



École Polytechnique de l'Université de Tours
64, Avenue Jean Portalis
37200 TOURS, FRANCE
Tél. +33 (0)2 47 36 14 14
www.polytech.univ-tours.fr

Département Informatique

Cahier de spécification système & plan de développement

Projet :	PFE "Réalisation d'une plateforme de gestion d'images médicales"		
Emetteur :	Guillaume DAGUIN	Coordonnées : guillaume.daguin@etu.univ-tours.fr	
Date d'émission :	19 janvier 2014		
Validation			
Nom	Date	Valide (O/N)	Commentaires

Donatello CONTE : 12/01/2014 ; O ;

Historique des modifications

Version	Date	Description de la modification
---------	------	--------------------------------

01 : 12/01/2014 ; Version initiale : Cahier de spécification

Table des matières

Cahier de spécification système	6
1.1 Introduction	6
1.2 Contexte de la réalisation	6
1.2.1 Contexte	6
1.2.2 Objectifs	6
1.3 Description générale	6
1.3.1 Environnement du projet	6
1.3.2 Caractéristiques des utilisateurs	6
1.3.3 Fonctionnalités et structure générale du système	7
1.3.4 Contraintes de développement, d'exploitation et de maintenance	7
1.4 Description des interfaces externes du logiciel	8
1.4.1 Interfaces matériel/logiciel	8
1.4.2 Interfaces homme/machine	8
1.4.3 Interfaces logiciel/logiciel	15
1.5 Architecture générale du système	15
1.6 Description des fonctionnalités	17
1.6.1 Définition de la fonction "Ouvrir fichiers"	17
1.6.2 Définition de la fonction "Sauvegarder fichiers"	17
1.6.3 Définition de la fonction "Imprimer fichiers"	17
1.6.4 Définition de la fonction "Anonymiser fichiers"	17
1.6.5 Définition de la fonction "Consulter l'aide"	17
1.6.6 Définition de la fonction "Consulter les infos du logiciel"	17
1.6.7 Définition de la fonction "Changer vue 2D courante"	17
1.6.8 Définition de la fonction "Bouger axe"	18
1.6.9 Définition de la fonction "Régler luminosité"	18
1.6.10 Définition de la fonction "Régler contraste"	18
1.6.11 Définition de la fonction "Mesurer angle"	18
1.6.12 Définition de la fonction "Mesurer distance"	18
1.6.13 Définition de la fonction "Ajouter annotation"	18
1.6.14 Définition de la fonction "Réinitialiser réglages"	18
1.7 Conditions de fonctionnement	19
1.7.1 Performances	19
1.7.2 Sécurité	19
 Plan de développement	 21
2.1 Découpage du projet en tâches	21
2.1.1 Documentation générale	21
2.1.2 Installation bibliothèque et prise en main	21
2.1.3 Téléchargement et Installation Visual Studio	21
2.1.4 Prise en main Visual Studio + C#	22
2.1.5 Élaboration du cahier de spécification	22
2.1.6 Conception du logiciel	22



2.1.7	Élaboration de l'interface graphique	22
2.1.8	Lecture des fichiers DICOM	23
2.1.9	Affichage images 2D dans l'interface graphique	23
2.1.10	Construction de la vue 3D + rotation	23
2.1.11	Réalisation de la fenêtre des réglages	23
2.1.12	Anonymisation des fichiers	24
2.1.13	Réalisation des tests	24
2.2	Planning	25
A	Glossaire	26

Cahier de spécification système



1.1 Introduction

Ce document est le cahier de spécification système pour le Projet de Fin d'Études "Réalisation d'une plateforme C# de gestion d'images médicales" pour le laboratoire de recherche RFAI de l'École Polytechnique de l'Université de Tours.

L'encadrant de ce projet est M. Donatello Conte.

Le but de ce projet est de réaliser une plateforme générique de traitement/analyse d'images médicales.

1.2 Contexte de la réalisation

1.2.1 Contexte

Nous aimerions créer une plateforme de gestion unique pour différents formats d'images. L'utilité repose sur le fait qu'on pourrait disposer d'un ensemble d'algorithmes de traitement et analyse d'images médicales, sans devoir les ré-implémenter pour chaque format connu.

Une telle plateforme permettrait aussi de réduire l'effort de développement lors de la sortie d'un nouveau format d'images ; dans ce cas, en effet, les seuls modules à implémenter seraient ceux de lecture et d'écriture des images dans le nouveau format.

1.2.2 Objectifs

L'objectif du projet est celui de concevoir l'architecture d'une plateforme générique de traitements d'images médicales, et d'implémenter les modules de base (ouvrir & sauvegarder des images de différents formats, UI...).

1.3 Description générale

1.3.1 Environnement du projet

Le projet est un nouveau projet, on ne repart d'aucune source existante au préalable. La modélisation du logiciel devra être faite également.

Cependant, il faut bien noter que nous nous baserons en partie sur les logiciels propriétaires déjà présents sur le marché - tant en terme de fonctionnalités qu'en terme d'User Interface.

1.3.2 Caractéristiques des utilisateurs

Nous supposons ici que les utilisateurs finaux ont des notions de base de médecine. Ce sera typiquement un médecin ou un analyste de radio.

Un autre type d'utilisateur peut utiliser le logiciel, un chercheur par exemple. Il pourra paramétrer le logiciel, contrairement au médecin.

Quelque soit l'utilisateur, ils auront tous les droits d'accès (lecture et écriture) sur les fichiers.

1.3.3 Fonctionnalités et structure générale du système

Le cas d'utilisation ci-contre montre toutes les possibilités offertes à l'utilisateur.

