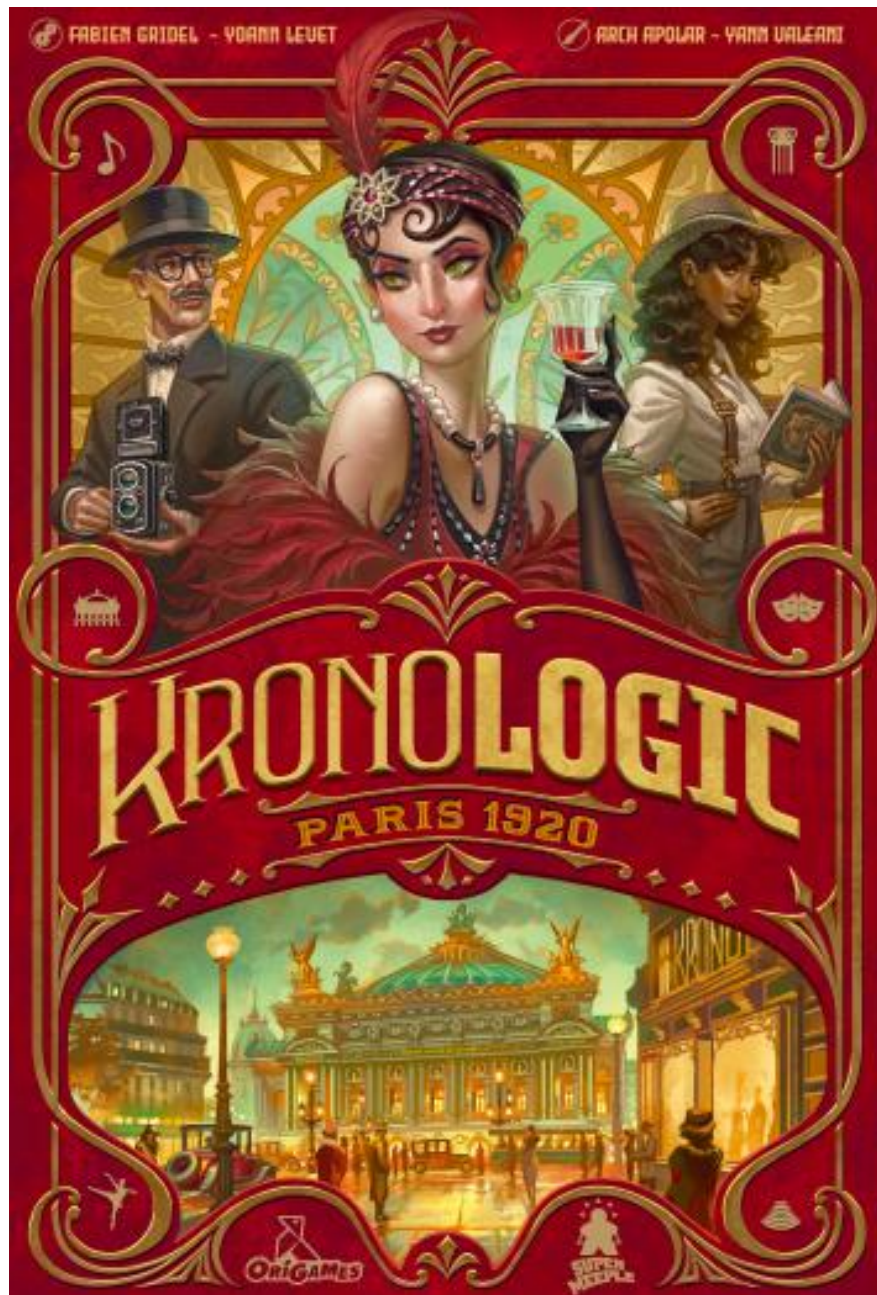


Projet tutoré - Premier document



Introduction

Le projet tutoré "Kronologic" vise à concevoir une intelligence artificielle (IA) capable d'accompagner ou de remplacer un joueur dans un jeu de déduction complexe. Le jeu de plateau Kronologic, créé par Fabien Gridel et Yoann Levet et édité par SuperMeeple en 2024, plonge les joueurs dans une enquête criminelle où ils doivent reconstituer les événements autour d'un meurtre en collectant des indices et en déduisant l'identité du coupable. Pour ce faire, le joueur doit analyser les lieux, les moments clés et les déplacements des personnages, tout en respectant des règles strictes de déduction.

