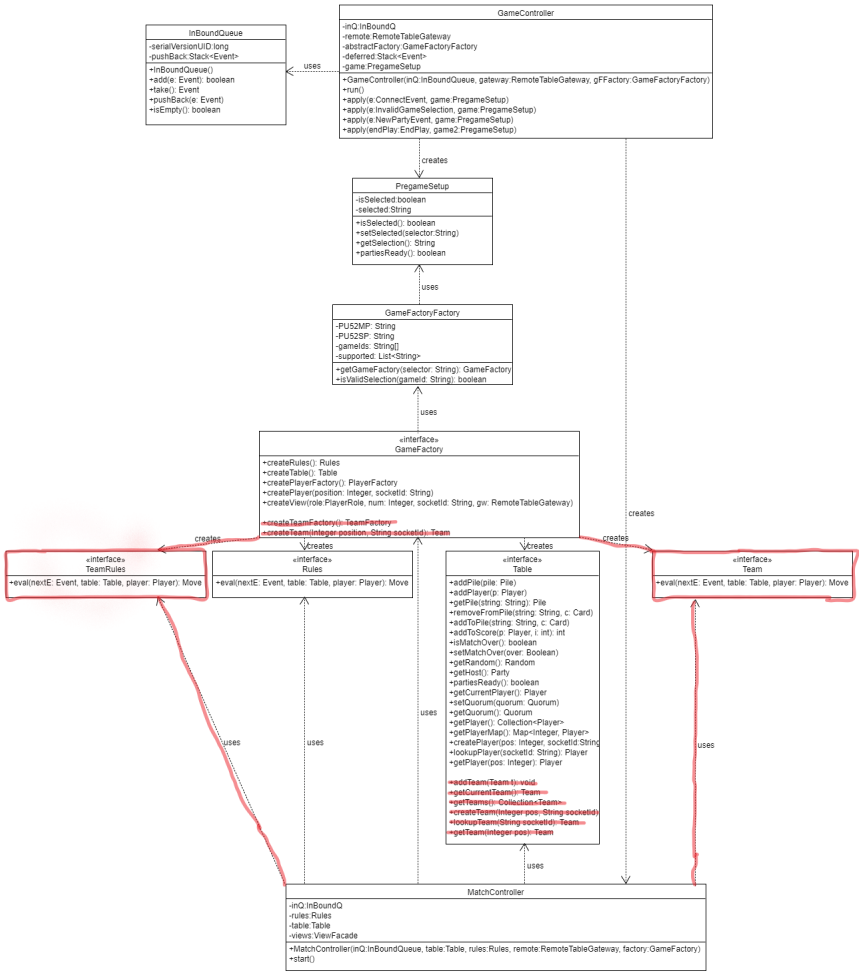
Report 1

COM S 362
Team 15: Jue Wang, Rundi Liu, Boxiang Guo, Caleb Donavon, Chuxiao Yu, Zachary Mass
Iteration 1: Controllers

