CPSC 2720 SPRING 2024

Cipher Game

Group V

Authors: Miller Fourie, Justin Patrick, Kaylan Brum

Due Date: 2024-04-05

TABLE OF CONTENTS

INTRODUCTION

PROJECT MANAGEMENT

TEAM ROLES

Team Member	Design - Draft	Design – Final	Implementation - Basic	Implementation - Final
Miller	Phase Lead	Design Lead	Implementation Lead	Implementation Lead
Justin	Design Lead	Phase Lead	Implementation Lead	Implementation Lead
Kaylan	QA Lead	QA Lead	General Oversight	General Oversight

TEAM ROLE RESPONSIBILITIES

Role	Responsibilities
Phase Lead	- Responsible for overseeing the specific phase of the project, ensuring its successful completion within the allocated time frame. - Communicates their vision effectively and spearheads the phase of the project, ensuring that other roles can align with the overarching goals and objectives. - Coordinates tasks and activities within the phase, ensuring that team members are aligned with project goals and objectives. - Communicates with team members to provide updates on the progress of the phase and address any issues or concerns.
Design Lead	- Takes the lead in ensuring the design of the Phase Lead meets the project requirements and objectives. - Reviews initial drafts of phase lead and provides feedback and edits. - Ensures that design decisions align with project goals, technical requirements and best practices.
Implementation Lead	- Leads the implementation phase of the project, focusing on translating design concepts into functional software solutions. Will typically focus on one specific core implementation of the project. - Manages the development process of the core implementation and oversees the coding and testing of those software components. - Ensures that the implemented solutions meet quality standards, are delivered on time, and align with project requirements.
QA Lead	- Takes the lead in quality assurance activities throughout the phase lifecycle, ensuring that the project meets quality standards and objectives. - Collaborates with team members to improve processes and practices related to quality assurance and testing.
General Oversight	- Provides general oversight and support across the entire phase, ensuring that tasks are completed efficiently and effectively. - Collaborates with team members to address any challenges or issues that arise during the phase. - Assists with coordination, communication, and problem-solving to ensure project success.

RISK MANAGEMENT

#	*Risk Management Process*
1	**Identification:** **Objective:** Identify potential risks that could impact the project. **Description:** The initial phase involves analyzing internal and external factors that may contribute to project delays or adverse outcomes. This comprehensive evaluation ensures a broad spectrum of potential risks are considered.
2	**Assessment:** **Objective:** Evaluate and categorize identified risks. **Description:** Following risk identification, each risk is assessed based on its likelihood of occurrence and the potential impact on project objectives and requirements. Risks are systematically categorized into levels: Medium, High, and Severe, to prioritize management efforts.
3	**Mitigation:** **Objective:** Develop strategies to reduce risk likelihood and impact. **Description:** Post-assessment, the focus shifts to creating mitigation strategies. These include preventive measures and the development of contingency plans to manage and lessen the potential adverse effects of the risks.
4	**Monitoring and Control:** **Objective:** Continuously track and manage risks throughout the project lifecycle. **Description:** This phase involves the ongoing monitoring of risks to identify any changes in the likelihood of impact. It includes activating contingency plans as necessary and addressing new risks according to the established framework.

Integral to each step of the *Risk Management Process* is the overarching necessity of thorough Communication and Documentation. These elements ensured that the risk management practices are transparent, aligned, and effectively implemented across all project phases.

Communication: **Objective:** Ensure consistent and effective dissemination of risk-related information. **Description:** Communication strategies include issuing regular updates on potential risks and realized issues through established channels, such as digital platforms like Microsoft Teams. This ensures all project members are informed and aligned with the risk management strategy.
Documentation: **Objective:** Record and maintain records of risk management activities. **Description:** Documentation encompasses the recording of identified risks, assessments, mitigation strategies, and monitoring activities. The documentation can be maintained through various channels, including digital collaboration tools (such as Microsoft Teams) for potential risks, and formal issue reports on the GitLab project page for fully realized issues.

DEVELOPMENT PROCESS

CODE REVIEW PROCESS

#	*Code Review Process*
1	**Individual Commitments:** Each team member commits their changes to their respective branches, adhering to a standard fo clear and concise commit messages. These messages provide transparency and facilitate efficient collaboration by allowing other members to understand the purpose and scope of the changes made.
2	**Merge Request and Review:** When a new feature of code change is ready for integration into the main branch, the responsible team member submits a merge request. The request prompts other team members to review the proposed changes thoroughly. The review process ensures that potential conflicts or issues are identified early and addressed promptly. If conflicts arise, the team collaborates to resolve them, employing manual and non-destructive merge techniques to preserve the integrity of the codebase.
3	**Resolution and Integration:** Upon successful resolution of conflicts and completion of the review process, the merge request is approved and merged into the main branch. Once merged, the merge request is closed providing a clear indication that the proposed changes have been successfully incorporated into the main branch and are ready for further development.

