Équipe 11

Projet Intégrateur de 3e Année Poly Paint
Plan de projet

Version 2.0

Historique des révisions

Date	Version	Description	Auteur
2019-01-15	1.1	Ébauche point 1,2,4	Nicolas Coutu
2019-02-06	1.2	Échéancier (point 5)	Tom Gautier
2019-02-06	1.3	Point 3	Mylene Pellemans
2019-02-06	1.4	Introduction et taille et performance	Maxim Cousineau
2019-02-06	1.5	Point 6	Équipe au complet
2019-04-07	2.0	Révision et correction suite aux commentaires	Tom Gautier

Table des matières

Plan de projet

1. Introduction

Ce présent document établit les bases nécessaires pour l'élaboration du projet. On y trouve une brève description de la tâche demandée, une liste des outils et méthodes utilisés pour assurer la qualité du travail créé, une estimation du temps nécessaires aux principales tâches, une description de l'équipe et une proposition d'entente.

2. Énoncé des travaux

2.1. Solution proposée

L'équipe propose de faire une suite d'applications de conceptions UML répondant aux exigences, composée d'un application C# pour intégrer les nouvelles fonctionnalités demandées, une application Java pour plateforme Android afin de permettre l'édition sur tablette, et un serveur Node pour gérer le travail collaboratif et les autres besoins réseautiques.

2.2. Hypothèses et contraintes

Ce projet repose sur l'hypothèse qu'il prendra un total de 1080 heures-personnes pour livrer le résultat attendu. On suppose également que les exigences ne changeront pas suite à la réponse à l'appel d'offre. Pour ce qui est des contraintes, le projet doit se faire en environ 13 semaines par une équipe de 6 étudiants au baccalauréat. Il faut donc tenir en compte leur horaire de travail hors-projet lors de la définition de l'échéancier et de la charge de travail hebdomadaire. Pour l'équipement, l'équipe aura à sa disposition des locaux réservés avec postes de travail 6 heures par semaines pour l'avancement du projet, en plus d'une tablette Nexus 9 pour les tests du client léger.

2.3. Biens livrables du projet

Pour le 8 février 2019, il faut remettre notre réponse à l'appel d'offre qui sera constituée des prototypes d'application de communication client-serveur et de plusieurs documents. Ces artefacts sont le plan de projet (ici), le SRS, la liste d'exigences, le document d'architecture logicielle et le protocole de communication.

Pour le 8 avril 2019, l'équipe livrera le produit final, qui inclut les applications ainsi que le code source. Accompagné de ce dernier seront les mises à jour des artefacts remis lors de la réponse à l'appel d'offre additionnés du plan de tests et des résultats de test.

3. Gestion et suivi de l'avancement

3.1. Gestion des exigences

Le logiciel de gestion de projet *Redmine* servira à garder une trace du temps consacré et du pourcentage de complétion de chacune des exigences. Pour chaque exigence, l'équipe créera sur *Redmine* une demande correspondant à la notation affichée dans le SRS. Pour les cas plus complexes, la demande sera divisée en sous-tâches.

Pour la création de demande, chaque membre de l'équipe s'engage à l'écrire comme suit:

- mettre un sujet pertinent et descriptif;
- préciser un temps estimé;
- préciser une date de début et une date d'échéance;
- l'assigner à un membre de l'équipe;
- être liée à une version cible.

Ceci dit, l'équipe 11 s'est entendu sur une convention quant à la nomenclature des titres des demandes sur *Redmine*. Les syntaxes de titres des demandes sont présentées ci-dessous:

- Pour la rédaction des artefacts de documentation:
  Syntaxe: DOC - <Nom de l'artefact> - <sujet>
  où <Nom de l'artefact> peut être: SRS, architecture logiciel (Arch. log.), Plan de projet, Protocole communication (Protocole comm.)

- Pour le client lourd:
  Syntaxe: HEAVY - <sujet>
  où HEAVY: Client lourd

- Pour le client léger:
  Syntaxe: LIGHT - <sujet>
  où LIGHT: Client léger

- Pour le serveur:
  Syntaxe: SERVER - <sujet>
  où SERVER: Serveur

- Pour les demandes qui deviennent non-pertinents pour une quelconque raison, mettre "**ANNULÉE -**" au début de la demande originale. Ceci est pour éviter de supprimer des demandes et pour garder un historique des demandes. C'est aussi pour éviter une confusion au sein de l'équipe, à savoir si nous faisons toujours une demande ou non, ou pour éviter qu'un membre croit qu'une tâche désuète n'a tout simplement pas été mise sur *Redmine.*

- Pour les activités de gestion:
  Syntaxe: PM - <Activité> - <sujet>
  où PM: Planification Management

Le risque de changement d'une exigence est minime étant donné que le cahier des charges fourni par le client est complet et qu'elles ont été établies de manière très précise dans le SRS. Afin de gérer une incertitude sur une exigence, nous avons un canal "Questions" sur Slack permettant aux membres d'exprimer leurs doutes afin d'en discuter durant la réunion hebdomadaire avec le client.
Si une exigence est amenée à changer pour une raison quelconque, nous suivrons les étapes suivantes:

- Vérification des implications dans l'architecture;
- Nouvelle estimation de la charge pour réaliser cette exigence;
- Mise à jour de la planification, si le développement de l'exigence était déjà prévu.