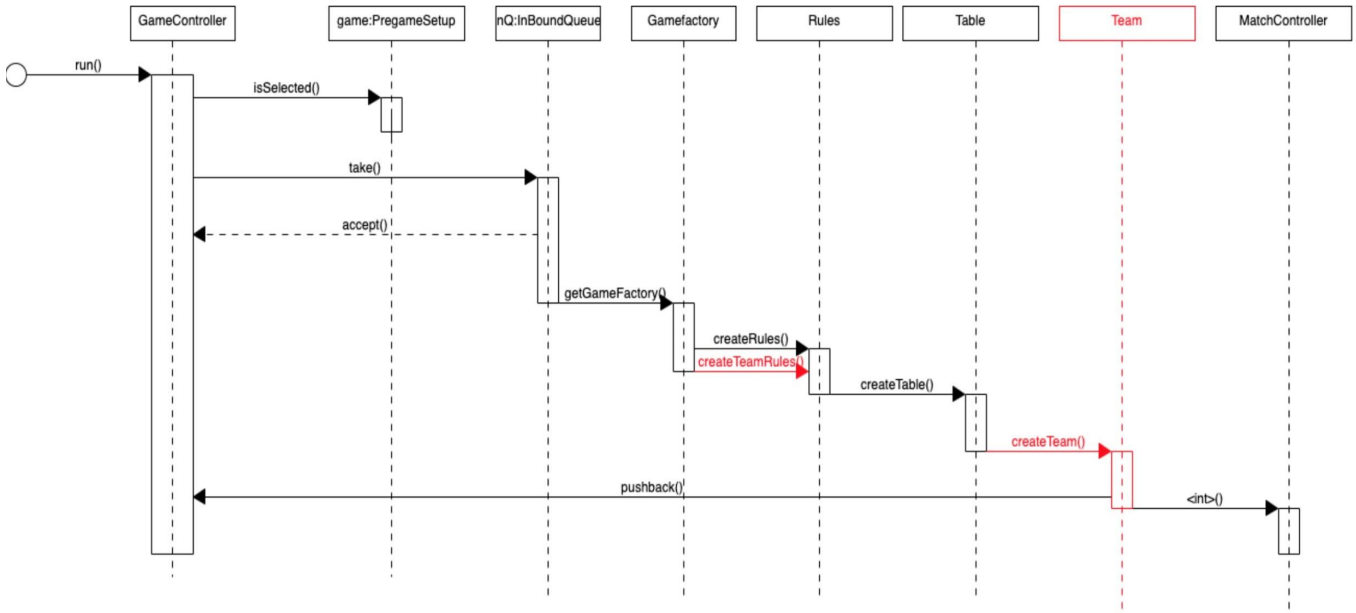
As a team, we chose to perform our impact analysis for support of team-based games. Overall, this change to the program would change the way that the ***GameController***, ***PlayController*** and ***MatchController*** classes function.

First, the following changes need to be made to the GameController:



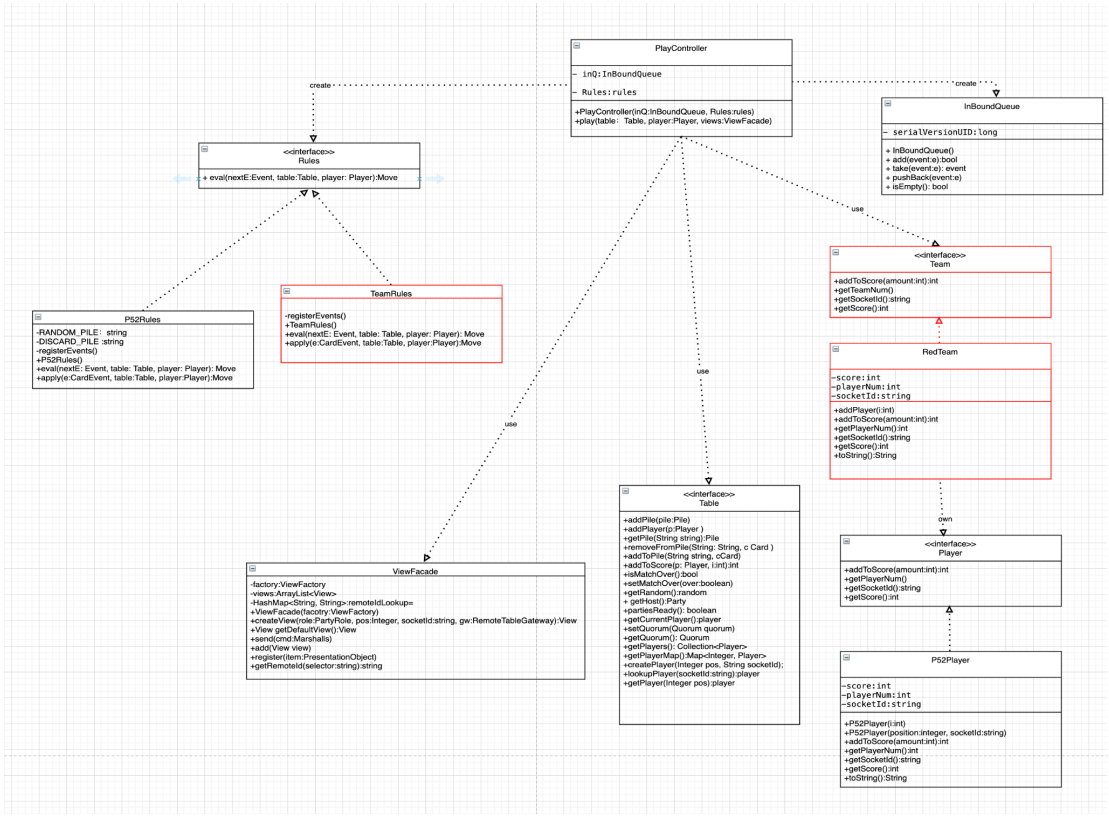
We added additional interfaces for TeamRules and Team that interact with the GameFactory class used by GameController. Further details are shown below.

