Plan du projet

# Introduction

# Présentation du projet

## Définition générale du projet

Le projet « Grenouille et crapaud » est un jeu où deux équipes (une équipe crapaud et l’autre celle des grenouilles) situé des deux extrémités du plateau de jeu vont s’affronter jusqu’à bloquer l’adversaire ou être arrivé à l’autre bout du plateau le premier. Il y pourra aussi jouer dans un casse-tête, le but du casse-tête est que le joueur pourra contrôler les deux équipes, le jeu se terminera quand tous les pions auront traversé le plateau ou quand des pions seront bloqués.

## Cahier des charges

Le but de ce projet est de développer un programme permettant de jouer au jeu et au casse-tête des grenouilles et des crapauds.

### Description du jeu des grenouilles et des crapauds

Le jeu oppose deux joueurs et consiste à déplacer des grenouilles et des crapauds sur un plateau quadrillé de N lignes et M colonnes. En début de partie, les grenouilles sont attribuées à l'un des joueurs, les crapauds à l'autre. Le détenteur des grenouilles commence, et ensuite les joueurs jouent à tour de rôle. Le premier qui ne peut plus avancer ses batraciens a perdu.

Les grenouilles avancent horizontalement de la gauche vers la droite, et les crapauds de la droite vers la gauche.

Si la case adjacente dans le sens de déplacement est libre, la grenouille (respectivement le crapaud) déplacée vient occuper cette case. Si, par contre, elle est occupée par un crapaud (respectivement une grenouille) et que la case suivante est libre, le batracien vient occuper cette case libre. Dans tous les autres cas, le batracien ne peut être déplacé.

**Exemple avec un plateau de 1 \* 5:**

1. Plateau initial

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G** | **G** |  | *C* | *C* |

* 1. 2 3 4 5

1. La grenouille de la case 2 avance en 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G** |  | **G** | *C* | *C* |

* 1. 2 3 4 5

1. Le crapaud de la case 4 avance en 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G** | *C* | **G** |  | *C* |

* 1. 2 3 4 5

1. La grenouille de la case 3 avance en 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G** | *C* |  | **G** | *C* |

* 1. 2 3 4 5

1. Le crapaud de la case 5 avance en 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G** | *C* | *C* | **G** |  |

* 1. 2 3 4 5

1. La grenouille de la case 4 avance en 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **G** | *C* | *C* |  | **G** |

* 1. 2 3 4 5

1. Le joueur qui détient les crapauds a perdu puisque ceux-ci ne peuvent plus avancer.

### Description du casse-tête des grenouilles et des crapauds

Les règles de déplacement sont identiques à celles du jeu. Cependant les déplacements des grenouilles et des crapauds n'alternent pas obligatoirement. Le but de l'unique joueur est d'amener toutes les grenouilles à droite et tous les crapauds à gauche du plateau. Si cette disposition est atteinte, la partie de casse-tête est gagnée.

Exemple de disposition gagnante :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *C* | *C* |  | **G** | **G** |
|  |  | | |  |
| **G** | *C* | *C* |  | **G** |

Exemple de disposition perdante :

### Application à développer

A son lancement l’application proposera à l’utilisateur d’afficher une aide pour son utilisation.

L’application devra permettre :

* À deux joueurs de s’affronter
* À un unique joueur de jouer contre l’ordinateur
* À un unique joueur d’utiliser le casse-tête
* De gérer des configurations initiales

Dans le cas d’une partie entre deux joueurs, ceux-ci devront saisir leur nom ce qui permettra ensuite à l’application de bien spécifier lequel des deux doit jouer. Les grenouilles et les crapauds seront attribués aux deux joueurs de manière aléatoire. En fin de partie, le nom du gagnant sera clairement affiché.

Dans le cas d’une partie contre l’ordinateur, ou bien lors de l’utilisation du casse-tête, l’application proposera à l’unique joueur d’entrer son nom. S’il n’en donne aucun, il sera désigné par un « joueur humain ». En fin de partie, un message indiquera le résultat de celle-ci et spécifiera le nom de l’unique joueur.

Après une partie, l’application affichera à nouveau le menu principal, ce qui permettra à l’unique joueur ou aux deux joueurs de lancer, éventuellement, une nouvelle partie.

### Configurations initiales

Une **configuration** est la description d'un plateau dans un état précis. Elle indique donc la taille du plateau (nombre de lignes et de colonnes) et la disposition des grenouilles et des crapauds présents sur le plateau. La taille du plateau aura une limite supérieure à fixer (liée aux contraintes d’affichage).