Les objectifs de ce projet sont :

- ❖ Assimiler les mécanismes et règles du jeu et les intégrer dans une interface intuitive et accessible ;
- ❖ Développer une IA capable d'assister le joueur en réalisant des déductions logiques à partir des indices, et garantir que ses raisonnements sont corrects ;
- ❖ Concevoir une IA autonome qui puisse analyser les indices de manière optimale et identifier rapidement le coupable ;
- ❖ Élaborer un générateur de scénarios permettant de créer de nouvelles enquêtes avec des indices variés, en s'assurant que chaque problème proposé offre une solution unique, accessible et exigeant une combinaison astucieuse d'informations pour être résolu.

Explication du jeu

1. Objectif du Jeu :

Résoudre une enquête de meurtre en découvrant **qui est le tueur, où le meurtre a eu lieu**, et à quel moment précis il s'est produit. Chaque enquête se déroule sur une carte avec plusieurs lieux, personnages et moments dans le temps. Les joueurs posent des questions pour obtenir des informations publiques et privées, et utilisent leur logique pour assembler les indices.

2. Éléments de Jeu :

- ❖ **Mode solo** : Le joueur enquête seul et doit résoudre le mystère en utilisant ses questions avec précision.
- ❖ **Mode multijoueur** : Plusieurs joueurs enquêtent en tour par tour. Le premier à résoudre l'enquête gagne, mais tous doivent utiliser astucieusement leurs questions et écouter attentivement les indices révélés par les questions des autres joueurs.

3. Composition d'une Enquête :

- ❖ Une enquête correspond à une carte avec une **difficulté** déterminée.

- Poison Mondain -					
	Enquête 1	Enquête 2	Enquête 3	Enquête 4	Enquête 5
Difficulté	★	★★	★★	★★	★★★

Exemple : enquête Poison Mondain et ses 5 difficultés.

- ❖ Chaque enquête a une seule carte comportant plusieurs lieux (ex : 6).



Exemple : Carte de l'Opéra National de Paris.

- ❖ **Un lieu** correspondant au **lieu du meurtre**.
- ❖ **Plusieurs personnages** (ex : 6) sont présents sur la carte.
- ❖ **Un tueur** est à identifier.
- ❖ **Plusieurs temps** (ex : 6) définissent les différents moments du **meurtre**.
- ❖ **Un temps spécifique** est celui du **meurtre**.

4. Règles de Déplacement :

- ❖ **Chaque personnage** se trouve dans **un lieu** à **un moment donné**.
- ❖ **Les personnages se déplacent obligatoirement** à **chaque temps** vers **un lieu relié** par un **passage**.
- ❖ **Un personnage NE PEUT PAS** rester sur le **même lieu**.
- ❖ **Un personnage** peut retourner **plusieurs fois** dans le **même lieu** à **des temps différents**.
- ❖ **Chaque lieu** peut être **occupé par un nombre variable de personnages** (de 0 à 6).
- ❖ **Chaque lieu** est **relié à d'autres lieux** par **des passages**. **Chaque lieu** doit être **relié à un minimum de deux autres lieux**.



Exemple : Lieu des Escaliers avec les passages en blanc.

5. Le Rôle du Joueur :

Le joueur peut poser un nombre illimité de questions pour résoudre l'enquête. Chaque question porte sur un lieu spécifique et concerne soit le nombre de personnages présents à un moment donné, soit les déplacements d'un personnage spécifique.

6. Les Questions :

Les questions donnent **deux types d'informations** : **une information publique** que **tous les joueurs reçoivent** et **une information privée** que **seul le joueur qui a posé la question reçoit**.

❖ Question sur **un lieu** et **un temps** :

- **Information publique** : Donne le nombre de personnages présents à ce moment précis.
- **Information privée** : Donne également l'identité d'un personnage présent sur ce lieu et ce temps choisi.

❖ Question sur **un lieu** et **un personnage** :

- Donne le nombre de fois que le personnage est passé dans le lieu.
- Donne également le temps d'un des passages du personnage dans ce lieu.

Exemple :

1. Vous demandez : *"Combien de personnages étaient dans la salle S au temps T ?"*
→ **Réponse publique** : 2 personnages.
→ **Réponse privée** : Un des personnages présents était le Détective.
2. Vous demandez : *"Combien de fois est-ce que le personnage Z est passé dans la salle S ?"*
→ **Réponse publique** : 2 fois.
→ **Réponse privée** : Le personnage est passé au temps 3.

