Requirements and Analysis Document for Risky Space

Contents

Version:	2.0
Date:	2012-05-15
Author:	Daniel Augurell, Jonatan Rapp, Sebastian Odbjer, Alexander Hederstaf

This version overrides all previous versions

1 Introduction

This section gives a brief overview of the project

1.1 Purpose of application

The application is a game set in a space enviroment. The game is played in a top down view similar to that of a board game. The goal of the game varies with game modes but it is usually focused on defeating your opponent(s). To accomplish the goals the player can build bases on planets throughout the game's galaxy and use the bases to build spaceships used to gather victory points or killing your enemy.

1.2 General characteristics of application

The application will be a desktop, standalone (with later possibility to be networked), two-player (can be extended to a four-player) application with a graphical user interface for the Windows/Mac/Linux platforms.

The application will be turn based. The actual player must explicitly end his or her turn. The next player is chosen by the application from a preset ordering. The ordering is generated randomly by the application at start of the round. There's no time constraints for a round. The application will end according to the rules or possibly be canceled. If the game is canceled the possibility there shall be a possibility to save, else the player who canceled loses. The GUI will look like a big board made of tiles with different environment e.g. planets and space.

1.3 Scope of application

The game is played by two to four players locally on the same computer. There is no AI or networking multiplayer. The game can not be played alone (unless one player play both teams). If a game is cancelled the player whose turn it was loses. You can save the game before cancelling to continue from that turn another time.

1.4 Objectives and success criteria of the project

First version:
1. It should be possible for two players playing through the same computer taking turns to finish a game.
2. There should be annihilation game mode with the victory goal of defeating the other player's home planet.
3. Construction of 4 different kinds of ships should be possible (Colonizer, Scout, Hunter, Destroyer).

4. You should be able to build buildings on your occupied planets.

Later version: (In addition to first version)
5. You should be able to host a network multiplayer game in which other players can join your lobby. In the lobby you can manage settings for the game before starting.
6. There should be several other game modes that can be chosen in the lobby.
7. You should be able to chose from several upgrades to ships and colonies which can be bought on planets.

1.5 Definitions, acronyms and abbreviations

All definitions and terms regarding the core RiskySpace game are as defined in the references section.

- GUI, graphical user interface
- Java, platform independant programming language.
- JRE, the Java Run time Enviroment. Additional software needed to run Java application.
- Round, one complete game ending with one winner.
- Turn, each player's turn. The player can only act during his or her turn. The player can be affected during other players turns.
- Metal, basic building resource used for all ships and building and some research.
- Gas, advanced resource used for some ships, buildings and research.
- Recruitment, The action where a player start constructing a ship on one of his or her planets.
- Movement, the action of moving ships to new places in space.
- Selection, the action of selecting one's ship or planet in order to perform movement or other action.
- Colonization, the action of sending one's Colonizer ship to an empty planet in order to set up a small base, gaining resources and maintaining a defensive position.
- Planet, a planet in the game that is either owned by a player or empty. There are two kinds of planets, metal and gas, where a player can mine resources.
- Ship, a space ship of any kind owned by a player.
- Enemy, the other player(s).
- Battle, a battle occurs after you have moved ships into enemy territory or a territory occupied by enemy ships. If a battle of a planet was lost the colony is destroyed and the planet is uninhabited.
- Distance, the amount of movements required to get from territory A to territory B.
- Energy, the amount of movements a ship is capable of during one turn.
- Shield Status, status during battle of how much damage this ship can take before being destroyed. All ships have base Shields and they give the starting Shield Status of every battle. Shields are regained when repaired at a Colony with such ability.
- Fire Power, the fire power of any ship or planetary defence is their damage capability during their attacks in battle. If a ship has fire power of (5-8) they can do any number from 5 to 8 damage to a random ships or defence's shield status.
- Host, the computer which runs the game. In a local game it is the computer that the game is played on and in network games it is the computer that started the

lobby.
- Clients, other computers that connect to the host in network games.
- Lobby, the lobby is a screen where settings can be made for new games and were you can wait for new players to join. If you chose to load a game other players may still join but you can not change the game’s settings.
- Map, the map on which the game will be played, this may also determine the game mode.
- Territory, an area on the map. The map is built up of 20x20 territories and each territory may contain a planet and any number of ships.