L'équipe 11 prévoit se rencontrer au moins 2 fois par semaine, soit le mardi et le vendredi après-midi. Les applications *Slack* et *Messenger* seront utilisées pour les membres qui ne peuvent pas être présents en personne lors des rencontres.

3.2. Contrôle de la qualité

Pour assurer la qualité du travail, des rencontres hebdomadaires sont organisées pour faire le retour sur le travail effectué et pour planifier les tâches à venir. Après chaque mise à jour importante du logiciel, une revue par les pairs est de mise le plus tôt possible pour soulever des erreurs qui auraient passé les tests.

S'il faut prendre une action corrective, la première étape est d'informer l'auteur de la faute ainsi que le problème soulevé avec des pistes de solutions. Une fois la correction appliquée, il faut tester le produit pour assurer qu'aucun autre bug n'ait été introduit. Si les tests passent sans fautes, il faut passer par la correction par les pairs pour assurer que tout fonctionne comme voulu.

3.3. Gestion de risque

La description des risques suit la convention suivante :

- Ampleur : sur une échelle de 1 à 10, 10 étant le risque le plus élevé. Cette analyse est basée sur la probabilité d'occurrence du risque, ainsi que ses impacts.
- Description : une description textuelle du risque ainsi que les problèmes attendus.
- Impact : échelle définissant la portée du risque
    - C – critique (affecte le projet en entier)
    - E – élevé (affecte les fonctionnalités principales du système)
    - M – moyen (devrait être maîtrisable en appliquant une stratégie d'atténuation adéquate)
    - F – faible (l'acceptation du risque est une stratégie envisageable)
- Facteurs : aspects (**métriques**) du système pouvant être compromis.
- Stratégie de gestion : mesures à prendre afin de gérer le risque.

1 - Problème de communication entre un ou des membres de l'équipe				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
4	Impossible de rejoindre un coéquipier à l'extérieur des réunions.	M	Disponibilité	Diversifier les méthodes de contact pour diminuer le risque.

2 - Consistance entre le client lourd et le client léger				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
1	Les interfaces utilisateurs entre le client lourd et le client léger semblent appartenir à deux équipes différentes au lieu de ressembler à un travail commun.	F	Cohérence	Communication constante entre les membres lors de la conception et le développement des interfaces.

3 - Une tâche bloquée par une autre				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
8	Une tâche en cours qui prend du retard empêche l'avancement d'une autre tâche.	M	Ponctualité	Structurer le projet pour séparer les tâches dépendantes dans différents sprints.

4 - Problème d'intégration des librairies				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Une librairie cause un problème avec l'architecture du logiciel ou avec l'intégration d'une autre librairie.	E	Complétude	Faire des recherches sur les problèmes de compatibilité des différentes librairies avant de les utiliser.

5 - Technologies non-compatibles				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
4	Les technologies des différents clients et du serveur ne fonctionnent pas ensemble comme espéré.	C		Faire des investigations technologiques afin de s'assurer de la compatibilités des modules. Les prototypes servent de preuve de concept à ce niveau-là.

3.4. Gestion de configuration

La gestion de configuration se fait lors des rencontres hebdomadaires. Lorsqu'un point est à soulever, il est transmis aux membres par les moyens de communication pour être ajouté à l'ordre du jour. En cas d'urgence, il est possible de prendre la décision en ligne, en s'assurant que chaque membre ait approuvé le changement.

Les normes de nommage pour les artéfacts sont plutôt simples : leur nom doit toujours être le type de document et son numéro de version suit la syntaxe suivante : X.Y où X correspond à la version à remettre et Y indique le nombre d'édition ayant été faites pour arriver au document actuel. Donc, pour la réponse à l'appel d'offre, les artéfacts devraient être de version 1.Y et 2.Y lors de la remise finale.

Nous utiliserons Github comme plateforme de gestion de versions. Il y a 2 branches principales:
- *master* qui comprend uniquement des versions stables et fonctionnelles;
- *dev* qui sera utilisée pour l'intégration continue. Chaque sous-branche, qu'elle soit destinée à une nouvelle fonctionnalité, un *refactoring*, ou un bug résolu, devra partir de *dev* afin de pouvoir par la suite être intégrée. Cette intégration sera faite via une *Pull Request* sur Github, permettant aux membres concernés de s'assurer du développement qui a été réalisé.

