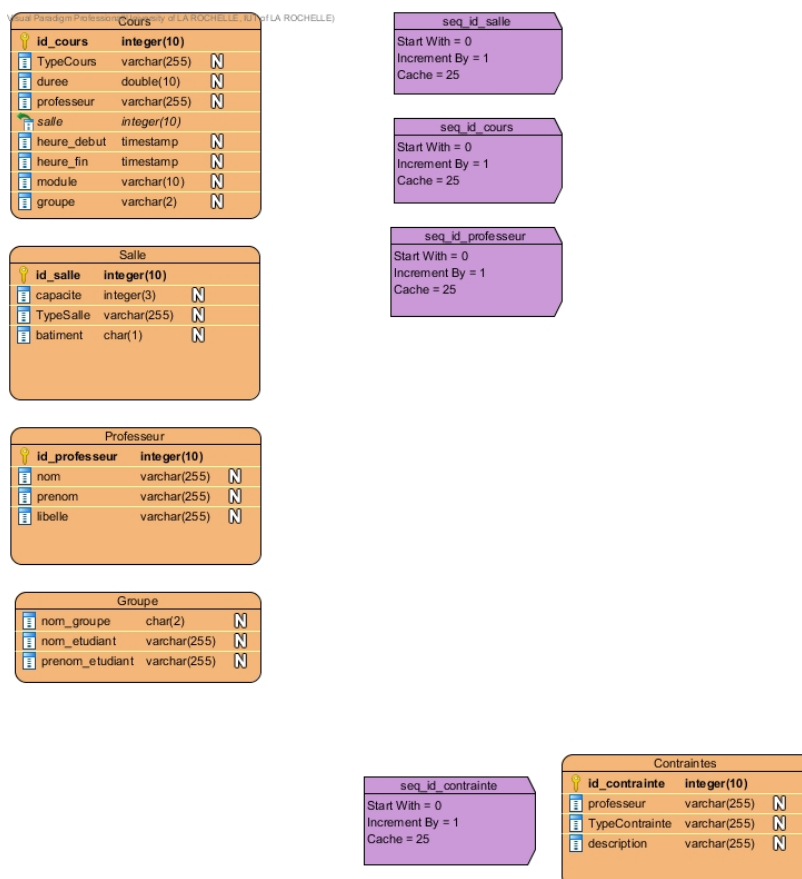
D- Problèmes rencontrés en programmation

Lors de la phase de réalisation du projet, plusieurs problèmes se sont posés, liés à des problèmes techniques et de compétences. Ces problèmes ont été soit résolus au fil de nos recherches soient tout simplement remplacés par des solutions alternatives plus ou moins proches de la solution souhaitée.

Exemples de problèmes rencontrés :

1. Mise en place de la base de données

Au départ, nous avons prévu de stocker les différents objets dans une base de données Oracle (celle de l'IUT ou sur un autre serveur). Malheureusement, nous n'avons pas pu mettre en place cette partie du projet à cause de trop nombreux problèmes, à la fois d'accès à la base depuis l'extérieur et de liaison entre la base de données et Java.



Nous avons donc opté pour une solution intermédiaire, qui est l'enregistrement des données dans un fichier et le chargement de celui-ci au lancement du programme. Ainsi, nous avons pu mettre en place un système d'enregistrement de données, mais il reste cependant un problème d'ordre fonctionnel : le fichier reste en local, et par conséquent, n'est pas accessible pour les autres utilisateurs.

2. Problèmes liés au FXML

Malgré l'accessibilité du FXML par l'intermédiaire de Scene Builder, nous avons rapidement rencontré de nombreux problèmes pour rassembler les fenêtres sur une seule interface. En effet, nous voulions faire en sorte de proposer à l'utilisateur une seule interface, et non une succession de fenêtres apparaissant pour chaque fonctionnalité demandée. Ainsi, de part ce choix, nous nous sommes rendus compte que nombre de parties de notre IHM ne pouvaient pas être redimensionnées, la seule solution que nous avons trouvée aurait été de rassembler tout nos blocs FXML dans un seul, ce qui aurait diminué de façon importante la lisibilité et la fiabilité de notre code, sachant que nous n'avons pas trouvé de moyens pour lier plusieurs controller à une seule interface FXML sans passer par la balise `<fx:include>`

Certains choix ont dû être faits concernant le FXML, ainsi, afin d'avoir une interface dynamique, nous avons mis en place un Canvas c'est à dire une zone où le programme va dessiner automatiquement les cours, cependant cela rend l'utilisation de Drag/Drop très compliquée à mettre en place, en plus de ne pas pouvoir interagir directement avec les cours dans cette zone. Nous étions aussi parti lors de la création des salles, modules, groupes et sessions utilisateurs sur l'idée de créer un tableau permettant de visualiser en temps réel la saisie et la modification de ces différents éléments. Cependant, à cause des restrictions sur l'utilisation des TableView nous avons décidé de laisser cette fonctionnalité en périmètre et de redéfinir nos priorités. Nous avons aussi eu beaucoup de difficultés pour effectuer la liaison entre le controller et le FXML, pour effectuer des appels de méthodes et transferts de variables entre les différents éléments.