2 Requirements

2.1 Functional requirements

1. Start a new game.
2. Do a turn. During the turn, he or she can
   a) Move ships, can trigger events like; battles with enemy ships or colonization of planets.
   b) Build ships (and choose where they should be built at).
   c) Self-destruct ships that you own.
   d) End the turn.
3. Exit the application. Will end turn, round and game.

2.2 Non-functional requirements

2.2.1 Usability

It should be easy to create games and share your ip address with friends or other people so that they can join your game.

The GUI should be easily navigated and intuitive to make sure that everyone experience smooth games.

The game should not be complicated enough to confuse new players too much, rather should they get into it after a few turns.

2.2.2 Reliability

The server should only start when the host presses start and should only shutdown when the host disconnects from the game. The clients should only disconnect from the server if the player disconnects from the game or if the server shuts down.

All user input should be handled by the server, no data should be lost in the transfer between server and client.

2.2.3 Performance
The game is set to render at 60 frames per second and this should be maintained by most computers, it should never drop below 30 frames per second.

Communications between the server and the clients should be fast and without data loss, even when under pressure.

2.2.4 Supportability
The application should be easy to extend with new game modes and features. The graphical style should be easy to replace without the model interferance.

The GUI should be easily replaceable without decreasing functionality of a client. An example could be a change to a different rendering system.

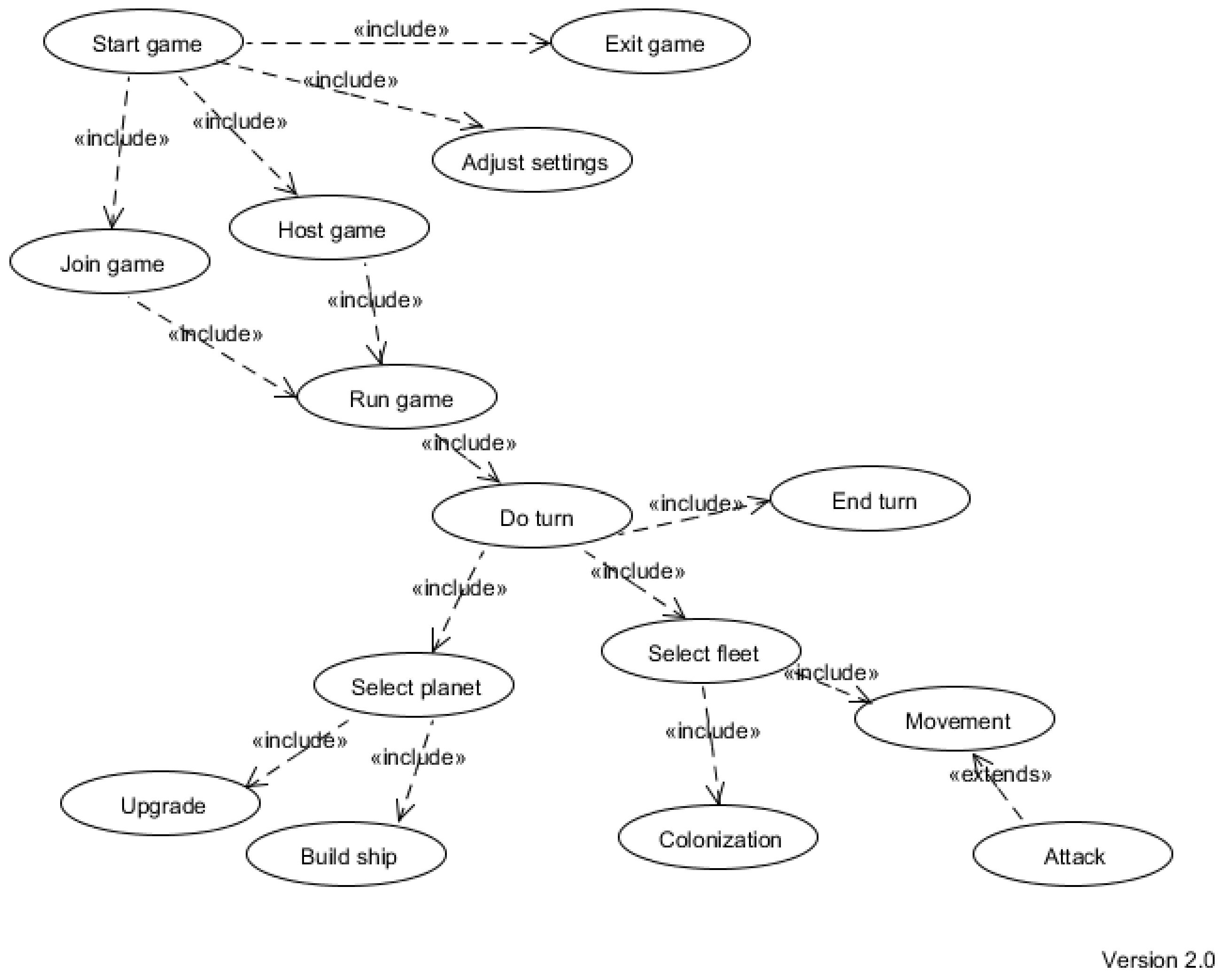
2.2.5 Implementation
The application will be implemented as a java application and the Java Runtime Environment will be needed in order to run properly. To maintain system independence we will avoid using libraries and packages that might reduce these properties, and instead find better solutions.