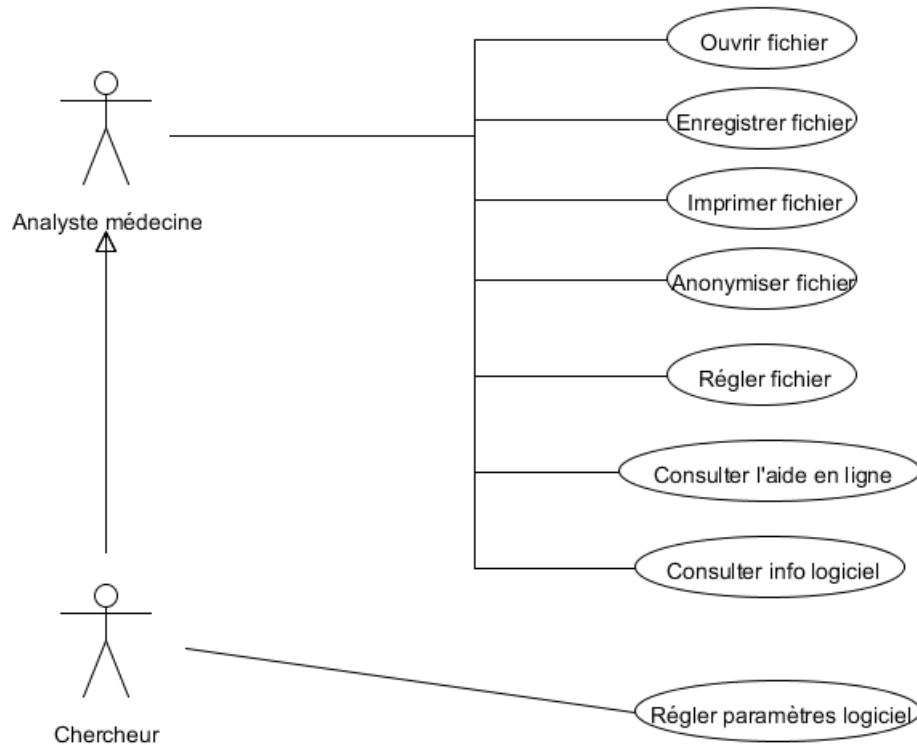


FIGURE 1.1 – Cas d'utilisation

Pour un soucis de visibilité et de simplicité, nous n'avons pas trop détaillé ce diagramme. En effet, certains tâches peuvent être divisées en sous catégories, mais cela aurait alourdi très fortement la lecture du diagramme.

Ainsi, nous détaillerons dans la partie 1.6 les spécificités et rôles de ces tâches.

1.3.4 Contraintes de développement, d'exploitation et de maintenance

Contraintes de développement

Le langage C# est un langage imposé par le laboratoire RFAI, car ils en ont une grande expérience.

Le langage C# étant un langage créé par Microsoft, le développement se fera sur Windows via l'IDE Visual Studio 2013.

L'interface graphique respectera donc les UI générales présents sous Windows.

Dans un premier temps, nous nous contenterons d'analyser uniquement les fichiers DICOM, car c'est la norme la plus présente sur le marché. Nous ne ferons cependant pas de portes aux autres formats de fichiers. Le logiciel devra être le plus modulaire possible, pour deux raisons :

- Contrainte du format : on devra pouvoir ajouter un format sans changer la structure du logiciel.
- Contrainte d'évolutivité : le logiciel sera amené à évoluer au fil des années.

1.4 Description des interfaces externes du logiciel

1.4.1 Interfaces matériel/logiciel

Le logiciel n'est pas amené à communiquer en réseau avec d'autres machines.

1.4.2 Interfaces homme/machine

Nous allons maintenant présenter l'IHM souhaité du logiciel, via plusieurs maquettes.

Maquette du logiciel lors de son lancement.

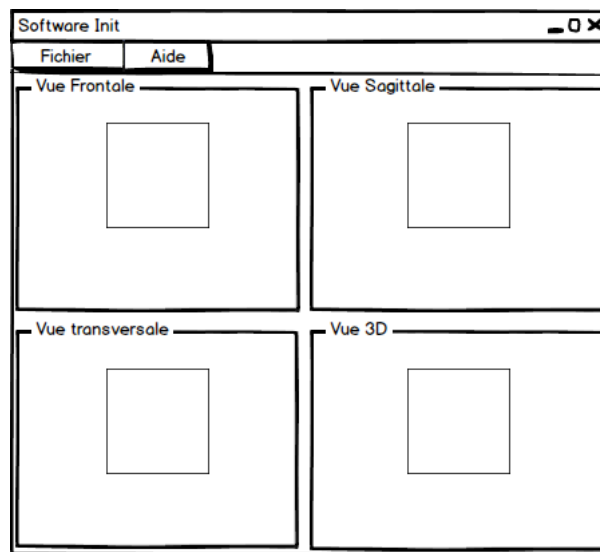


FIGURE 1.2 – Maquette du logiciel souhaité, lors de son lancement.

Lors de son lancement, le logiciel est très épuré, il faudra en effet charger les images pour avoir plus de fonctionnalités.

De ce fait, l'utilisateur ne pourra effectuer que ces actions :

- Ouvrir des fichiers (Raccourci clavier Ctrl + O).
- Consulter l'aide en ligne.
- Consulter les informations du logiciel.
- Quitter le logiciel (Raccourci clavier Ctrl + Q).

En ouvrant des fichiers valides, on aura bien évidemment beaucoup plus de possibilités. A noter que l'utilisateur pourra voir les possibilités du logiciel, mais qu'elles seront grisées pour qu'il ne puisse pas s'en servir.

Maquette du logiciel une fois des images chargées.

