Génération d'image fractale

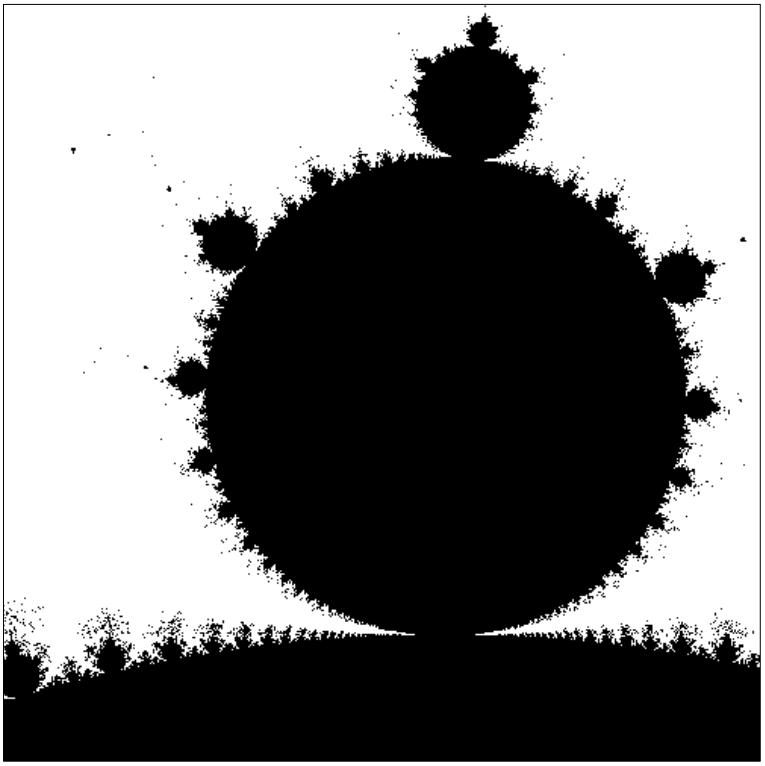


Image générée avec le programme
(x : -0.3 à 0 ; y : 0.6 à 0.9 ; 150 itérations)

Auteur : HAUTIER Jonas
Email : jonas.hautier@cpnv.ch
Dernière modification : 04.06.2021

Table des matières

1 Analyse préliminaire

1.1 Introduction

L'objectif général de mon projet est d'implémenter un générateur d'image fractale.

Le programme sera développé en C#, utilisera SQLite comme SGBD (système de gestion de base de données), sera versionné avec git et suivra la méthode de gestion de projet « Waterfall ».

L'utilisation du .NET Framework en C# limite mon programme à Windows, ce qui n'est pas un problème. J'ai choisi d'utiliser SQLite comme SGBD car cela m'évite de devoir mettre en place un serveur de BDD (base de données).

Ce projet suivra les normes suivantes :

- La norme appliquée au CPNV pour le MCD (Modèle Conceptuel de Données) et le MLD (Modèle Logique de Données)
- Les normes de programmation et conventions de nommage en C# basées sur les recommandations de Microsoft

Ce projet est réalisé dans le condition du TPI, c.-à-d. :

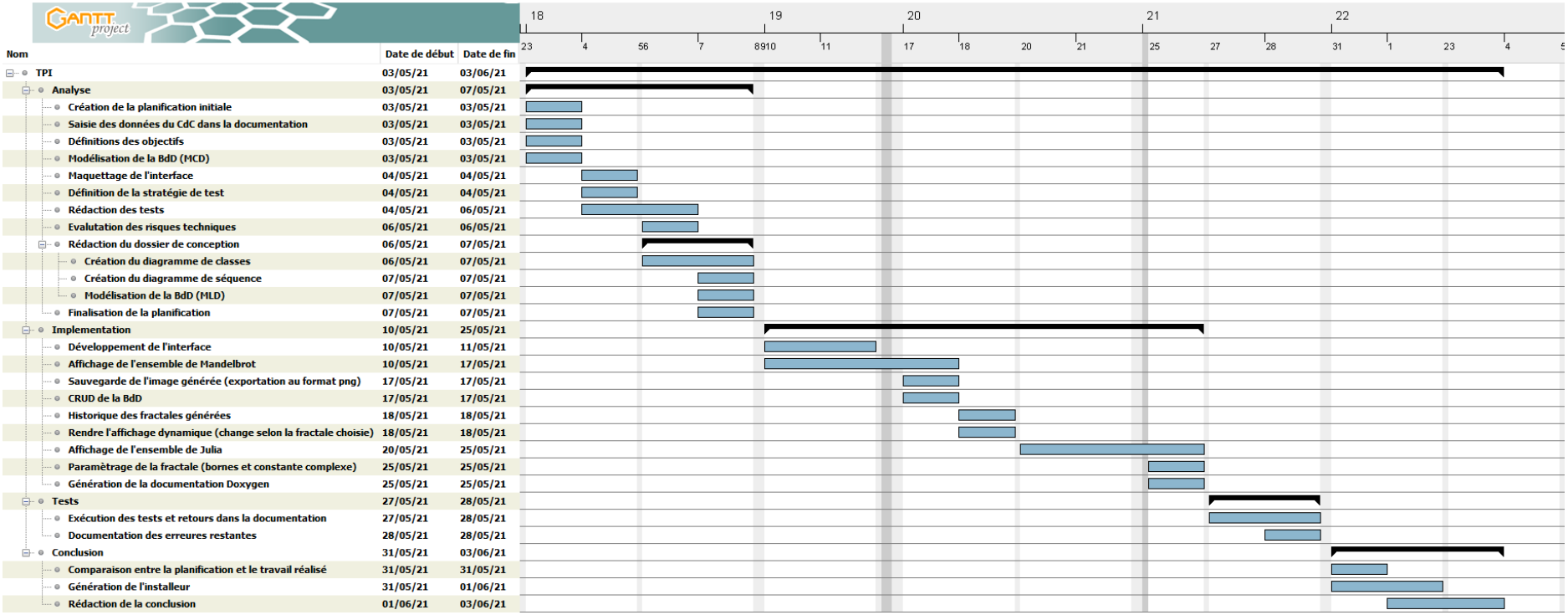
- Le projet est réalisé en 90h ;
- Le projet est individuel ;
- Le projet est commandé et évalué par le chef de projet ;
- Le rapport de projet est évalué par le chef de projet et deux experts ;
- La présentation et la défense sont évaluées par les experts.

Ce projet a pour but de confirmer que je suis capable d'exécuter un CdC (cahier des charges) correspondant aux compétences d'un informaticien CFC.

