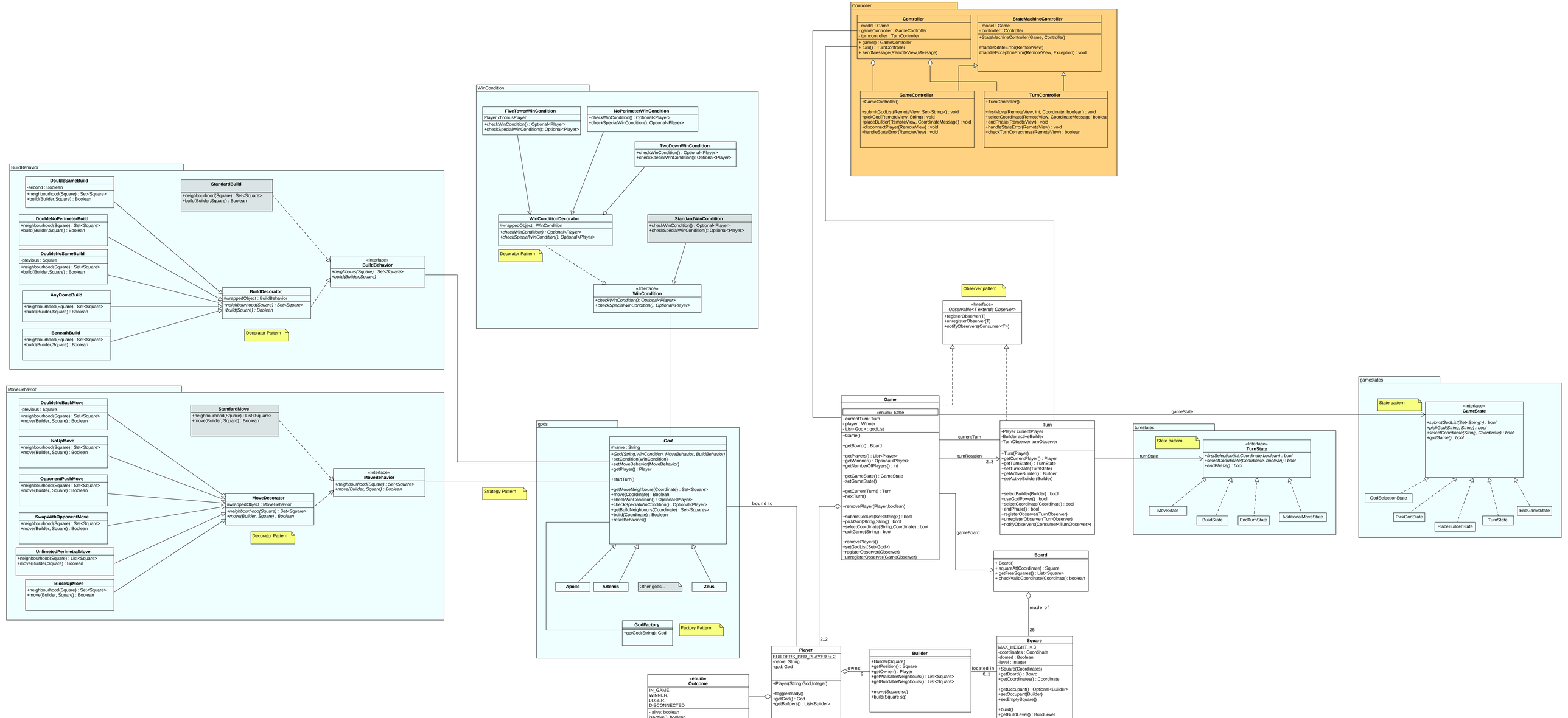
Controller
Controller
- model : Game
- gameController : GameController
- turncontroller : TurnController
+ game() : GameController
+ turn() : TurnController
+ sendMessage(RemoteView,Message)
StateMachineController
- model : Game
- controller : Controller
+StateMachineController(Game, Controller)
#handleStateError(RemoteView)
#handleExceptionError(RemoteView, Exception) : void
GameController
+GameController()
+submitGodList(RemoteView, Set<String>) : void
+pickGod(RemoteView, String) : void
+placeBuilder(RemoteView, CoordinateMessage) : void
+disconnectPlayer(RemoteView) : void
+handleStateError(RemoteView) : void
TurnController
+TurnController()
+firstMove(RemoteView, int, Coordinate, boolean) : void
+selectCoordinate(RemoteView, CoordinateMessage, boolean
+endPhase(RemoteView) : void
+handleStateError(RemoteView) : void
+checkTurnCorrectness(RemoteView) : boolean
WinCondition
FiveTowerWinCondition
Player chronusPlayer
+checkWinCondition() : Optional<Player>
+checkSpecialWinCondition(): Optional<Player>
NoPerimeterWinCondition
+checkWinCondition() : Optional<Player>
+checkSpecialWinCondition(): Optional<Player>
TwoDownWinCondition
+checkWinCondition() : Optional<Player>
+checkSpecialWinCondition(): Optional<Player>
WinConditionDecorator
#wrappedObject : WinCondition
+checkWinCondition() : Optional<Player>
+checkSpecialWinCondition(): Optional<Player>
Decorator Pattern
StandardWinCondition
+checkWinCondition() : Optional<Player>
+checkSpecialWinCondition(): Optional<Player>
«Interface»
WinCondition
+checkWinCondition(): Optional<Player>
+checkSpecialWinCondition(): Optional<Player>
BuildBehavior
DoubleSameBuild
-second : Boolean
+neighbourhood(Square) : Set<Square>
+build(Builder,Square) : Boolean
StandardBuild
+neighbourhood(Square) : Set<Square>
+build(Builder,Square) : Boolean
DoubleNoPerimeterBuild
+neighbourhood(Square) : Set<Square>
+build(Builder,Square) : Boolean
DoubleNoSameBuild
-previous : Square
+neighbourhood(Square) : Set<Square>
+build(Builder,Square) : Boolean
«Interface»
BuildBehavior
+neighbours(Square) : Set<Square>
+build(Builder,Square)
AnyDomeBuild
+neighbourhood(Square) : Set<Square>
+build(Builder,Square) : Boolean
BuildDecorator
#wrappedObject : BuildBehavior
+neighbourhood(Square) : Set<Square>
+build(Square) : Boolean
Decorator Pattern
BeneathBuild
+neighbourhood(Square) : Set<Square>
+build(Builder,Square) : Boolean
MoveBehavior
DoubleNoBackMove
-previous : Square
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
StandardMove
+neighbourhood(Square) : List<Square>
+move(Builder, Square) : Boolean
NoUpMove
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
OpponentPushMove
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
SwapWithOpponentMove
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
UnlimetedPerimetralMove
+neighbourhood(Square) : List<Square>
+move(Builder,Square) : Boolean
BlockUpMove
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
«Interface»
MoveBehavior
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
MoveDecorator
#wrappedObject : MoveBehavior
