Design Pattern Paper COMP 2004 Group 3 Iteration 4

With the progress of creating the Malefiz board game we have ran into many situations in which GRASP patterns were utilized. The patterns in which were used for this project was information expert, low coupling, high cohesion, creator, controller, and observer. These patterns are used to create clear and efficient code that can be fixed in the future in an easier way.

The first pattern which was used is information expert. Information expert is a class in which has all the information and is responsible for doing a task. In the case of the Malefiz game, the information expert class was our game board class, this is because the game board interacts with the piece class and creates pieces and is its responsibility to place it in the correct places. This allows the piece class to be only focused on making pieces and letting other classes utilize it to place them like how the game board class did.

Along with information expert the next pattern which is used in low coupling. Low coupling is very important as it deals with how much each class is reliant on each other. This is used in our code throughout so that classes can be used with eachother without having to worry about what the classes internal implementation is. This allowed us to be able to change code within our project's classes with little concern since it would not change the implementation of other classes.

Thirdly, high cohesion was used as a pattern within the Malefiz project. High cohesion means classes only have information that is important to that class's success. For our project we tried to make it highly cohesive so that the code was more modularized and if any major changes needed to be done it would make it easier to locate. High cohesion and low coupling go together and is what we did as it is important for well written code.

Another pattern that we used within our code was controller. Controller is important for dealing with input events and was important for navigating our GUI. Controller was used within the game board class with the use of a menu bar to change menus and the game. The menu bar is used and is part of the controller pattern since it is used to navigate other GUIs created like our display settings and player settings menu. It is also vital for the interaction of the pieces and button to play the game.

A pattern which is very important to use is the observer pattern. This pattern is used to update the other objects on changes. This is important for the Malefiz game as there is always changes happening to the objects within the board and the other objects needs to be aware of these changes. This would make it to assure that pieces move correctly and that there are no bugs with pieces being able to move where they should not be able to.

Two patterns we could have used is the factory method pattern and the command pattern. The factory method pattern is used to create concrete subtypes for the client. This would have been a good idea to use to create the pieces for the number of players playing to have it isolated from the game code since the object creation does not matter to the game itself. Another pattern that also could have been used is the command pattern. This pattern is used to deal with user inputs and would have helped to separate the commands done by pieces and buttons from the actual code.

Using these patterns helped us create code for the Malefiz board game and change existing code. Along with information expert, low coupling, high cohesion, controller, and observer, two other patterns that we could have implemented is the factory method and the command pattern.