Pour spécifier la configuration initiale d’une partie, l'utilisateur choisira l'une des deux alternatives suivantes :

* Entrer au clavier cette configuration.
* Sélectionner une configuration préétablie. Par conséquent, le programme devra permettre de gérer un ensemble de configurations préétablies que l'utilisateur pourra modifier à son gré par insertion ou retrait d'une configuration. En particulier, lorsque l'utilisateur aura saisi une configuration initiale, le programme lui demandera s'il souhaite l'inclure dans l'ensemble des configurations préétablies

Le placement des batraciens sur le plateau est libre. Toutefois, l’application vérifiera qu’il y a autant de batraciens de chaque catégorie. Si ce n’est pas le cas, la configuration sera considérée comme invalide.

Exemple de configuration initiale :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **G** |  |  | *C* |  | *C* |
|  | *C* | **G** |  | **G** | **G** |  |
| **G** | **G** |  |  | *C* | *C* |  |
|  | **G** | *C* | **G** |  | *C* | ***G*** |
|  | **G** | *C* |  | *C* | *C* |  |

Les configurations préétablies devront être stockées en mémoire permanente afin d’être retrouvées lors du prochain lancement de l’application.

### Remarque

L’application fonctionnera en mode console ou bien sera dotée d’une interface graphique, au choix du groupe réalisant le projet.

## Définitions et acronymes

## Charte du projet

### Objectif du projet

L’objectif de ce projet sera la réalisation d’un jeu vidéo « Grenouille et crapaud » avec une interface.

### Périmètre du projet

Les périmètres du projet sont :

* Gestion des Pions et des plateaux.
* Sauvegarde des données du jeu dans un plateau.
* Création d’une IA.
* Création d’une interface.

### Demandes hors périmètre

Les demandes hors périmètre sont :

* Création d’un compteur pour le nombre de déplacement dans le casse-tête.
* Création de plusieurs niveaux d’IA.

### Principaux livrable

1er Livrable :

Premier aperçu d’un jeu fonctionnel permettant de jouer contre un joueur mais aussi en casse-tête.

Premier aperçu de l’interface fonctionnelle mais qui n’est pas lier avec le code.

2eme Livrable :

Premier aperçu du jeu et de l’interface ensemble.

Possibilité de jouer contre une IA très facile à battre.

Possibilité de sauvegarder une configuration

Apparition d’un nombre de coups dans le casse-tête

### Cadre : coûts, délai, qualité

### Acteur du projet : équipe MOE, MOA, contrôle qualité

Les différentes équipes dans le projet :

L’équipe MOE :

* Mael Le folgoc pontis
* Piere marie Combalbert (Gestionnaire de configuration)
* Yoan Guiraud (secrétaire)
* Yanis Delmas (Chef de projet)

L’équipe MOA :

* Corinne Servieres

Contrôle qualité :

* Mael Le folgoc pontis

### Autre moyens et ressources

### Condition d’acceptation : exigences, contraintes, qualité, critères et mesures

### Principaux risque identifiés et politique de gestion des risques

### Risque identifier

Les risques que nous avons identifiés :

* Des javadocs mal écrite ou oublié d’être remplis.
* Erreurs dans les conventions java.
* Erreur simple dans le code (oublie de vider le tampon, etc.).
* Mauvaise compréhension du code à programmer par les programmeurs.

### Politique de gestion des risques

* Création automatique de « TODO Commentaire pas terminer » lors de la création d’un commentaire par Eclipse.
* Relecture du code par l’équipe du contrôle de qualité et vérification des conventions par Eclipse.
* Vérification par les tests et vérification usuel par l’équipe du contrôle de qualité (Test durant une partie de jeu pour découvrir des erreurs ou voir le bon fonctionnement du jeu).
* Réunion organisée pour éviter de mauvaise compréhension et réorganiser les idées du groupe.