1.2 Objectifs

- Le programme est capable de générer l'une des fractales suivantes : l'ensemble de Mandelbrot et les ensembles de Julia de degré 2 ;
- Le programme donne à l'utilisateur la possibilité de définir les bornes réelles (x min et x max) et imaginaires (y min et y max) de la fractale générée ;
- Le programme adapte son affichage à la fractale choisie (c.-à-d., dans le cas des ensembles de Julia, il affiche un champ supplémentaire pour configurer *c*) ;
- Lorsque le programme a généré l'image de la fractale, il affiche le temps de génération de l'image ;
- Le programme donne à l'utilisateur la possibilité de sauvegarder l'image générée au format png ;
- Le programme est capable d'afficher un historique des derniers paramètres de génération utilisés ;
- Le programme donne à l'utilisateur la possibilité de régénérer une image à partir de l'historique ;

1.3 Planification initiale



Nom	Date de début	Date de fin
TPI	03/05/21	03/06/21
Analyse	03/05/21	07/05/21
Création de la planification initiale	03/05/21	03/05/21
Saisie des données du CdC dans la documentation	03/05/21	03/05/21
Définitions des objectifs	03/05/21	03/05/21
Modélisation de la BdD (MCD)	03/05/21	03/05/21
Maquettage de l'interface	04/05/21	04/05/21
Définition de la stratégie de test	04/05/21	04/05/21
Rédaction des tests	04/05/21	06/05/21
Evalutation des risques techniques	06/05/21	06/05/21
Rédaction du dossier de conception	06/05/21	07/05/21
Création du diagramme de classes	06/05/21	07/05/21
Création du diagramme de séquence	07/05/21	07/05/21
Modélisation de la BdD (MLD)	07/05/21	07/05/21
Finalisation de la planification	07/05/21	07/05/21
Implementation	10/05/21	25/05/21
Développement de l'interface	10/05/21	11/05/21
Affichage de l'ensemble de Mandelbrot	10/05/21	17/05/21
Sauvegarde de l'image générée (exportation au format png)	17/05/21	17/05/21
CRUD de la BdD	17/05/21	17/05/21
Historique des fractales générées	18/05/21	18/05/21
Rendre l'affichage dynamique (change selon la fractale choisie)	18/05/21	18/05/21
Affichage de l'ensemble de Julia	20/05/21	25/05/21
Paramètrage de la fractale (bornes et constante complexe)	25/05/21	25/05/21
Génération de la documentation Doxygen	25/05/21	25/05/21
Tests	27/05/21	28/05/21
Exécution des tests et retours dans la documentation	27/05/21	28/05/21
Documentation des erreures restantes	28/05/21	28/05/21
Conclusion	31/05/21	03/06/21
Comparaison entre la planification et le travail réalisé	31/05/21	31/05/21
Génération de l'installeur	31/05/21	01/06/21
Rédaction de la conclusion	01/06/21	03/06/21

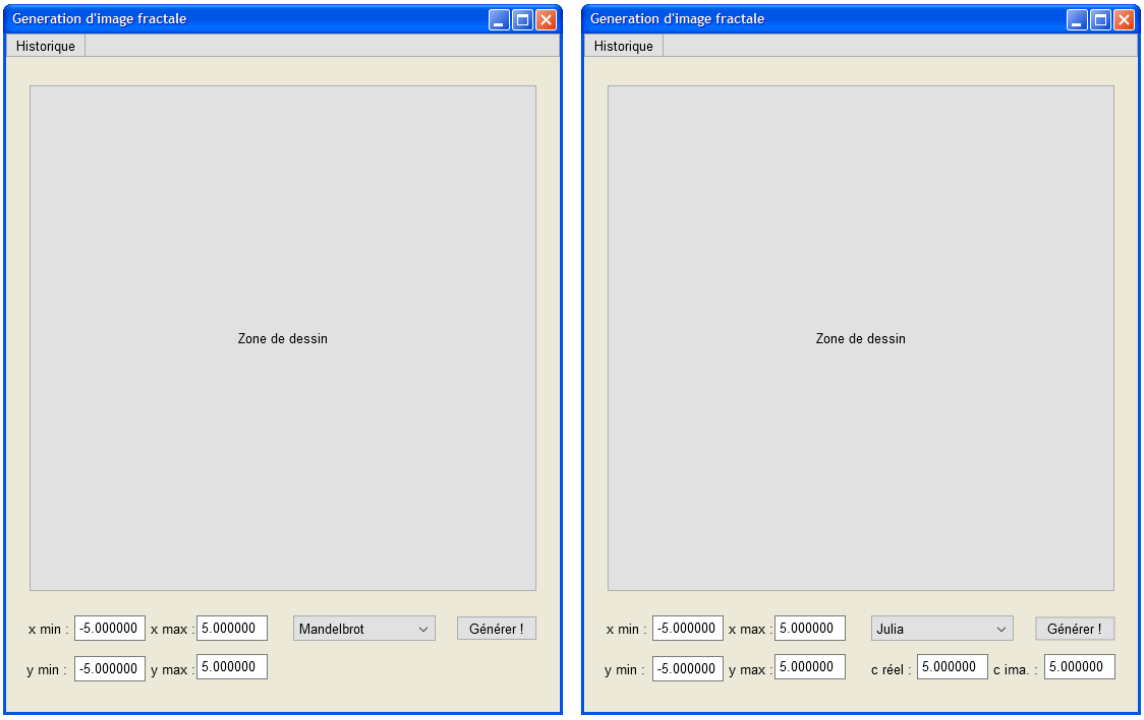
Ma planification initiale.

J'ai prévu une semaine pour l'Analyse/Conception, deux semaines pour la Réalisation, deux jours pour les tests et trois jours pour la Conclusion.

2 Analyse / Conception

2.1 Concept

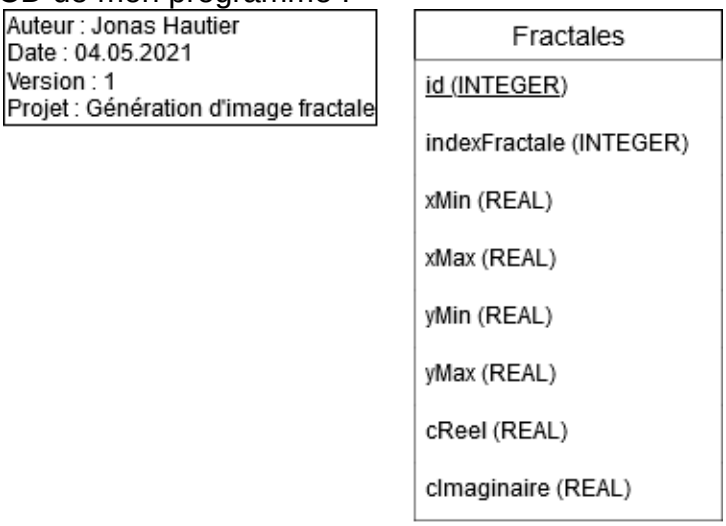
Voici les maquettes de mon programme :



A gauche, la vue avec l'ensemble de Mandelbrot sélectionné.
A droite, la vue avec l'ensemble de Julia sélectionné. Le paramétrage de « c » est visible.

La vue de l'historique. Régénère une entrée de l'historique quand on clique dessus.

Ci-dessous, le MCD de mon programme :



MCD

2.2 Stratégie de test

Pour mon projet, j'ai choisi d'effectuer la majorité de mes tests après avoir réalisé tous les modules de mon programme.
Après avoir développé un module, je ferai uniquement les tests unitaires puis passerai à la suite s'il fonctionne. Une fois les modules développés, je referai les tests unitaires, pour m'assurer de la non-régression de mes modules, puis ferai les tests d'intégration et d'acception, basés sur les objectifs du projet.

Tous les tests seront exécutés sur mon poste de développement. Par précaution, j'effectuerai les tests d'acceptation également sur une autre machine du CPNV pour m'assurer que le programme valide les objectifs.

Pour les données de tests, je générerai une dizaine de fractales et vérifierai qu'elles sont correctes avec l'application XaoS (lien de téléchargement de XaoS).

Les tests seront faits par moi-même et ne seront pas automatisés.

2.2.1 Liste des scénarios

J'ai groupé mes scénarios selon les modules de mon programme.

