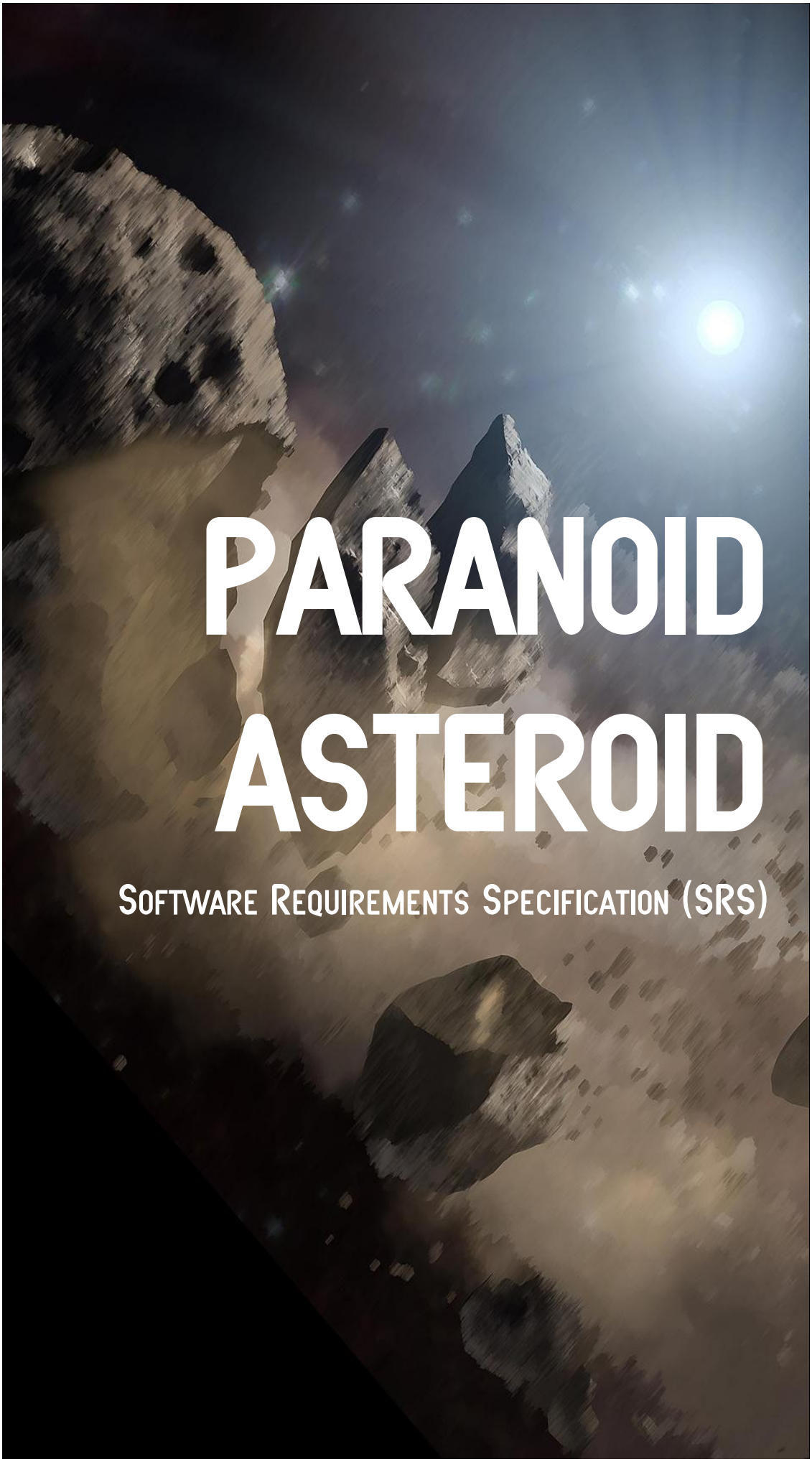


PARANOID ASTEROID

Software Requirements Specification (SRS)

ECSE 321
Introduction to Software Engineering

Group 8:
Alex Bourdon
Alexander Coco
Payom Meshgin
Daniel Ranga
Jad Sayegh
Yi Qing Xiao

Client
Prof. Haibo Zeng

Revision No.: 1

Date: Sunday, February 10, 2013

Table of Contents

1. Introduction

1.1 Purpose

This document is the system's requirement document for the design project of the first release of the "Paranoid Asteroid" (referred to as PA thereafter) video gaming system and describes in details and entirety the specifications and functions of the product.

1.2 Scope

The gaming system will be designed to facilitate the game play of the arcade game Asteroids on modern household personal computers. A user interface will allow the user to either access their high scores or game statistic, or play the game which will be graphically implemented.

The end product will allow the user to re-experience the classic arcade game on their own personal computer. The game will adhere to the original's gameplay but will nonetheless support added features not present in the original arcade game. In the game, the user will have control over a ship's heading and movement and firing rate. Game features also include impact detection, two player modes, difficulty adjustment, bonus life drops, a sound system, a physics engine for the asteroids and enemy spaceships, as well as a simple AI implementation for the latter.

