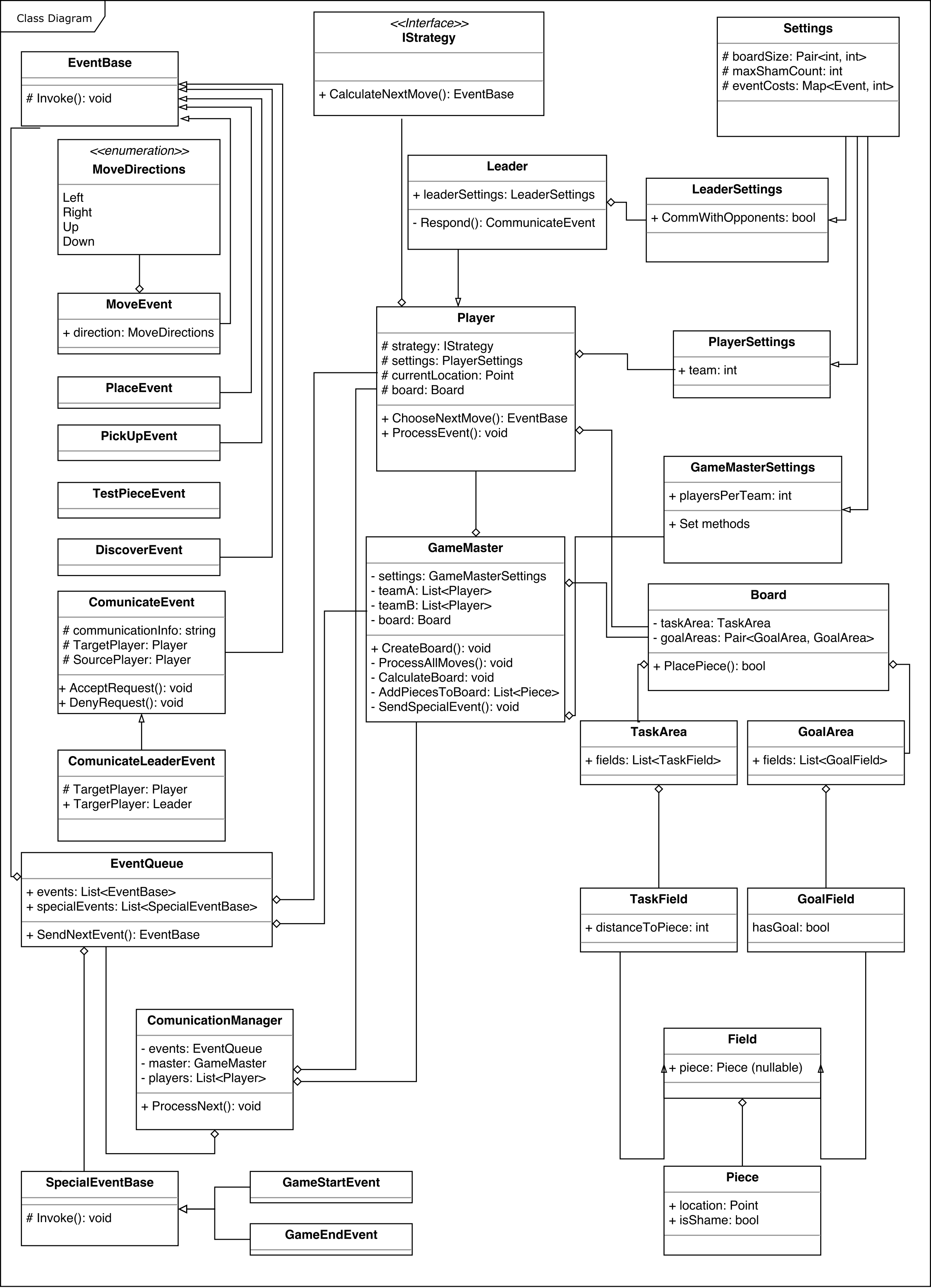
Class Diagram
EventBase
Invoke(): void
<<Interface>>
IStrategy
+ CalculateNextMove(): EventBase
Settings
boardSize: Pair<int, int>
maxShamCount: int
eventCosts: Map<Event, int>
<<enumeration>>
MoveDirections
Left
Right
Up
Down
Leader
+ leaderSettings: LeaderSettings
- Respond(): CommunicateEvent
LeaderSettings
+ CommWithOpponents: bool
MoveEvent
+ direction: MoveDirections
Player
strategy: IStrategy
settings: PlayerSettings
currentLocation: Point
board: Board
+ ChooseNextMove(): EventBase
+ ProcessEvent(): void
PlayerSettings
+ team: int
PlaceEvent
PickUpEvent
TestPieceEvent
DiscoverEvent
GameMasterSettings
+ playersPerTeam: int
+ Set methods
GameMaster
- settings: GameMasterSettings
- teamA: List<Player>
- teamB: List<Player>
- board: Board
+ CreateBoard(): void
- ProcessAllMoves(): void
- CalculateBoard: void
- AddPiecesToBoard: List<Piece>
- SendSpecialEvent(): void
Board
- taskArea: TaskArea
- goalAreas: Pair<GoalArea, GoalArea>
+ PlacePiece(): bool
ComunicateEvent
communicationInfo: string
TargetPlayer: Player
SourcePlayer: Player
+ AcceptRequest(): void
+ DenyRequest(): void
TaskArea
+ fields: List<TaskField>
GoalArea
+ fields: List<GoalField>
ComunicateLeaderEvent
TargetPlayer: Player
+ TargerPlayer: Leader
EventQueue
+ events: List<EventBase>
+ specialEvents: List<SpecialEventBase>
+ SendNextEvent(): EventBase
TaskField
+ distanceToPiece: int
GoalField
hasGoal: bool
ComunicationManager
- events: EventQueue
- master: GameMaster
- players: List<Player>
+ ProcessNext(): void
Field
+ piece: Piece (nullable)
SpecialEventBase
Invoke(): void
GameStartEvent
GameEndEvent
Piece
+ location: Point
+ isShame: bool