+neighbourhood(Square) : Set<Square>
+move(Builder, Square) : Boolean
Decorator Pattern
Strategy Pattern
gods
God
#name : String
+God(String,WinCondition, MoveBehavior, BuildBehavior)
+setCondition(WinCondition)
+setMoveBehavior(MoveBehavior)
+getPlayer() : Player
+startTurn()
+getMoveNeighbours(Coordinate) : Set<Square>
+move(Coordinate) : Boolean
+checkWinCondition() : Optional<Player>
+checkSpecialWinCondition() : Optional<Player>
+getBuildNeighbours(Coordinate) : Set<Squares>
+build(Coordinate) : Boolean
+resetBehaviors()
Apollo
Artemis
Other gods...
Zeus
GodFactory
+getGod(String): God
Factory Pattern
bound to
Observer pattern
«Interface»
Observable<T extends Observer>
+registerObserver(T)
+unregisterObserver(T)
+notifyObservers(Consumer<T>)
Game
«enum» State
- currentTurn: Turn
- player : Winner
- List<God> : godList
+Game()
+getBoard() : Board
+getPlayers() : List<Player>
+getWinnner() : Optional<Player>
+getNumberOfPlayers() : int
+getGameState() : GameState
+setGameState()
+getCurrentTurn() : Turn
+nextTurn()
+removePlayer(Player,boolean)
+submitGodList(Set<String>) : bool
+pickGod(String,String) : bool
+selectCoordinate(String,Coordinate) : bool
+quitGame(String) : bool
+removePlayers()
+setGodList(Set<God>)
+registerObserver(Observer)
+unregisterObserver(GameObserver)
currentTurn
turnRotation
2..3
gameBoard
Turn
-Player currentPlayer
-Builder activeBuilder
-TurnObserver turnObserver
+Turn(Player)
+getCurrentPlayer() : Player
+getTurnState() : TurnState
+setTurnState(TurnState)
+getActiveBuilder() : Builder
+setActiveBuilder(Builder)
+selectBuilder(Builder) : bool
+useGodPower() : bool
+selectCoordinate(Coordinate) : bool
+endPhase() : bool
+registerObserver(TurnObserver)
+unregisterObserver(TurnObserver)
+notifyObservers(Consumer<TurnObserver>)
gameState
turnstates
State pattern
«Interface»
TurnState
+firstSelection(int,Coordinate,boolean) : bool
+selectCoordinate(Coordinate, boolean) : bool
+endPhase() : bool
turnState
MoveState
BuildState
EndTurnState
AdditionalMoveState
gamestates
State pattern
«Interface»
GameState
+submitGodList(Set<String>) : bool
+pickGod(String, String) : bool
+selectCoordinate(String, Coordinate) : bool
+quitGame() : bool
GodSelectionState
PickGodState
PlaceBuilderState
TurnState
EndGameState
Board
+ Board()
+ squareAt(Coordinate) : Square
+ getFreeSquares() : List<Square>
+ checkValidCoordinate(Coordinate): boolean
made of
25
Square
MAX_HEIGHT := 3
-coordinates : Coordinate
-domed : Boolean
-level : Integer
+Square(Coordinates)
+getBoard() : Board
+getCoordinates() : Coordinate
+getOccupant() : Optional<Builder>
+setOccupant(Builder)
+setEmptySquare()
+build()
+getBuildLevel() : BuildLevel
located in
0..1
Builder
+Builder(Square)
+getPosition() : Square
+getOwner() : Player
+getWalkableNeighbours() : List<Square>
+getBuildableNeighbours() : List<Square>
+move(Square sq)
+build(Square sq)
owns
2
Player
BUILDERS_PER_PLAYER := 2
-name: String
-god: God
+Player(String,God,Integer)
+toggleReady()
+getGod() : God
+getBuilders() : List<Builder>
2..3
«enum»
Outcome
IN_GAME,
WINNER,
LOSER,
DISCONNECTED
- alive: boolean
isActive(): boolean