1.3 Definitions, acronyms, and abbreviations

- *Asteroids*: original name of the Atari space game.
- playing field : area with wrap-around edges where game sprites move (star field, outer space)
- spaceship, player : player animated sprite, controlled via the keyboard
- Asteroid: inanimate game sprite, travels on the field at constant velocity. varies in size (velocity depends on size; smaller, faster)
- Alien: non-playable character which exists as two different versions, Large Alien and Small Alien, which travel on the field at constant velocity and fire projectiles which cause the PC to lose a life on impact.
- Flying saucer, alien: AI animated game sprite, travels across playing field, able to fire projectiles. exists in 2 formats, large and small (behaviour differs)
- Hyperspeed: player action. When activated (via keypress), spaceship disappears from current location and reappears at a random space in the playing field.

- SRS: Software Requirements Specification document, which specifies the functionalities of the system and its operation constraints.
- PC: playable character, sprite controlled by the user in-game and during the tutorial. ## References

1.4 Overview

The description will be given in form of text, mock-ups and prototypes of some of the graphical elements displayed by the system. Its intended audience consists of the client company, its project managers, marketing staff, users, tester and documentation writers, in addition to McGill University professor Haibo Zeng and his teaching staff. The rest of the document is organised in two sections and contains all the information about the requirements of the system. Section 2 is primarily intended for users and contains an overall description of the system, its functions, and constraints. It outlines product use assumptions and use cases. Section 3 describes the specifics of the required product characteristics and behaviors relevant to functionality, quality and system testing. It is intended primarily for the product developers.

2. Overall description

2.1 Product perspective

Our Asteroids remake will be an independent and self-contained PC application, executed in a Java runtime environment. The simplicity of this game will not strain any modern PC's processor, RAM or memory. The Game package should not be bigger than 20 MB. The input devices will be a standard keyboard and mouse. Gameplay controls will implemented on the keyboard and the choice of keys will be programmable. Game configuration will be done with the keyboard and mouse. The output will be shown on the PC's screen. The game will run on a simple graphics generator similar to Java Swing. The game window will be a static size (600 x 800) to make gameplay consistent. This third person shooter set in outer space will be a 2D environment which aims to be a faithful to the original game as possible. The game will have three possible modes, personal high score, two player alternating and two player simultaneous. The game structure will be multi-level with increasing difficulty. Weapon capabilities will also evolve in step with the difficulty.

2.2 Product functions

2.2.1 Overview

The entire gameplay revolves around controlling a spaceship while destroying enemies (using a weapon) and dodging them. The main objective of the game is to complete as many levels as possible. Each level consists of clearing the playing field of asteroids by shooting a projectile (or missile) at them.

At the beginning of each level, a number of asteroids appears near the edge of the playing field. The player character, represented by a spaceship, is placed in the middle of the playing field. The player is able to navigate the spaceship by controlling its direction as well as its thrust using the keyboard. The player can also, at any time, activate a "hyperdrive", that is, teleport the ship to a random area on the field in order to evade an oncoming enemy. When hit by an enemy or an enemy weapon, the ship explodes and signals the loss of a life. When the player has no more lives left, the game is over.

All user options, in addition to save game and high score data will be stored in files.

2.2.2 Modes of Play

The game can be played in three distinct modes.

- The first, called **"single-player" mode**, a user plays alone, controlling a single spaceship as each level is cleared. The player starts out with 3 lives, which can be lost if the player's ship collides with an enemy or an enemy weapon (see game elements). The player accumulates a score as enemies are destroyed. After the player's game is over and if his score is among the ten high scores, a high score screen appears, allowing the player to enter his name.
- The second mode, called **"two-player contest"**, involves, as its name suggests, two players playing competitively against each other. The gameplay is nearly identical to the single-player mode (each player plays alone and starts off with 3 lives), but in this mode, the two players take turns playing solo. A player loses a turn when he loses a life. When one of the players has lost all their lives, the other player gets to play. When a player regains his turn, he resumes his game exactly how it was left before
- A third mode, called **"two-player co-op"**, allows the players to play cooperatively by playing simultaneously on the same playing field and racking a common score.

2.2.3 Game elements

2.2.3.1 The user interface / Menu system

The user interacts with the game through a menu system. There are two different menu systems in the game: the main menu system and the pause menu screen.

2.2.3.1.1 Main menu system

The main menu system, which is shown upon the launch of the game, includes the following options:

- Start Game: Starts the game setup screen
- Load Game: Loads a previously saved game
- Options: The options menu changes various personal options, such as controls, volume settings, difficulty and enabling/disabling various bonus game features.
- Tutorial/Instructions: Introduces, using text and images, the controls of the game to the player. A button allows the player to access a tutorial mode which accustoms the player to the gameplay and game rules. This playing mode instantiates a PC in an enemy-less field. This mode helps the player get accustomed to the control of the PC.
- High Scores: A new frame shows the top 10 high scores as well as the names of the players who attained those scores.
- Credits: A simple static frame displaying the names of the development team.