2.2.1.1 Interface

ID	Etant donné	Lorsque	Alors
1.1	Le programme vient d'être lancé	Le programme est affiché à l'écran	La fractale sélectionnée est « Mandelbrot »
1.2	La fractale sélectionnée est « Mandelbrot »	Je sélectionne la fractale « Julia »	Les champs « c réel » et « c ima. » deviennent visibles
1.3	La fractale sélectionnée est « Julia »	Je sélectionne la fractale « Mandelbrot »	Les champs « c réel » et « c ima. » deviennent invisibles

1.4	Le programme vient d'être lancé	Le programme est affiché à l'écran	Les champs suivants contiennent : x min → -2 x max → 2 y min → -2 y max → 2
1.5	La limite minimale est plus grande que la limite maximal dans l'axe des réels	Je clique sur « Générer ! »	La fractale a subi une symétrie axiale, avec l'axe de symétrie sur l'axe vertical
1.6	La limite minimale est plus grande que la limite maximal dans l'axe des imaginaires	Je clique sur « Générer ! »	La fractale a subi une symétrie axiale, avec l'axe de symétrie sur l'axe horizontal
1.7	Une fractale a été générée	Le programme a fini de générer la fractale	Le temps de génération de la fractale est indiqué

2.2.1.2 Génération de l'ensemble de Mandelbrot

Pour tous les scénarios de cette section, la fractale sélectionnée sera toujours « Mandelbrot ».

ID	Etant donné	Lorsque	Alors
2.1	Tous les champs sont remplis avec des données valides	Je clique sur « Générer ! »	La fractale est générée dans les limites données
2.2	Un des champs suivants n'a pas été rempli : « x min », « x max », « y min » ou « y max »	Je clique sur « Générer ! »	Le programme affiche une pop-up informant l'utilisateur du champ qu'il a omis
2.3	La fractale sélectionnée est « Mandelbrot » et un des champs suivants a été rempli avec des données invalides : « x min », « x max », « y min » ou « y max »	Je clique sur « Générer ! »	Le programme affiche une pop-up informant l'utilisateur du champ dans lequel une valeur incorrecte a été saisie

2.2.1.3 Génération de l'ensemble de Julia

Pour tous les scénarios de cette section, la fractale sélectionnée sera toujours « Julia ».

ID	Etant donné	Lorsque	Alors
3.1	Tous les champs sont remplis avec des données valides	Je clique sur « Générer ! »	La fractale est générée dans les limites données
3.2	Un des champs suivants n'a pas été rempli : « x min », « x max », « y min », « y max », « c réel » ou « c ima. »	Je clique sur « Générer ! »	Le programme affiche une pop-up informant l'utilisateur du champ qu'il a omis
3.3	Un des champs suivants a été rempli avec des données invalides : « x min », « x max », « y min », « y max », « c réel » ou « c ima. »	Je clique sur « Générer ! »	Le programme affiche une pop-up informant l'utilisateur du champ dans lequel une valeur incorrecte a été saisie

2.2.1.4 Historique des générations

ID	Etant donné	Lorsque	Alors
4.1	Le programme vient d'être installé et exécuté	Je regarde dans l'historique	L'historique est vide
4.2	Une fractale vient d'être générée	Je regarde dans l'historique	Les paramètres de génération de la fractale ont été ajouté à l'historique
4.2.1	Une fractale « Mandelbrot » vient d'être générée	Je regarde dans l'historique	Les paramètres de génération sont sous cette forme : *Type : Mandelbrot* *x :* **xMin** *à* **xMax** *y :* **yMin** *à* **yMax**
4.2.2	Une fractale « Julia » vient d'être générée	Je regarde dans l'historique	Les paramètres de génération sont sous cette forme : *Type : Julia* *x :* **xMin** *à* **xMax** *y :* **yMin** *à* **yMax** *c :* **cReel** +/- **cIma.** *i*
4.3	J'ai cliqué sur le bouton « Générer ! » mais il y avait un problème dans les paramètres de génération	Je regarde dans l'historique	L'historique contient les mêmes données/!\
4.4	L'historique contient déjà des données	Je ferme le programme puis le relance	L'historique contient les mêmes données
4.5	L'historique est affiché et contient des données	Je clique sur l'une des entrées de l'historique	Le programme régénère la fractale sur laquelle j'ai cliqué

2.2.1.5 Sauvegarde de la fractale générée

ID	Etant donné	Lorsque	Alors
5.1	Une fractale a été générée et est affichée dans la zone de dessin	Je clique sur la zone de dessin	Le programme ouvre une fenêtre pour enregistrer l'image générée
5.2	Une fractale a été générée et sauvegardée sur l'ordinateur	Je cherche le fichier sur mon ordinateur	Je trouve le fichier dans le répertoire défini à l'enregistrement et le nom de fichier est celui demandé
5.3	Une fractale a été générée et sauvegardée sur l'ordinateur	J'ouvre l'image générée	L'image correspond à la fractale générée
5.4	Le programme me propose d'enregistrer l'image de la fractale générée	Je saisi un nom de fichier invalide (dans un système NTFS)	La fenêtre « Enregistrer Sous » refuse le nom saisi et demande pour .

2.3 Risques techniques

Ci-dessous, vous trouverez les risques techniques que je prévois pour mon projet. Après chaque risque décrit se trouvent les solutions mises en place pour réduire et surveiller les potentiels impacts du risque.

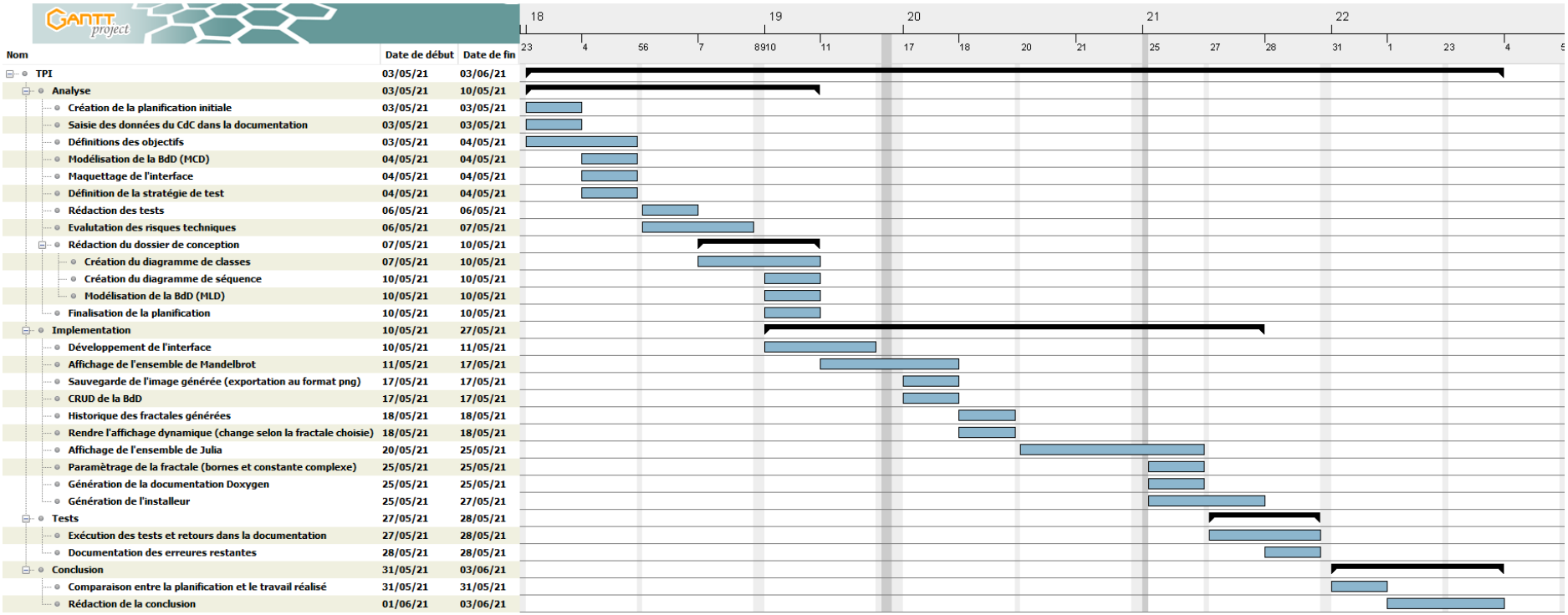
- Manipuler des nombres complexes en C# et les connecter correctement à mon parser :
  C# a une structure pour gérer des nombres complexes et le parser que je compte utiliser n'accepte pas les nombres complexes. J'ai effectué des tests de mon côté pour m'assurer que je pouvais les connecter en décomposant les nombres complexes.
- Vitesse de génération de la fractale :
  Récemment, je me suis documenté sur le parallélisme pour pouvoir l'implémenter dans ce projet.
- Mauvaise gestion du temps dans l'exécution du CdC :
  Création d'une copie de la planification que je tiens à jour durant le déroulement du projet. Ce fichier, « TempsReel.gan », représente le temps réellement passé sur chaque tâche contrairement à la planification.

