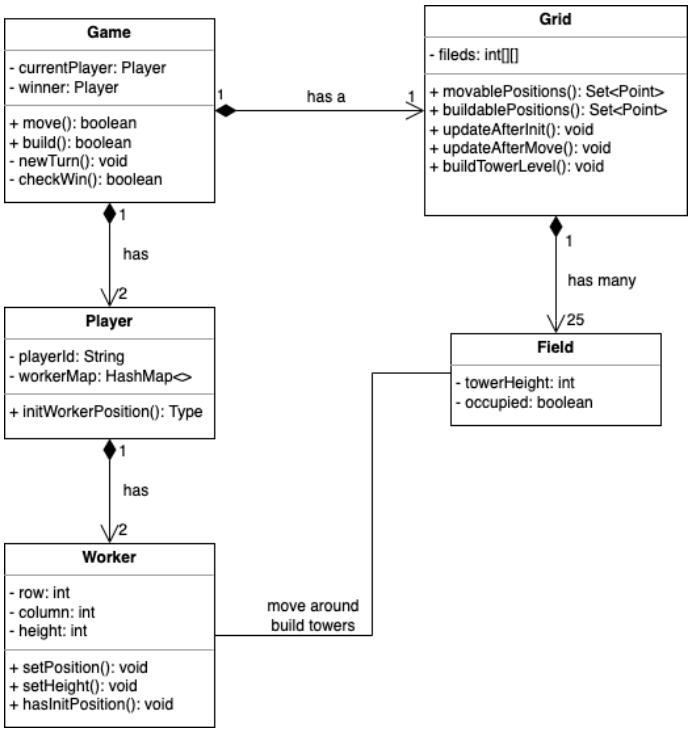
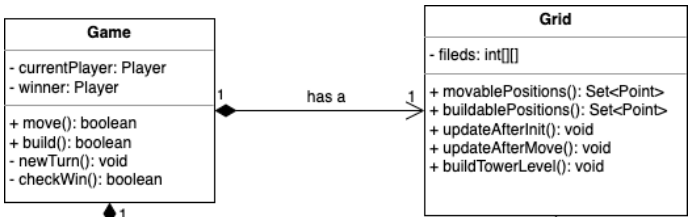
1. Player can choose which worker to move in his/her turn, decide which unoccupied adjacent field the worker moves to, and which unoccupied filed to build a tower level around worker's new position. Move and build are two main possible actions. For the controller design heuristic, Game class coordinates a system operation and delegates jobs to Grid class like getting movable positions, getting buildable positions and updating grid status after moving and building. Façade design pattern is applied. In this design pattern, several classes are coupled to the controller, which serves as a mediator, but this coupling is less harmful. And it supports reuse and evolvability because the controller serves as in interface to the domain logic and can be split into smaller interfaces. This design pattern increases coupling, but increases cohesion, which enhances understandability and reuse.



2. Game needs to store workers' positions, tower status on every field of the grid, the current player and whether the game has winner. Current player and winner should be stored in Game class. Fields' status and worker positions should be stored in Grid class. For information expert design heuristic, we should assign responsibilities to the class that has the information necessary to fulfill the responsibility. The Game class coordinates the system, it should keep record of the current player and check if the game has produced the winner and should end. The Grid class has all fields' status information, so it should store and update filed information after every action. This is design for understandability, division for labor and reuse. In this way, there are fewer dependencies to understand and it's easier to reuse without complicated dependencies.



3. Game should firstly check if the target position is an adjacent filed of the worker's new position after its moving. Then it should check that the field had no worker on it and there's no dome built on the tower of the field. Game builds tower a level higher through increasing the towerHeight of the field. If towerHeight before building is less than three, worker builds a normal block; if towerHeight is three, worker builds a dome on the tower. After building, if towerHeight reaches four, which means a dome has been built, game marks this field as occupied and no worker can be moved to here and no tower can be built on this field. The Game class is a controller coordinating the system, so it has the build function to coordinate between player, worker and grid. For the design goal of division of labor, if delegates jobs of retrieving valid adjacent fields and updating fields status after action to the Grid class, which has information of all fields. Also, Game class gets information about worker form player and sends it to Grid, which decreases coupling between player, worker and grid.