Carnet de bord

Semaine du 16 Novembre au 21 Novembre 2009 :

Nous avons pris connaissance du projet à l'aide d'une réunion avec notre tuteur de projet. Après quelques réflexions, nous avons réussi à bien cerner le sujet. Nous avons établi un planning en prenant en compte toutes les phases du projet, de l'analyse au rendu de celui-ci.

Nous avons, en parallèle, commencé un rapport préliminaire pour notre tuteur afin de mettre en évidence les points forts de ce projet, ainsi que les premières analyses.

Semaine du 23 Novembre au 5 Décembre 2009 :

Nous avons commencé l'analyse du projet, pour cela, nous avons, dans un premier temps réalisé un diagramme de use case afin de cerner les principales fonctionnalités du logiciel. Une fois ce diagramme réalisé, nous avons fait un diagramme de classe permettant de visualiser les différents composants utiles à la réalisation du logiciel. Et enfin, grâce à celui-ci nous avons pu réfléchir à la meilleure façon de relier les objets entre eux.

Enfin, lors de nos rendez-vous de projet, nous avons discuté à propos du rapport préliminaire pour approfondir certaines idées et les développer un peu plus.

Semaine du 7 Décembre au 12 Décembre 2009 :

A partir de notre diagramme de classe d'analyse, nous avons pu faire un diagramme de classe de conception, celui-ci intégrant les diverses idées contenues dans notre rapport préliminaire telles que :

- L'architecture du logiciel basée sur une programmation en trois couches utilisant le pattern MVC (modèle, vue, contrôleur).
- La notion de graphes en intégrant une structure de graphe de type matrice.
- Une bibliothèque de graphes par le biais d'une collection de graphes.
- Une bibliothèque d'algorithmes par le biais d'une collection d'algorithmes.

De plus, nous avons également commencé la réalisation, en créant, dans un premier temps l'architecture du logiciel, une structure de graphe abstraite contenant les fonctions essentielles concernant les graphes, telles que la liste des successeurs d'un sommet du graphe, les fonctions d'ajout et de suppression d'arcs et de nœuds. Nous avons également implémenté une structure de graphe (représentation par matrice) en redéfinissant les méthodes essentielles d'un graphe. La librairie de graphe a également été réalisée.

Des recherches concernant l'affichage des graphes à l'écran ont été faites. Parmi ces recherches, nous avons retenu deux méthodes:

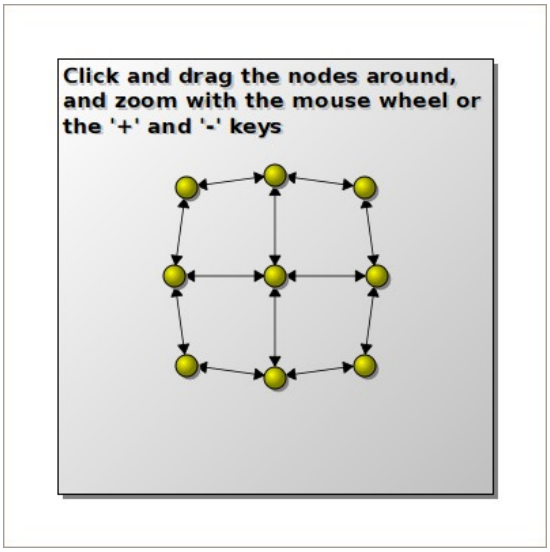
- La retranscription d'une libraire de représentation de graphe déjà existante qu'est Prefuse (librairie développée en Java et open source http://prefuse.org/).
- L'utilisation d'un des exemples de la librairie graphique Qt qu'est Elastic nodes (http://doc.trolltech.com/4.3/graphicsview-elasticnodes.html)

Semaine du 14 Décembre au 19 Décembre :

Nous avons améliorer notre objet représentant les graphes en redéfinissant chacune de ses méthodes à l'aide principalement de la méthode permettant de savoir si un sommet a est successeur d'un sommet b. Grâce à cela, il nous est maintenant possible d'implémenter une nouvelle structure de graphe en redéfinissant seulement la méthode isSuccessor ainsi que les méthodes d'ajout et de suppression de nœuds et d'arcs. Cela nous permet donc une évolution beaucoup plus aisé du logiciel.

De plus nous avons également travailler sur la partie graphique du logiciel en créant une fenêtre permettant la visualisation d'un graphe à l'aide de cercles représentant les nœuds et de flèches représentant les arcs. Pour cela nous nous sommes appuyé sur un des exemples de la librairie Qt : Elastic nodes. Nous l'avons donc adapté à notre logiciel et notre logiciel est donc maintenant en mesure d'afficher un graphe à l'écran.

