INSA RENNES

Rapport de planification
Version finale

CloudBook

« A social network of applications »

Abderrahmane AMALOU
Matthieu BERTIN
Sami EL ALAMI TREBKI
Gwendal LE MOULEC
Bruno MATRY

Encadrant :
Jean-Louis PAZAT

Institut National des Sciences Appliquées
Ecole publique d'ingénieurs

Table des matières

Introduction

Le présent dossier a pour objectif d'expliquer en détails la planification du projet The CloudBook. Il devra notamment établir la liste des tâches atomiques qui composent ce projet, en présentant les interdépendances qu'il existe entre chacune d'entre elles. Ce document servira de référence en cas d'imprévus, comme par exemple une difficulté technique rencontrée pour l'accomplissement d'une tâche : il faudra alors déterminer combien de temps pourra être consacré à la résolution des problèmes sans mettre en péril le reste du projet.

Nous commencerons par rappeler les objectifs du réseau The CloudBook. Nous présenterons ensuite le schéma de planification globale depuis la date d'aujourd'hui jusqu'à la fin du projet, prévue pour le mois de mai. Les grandes tâches qui auront été définies seront alors découpées unes à unes en tâches atomiques.

I. Présentation du projet : The CloudBook

The CloudBook est un réseau social d'applications. Il a pour objectif d'aider les utilisateurs à trouver la meilleure offre de cloud pour leurs applications en se basant sur l'expérience utilisateur partagée.

A. Contexte

Le but du projet est de fournir des conseils sur le choix du meilleur fournisseur, aux applications installées sur le cloud. L'idée sur laquelle se fonde le projet est que les applications puissent communiquer entre elles pour se conseiller mutuellement. Ainsi, une application qui a besoin de trouver une plate-forme de cloud computing adaptée à ses besoins peut tisser des liens avec des applications qui sont proches d'elle en termes de besoins. Ces applications pourront se baser sur leur expérience pour la conseiller. A long terme, une multitude de groupes d'applications similaires pourront se former pour échanger des conseils.

The CloudBook est donc un réseau social d'applications installées sur le cloud (ou à vocation d'y être). Celles-ci sont enregistrées sur le réseau par leur propriétaire, qui peut renseigner des informations sur son application, comme la plate-forme de cloud computing (ex : Windows Azure, Rackspace) sur laquelle elle tourne ou a tourné. *The CloudBook* devra permettre de prendre des mesures de performances qui caractérisent une application, qui pourront également être renseignées par son propriétaire.

The CloudBook est un concept pensé par l'équipe Myriads de l'IRISA qui nous en a confié la majeure partie du développement. Elle nous a toutefois laissés libres et aucune contrainte n'a été imposée. Elle n'intervient pas dans les choix qui sont faits sur le projet et n'a pas d'exigences particulières, si ce n'est qu'une première version soit proposée d'ici fin mai.

Une fois enregistrée sur le réseau, une application pourra communiquer avec d'autres afin de constituer un réseau d'applications amies, sélectionnées selon la proximité qu'elles entretiennent en termes de besoins. Elle pourra ensuite partager des informations avec elles dans le but d'établir un classement des plates-formes de cloud les plus adaptées à ses besoins, par comparaison avec les autres.

The CloudBook sera décentralisé, les échanges se feront au moyen d'un réseau pair-à-pair avec des algorithmes de type Gossip.

La **figure 1** représente les différents modules qui doivent composer *The CloudBook* :

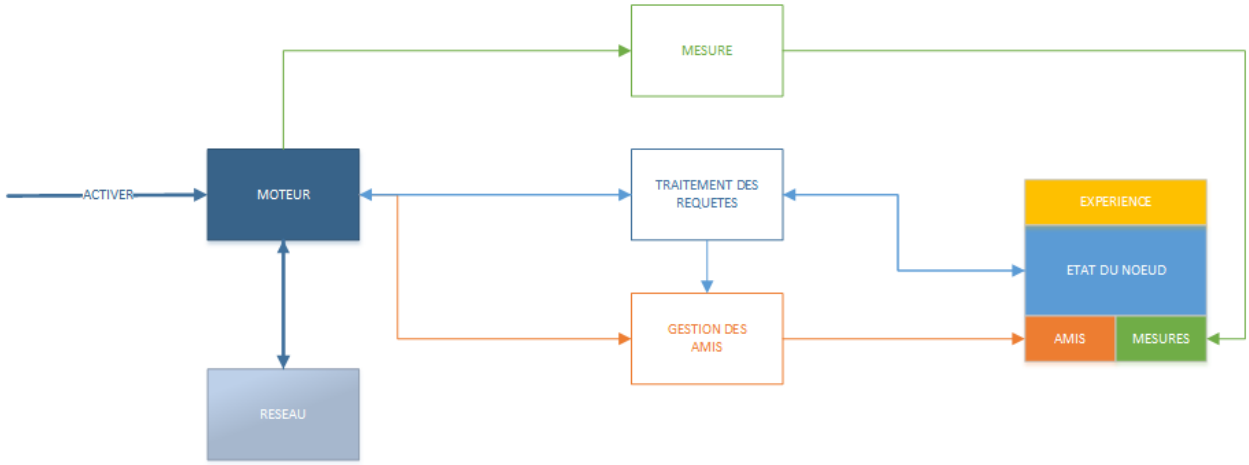

Figure 1 : Modules

- **Moteur :** module chargé d'ordonnancer les actions et les interactions des modules.