COMMUNICATION TOOLS

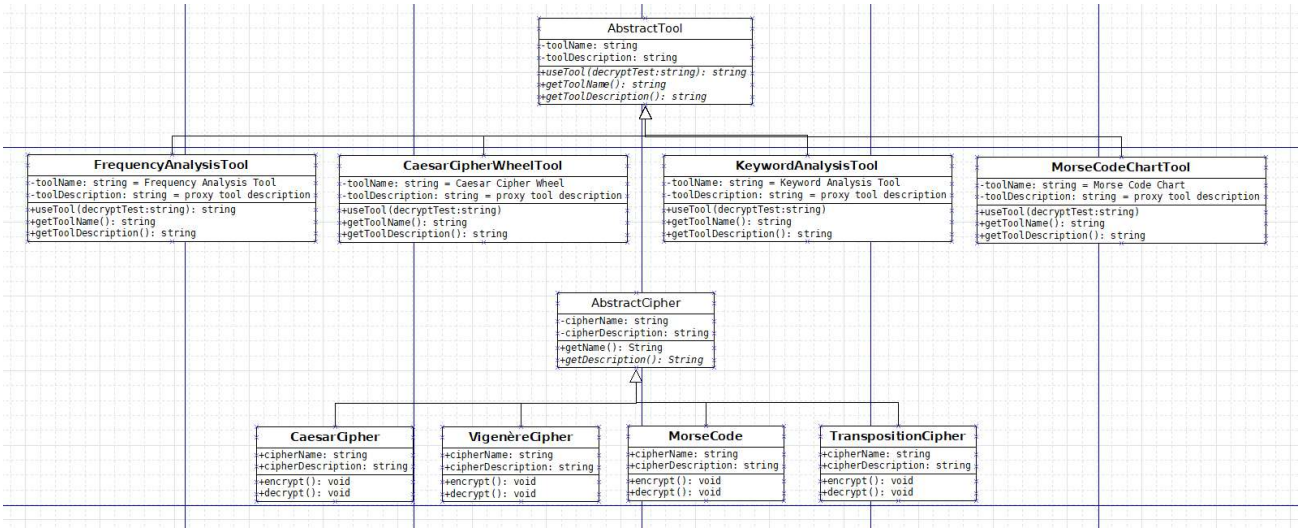
The project used Microsoft Team's Group Channel for communication and collaboration purposes. GitLab served as a designated issue tracker for comprehensive project management and tracking of developing tasks.