2.2.3.1.2 Pause menu system

While in-game, the player can pause the game to reveal a pause menu screen. In the menu, the user is able to unpause his game, save the state of the game, review options, change volume settings and exit the game

2.2.3.2 In-game:

There are a number of visual elements displayed to the player to track his progression in the game. While in-game, the player is able to view his current game score, his remaining number of lives as well as the current global high score attained in the game.

2.2.3.2.1 Playing Field

A graphical representation of the playing area that all game elements are bounded by. It is represented by a solid black rectangle spanning the entire game window. The key feature of the playing field is the **wrap-around** effect, that is, any object passes the edge of the field appears at the opposite edge of the field.

2.2.3.2.2 Enemy

Any non-player character that harms the player or hinders the player's progress. Some enemies may also be equipped with a weapon. They consist of asteroids, small aliens and large aliens.

Asteroid

Asteroids are the main obstacle in the game. When they hit something or get hit by a weapon, they explode. There are three stages of asteroids:

1. Large stage
2. Medium stage
3. Small stage

While in the large stage or medium stage, an asteroid splits into two asteroids of a smaller stage (large => middle => small).

Large Aliens

Large aliens are spaceships armed with a missile similar to the PC's weapon. They are slow and non-intelligent, regularly firing their weapon in a random direction while wandering the field on a random path.

Small Aliens

Small aliens are the toughest enemy in the game, since they are the only enemies equipped with an AI. They are four times smaller than large aliens but are just as fast. They are also equipped with a simple missile weapon, and unlike their larger counterparts, they fire in the direction of the PC.

2.2.3.3 Music/Sound

The various sound effects used to enhance the user experience. All sounds will be borrowed from the original 1979 arcade version of Asteroids. In particular, the music beat, as was implemented in the original game (see reference), plays at an increased tempo as the density of asteroids on the field

decreases, like the original. Sounds will be used to signal: the explosion of the ship, the firing of a weapon, the pickup of an item and the siren of an alien ship.

2.2.3.4 High Score screen:

After the player's game is over and his score is among the ten high scores, a high score screen appears, allowing the player to enter his name.

2.3 Use Cases

2.3.1 Program Launch

User case name:	Program Initiate
Participating Actors	Player request game software to start
Entry Conditions:	The player open the software
Flow of Events	- Player click "open" - Software initialized - Game display main menu - Game waits.
Exit Conditions	Main menu displayed and game is waiting
Quality Requirements	- The game must be able to open the main menu and display it with all options of the menu available.

2.3.2 Main menu navigation

User case name:	Pick main menu choice
Participating Actors	Player select one of the options on the main menu
Entry Conditions:	Player clicks on one of the options
Flow of Events	- Player clicks on one of the options - Game display the requested section accordingly
Exit Conditions	The correct requested section has been reached
Quality Requirements	- The links to the next sections must be correctly associated with each of the options on the main menu

User case name:	Input Control
Participating Actors	Player input in-game commands
Entry Conditions:	Player pushes the correct button for the intended resulting in-game action (shoot, thrust, turn, level clear bomb, hyper drive, pause, save, exit)
Flow of Events	- Player pushes button - The game output the correct response
Exit Conditions	The expected response from the game occur
Quality Requirements	- Buttons must be correctly assigned to their intended purposes. - The response time must be quick, especially when the player is attempting to fire bullets. - If two or more button pushed at once, use a priority list to determine which button dominates.

User case name:	Display score board
Participating Actors	Player playing the game and the game
Entry Conditions:	Player select score board in the main menu
Flow of Events	- Player select score board - Game displays the score board.
Exit Conditions	The score board has been displayed
Quality Requirements	The score board must be displayed when the button is clicked, and the main menu will disappear, though a button which allows the player to return to the main menu will become available.

User case name:	Display Instructions
Participating Actors	Player playing the game and the game
Entry Conditions:	Player select the display instruction button in the main menu

Flow of Events	- Player click on "Instructions" in main menu. - Game display the instructions for the gameplay.
Exit Conditions	The instructions are displayed.
Quality Requirements	- The displayed instructions must correctly demonstrate how each and every in-game related button functions.

2.3.3 START GAME

User case name:	**Start game**
Participating Actors	The player playing the game and the game
Entry Conditions:	Player select the "Start Game" option in main menu
Flow of Events	- Player click on "Start Game" - Game display gameplay options: 1 or 2 player modes | coop or alternate modes. - Player select mode - Game display the actual Asteroid game plain, and start the game.
Exit Conditions	The asteroid plain is displayed and the player ship(s) and the asteroids are created.
Quality Requirements	- Game modes must correspond to their intended results. - When choosing the 2 player mode, choices between coop and alternate modes must be displayed for the player(s) to choose. - After all choices are made, the asteroid field generation should be slightly delayed after the player ship generation to let the player(s) get ready.