## Etude générale du besoin

# Organisation du projet

## Présentation du cycle de vie et ses conséquences

## Identifier les rôles

Notre projet est composé de 4 membres avec différents rôles :

* Responsable MOA : Corinne Servières
* Responsable gestion projet : Frédéric Barrios
* Chef de projet MOE et MOA : Yanis Delmas
* Secrétaire de projet : Yoan Guiraud
* Gestionnaire de configuration : Pierre-Marie Combalbert

Programmeur : Maël Le-Folgoc-Pontis

## Plan de communication

### Localisation géographique des intervenants

Yoan : Foyer saint pierre Rodez (12)

Pierre-Marie : 26 avenue de Montpelier Rodez (12)

Yanis : Rue de Saunhac Rodez (12)

Maël : Foyer Sainte Thérèse Rodez (12)

### Moyens de communication utilisés

Les moyens de communication utilisés pour les réunions et les échanges de messages sont faits avec Discord, le compte-rendu et les autres documents comme le plan projet par exemple sont réalisés avec Word (format .docx) et peuvent être fournis en PDF (format .pdf) et tout le suivi de l’avancement du projet avec les sauvegardes continuent des documents et de l’avancée du projet se fait à l’aide de GitHub.

Nous utilisons une communication centralisée grâce au logiciel de communication Discord, ce qui implique que toutes les parties prenantes soient reliées à une interface centrale qui stocke toutes les communications en un seul endroit, y compris toutes les documentations et informations nécessaires.

### Réunion projet MOE

### Pour l’organisation des réunions de groupe (MOE), nous nous sommes fixés à une ou deux réunions par semaines et elles ont pour but d'organiser, de planifier et d’améliorer les différents objectifs prévus pour l’itérations en cours.

### Et pour l’organisation des réunions MOA avec Corinne Servières, nous nous sommes fixés à une réunion toutes les deux à une semaine et demie (chaque itération) qui a pour objectif de faire un point sur l’itération en cours et la prochaine itération prévue, avec des améliorations, modifications à apporter sur le projet et avec des points incompris ou confus à éclaircir. Ces types de réunions durent en moyenne 30 minutes et se font principalement entre les inters cours ou en fin de journée.

### Comité de pilotage

## Assurance qualité

Nous suivons la convention Java du 12 septembre 1997 pour toute la programmation Java et le formalisme de modélisation est l’UML2 pour toute la partie conception orientée objet. Ensuite, nous utilisons la version 11 de Eclipse et JavaFx pour une corrélation des versions entre l’IDE et la programmation de l’interface graphique. Le cycle itératif nous impose d’avoir un contrôle de suivi toutes les deux semaines avec la MOA et nous permet de livrer des livrables à chaque itération.

### Manuel qualité et démarche qualité

### Le suivi et le contrôle de qualité adopté nous permet d’avoir un suivi continuel et permet d’avoir du recul sur le projet et les éléments manquants pour la globalité du projet.

## Ressource matérielle et logicielles

Logiciels et modules pour la programmation du projet :

|  |  |
| --- | --- |
|  | **Sublime text 3:**  Programmation, Mise en page du code… |
|  | **Eclipse:**  Programmation de l’application |
|  | **Version JAVA utilisée:**  JDK 11 |
|  | **Module pour la création de l’interface :**   * JAVA FX 11 * Efxclipse * Java Scene builder 2.0 |

Logiciels de partage fichier :

|  |  |
| --- | --- |
|  | **Git hub desktop :**  Mise en commun des fichiers de l’application après chaque modification |
|  | **Google Drive :**  Partage des documents du plan de projet qui sont plus facilement accessible sur les ordinateurs non équipés de git hub desktop |

Logiciels de communication dans le projet :

|  |  |
| --- | --- |
|  | **Discord :**  Logiciel pour communiquer et mettre en commun, organiser les réunions et les tâches à accomplir |
|  | **Modules de discord :**  Module pour avertir après chaque commit tous les utilisateurs du projet en les notifiant après chaque commit d’un message.  De plus chaque membre du projet possède l’application sur son pc et son téléphone pour être notifié en permanence |

Logiciels pour la rédaction des documents du projet :

|  |  |
| --- | --- |
|  | **Google docs :**  Pour rédiger des documents en direct avec les différents membres pour penser la rédaction des documents ensemble |
|  | W**ord :**  Rédactions des documents ne nécessitants pas d’être rédigé par plusieurs personne ou d’être accessible depuis plusieurs endroit |

Logiciels pour la présentation des documents :

|  |  |
| --- | --- |
|  | **Prezi :**  Afin de créer le diaporama final qui sera présenté le jour de l’oral de projet. |
|  | **Google Slides + Powerpoint :**  Création du diaporama de secours (Version hors ligne)  + Création au même moment des différentes diapositives et du choix des éléments à présenter (Non possible avec Prezi qui ne peut être modifié que par un seul utilisateur) |

