The Deadwood class diagram was designed with modularity and separation of concerns in mind. Each class has a clear responsibility while staying loosely coupled to the others. The Deadwood class acts as the main controller, handling the game loop, player turns, and overall flow, while delegating specific mechanics to helper classes. Board manages all spatial and scene data, including rooms, adjacency, and active scenes, making it the core of the game's physical state. Actions handles all player operations—moving, acting, rehearsing, taking roles, and upgrading—so gameplay logic stays separate from turn management and UI. This structure makes the system easier to debug, test, and extend since changes to one mechanic don't affect unrelated parts of the code.

The class relationships reflect the structure of the game rules. Composition is used where objects depend on their parent, such as Board containing Rooms, and Rooms containing extra Roles. If a Room is removed, its Roles go with it. Similarly, a Scene contains its starring Roles. Association represents more flexible relationships, like a Player being linked to a Room or Role during gameplay. The Room self-association models adjacency cleanly without needing a separate graph. Overall, the design supports required features like day tracking and scene wrapping while keeping the architecture clean, scalable, and aligned with the board game's logic.