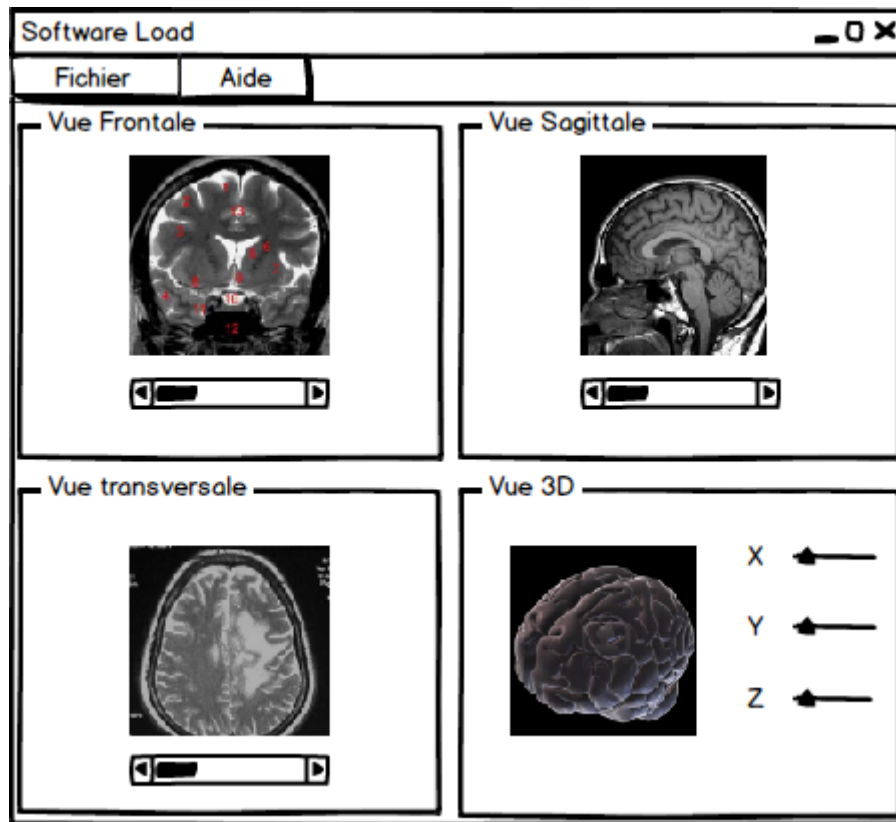


FIGURE 1.3 – Maquette du logiciel souhaité, une fois des images chargées.

Une fois que l'utilisateur a chargé des images valides dans le logiciel, voici à quoi il ressemblera.

Désormais, les possibilités offertes seront bien plus grandes :

- Ouvrir des fichiers (Raccourci clavier Ctrl + O).
- Enregistrer des fichiers (Raccourci clavier Ctrl + S pour enregistrer ou Ctrl + MAJ + S pour enregistrer sous).
- Fermer les fichiers (Raccourci clavier Ctrl + W).
- Imprimer des fichiers (Raccourci clavier Ctrl + P).
- Anonymiser les fichiers.
- Effectuer des réglages sur les fichiers (Raccourci clavier Ctrl + R).
- Consulter l'aide en ligne.
- Consulter les informations du logiciel.
- Quitter le logiciel (Raccourci clavier Ctrl + Q).

On dégrisera ainsi les possibilités du logiciel.

Des sliders horizontaux sont apparus sous les images 2D (sagittale, frontale & transversale), pour naviguer entre elles.

Un double clic sur une des images 2D ouvrira la fenêtre de réglages des images (également disponible via le menu fichier -> réglages).

Pour une raison de clarté, la vue 3D a été simplifiée ici et sera expliquée ci-dessous.

Maquette de la sous partie vue 3D.

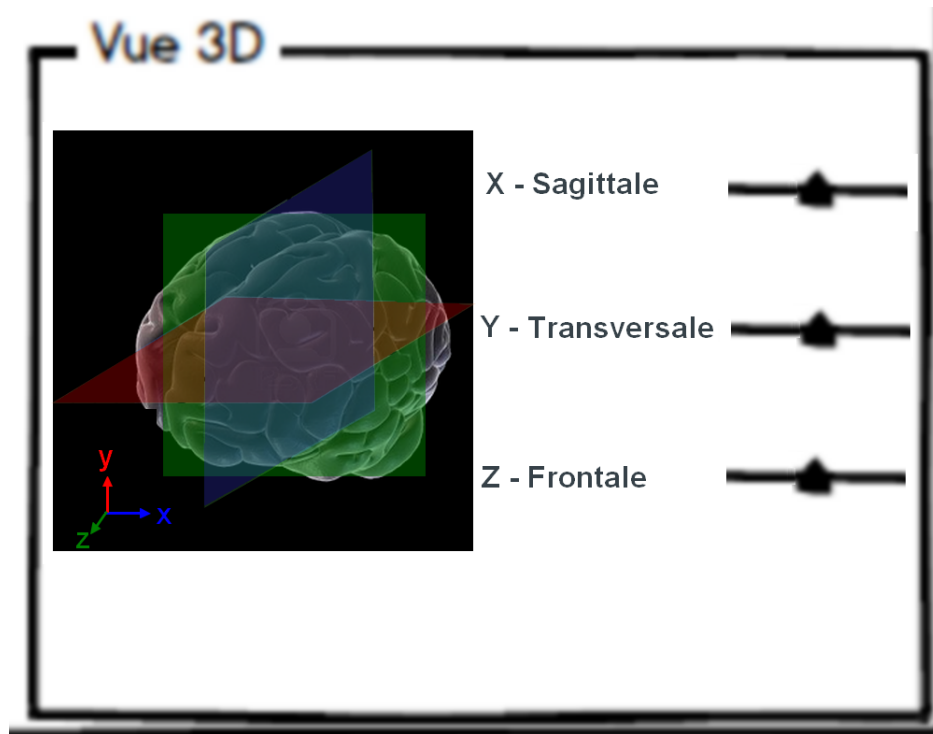


FIGURE 1.4 – Maquette de la sous partie 3D.

Nous expliquons ici la partie vue 3D du logiciel souhaité.

Au sujet de l'image de visualisation 3D, elle comporte le modèle 3D, obtenu en superposant les images 2D de nos fichiers d'entrée.

Cette image comporte aussi 3 plans 2D sur les axes x, y et z. Nous pouvons bouger ces axes via les sliders associés, à droite de l'image.

Les 3 axes représentent les 3 vues 2D que nous avons présentées précédemment. Ainsi, en bougeant ces axes, on changera les vues 2D (frontale, sagittale, transversale) du logiciel. De même, si on change les vues 2D, nous verrons le changement de plan dans la vue 3D.

Maquette de la fenêtre des réglages des images 2D.