- **Mesure :** module chargé de la prise de mesures sur les performances d'une application sur une ou plusieurs plates-formes de cloud computing.

- **Traitement des requêtes :** module récupérant les données acquises lors des échanges sur le réseau Gossip et envoyant les ordres de partage des données de l'application courante avec d'autres applications du réseau.

- **Gestion des amis :** module listant les applications membres connues par l'application membre courante et déterminant celles qui peuvent être considérés comme amies (membres avec lesquels on partagera des données).

- **Réseau :** structure "physique" du réseau, organisé en couches : Pair-à-pair, Gossip, réseau "utilisateur" (introduisant la notion d'amis).

Le dernier bloc, composé de l'expérience, des amis, des mesures et de l'état du nœud, est la représentation des données stockées par une application membre. Les modules qui y sont reliés sont chargés de mettre à jour ces données. L'expérience représente l'historique d'une application membre, c'est à dire la liste des plates-formes de cloud computing sur lesquelles elle a fonctionné.

B. Contraintes

The CloudBook est un projet très vaste. L'IRISA ne nous a donc pas demandé de développer The CloudBook dans sa totalité mais de fournir une base fonctionnelle qu'elle pourra améliorer et compléter ensuite. Nous avons alors convenu que le choix se ferait entre les deux grands aspects fonctionnels de The CloudBook : la détermination effective et fiable des plates-formes de cloud computing les mieux adaptées à une application donnée d'une part et la mise en place du réseau d'un point de vue architectural. Nous avons opté pour le deuxième choix. L'implémentation des fonctionnalités volontairement omises sera laissée à la discrétion de ceux qui reprendront le projet.

Notre choix de nous affranchir d'une partie des fonctionnalités est aussi motivé par le fait que nous fonctionnons en effectif réduit : nous commençons à 5 sur un projet prévu à l'origine pour 6 ou 7 personnes. Nous ne serons ensuite plus que 3 pour le développement et la finalisation du projet. Au regard de ces contraintes, nous considérons qu'il vaut mieux s'engager pour une solution intermédiaire mais fonctionnelle.

II. Planification générale

A. Méthode de planification

Comme nous venons de le mettre en évidence, le cahier des charges n'est pas figé ; il existe en fait plusieurs versions de The CloudBook qui sont acceptables pour un rendu à l'IRISA. C'est pour cela que nous jugeons pertinent d'adopter une stratégie itérative donnant lieu à des livrables intermédiaires avant la version finale de notre travail. Nous ne souhaitons donc pas adopter une méthode de cycle en V classique, qui n'intègre pas suffisamment l'idée d'ajustement des objectifs en cours de route.

Les méthodes agiles, qui sont pourtant là pour régler ce problème, ne sont pas très pertinentes dans notre cas car nous sommes finalement laissés très libres par l'IRISA, auquel nous ne sommes pas tenu de faire des rapports sur l'avancement de notre travail. De plus, nous jugeons le format des projets inadapté à ces méthodes, pour lesquelles il faut avoir une idée très précise du nombre d'heures consacrées au projet en une semaine, ce qui n'est pas notre cas. Il est difficile de définir un nombre satisfaisant de versions intermédiaires dans ces conditions de développement.

Nous avons donc opté pour une méthode hybride, le "cycle en W", qui se constitue de deux cycles en V pour deux versions du réseau The CloudBook.

B. Survol du semestre

La **figure 2** présente une vue globale du travail restant jusqu'à la soutenance finale :

