Architecture

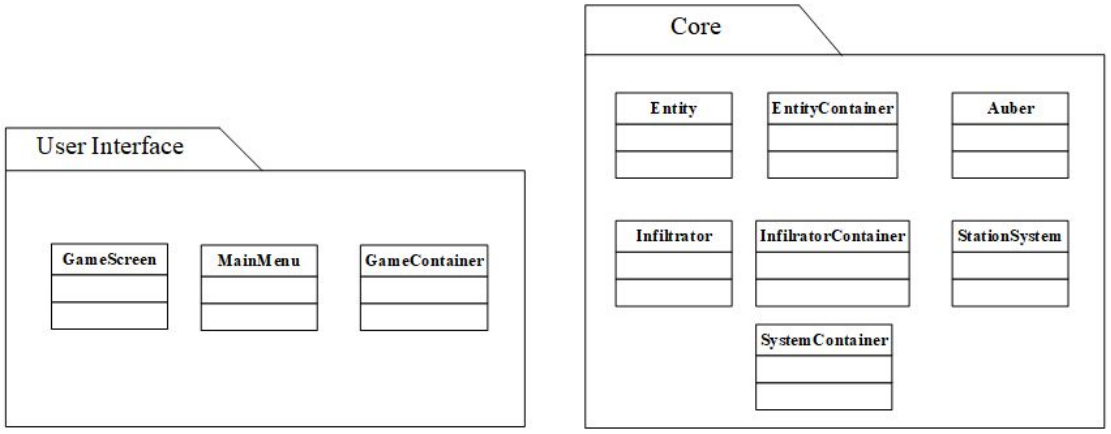
Team: No

Team Members:

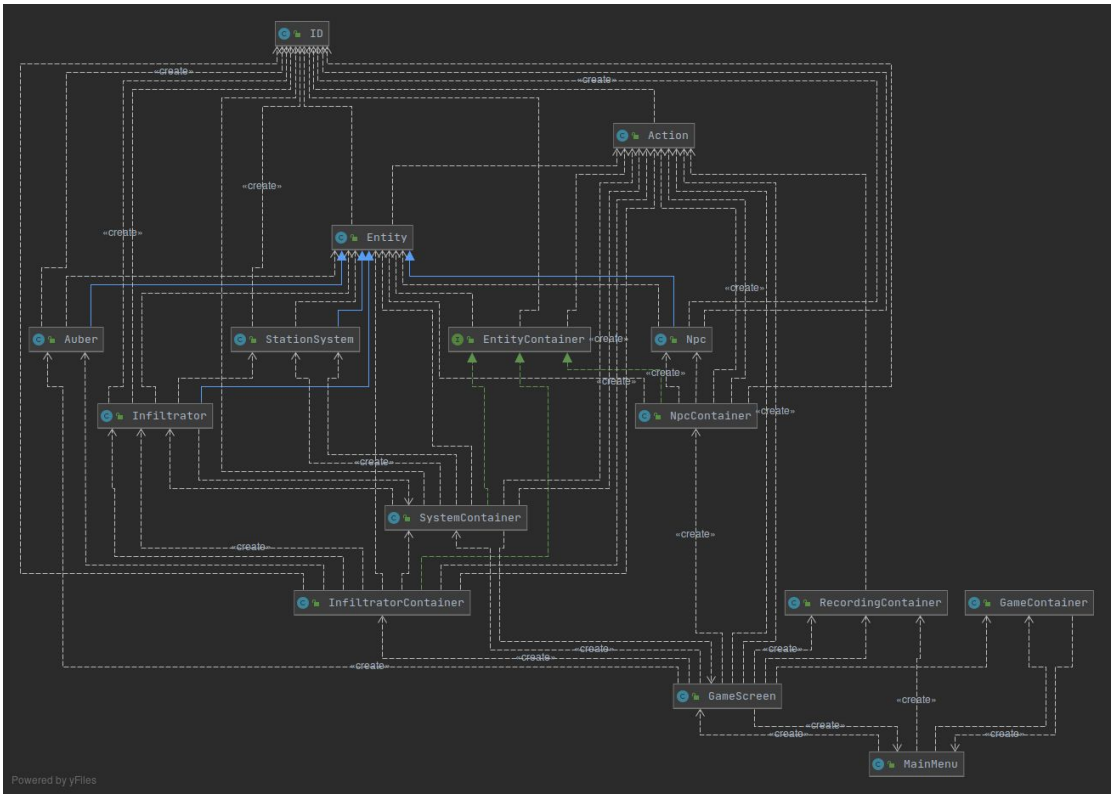
- Jack Burman
- Sam Churchill
- C Lloyd
- Sam Ralph
- Rebecca Wardle
- Jiacheng Wu

a(1) Architecture of the team's software

The abstract architecture is as below. In the abstract architecture, the system is divided into two parts: core and user interface. The user interface is for user input. The core is designed for backend system logic.