2.2.6 Packaging and installation
The game should be runnable with all binary .class files and resource files available. JAR files containing all resources will not be available as this disables some code structures.

2.2.7 Legal
N/A

2.3 Application models

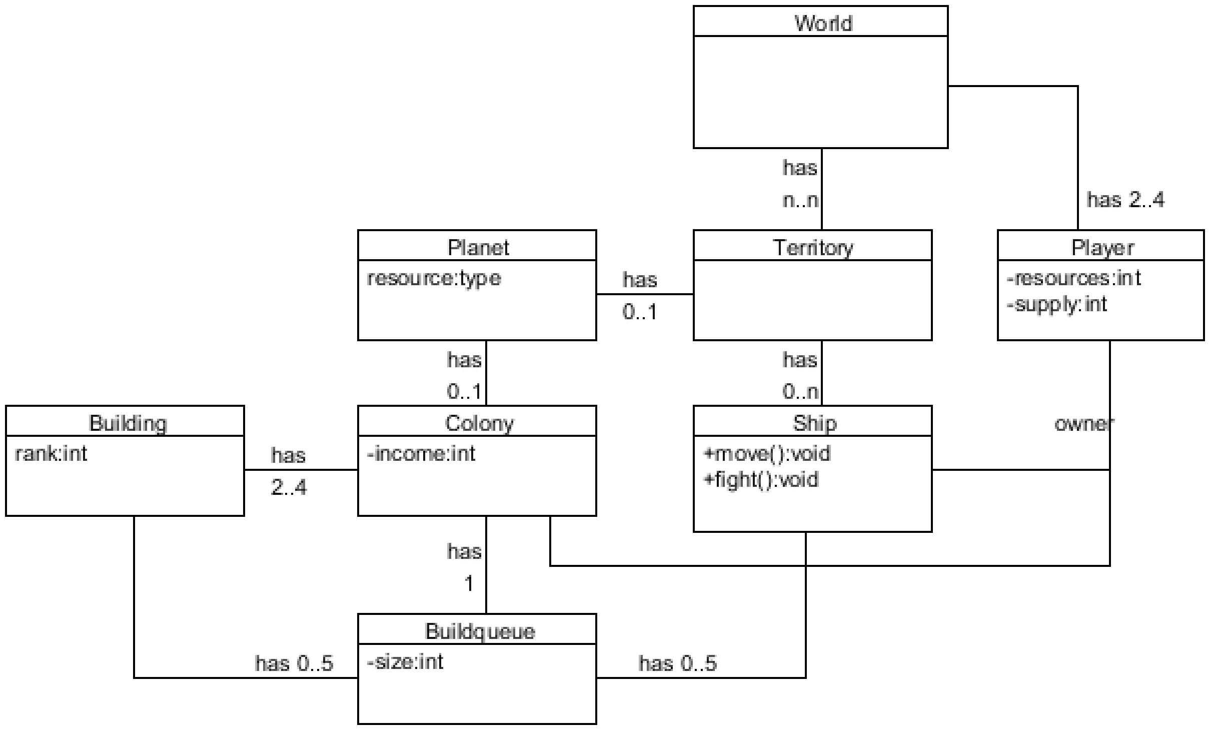
2.3.1 Use case model



2.3.2 Use cases priority

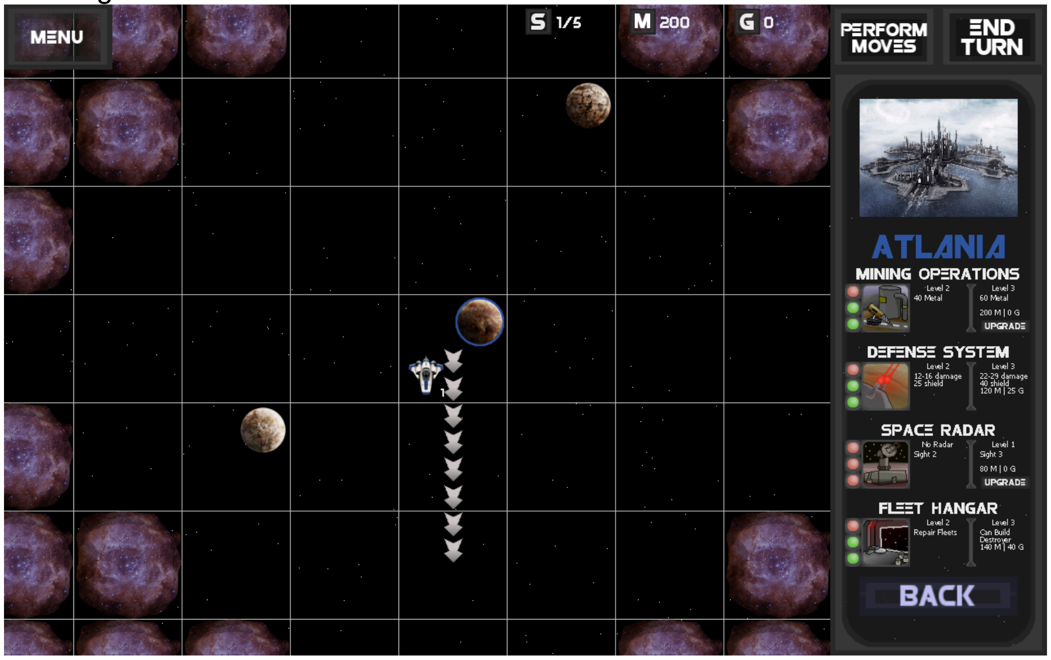
See Appendix I.

2.3.3 Domain model



2.3.4 User interface

GUI in-game.



Appendix I

Use case: ***Movement***
Short description: Move spaceships based on their speed
Priority: High
Extends or Includes (Attack, Select unit)
Participating actors
- Active Player

Normal flow of events
1. The active player selects the unit he wants to move
   a. The fleet is selected, see UC Select Unit.
2. Right click to set target
   a. Arrows representing the fleets path are drawn
3. Click the button “Preform Movements” to start movements
   a. Ships move to their target with an animation, energy is used.

Alternate flow: Enemy present at target location
3.1 Ships move into enemy territory
a. Battle is fought at target location. See UC Attack

Use case: ***Select Planet***
Short description: Select the planet you want to interact with.
Priority: High
Extends or Includes ()
Participating actors
- Active Player

Normal flow of events
1. Player clicks the planet he wants to select.
   a. If the unit is a neutral planet a picture is displayed based on the planet type. If the planet belongs to the active player the planet menu will appear and you will be able to recruit ships or upgrade buildings. (see UC RecruitShips and UpgradeBuildings).

Use case: ***Select Fleet***
Short description: Select the fleet you want to interact with.
Priority: High
Extends or Includes ()
Participating actors
- Active Player

Normal flow of events
1. Player clicks the fleet he wants to select.
   a. If the unit is a fleet of ships and it belongs to the active player the ship types and the number of ships will be displayed.

Use case: ***Attack***
Short description: attack an enemy fleet or planet.
Priority: High
Extends or Includes (Select Unit, Movement)
Participating actors

- Player

Normal flow of events:

1. Player selects a fleet.
    a. The fleet is selected, see UC Select unit.
2. Player moves them to the location of an enemy fleet or planet, see UC Movement.
    a. A battle is fought based on the number of ships, the ship types and if a planet is present, the planet´s defence systems is also taken into accord. The battle goes on until one of the forces are destroyed.

