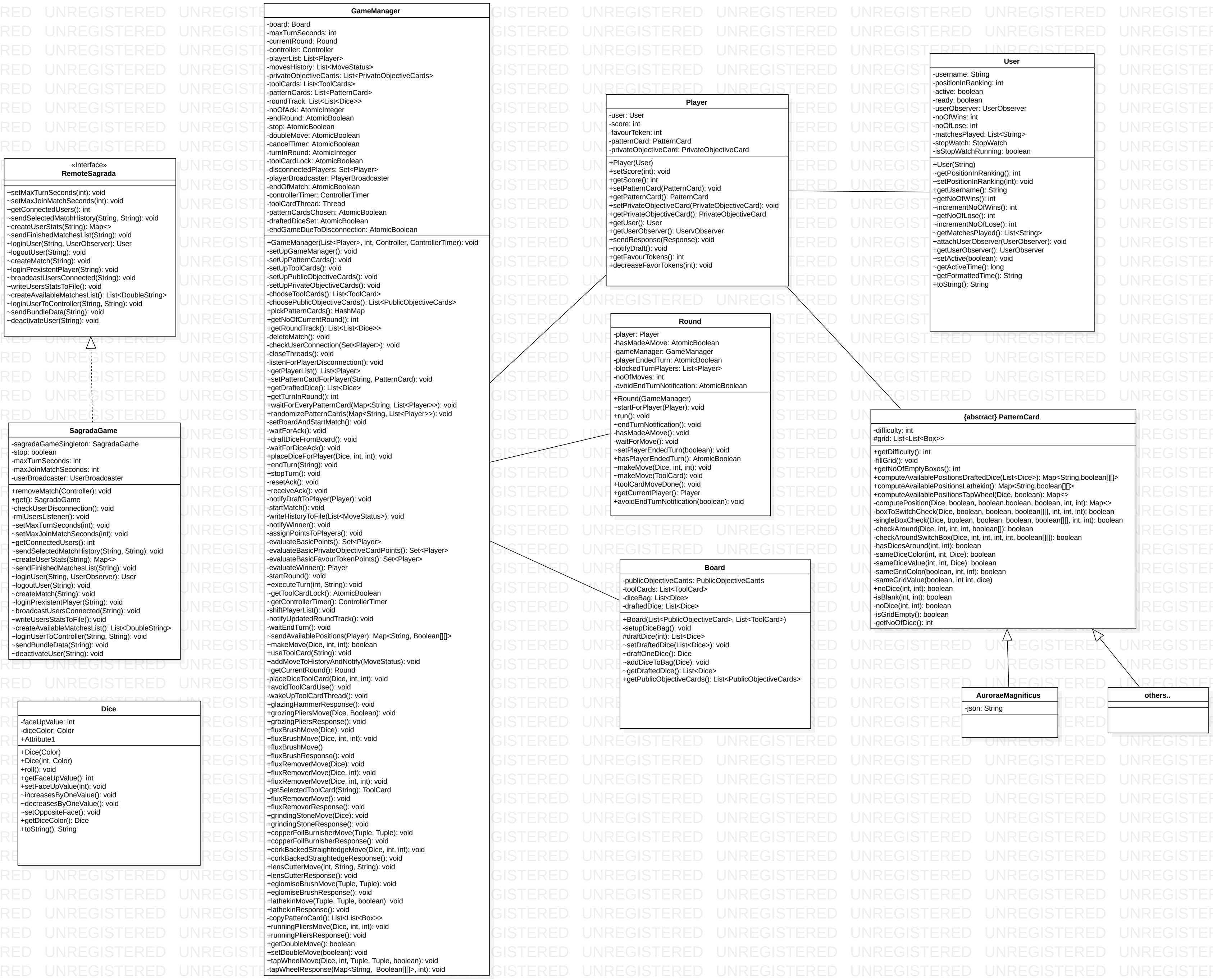
Model::ClassDiagram1



```
«Interface»
RemoteSagrada

~setMaxTurnSeconds(int): void
~setMaxJoinMatchSeconds(int): void
~getConnectedUsers(): int
~sendSelectedMatchHistory(String, String): void
~createUserStats(String): Map<>
~sendFinishedMatchesList(String): void
~loginUser(String, UserObserver): User
~logoutUser(String): void
~createMatch(String): void
~loginPrexistentPlayer(String): void
~broadcastUsersConnected(String): void
~writeUsersStatsToFile(): void
~createAvailableMatchesList(): List<DoubleString>
~loginUserToController(String, String): void
~sendBundleData(String): void
~deactivateUser(String): void
```

```
SagradaGame

-sagradaGameSingleton: SagradaGame
-stop: boolean
-maxTurnSeconds: int
-maxJoinMatchSeconds: int
-userBroadcaster: UserBroadcaster

+removeMatch(Controller): void
+get(): SagradaGame
-checkUserDisconnection(): void
-rmiUsersListener(): void
~setMaxTurnSeconds(int): void
~setMaxJoinMatchSeconds(int): void
~getConnectedUsers(): int
~sendSelectedMatchHistory(String, String): void
~createUserStats(String): Map<>
~sendFinishedMatchesList(String): void
~loginUser(String, UserObserver): User
~logoutUser(String): void
~createMatch(String): void
~loginPrexistentPlayer(String): void
~broadcastUsersConnected(String): void
~writeUsersStatsToFile(): void
~createAvailableMatchesList(): List<DoubleString>
~loginUserToController(String, String): void
~sendBundleData(String): void
~deactivateUser(String): void
```

```
Dice

-faceUpValue: int
-diceColor: Color
+Attribute1

+Dice(Color)
+Dice(int, Color)
+roll(): void
+getFaceUpValue(): int
+setFaceUpValue(int): void
~increasesByOneValue(): void
~decreasesByOneValue(): void
~setOppositeFace(): void
+getDiceColor(): Dice
+toString(): String
```

```
GameManager

-board: Board
-maxTurnSeconds: int
-currentRound: Round
-controller: Controller
-playerList: List<Player>
-movesHistory: List<MoveStatus>
-privateObjectiveCards: List<PrivateObjectiveCards>
-toolCards: List<ToolCards>
-patternCards: List<PatternCard>
-roundTrack: List<List<Dice>>
-noOfAck: AtomicInteger
-endRound: AtomicBoolean
-stop: AtomicBoolean
-doubleMove: AtomicBoolean
-cancelTimer: AtomicBoolean
-turnInRound: AtomicInteger
-toolCardLock: AtomicBoolean
-disconnectedPlayers: Set<Player>
-playerBroadcaster: PlayerBroadcaster
-endOfMatch: AtomicBoolean
-controllerTimer: ControllerTimer
-toolCardThread: Thread
-patternCardsChosen: AtomicBoolean
-draftedDiceSet: AtomicBoolean
-endGameDueToDisconnection: AtomicBoolean

+GameManager(List<Player>, int, Controller, ControllerTimer): void
-setUpGameManager(): void
-setUpPatternCards(): void
-setUpToolCards(): void
-setUpPublicObjectiveCards(): void
-setUpPrivateObjectiveCards(): void
-chooseToolCards(): List<ToolCard>
-choosePublicObjectiveCards(): List<PublicObjectiveCards>
+pickPatternCards(): HashMap
+getNoOfCurrentRound(): int