The concrete architecture is as below. The concrete architecture clearly describes the specific implementation of the system. The properties and methods contained in each class, and how they are linked together.



a(2) Language used to describe the architecture

In the process of system architecture design, UML is used. UML is a unified software modeling language. It has a wide range of modeling capabilities. UML symbols have good semantics and will not cause ambiguity. The visual model based on UML makes the system structure intuitive and easy to understand. When using UML to build the model of software system, it is not only conducive to the communication between system developers and system users, but also conducive to system maintenance. The model is the blueprint of the system, which can complement the planning of developers. The model can help developers plan the system to be built. UML uses a group of graphical symbols to describe the software model. These graphical symbols have the characteristics of simple, intuitive and normative, which is easy for developers to learn and master. The software model described by UML can be understood and read intuitively. Because of its standardization, it can ensure the accuracy and consistency of the model. We use class diagram to describe the architecture. The class diagram being the major component of the object centre modeling, it exhibits the static structure of the model, which indicates that it displays the structure of class and their connection and relationship with other classes.

a(3) Tool used to create the architecture

Intellij UML editor is used to create the architecture**.** It is a feature in Intellij that can automatically build class structure and relationship diagrams. It can help people quickly understand the code hierarchy and system architecture. This generation feature that it is convenient for it and used to process, analyze and exchange the complex information with their code. The Intellij UML editor is convenient for users to understand and mainly for making better visualization of relationships between code.

b(1)Systematic justification for the abstract and the concrete architecture

In the abstract architecture, it is divided into two parts: the interaction with players and the back-end core part. The back-end core includes: Entity class, EntityContainer class, Auber class, Infiltrator class, Infiltrator container, StationSystem class and SystemContainer class. As can be seen from the abstract architecture, there are three classes that interact with the user: the GameScreen class, MainMenu class and GameContainer class. The GameScreen class is shown to the player interface. The MainMenu class is provided for users to set the game mode. Lastly the GameContainer class is used to start and load the game which also sets the "screen" to the MainMenu class. These three classes are designed to be used by players.

In the back-end core part. Entity class provides some default methods for Auber, Infiltrator's and Npc's (through child/parent hierarchy). The Auber class is responsible for the new player Auber, and provides input from the user, health healing, teleporting and taking damage from systems. When Auber health is less than or equal zero, the game is over and the player loses.

The Infiltrator class handles moving of individual infiltrators, and damaging of systems. The InfiltratorContainer class creates different infiltrators with different special abilities and updates the number of infiltrators captured in real time to determine whether the game is over and whether Auber wins. The StationSystem class is responsible for system special abilities (like teleportation) and SystemContainer class can update the number of damaged systems and keeps track of "active" systems, for when the game is lost.

b(2)concrete architecture related to the requirements

Requirements name	Description	concrete architecture name
UR_PLAYER_CONTROL	The player must be able to control something that moves around a space station	class GameScreen has method: keyDown() for moving the Auber on key presses
UR_TELEPORTATION	Rooms can have teleportation pads from which Auber (but not infiltrators) can teleport to any other teleportation pad in the station	class Auber has methods: teleport(),Checks if the auber can teleport and if so performs it and adds cooldown
UR_INFILTRATOR_ABILITIES	There must be at least 3 distinct special abilities within the group of infiltrators	class Hallucinogenic and class Invisible have method: Invisible() and Hallucinogenic (). The special ability also have faster speed to change the MAX_VELOVITY and VELOCITY_CHANGE

UR_HEAL	Auber can teleport to the close healing system to heal	class Auber has method: healFromSystem(),Heals the auber if they are close to a healing system
UR_GAME_OVER	If Auber reaches 0 health the game ends	Class Auber has method: damageForSystem() to judge whether Auber health<=0. If Auber health <=0 game will end in Class GameScreen method render()
UR_ARREST	Auber should be able to arrest and move an infiltrator to the brig	class InfiltratorContainer's method: collisionCheck() provide the function to check whether infiltrator is inside the collision box, and checkCaptured() provide the function to delete infiltrators
UR_INFRILTRATOR_DAMAGE	The infiltrators should be able to damage systems of the space station	class StationSystem method: applyDamage() applies damage to the closest system.
UR_WIN	The game will be won when 8 infiltrators have been arrested	class Auber method: hasPlayerWon() and real-time update the arrested infiltrators. When there is no current Infiltrators will end the game.
UR_LOSS	The game will be lost when 15 or more systems are destroyed	class GameMechanics's method: getDestroyedSystemNum() and real-time update the damaged systems. When the number becomes 15, endGame() will end the game.
UR_MODES	There must be "single player mode" and a "demo mode"	class MainMenu method to provide mode selector "Start Game" for "single player mode" and "Playback Recording" for "demo mode".
UR_ART_STYLE	The game should be a 2D top-down game	class GameScreen displays the game in 2D.
UR_GAME_REPLAY	Record the movements of the game being played so that it can be played back	Through the Action class, this class is used to record the movements of the game and play back by class RecordingContainer