7. Cas du "Rejouer" :

Le "rejouer" est parfois présent à la place de l'**information privée** de la réponse.

- ❖ **En multijoueur** : cela permet de **poser une nouvelle question au lieu de mettre fin à son tour**, et la question révélant le "rejouer" **ne compte pas dans le total des questions posées**.
- ❖ **En solo** : ne possède pas d'utilité, il s'agit d'**une information privée manquée** ! En plus de cela, celle-ci **compte dans le total des questions posées**.




8. Déduction :

Chaque joueur peut à tout moment effectuer une déduction.

- ❖ Si cette dernière est juste, il **gagne la partie** et cette dernière **s'arrête**.
- ❖ S'il s'est trompé, il **ne peut plus faire de déduction** mais il peut quand même continuer de jouer.

9. Fin de l'Enquête :

Le nombre de questions posées détermine votre performance dans l'enquête :

 Loupe d'Or	1-7	1-9	1-8	1-9	1-12
 Loupe d'Argent	8-13	10-15	9-14	10-15	13-18
 Loupe de Bronze	14+	16+	15+	16+	19+

Exemple : les difficultés d'une enquête et le nombre de questions correspondant à la réussite de l'enquête.

Etude de l'existant

Pour développer une IA efficace dans le cadre du jeu Kronologic, nous devons combiner des outils techniques et des méthodologies de raisonnement adaptées aux exigences du jeu de déduction. En effet, le joueur doit interpréter des indices pour déduire des déplacements et identifier les présences ou absences de personnages dans différents lieux et à différents moments, tout en respectant des contraintes strictes de logique temporelle et spatiale. L'IA doit donc non seulement suivre les règles du jeu, mais aussi simuler un raisonnement déductif réaliste. Voici une analyse des outils et méthodes sélectionnés pour répondre à ces défis.

1. Outils de Programmation par Contraintes

Choco-Solver :

Choco-Solver est une bibliothèque Java spécialisée dans la programmation par contraintes, bien adaptée à la gestion des contraintes complexes de Kronologic. Les déplacements des personnages, organisés selon des séquences temporelles spécifiques, nécessitent d'éliminer les options impossibles en temps réel. La propagation de contraintes de Choco-Solver est essentielle pour modéliser ces restrictions, en assurant que, par exemple, un personnage vu dans une salle ne peut pas être simultanément ailleurs.

Comparaison avec Alternatives :

- ❖ **Google OR-Tools** : Bien que performante pour optimiser les parcours, cette bibliothèque se concentre davantage sur l'optimisation pure et non sur les contraintes logiques strictes essentielles dans Kronologic.
- ❖ **OptaPlanner** : Conçu pour la planification, OptaPlanner est moins performant dans la gestion des contraintes temporelles détaillées, cruciales pour garantir la cohérence des déplacements dans Kronologic.

Conclusion : Choco-Solver se distingue par sa capacité à gérer des règles de manière stricte tout en restant efficace, en faisant le choix idéal pour modéliser les contraintes du jeu et structurer les déductions de l'IA.

2. Méthodes de Raisonnement

Inférence Déductive et Propagation de Contraintes :

L'inférence déductive est essentielle pour que l'IA tire des conclusions précises à partir d'indices spécifiques, comme l'exclusion logique de la présence d'un personnage ailleurs une fois sa localisation confirmée. Associée à la propagation de contraintes, cette méthode réduit automatiquement les hypothèses en fonction des nouvelles informations, permettant à l'IA de gérer des situations complexes sans générer de fausses pistes.

Comparaison avec Alternatives :

- ❖ **Inférence Abductive** : L'abduction crée des hypothèses incertaines, ce qui risque de multiplier les fausses pistes, inadaptées pour le raisonnement rigoureux exigé par Kronologic.
- ❖ **Apprentissage Automatique** : Bien que puissant, l'apprentissage automatique n'est pas naturellement adapté aux règles strictes de Kronologic et exige des données d'entraînement volumineuses, alors que Kronologic requiert une logique déductive.

Conclusion : L'inférence déductive et la propagation de contraintes, combinées avec Choco-Solver, permettent un raisonnement fidèle aux règles de Kronologic, garantissant rigueur et efficacité sans créer de pistes incertaines.