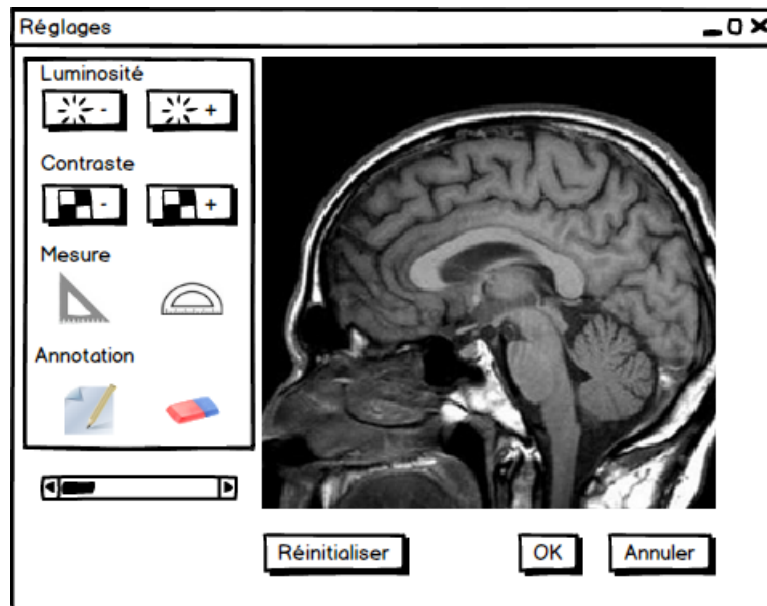


FIGURE 1.5 – Maquette de la fenêtre des réglages des images 2D.

Voici la fenêtre des réglages du logiciel, obtenue lorsque l'utilisateur sélectionne l'option des réglages dans le menu (fichier -> réglages) ou lorsqu'il double clique sur une des images 2D.

Via cette fenêtre, l'utilisateur peut ainsi :

- Régler la luminosité (augmenter et diminuer).
- Régler le contraste (augmenter et diminuer).
- Mesure la distance entre deux points.
- Mesure un angle entre deux points.
- Ajouter du texte sur notre image.
- Supprimer du texte.

A noter que ces réglages seront sauvegardés dans un fichier XML pour que l'utilisateur puisse retrouver ses réglages quand il rechargera les images. Il pourra réinitialiser les réglages avec le bouton "réinitialiser".

Maquette de la fenêtre de visualition de l'image 3D.

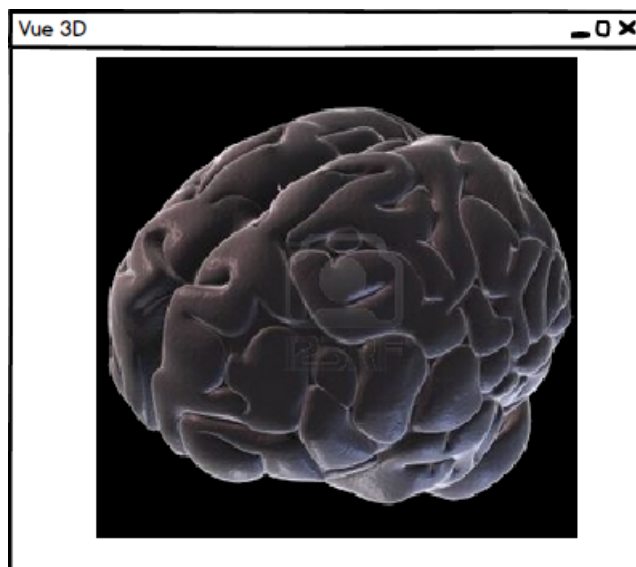


FIGURE 1.6 – Maquette de la fenêtre de visualition de l'image 3D.

Lorsque nous double cliquons sur l'image 3D, nous pouvons voir cette vue 3D dans une fenêtre plus grosse, mais cette fois-ci sans les axes.

Maquette de la fenêtre d'ouverture de fichiers.

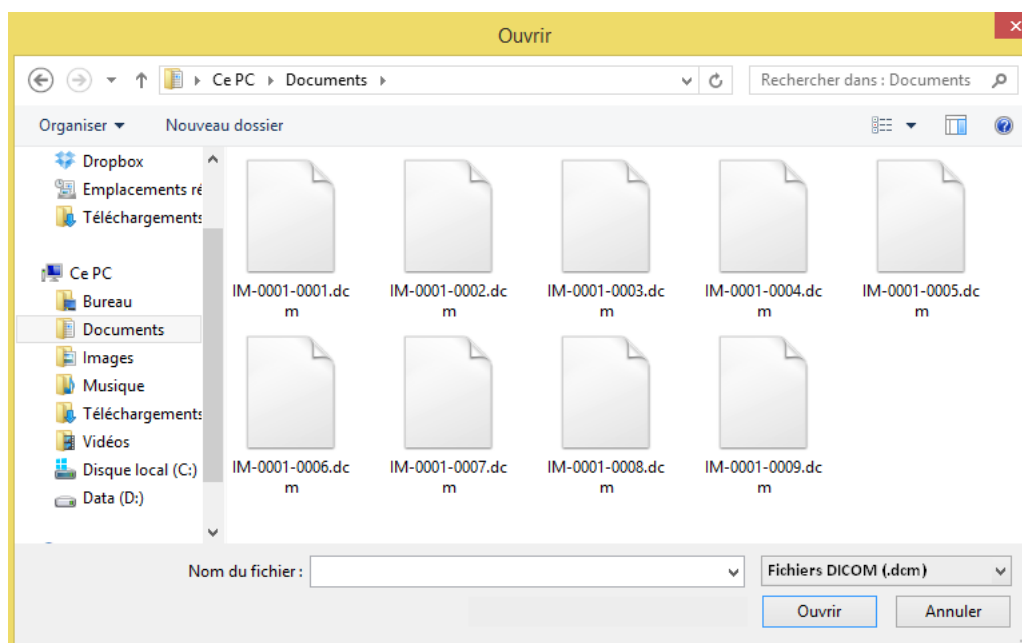
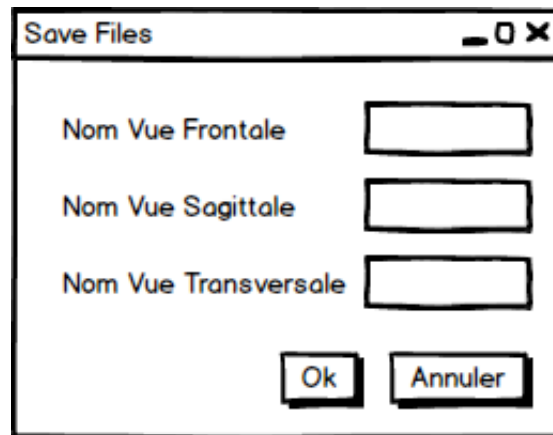


FIGURE 1.7 – Maquette de la fenêtre d'ouverture de fichiers.

