Pacman Requirement Specification

Version 2

Loren Lugosch - 260404057
Ze Yu Yang - 260447989
Thaer Jleilati - 260449679
Christopher Peng - 260364364

Table of Contents

Introduction

Purpose and Overview

The project's main objective is to recreate the classic arcade game *Pacman*. The purpose of the project is to give the client a desktop version of *Pacman* and to help the development team learn about the practice of software engineering through the application of the theory learned in lectures. The development team will perform the same tasks that software engineers perform professionally such as planning, scheduling and testing.

The project will also require the developers to become familiar with a few important techniques in game design:
i. *Graphical user interface creation*. Since *Pacman* players need be able to see the maze and watch all the characters in the game, the system needs to have a GUI in place.
ii. *Game artificial intelligence algorithm implementation*. Enemies in *Pacman* don't simply move randomly; they need to navigate the environment using preprogrammed pursuit and path-finding algorithms.
iii. *User experience consideration*. Using the application should be fun, menus should be readily navigable, and the gameplay flow should be intuitive.

Definitions, Acronyms, and Abbreviations

Pacman - title of game.
Pacman - name of player-controlled character.
NPC - non-player character, or computer.
dot(s) - "food" represented by dots on the screen that Pacman must eat to clear the level.
power pellet(s) - when Pacman eats one, the ghosts temporarily turn blue and become edible ("Frightened" mode)
ghost(s) - NPC's that try to eat Pacman
cherry - special "food" Pacman that gives bonus points
crash - Application ceasing to function properly, often exiting due to errors
FPS - Frames per second
GUI - Graphical User Interface
UI - User Interface
AI - Artificial Intelligence
E - Essential
D - Desirable
O - Optional

References

a. Rules of *Pacman*

Controlling Pacman

1. Pacman and the ghosts spawn in the maze with initial positions.
2. The player chooses a direction using the up, down, left, and right arrow keys on the keyboard.
    i. Pacman begins moving in that direction. (Pacman will walk in that direction until the player picks a new direction, again using the arrow keys.)
    ii. The ghosts begin moving according to their own rules.
3. If Pacman attempts to move forward and a wall is in front of him, he stops until the player chooses a new direction.
4. If Pacman attempts to move to the left or the right and a wall is in the way, he "corners"- that is, he continues to move forward until there is a space he can move to in the direction he attempted.

Eating a collectible

1. Pacman "eats" collectibles when he moves over them.
2. Collectibles can take the form of dots (of which there are 240 per level), power pellets (4), and fruits (2).
    i. Eating a dot earns the player 10 points.
    ii. Eating a power pellet earns the player 50 points.
    iii. Eating a fruit earns the player 100 or more points. (The exact value depends on which level the player is on.)
3. When a collectible is eaten, it disappears from the maze.

Complete a level

If Pacman has collected all the dots, the level is complete, and the game loads the next level to play, where a "level" is a metric of difficulty (the speed of Pacman, the speed of the ghosts).

Eating a ghost

1. Ghosts enter "Frightened" mode (become edible) for 6 seconds or less (depending on what level is being played) when Pacman eats a power pellet.
2. When a ghost has been eaten, it is returned to the starting box and is reborn (no longer able to be eaten by Pacman until another power pellet has been eaten).

Die

1. If Pacman hits a ghost when the ghost is not in "Frightened" mode, Pacman dies.
2. If Pacman dies and has more than 0 lives, the level restarts with one less life and all the collectibles eaten or not eaten as they were when Pacman died.
3. If Pacman dies and has 0 lives, the game ends and the player returns to the main menu.

For more detailed information about the backend of *Pacman*, see "The Pac-Man Dossier"[1] by Jamey Pittman: http://home.comcast.net/~jpittman2/pacman/pacmandossier.html

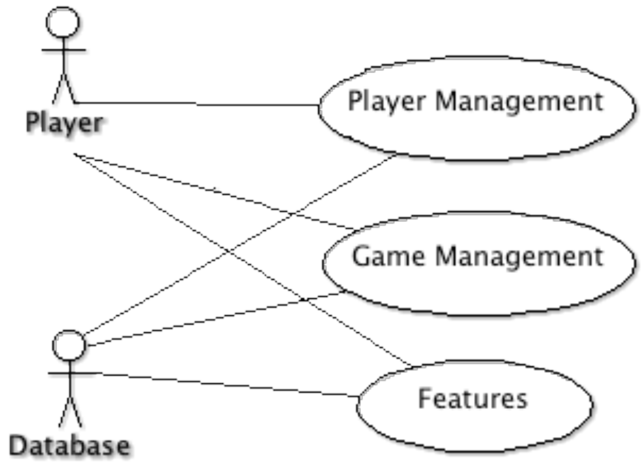
Project Requirements

High-level description

The main goal of the game *Pacman* is to collect all the little dots present in a maze. When all the dots are collected, the player advances to the next level. The player can move Pacman through the maze to pick up collectibles while avoiding the nefarious ghosts. The player can also collect power ups which grant him additional points and allow him to the eat the ghosts for a limited time period. (For more rules of the game, see “References”.)