2.4 Planification

J'ai choisi la méthode en cascade (Waterfall) comme méthodologie de gestion de projet car :

- je me sens plus à l'aise avec elle qu'avec la méthode Agile ;
- elle est plus adaptée à ma manière de travailler et de fonctionner : définir clairement ce qui va être fait avec un cadre bien délimité ;
- elle correspond parfaitement au cadre du TPI, c.-à-d. : elle est faite pour des projets où le client attend une seule livraison et où ses besoins ne vont pas de changer.

Voici la planification finale de mon projet :



Nom	Date de début	Date de fin
TPI	03/05/21	03/06/21
Analyse	03/05/21	10/05/21
Création de la planification initiale	03/05/21	03/05/21
Saisie des données du CdC dans la documentation	03/05/21	03/05/21
Définitions des objectifs	03/05/21	04/05/21
Modélisation de la BdD (MCD)	04/05/21	04/05/21
Maquettage de l'interface	04/05/21	04/05/21
Définition de la stratégie de test	04/05/21	04/05/21
Rédaction des tests	06/05/21	06/05/21
Evalutation des risques techniques	06/05/21	07/05/21
Rédaction du dossier de conception	07/05/21	10/05/21
Création du diagramme de classes	07/05/21	10/05/21
Création du diagramme de séquence	10/05/21	10/05/21
Modélisation de la BdD (MLD)	10/05/21	10/05/21
Finalisation de la planification	10/05/21	10/05/21
Implementation	10/05/21	27/05/21
Développement de l'interface	10/05/21	11/05/21
Affichage de l'ensemble de Mandelbrot	11/05/21	17/05/21
Sauvegarde de l'image générée (exportation au format png)	17/05/21	17/05/21
CRUD de la BdD	17/05/21	17/05/21
Historique des fractales générées	18/05/21	18/05/21
Rendre l'affichage dynamique (change selon la fractale choisie)	18/05/21	18/05/21
Affichage de l'ensemble de Julia	20/05/21	25/05/21
Paramètrage de la fractale (bornes et constante complexe)	25/05/21	25/05/21
Génération de la documentation Doxygen	25/05/21	25/05/21
Génération de l'installeur	25/05/21	27/05/21
Tests	27/05/21	28/05/21
Exécution des tests et retours dans la documentation	27/05/21	28/05/21
Documentation des erreures restantes	28/05/21	28/05/21
Conclusion	31/05/21	03/06/21
Comparaison entre la planification et le travail réalisé	31/05/21	31/05/21
Rédaction de la conclusion	01/06/21	03/06/21

Planification finale

2.5 Dossier de conception

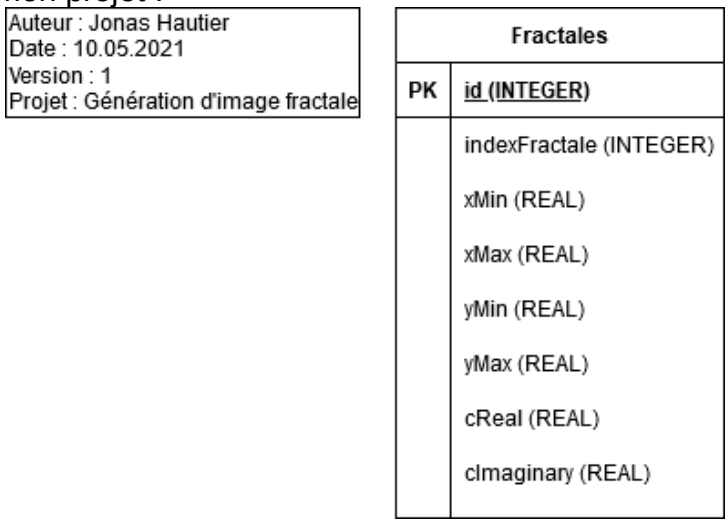
Pour le matériel hardware, j'utiliserai un poste de 4e année du CPNV :

Propriété	Valeur
OS	Windows 10
Architecture 32/64 bits	64 bits
RAM	16 Go
Nombre de processeurs logiques	8

Ci-dessous, vous trouverez la liste des logiciels utilisés dans mon projet avec l'utilisation qui en sera faite et la version utilisée.

Logiciel	Utilisation	Version
7-Zip	Créer des archives .zip	16.04
DB Browser for SQLite	Tester et modéliser ma BDD	3.12.1
diagrams.net (anciennement draw.io)	Faire les différents diagrammes (BDD et code)	En ligne
Doxygen	Générer la documentation logicielle	1.9.1
Excel 2019	Tenir mon journal de travail	MS Office 2019
GanttProject	Faire les planifications initiale et finale	2.8.1
GitHub Desktop	Publier sur GitHub et versionner avec git	2.7.1
Pencil	Faire les maquettes du programme	2.0.5
Visual Studio 2019	Réaliser le programme et les tests	16.4.2
Word 2019	Rédiger ma documentation	MS Office 2019
XaoS	Tester la génération de fractales	4.2.1

Voici le MLD de mon projet :



MLD

Voici le diagramme de classes du programme :