+getRoundTrack(): List<List<Dice>>
-deleteMatch(): void
-checkUserConnection(Set<Player>): void
-closeThreads(): void
-listenForPlayerDisconnection(): void
~getPlayerList(): List<Player>
+setPatternCardForPlayer(String, PatternCard): void
+getDraftedDice(): List<Dice>
+getTurnInRound(): int
+waitForEveryPatternCard(Map<String, List<Player>>): void
+randomizePatternCards(Map<String, List<Player>>): void
-setBoardAndStartMatch(): void
-waitForAck(): void
+draftDiceFromBoard(): void
-waitForDiceAck(): void
+placeDiceForPlayer(Dice, int, int): void
+endTurn(String): void
+stopTurn(): void
-resetAck(): void
+receiveAck(): void
-notifyDraftToPlayer(Player): void
-startMatch(): void
-writeHistoryToFile(List<MoveStatus>): void
-notifyWinner(): void
-assignPointsToPlayers(): void
-evaluateBasicPoints(): Set<Player>
-evaluateBasicPrivateObjectiveCardPoints(): Set<Player>
-evaluateBasicFavourTokenPoints(): Set<Player>
-evaluateWinner(): Player
-startRound(): void
+executeTurn(int, String): void
~getToolCardLock(): AtomicBoolean
~getControllerTimer(): ControllerTimer
-shiftPlayerList(): void
-notifyUpdatedRoundTrack(): void
-waitEndTurn(): void
~sendAvailablePositions(Player): Map<String, Boolean[][]>
~makeMove(Dice, int, int): boolean
+useToolCard(String): void
+addMoveToHistoryAndNotify(MoveStatus): void
+getCurrentRound(): Round
-placeDiceToolCard(Dice, int, int): void
+avoidToolCardUse(): void
-wakeUpToolCardThread(): void
+glazingHammerResponse(): void
+grozingPliersMove(Dice, Boolean): void
+grozingPliersResponse(): void
+fluxBrushMove(Dice): void
+fluxBrushMove(Dice, int, int): void
+fluxBrushMove()
+fluxBrushResponse(): void
+fluxRemoverMove(Dice): void
+fluxRemoverMove(Dice, int): void
+fluxRemoverMove(Dice, int, int): void
-getSelectedToolCard(String): ToolCard
+fluxRemoverMove(): void
+fluxRemoverResponse(): void
+grindingStoneMove(Dice): void
+grindingStoneResponse(): void
+copperFoilBurnisherMove(Tuple, Tuple): void
+copperFoilBurnisherResponse(): void
+corkBackedStraightedgeMove(Dice, int, int): void
+corkBackedStraightedgeResponse(): void
+lensCutterMove(int, String, String): void
+lensCutterResponse(): void
+eglomiseBrushMove(Tuple, Tuple): void
+eglomiseBrushResponse(): void
+lathekinMove(Tuple, Tuple, boolean): void
+lathekinResponse(): void
-copyPatternCard(): List<List<Box>>
+runningPliersMove(Dice, int, int): void
+runningPliersResponse(): void
+getDoubleMove(): boolean
+setDoubleMove(boolean): void
+tapWheelMove(Dice, int, Tuple, Tuple, boolean): void
-tapWheelResponse(Map<String,  Boolean[][]>, int): void
```

```
Player

-user: User
-score: int
-favourToken: int
-patternCard: PatternCard
-privateObjectiveCard: PrivateObjectiveCard

+Player(User)
+setScore(int): void
+getScore(): int
+setPatternCard(PatternCard): void
+getPatternCard(): PatternCard
+setPrivateObjectiveCard(PrivateObjectiveCard): void