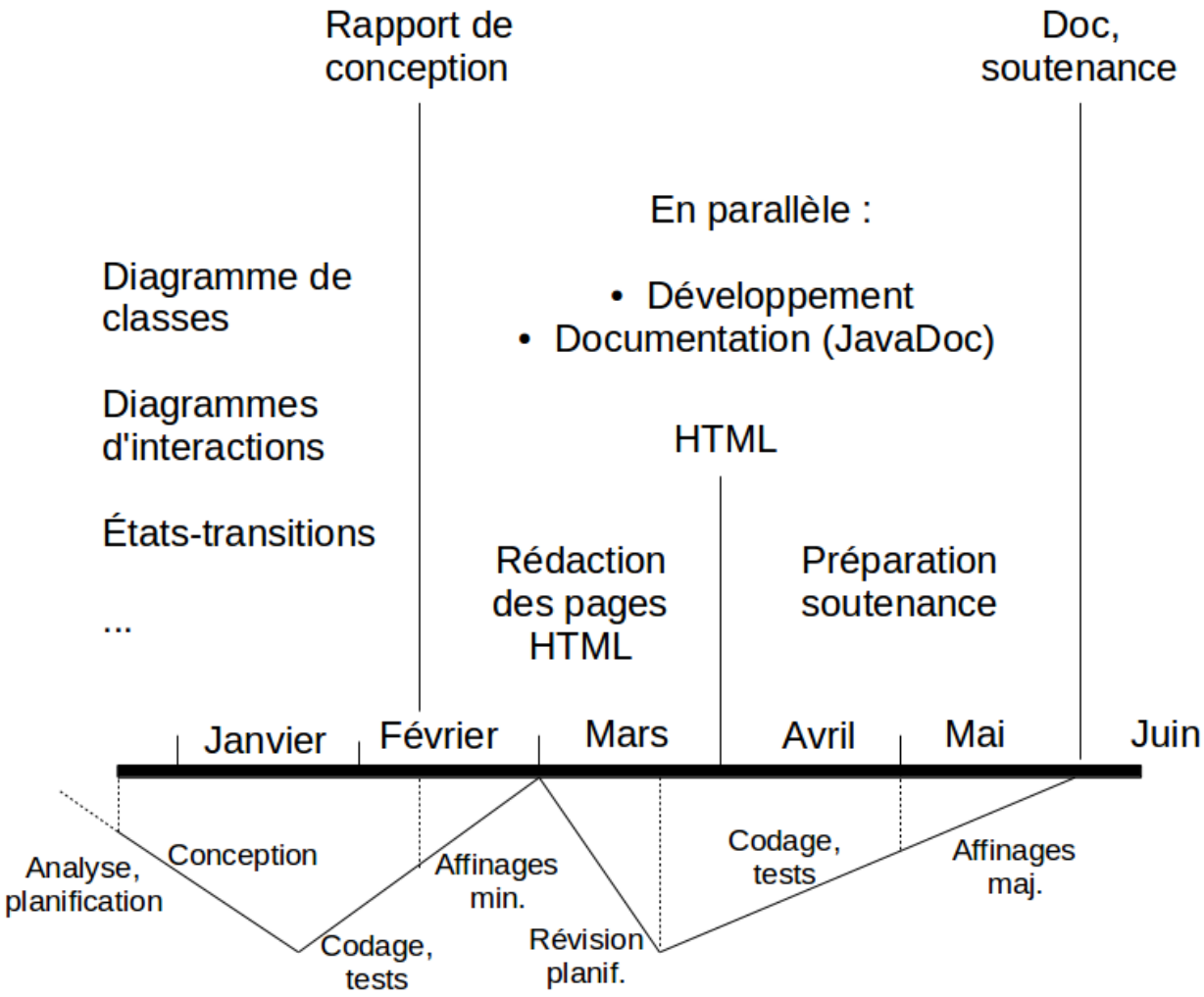

figure 2 : Chronologie globale

Premier cycle en V : il a déjà commencé ; il s'agit de ce que nous faisons actuellement, c'est à dire l'analyse et la planification. La prochaine étape est la conception.

La deuxième branche de ce premier cycle (débutant en février) correspond au codage et aux tests des fonctionnalités les plus fondamentales (nous les précisons dans la suite de ce rapport). Le développement se fera en Java. Nous avons ajouté à la fin une courte phase d'affinage (1 semaine) : nous prévoyons d'arrêter de développer des fonctionnalités durant cette courte période, pour nous concentrer sur l'aspect "livrable" du logiciel. Concrètement, cette période servira à :

- soigner l'ergonomie et le confort d'utilisation ;
- supprimer ou masquer les fonctionnalités qui ne sont pas au point ;
- ajouter quelques touches finales sur l'interface graphique.

Notons tout de même que ces retouches seront mineures car elles ne concerneront pas la version finale de notre projet.

Deuxième cycle en V : il est deux fois plus court que le premier car nous y prévoyons moins de fonctionnalités à implémenter. C'est normal, car comme cela a été dit, il fait office de sécurité, afin de s'assurer d'avoir une version fonctionnelle à la fin du temps imparti pour le projet.

La branche descendante nous permettra de faire un bilan du premier cycle. Nous en profiterons pour revoir les spécifications et le planning initialement prévus si nous jugeons cela nécessaire.

Comme pour le premier cycle, nous prévoyons une période d'affinage, qui sera cette fois plus important.

C. Versionning et travail collaboratif

Nous avons déjà commencé à utiliser **Git** pour partager nos premiers diagrammes de conception (cas d'utilisations). Nous prévoyons d'utiliser le même outil pour le développement sous Eclipse.

Git sera également utilisé pour partager les pages HTML. Les mises à jour du planning sur Microsoft Project ainsi que les rapports et les slides de présentation seront partagés grâce à l'outil de Microsoft **SkyDrive**.

III. Découpage des tâches principales

Les modules rappelés en **figure 1** peuvent être découpées en sous-tâches atomiques. Le module Moteur n'est là que pour activer successivement les autres modules: nous considérons donc que sa mise en place est déjà une tâche atomique. Le développement du module Mesure est laissé à la discrétion de nos successeurs, nous nous contenterons seulement de lui donner un fonctionnement "naïf", juste ce qu'il faudra pour permettre la réalisation de tests. Sa mise en place en tant que "boîte vide" constituera donc elle aussi une tâche atomique. Implicitement, chaque sous-tâche atomique fera l'objet de tests unitaires JUnit. La durée estimée nécessaire à l'accomplissement d'une sous-tâche est exprimée en Semaine Homme. A titre indicatif, nous estimons qu'une Semaine Homme équivaut dans notre cas à 8h de travail pour une personne.

A. Module Réseau

Cycle 1:

1. Création des interfaces :

Réalisation des interfaces à partir des besoins des autres modules.

Ces interfaces seront complètement indépendantes de l'implémentation du module.

2. Mise en place d'un réseau de type client / serveur

a. Développement du serveur (parallélisable avec l'étape 1).

Il aura une table complète des IP des clients et redirigera les connexions. Le serveur sera un objet indépendant des interfaces réalisées à l'étape 1 puisqu'il ne fera pas partie des nœuds de The Cloudbook. Le processus sera lancé en mode démon sur une machine à part.

b. Développement du programme client.

Les clients implémenteront les interfaces réalisées à l'étape 1. Ils connaîtront l'adresse IP du serveur et ne communiquerons directement qu'avec le serveur.