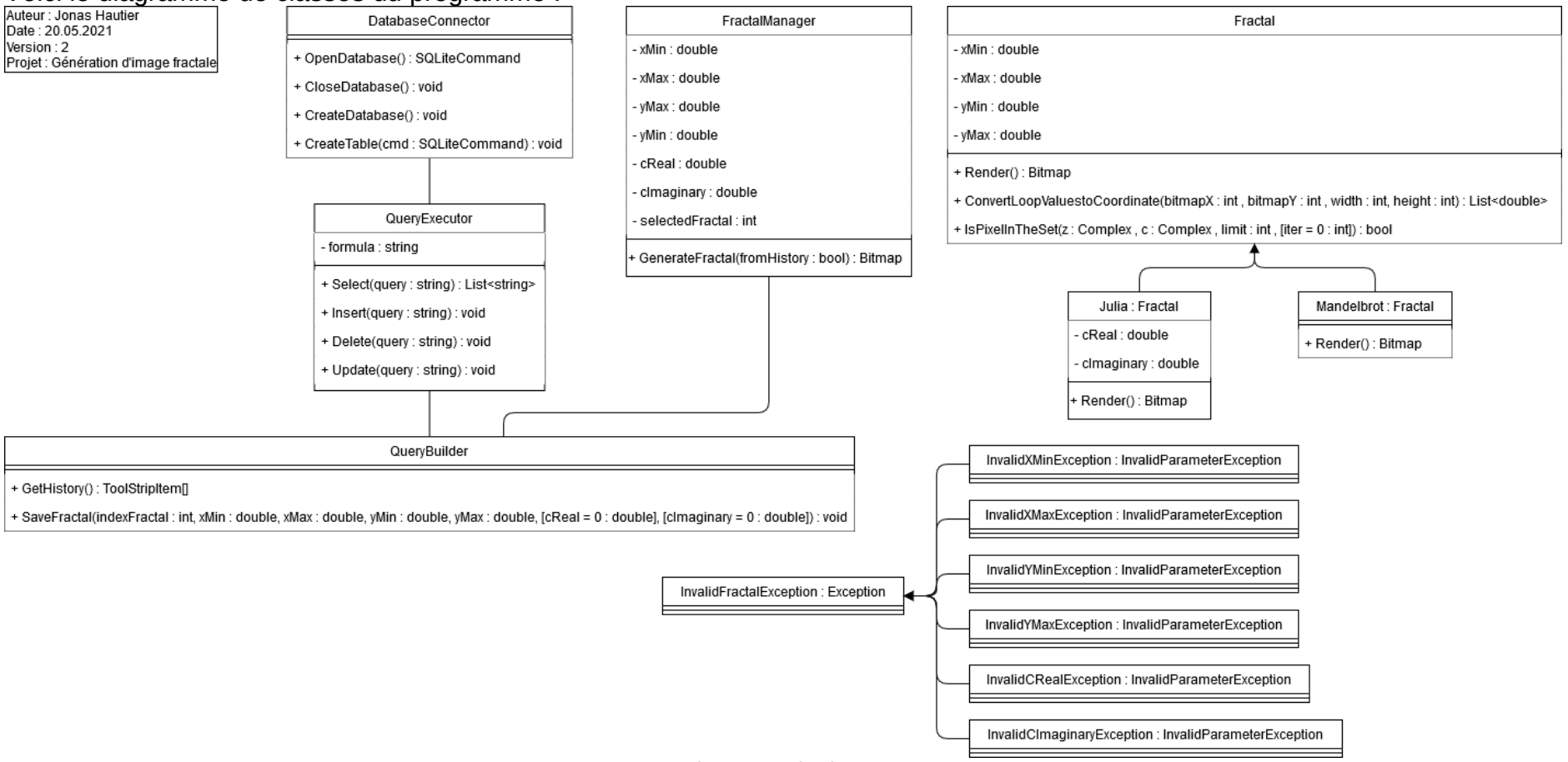


Diagramme de classes

Voici le diagramme de séquence du cas « On génère l'ensemble de Mandelbrot » :

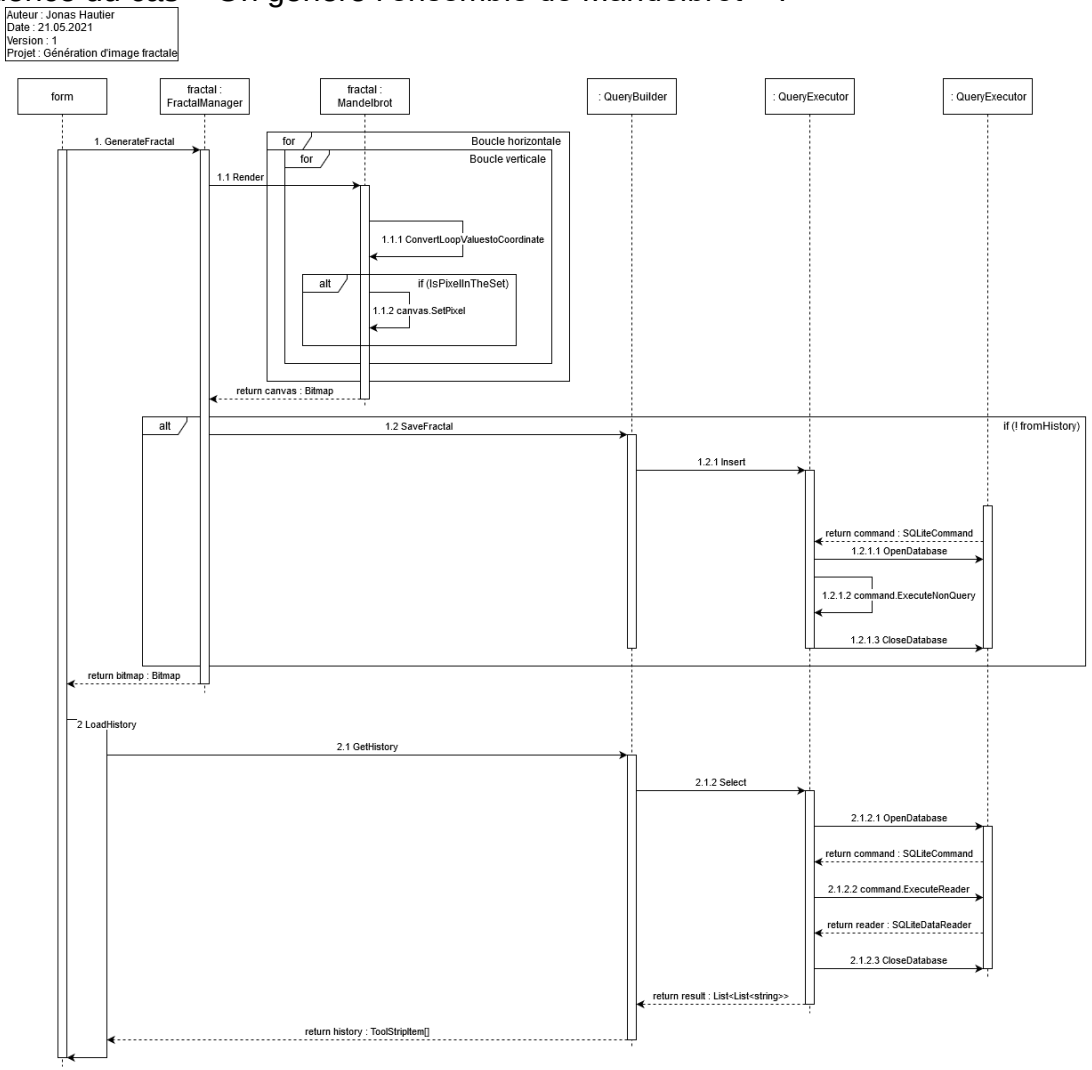


Diagramme de séquence du cas « On génère l'ensemble de Mandelbrot »

3 Réalisation

3.1 Dossier de réalisation

3.1.1 Structure du répertoire « Code »

A côté de chaque fichier ou dossier importants se trouve une description de son utilité. Les dossiers sont en italiques pour mieux les différencier des fichiers. Des points de suspension indiquent que les fichiers ou dossiers se trouvant dans le répertoire ne nécessitent pas de description.