Nous pouvons ouvrir cette fenêtre via le menu (fichier -> ouvrir) ou via le raccourci Ctrl + O. Si l'information sur la vue est présente dans le fichier DICOM, l'utilisateur sélectionnera uniquement l'ensemble des vues et on triera à la volée pour placer les images dans les vues associées. Sinon, nous utiliserons une norme de nommage commune pour les fichiers.

Maquettes de l'enregistrement de fichiers.



A wireframe of a 'Save Files' dialog box. It has a title bar with 'Save Files' and standard window controls. Inside, there are three labels: 'Nom Vue Frontale', 'Nom Vue Sagittale', and 'Nom Vue Transversale', each followed by a rectangular input field. At the bottom right, there are two buttons labeled 'Ok' and 'Annuler'.

FIGURE 1.8 – Maquette de l'enregistrement de fichiers - 1.

Pour l'enregistrement de fichiers, nous aurons tout d'abord besoin de donner les noms souhaités pour les 3 vues 2D associés.

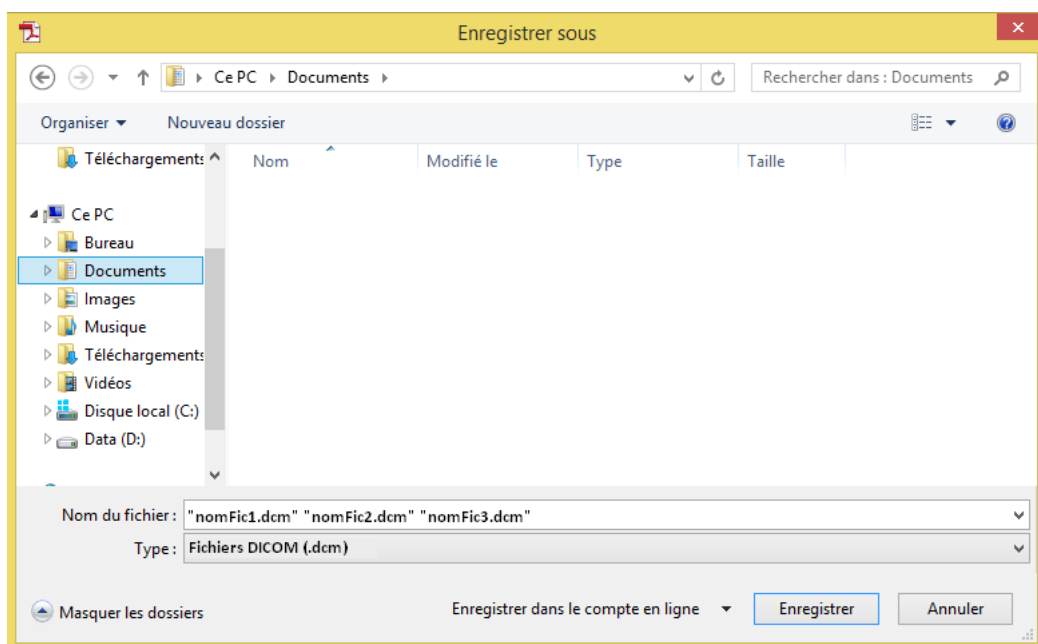


FIGURE 1.9 – Maquette de l'enregistrement de fichiers - 2.

Une fois les noms choisis, une autre fenêtre est ouverte et le champs des noms est alors directement rempli. L'utilisateur devra alors choisir l'emplacement des fichiers et leur type.

Maquette de l'impression des vues.

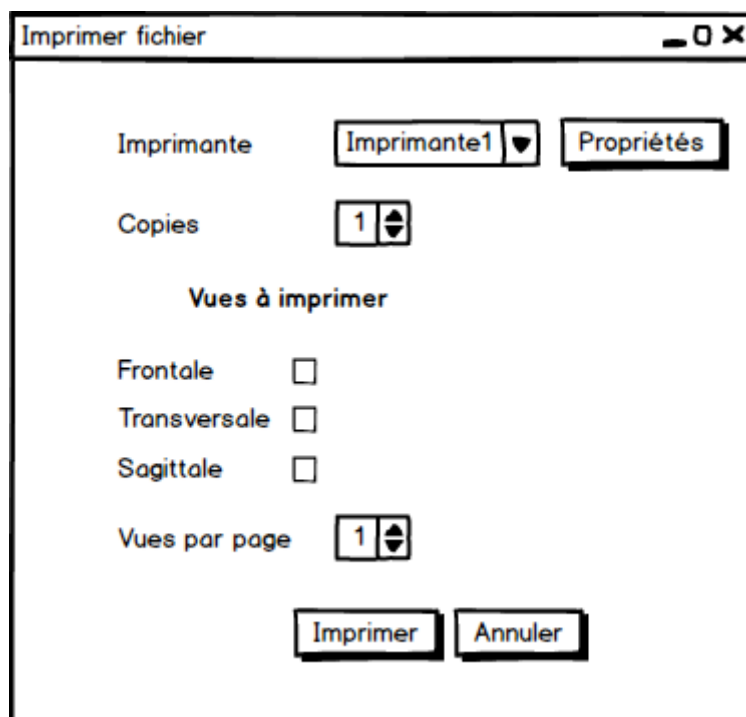


FIGURE 1.10 – Maquette de l'impression des vues.

La fenêtre des impressions est assez standard, nous choisissons l'imprimante, ses propriétés (qualité de l'encre...), le nombre de fois que nous voulons imprimer, les vues à imprimer le nombre de vues par page.

Maquette de l'anonymisation des fichiers.