- Test de base client/serveur
- Tests à échelle réduite (une dizaine de PC)

Cycle 2 :

1. Mise en place du réseau P2P

a. Implémentation

Création d'une (ou plusieurs) classe(s) permettant d'instancier des objets de type noeudRéseau. Ces instances seront capables de mémoriser et de sauvegarder lors de leur fermeture les adresses IP de certains nœuds. Tout au long du développement, il sera nécessaire de réaliser des tests unitaires.

b. Tests et intégration

Il faudra ensuite réaliser l'intégration du module dans le reste du projet, voire prévoir des intégrations intermédiaires.

- Tests à échelle réduite (une dizaine de PC)
- Tests grande échelle (IRISA)

B. Module Gestion des amis

Tout se fera au cycle 1 pour ce module

1. Mise en place d'une classe Profil (~ ½ semaine)

Un profil possède une adresse et des caractéristiques (de l'application cloud) et une liste de profils d'ami (liste associative, adresse du profil → indice de pertinence) puis une liste de profils "connaissance".

2. Mise en place d'une méthode Calcul de pertinence (~ ½ semaine)

Le calcul de pertinence permet de savoir quelle application a les caractéristiques la plus proche de celle que l'on veut déployer sur le Cloud. Une application étant représenté par un vecteur de donnée : plus la distance entre deux vecteurs d'applications est courtes plus l'indice de pertinence est grand.

3. Mise en place d'une méthode de mise à jour (~ 1 semaine)

Cette méthode récupère sur le réseau une liste de personne au hasard puis rajoute les profils qui ont un indice de pertinence supérieure à un seuil défini dans la liste d'ami et les autres dans connaissances.

C. Module Traitement des requêtes

Cycle 1 :

Exploitation des données

1. Récupération des données relatives à l'expérience des autres (depuis la couche Grossi). (~ ½ semaine)

Les algorithmes de Gossip auront permis d'acquérir la connaissance de l'expérience des autres (les plates-formes de cloud computing sur lesquelles une application a déjà été installée). Il faut maintenant traiter ces données pour les rendre exploitables par la couche The CloudBook.

2. Récupération des données relatives aux mesures des autres (depuis la couche Grossi). (~ ½ semaine)

De la même manière, les algorithmes de Gossip auront permis d'acquérir la connaissance des besoins des autres. Il faut également traiter ces données.

3. Réalisation d'un classement des clouds (~ 1 semaine) :

Pour chaque cloud d'une même "expérience", affecter un poids inversement proportionnel à la distance entre le nœud courant et le nœud correspondant à l'expérience.

Au sein d'une même expérience, il faut pouvoir discriminer les cloud : affectation d'un sous-poids inversement proportionnel à l'ancienneté.

Réalisation d'un classement basique en fonction de ces poids.

Partage d'informations :

Identification des destinataires successifs de l'envoi --> classement par indice de pertinence. (~ ½ semaine)

Cycle 2 :

Exploitation des données

1. Changer l'opération permettant de classer ainsi les clouds en fonction des poids définis (~ ½ semaine)

La nouvelle opération doit être pertinente, répondre à des critères précis à déterminer.

Partage d'informations.

2. Déterminer les informations à masquer en fonction de l'indice de confiance. (~ ½ semaine)

3. Implémenter les algorithmes de Gossip permettant l'échange des informations filtrées.

D. Stratégie de tests d'intégration

Nous partons du principe que tout assembler d'un seul coup est dangereux : les risques d'échecs sont multipliés par le nombre de modules mis en jeux dans l'assemblage. C'est pour cela que nous prévoyons d'assembler les modules deux à deux, avant d'assembler les composants obtenus.

Avant de passer les tests d'intégrations, tous les modules seront testés isolément. Le lien avec les autres modules sera alors simulé, pour "donner l'impression" au module testé qu'il est lié avec les autres modules alors que ce ne sera pas le cas.

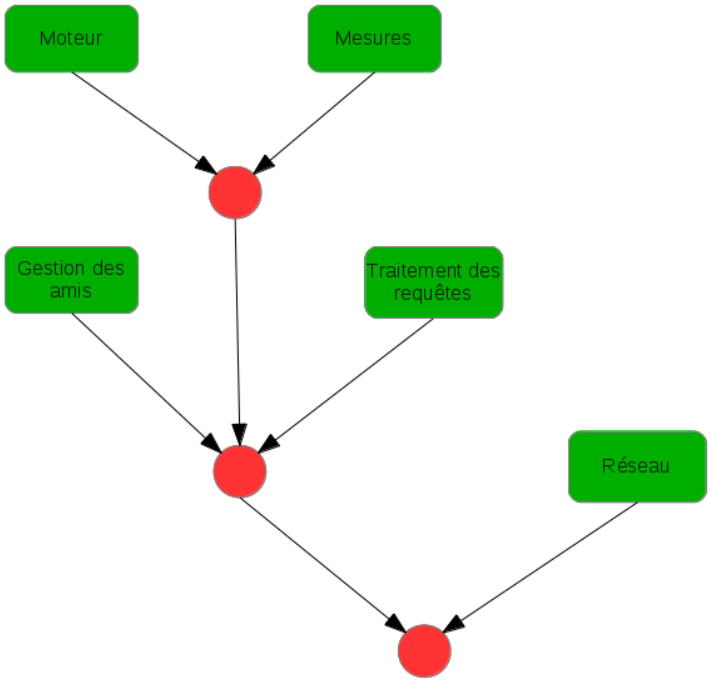

Figure 4 : stratégie d'intégration

Plusieurs intégrations successives permettent de mieux cibler les problèmes qui seront rencontrés lors de ces tests. L'ordre suivi permettra de réaliser les premières intégrations en local, pour terminer par celle qui implique le réseau et qui se fera donc dans des conditions particulières : salles de TP, IRISA…