Fichier / dossier	Description
GenerationImageFractale	Projet Visual Studio du logiciel
Properties	Les propriétés du projet
...	
App.config	Un des fichiers de configuration du logiciel
DatabaseConnector.cs	Classe gérant la connexion à la BDD
Exception.cs	Exception personnalisée pour le logiciel
Fractal.cs	Classe représentant une fractale
FractalManager.cs	Classe de gestion des fractales
GenerationImageFractale.csproj	Fichier de projet du logiciel
GenerationImageFractale.sln	
Julia.cs	Classe représentant une Julia
Main.Designer.cs	Crée les composants de la fenêtre du logiciel
Main.cs	La fenêtre du programme
Main.resx	
Mandelbrot.cs	Classe représentant un Mandelbrot
Program.cs	Point d'entrée du programme
QueryBuilder.cs	Classe créant les requêtes pour la BDD
QueryExecutor.cs	Classe exécutant les requêtes à la BDD
packages.config	Fichier de configuration des packages NuGet.
Setup	Projet Visual Studio de l'installeur
Setup.vdproj	Fichier de projet de l'installeur

3.1.2 Liste des paquets NuGet utilisés

Logiciels	Auteur(s)	Version
EntityFramework	Microsoft	6.3.0
MathParser.org-mXparser	Mariusz Gromada	4.4.2
Stub.System.Data.SQLite.Core.NetFramework	SQLite Development Team	1.0.113.3
System.Data.SQLite	SQLite Development Team	1.0.113.7
System.Data.SQLite.Core	SQLite Development Team	1.0.113.7
System.Data.SQLite.EF6	SQLite Development Team	1.0.113
System.Data.SQLite.Linq	SQLite Development Team	1.0.113

3.2 Résultat des tests effectués

Module	ID du test	Test passé ?
Interface du programme	1.1	Oui
	1.2	Oui
	1.3	Oui
	1.4	Oui
	1.5	Oui
	1.6	Oui
	1.7	Oui
Génération de l'ensemble de Mandelbrot	2.1	Oui
	2.2	Oui
	2.3	Oui
Génération de l'ensemble de Julia	3.1	Oui
	3.2	Oui
	3.3	Oui
Historique des générations	4.1	Oui
	4.2	Oui
	4.2.1	Oui
	4.2.2	Oui
	4.3	Oui
	4.4	Oui
	4.5	Oui
Sauvegarde de la fractale générée	5.1	Oui
	5.2	Oui
	5.3	Oui
	5.4	Oui

3.3 Erreurs restantes

Ci-dessous, vous trouverez la liste des erreurs restantes identifiées dans le programme :

- Il est possible de générer une fractale avec une borne inférieure plus grande que la borne supérieure.

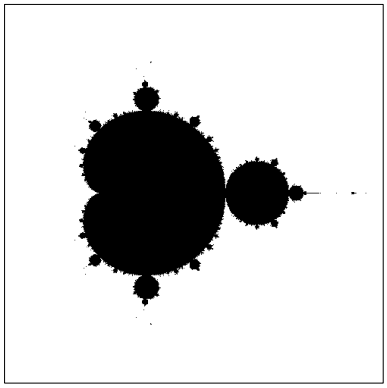
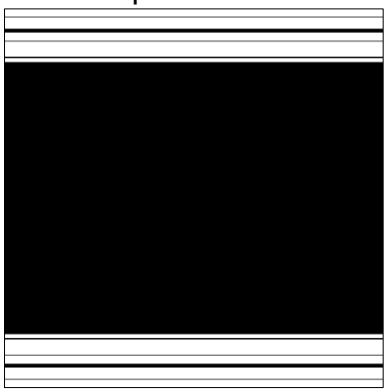
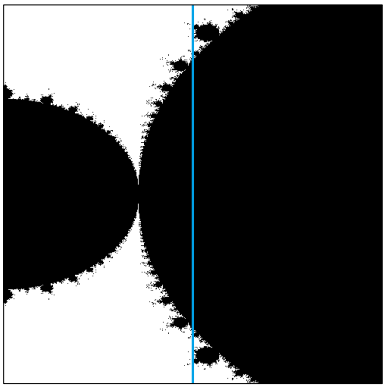


Image illustrant la symétrie d'axe, générée avec pour valeurs « x : 1 à -2 ; y : -1,5 à 1,5 »

Cela a pour conséquence d'effectuer une symétrie axiale sur la fractale, l'axe où les valeurs n'ont pas été inversée étant l'axe de symétrie. J'ai laissé cette fonctionnalité car il me semble qu'elle peut être utile.

- Il est possible de générer une fractale avec la même valeur comme borne inférieure et supérieure.





A gauche, un exemple de « tranche » du Mandelbrot. A droite, la « tranche », indiquée par la ligne bleue, dans l'ensemble de Mandelbrot. Voici les valeurs utilisées pour la génération :
Image de gauche : « x : -0.65 à -0.65 ; y : -0.5 à 0.5 »
Image de droite : « x : -1 à -0.3 ; y : -0.5 à 0.5 »

Cela a pour conséquence d'afficher une « tranche » de la fractale. Je ne pense pas que cette fonctionnalité puisse servir à beaucoup de monde, mais je ne vois pas non plus de raison d'empêcher ce cas.

3.4 Liste des documents fournis

Voici la liste des documents fournis dans l'archive de rendu :

- Code.zip
  - Code source du programme
- Documentation.zip
  - Documentation.docx
  - Journal de travail.xlsm
  - Manuel d'installation.docx
  - Manuel d'utilisation.docx
  - Exports.zip
    - Tous les documents exportés, au format pdf ou png

4 Conclusions

Je suis satisfait du résultat final, car il valide les objectifs définis en début de projet. Je suis étonné de ne pas avoir eu beaucoup de difficultés pour implémenter les algorithmes de génération des fractales.

Je suis content de ce projet. Selon moi, Il n'était ni trop difficile, ni trop simple, ce qui l'a rendu pertinent et gratifiant

4.1 Objectifs

Tous les objectifs ont pu être accomplis.

Pour la plupart, ils n'étaient pas particulièrement difficiles, la dynamisation de l'interface est un bon exemple d'objectif relativement simple. Cependant, j'ai eu un peu de peine à implémenter la fonctionnalité de régénération de fractale depuis l'historique. Le défi de cette fonctionnalité était de stocker les paramètres de génération de la fractale dans la ligne de l'historique et de les récupérer lors de la régénération.

4.2 Points positifs et négatifs

Parmi les points positifs, je peux citer le fait que j'ai terminé le programme et que les objectifs ont été atteints. Un autre point positif est que j'ai pu beaucoup apprendre en C#, notamment découvrir la structure `Complex`, la gestion de Bitmap ou encore la génération de fractales.

Pour les points négatifs, je suis déçu de pas avoir pu aller plus loin que les objectifs qui m'étaient données. Cependant, je suis satisfait d'avoir, au moins, pu atteindre les objectifs.

4.3 Difficultés

Durant ce TPI, j'ai rencontré différentes difficultés et je vais en documenter dans ce paragraphe. Parmi ces difficultés, il y a eu la génération de l'installeur, où je manquais passablement de connaissances, et la rédaction du manuel d'utilisation, où la difficulté était de trouver le bon niveau de détails pour que le manuel soit à la fois compréhensible pour un néophyte, mais pas trop détaillé pour éviter de perdre le lecteur.

J'ai également comparé les risques techniques imaginés durant l'analyse avec la réalisation. Voici les résultats :

- *Manipuler des nombres complexes en C# et les connecter correctement à mon parser*.
  Au final, je n'ai pas eu besoin d'utiliser le parser avec des nombres complexes. Le parser est uniquement utilisé pour vérifier que les valeurs saisies depuis l'interface sont valides.
- *Vitesse de génération de la fractale et parallélisme*.
  Au final, je n'ai pas eu besoin d'implémenter de parallélisme dans la génération des fractales étant donné qu'elle est suffisamment rapide.

- *Mauvaise gestion du temps dans l'exécution du CdC*.
  Je trouve que la gestion du temps n'a pas été un problème pour ce projet. J'ai pu atteindre tous les objectifs et documenter mon travail correctement.