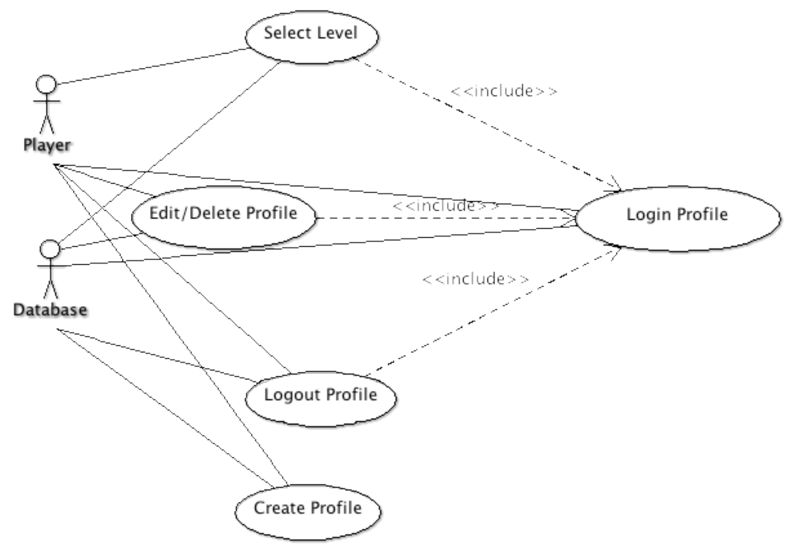
Much of *Pacman*’s functionality requires the system to keep track of some of the player’s information. If someone has reached a certain level, made a high score, or otherwise achieved something in the game, that person should be able to quit and return to gameplay later with that information being preserved. The external information storage and the player are the main actors to the system.

Use case overview



Main use cases

a. Player Management (E)



Create profile[1]

Participating actors: Non-logged-in player, local database
Flow of events:

1. A user enters all information to register his or her profile. This information includes a username, password (entered twice for verification) and email address (in case of password loss).
2. The system checks availability of the username. If the username has been taken an error message occurs and the user is prompted to enter another username.
3. The system checks to make sure the passwords provided match. If not, an error message occurs and the user is prompted to fill out both fields again.

Entry conditions: User selects “Create profile” from menu or when saving a score after playing a game.
Exit conditions: If the user cancels the account creation, the program will return to the home menu screen.
Quality requirements: n/a
Exceptions: n/a

Login profile[2] - When selecting a level or editing a profile, the user is prompted to login.

Participating actors: Non-logged-in player, local database

[1] Adapted from example use cases.
[2] Adapted from example use cases.

Flow of events:

1. The user enters Username and Password.
2. The user is prompted with the option to submit request or cancel
3.
    i. If option is to submit is chosen the entered username and password will be cross-checked in the database. If the password provided matches the password on file, the user is logged in and brought to the home screen. Otherwise the user is prompted for another username/password.
    ii. if option to cancel is chosen, the user is brought back to the main menu.

Entry conditions: User selects “Login” from main menu (or any of the use cases which require a login).
Exit conditions: User logs in or selects cancel.
Quality requirements: n/a
Exceptions: n/a

Logout profile

Participating actors: Logged-in player, local database
Flow of events:

1. From main menu, user selects “Logout”.
2. User is logged out, system returns to main menu with no profile selected.
3. If the user selects “cancel”, the system returns to main menu without logging out.

Entry conditions: The logged in user selects “logout” from the main menu.
Exit conditions: The user either logs out or cancels logout.
Quality requirements: n/a
Exceptions: n/a

Edit/Delete profile[3] - After logging in the user has the option to edit or delete their own profile.

Participating Actors: Logged in user, local database
Flow of events:

1. User is prompted for password. This is to identity check the user.
2. User is provided with a screen very similar to registration screen (with the “Username” field unavailable since usernames are not allowed to be changed). The user can choose to change Email Address or Password.
3.

[3] Adapted from example use cases.

i. The user can choose to “Save” where all new information is updated to database, or “Cancel” where all changes are discarded.
ii. The user can choose “Delete Account” option as well which will display message asking the user to confirm profile deletion.

Entry conditions: User chooses “Edit Account Info” from home menu.
Exit conditions: User saves edits, deletes account, or cancels
Quality requirements: n/a
Exceptions: n/a

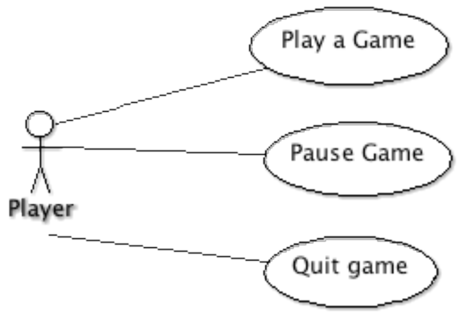
Select Level

Participating actors: Player, local database
Flow of events:

1. From the main menu, user selects “Select Level”.
2. The system prompts the user to login.
   i. If the user logs in, a menu of each of the levels accessible to the player (as well as the command for “exit”) opens.
      -If user selects a level, the player plays that level (starting with score 0).
      -If user selects exit, the system returns to the main menu.

