

Université Paul Sabatier – Toulouse III
IUT A - Toulouse Rangueil
Projet tutoré

Antoine de ROQUEMAUREL
Mathieu SOUM
Geoffroy SUBIAS
Marie-Ly TANG
Groupe B

Pour Monsieur MILLAN (Client)
Pour Madame KROSS (Tuteur)
Pour Monsieur MARQUIÉ (Correcteur)

Cahier des Charges Fonctionnel

Bibliothèque d'objets graphiques UML

Toulouse, le 20 octobre 2011

Table des matières

1 Contexte

1.1 Présentation du groupe projet

Notre groupe projet est composé de quatre étudiants de deuxième année de DUT¹ Informatique à l'IUT² A de Toulouse, voici la composition de l'équipe :

- Antoine de ROQUEMAUREL
- Mathieu SOUM
- Geoffroy SUBIAS
- Marie-Ly TANG

Nous avons monté ce groupe, car nos compétences sont complémentaires et que nous savons déjà comment chacun travaille. Antoine de ROQUEMAUREL et Mathieu SOUM sont spécialisés en programmation par objet, Geoffroy SUBIAS est le plus compétent lorsqu'il s'agit de modélisation UML³ et Marie-Ly TANG s'occupera principalement de l'organisation et de la gestion de projet. Nos compétences sont différentes mais sont complémentaires pour mener à bien notre projet.

1.2 Présentation du commanditaire

Monsieur Thierry MILLAN est un enseignant à l'IUT A Toulouse et chercheur à l'IRIT⁴

1.3 Présentation du projet

L'objectif du projet est de réaliser une bibliothèque d'objets graphiques représentant les différents éléments de modélisation de la norme UML 2.

1.3.1 La norme UML

UML est un langage de modélisation graphique à base de pictogramme simples, chacun représentant un concept particulier.

Ce langage est utilisée dans le cadre de la conception orientée objet, il est divisé en plusieurs diagrammes, en voici quelques exemple (la liste n'est pas exhaustive) :

- Diagramme de classes
- Diagramme des cas d'utilisations
- Diagramme de séquence
- Diagramme de composants

Dans le cadre du projet, nous devons créer une bibliothèque graphique permettant de dessiner différents pictogrammes.

1.3.2 Réutilisation de la bibliothèque

La bibliothèque devra pouvoir être réutilisée en tant que composant dans d'autres logiciels, chaque objet graphique de la librairie pourra être intégré séparément à d'autres projets, et ainsi être réutilisée dans d'autres objectifs ou en association avec d'autres composants.

1.3.3 Propreté du code

Le projet ayant dans l'optique une réutilisation du résultat dans d'autres applications, le client souhaite avoir un code propre et optimisé (factorisé) afin qu'il soit le plus rapide et le plus abordable

1. Diplôme Universitaire de Technologie

2. Institut Universitaire de Technologie

3. Unified Modelling Language

4. Institut de Recherche Informatique de Toulouse

possible pour les futurs utilisateurs de notre composant. De plus, une documentation sera générée via l'outil Javadoc et devra être elle aussi complète, détaillée et facile à comprendre.

1.3.4 Risques

| Risques | Pertinence | Solution | Responsable |
|---|------------|--|-------------|
| Évolution du besoin du client | Moyenne | Nous travaillerons par incréments, en rencontrant régulièrement le client nous aurons le temps d'implémenter ses besoins et éviter les demandes de dernières minutes | Marie-Ly |
| Non respect du besoin du client | Moyenne | Voir le client régulièrement (environ toutes les deux semaines) | Marie-Ly |
| Retard du projet | Haute | Respecter scrupuleusement le planning et le Gantt | Geoffroy |
| Limite des compétences | Haute | Se renseigner en autodidactie (internet) | Geoffroy |
| Mauvaise coordination entraînant des divergences de développement | Moyenne | Utiliser une plateforme de travail collaboratif (Redmine) afin que chaque membre soit au courant des évolutions du projet | Antoine |
| Crash du disque dur contenant le projet | Faible | Avoir le projet sur plusieurs périphériques | Mathieu |
| Indisponibilité du serveur permettant le travail collaboratif | Moyenne | Héberger le serveur à domicile pour effectuer une maintenance rapide. Ajout d'un onduleur | Antoine |

2 Description de la demande

Le cahier des charges fonctionnel sera évolutif car le projet sera incrémental et chaque incrément devra être validé par le client. Le cycle de développement sera un cycle à incrément court (Deux à trois semaines).

Le logiciel sera codé en Java et devra être utilisable comme composant par d'autres logiciels.

Objectif du client à court terme Au terme de notre premier incrément, l'objectif sera de pouvoir dessiner des diagrammes de classes reprenant la plupart des pictogrammes, sans aucune contrainte vis-à-vis de la norme UML 2.0.

Objectif du client à long terme Le projet une fois terminé devra permettre à l'utilisateur de dessiner des diagrammes UML de séquence ou de classes.