Il est important de préciser qu'une approbation de la *Pull Request* est nécessaire afin de *push* sur *dev* tandis qu'il en faut deux pour fusionner *dev* avec *master*. De même, avant une remise importante, nous

ferons un *push* de notre branche *master* de Github sur le Git utilisé par le client.

4. Échéancier du projet

4.1. Calendrier

Nous avons élaboré un calendrier divisé en 4 itérations de 2 semaines chacune, jusqu'au jour de la remise finale.

Itération 1	12 fev - 18 fev	19 fev - 25 fev
Itération 2	26 fev - 04 mars	05 mar - 11 mar
Itération 3	12 mar - 18 mar	19 mar - 25 mar
Itération 4*	26 mar - 1 avr	2 avr - 8 avr
Remise finale	8 avril	
Présentation finale	9 avril	

* La première semaine de la 4e itération sera réservée à la finalisation de certaines exigences souhaitables ainsi qu'à l'intégration. La deuxième semaine de la 4e itération est réservée exclusivement aux tests. Cela explique une charge de travail beaucoup plus faible durant celle-ci dans l'échéancier.

4.2. Échéancier

4.2.1 Échéancier statistique

Le tableau suivant nous permet d'avoir une vision globale sur la planification du projet.

		Itération 1	**Itération 2**	**Itération 3**	**Itération 4**	**Projet**
HC	Charge (h-p)	108	112	104	32	356
	Charge (%)	30%	31%	29%	9%	
	Avancement cumulatif (%)	30%	62%	91%	100%	
LC	Charge (h-p)	104	128	104	28	364
	Charge (%)	29%	35%	29%	8%	
	Avancement cumulatif (%)	29%	64%	92%	100%	
Projet	Charge (h-p)	212	240	208	60	720
	Charge (%)	29%	33%	29%	8%	
	Avancement cumulatif (%)	29%	63%	92%	100%	

On remarque rapidement que la charge de travail prévu pour l'itération 4 est plus faible que pour les autres, dû aux phases d'intégration et de test mentionnées au point 4.1.
De même, on peut noter que la charge totale de développement pur est planifiée à 720 heures-personne. À cette charge, on ajoute un estimé 180 heures-personne dédiées aux activités de test et 60 heures-personne dédiées à la documentation du livrable final, totalisant le travail de développement à 960 heures-personne.
Finalement, on estime 120 heures-personne de gestion de projet, totalisant le projet à une valeur de 1080 heures-personne.

4.2.2 Échéancier détaillé

Itération	Client	Lot	Tâche	Valeur (h-p)
Itération 1	HC	Clavardage	Intégration à même l'environnement	16
		Clavardage	Canaux de discussion multiples	8
		Édition de base collaborative	Enlever fonctionnalités et les adapter pour l'édition collaborative	20
		Édition de base collaborative	Implémenter les nouvelles fonctionnalités	20
		Édition de base collaborative	Implémenter le mode collaboratif	44
	LC	Clavardage	Intégration à même l'environnement	16
		Clavardage	Canaux de discussion multiples	8
		Édition de base collaborative	Implémenter les fonctionnalités existantes	40
		Édition de base collaborative	Implémenter les nouvelles fonctionnalités	40
Itération 2	HC	Galerie	Galerie d'images publiques	16
		Galerie	Galerie d'images privées	8
		Édition de formes collaborative	Rotation	44
		Édition de formes collaborative	Redimensionnement	44
	LC	Galerie	Galerie d'images publiques	16
		Galerie	Galerie d'images privées	8
		Édition de base collaborative	Connecter les deux usagers à la session	20
		Édition de base collaborative	Permettre aux deux utilisateurs de collaborer	20
		Édition de base collaborative	Gérer les différents cas de conflits	20
		Édition de formes collaborative	Formes de diagrammes et formes de connexion	40
		Interface utilisateur	Utilisation d'une gesture	4