Entry conditions: From the main menu, user selects “Select Level”.
Exit conditions: The user plays a level or exits to the main menu.
Quality requirements: n/a
Exceptions: n/a

b. Game Management (E)



Play a Game - a non-logged in user or a user who has logged in and selected a level starts a game.

Participating actors: Player
Flow of events:

1. A player starts a game at a certain level.
2. The player plays the game according to the rules of *Pacman* (see References section).
3. If the player wins the game, he/she advances to the next level.
4. If the player loses the game, the system prompts the player to upload his/her score, and he/she returns to the main menu.

Entry conditions: The user selects a level or starts from the main menu.
Exit conditions: The user quits the game or loses the game.
Quality requirements: n/a
Exceptions: n/a

Pause game

Participating actors: Player
Flow of events:

1. User enters command for “pause” (the command to pause should be displayed on the GUI at all times).
2. The game pauses, and an indicator displaying “PAUSED” and the command to unpause appears on the screen.
3. The game remains paused until the user enters the command for “unpause”.

Entry conditions: User enters command for “pause” (the command to pause should be displayed on the GUI at all times).
Exit conditions: The user enters the command for “unpause”.
Quality requirements: n/a
Exceptions: n/a

Quit game

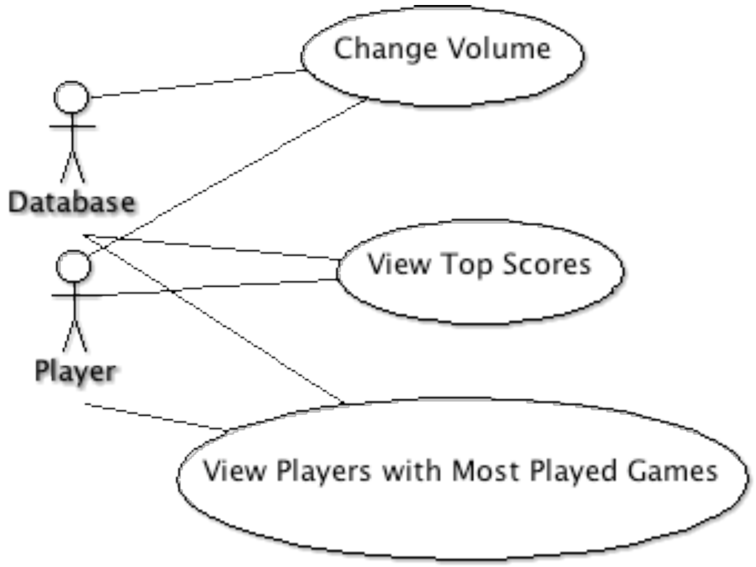
Participating actors: Player, local database
Flow of events:

1. User enters command for “quit.”
2. The system prompts the user to save his/her progress: the user selects yes or no.
    i. if yes, the player’s progress is saved, the system returns the user to the main menu.
    ii. if no, the system returns the user to the main menu.

Entry conditions: User enters command for “quit.”
Exit conditions: User selects “save” or “don’t save.”

Quality requirements: The command to quit should be displayed on the GUI at all times.
Exceptions: n/a

c. Features (E)



View Top Scores

Participating actors: Player, local database
Flow of events:

1. A list of the top 10 scores with their players' names appears (as well as the command for "exit").

Entry conditions: From the main menu, user selects "View Top Scores".
Exit conditions: User enters the command to exit.
Quality requirements: n/a
Exceptions: n/a

View players with most played games

Participating actors: Player, local database
Flow of events:

1. From the main menu, user selects "View Most Plays".
2. A list of the 10 players with the most played games appears (as well as the command for "exit").
3. Once the user enters the command for exit, the system returns to the main menu.

Entry conditions: From the main menu, user selects “View Most Plays”.
Exit conditions: User enters the command to exit.
Quality requirements: n/a
Exceptions: n/a

Change Volume

Participating actors: Player
Flow of events:

1. From the main menu, user selects “Settings”.
2. A menu with “Volume” opens (as well as the command for “exit”).
3. The user selects a volume.
4. The system returns to the main menu.

Entry conditions: From the main menu, user selects “Settings”.
Exit conditions: The user selects a volume or cancels.
Quality requirements: n/a
Exceptions: n/a

Specific Requirements

Non-functional or quality requirements

- The game’s frame rate is greater than or equal to 30 FPS for 90% of the time.
- The application crashes less than 1 in 5 times while running.
- 90% of users are able to access all menu features within 30 seconds.
- Players are able to distinguish between different visual game elements (player, ghost, wall, …) when asked about them 90% of the time.
- The game can be downloaded from the internet as a single archive file.
- The game is a desktop application that can be executed on a Windows 7 32/64 bit operating system with Java 7 installed.
- The game’s total size cannot exceed 10 MB of data.