GameController Sequence Diagram:



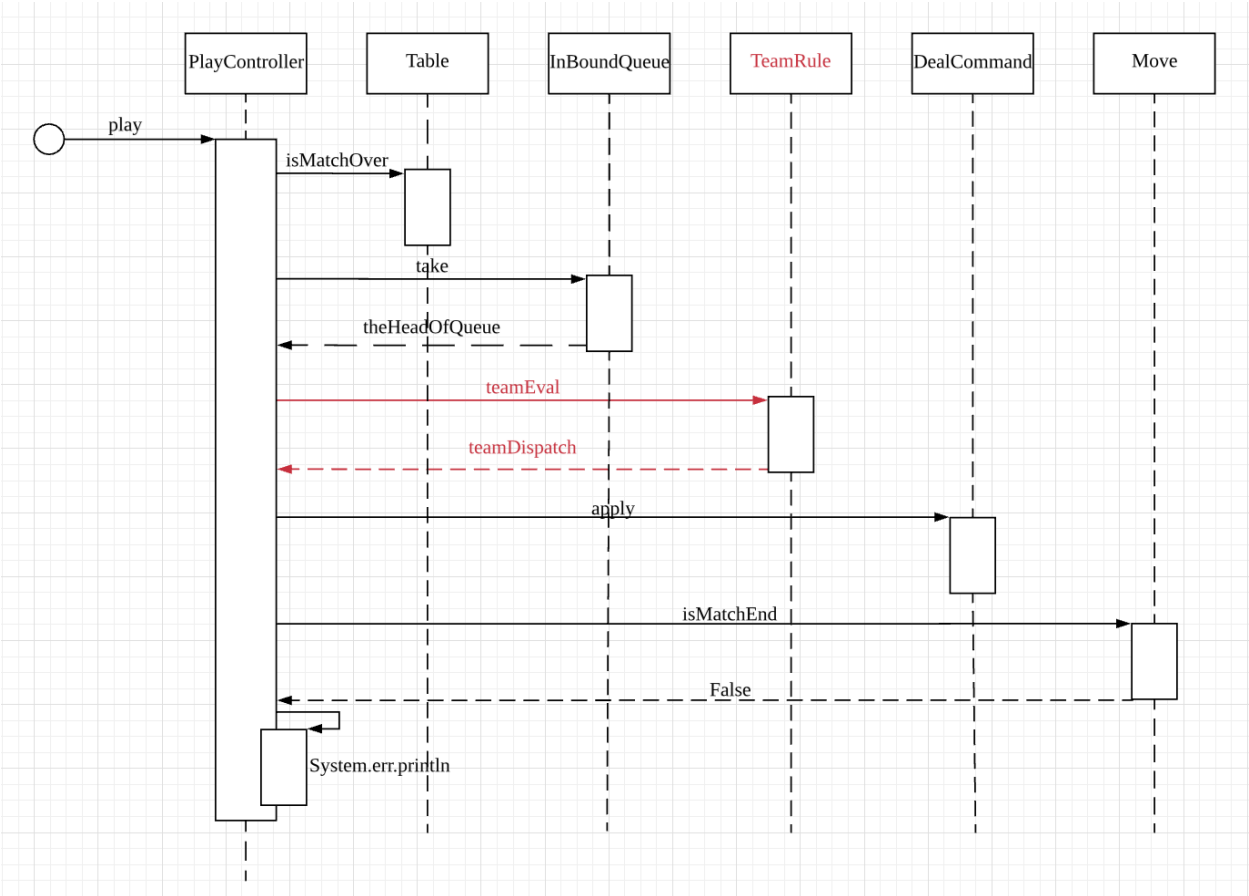
Based on what we did on the GameController sequence diagram before, our team added another game rule called TeamRule. In the meantime, we also have to create a new interface Team. The following two diagrams show the changes we did on PlayController.