IV. Synthèse : Décisions de planification

A. Analyse des risques

Une grande partie de la prévention des risques a déjà été exposée : il s'agit de nos précautions prises pour s'assurer de rendre une version minimale de *The CloudBook* (cycle en W principalement). Nous considérons qu'un diagramme de Gantt tient aussi ce rôle. Il se révélera particulièrement utile dans le cas d'un problème sur une tâche, puisqu'il permettra de connaître le temps maximum qu'il faudra passer sur la résolution du problème pour que l'ensemble du projet ne soit pas mis en péril.

Il reste cependant des risques liés à la mise en place des couches réseau pair-à-pair et Gossip. Il s'agit en effet de technologies pour lesquelles nous n'avons que peu d'expérience. Leur expérimentation est d'ailleurs contraignante puisqu'elle devra se faire sur un nombre restreint de machines, en l'occurrence celles du département. Les essais feront peut-être l'objet de démarches auprès de l'administration réseau et nous seront évidemment limités en temps. Nous savons aussi que d'autres étudiants avant nous ont eu des problèmes majeurs pour faire travailler en réseau ces machines : le risque d'être bloqués est évident.

Nous identifions pour ce problème plusieurs niveaux de risques et les stratégies associées:

- **Cas extrême : nous n'arrivons absolument pas à faire fonctionner le réseau Gossip.**
    - **Solution :** tenter au moins de simuler le réseau Gossip. Si cela ne marche pas, il faudra alors nous rabattre sur le modèle client-serveur préalablement développé.

- **Niveau moyen : nous parvenons à mettre en place un réseau, mais très limité, ou fonctionnant mal, ou peu performant.**
    - **Solution :** elle dépend du problème précis. Il faudra alors faire un choix : soit abandonner le réseau Gossip et donc se placer dans le cas extrême, soit adapter notre version de *The CloudBook* aux limitations du réseau.

- **Niveau faible : nous arrivons à mettre en place un réseau Gossip performant, mais seulement de manière simulée.**
    - **Solution :** nous jugeons cette solution acceptable, à condition de laisser les traces de nos tentatives sur un vrai réseau.

Dans tous les cas, une protection supplémentaire, que nous pensons être même primordiale, est l'ajout d'un effet "waouh" : quelque soit l'état de notre projet au moment de la soutenance finale, il faut mettre en évidence ses qualités, en impressionnant le plus possible le jury.

Conclusion

En résumé, notre projet consiste à développer un prototype robuste du réseau d'applications *The CloudBook*, imaginé par l'IRISA. Ce prototype permettra d'intégrer des applications "du cloud" dans un réseau de partage de connaissances.

La complexité de l'application, qui devra fonctionner sur un réseau pair-à-pair de type Gossip, nous a mené à planifier son développement bloc par bloc, dans le cadre d'un processus d'intégration progressive stratégique. Le développement de chaque module se fera de manière itérative, afin d'assurer un noyau fonctionnel rapidement et de faciliter l'ajout successif des fonctionnalités.

Annexes :

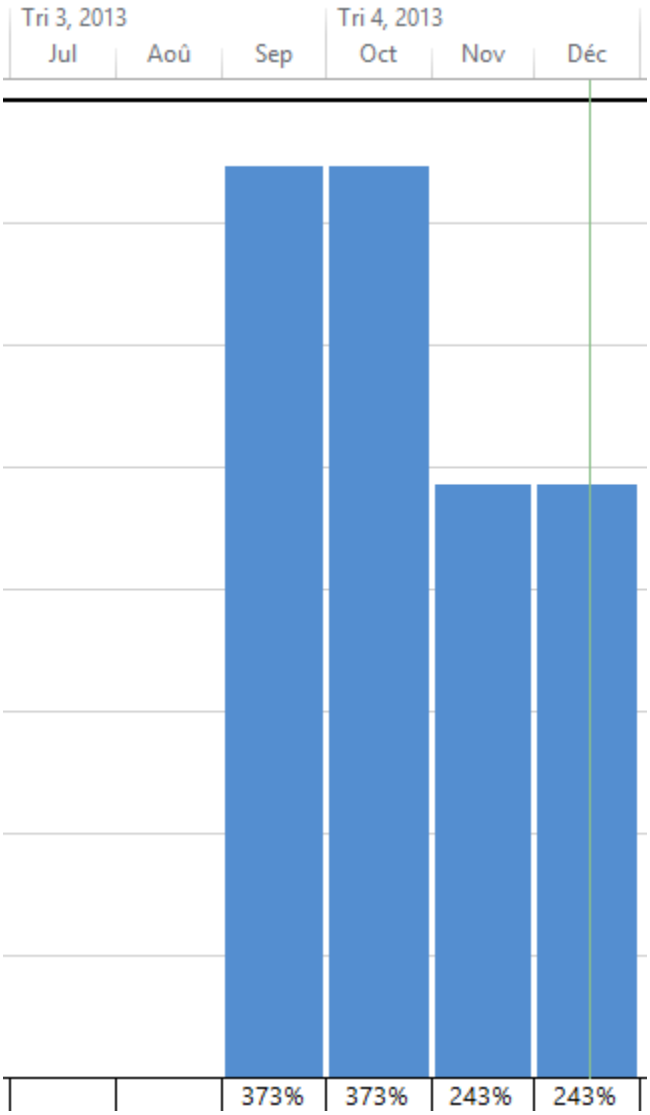

Figure 5 : Surcharge maximum S1(effectif de 5 personnes)