3. Problème avec l'IDE Netbeans et JDK8 (la classe java.io.File)

Suite aux différents problèmes techniques lors de la mise en place d'une base de données, nous avons fait le choix d'utiliser un fichier pour sauvegarder et charger les données à partir de l'interface graphique du gestionnaire

d'emploi du temps. Mais, après la mise en place du système de fichier, un problème d'accessibilité à la classe `java.io.File` est survenu dans l'IDE Netbeans empêchant l'enregistrement des données de manière persistante.

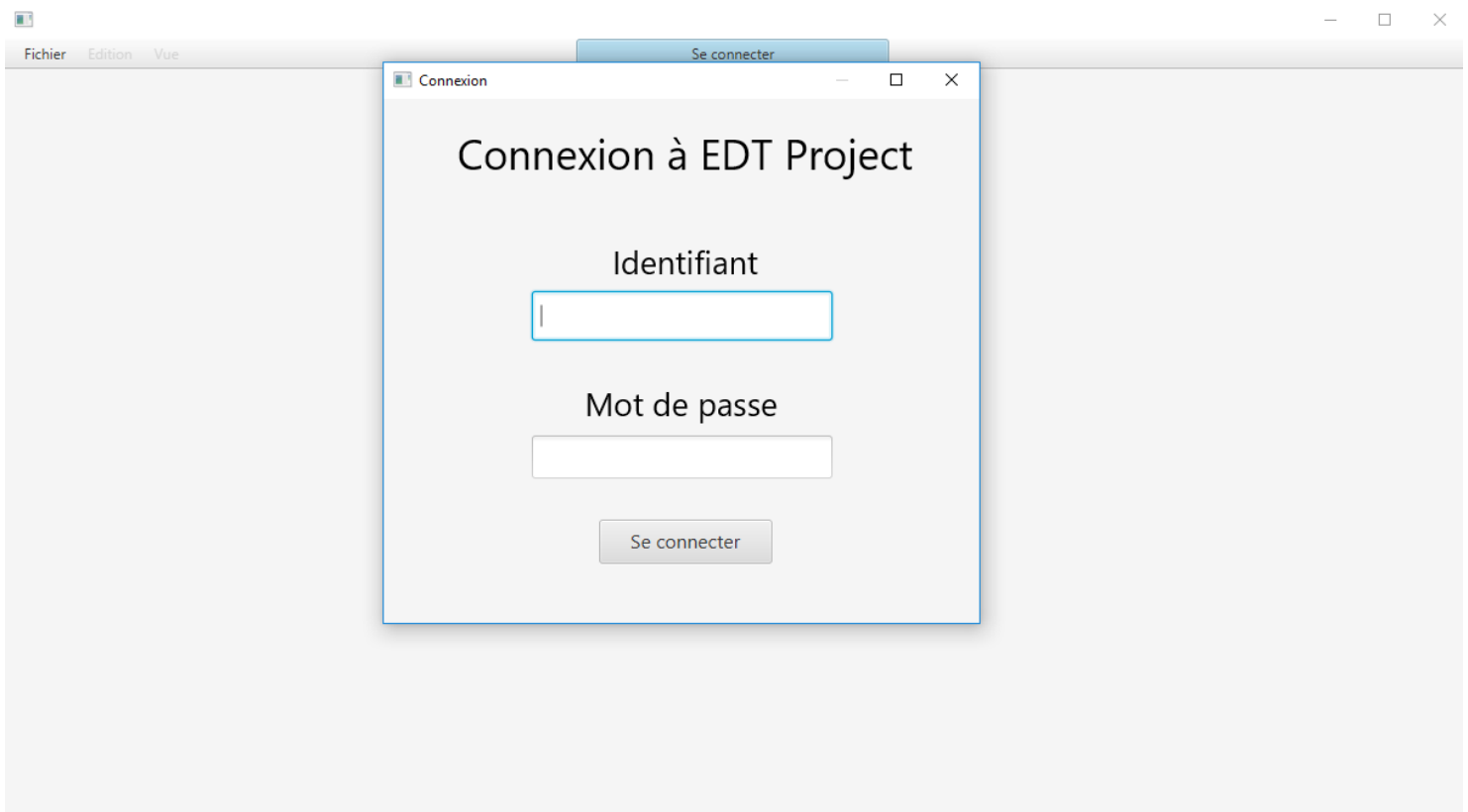
4. Problème de compréhension et d'utilisation de l'outil git