CHANGE MANAGEMENT

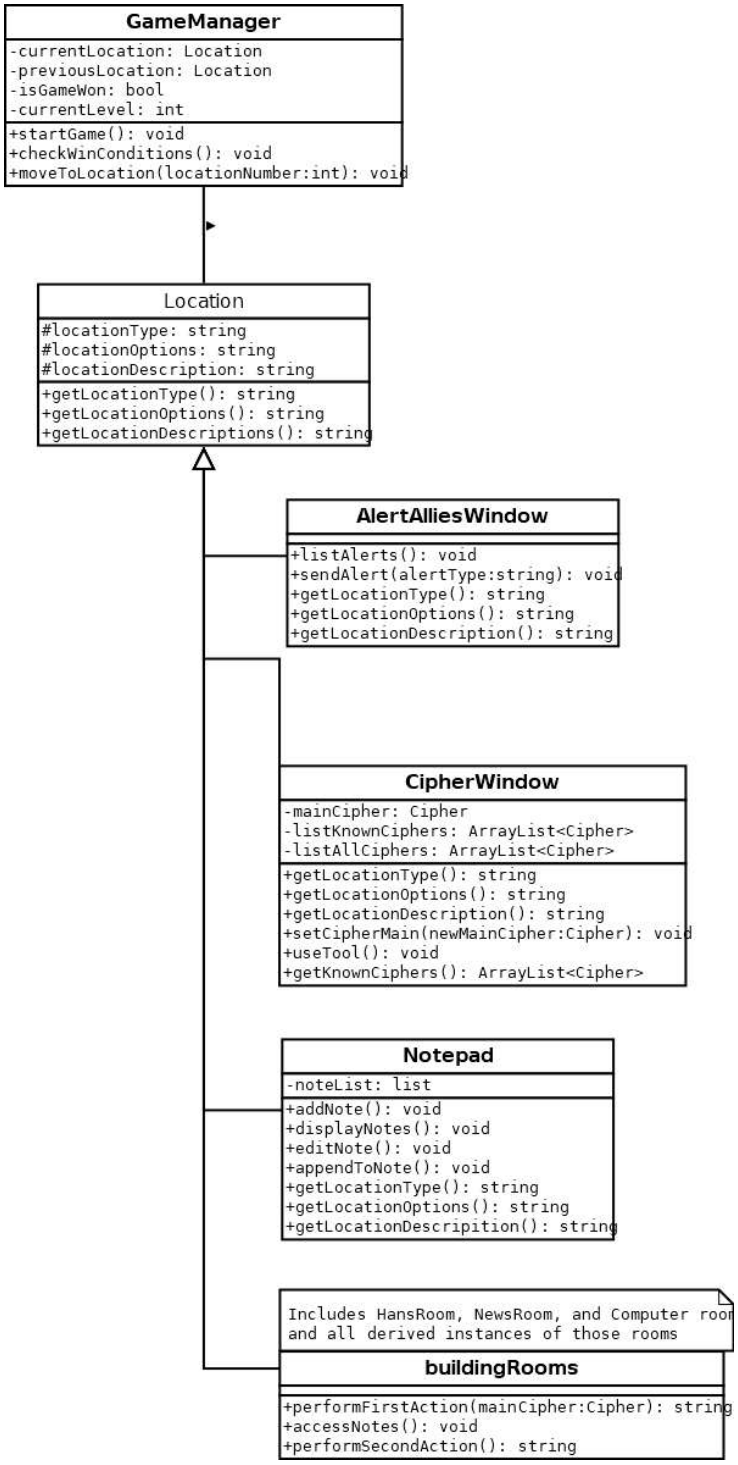
#	*Code Review Process*
1	**Preparation and Planning:** - Assess the scope and impact of the proposed changes. - Development a comprehensive understanding of the effect that this change will have on other aspects of the code.
2	**Communication:** - Communicate with team members about the changes and address any questions they have on how this will affect the overall project. - Maintain open and ongoing communication to manage any issues and feedback group members may have. Continue onto implementation when all issues and feedback and been addressed.
3	**Implementation:** - Execute the change, ensuring alignment with overall project goals and objectives. - Utilize project management and change management tools to monitor progress and address any issues that arise.
4	**Monitoring and Review** - Continuously monitor the change process, assessing the effectiveness of the change. - Make necessary adjustments to ensure the change achieves the desired outcome.

SOFTWARE DESIGN

DESIGN – CLASS DIAGRAMS



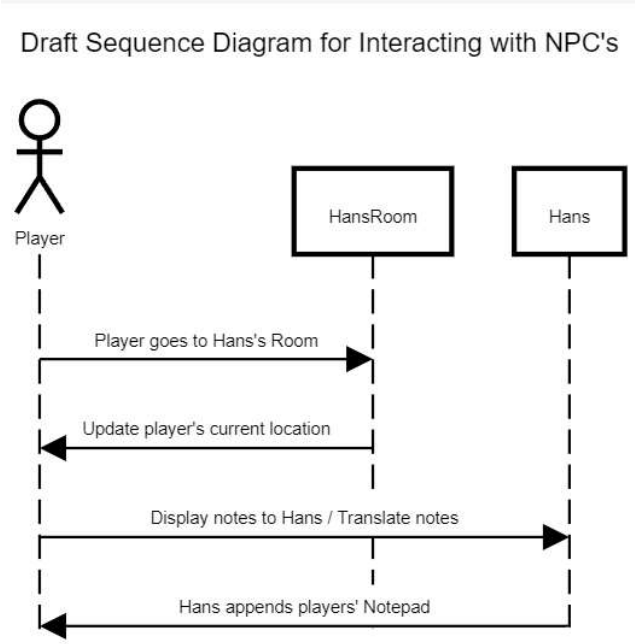
Design phase Class Diagram for Abstract Tool class and all concrete derived classes.



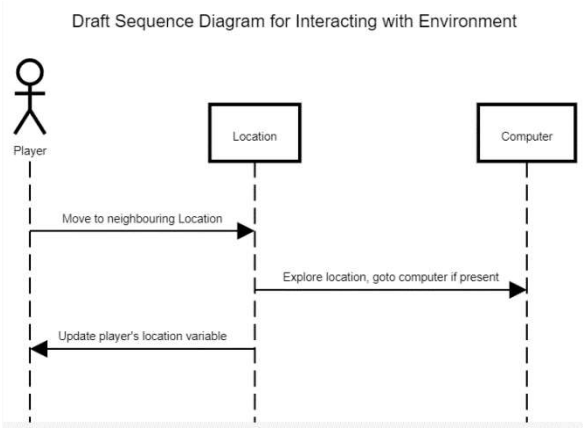
Design phase Class Diagram for GameManager and Abstract Location class with all concrete derived classes.

