CSC 330 Homework Assignment 2

Due on 10/27/2017 11:59pm

Requirements Elicitation

1. Asteroids Problem Statement

This is the Asteroids Problem Statement as provided by the client:

**Asteroids Problem Statement**

1. Problem

The computer market for space games is moving towards complex, photo-realistic 3D simulation games. The problems are long startup times and high learning curves for the player. Furthermore these games require a large amount of CPU time, which forces the players to upgrade their computer system to the hardware needs for these games. Solitaire, Minesweeper, or Moorhuhnjagd are examples for the success of simple and easy to use games. They have a high fun factor, no learning curve and short startup times. As the duration of a game is short, they can be played in nearly every situation. The goal of Asteroids is to provide a simple, fast and cheap space game that has a high fun factor. Players have to survive in a shuttle in a region in outer space full of asteroids. It should not require any special hardware.

1. Objectives

The objectives of the Asteroids game are to:

* provide a simple, fast and interactive space shuttle game with a high fun factor.
* be platform independent.
* require no extra administrative knowledge. The player should be able to do all administrative work (e.g. installing).

1. Scenarios

This section presents typical scenarios of system usage for illustration.

* 1. Actor Instances

**Actor Instance: Eve**

Eve is a video game player with no special computer knowledge. She likes space shuttle games and plays asteroids.

* 1. Scenarios

**Scenario: Configuring Asteroids**

Eve can configure the frame rate, the number of the initial asteroids, and the images used for the illustration of asteroids and the space shuttle. When configuration is finished, the game board reflects all changes immediately.

**Scenario: Losing and Asteroids game**

If any asteroid collides with the space shuttle, a ‘You lost!’ dialog window appears.

**Scenario: Playing an Asteroids game**

Eve initiates the game. All asteroids start to move in different directions with different speeds. The space shuttle, navigated by Eve, is located at the center and waits for instructions from Eve. An instrument panel displays the shuttle’s current speed and position. Eve can control the shuttle’s speed and direction. Shuttle and asteroids change their direction when they hit a game board boundary. The angle of incidence is the angle of reflection.

**Scenario: Starting the Asteroids application**

The Asteroids user interface displays a game board, which consists of a space shuttle and an arbitrary number of asteroids, a tool bar with game control buttons and an instrument panel. There are two different kinds of asteroids: small ones and big ones. Small asteroids move faster than big ones. All asteroids are computer controlled. The space shuttle starts in the middle of the game board.

**Scenario: Winning an Asteroids game**

Eve can fire rockets. When a rocket collides with a small asteroid, the asteroid explodes and disappears from the game board. When a rocket collides with a big asteroid, it explodes and many small asteroids appear instead of the big one. Eve wins the game when all asteroids are destroyed.

4. Requirements

The Asteroids player should be able to :

* Download and install the Asteroids game.
* run the game on multiple platforms.
* start the Asteroids application.
* stop the Asteroids application.
* start a game.
* learn how to play while playing the game, i.e. Asteroids must be simple to play: The user interface and the rules of playing Asteroids must be intuitive.
* play in real time. The player should see the result of a mouse click immediately. The  game must provide short response time. Even the startup time of the game must be low.
* fire rockets.
* observe the smooth movements of asteroids and the space shuttle.
* Collide the user-navigated space shuttle with an asteroid.
* see all the relevant information of the space shuttle during the asteroids game. This  includes all properties needed for navigating the shuttle (shuttle speed, position etc.).
* stop a game.
* change the direction of the navigated space shuttle.
* change the speed of the navigated space shuttle.
* configure the number of asteroids of each kind. There are two different kinds of  asteroids. One is small and fast and does not change its direction until it reaches the boundary of the game board. The other is big and slow, and it may change its direction any time. Both kinds of asteroids should be able to change their speed. They have a different minimum and maximum speed.
* customize the game. The game must be adaptable to both North American and Germany markets, i.e. it must support different metric systems, languages and imperial measurement systems.