2.3.4 Use cases directly related to gameplay

Input Commands	**Player inputs the commands in the actual game: shoot, thrust, turn, level clear bomb, hyper drive, pause, save, exit.**

Game Pause	**Game pauses, all in-game objects cease their movement.**
Activation Cond.	- Player press pause button (p by default) - Automatically pause game when saving or when the game just loaded

Save Game	**Save the present state of game in a file**
Activation Cond.	When player press the save game button.
Include	Game Pause

Load Game	**Load a save file and resume the session**
Activation Cond.	When player select the "Load Game" button in the main menu.
Include	Game Pause

Control Ship	**Turn or thrust the player ship**
Activation Cond.	Player pressed the turn or thrust button during game

Create Drops	**Create a power-up item**
Activations Cond.	When an alien or an asteroid gets destroyed. Frequency of drops must be respected.

Destroy Drops	**Destroy a drop item**
Activations Cond.	Drop's timeout or picked up by the player's ship.

Destroy Bullets	**Destroy a bullet**
Activations Cond.	Bullet timeout or bullet impact against any other in game object (even bullets)

Destroy Aliens	**Destroy an alien, leaving a chance of generating a drop item.**
Activations Cond.	- Collision with any in game objects - Activation of "level clear bomb"
Extends	Create Drops

Destroy Asteroids	**Destroy an asteroid, maybe splitting it (depending on original size).** **May drop power-up when original size was minimal.**
Activations Cond.	- Collision with any in game objects - Activation of "level clear bomb"
Extends	Create Drops, Split Asteroids

Split Asteroids	**Generate smaller asteroids when a large asteroid got destroyed. Initial momentum and directions of the generated smaller asteroids depend on original asteroid's momentum and direction**
Activations Cond.	A large asteroid got destroyed DUE TO COLLISION.
Include	Create Asteroids

Create Bullets	**Create a bullet moving in the direction the human ship is facing**
Activations Cond.	Player pressed the fire button

Create Aliens	**Create an alien (large or small), depending on time.**
Activations Cond.	- Present level play time has reached a certain length - A set amount of time passed since last creation of an alien

Reset Level	**Refill the asteroids in the plain after the previous plain has been cleared.**
Activations Cond.	All non-layer vessel objects are destroyed in the present level.
Include	Create Asteroids

Create Asteroids	**Create asteroids, whose size depends on the activation conditions.**
Activations Cond.	- Beginning of a level - A large asteroid got destroyed (i.e. Splitting)

Start Game!	**Begin brand new session**
Activations Cond.	- Player has pressed the "Start Game" button and made choices about the game mode.
Include	Create Asteroids, Create Player Ship

Create Player Ship	**Create the player's ship**
Activations Cond.	- Player started a new game - Player's previous ship got destroyed while having non-zero life count - Player used Hyper Drive

Hyper Drive	**Change the position of the player's ship at a random location on the plain.**
Activations Cond.	Player pressed the hyper drive button
Include	Create Player Ship, Destroy Player Ship

Destroy Player Ship	**Destroy the player's ship**
Activations Cond.	- Player's ship collided with any object (Except other player's ship in coop mode) - Player used hyper drive
Extend	Create Player Ship, Drop Life Count

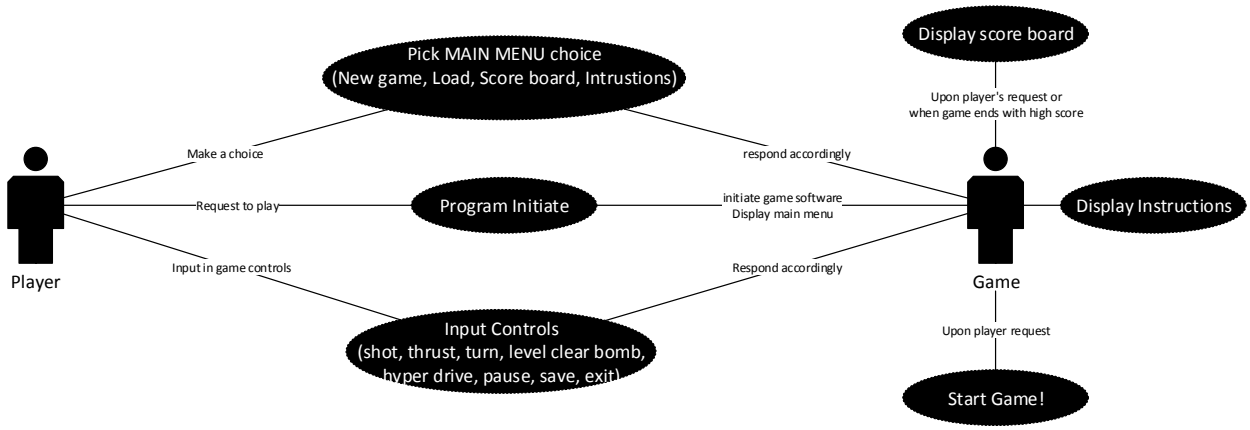
Drop Life Count	**Subtract the amount of remaining player's life by 1**
Activations Cond.	Player's ship got destroyed DUE TO COLLISION
Extend	End Game