Ci dessous : la capture d'écran de Elastic nodes, l'exemple de la librairie Qt



Semaine du 21 Décembre au 1 Janvier 2010 :

Lors de cette semaine, nous avons réfléchi sur le développement des algorithmes et donc des principales fonctions nécessaires au développement d'algorithmes supplémentaires. Nous avons donc implémenté des fonctions permettant d'assigner un graphe à l'algorithme, une fonction permettant de choisir un nœud parmi une liste de nœuds calculée par l'algorithme de diverses manières.
Cette dernière fonction s'appuie sur quatre types de choix :

- FIFO (First In First Out) : Cette option permet de choisir automatiquement la première valeur de la liste de nœuds.
- LIFO (Last In First Out) : Cette option permet de choisir automatiquement la dernière valeur de la liste de nœuds.
- RDM (Aléatoire) : Cette option permet de choisir automatiquement une valeur aléatoirement parmi la liste de nœuds.
- USER (Utilisateur) : Cette option permet à l'utilisateur de choisir le sommet parmi la liste de nœud. Ce choix se fait actuellement en mode console, mais à terme, il se fera par le biais d'un clic sur le nœud choisi sur l'interface graphique du logiciel.

Afin de tester ces fonctions utiles, nous avons implémenté un algorithme de descente en

profondeur d'un graphe. Celui-ci permet de trouver l'ensemble des descendants d'un nœud. L'idée de cet algorithme est de suivre un chemin le plus loin possible, puis de revenir en arrière jusqu'à trouver une arrête menant à un nœud non exploré.

Pour faciliter la création et le chargement de nos algorithmes, nous voulions le charger de façon dynamique (c'est-à-dire sans avoir à recompiler le logiciel à chaque ajout d 'algorithmes). Pour ce faire, nous avons effectué des recherches et en avons retiré une première solution qui est l'utilisation de la fonction *dlopen* et *dlsym* (cf http://www.linux-kheops.com/doc/man/manfr/man-html-0.9/man3/dlopen.3.html). A ce stade, uniquement les recherches ont été effectuées et quelques tests sur des exemples dissociés du projet ont été réalisés.

Semaine du 4 Janvier au 9 Janvier :

Durant cette semaine de reprise, nous avons eu quelques soucis de type organisationnel, afin de les résoudre nous avons décidé de mettre fin au binôme et de partir sur deux projets séparés. Pour ma part, je poursuivrai ce projet. Après ce petit désagrément, des recherches ont été effectuées sur la sérialisation xml qui permettra donc à l'utilisateur de sauvegarder ses graphes. Pour ce faire, la librairie TinyXml (cf http://www.grinninglizard.com/tinyxml/) sera utilisée. De même que pour le chargement dynamique des algorithmes, les recherches ont été effectuées ainsi que quelques tests sur des exemples hors contexte.

Lors du rendez-vous pédagogique, un problème a été soulevé, celui des arcs valués. En effet, le modèle implémenté n'incluait pas le système de valuation des arcs, or ceci peut être utilisé lors de différents algorithmes tel que l'algorithme de Dijkstra qui permet de trouver le chemin le plus court pour aller d'un sommet à un autre. Cette amélioration devra donc être implémentée par la suite.

Ensuite il a été important de pouvoir faire une première version utilisable du logiciel uniquement en mode fenêtre, c'est-à-dire se focaliser sur le choix des nœuds à l'aide de la visualisation du graphe et de représenter celui-ci à chaque étape d'un algorithme. Des améliorations ont donc été réalisées sur le logiciel afin que celui-ci mette à jour automatiquement la fenêtre et que l'on puisse choisir un nœud uniquement en cliquant sur celui-ci.

Pour cela, des écouteurs de souris ont été rajoutés. De plus afin que la fenêtre garde sa fluidité et ne soit pas saccadée par les différentes étapes de l'algorithme, un thread a été réalisé. Celui ci permet en effet de créer, lors du lancement d'un algorithme un nouveau thread permettant, en conséquent de faire tourner l'algorithme dans un autre processus que celui de l'affichage.

Semaine du 11 Janvier au 16 Janvier :

Tout d'abords, comme il avait été précisé la semaine passée, la structure de graphe devait pouvoir permettre d'attribuer des valeurs à des arrêtes. Ceci a donc été ajouté durant cette semaine. De plus, cette semaine a surtout été le fruit de recherches et de test concernant le chargement dynamique de nos algorithmes. Après de multiples tentatives suite à diverses lectures et à divers avis de la part de professeurs, le logiciel est capable de charger les objets de façon dynamique, cependant, un problème survient lors de l'appel de fonctions issues de l'héritage. De ce fait, le logiciel n'est pas encore en mesure de faire tourner entièrement un algorithme chargé dynamiquement. Par ailleurs il est toujours possible d'exécuter des algorithmes on recompilant à chaque ajout le logiciel.

Enfin, de manière à rendre l'interface graphique plus intuitive et plus attractive, les fonctionnalités d'ajout de nœuds et d'arêtes ont été rajoutés. L'ajout de nœud se fait lors d'un double clic dans la zone graphique, tandis que l'ajout d'arêtes se fait par le clic simultané de deux nœuds de la zone graphique.

Semaine du 18 Janvier au 23 Janvier :

Cette semaine ci était réservée pour nos partiels, de ce fait, le logiciel n'a pas eu de modifications tout au long de la semaine.