Selon l'évolution du projet, le client se réserve le droit de modifier ces conditions pour y intégrer des contraintes vis-à-vis de la norme UML 2.0 et de différencier les types de diagramme lors de leur conception. Chaque diagramme sera un composant à part et pourra donc être distingué comme composant indépendant.

L'équipe de projet a la possibilité de se renseigner sur d'éventuels logiciels existant et allant le même but afin de s'en inspirer ou de récupérer des morceaux de codes pouvant être intégrés au projet, si le logiciel existant distribue son code source librement.

Le client propose d'utiliser la librairie java JGraph permettant de dessiner des pictogrammes simples, cependant l'équipe peut décider d'utiliser une autre librairie si elle en trouve une plus simple ou qui correspond mieux aux attentes du client.

2.1 Évaluation des fonctions

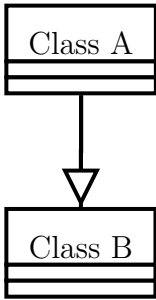
2.1.1 Fonctions principales

Les fonction principales, sont les fonctionnalités attendu lors de l'utilisation du logiciel.

| Référence | Fonction | Critères d'appréciations | Niveau | Flexibilité |
|-----------|---|--|---|-----------------|
| FP1 | Permet de dessiner un diagramme de classes | La durée pour effectuer un diagramme de 10 classes | 10 minutes | ± 5 minutes |
| FP2 | Permet de dessiner un diagramme de séquence | La durée pour effectuer un diagramme de 5 objets | 10 minutes | ± 5 minutes |
| FP3 | Avoir un démonstrateur permettant de tester la bibliothèque | Toutes les fonctionnalités disponibles doivent pouvoir être testés | Toutes les fonctionnalités implémentées au moment du test | Aucune |

2.1.2 Fonctions contraintes

Les fonctions contraintes, sont à mettre en place pour améliorer l'utilisation du logiciel.

| Référence | Fonction | Critères d'appréciations | Niveau | Flexibilité |
|-----------|----------------------|---|--------------------------------------|-------------------|
| FC1 | Rapidité et légèreté | Peu gourmand en mémoire | 10% d'une RAM ⁵ de 2 Go | ± 3 % |
| FC2 | Portabilité | Utilisable sur différents systèmes d'exploitation | Fonctionne sur Windows, MacOS, Linux | Aucune |
| FC3 | Ergonomie | Nombre de clics pour un élément simple :  | 5 clics | ± 2 |
| FC4 | Documentation | Temps passé pour trouver la documentation se rapportant à une méthode | 30 secondes | ± 10 secondes |
| FC5 | Propreté du code | Complexité cyclomatique | 15 | +5 |

5. Random Access Memory

3 Les contraintes

3.1 Contraintes de délais

Afin de bien s'organiser, nous avons décidé de choisir des horaires fixes de réunions. L'équipe de projet se réunira tous les jeudis entre 11h et 12h30. Nous rencontrerons le client un lundi sur deux à 17 heures afin de valider l'incrément et d'évaluer les besoins de l'incrément suivant et enfin nous verrons notre tuteur un mercredi toutes les deux semaines à 13 heure. Pour plus de détails sur la gestion du temps dans le projet, veuillez vous reporter à l'annexe ?? page ??.

3.2 Contraintes de ressources humaines

Il ne sera pas tenu rigueur de tout retard pris sur des tâches dépendantes du client.

- Le client doit valider chaque incrément et nous donner les directives pour l'incrément suivant afin de pouvoir continuer le projet. Cette étape sera réalisée lors des réunions citées un paragraphe plus haut.
- Nous n'avons pas d'horaires aménagés pour le projet et nous n'habitons pas tous dans la même ville, ce qui peut poser des difficultés pour se voir, cependant nous aurons la possibilité de travailler à distance et le jeudi midi.

3.3 Contraintes de ressources matérielles

- Le projet devra être programmé en Java et le rendu devra s'effectuer sous l'EDI⁶ Netbeans.
- Il devra être possible d'intégrer la bibliothèque en tant que composant dans d'autres logiciels.
- La documentation relative au projet ne devra pas être éditée sur support papier, nous utiliserons donc Javadoc pour produire une documentation au format HTML⁷.

Signatures

Client :

Le
A

Groupe :