End Game	**End the present session**
Activations Cond.	Player's LAST ship got destroyed (or negative life count)
Include	Input Name for Score Board

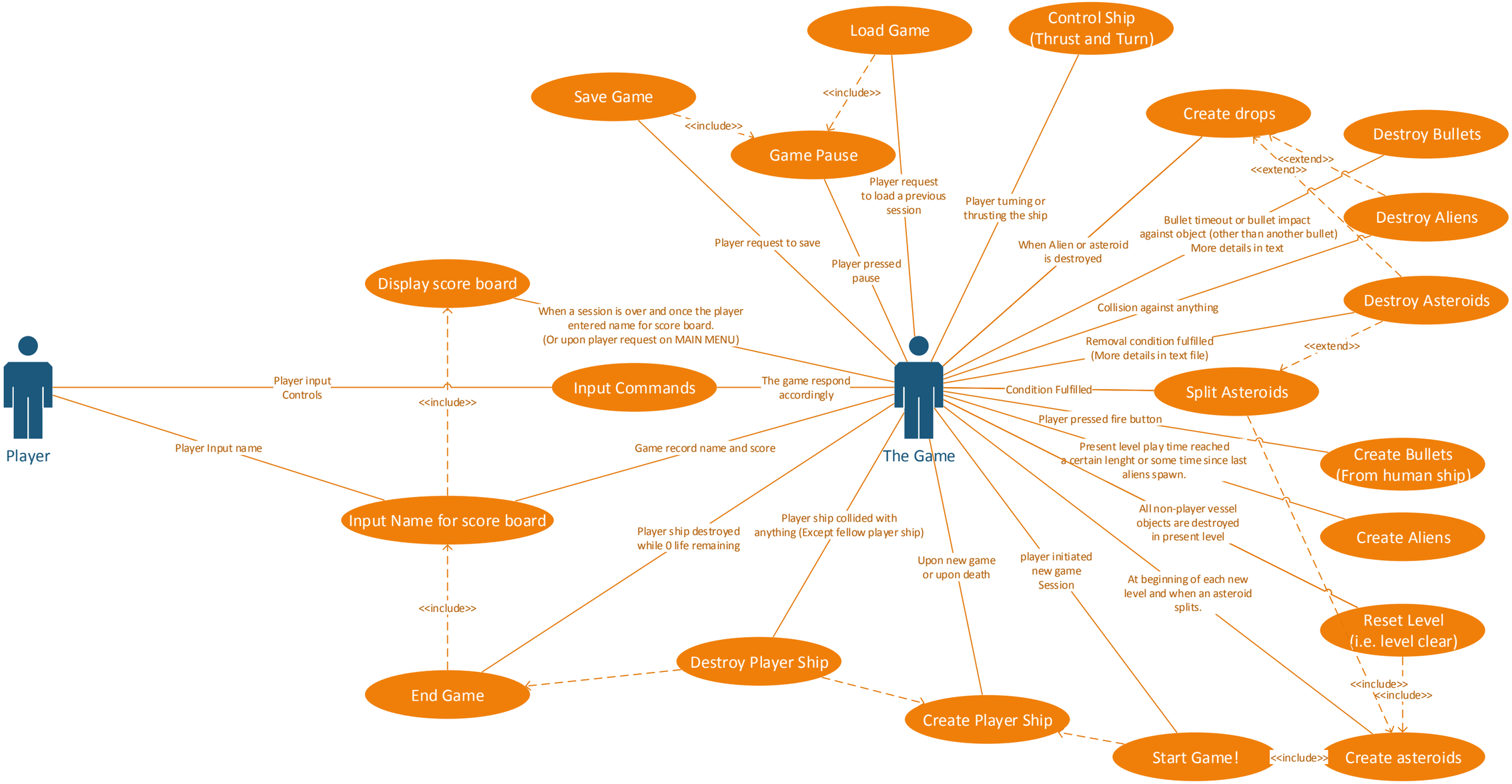
Input Name for Score Board	**The game prompts the user to input a name for registering this session's score.** **The player input a 3 characters long nickname.**
Activations Cond.	A game session just ended.
Include	Display Score Board

Display Score Board	**Display the high score board**
Activations Cond.	- Player clicked the "Score Board" button in the main menu - Player has ended inputting nickname for registering a session's score

2.3.5 Use Case Diagram for Game Launch and Menu Navigation



2.3.6 Use Case Diagram for General Gameplay



2.4 User characteristics

The users of this system are the human players. The human user will be controlling the game start-up and gameplay though the PC's keyboard and mouse. Once the game has begun the player will only have control over the space craft and its weapons. The experience of the human player is paramount. Start-up should be simple and short with easily found parameters. Response to player commands should be quick and forgiving. The reactions of enemies and obstacles should be simplified to make the game play more enjoyable. That is to say they will be more vulnerable than the human player.