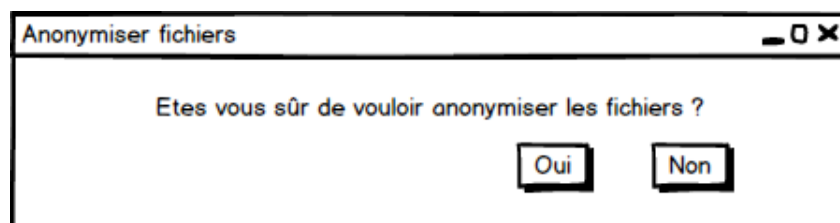


FIGURE 1.11 – Maquette de l'anonymisation des fichiers.

La fenêtre d'anonymisation est assez basique.

Consulter de l'aide en ligne.

Cette option est obtenue via le menu (Aide -> Aide en ligne). Elle ouvrira le navigateur, et possiblement un site internet explicatif.

Consulter les informations du logiciel.

Cette option est obtenue via le menu (Aide -> Informations). Elle donnera les informations (noms des développeurs, nom du laboratoire de recherche ...) du logiciel.

1.4.3 Interfaces logiciel/logiciel

Le logiciel sera amené à traiter des données issues d'un scanner. Ces données seront stockées sur l'ordinateur où est installé le logiciel. Nous ne sommes donc pas responsable de ces données, il en va de la responsabilité de l'hôpital de les garder anonyme.

1.5 Architecture générale du système

Le logiciel sera amené à être le plus modulaire possible, pour pouvoir gérer le plus de format d'images possible.

Nous utiliserons le design pattern Modèle-Vue-Contrôleur (MVC), qui permettra de séparer distinctement l'interface graphique, les traitements et les événements. Ainsi, nous aurons une architecture générale du type :

Vue	Contrôleur	Modèle
MainView	MainControler	MainModel
SettingsView	SettingsControler	SettingsModel
Visu3DView	Visu3DControler	Visu3DModel
OpenView	OpenControler	OpenModel
SaveView	SaveControler	SaveModel
PrintView	PrintControler	PrintModel
AnonymView	AnonymControler	AnonymModel
InfoView	InfoControler	-
HelpView	HelpControler	-

Seules les vues InfoView et HelpView n'ont pas de modèles associés, car ce sont juste des fenêtres de consultation.

Les vues sont chargées d'afficher les informations pour l'utilisateur, elles auront la forme vue dans la section 1.4.2.

Chaque contrôleur sera chargé de prévenir sa vue associée lors d'un événement spécifique (comme un clic sur un bouton par exemple), comme le veut le pattern MVC.

Chaque modèle effectuera les calculs (comme le calcul de la vue 3D, par exemple).

Voici l'architecture générale de notre MVC. Nous ne détaillerons pas les méthodes et attributs, nous resterons donc général.

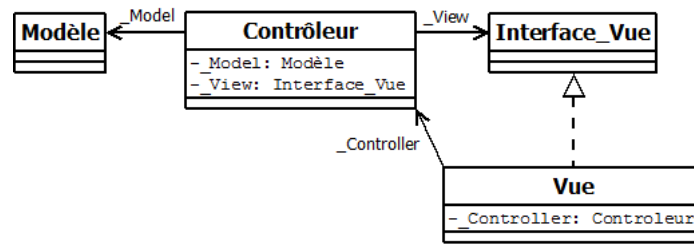


FIGURE 1.12 – Diagramme de classe générale du MVC.

Ici, nous pouvons voir que la vue hérite d'une interface Interface_Vue. Le principe d'une vue dans un MVC est juste d'afficher les informations, le contrôleur gérant tous les événements spécifiques de l'UI. Cependant, dans notre cas, il sera plus simple pour nous d'abonner la vue aux événements de l'interface graphique puis d'en avvertir le contrôleur pour la suite.

De manière générale, le diagramme pour les vues se représenterait comme suit :

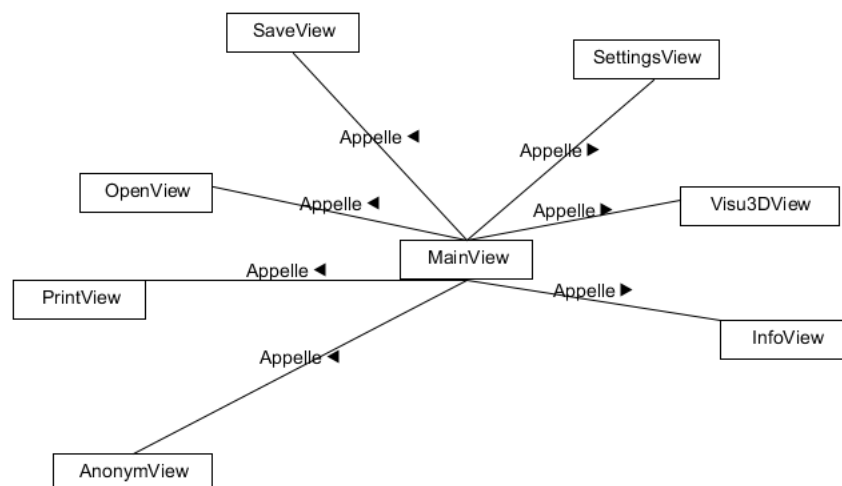


FIGURE 1.13 – Diagramme de classe général.

La fenêtre MainView peut ainsi appeler l'ensemble des fenêtres du logiciel. Il est à noter que chacune de ces vues possède son propre modèle et son propre contrôleur.

De part la nature très large du projet, allant de la conception de l'interface graphique à la création de la structure de données pour les images issues du scanner nous utiliserons une méthode agile pour concevoir le logiciel. Ainsi, nous pourrions diviser en sous tâches très distinctes le projet.

1.6 Description des fonctionnalités

Nous allons maintenant décrire l'ensemble des fonctionnalités.

1.6.1 Définition de la fonction "Ouvrir fichiers"

Fonction faisant partie de la vue OpenView. Fonction qui prendra en paramètres une liste de fichiers et qui les mettra en mémoire, sur une structure à définir. A un instant t donné, toutes les images devront être chargées en mémoire, pour la vue 3D notamment. Cette fonction prendra uniquement en compte les fichiers d'images pris en compte (type DICOM), ou lèvera une exception. Cette fonction a une priorité importante.

