Chicken Zombie Bonanza
High Level Design
COP 4331C Processes of Object Oriented Software Fall 2011

Team 19 - Team (Cauc)asians

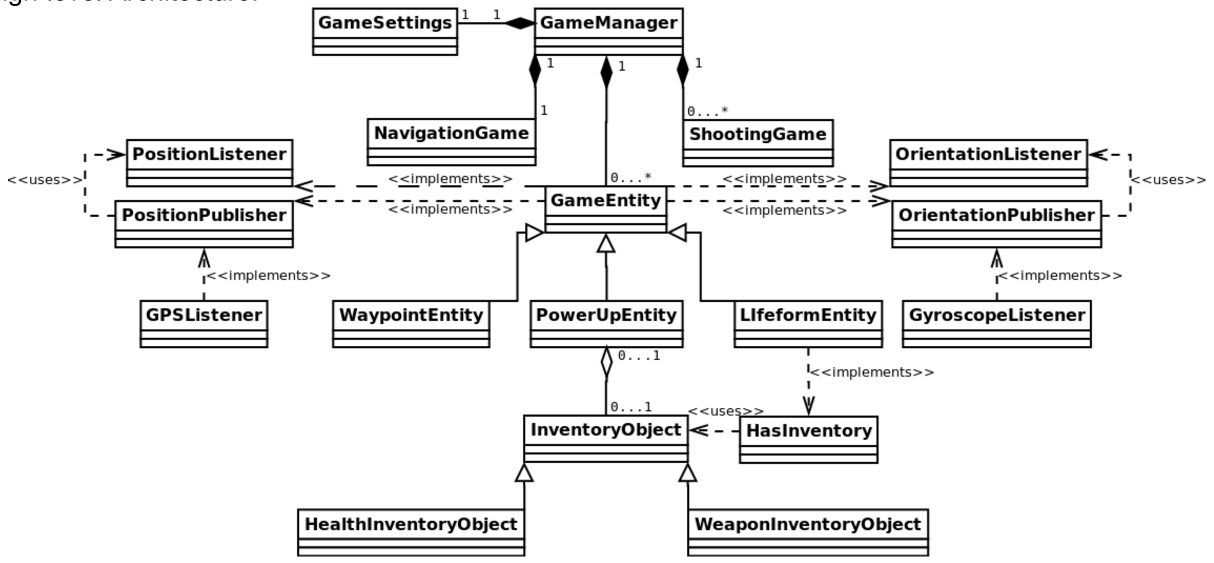
Team Members:
- Bernard Feeser
- Jon Leonard
- Danh Nguyen
- Jolene Wan
- Juan Chen

Modification history:

Version	Date	Who	Comment
v0.0	08/15/00	G. H. Walton	Template
v1.0	11/04/11	Jon Leonard	For Delieverables 2

Contents of this Document

High-level Architecture:



One aspect of the architecture is getting data from sensor sources like the GPS and gyroscope. Much like how the Android OS already handles sensor data, we will be utilizing the publish-subscribe paradigm to notify the system of sensor updates, this will allow infinitely many objects to get sensor updates. The more profound use of this paradigm will be used to notify the system of changes in the GameEntity, particularly changes in their position and orientation. For example, this will allow infinitely many objects keep track of the player's position, such as enemies and the game itself, without requiring direct access to the player's position data. Another detail we have architected the system is by implementing a high level of abstraction so many common parameters are not duplicated by what would be considered concrete classes, such as the GameEntity being a parent to all objects that are in the world as well as InventoryObject being the parent class of all objects that are stored in the inventory. By doing so we allow future modifications to the system to be minimal if another object that is a concrete example of an abstract class needs to be created, such as another type of InventoryObject, while the system may not know how to handle the object, it will be able to represent it. As such, the application is highly modular now which we feel is important for a video game so new content and enhancements can be added with no changes in the architecture.

The game manager creates and manages instances of the navigation game and shooting game. It will also create and manage the game entity's. A game entity is an abstraction of the objects in the world which currently includes the enemies, way points, power ups, and the player. Again, each game entity can publish their parameter updates to interested objects but also they are able to listen to other objects to get parameter updates for themselves, such as the player listening for GPS and gyroscope updates to keep the actual state of the player and the device one in the same. LifeformEntity is a concrete implementation of a game entity that represents an object in the world that has health and can die, this will include all enemies and the player. Aside from having life, lifeform entities also have a interesting property of having an inventory of items to use. The inventory items are obtained by activating power up entities, another concrete implementation of a game entity, which contain an inventory item. Lifeform entities can also interact with way points that will notify the system that the shooting game should be started if it is the player that is activating the waypoint. The shooting and navigation game will required to be listeners of the player’s position and orientation to properly display to the user the player’s location in the world and their correct orientation.

Design Issues

The primary concern of this project is that a mobile device game that requires the user to actively monitor and react to the virtual environment within the game will distract them from preforming actions in the real world to maintain their well being. To address this concern, and noting that we can only do so much in this aspect, we will display a disclaimer at the start of the application to warn the user of potential hazards that would arise from playing the game. Another feature we will incorporate that will help the safety of the user is that they will have the ability to generate another waypoint if the current one is in a dangerous area or requires the user to navigate through dangerous areas. We have also made a note of the portability of Java code and have designed the architecture to be easy enough to plug the game into other environments such as a stand alone web application.