Logiciels pour la modélisation:

|  |  |
| --- | --- |
|  | **Modelio :**  Création diagramme de cas d’utilisation + diagramme de classe |
|  | **Win Design :**  Création des premiers documents pour les premières échéances |

Ressource matérielle:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1er PC  Ecran:  VG248QE  Processeur:  I7-6700K  Carte graphique:  GTX 970  Carte mère:  ASUS MAXIMUS VIII  Ram:  32GB de RAM | 2ème PC  Ecran:  ILLYAMA  Processeur:  I5  Carte Graphique:  GTX 960  Carte mère:  MSI  Ram:  8GB de RAM | 3ème PC  Ecran:  Processeur:  AMD  Carte Graphique:  Radeon sapphire 7550  Carte mère  ASUS  Ram:  8GB de RAM | 4ème PC  ASUS X302LJ  (PORTABLE) | 5ème PC  PC PORTABLE | 6ème PC  Ordinateur Région |

# Pilotage du projet

## Cycle de vie

Nous avons choisi un modèle de cycle itératif car ce modèle nous permet de pouvoir de nous familiariser avec ce cycle de vie mais aussi ce cycle permet de pouvoir ajouter des besoins qui n’auraient pas était demander au départ de l’application.

## Planification prévisionnel initiale

### Ordonnancement des principales itérations/taches

Itération 1 :

* Création de l’objet Pion et des fonctions permettant de les gérer.
* Création de l’objet Plateau et des fonctions permettant de le gérer.
* Création de la configuration par défaut d’un plateau.
* Création du jeu (joueur contre joueur et casse-tête) sur la console.
* Création d’une interface fonctionnelle

Itération 2 :

* Création des fichiers permettant la sauvegarde des données sauvegarder sur le jeu
* Création des fonctions permettant de faire les sauvegardes sur le jeu
* Création d’une IA simple pour le premier niveau.
* Lier l’interface et le code.

### Identification des principaux jalons

Nous avons identifié 3 principaux jalons :

* 1er jalon est la fin de l’itération 1, C’est à partir de ce moment que les class plateau et plateau devront être finis et que le code du jeu joueur contre joueur doit être fini.
* 2e jalon est la fin de l’itération 2, c’est à ce moment où toutes les configurations devront être fini ainsi que l’interface. On devra avoir un jeu avec une interface fonctionnelle avec des fonctions manquante ainsi qu’une IA simplifier.
* 3e jalon est la fin du projet ( donc fin de l’itération 3), nous devrons donc avoir un jeu avec un interface complet et une IA un peu plus complexe.

### Calendrier prévisionnelle

Diagramme de Gantt de l’itération numéro 1 :