2.5 Constraints

The major constraints are time and expertise. We have one fifth of a semester and no previous experience with a gaming project. The graphics interface will also be a constraint; Swing is a UI package, not a game engine. This will force us to simplify the game aesthetics and play. Fidelity to the original game also poses constraints on the design. As such the graphics will be two tone, two dimensional and low resolution. The sound will also be simplified to reflect the original game.

2.6 Assumptions and dependencies

It is assumed that the user's computer can run provided binary game files either compiled for the specific platform or can run them through an interpreter or virtual machine. The computer must have as a minimum the performances outlined in the product perspective. It is also assumed that the player has basic knowledge of launching programs in their operating system. It is assumed that the player has played a video game before and might not use the tutorial before starting the game. It is assumed that the system will have a standard keyboard with US mapping. We also assume that the system will have a built in colour screen. It will be also assumed that the cooperative two player will be played on one shared keyboard.

3. Specific requirements

1. Mode 1: Menu System

1.0 General Functionality

1.0.1 The menu system provides a way for the user to access other parts of the game as well as change settings for the game. It is either a separate window or a menu bar embedded into the gameplay window **(essential, easy)**.

1.1 External Interfaces

1.1.1 The main way to interact with the menu will be with the display and the mouse **(essential, easy)**.

1.1.2 The menu window includes the following buttons : 'Load Game', 'New Single Player Game', 'New Two-Player Contest', 'New Coop Game', 'High Score', 'Game Instructions and Preferences', 'Credits', 'Exit' **(essential, easy)**.

1.1.3 All windows (Menu, High Score, Gameplay, etc.) will include an exit button, in the top right corner, marked as an 'X' which will close the game, cancelling any activities that are occurring **(essential, medium)**.

1.1.4 When the user clicks on a button in the menu, the game will respond appropriately and display the correct sub-menu such as an options page or help dialog **(essential, medium)**.

1.2 Functional Requirements

1.2.1 Clicking the 'Load Game' button will close the main menu window, open the gameplay window and load a previously stored game **(essential, medium)**.

1.2.2 Clicking the 'New Single Player Game' button will close the main menu window, open the playing field window and will start a new single player game **(essential, medium)**.

1.2.3 Clicking the 'New Coop Game' button will close the main menu window, open the game play window and will start a new game in which two players play together on the same game play field **(desirable, hard)**.

1.2.4 Clicking the 'New Two-Player Contest' button will close the main menu window, open the game play window and will start a new game in which two players play one after the other **(essential, hard)**.

1.2.5 Clicking the 'High Score' button will open the High Score window (leaving the main menu open) **(essential, medium)**.

1.2.6 Clicking the 'Game Instructions and Preferences' button will open the Control/How-To-Play window (leaving the main menu open) **(desirable, medium)**.

1.2.7 Clicking the 'Exit' button will close the program **(essential, medium)**.

1.2.8 Clicking the 'Credits' button will open the Credits window (leaving the main menu open) **(desirable, medium)**.

1.2.9 In the settings menu, the user will be allowed to change the controls for the game **(optional, medium)**. In this situation, keyboard interaction will be required as well as the mouse and display.

1.2.10 The user will be able to choose between different game modes such as single player **(essential, medium)**, turn-based two player **(essential, difficult)**, and two player cooperative **(optional, difficult)**.

1.3 Performance

1.3.1 Starting the game will take up to 1 second **(essential, medium)**.

1.3.2 Transitions from the Main menu to any other non-gameplay window will be executed within 500ms **(essential, medium)**.

1.3.3 Transitions from the Main menu to any new game gameplay window will be executed within 1 second **(essential, medium)**.

1.3.4 Transitions from the Main menu to loading a saved game to the gameplay window will be executed within 1.5 second **(essential, medium)**.

2. Mode 2: In-Game

2.0 General Functionality

2.0.1 This section discusses the actual gameplay component of the game. These requirements are specific to the time between the game has been started and the player has lost or ended the game. These requirements involve the player, the player's ship in the game, and the enemy units in the game **(asteroids and aliens)**.

2.1 External Interfaces

2.1.1 During the game, the user will interact with the system with their keyboard **(essential, medium)**.

2.1.2 Almost all feedback to the user will be delivered via the display **(essential, medium)**.

2.1.3 Some additional feedback may be present in the form of sound effects **(desirable, difficult)**.

2.2 Functional Requirements

2.2.1 A player will be able to interact with his or her ship using the keyboard **(essential, medium)**.

2.2.2 They will be able to turn their ship clockwise or counter-clockwise using two different keys on the keyboard **(essential, medium)**.