3. Conclusion Globale

Les choix finaux sont :

- ❖ **Choco-Solver** pour la gestion des contraintes complexes et l'application des règles du jeu,
- ❖ Les **méthodes d'inférence déductive et de propagation de contraintes** pour assurer un raisonnement logique et structuré.

Ensemble, ils offrent une solution efficace et fiable pour que l'IA résolve les enquêtes de manière réaliste, optimisant la précision et la conformité aux règles du jeu Kronologic.

Cahier des charges

1. Objectifs Fonctionnels

Reproduction des Règles du Jeu

L'objectif principal est de reproduire fidèlement les règles du jeu Kronologic pour pouvoir réaliser une énigme, notamment en ce qui concerne :

- ❖ **Gestion des indices** : Le jeu doit permettre l'affichage et la collecte des indices disponibles à chaque étape, tout en préservant leur valeur stratégique.
- ❖ **Déplacements des personnages** : Chaque personnage dans le jeu est lié à une localisation spécifique, ainsi qu'à des horaires et itinéraires précis.
- ❖ **Conditions de victoire** : La victoire est déterminée par la découverte du coupable. Le joueur doit interpréter les indices pour innocenter des personnages ou pour désigner le véritable coupable.

Assistance au Joueur (Dédution)

Une IA doit être réalisée afin d'assister le joueur dans son processus de déduction, en fournissant des suggestions basées sur les indices collectés. Cette assistance se traduira par :

- ❖ **Interprétation des indices** : L'IA sera capable d'analyser les indices disponibles pour suggérer des actions au joueur, comme interroger un personnage particulier ou se rendre à un endroit précis pour trouver un nouvel indice.
- ❖ **Innocenter et déduire** : En fonction des indices disponibles, l'IA devra être en mesure de proposer des hypothèses pour écarter certains personnages ou, au contraire, les suspecter davantage. Cela aidera le joueur à affiner son raisonnement pour progresser efficacement vers la solution finale.

❖ Réalisation de l'IA

On va tester deux possibilités d'IA et voir à la fin ce qui est le plus rapide :

- **Développement manuel** : Une première version de l'IA sera implémentée manuellement pour tester la logique de déduction et assurer la fiabilité des algorithmes.
- **Implémentation avec un solveur (Choco-Solver)** : Pour gérer des scénarios complexes, le projet intégrera Choco-Solver, une bibliothèque de programmation par contraintes, qui permettra à l'IA de résoudre des contraintes logiques avancées. L'IA

pourra ainsi analyser les indices en fonction de plusieurs variables (lieu, moment, personnages) pour obtenir une solution optimisée.

Joueur Autonome (Optionnel)

Un objectif optionnel que l'on fera si on a le temps, consiste à développer une version autonome de l'IA qui pourrait jouer seule et résoudre une enquête sans intervention humaine. Elle sera différente de l'IA précédente mais on verra qu'on pourra s'en servir. Cette IA autonome :

- ❖ **Collecte d'indices pertinents** : L'IA se focalise sur les indices les plus significatifs et exclura automatiquement ceux de moindre importance.
- ❖ **Résolution optimisée** : L'IA pourra déterminer le coupable de manière optimisée, en limitant au maximum le nombre d'étapes nécessaires pour aboutir à une conclusion.

Génération Automatique de Nouvelles Enquêtes (Optionnel)

Pour enrichir le gameplay, le système pourra inclure une fonction de génération d'enquêtes qui crée des scénarios variés en termes de difficulté et de complexité.

- ❖ **Création de scénarios possibles** : La génération de scénarios doit s'assurer qu'il est possible de résoudre chaque enquête à partir des indices fournis, tout en préservant un niveau de difficulté progressif. Notre seconde IA devra donc tester cette énigme pour vérifier si elle est réalisable.

2. Contraintes et Limitations

Optimisation du Temps de Calcul

L'IA doit être optimisée pour offrir une réponse rapide aux joueurs, même dans les scénarios les plus complexes. Ceci implique :

- ❖ **Réduction du temps de déduction** : L'IA doit résoudre chaque étape de déduction rapidement, sans ralentir l'expérience de jeu.
- ❖ **Exclusion d'options impossibles** : Pour améliorer la rapidité, le moteur devra exclure dès le début les scénarios non viables, réduisant ainsi le nombre de calculs nécessaires.