1.6.2 Définition de la fonction "Sauvegarder fichiers"

Fonction faisant partie de la vue SaveView. Fonction chargée de sauvegarder en local, sur un emplacement choisi au préalable par l'utilisateur, les fichiers présents en mémoire. Le réglage des fichiers (luminosité par ex) sera également sauvegardés dans cet emplacement, via un fichier XML. L'utilisateur pourra sauvegarder les images avec des types voulues par les programmeurs (comme le type DICOM, png, jpg ...). Cette fonction a une priorité normale.

1.6.3 Définition de la fonction "Imprimer fichiers"

Fonction faisant partie de la vue PrintView. Fonction chargée d'imprimer les vues sélectionnées par l'utilisateur. Il aura ainsi le choix entre les trois vues (sagittale, transversale, frontale), le nombre de pages à imprimer et le nombre de vues par page. L'utilisateur pourra évidemment choisir l'imprimante sur laquelle il voudra imprimer, ainsi que les réglages pour l'imprimante (niveau de gris, qualité de l'impression...). Cette fonction a une priorité peu importante.

1.6.4 Définition de la fonction "Anonymiser fichiers"

Fonction faisant partie de la vue AnonymView. Fonction qui va enlever les informations du patient (nom, prénom) des fichiers présents en mémoire. Cette fonction a une priorité peu importante.

1.6.5 Définition de la fonction "Consulter l'aide"

Fonction faisant partie de la vue HelpView. Fonction qui va afficher l'aide pour l'utilisateur, avec les informations utiles au problème de l'utilisateur. Cette fonction a une priorité peu importante.

1.6.6 Définition de la fonction "Consulter les infos du logiciel"

Fonction faisant partie de la vue InfoView. Fonction qui va afficher les informations (nom des développeurs, de l'université de Tours...) sur le développement du logiciel. Cette fonction a une priorité peu importante.

1.6.7 Définition de la fonction "Changer vue 2D courante"

Fonction faisant partie de la vue MainView. Fonction associée aux vues 2D frontale, sagittale et transversale. Lors de l'appui sur les boutons suivant et précédent, on changera la vue courante. Ainsi, par exemple, si l'utilisateur appuie sur le bouton précédent de la vue frontale, on chargera comme vue courante la vue précédente. Cette fonction a une priorité importante.

1.6.8 Définition de la fonction "Bouger axe"

Fonction faisant partie de la vue MainView. Fonction appelée lors d'un appel à la fonction de changement de la vue courante et qui se chargera de faire bouger l'axe dédié à la vue qui va être changée. Cette fonction a une priorité normale.

1.6.9 Définition de la fonction "Régler luminosité"

Fonction faisant partie de la vue SettinsView. Fonction qui changera la luminosité des images chargées en mémoire, et qui sauvegardera ces valeurs dans un fichier XML. Cette fonction augmentera la luminosité quand l'utilisateur cliquera sur le bouton d'augmentation de la luminosité (bouton "+") et baissera la luminosité quand l'utilisateur cliquera sur le bouton "-" de l'interface graphique. Cette fonction a une priorité normale.

1.6.10 Définition de la fonction "Régler contraste"

Fonction faisant partie de la vue SettingsView. Fonction qui changera le contraste des images chargées en mémoire, et qui sauvegardera ces valeurs dans un fichier XML. Cette fonction augmentera le contraste quand l'utilisateur cliquera sur le bouton d'augmentation du contraste (bouton "+") et baissera le contraste quand l'utilisateur cliquera sur le bouton "-" de l'interface graphique. Cette fonction a une priorité normale.

1.6.11 Définition de la fonction "Mesurer angle"

Fonction faisant partie de la vue SettingsView. Cette fonction prendra en paramètre les coordonnées des trois points qui représenteront les points choisis par l'utilisateur et dont il veut l'angle. Elle retournera l'angle entre ces trois points. Une autre fonction sera utilisée pour afficher ces informations à l'utilisateur. Cette fonction a une priorité normale.

1.6.12 Définition de la fonction "Mesurer distance"

Fonction faisant partie de la vue SettingsView. Cette fonction prendra en paramètre les coordonnées des deux points qui représenteront les points choisis par l'utilisateur. Elle retournera la distance entre ces deux points. Une autre fonction sera utilisée pour afficher ces informations à l'utilisateur. Cette fonction a une priorité normale.

1.6.13 Définition de la fonction "Ajouter annotation"

Fonction faisant partie de la vue SettingsView. Cette fonction permettra à l'utilisateur d'ajouter des annotations sur une image donnée. L'utilisateur cliquera sur le bouton d'annotation, puis cliquera quelque part dans l'image en question, pour ensuite saisir des informations sur l'image. On sauvegarder ces annotations dans le fichier XML. Cette fonction a une priorité normale.

1.6.14 Définition de la fonction "Réinitialiser réglages"

Fonction faisant partie de la vue SettingsView. Cette fonction réinitialise les réglages dans le fichier XML, et l'utilisateur voit également la réinitialisation dans l'interface graphique, via une remise à 0. Cette fonction a une priorité normale.

1.7 Conditions de fonctionnement

1.7.1 Performances

Du fait des traitements lourds que nous serons amenés à faire (création de l'image 3D notamment) , les temps de réponses devront être optimaux, l'utilisateur devra attendre au maximum 20 secondes, avec une progress bar pour qu'il sache où il en est.

1.7.2 Sécurité

Nous ne sommes pas responsables des images issues du scanner, mais nous ne devons en aucun cas les altérer, ou les modifier sans que l'utilisateur en soit averti.

Plan de développement

2.1 Découpage du projet en tâches

2.1.1 Documentation générale

Description de la tâche

Cette tâche comporte la documentation sur le projet en lui même, et sur tout ce qui le compose, entre autre une documentation sur l'existant et sur la norme DICOM.

Estimation de charge

2 jours homme.

Temps réellement passé

2 jours homme.

2.1.2 Installation bibliothèque et prise en main

Description de la tâche

Cette tâche consiste en l'installation des bibliothèques VTK/ITK/OpenCV et de leur comparaison pour notre projet.

Estimation de charge

1 jour homme.

Temps réellement passé

1 jour homme.

2.1.3 Téléchargement et Installation Visual Studio

Description de la tâche

Téléchargement de l'IDE Visual Studio et son installation.

