The use of patterns and what would to use

The first design pattern our group uses is command pattern. The reasons our group uses it are:

1. We need to create and execute requests at different times
2. We need to parameterize objects according to an action perform

In our design, after changing the role, the next player is going to set his piece on the board. So the function is invoked. However, the player could also rotate or flip before setting the piece. This produces the problem of creating and executing requests at different times. Hence, our group uses command pattern to solve this. The command pattern helps us separate the object that invokes the operation from the object that actually performs the operation. It also makes easy to add new commands, because existing classes remain unchanged.

Another design pattern our group uses is factory method pattern. When we write the code, sometimes we all feel confused about what sub-classes will be required to create. Such as the chess board. As a result, we use factory method pattern. On the one hand, it allows the sub-classes to choose the type of objects to create. On the other hand, it promotes the low-coupling by eliminating the need to bind application specific classes into the code.

If we have more time, our group is going to use observer pattern. Because sometimes we want to change a state in one object which must be reflected in another object without keeping the objects tight coupled. Using observer pattern can help us describes the coupling between the objects and the observer. And it can also provide the support for broadcast-type communication.