Itération 3	HC	Clavardage	Alterner mode fenêtré ou mode intégré	8
		Édition de formes collaborative	Finaliser intégration	16
		Sauvegarde image et chargement	Sauvegarde et chargement distant	24
		Accessibilité des images	Mode protégé activable et désactivable	12
		Tutoriel	Tutoriel non-interactif	8
		Amis	Ajouter et retirer des amis	16
		Langues	Changer langue d'affichage et langue préférence	20
	LC	Édition de formes collaborative	Personnalisation des formes	40
		Sauvegarde d'image et chargement	Sauvegarde et chargement distant	8
		Accessibilité des images	Mode protégé	4
		Effets visuels et sonores	Un groupe d'effets présent.	4
		Tutoriel	Tutoriel non-interactif	8
		Interface utilisateur	Permettre deux gestures concurrentes	8
		Amis	Ajouter et retirer des amis	16
		Amis	Liste amis en ligne	8
		Chat	Envoyer invitation session collaborative	8
Itération 4	HC	Connection	Mot de passe oublié	24
		Amis	Liste amis en ligne	8
	LC	Profil utilisateur et galerie	Association d'images à des tags	12
		Profil utilisateur et galerie	Barre de recherche d'images par tag	8
		Profil utilisateur et galerie	Aimer une image et retirer le "j'aime"	8

5. Équipe de développement

5.1. Tristan Beaulieu

Tristan possède de bonnes aptitudes en C# ainsi qu'en Java. Toutefois, son expérience de travail en conception mobile le place en bonne position pour participer au développement du client léger. Naturellement, il sait travailler en équipe et résoudre efficacement des problèmes variés. Il pourrait ainsi être amené à supporter l'équipe du client lourd en cas de besoin.

5.2. Olivier Boivin

Olivier possède une bonne connaissance en C# et effectuera donc principalement des tâches reliées au développement du client lourd. Ayant aussi une base en Java et une bonne capacité à résoudre des problèmes de codage, Olivier pourra être amené à fournir de l'aide à l'équipe de développement du client léger lorsque nécessaire.

5.3. Maxim Cousineau

Maxim a un peu d'expérience avec Java et participera donc principalement au développement du client léger. Il possède aussi de l'expérience en ce qui a trait à la conception et à l'architecture logicielle, ce qui lui permettra d'assurer une bonne structure du code source. De plus, Maxim tend à apprendre de nouvelles technologies de développement plutôt rapidement et peut donc servir de guide envers les autres développeurs s'ils éprouvent des difficultés dans leur apprentissage.

5.4. Nicolas Coutu

Nicolas possède beaucoup d'expérience en C# .Net et donc est assigné au développement du client lourd. Il a aussi un peu d'expérience en serveur et en développement mobile, et peut donc aider au besoin dans ces technologies. Il a également assez d'expérience en gestion d'équipe pour supporter le projet dans la bonne direction.

5.5. Tom Gautier

Tom a une base solide en Node.js et Java, raison pour laquelle il sera principalement basé sur le développement du serveur et de l'application mobile. C'est une personne versatile qui pourra facilement s'adapter aux besoins immédiats de l'équipe. Tom a de l'expérience en gestion de projets ce qui lui permet de garder une vision globale du projet et de s'assurer que l'équipe est sur la bonne voie.

5.6. Mylène Pellemans

Mylène possède une bonne connaissance en ce qui a trait à la langue française. Elle a de l'expérience pour programmer des éléments audio dans une application. Le principal système d'exploitation qu'elle utilise est Windows, quoiqu'elle a aussi programmé avec Linux. Mylène n'a pas d'expérience pour programmer des applications mobiles, donc elle va concentrer ses efforts en partie sur le client lourd et le serveur. De plus, elle priorise une bonne qualité du code, telle qu'un nommage approprié des variables.

6. Entente contractuelle proposée

Puisque nous connaissons clairement les spécifications techniques demandées et que nous avons une échéance fixe, nous proposons donc un contrat clé en main. Ainsi, l'autorité contractante de Polytechnique Montréal et l'équipe 11 s'engage à respecter les modalités qui suivent:

- L'équipe 11 s'engage à respecter les échéanciers décrits dans ce document.
- L'équipe 11 livrera le produit final au plus tard le 8 avril 2019.
- L'équipe 11 consent à respecter le droit de propriété intellectuelle déterminé par l'autorité contractante.
- L'équipe 11 s'engage à respecter la charge de travail requise pour la livraison du projet.
- L'équipe 11 respectera l'article 8 du document *Règlements des études du baccalauréat en ingénierie pour l'année 2018-2019,* portant sur la fraude.

Pour l'équipe 11, qui est composée de 6 membres, la charge de travail requise pour la livraison du projet est de 1080 heures-personnes. De cette charge, notre équipe s'est entendu pour dire que la gestion de projet prendra 12% du temps. Ainsi, selon le tableau ci-dessous, le coût de la charge à payer dans le cadre du projet sera de 111000$.

Tableau 1: Prix demandé en fonction de la charge de travail

Lot de travail	**Charge de travail (h-p)**	**Tarifs par heure ($/h)**	**Coût de la charge ($)**
Développeur	960	100	96000
Gestionnaire de projet	120	125	15000
		Total:	111000