+getPrivateObjectiveCard(): PrivateObjectiveCard
+getUser(): User
+getUserObserver(): UservObserver
+sendResponse(Response): void
~notifyDraft(): void
+getFavourTokens(): int
+decreaseFavorTokens(int): void
```

```
User

-username: String
-positionInRanking: int
-active: boolean
-ready: boolean
-userObserver: UserObserver
-noOfWins: int
-noOfLose: int
-matchesPlayed: List<String>
-stopWatch: StopWatch
-isStopWatchRunning: boolean

+User(String)
~getPositionInRanking(): int
~setPositionInRanking(int): void
+getUsername(): String
~getNoOfWins(): int
~incrementNoOfWins(): int
~getNoOfLose(): int
~incrementNoOfLose(): int
~getMatchesPlayed(): List<String>
+attachUserObserver(UserObserver): void
+getUserObserver(): UserObserver
~setActive(boolean): void
~getActiveTime(): long
~getFormattedTime(): String
+toString(): String
```

```
Round

-player: Player
-hasMadeAMove: AtomicBoolean
-gameManager: GameManager
-playerEndedTurn: AtomicBoolean
-blockedTurnPlayers: List<Player>
-noOfMoves: int
-avoidEndTurnNotification: AtomicBoolean

+Round(GameManager)
~startForPlayer(Player): void
+run(): void
~endTurnNotification(): void
-hasMadeAMove(): void
-waitForMove(): void
~setPlayerEndedTurn(boolean): void
+hasPlayerEndedTurn(): AtomicBoolean
~makeMove(Dice, int, int): void
~makeMove(ToolCard): void
+toolCardMoveDone(): void
+getCurrentPlayer(): Player
+avoidEndTurnNotification(boolean): void
```

```
Board

-publicObjectiveCards: PublicObjectiveCards
-toolCards: List<ToolCard>
-diceBag: List<Dice>
-draftedDice: List<Dice>

+Board(List<PublicObjectiveCard>, List<ToolCard>)
-setupDiceBag(): void
#draftDice(int): List<Dice>
~setDraftedDice(List<Dice>): void
~draftOneDice(): Dice
~addDiceToBag(Dice): void
~getDraftedDice(): List<Dice>
+getPublicObjectiveCards(): List<PublicObjectiveCards>
```

```
{abstract} PatternCard

-difficulty: int
#grid: List<List<Box>>

+getDifficulty(): int
-fillGrid(): void
+getNoOfEmptyBoxes(): int
+computeAvailablePositionsDraftedDice(List<Dice>): Map<String,boolean[][]>
+computeAvailablePositionsLathekin(): Map<String,boolean[][]>
+computeAvailablePositionsTapWheel(Dice, boolean): Map<>
-computePosition(Dice, boolean, boolean.boolean, boolean, int, int): Map<>
-boxToSwitchCheck(Dice, boolean, boolean, boolean[][], int, int, int): boolean
-singleBoxCheck(Dice, boolean, boolean, boolean, boolean[][], int, int): boolean
-checkAround(Dice, int, int, int, boolean[]): boolean
-checkAroundSwitchBox(Dice, int, int, int, int, boolean[][]): boolean
-hasDicesAround(int, int): boolean
-sameDiceColor(int, int, Dice): boolean
-sameDiceValue(int, int, Dice): boolean
-sameGridColor(boolean, int, int): boolean
-sameGridValue(boolean, int int, dice)
+noDice(int, int): boolean
-isBlank(int, int): boolean
-noDice(int, int): boolean
-isGridEmpty(): boolean
-getNoOfDice(): int
```

```
AuroraeMagnificus

-json: String
```

```
others..
```