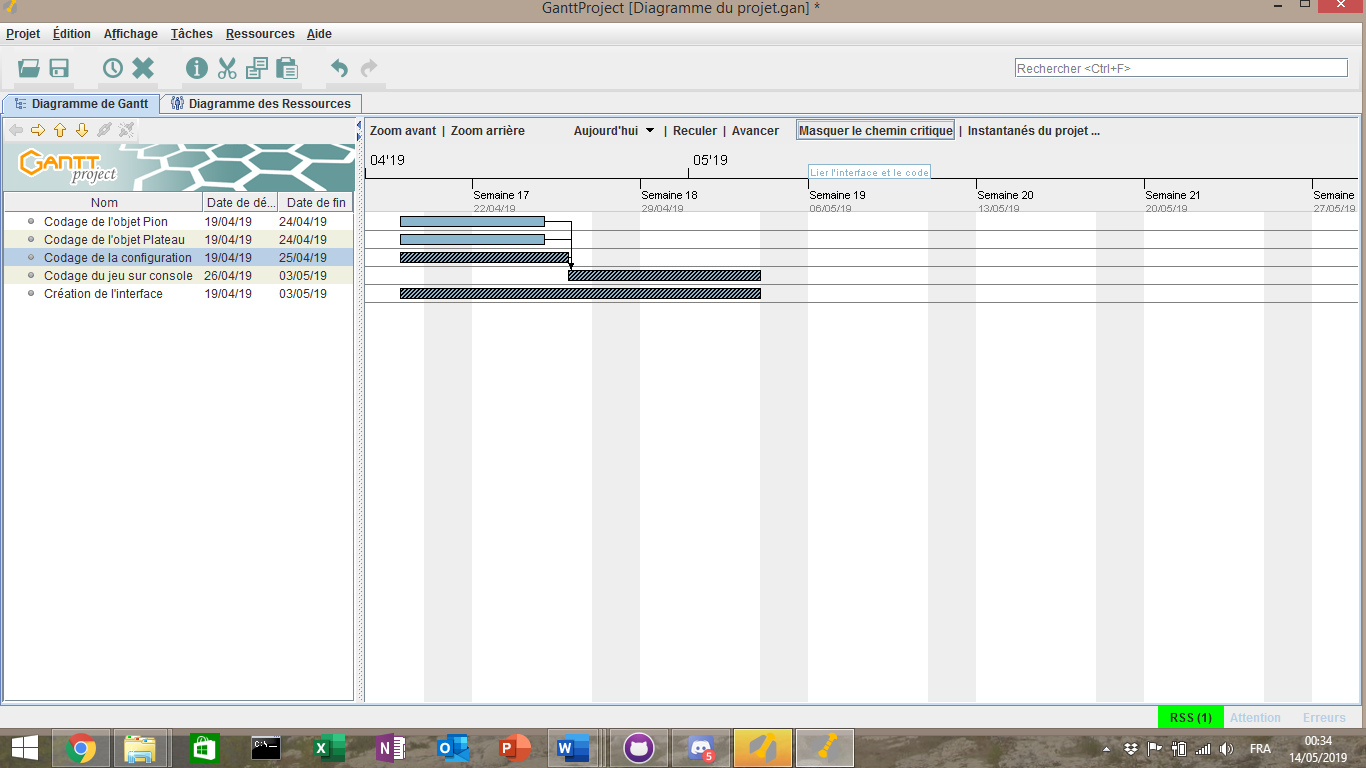


Diagramme de Gantt de l’itération numéro 2 :