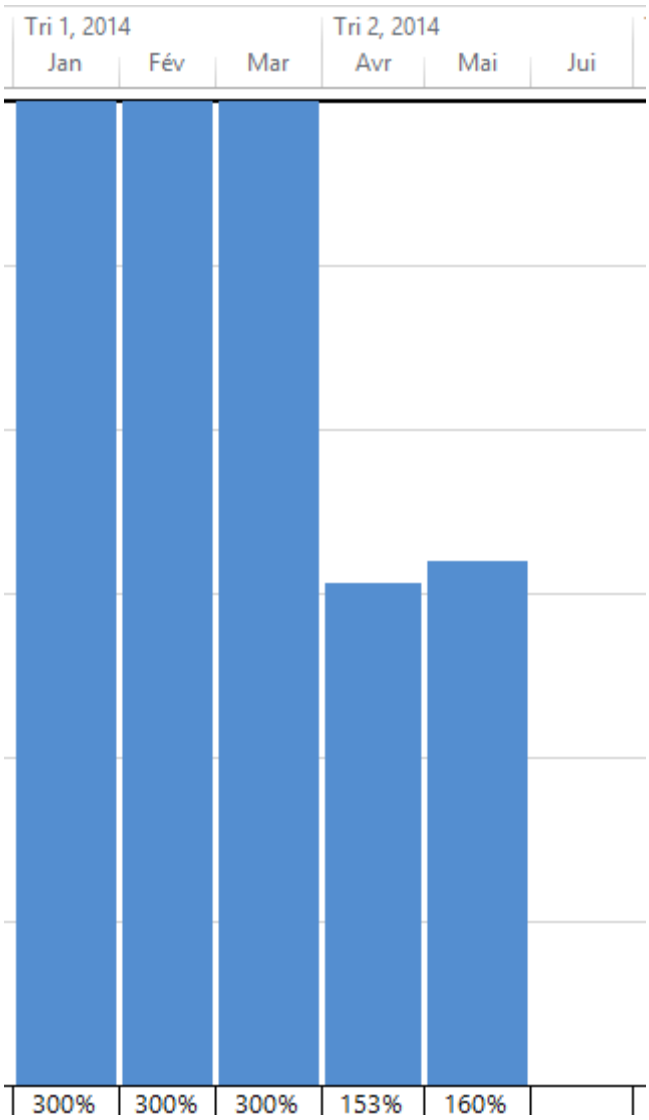

Figure 6 : Surcharge maximum S2 (effectif de 3 personnes)