Semaine du 25 Janvier au 30 Janvier :

Suite à de nombreux bugs dus en partie à la mise à jour d'une version bêta de la librairie graphique Qt, cette semaine m'a été utile pour corriger le maximum de bugs du logiciel. Dans un premier temps, nous sommes repassé sur une version stable de la librairie graphique Qt. Grâce à cela, les bugs ont donc pus être identifiés, ainsi que leur provenance. La plupart des bugs faisaient planter le programme des fois sans aucunes actions de la part de l'utilisateur, des fois lorsque l'utilisateur essayait de créer ou d'importer un graphe. Ces bugs ont donc étés corrigés. A ce jour aucun autres bugs n'a été répertorié.

Semaine du 01 Février au 06 Février :

Lors de cette semaine, peu d'amélioration ont été apportées sur le projet, cependant, une réflexion sur le rapport final a été apporté. En effet, un plan détaillé a été rédigé, et quelques paragraphes ont été remplis.

Concernant le projet, la visualisation des arcs par lesquels on est passé à été rajouté. Grâce à cela, il nous est maintenant possible de visualiser par exemple l'arbre de descente lors de l'exécution d'une descente en profondeur. De plus un algorithme a été rajouté, celui-ci permet maintenant de calculer la numérotation compatible dans un graphe sans circuit, et si le graphe contient un circuit, l'algorithme permet de calculer la numérotation compatible du sous graphe ne contenant pas de circuits.

Semaine du 08 Février au 13 Février :

Quelques ajouts ont été effectués au rapport, principalement la rédaction de la partie analyse du projet et les différents diagrammes utiles ont été réalisés. De plus, après quelques recherches de générateur de documentation (comme la JavaDoc pour Java). Un de ces générateurs de documentation a été retenu. Celui-ci nommé Doxygen (cf http://www.doxygen.org/), est un logiciel libre permettant de générer à partir principalement des commentaires contenus dans les sources d'un programme une documentation pour les programmes développés en C/C++, Java, Php, et bien d'autre. Cette documentation peut être générée en différents formats dont le HTML.

De plus, différents composants et fonctionnalités ont été rajoutés permettant a configuration d'un graphe (définir l'état des sommets, leurs étiquettes, la valeur des arcs ...) ainsi que la configuration de l'algorithme comme la vitesse de déroulement de l'algorithme ou bien le type d'interaction avec l'algorithme.

Enfin, quelques modifications sur la structure du logiciel ont été effectuées afin de permettre le chargement dynamique de nouveaux algorithmes, et donc d'ajouter cela à la manière d'un plugin. Après de nouvelles recherches concernant ce sujet, il semblerait qu'avec la modification effectuée sur le logiciel, cela soit réalisable, ce sera donc l'objectif de la semaine prochaine.

Semaine du 15 Février au 20 Février :

Cette semaine a surtout été utilisée pour remplir le carnet de bords en remplissant la partie interface graphique et fonctionnalités, et également en complétant la partie conception du rapport.

Concernant le chargement dynamique des algorithmes, après de multiples tentatives, cette fonctionnalité n'est toujours pas disponible, cependant, avec les nouvelles modifications effectuées sur la structure une autre méthode sera testée, et, espérons le sera fonctionnelle.

Semaine du 22 Février au 27 Février :

Le chargement dynamique des algorithmes a enfin été réalisé, grâce à cela, il est donc maintenant possible d'ajouter des algorithmes facilement. Une option dans la fenêtre a été rajoutée permettant de charger ses propres algorithmes. De plus, le logiciel a été adapté pour Windows, en effet, celui-ci était développé sous Linux, cependant, Windows étant un système d'exploitation très utilisé, il était bon de faire en sorte que le logiciel puisse tourner également sous Windows. Pour cela, nous avons utilisé un pré-processeur, celui-ci permet à partir d'instructions écrite dans le code source uniquement destinées au compilateur d'effectuer certaines opérations et pas d'autre. Pour le chargement dynamique, les fonctions à utiliser ne sont pas les mêmes, de ce fait, pour ouvrir une librairie, nous avons donc le morceau de code suivant :

```
#if defined(WIN32) || define(WIN64)
    librairy = LoadLibrary(libName);
#else
    library = dlopen(libName, RTLD_LAZY);
#endif
```

Ici, le compilateur va s'il est dans un environnement Windows compiler l'instruction qui utilise la fonction LoadLibrary, sinon il compilera l'instruction contenant dlopen.

Par la suite, des modifications ont été effectuées sur l'interface graphique. Lorsque celle-ci s'ouvre elle charge la configuration de la dernière utilisation du logiciel, et lorsqu'elle se ferme sauvegarde la nouvelle configuration. De ce fait, lorsque nous choisissons un algorithme, lorsque nous réouvrirons le logiciel, celui-ci sera positionné automatiquement sur cet algorithme.

En parallèle, le rapport à été complété et terminé.