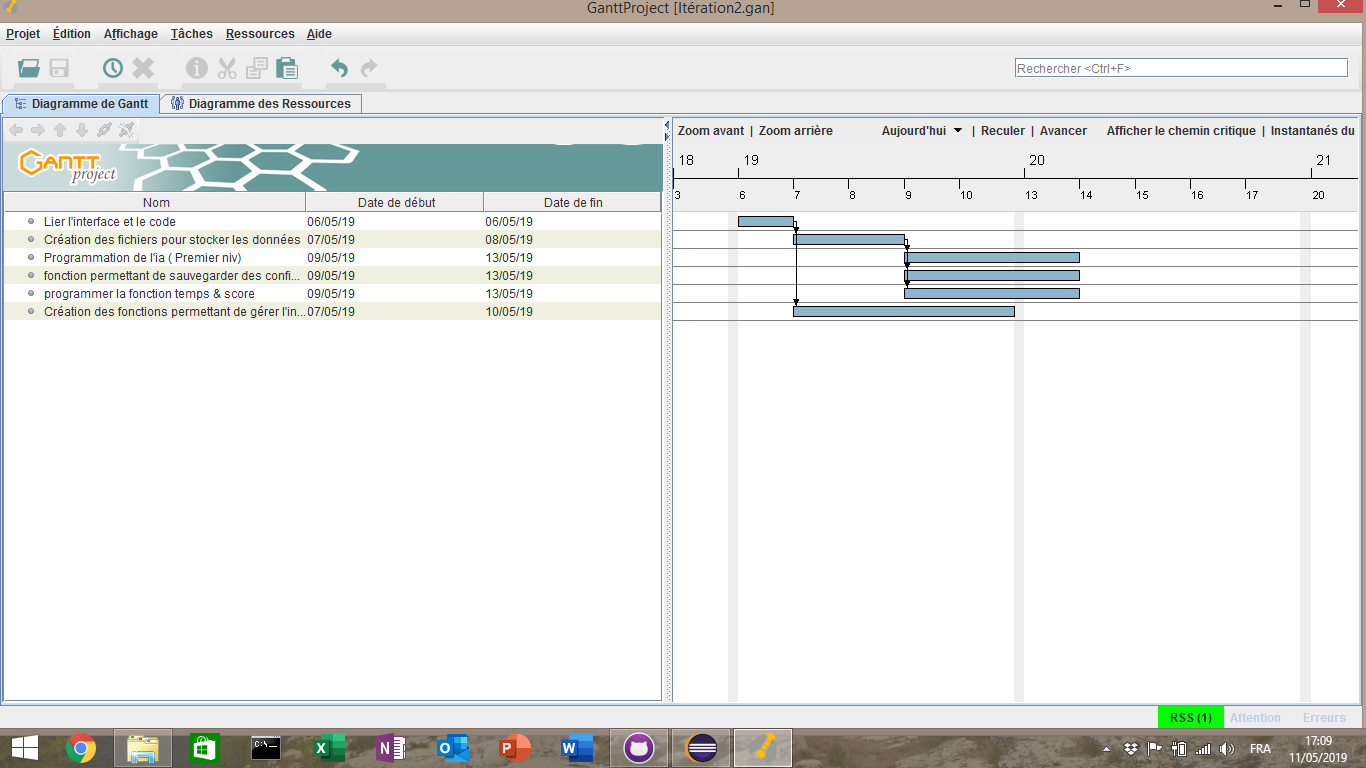
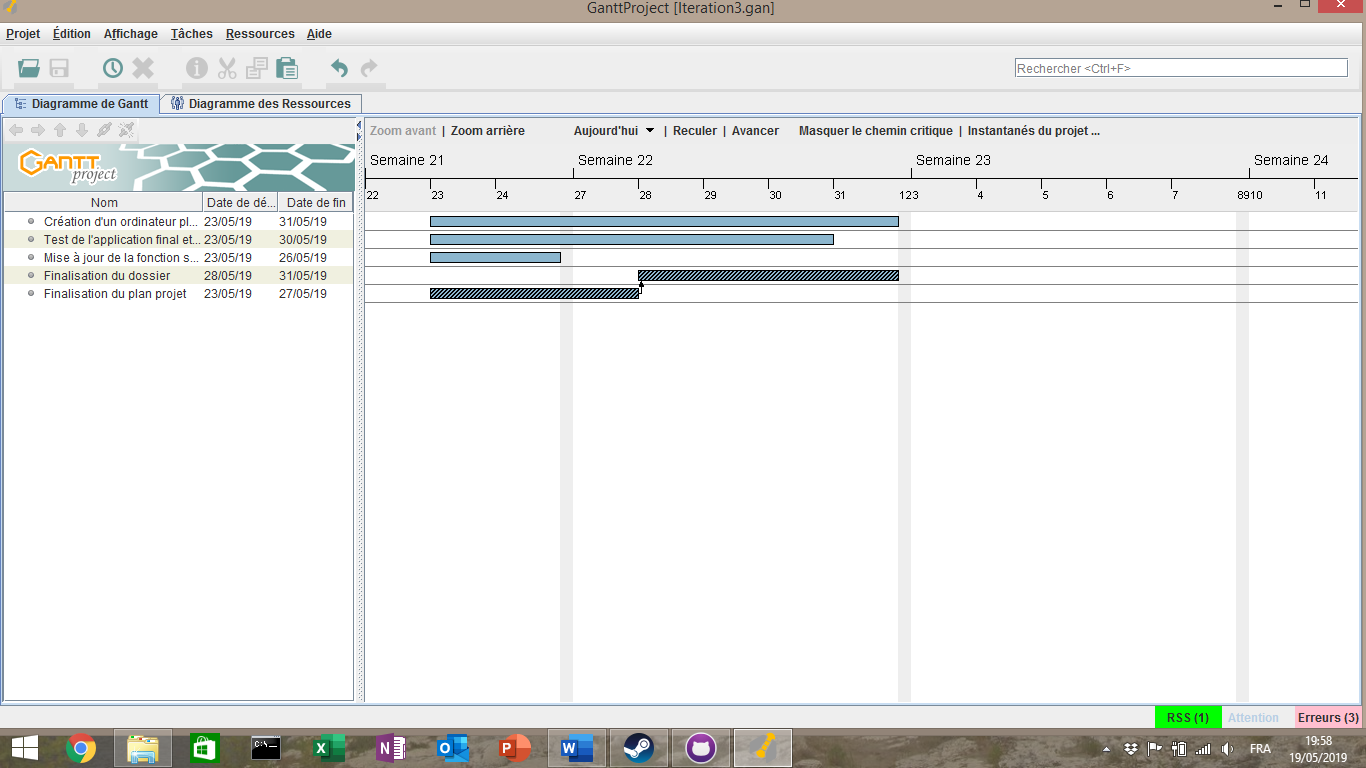


Diagramme de Gantt de l’itération 3 :



## Organisation des réunions et comités de pilotage

## Suivi du projet

### Suivi d’avancement et mesure des écarts par rapport au prévisionnel

### Synthèse par tableau de bord

### Résultat des tests et recette de prototype de la période

Itération 1 :

Tous les tests ont été bien réalisé sauf que suite à des modifications dans des codes certaines fonctions test ont dû être amélioré car ils y a eu de nouvelles erreurs à devoir être géré par notre application.