4.4 Suite pour le projet

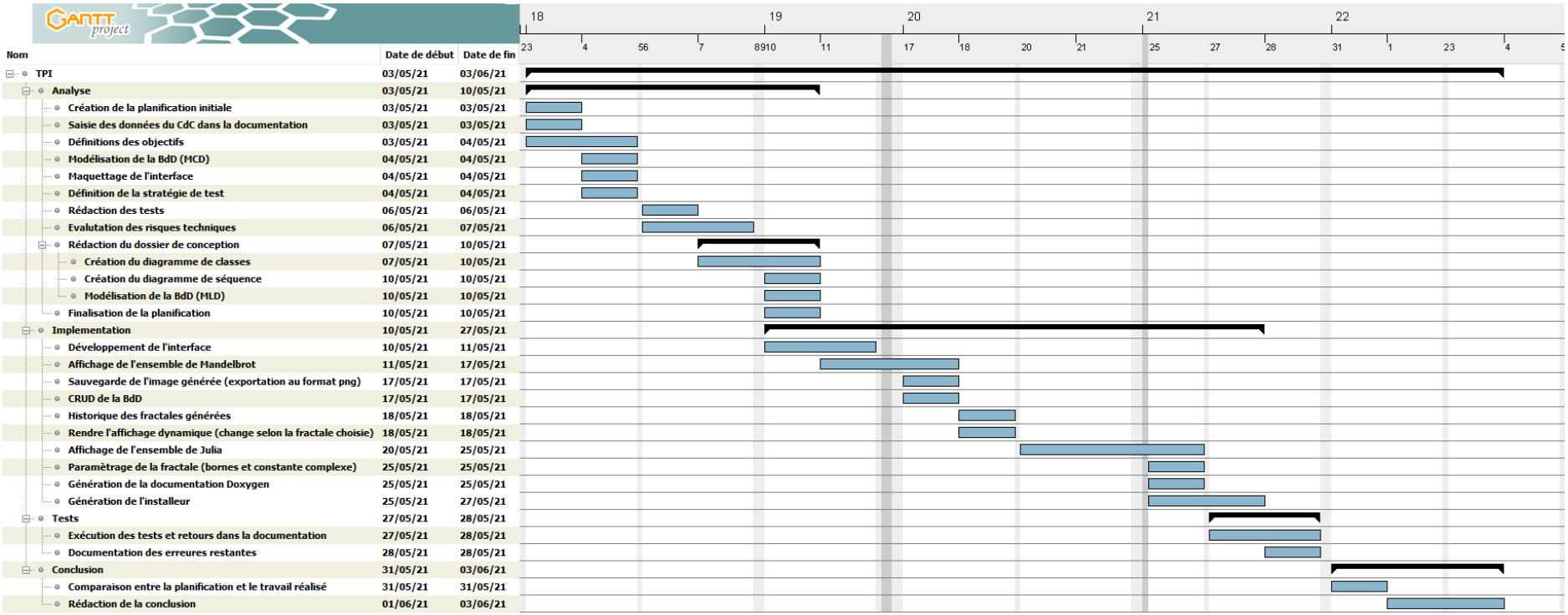
Il y plusieurs fonctionnalités que j'aimerai ajouter à ce projet après avoir terminé le TPI :

- Implémenter une palette de couleur pour la génération des fractales.
  - Pour aller plus loin, la palette pourrait être personnalisable et de taille variable.
- Donner la possibilité de paramétrer le nombre d'itération utilisé pour générer les fractales.
- Dans l'historique, afficher une miniature de la fractale.

Je me réjouis d'essayer d'implémenter ces fonctionnalités, particulièrement celle de la miniature, pour laquelle je pense que la difficulté est la plus haute par rapport à mes connaissances.

4.5 Comparaison entre la planification et le travail réalisé

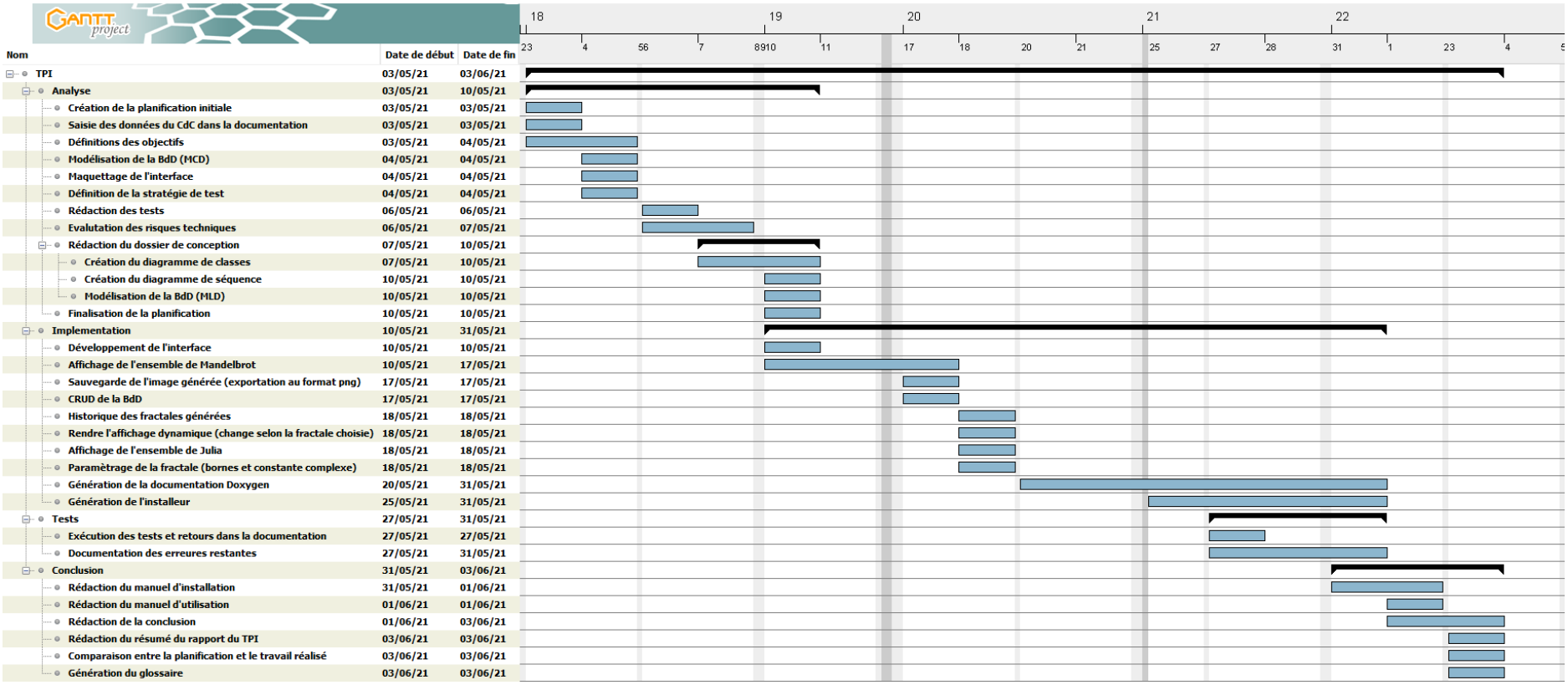
Sur cette page, vous trouverez la planification finale, le temps passé par partie (Analyse, Implémentation, …) et le nombre de tâches planifiées. En page suivante, vous trouverez le temps réellement passé sur chaque tâche et, à nouveau, le temps passé par partie (Analyse, Implémentation, …) et le nombre de tâches planifiées. Vous y trouverez également une réflexion sur les différences entre la planification et le travail réalisé.