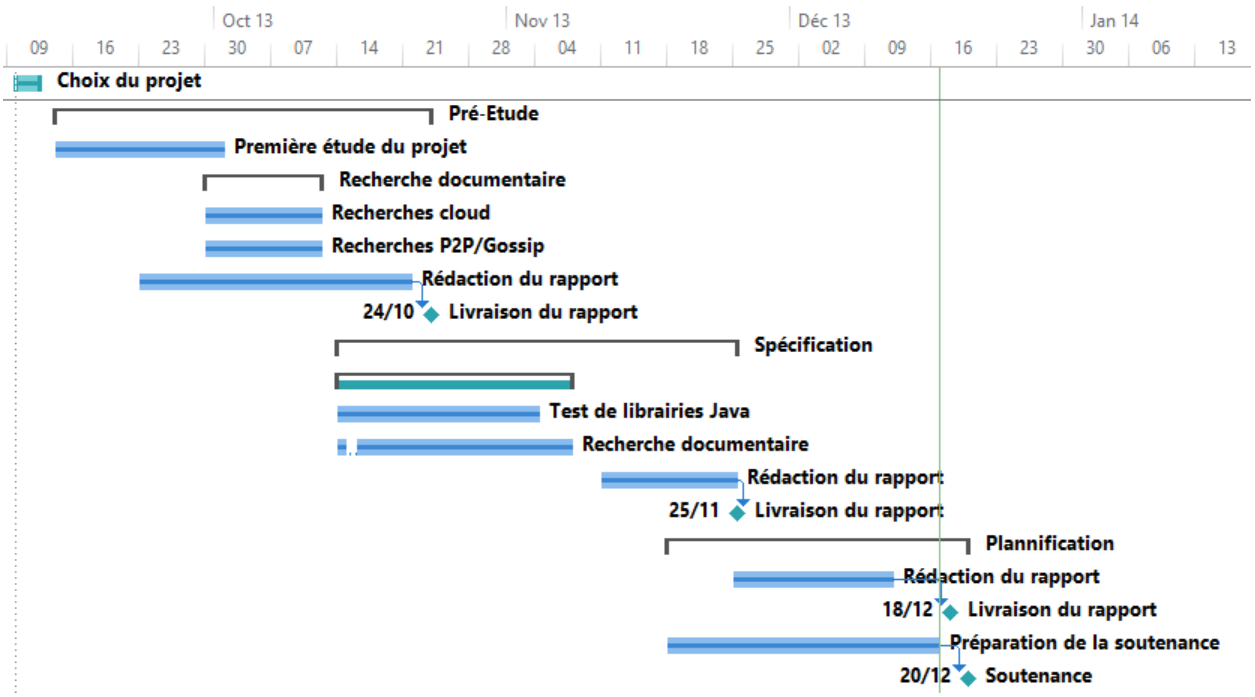

Figure 7 : Diagramme de Gant (1/6) – Début du projet

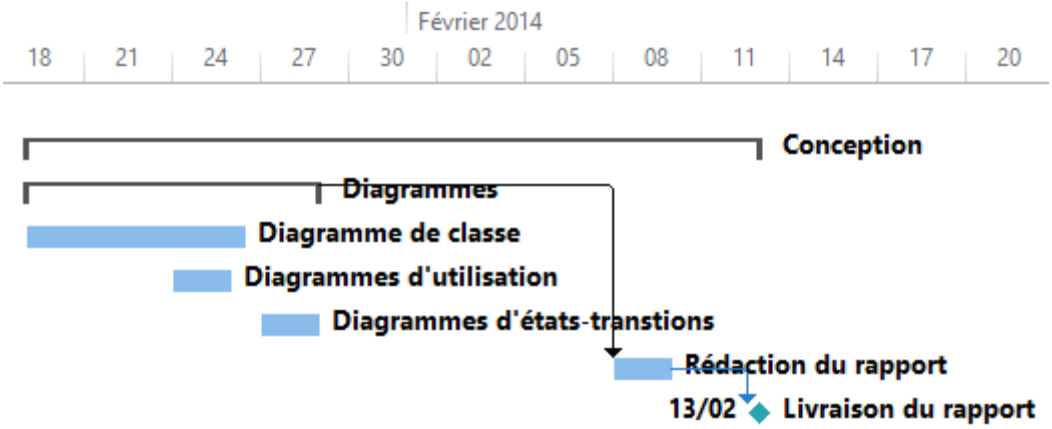

Figure 8 : Diagramme de Gant (2/6) - Conception

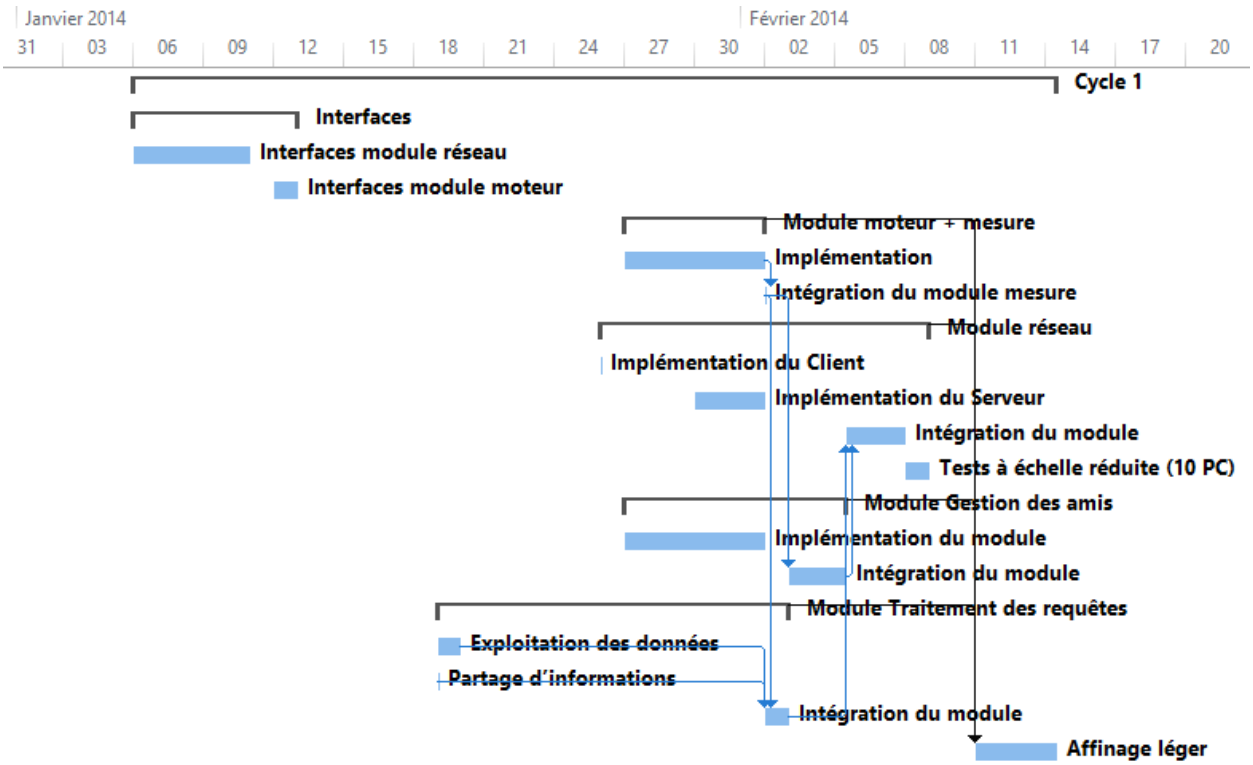

Figure 9 : Diagramme de Gant (3/6) – Développement Cycle 1

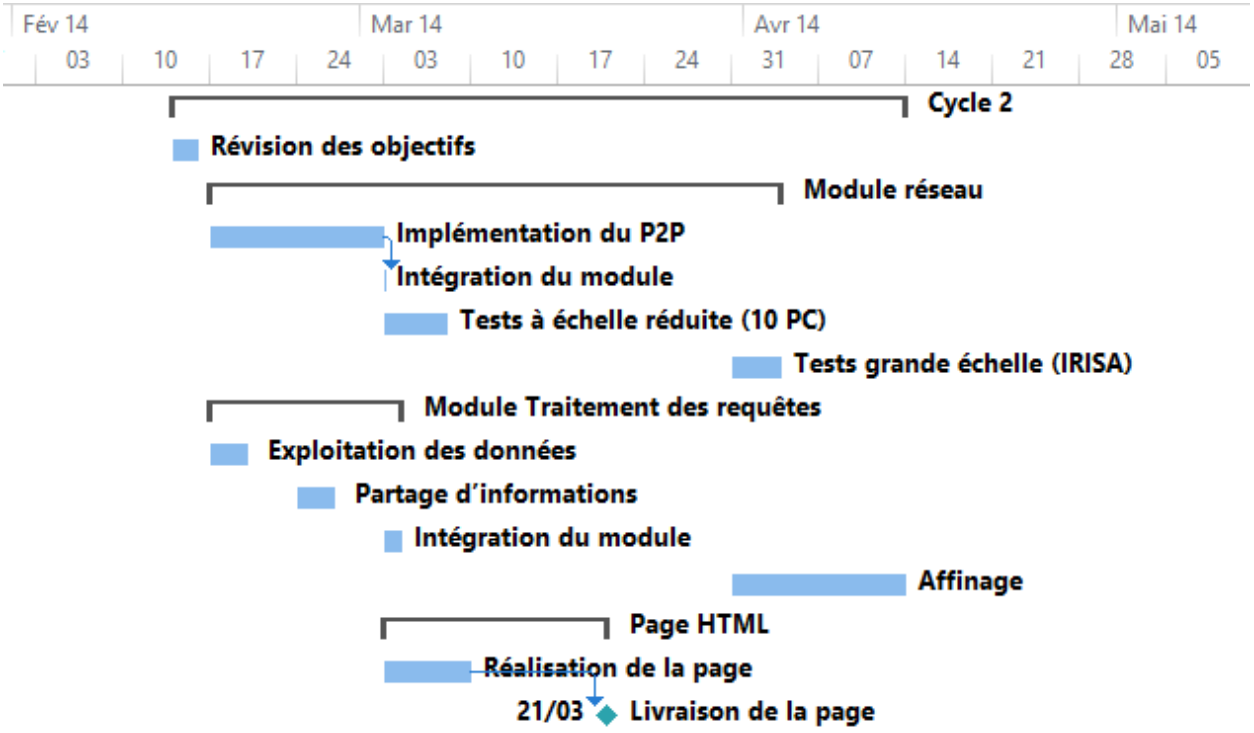

Figure 10 : Diagramme de Gant (4/6) – Développement Cycle 2