2.2.3 The player will also be able to move the ship forward with a key on the keyboard **(essential, medium)**.

2.2.4 The ship is able to enter hyperspace which will allow the ship to disappear and reappear at a random location on the screen using a key on the keyboard **(desirable, medium)**.

2.2.5 The ship may be subject to inertia: the ship will continue to move in the direction it was moving while it slows down **(optional, difficult)**.

2.2.6 The ship and asteroids will be able to wrap around the screen. In other words, if they pass the edge of the screen they will continue moving in the same direction but will appear at the opposite edge **(desirable, difficult)**.

2.2.7 The player's ship and the aliens will be able to fire shots **(essential, medium)**.

2.2.8 The player's ship can only fire shots in the direction they are facing **(essential, easy)**.

2.2.9 The shots fired have a limited time to live in the game and will disappear after this time has expired **(essential, medium)**.

2.2.10 When the shots collide with an enemy unit the shots will disappear **(essential, easy)**.

2.2.11 When aliens interact with a shot, it will either die or lose hit points until they reach zero, then die **(essential, medium)**.

2.2.12 If an asteroid interacts with a shot, it will break up into smaller asteroids **(essential, difficult)**.

2.2.13 When the smallest sized asteroid interacts with a shot, it will disappear **(essential, medium)**.

2.2.14 Every time the player's ship fires a shot and hits an enemy it will gain points **(essential, easy)**.

2.2.15 When an enemy is destroyed, there is a small chance that a power up is dropped **(optional, medium)**.

2.2.16 When the player's ship collides with the power up, the ship will gain points and receive an ability or upgraded weapon **(optional, difficult)**.

2.2.17 If the player is able to clear the screen of all the asteroids, then they receive extra points **(essential, easy)** and the next level starts with a new set of asteroids **(desirable, medium)**.

2.3 Performance

2.3.1 The game will run on one machine only. Up to two users may be supported but it will not be over a network. Cooperative play will use the same keyboard for two players **(optional, medium)**.

2.3.2 There will be no writing of files during the gameplay **(optional, medium)**.

2.3.3 The game is expected to update and render several times per second, ideally at a variable frame rate and controlled update count **(desirable, medium)**. As such, the responsiveness for the gameplay may depend in part on the hardware the user is running the game on.

2.3.4 The game will target a large number of standard hardware and should run on most modern household computers **(desirable, medium)**.

3. Mode 3: High Score Screen

3.0 General Functionality

3.0.1 The high score screen allows the player to view the highest scores achieved in the game on that machine. The window will contain the 10 best scores and the associated text to the score, and the number of games played **(desirable, easy)**.

3.1 External Interfaces

3.1.1 The high score screen will be an extension of the menu system. This screen will interface with the display and mouse. The screen will also include an 'OK' button **(desirable, easy)**.

3.1.2 File system access will be required in order to store the score information **(desirable, easy)**.

3.2 Functional Requirements

3.2.1 After a game, once the High Score screen is open, if the player's score is better than the last high score, the player's score is inserted below the score immediately larger in the scoreboard. Scores below the current score are shifted down. Scores beyond the 10th score in the scoreboards are removed from the list **(essential, medium)**.

3.2.2 Once the score shift occurs the player is permitted to add a name next to the score. The score can be composed of no more than 32 printable ASCII character **(essential, medium)**.

3.2.3 Clicking the 'OK' button will close the High Score screen and return focus to the Main menu window **(essential, medium)**.

3.2.4 The High Score screen will also have a field displaying the number of games played and lost lives **(essential, medium)**.

3.3 Performance

3.3.1 All stored saved game information is to be saved in binary form as to prevent unauthorized modification. **(desirable, easy)**

3.3.2 Returning to the Main menu (via the 'OK' button) will be executed within 500ms **(desirable, easy)**.

4. Mode 4: Game Instructions and Preferences

4.0 General Functionality

4.0.1 The Game Instructions and Preferences screen allows the player to view the different game sprites that appear in the game and understand their interactions with the spaceship, show the effects of the different controls and enable the player to modify the keys that control the different controls **(desirable, hard)**.

4.0.2 The first time the game is launched the Game Instructions and Preferences screen will appear, to help the user understand the game dynamics **(desirable, easy)**.

4.1 External Interfaces

4.1.1 The screen will interface with the display, keyboard and mouse **(essential, easy)**.

4.1.2 The screen will include static text describing the game, images illustrating game elements and sprites, and labeled field where the user can choose game controls (for both users). The screen will also include an 'OK' button **(desired, medium)**.

4.1.3 The screen will include controls for volume and enabling and disabling features **(optional, medium)**.

4.2 Functional Requirements

4.2.1 Clicking the 'OK' button will close the Game Instructions and Preferences screen and return focus to the Main menu window **(essential, medium)**.