Nom	Date de début	Date de fin
TPI	03/05/21	03/06/21
Analyse	03/05/21	10/05/21
Création de la planification initiale	03/05/21	03/05/21
Saisie des données du CdC dans la documentation	03/05/21	03/05/21
Définitions des objectifs	03/05/21	04/05/21
Modélisation de la BdD (MCD)	04/05/21	04/05/21
Maquettage de l'interface	04/05/21	04/05/21
Définition de la stratégie de test	04/05/21	04/05/21
Rédaction des tests	06/05/21	06/05/21
Evalutation des risques techniques	06/05/21	07/05/21
Rédaction du dossier de conception	07/05/21	10/05/21
Création du diagramme de classes	07/05/21	10/05/21
Création du diagramme de séquence	10/05/21	10/05/21
Modélisation de la BdD (MLD)	10/05/21	10/05/21
Finalisation de la planification	10/05/21	10/05/21
Implementation	10/05/21	27/05/21
Développement de l'interface	10/05/21	11/05/21
Affichage de l'ensemble de Mandelbrot	11/05/21	17/05/21
Sauvegarde de l'image générée (exportation au format png)	17/05/21	17/05/21
CRUD de la BdD	17/05/21	17/05/21
Historique des fractales générées	18/05/21	18/05/21
Rendre l'affichage dynamique (change selon la fractale choisie)	18/05/21	18/05/21
Affichage de l'ensemble de Julia	20/05/21	25/05/21
Paramètrage de la fractale (bornes et constante complexe)	25/05/21	25/05/21
Génération de la documentation Doxygen	25/05/21	25/05/21
Génération de l'installeur	25/05/21	27/05/21
Tests	27/05/21	28/05/21
Exécution des tests et retours dans la documentation	27/05/21	28/05/21
Documentation des erreures restantes	28/05/21	28/05/21
Conclusion	31/05/21	03/06/21
Comparaison entre la planification et le travail réalisé	31/05/21	31/05/21
Rédaction de la conclusion	01/06/21	03/06/21

Planification finale

Temps pour chaque partie : Analyse : 4 jours, Implémentation : 9 jours, Tests : 2 jours, Conclusion : 3 jours.
Nombre de tâches : 26 tâches.



Nom	Date de début	Date de fin
TPI	03/05/21	03/06/21
Analyse	03/05/21	10/05/21
Création de la planification initiale	03/05/21	03/05/21
Saisie des données du CdC dans la documentation	03/05/21	03/05/21
Définitions des objectifs	03/05/21	04/05/21
Modélisation de la BdD (MCD)	04/05/21	04/05/21
Maquettage de l'interface	04/05/21	04/05/21
Définition de la stratégie de test	04/05/21	04/05/21
Rédaction des tests	06/05/21	06/05/21
Evalutation des risques techniques	06/05/21	07/05/21
Rédaction du dossier de conception	07/05/21	10/05/21
Création du diagramme de classes	07/05/21	10/05/21
Création du diagramme de séquence	10/05/21	10/05/21
Modélisation de la BdD (MLD)	10/05/21	10/05/21
Finalisation de la planification	10/05/21	10/05/21
Implementation	10/05/21	31/05/21
Développement de l'interface	10/05/21	10/05/21
Affichage de l'ensemble de Mandelbrot	10/05/21	17/05/21
Sauvegarde de l'image générée (exportation au format png)	17/05/21	17/05/21
CRUD de la BdD	17/05/21	17/05/21
Historique des fractales générées	18/05/21	18/05/21
Rendre l'affichage dynamique (change selon la fractale choisie)	18/05/21	18/05/21
Affichage de l'ensemble de Julia	18/05/21	18/05/21
Paramètrage de la fractale (bornes et constante complexe)	18/05/21	18/05/21
Génération de la documentation Doxygen	20/05/21	31/05/21
Génération de l'installeur	25/05/21	31/05/21
Tests	27/05/21	31/05/21
Exécution des tests et retours dans la documentation	27/05/21	27/05/21
Documentation des erreures restantes	27/05/21	31/05/21
Conclusion	31/05/21	03/06/21
Rédaction du manuel d'installation	31/05/21	01/06/21
Rédaction du manuel d'utilisation	01/06/21	01/06/21
Rédaction de la conclusion	01/06/21	03/06/21
Rédaction du résumé du rapport du TPI	03/06/21	03/06/21
Comparaison entre la planification et le travail réalisé	03/06/21	03/06/21
Génération du glossaire	03/06/21	03/06/21

Temps réel

Temps pour chaque partie : Analyse : 4 jours, Implémentation : 11 jours, Tests : 3 jours, Conclusion : 3 jours.
Nombre de tâches : 30 tâches.

Au niveau du temps passé par partie, l'implémentation et les tests ont les deux pris un jour de plus que prévu. Cela est dû à un bug que je n'avais pas remarqué. Les paramètres saisis pour la fractale « Julia » n'étaient pas stockés dans l'historique.
Concernant le nombre de tâches, la différence est due aux tâches ajoutées à la partie « Conclusion ».

5 Glossaire

BDD
: base de données 3, 11, 14, 23

CdC
: cahier des charges 3, 9, 19, 23

MCD
: modèle conceptuel de données 3, 6

MLD
: modèle logique de données 3, 11

NTFS
: New Technology File System. C'est un système de fichiers utilisé sur Windows depuis NT 3.1 9

parser
: analyseur syntaxique en français 9, 18

SGBD
: système de gestion de base de données 3, 23

6 Annexes

En annexe, vous trouverez, dans cet ordre, le journal de travail, le manuel d'installation et le manuel d'utilisation.

6.1 Résumé du rapport du TPI

Dans le cadre de ma formation d'informaticien CFC, j'ai eu l'occasion de réaliser différents projets pour me préparer au TPI. Le TPI, selon mon CdC, doit être réalisé en 90h. Mon projet est un générateur d'images fractales. Le but de mon programme est de générer des images de l'ensemble de Mandelbrot ou de l'ensemble de Julia. Les images générées sont en noir en blanc.

Le programme, réalisé en C#, utilisant SQLite comme SGBD et versionné avec git, a une interface épurée. Il donne à l'utilisateur la possibilité de configurer les limites inférieures et supérieures des axes réels et imaginaires. Pour suivre attentivement mon projet, j'ai créé une copie de ma planification finale et l'ai tenue à jour tout au long du projet pour pouvoir comparer temporellement ce qui était planifié à ce qui a été réalisé. J'ai également tenu un journal de travail quotidiennement, comme demandé.

Je suis content d'avoir pu arriver au bout de ce projet. Malgré quelques cas particuliers, décrits en détails dans la partie 3.3 Erreurs restantes, je n'ai pas trouvé de bugs dans mon programme. Je suis satisfait de ce que j'ai pu apprendre au cours de ce projet et n'ai pas de regret.

6.2 Sources – Bibliographie

Aide externes :

- 06.05.2021 – Louis Richard (camarade de classe) :
  Louis m'a aidé pour détecter des cas qui n'auraient pas été pris en compte dans mes scénarios.
- 20.05.2021 – Frédérique Andolfatto (cheffe de projet)
  Madame Andolfatto m'a aidé pour résoudre un problème de structure de classe. J'ajoutais une gestion de clic dans une classe liée à la BDD et cela rendait la classe inutilisable car elle devait être statique et non-statique à la fois.

Sites web :

- 18.05.2021 https://complex-analysis.com/content/julia_set.html
  https://youtu.be/oCkQ7WK7vuY (vidéo)
  Ces sources m'ont permis de mieux comprendre comment je devais adapter mon algorithme de génération du Mandelbrot pour pouvoir générer l'ensemble de Julia.
- 01.06.2021 https://fr.wikipedia.org/wiki/Fractale
  https://fr.wikipedia.org/wiki/Autosimilarité
  https://fr.wikipedia.org/wiki/Plan_complexe
  https://fr.wikipedia.org/wiki/Coordonnées_cartésiennes
  https://fr.wikipedia.org/wiki/Nombre_complexe
  https://fr.wikipedia.org/wiki/Ensemble_de_Mandelbrot

https://fr.wikipedia.org/wiki/Ensemble_de_Julia

Ces sources m'ont permis de rédiger correctement la partie « Quelques définitions » du manuel d'utilisation.