Quand au prototype, nous avons eu quelque petit bug mineur qui ont était vite réglé.

Itération 2 :

Certaines difficultés pour les tests avec l’interface mais ce fut réglé rapidement.

Ayant eux certains problèmes nous avons du reporté la démonstration du prototype de une semaines pour pouvoir proposer un prototype.

### Identification des principaux écarts et problèmes constaté, solutions possibles

Problèmes dans l’itération 1 :

* Problèmes dans les classes Pion et plateau qui ont pris plus de temps que prévue. Les solutions possibles sont de rajouter du temps donc devoir raccourcir le temps de programmations sur le code du jeu ou/et enlever des ressources dans l’itération pour rattraper le retard perdu.

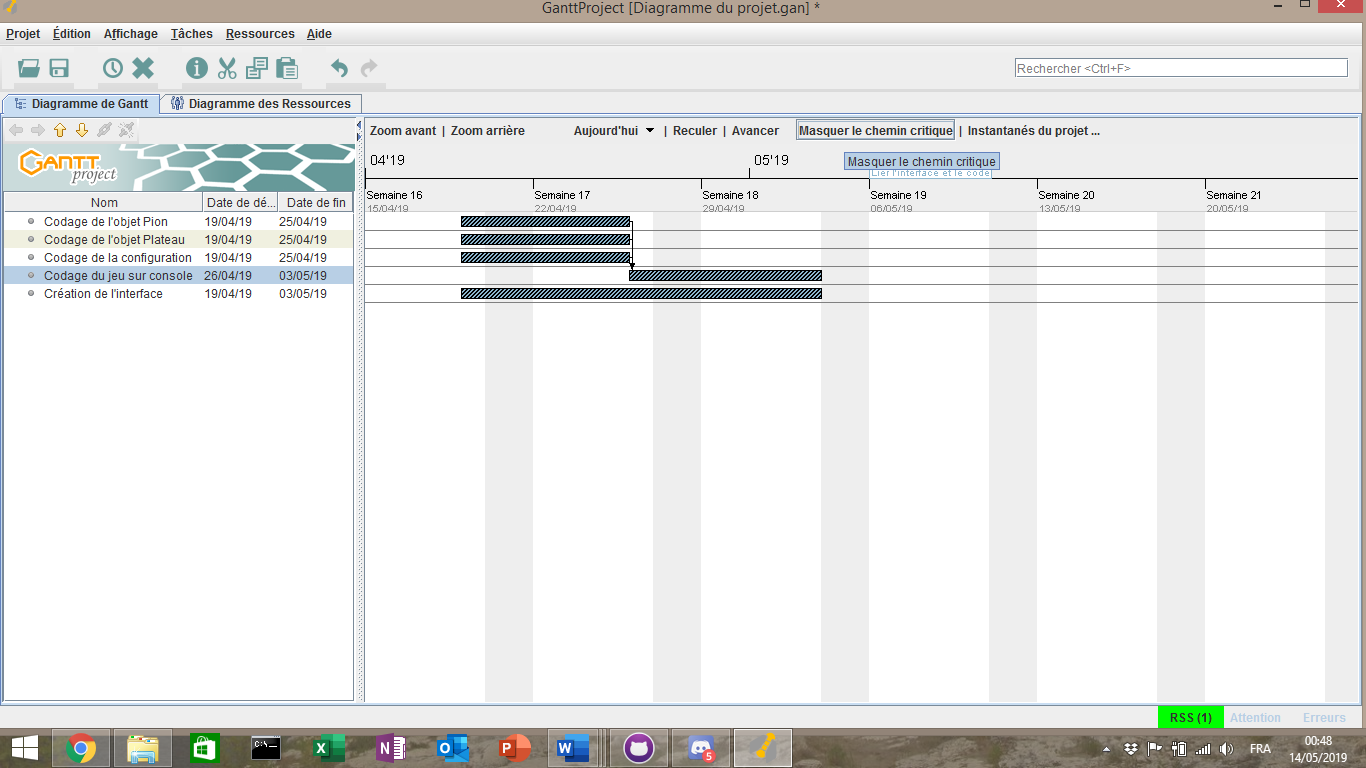
Problèmes dans l’itération 2 :

* Un problème entre l’interface et le code. Le code et l’interface étant pas programmer par la même personne, il y a eu des différences dans la façon de programmer des fonctions.

