The first component of our system will be our **Fish Model** that handles a single game of Fish. This will enforce the rules of the game. There will also be a **Fish View** component that can visualize a game with a GUI.

Next we will have a **Game Runner** component that runs a game of Fish using the Fish Model with players over the internet. The Game Runner will send the game state to players who will respond with moves using a protocol we will define. It will also tell the players which color they are in this manner.The Game Runner will mainly focus on the communication between the players and the local game model, also handling player termination. We will also develop a **Manual Player** component that we can use to play the game as a player over the internet, so that we can test and demonstrate the Game Runner.

The **Tournament Manager** will create a tournament and pit players against each other. Given the players in the tournament, it will organize the tournament format of our choosing and can be kicked off to run the games and determine the winner(s). The games will be run with instances of the Game Runner. All tournament logic (bracket organization, handling multiple victors) will be handled here. The Tournament Manager might also handle notifying players of the tournament outcomes if necessary.

Finally, the **Sign-Up Server** will accept tournament sign-ups during the sign-up period (via a protocol we have defined) and will pass the player information to a Tournament Manager and kick off the tournament when the sign-up period has ended.

In terms of a standard “three-layer” architecture, most of our software fits in the application layer. Since the clients will likely communicate over TCP, there is no presentation layer required for the actual operation of the business. The only presentation layer software will be the view for the Fish Model, and any other views we choose to make for milestone visualization purposes. If we were using databases in the project (which would be likely for a large scale operation), we would store the player information in a data layer module, possibly along with the tournament data, depending on its size. Currently, we plan to store most of our data along with our software in the application layer.