PlayController Class Diagram:



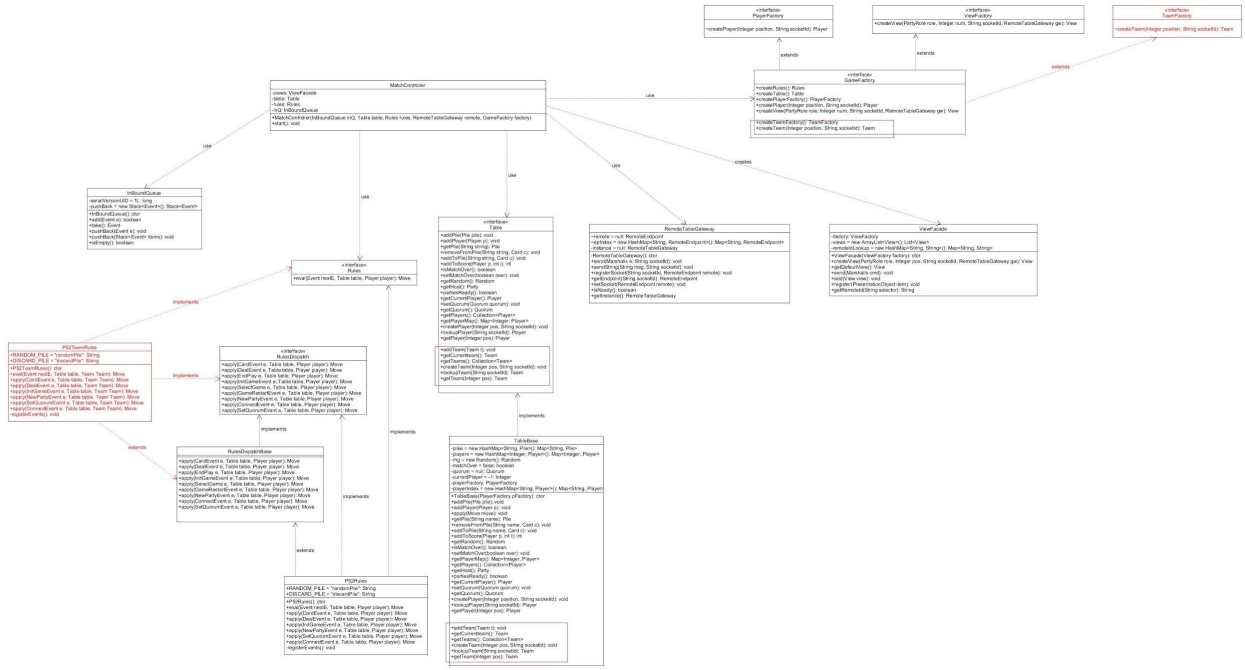
Add a team interface and a player is an extension of team interface. Which makes sense because every player has to be in a team to play the game. Each team can have one or more players.
Add team attributes to the player. The team interface has some functions like getScore() getTeamID().
Implemented TeamRules that the player has to follow.

PlayController Sequence Diagram:



Here we can see the playController will not call eval from P52Rules, because the original one(p52 rules) cannot satisfy our new features. Since we are going to support team based games, we should replace P52Rules with TeamRule which provides the different rules. And everything else is still working as before, we still need to check if the match is over and so on.

MatchController Class Diagram:

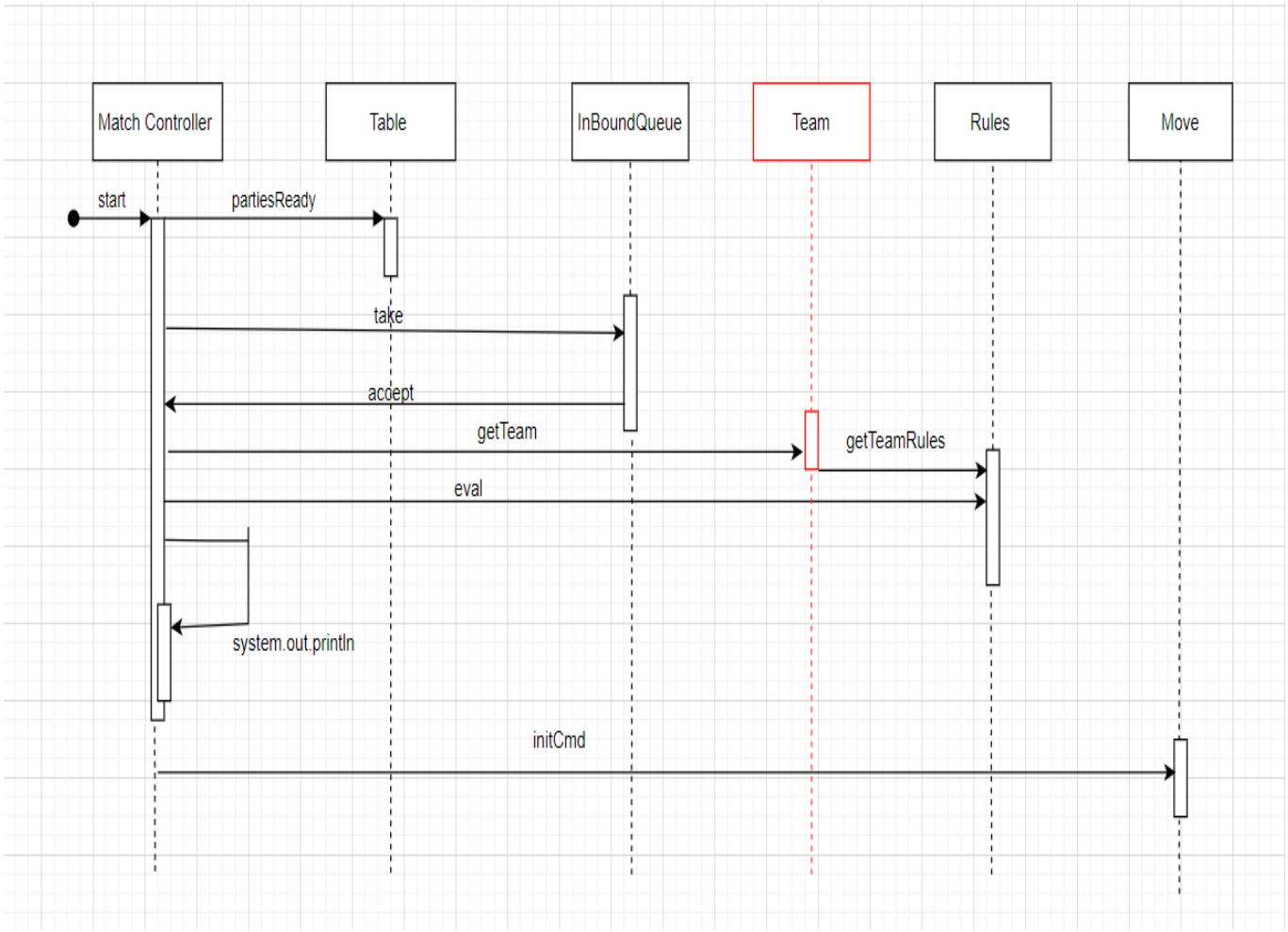


As the diagram shows, six methods will be added to Table interface:
+addTeam(Team t): void
+getCurrentTeam(): Team
+getTeams(): Collection<Team>
+createTeam(Integer pos, String socketId): void
+lookupTeam(String socketId): Team
+getTeam(Integer pos): Team
They basically act similar to methods for Player except that the object is changed to Team. This would support the team game table.These six methods will be implemented in TableBase.

A new interface TeamFactory will be added, and GameFactory interface will extends TeamFactory. Two methods will be added to GameFactory which act similar to createPlayerFactory() and createPlayer():
+createTeamFactory(): TeamFactory
+createTeam(Integer position, String socketId): Team

There will be a P52TeamRules class which implements Rules, RulesDispatch, and extends RulesDispatchBase to support the team game. It acts similar to P52Rules.

MatchController Sequence Diagram:



By adding on the support for the team-based games the interface team is added. In the case of the MatchController added to setting up the match is to get the teams created and the team rules for the certain match chosen. That can be seen in the red with the getTeam and getTeamRules with the Team interface.