DESIGN – SEQUENCE DIAGRAMS

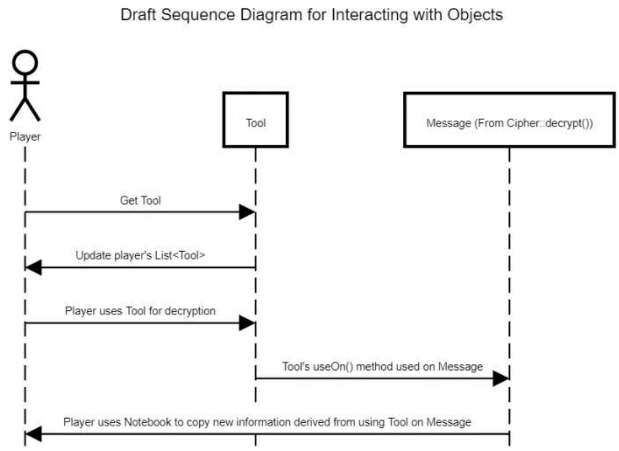
- Sequence diagrams created during the design phase:



NPC interaction draft sequence diagram



Environment Interaction Draft Sequence Diagram.



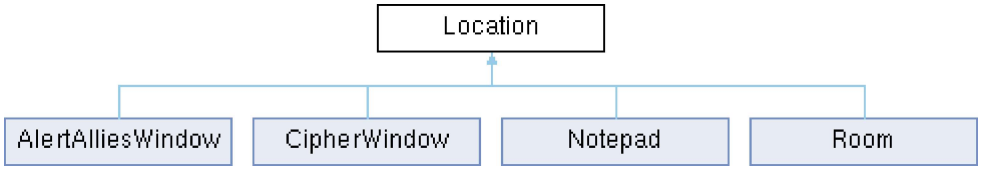
Object Interaction Draft Sequence Diagram.

CLASS DESCRIPTIONS

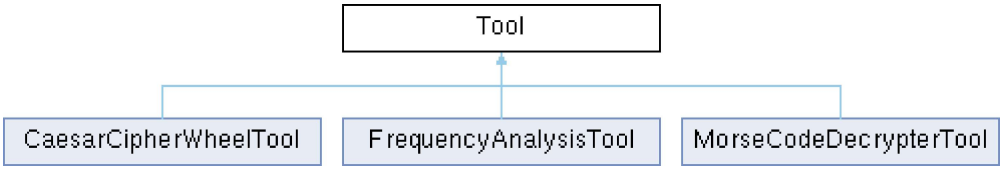
- **AlertAlliesWindow**: concrete child of Location, implements functionality for the AlertAlliesWindow screen.
- **CaesarCipherWheelTool**: concrete child of Tool, implements functionality for the CaesarCipherWheel tool.
- **Cipher**: Object which holds information for Ciphers such as its text, untranslated test etc.
- **CipherManager**: Manager for all Cipher objects within GameManager
- **CipherWindow**: concrete child of Location, implements functionality for the CipherWindow screen.
- **Event**: Object which holds and handles information regarding Location events, such as modifying, adding, and firing events.
- **Flag**: Basic object which encapsulates a replicable Boolean flag design which is used by GameManager and Event.
- **FrequencyAnalysisTool**: concrete child of Tool, implements functionality for the FrequencyAnalysisTool tool.
- **GameManager**: Main object which stores and manages game data and controls the game loop.
- **Location**: Abstract class which contains base information and virtual methods for creating an in-game location.
- **LocationFactory**: Factory which allows for the creation of specific derived types of Location, used in setting up the game levels.
- **MorseCodeDecrypterTool**: concrete child of Tool, implements functionality for the MorseCodeDecrypterTool tool.
- **Notepad**: concrete child of Location, implements functionality for the Notepad program.
- **Room**: concrete child of Location, implements functionality for a generic Room object.
- **Tool**: Abstract class which contains base information and virtual methods for creating an in-game Tool.
- **ToolManager**: Initialises and manages tools for the game.

APPENDICES

APPENDIX A: FIGURES AND TABLES



Abstract Location class and all concrete derived classes as created by docs



Abstract Tool class and all concrete derived classes as created by docs