2012

Space Shooter

Concept of Operations

COP 4331: Processes for Object Oriented Software Development
Fall – 2012

Team 2
8/25/2012

Table of Contents

Prefatory Information

Modification history:

Version	Date	Who	Comment
v0.5	08/25/2012	Joshua Thames	Initially unfinished compilation
v1.0	8/27/2012	Joshua Thames	First Completed version
v1.1	8/31/2012	Group meeting	Updates to operational features, expected impact, analysis
v1.2	9/12/2012	Joshua Thames	Formatting
v1.3	9/18/2012	Joshua Thames	Formatting, updated Expected Impacts and Analysis

Team Name:

Team 2

Team Members:

Name	Email address
Andre Meireles	andre.meireles@knights.ucf.edu
Alex Banke	banke@knights.ucf.edu
Christopher Margol	margol_chris@knights.ucf.edu
Chris Lin	christophercklin@gmail.com
Josh Thames	jthames88@knights.ucf.edu
Thaddeus Latsa	tlatsa@knights.ucf.edu

Current System

Overview

This project will be a 2D space shooter game in which the user can control an upgradable “space ship”. The game will feature an interactive environment with asteroids moving across the screen at various times, each with their own gravitational pull. The environment will also feature enemy NPC ships which will shoot at and possibly collide with the user’s ship. The user will be able to shoot at other ships to gain currency to upgrade hull, weapons, engines, and thrusters. Each upgrade will change the appearance of the user’s ship in some way.

The Proposed System

Needs

- Code hosting application
- Java IDE
- Loadable gameplay interface
- Graphics for various hulls, weapons, engines, thrusters, NPC ships, asteroids, simulated “laser shot”, background environment picture, etc.
- Web hosting server
- Internet connection
- Sound samples for laser shots, explosions, etc.
- Documentation

Users and Modes of Operation

The project will include one class of user, the pilot, who will control the ship’s movement and shooting during gameplay and will also be able to load/save gameplay, upgrade ship, move on to next level.

Two modes of operation: new game in which the user will start from the very beginning with the most basic ship at level one, and load game in which the user will start from a saved file in directory.

Operational Scenarios

Scenario 1: New Game

User selects “New Game” option from main menu in which the application will start a new game. The new game option will prompt user input for user’s name then, application will start the user with the simplest ship, no upgrades, at level one. The user will then progress from level to level if the user does not die during the level. After completion of a level or death during a level, the user will be brought to an internal menu screen with the option to save, load, upgrade ship, or exit.

Scenario 2: Load Game

User selects “Load Game” from main menu to load a previous gameplay save from a file saved in directory. The application will load all of the user’s name, currency, level number, and ship upgrades.

Scenario 3: Save Game

User selects “Save Game” from main menu in which the current user’s name, currency, level number, and ship upgrades will all be saved to a file in the directory which can be accessed at a later time.

Scenario 4: Atypical - Quit during gameplay

In this situation, if the user quits during a combat level of gameplay, he or she will lose all data until the last save point. We will have to guard against file corruption during this case.

Operational Features

Must Have:

- Menu screens(Main, Load File, Shop/save point)
- User controlled ship (movement and firing option)
- Interactive environment with asteroids having a gravitational pull and enemy NPC ships
- Upgradable ship
- Top score tracker - based on kills and asteroids destroyed.
- Visible heath meter for ship
- Auto-load function (after death)
- Currency rollover system
- Visualization
- Collision detection
- Upgradable hulls, thrusters, and weapons

Would Like to Have:

- Sound effects
- Story line(possibly educational)
- Android Support
- Mac Support
- Linux Support
- Background Music
- Multiplayer

Expected Impacts

For Users:

The purpose of this game will be for entertainment. Our goal is to provide the client with a fun, interactive, realistic, and a replay-able gameplay experience.

For Programmers:

By the completion of this project, we will have gotten practical experience with joint coding efforts, object oriented programming, professional documentation, a project management on one of the largest scales that UCF has to offer.

Analysis

Expected Improvements:

We expect to improve the 2-D “space shooter” genre by having more realistic physic forces in the environment as well as a bit of a “free form” type shooter in which the ship can shoot from any direction at any time and the user can move the ship anywhere within the environment. The user will not be fixed to only shooting “up”, or stuck to the bottom of the screen, for instance. We also see that our program will improve the current market in that, with the two above mentioned features, it will also implement a “upgrade shop” between levels so that the user can try different hull, thruster, and weapon combinations to see which best suites that particular level.

Disadvantages:

2-D instead of current more popular 3-D models, many space shooters already exist, creating our own physics engines instead of a well-tested one, creating our own graphics and sounds.

Limitations: Limited to Java and what its inherent capabilities are included. We are also limited in the scope of time allowed in that we have only this one semester to develop the entire game. We are limited in topic of media such as graphics and sounds that will be used. We will likely have to search for previously built libraries for this type of application so that other limitations mentioned like time, will not further bottleneck our project.

Risks:

We have developed a two lead project team management style that will take the load off of each of the Programming Lead and Project Lead respectively, but if one is absent for illness, drops the class, or has a heavy week of exams, the other lead will have to take the remainder responsibilities. These same dangers for the two leads are also significant for the other team members as well. Everyone's support and effort is greatly needed to accomplish this project as fairly and efficiently as possible.

Another significant risk is that we are developing the application in Java so it is possible that we could be months into the project and find that we are unable to implement some of our features with the remaining time we have left.

A third risk is that we should consider is that our application could run fine on the test machines but the final deliverable crashes on the grader's computers.

Alternatives and Tradeoffs:

- Geometry Wars
- Havoc – Alpha 42
- Asteroids
- Continuum or Subspace
- Danmakufu