Pour mutualiser les sources du programme, nous avons utilisé l'outil git. Mais nos connaissances insuffisantes de cet outil ont nettement compromis notre efficacité à son utilisation. Pour récupérer les données envoyées sur le git avec la commande `pull`, nous avons le choix pour régler les conflits entre les fichiers entre utiliser la méthode de `Rebase` ou celle du `Merge` mais nous n'avons jamais compris la différence entre les deux. Nous avons donc principalement utiliser la méthode `Merge` car celle-ci nous permettait de comparer les différences entre les fichiers sans écraser nos modifications.

V- Bilan : rendu final

Après avoir implémenter les différentes fonctionnalités demandées, nous pouvons maintenant vous proposer quelques aperçus de l'IHM.

A- Fenêtre de connexion au gestionnaire d'emploi du temps :



B- Fenêtre de saisie des groupes :

The screenshot shows a window titled "Saisie des groupes". At the top, there is a menu bar with "Fichier", "Edition", and "Vue". On the right, there is a user profile icon labeled "Regis Nohe", a settings gear icon, and a "Se déconnecter" button. The main content area contains a form with the following elements:

- A label "Entrer le nom du groupe de TD en une lettre:" followed by a text input field containing the letter "C".
- A label "Nombre de groupes de TP associés :" followed by a numeric input field containing the value "2".
- A label "Année associée:" followed by two radio buttons: "Année 1" (which is selected) and "Année 2".
- A button labeled "Ajouter le groupe" located below the form.

At the bottom of the window, there is a tabbed interface with the following tabs: "Affichage Emploi du temps", "Modification Emploi du temps", "Création des sessions utilisateurs", "Création des modules", "Création des groupes" (which is the active tab), and "Création des salles".

C- Fenêtre de saisie des modules:

The screenshot shows a window titled "Saisie des modules". At the top, there is a menu bar with "Fichier", "Edition", and "Vue". On the right, there is a user profile icon labeled "Regis Nohe", a settings gear icon, and a "Se déconnecter" button. The main content area contains a form with the following elements:

- A label "Entrer le nom du module :" followed by a text input field containing the text "LAN".
- A label "Sélectionner la couleur :" followed by a color selection dropdown menu showing a red color swatch and the hex code "#b31a1a".
- Four checkboxes for semesters: "Semestre 1" (checked), "Semestre 2" (checked), "Semestre 3" (unchecked), and "Semestre 4" (unchecked).
- A button labeled "Ajouter le module" located below the form.

At the bottom of the window, there is a tabbed interface with the following tabs: "Affichage Emploi du temps", "Modification Emploi du temps", "Création des sessions utilisateurs", "Création des modules" (which is the active tab), "Création des groupes", and "Création des salles".

D- Fenêtre de saisie des cours :

Fichier Edition Vue

Regis Nohe Se déconnecter

Réserve:

POO, COURS, 1h30

Saisie des cours

Sélectionner le module : POO

Sélectionner le type de cours : COURS

Sélectionner la semaine : 25

Sélectionner durée : 1 h 30 m

Ajouter le cours

Alternier affichage Affichage par jour 21/06/2017

Filtrer par : Groupes

Mon emploi du temps Semaine 2

Affichage Emploi du temps Modification Emploi du temps Création des sessions utilisateurs Création des modules Création des groupes Création des salles

E- Fenêtre d'affichage de l'emploi du temps :

Fichier Edition Vue

Regis Nohe Se déconnecter

Alternier affichage Affichage par semaine 21/06/2017

Filtrer par : Groupes Année1

Mon emploi du temps Semestre 2 Semaine 25

8h00 LUNDI 19 Juin MARDI 20 Juin JEUDI 22 Juin VENDREDI 23 Juin

10h00 AP, TD, D206, RNo, C

10h15

12h15 AP, COURS, C205, PGo, C

16h15 BD, TD, D301, FAm, A1 BD, TD, D301, FAm, A1

18h15

Affichage Emploi du temps Modification Emploi du temps Création des sessions utilisateurs Création des modules Création des groupes Création des salles

VI- Extensions possibles

En terme de fonctionnalités que nous pourrions ajouter au logiciel afin de le compléter, on peut citer :

- l'ajout des contraintes professeur (et le types de contraintes associées)
- mise en place d'une version mobile
- l'ajout d'une liaison à une base de données
- la possibilité de générer automatiquement l'emploi du temps