Le
A

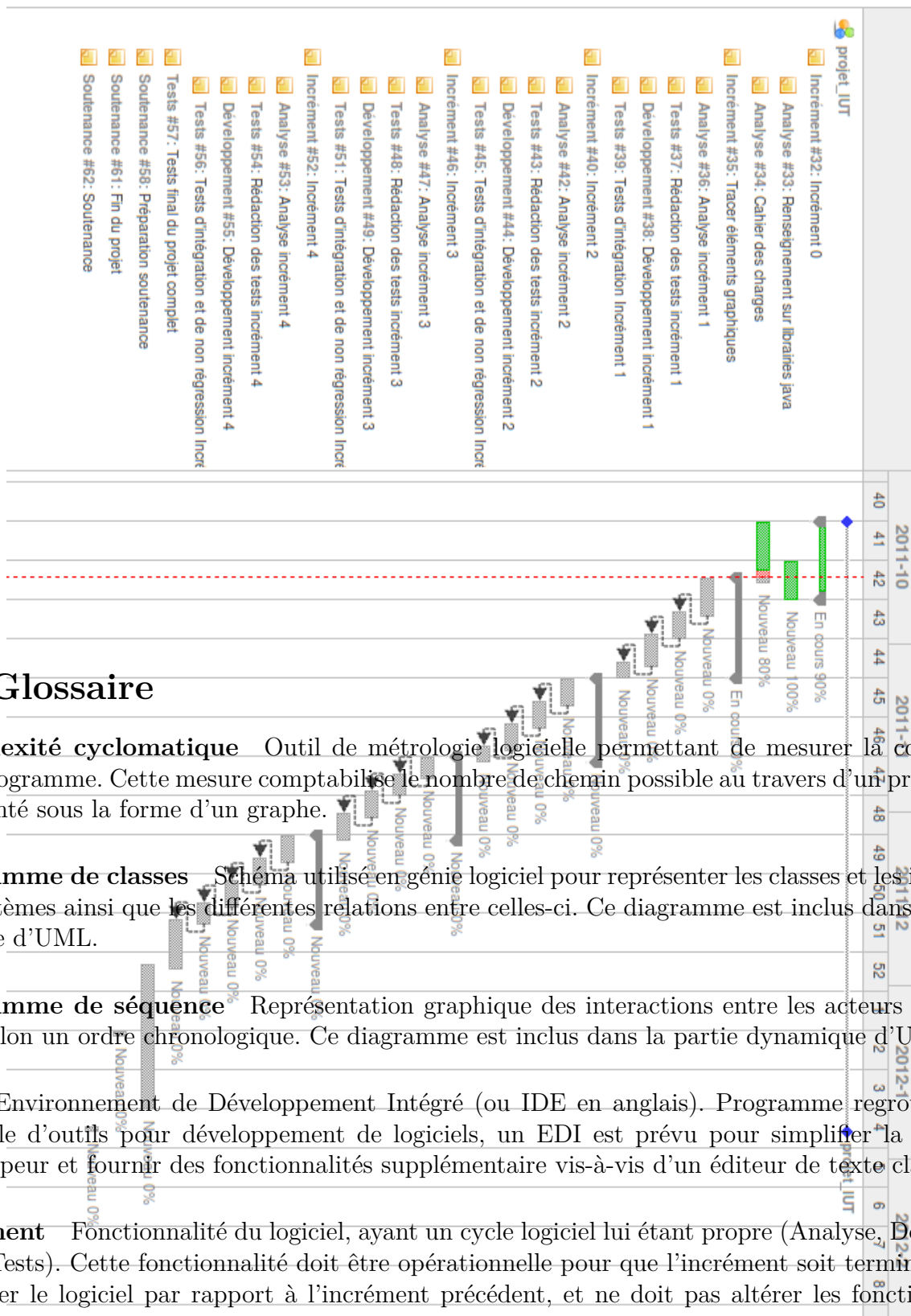
Tuteur :

Le
A

6. Environnement de Développement Intégré

7. HyperText Markup Language

A Diagramme de Gantt



B Glossaire

Complexité cyclomatique Outil de métrologie logicielle permettant de mesurer la complexité d'un programme. Cette mesure comptabilise le nombre de chemin possible au travers d'un programme représenté sous la forme d'un graphe.

Diagramme de classes Schéma utilisé en génie logiciel pour représenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci. Ce diagramme est inclus dans la partie statique d'UML.

Diagramme de séquence Représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique. Ce diagramme est inclus dans la partie dynamique d'UML.

EDI Environnement de Développement Intégré (ou IDE en anglais). Programme regroupant un ensemble d'outils pour développement de logiciels, un EDI est prévu pour simplifier la tâche du développeur et fournir des fonctionnalités supplémentaire vis-à-vis d'un éditeur de texte classique.

Incrément Fonctionnalité du logiciel, ayant un cycle logiciel lui étant propre (Analyse, Développement, Tests). Cette fonctionnalité doit être opérationnelle pour que l'incrément soit terminé. Il doit améliorer le logiciel par rapport à l'incrément précédent, et ne doit pas altérer les fonctionnalités précédentes.

Java Langage de programmation orienté objet moderne, il compile le programme pour ensuite l'exécuter sur une machine Java, ainsi le programme une fois compilé peut être exécuté sur différentes plateforme (Windows, Linux, Mac OSX, ...).

UML (Unified Modeling Language) Langage de modélisation graphique à base de pictogramme. Il est apparu dans un monde du génie logiciel dans le cadre de la conception orientée objet. Ce langage est composés de différents diagrammes, allant du développement à la simple analyse des besoins.

C Bibliographie

Références

- [1] Thierry MILLAN Cours sur la modélisation UML, 2011
- [2] Thierry MILLAN Cours sur la qualité logicielle, 2011
- [3] Article sur UML http://fr.wikipedia.org/wiki/Unified_Modeling_Language, 2011