4.2.2 Any of the controls can be changed to any keyboard key **(desirable, hard)**.

4.2.3 If a key is already assigned to a control, no other control can be assigned to that key **(desirable, medium)**.

4.2.4 The game begins with a set configuration of controls for both players **(essential, easy)**.

4.2.5 Settings are saved when exiting the screen **(essential, easy)**.

4.3 Performance

4.3.1 All stored preference information is to be saved in binary form as to prevent unauthorized modification. **(desirable, easy)**

4.3.2 Returning to the Main menu (via the 'OK' button) will be executed within 500ms **(desirable, easy)**.

5. Mode 5: Credits

5.0 General Functionality

5.0.1 The Credits screen contains static text, it contains developer names, company logo and copyright information **(desired, easy)**.

5.1 External Interfaces

5.1.1 The Credits screen interacts with the screen and the mouse, it has only an 'OK' button **(essential, easy)**.

5.2 Functional Requirements

5.2.1 Clicking the 'OK' button will close the Credits screen and return focus to the Main menu window **(essential, medium)**.

5.3 Performance

5.3.1 Returning to the Main menu (via the 'OK' button) will be executed within 500ms **(desirable, easy)**.

6. Mode 6: Pause Menu

6.0 General Functionality

6.0.1 In the Pause Menu the game is stopped and may either be resumed or saved **(desirable, easy)**.

6.1 External Interfaces

6.1.1 The Pause menu interacts with the screen and the mouse, it has a 'Resume Game' button and a 'Save and Exit' button **(essential, easy)**.

6.2 Functional Requirements

6.2.1 Clicking the 'Resume Game' button will close the Pause menu and return focus to the In-Game screen, continuing the game **(essential, easy)**.

6.2.2 Clicking the 'Save and Exit' button will save the game, close the In-Game window and return focus to the Main menu **(essential, easy)**.

6.3 PERFORMANCE

6.3.1 Returning to gameplay will executed within 500ms **(essential, easy)**.

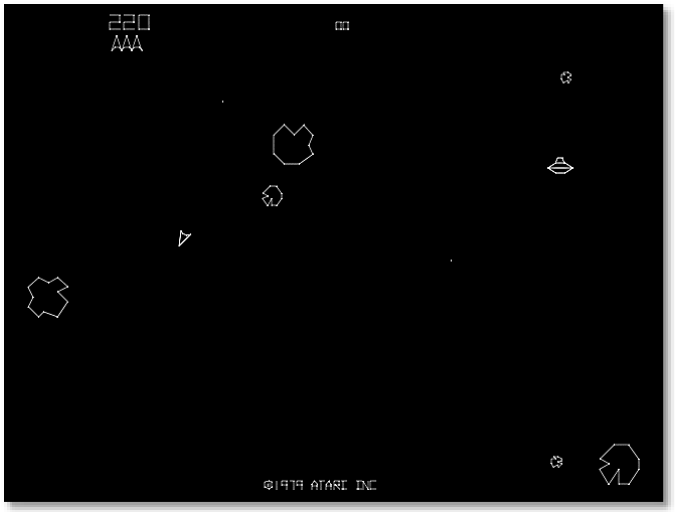
6.3.2 Saving the game and returning to the Main menu will be executed within 1 second **(essential, easy)**.

4. Appendices

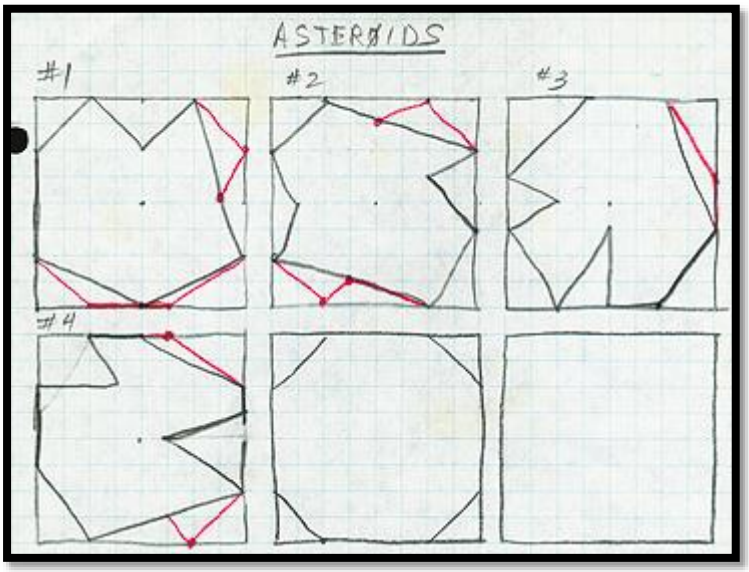
4.1 Reference in-game windows

In the screenshot below, notice the placement of the scores of players one and two on either side of the top edge of the screen. Note that the highest score is shown on the top of the screen in the centre.

In the next screenshot, note the representation and placement of the "remaining lives" indicator at the top left of the screen. We also see the three different sizes of asteroids on the playing field as well as a large alien.



4.2 Reference sprite data



4.3 Reference font data