5. Target environment

The Asteroids application must be built in Java using the Eclipse IDE. The game must run at least on MS Windows (2000 and above), Ubuntu Linux (7.04) and Mac OS X (10.4.1 and above).

***a) Functional Requirements***

Find all functional requirements and actors in the Asteroids Problem Statement above and represent them as a use case model.

* start the Asteroids application
  1. **Name:** Start the app
  2. **Actor:** User/Player
  3. **Entry:** User has the application installed
  4. **Exit:** The application opens successfully
  5. **Flow:** With the application installed the user clicks or presses enter, the application opens.
* stop the Asteroids application
  1. **Name:** Stop the app
  2. **Actor:** User/Player
  3. **Entry:** User has the application installed and opened
  4. **Exit:** The application closes successfully
  5. **Flow:** With the application opened the user clicks the close button; the application closes.
* start a game
  1. **Name:** Start game
  2. **Actor:** User/Player
  3. **Entry:** User has the application opened
  4. **Exit:** The game has begun
  5. **Flow:** With the application opened the user clicks the start button; the game begins.
* stop a game
  1. **Name:** Stop game
  2. **Actor:** User/Player
  3. **Entry:** User has the application opened and a game running
  4. **Exit:** The game stops running
  5. **Flow:** While a game is running the user clicks the pause button and then stop button; the game ends.
* shoot rockets
  1. **Name:** Shoot rockets
  2. **Actor:** User/Player
  3. **Entry:** User has started a game
  4. **Exit:** The spacecraft shoots rockets
  5. **Flow:** While a game is running the user clicks or presses a key to make the spaceship shoot a rocket.
* Collide the user-navigated space shuttle with an asteroid.
  1. **Name:** Asteroid collision
  2. **Actor:** User/Player
  3. **Entry:** User is playing a game and their spacecraft is too close to an asteroid
  4. **Exit:** The game ends
  5. **Flow:** While playing a game the user collides with an asteroid and the game ends.
* display in-game information
  1. **Name:** In-game display
  2. **Actor:** User/Player
  3. **Entry:** The user has started a game
  4. **Exit:** The user sees relevant information about the current gameplay
  5. **Flow:** With the application opened the user clicks the start button, the application displays relevant information about the current game.
* change the direction of the navigated space shuttle
  1. **Name:** Spaceship direction control
  2. **Actor:** User/Player
  3. **Entry:** User is playing a game and presses directional keys
  4. **Exit:** The spaceships moves in the direction of a corresponding key
  5. **Flow:** The user starts a game and presses keys for a direction to turn to, the spaceships begins to move in that direction.
* change the speed of the navigated space shuttle
  1. **Name:** Spaceship speed control
  2. **Actor:** User/Player
  3. **Entry:** User is playing a game and presses directional keys
  4. **Exit:** The spaceship accelerates or decelerates
  5. **Flow:** The user starts a game and presses the speed keys, the spaceships begins to move or slows.

***b) Non-functional Requirements***

Write down all non-functional and pseudo requirements in the Asteroids Problem Statement, categorized into Usability, Reliability, Performance, Supportability, Implementation, Interface, Operation, Packaging and Legal.

* + Usability
    - intuitive game dynamics
  + Reliability
    - no special hardware
  + Performance
    - fast startup
    - real time play
  + Supportability
    - Run on Windows (2000 and above)
    - Run on Ubuntu (7.04)
    - Run on MacOS (10.4.1 and above)
  + Implementation
    - Build with java
    - run on multiple platforms
  + Interface
    - customize spaceship
    - smooth graphics
    - customize asteroids
  + Operation
    - Built using Eclipse IDE
  + Packaging
    - Download and install the game
    - localization for NA and Germany
  + Legal

***c) Mapping***

For each non-functional requirement, find all relevant use cases from the problem statement.