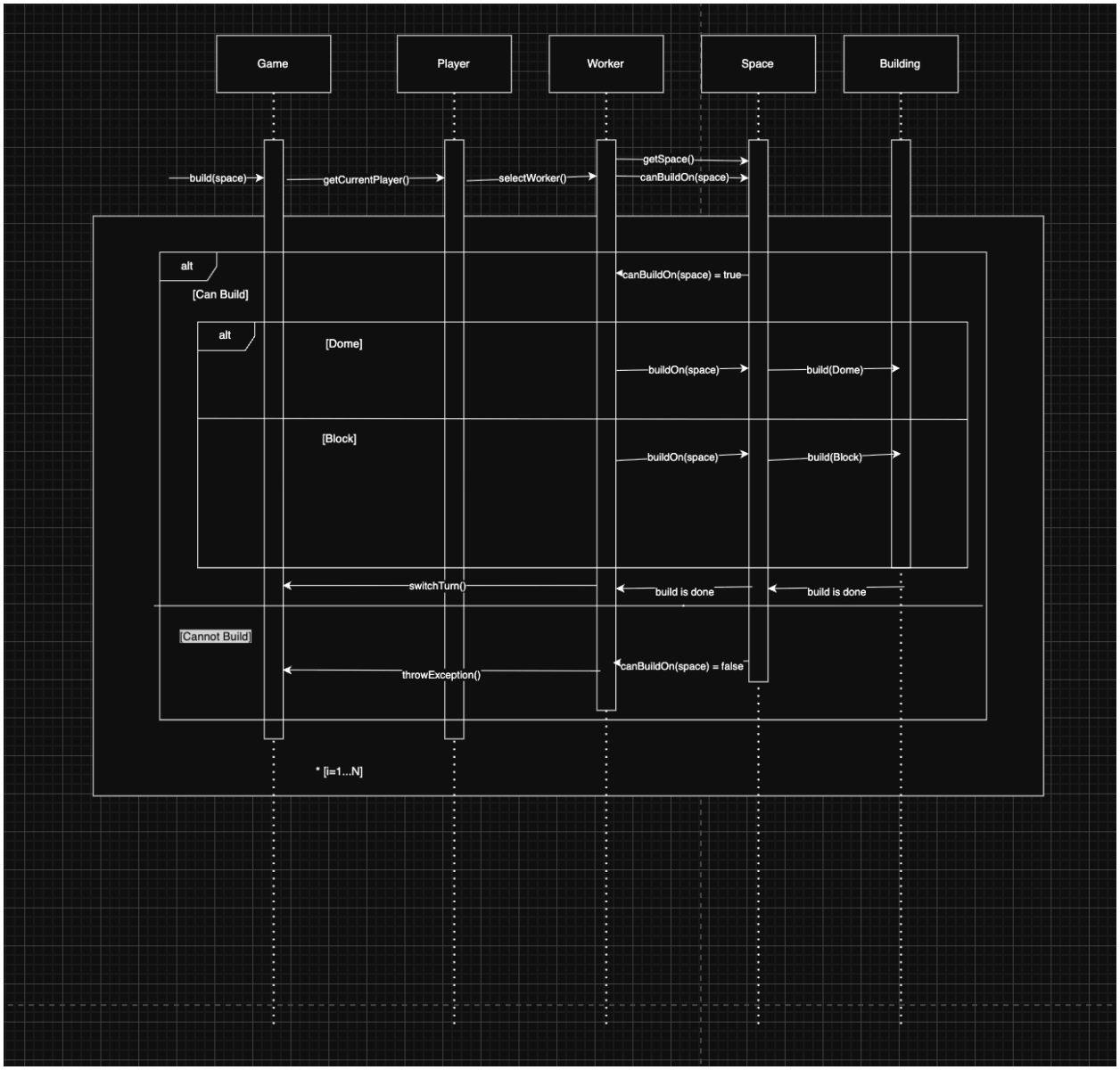

Information Expert: Worker.selectWorker() and Space.canBuildOn() are placed in classes with the relevant data and logic to perform their function. The Space has the necessary information and can access its own building's level in order to check if it can be built on.

Low Coupling: Building objects handle the specifics of constructing various structures (build(Dome), build(Block)).

High Cohesion: Space's role is to validate builds while Building executes them, which separates the action of building into two classes, preventing a god class situation.

Overall, the responsibility for handling game logic, player interactions, worker selection, and construction validation is cleanly divided across classes, so game rule checks aren't intermixed with the state management.

Alternatives Considered:

Originally, I thought of having Building notify Space, Space notifies Worker, Worker notifies Player, and Player informs Game to switch turns. This mirrors real-life gameplay, but creates a long chain and blurs responsibility boundaries. Each class relays a message without adding logic, and it is unclear why every step is needed.

Instead, now the Game class handles the entirety of the Turn logic—and keeps track of the currentPlayer and the selected Worker, which will automatically notify it if certain actions throw errors. If not, it can safely switch the turn.