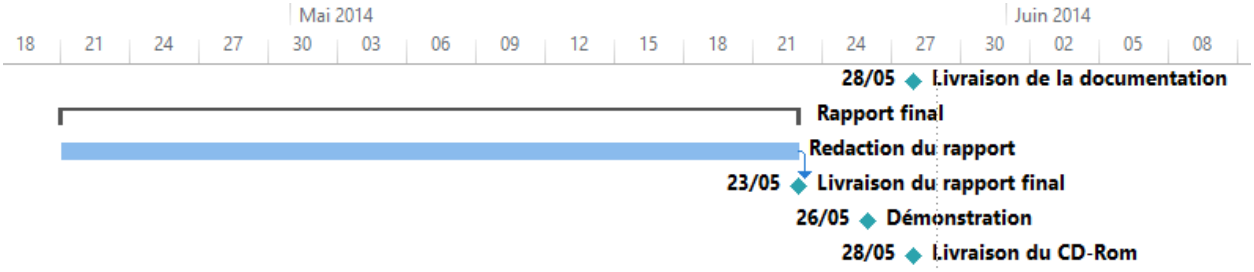

Figure 11 : Diagramme de Gant (5/6) – Fin du projet

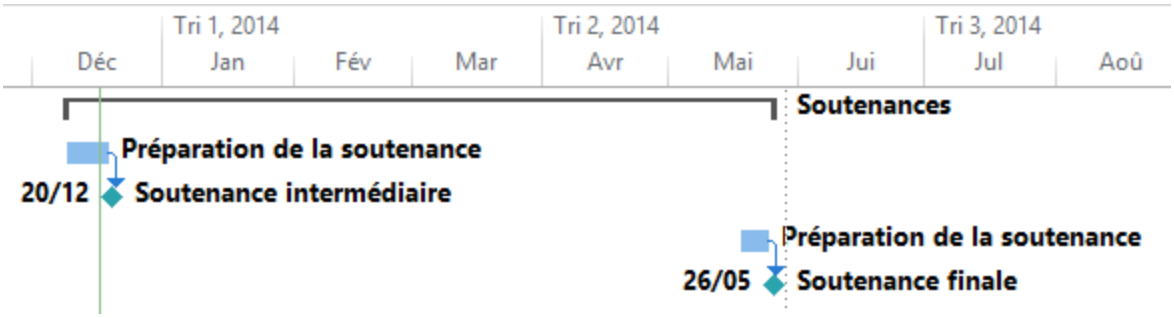

Figure 12 : Diagramme de Gant (6/6) - Soutenances

Institut National des Sciences Appliquées
École publique d'ingénieurs

INSA RENNES

Membre fondateur de l'UEB

UNIVERSITÉ EUROPÉENNE DE BRETAGNE