La solution possible était de faire encore plus communiquer les programmeurs entre eux pour éviter qui est de tel problème pour les futures itérations.

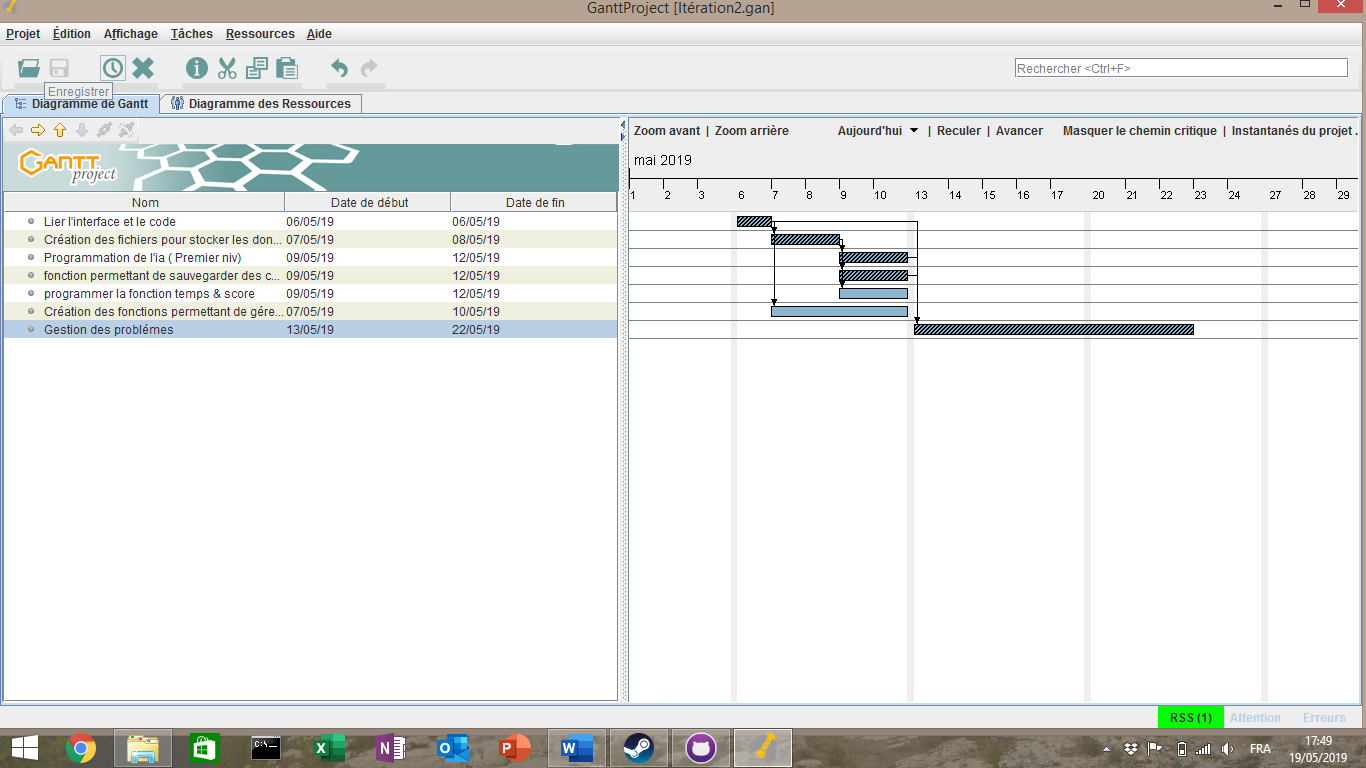
### Proposition de modification de la planification prévisionnelle pour tenir compte des corrections à apporter

Modification de la planification prévisionnelle de l’itération 1 :



Ajout de 1j sur l’objet plateau et Pion

Modification de la planification de l’itération 2 :



Nous avons pris 1 semaine de plus pour régler les problèmes qui sont apparu et pouvoir montrer un jeu fonctionnel.

### Comptes-rendus des réunions projets de la période

### Compte-rendu du comité de pilotage de la période

### Planification prévisionnelle révisée pour les périodes suivantes

# Bilan projet

# Glossaire

Annexes