Traitement d'Indices Partiels et Multiples Sources d'Informations

L'IA sera confrontée à des indices fragmentaires provenant de diverses sources (lieux, personnages, objets) et devra être capable de croiser ces informations pour obtenir des déductions fiables.

- ❖ **Gestion des données incomplètes** : L'IA doit pouvoir interpréter des informations incomplètes et anticiper les conclusions possibles en fonction des éléments obtenus.
- ❖ **Croisement d'informations** : En utilisant la logique, l'IA croisera les indices pour isoler des relations et des hypothèses valides, tout en éliminant les pistes qui ne concordent pas.

Respect des Règles du Jeu Kronologic

L'IA doit se conformer rigoureusement aux règles du jeu Kronologic, sans recourir à des actions ou des stratégies non prévues par les règles.

- ❖ **Suivi strict des règles** : Chaque décision et chaque action de l'IA doivent respecter les limitations du jeu en termes de déplacement, de collecte d'indices, et de vérification des hypothèses.

Densité de Personnages en Fonction des Salles

Pour assurer une expérience de jeu réaliste et cohérente, l'algorithme de création d'énigmes doit gérer la densité des personnages présents dans chaque salle. Cette contrainte implique de limiter le nombre de personnages dans un même espace en fonction de la complexité de l'énigme.

- ❖ **Priorité aux règles de gameplay** : Lorsqu'un personnage tente de rejoindre une salle possédant le nombre de personnage maximal, l'algorithme devra soit rediriger ce personnage vers un autre emplacement pertinent. Ces choix devront être faits sans perturber l'avancement logique de l'enquête.
- ❖ **Test avec l'IA joueuse** : Afin de vérifier la bonne jouabilité, on utilisera l'IA joueuse pour vérifier que l'enquête est bien solvable.

3. Outils et Technologies

- ❖ **Langage de programmation** -> Java
- ❖ **Bibliothèque de programmation par contraintes** -> Choco-Solver : Solveur qui réalise des tâches, telles que la planification, la recherche de solutions optimales, et la résolution de puzzles logiques. Cette bibliothèque marche de la façon suivante :

- **Modélisation du problème** : Choco-Solver permet de structurer les relations entre personnages, indices et lieux en gérant les contraintes complexes. Vous définissez les variables, leurs domaines de valeurs et les contraintes qui régissent les combinaisons de valeurs autorisées.
- **Résolution** : Choco Solver affecte des valeurs aux variables de manière à ce que toutes les contraintes soient satisfaites.
- **Réduction des espaces de recherche (optimisation)** : En excluant les options non viables, Choco-Solver accélérera les calculs et optimisera l'expérience de jeu.

Ébauche partie technique

1. Technologies utilisées

Pour implémenter les fonctions nécessaires à l'Intelligence Artificielle du jeu *Kronologic*, nous avons sélectionné une bibliothèque et le langage de programmation Java. Ils ont pour but de respecter les contraintes de complexité du jeu et d'assurer la fluidité du raisonnement logique de l'IA. Nos choix sont détaillés ci-dessous :

2. Bibliothèques et Langages :

- ❖ **Java** : Langage principal de développement. Sa compatibilité avec Choco-Solver et sa modularité permet de réaliser des projets complexes nécessitant une gestion fine des contraintes.
- ❖ **Choco-Solver** : Une bibliothèque de programmation par contraintes spécialisée en Java. Elle est essentielle pour gérer les contraintes de déplacements, de temps et d'interactions entre les personnages. Ainsi, elle permet d'éliminer les options impossibles en fonction des indices fournis, optimisant donc le raisonnement.

3. Modélisation des données

L'architecture du projet repose sur plusieurs classes de données qui représentent les éléments essentiels du jeu. Ces classes facilitent le stockage des informations nécessaires au raisonnement de l'IA.

