**Requirement 4**

**Group F**

**Data structures:**

This group created two data types, one for turn management and another for updating the boards. The data types seem reasonable with the data they contain and how they are used in the system.

**Communications structures and interactions:**

Communication structure was the same as the original implementation, no new channels were added to the system.

Interactions between the processes were amended to include turns and board updating. The interactions were clear and they worked in the actual system, but they did not stray from the original design too much.

**Did the game work as intended:**

The turn system did work in general, since when a player got a non-matching pair and pressed the button, it moved to the next player and disabled the first players option to pick pairs as it isn’t their turn anymore. The board was also updated for each player when cards were flipped.

There was an issue with the game however, which was that if a player selects the same card twice it will act like he actually picked two different cards and will force him to finish his turn.

**Was the operation of the game intuitive and reflect the Challenge Requirements:**

The game did not stray from the original design so it was rather intuitive to use. This implementation only allowed one player to make a turn, while others waited for that player to flip a non-matching pair so they can have their turn. When a player got a matching pair, he was still allowed to continue his turn as was specified in the challenge.

**Group G**

**Data Structures:**

No new data structures were added in this approach.

**Communications structures and interactions:**

This teams approach added a new process to the system which manages the turns. It has three channels between this new process and the ControllerManager; these channels have their own purposes which makes the system much simpler and less prone to deadlock as long as these channels were implemented well enough. The new channel added in the player process also has a good function and it definitely makes sure no deadlocks will happen due to the turn system.

Since no interaction diagram was included with the document, no comment can be made on the interactions.

**Did the game work as intended:**

The game had a working turn system, however the board for a player was only updated after they made their turn which causes a deadlock in the system. This deadlock occurs because a player can claim a matching pair and then finish their turn after getting a non-matching pair. Since the next player has not had their board updated at this point, they can claim the pair that was already claimed by the previous player. The system then deadlocks as this pair was already claimed and doesn’t exist anymore.

Withdrawing from the game breaks the turn system, deadlocking the system so this function of the game is unusable.

**Was the operation of the game intuitive and reflect the Challenge Requirements:**

The game operation was intuitive as it worked in the same manner as the original implementation. It has a working turn system that behaves in a manner that was specified in the challenge requirements.