Use case: *Colonize*

Short description: Establishing a colony on a neutral planet.

Priority: High

Extends or Includes (Select Unit)

Participating actors

- Player

Normal flow of events:

1. Player moves his colonization ship to a neutral planet.
    a. A colonize button becomes visible when you select the neutral planet
2. Players clicks on the colonize button.
    a. A colony is built and the planet now generate additional income to the players resources. Depending on what kind of planet it is, either metal or gas is generated. The colony receives a basic defence turret to protect it from raids and the player can now use additional supply.

Use case: *Recruit Ships*

Short description: Recruit more ships

Priority: High

Extends or Includes (Select Unit)

Participating actors

- Active Player

Normal flow of events

1. Player selects the planet where he wants to recruit his units.
    a. The planet is selected, see UC Select Unit.
2. Player chooses which kind of ship to recruit, then clicks the respective symbol. Based on the hangar that is available on the planet different kind of ships constructable.
    a. The ship is put in a build queue and resources are depleted based on the ships cost. Based on it´s construction time the ship will appear next to the planet in a set amount of rounds.

Use case: *Upgrade Buildings*

Short description: Upgrade the buildings on a planet

Priority: Medium

Extends or Includes (Select Unit)

Participating actors

- Active Player

Normal flow of events

1. Player selects the planet where he wants to upgrade his buildings.
    a. The planet is selected, see UC Select Unit.
2. Player chooses which kind of building to upgrade, then clicks the respective symbol.
    a. The building is put in a build queue and resources are depleted based on the

upgrades cost. Based on it´s construction time the building will be upgraded in a set amount of rounds.

Use case: ***Move Map***

Short description: Moving the map to view a new location.
Priority: High
Extends or Includes ()
Participating actors
- Player

Normal flow of events:
1. Player moves the cursor to the edge of the screen
    a. The part of the map that is currently viewed is changed for one that is in the direction of where your mouse cursor is.

Alternate flow: player uses the keyboard instead of the cursor.
1. Player presses either one of the arrow keys
    a. The map is moved in the respective direction of the key pressed. left for west, up for north, down for south and right for east.

Use case: Host Game

Short description: Host a game
Priority: High
Extends or Includes (Start Game)
Participating actors
- Player

Normal flow of events:
1. Player clicks on multiplayer button in the start screen.
    a. A small menu appears
2. Player clicks on Host Game..
    a. A lobby appears where you can set the game mode and the number of players.
3. Player clicks on Create Game.
    a. The chosen game settings are locked and the lobby appears with the data that was selected before you created the game, other players are now able to connect to your game.
4. Player clicks Start Game button.
    a. The game starts, can only be done when the selected number of players have connected.

Use case: ***Join Game***

Short description: Join a hosted game
Priority: High
Extends or Includes (Start Game)
Participating actors
- Player

Normal flow of events:
1. Player clicks on multiplayer button in the start screen.
    a. A small menu appears
2. Player enters the IP address of the game in the textbox and then presses Join Game.
    a. The players enters a Lobby with a host and possibly other players. The player can only wait for the game to start or leave it. No configurations can be made to

the game settings by a player that joins a game.

Use case: ***Start Game***

Short description: start the game.
Priority: High
Extends or Includes (Start Game)
Participating actors

- Players

Normal flow of events:

1. Player clicks the Start Game button.
    a. The map is generated and the players receive their start planets, resources and start units. The player that gets to do his moves first is based on connection order. The order is then maintained through the game.

Use case: ***Load Game***

Short description: load a previously saved game.
Priority: Medium
Extends or Includes (Start Game)
Participating actors

- Player

Normal flow of events:

1. Player clicks Load Game button.
    a. A list appears with saved games.
2. Player chooses which game to load by selecting it.
    a. The chosen game is somehow displayed differently so you know which game has been selected.
3. Player clicks Load Selected Game button.
    a. The chosen game is generated and the lobby appears with the data that was selected when you created the game.
4. Player clicks Start Game button.
    a. The game starts, the map and whatever resources, planets and units players had acquired is generated.