- ❖ **Personnages** : Chaque personnage possède un **nom** (*String*). Ils possèdent aussi **des attributs pour suivre ses déplacements** (*List<Int>* : liste des lieux visités par le personnage dans l'ordre chronologique) et **les indices associés** (*List<Indice>*), permettant de vérifier sa présence dans une salle à un moment donné.
- ❖ **Indices** : Les indices sont structurés avec les attributs :
 - de **Lieu** (*Lieu*),
 - de **Personnage** (*Personnage*) et
 - de **Temps** (*Int*) et sont classés en deux catégories :
 - **publics** (accessibles à tous les joueurs) et
 - **privés** (uniquement visibles pour l'IA ou le joueur qui a posé la question).
- ❖ **Lieux** : Chaque lieu représente une salle où les personnages peuvent se déplacer. Les lieux possèdent trois attributs :
 - Un **Nom** (*String*),

- Un **Id** (*Int*) permettant d'identifier les lieux et
- Une **Liste des lieux adjacents** (*List<Int>*) afin de vérifier les déplacements des personnages.

4. Raisonnement automatique et propagation de contraintes

Le raisonnement de l'IA repose sur une combinaison de techniques d'inférence et de propagation de contraintes, permettant à l'IA de formuler des déductions précises.

- ❖ **Raisonnement automatique** : L'IA utilise l'inférence déductive pour relier les indices et en tirer des conclusions logiques. Par exemple, si un personnage est observé dans une salle à un moment donné, l'IA peut immédiatement exclure sa présence dans d'autres salles au même moment.
- ❖ **Propagation de contraintes avec Choco-Solver** : Chaque nouvel indice déclenche une propagation de contraintes, ce qui élimine automatiquement les hypothèses impossibles et réduit l'espace de recherche. Cela rend le raisonnement plus efficace et précis, sans explorer les fausses pistes.

5. Planification des actions

La planification des actions de l'IA joueuse se divise en deux parties : la collecte d'indices et l'utilisation de la première IA aidant le joueur. Ces deux étapes sont toutes deux cruciales pour résoudre les enquêtes de manière optimale.

- ❖ **Collecte d'indices** : L'IA choisit les indices à prioriser en fonction de leur pertinence pour la résolution de l'enquête. Choco-solver pourrait ici être utilisé pour optimiser le choix des actions à chaque étape de l'enquête.
- ❖ **Utilisation de l'IA aide** : En utilisant notre première IA, nous allons pouvoir l'utiliser dans la construction de la seconde. Son rôle va être de choisir la meilleure question à poser avec les données fournies. La différence va résider dans le fait que cette nouvelle IA va voir plusieurs coups dans le futur en réalisant des hypothèses avec l'IA.

6. Interface utilisateur

Une interface utilisateur simple sera développée pour permettre aux joueurs de suivre l'enquête visuellement. Elle affichera :

- ❖ X **Cartes** avec ses **Lieux** (X étant le nombre de temps).
- ❖ Les indices recueillis.
- ❖ Les informations récupérées sont affichées sur les **Cartes** (*pions de personnages, nombre de personnages*).



Exemple de la grille du jeu

Cette interface sera essentielle pour tester les fonctionnalités du jeu et évaluer l'efficacité du raisonnement de l'IA en temps réel. Elle sera également conçue pour rendre les actions de l'IA et les informations visibles facilement accessibles, afin que le joueur puisse se concentrer sur la logique et les déductions nécessaires pour résoudre l'enquête.

Conclusion

Ce document constitue la première partie de notre projet et nous permet de définir clairement les objectifs et les différentes étapes. En effet, il ne s'agit pas seulement de reproduire le jeu sous forme informatique, car cela constitue uniquement la première partie du projet.

Le cœur de celui-ci réside dans la partie algorithmique et la manière de proposer des raisonnements automatiques intelligents et complexes. Cela se traduit par :

- ❖ Le développement d'une IA qui va aider le joueur à gagner la partie.
- ❖ Le développement d'une IA capable de jouer de manière autonome en résolvant l'énigme le plus rapidement possible.

Des méthodes de raisonnement telles que l'inférence déductive et la propagation de contraintes pourront nous aider à implémenter cela. De plus, nous pourrions nous appuyer sur une bibliothèque Java spécialisée dans la programmation par contraintes, Choco-Solver.

Cette dernière est en effet bien adaptée à la gestion des contraintes complexes imposées par Kronologic.

Le cahier des charges servira de base pour structurer et prioriser les fonctionnalités essentielles de l'IA, en s'assurant que chaque aspect du projet respecte les objectifs initiaux et les contraintes spécifiques du jeu Kronologic. L'ébauche de la partie technique, quant à elle, nous permettra d'orienter les choix de conception et de modélisation, en définissant les structures de données, les algorithmes et l'architecture générale du programme.