Estimation de charge

2 heures.

Temps réellement passé

2 heures.



2.1.4 Prise en main Visual Studio + C#

Description de la tâche

Prise en main de l'IDE Visual Studio et prise en main du langage C# et de ses spécificités. Création de projets exemple pour voir la syntaxe puis des UI de tests.

Estimation de charge

4 heures.

Temps réellement passé

4 heures.

2.1.5 Élaboration du cahier de spécification

Description de la tâche

Élaboration du cahier de spécification, avec modélisation du besoin, des maquettes du logiciel, des langages qui seront utilisés durant le projet

Estimation de charge

6 jours hommes.

Temps réellement passé

7 jours hommes.

2.1.6 Conception du logiciel

Description de la tâche

Modélisation du logiciel via un diagramme de classe UML spécifique et des cas d'utilisation.

Estimation de charge

3 jours homme.

2.1.7 Élaboration de l'interface graphique

Description de la tâche

Création de l'ensemble de l'interface graphique, avec les liaisons entre chaque fenêtre.

Estimation de charge

4 jours homme.

Contraintes temporelles

Cette tâche intervient après la tâche "Conception du logiciel".

2.1.8 Lecture des fichiers DICOM

Description de la tâche

Élaboration des fonctions afin de lire les fichiers DICOM et les transformer en image.

Estimation de charge

6 jours homme.

Contraintes temporelles

Cette tâche intervient après la tâche "Conception du logiciel".

2.1.9 Affichage images 2D dans l'interface graphique

Description de la tâche

Affichage des images dans l'UI et intégration avec les boutons spécifiques (précédent, suivant).

Estimation de charge

6 jours homme.

Contraintes temporelles

Cette tâche intervient après la tâche "Conception du logiciel".

2.1.10 Construction de la vue 3D + rotation

Description de la tâche

Création de la vue 3D avec l'ensemble des images 2D, rotation de l'image 3D et translation des plans associés.

Estimation de charge

25 jours homme.

Contraintes temporelles

Cette tâche intervient après les tâches "Conception du logiciel" et "Affichage images 2D dans l'interface graphique".

2.1.11 Réalisation de la fenêtre des réglages

Description de la tâche

Réalisation de la fenêtre des réglages, avec sauvegarde dans un fichier XML de la configuration, mesure des distances et des angles.

Estimation de charge

5 jours homme.



Contraintes temporelles

Cette tâche intervient après la tâche "Conception du logiciel".

2.1.12 Anonymisation des fichiers

Description de la tâche

Enlever les informations de l'utilisateur dans l'ensemble des fichiers DICOM.

Estimation de charge

2 jours homme.

Contraintes temporelles

Cette tâche intervient après la tâche "Conception du logiciel".

2.1.13 Réalisation des tests

Description de la tâche

Réalisation de tests afin de vérifier que le logiciel soit conforme aux attentes.

Estimation de charge

4 jours homme.

Contraintes temporelles

Cette tâche intervient après toutes les autres tâches.

2.2 Planning

Voici le planning prévisionnel pour l'ensemble de mes tâches.

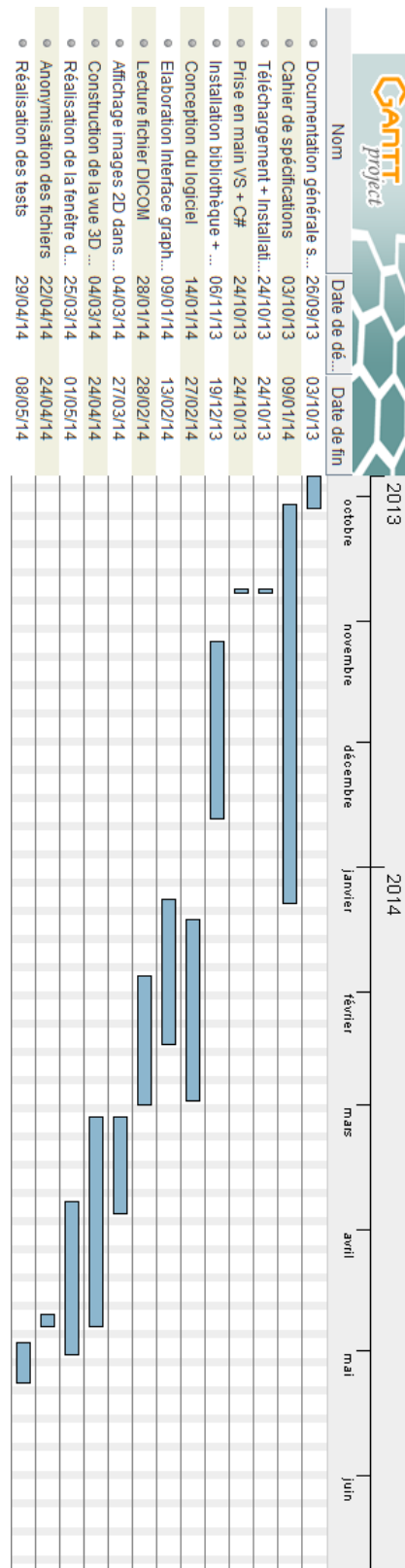


FIGURE 2.14 – Diagramme de Gantt du projet

Glossaire

UML : Unified Modeling Language, Langage de modélisation unifié permettant de spécifier les fonctionnalités d'un logiciel via des diagrammes.

MVC : Modèle-Vue-Contrôleur, modèle destiné à diviser les travaux différents d'un interface graphique. La vue est chargé d'afficher les composants, le contrôleur se charge des événements comme le clic de boutons et enfin le modèle se charge du traitement des données.

DICOM : Digital imaging and communications in medicine, norme standardisé d'imagerie médicale.

IHM : Interface Homme Machine, manière dont l'être humain communique avec un ordinateur, de manière graphique, pour se servir d'un logiciel.

Luminosité : Caractéristique de ce qui émet la lumière. Lorsqu'on joue sur la luminosité, on joue donc sur les couleurs de l'image (+/- clair).

Contraste : Propriété intrinsèque d'une image qui désigne et quantifie la différence entre les parties claires et foncées d'une image.

Source : Wikipedia.org