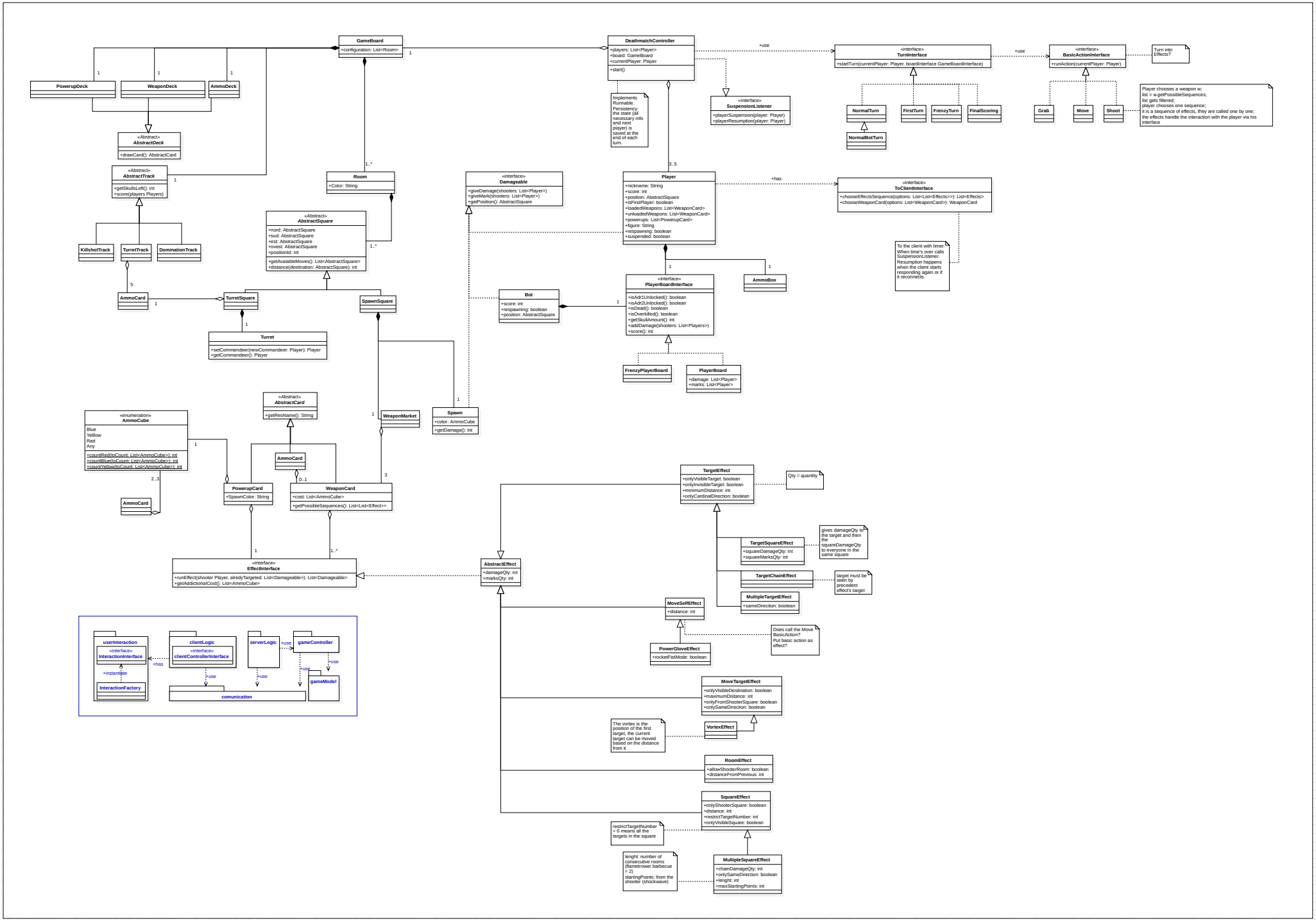
GameBoard
+configuration: List<Room>
DeathmatchController
+players: List<Player>
+board: GameBoard
+currentPlayer: Player
+start()
+use
«interface»
TurnInterface
+startTurn(currentPlayer: Player, boardInterface GameBoardInterface)
+use
«interface»
BasicActionInterface
+runAction(currentPlayer: Player)
Turn into Effects?
PowerupDeck
WeaponDeck
AmmoDeck
«Abstract»
AbstractDeck
+drawCard(): AbstractCard
Implements Runnable.
Persistency: the state (all necessary info and next player) is saved at the end of each turn.
«interface»
SuspensionListener
+playerSuspension(player: Player)
+playerResumption(player: Player)
NormalTurn
FirstTurn
FrenzyTurn
FinalScoring
Grab
Move
Shoot
Player chooses a weapon w;
list = w.getPossibleSequences;
list gets filtered;
player chooses one sequence;
it is a sequence of effects, they are called one by one;
the effects handle the interaction with the player via his interface
NormalBotTurn
«Abstract»
AbstractTrack
+getSkullsLeft(): int
+score(players Players)
Room
+Color: String
«interface»
Damageable
+giveDamage(shooters: List<Player>)
+giveMark(shooters: List<Player>)
+getPosition(): AbstractSquare
Player
+nickname: String
+score: int
+position: AbstractSquare
+isFirstPlayer: boolean
+loadedWeapons: List<WeaponCard>
+unloadedWeapons: List<WeaponCard>
+powerups: List<PowerupCard>
+figure: String
+respawning: boolean
+suspended: boolean
+has
«interface»
ToClientInterface
+chooseEffectsSequence(options: List<List<Effects>>): List<Effects>
+chooseWeaponCard(options: List<WeaponCard>): WeaponCard
To the client with timer. When time's over calls SuspensionListener. Resumption happens when the client starts responding again or if it reconnects.
KillshotTrack
TurretTrack
DominationTrack
«Abstract»
AbstractSquare
+nord: AbstractSquare
+sud: AbstractSquare
+est: AbstractSquare
+ovest: AbstractSquare
+positionId: int
+getAvaiableMoves(): List<AbstractSquare>
+distance(destination: AbstractSquare): int
AmmoCard
TurretSquare
SpawnSquare
Turret
+setCommendeer(newCommandeer: Player): Player
+getCommandeer(): Player
Bot
+score: int
+respawning: boolean
+position: AbstractSquare
«interface»
PlayerBoardInterface
+isAdr1Unlocked(): boolean
+isAdr2Unlocked(): boolean
+isDead(): boolean
+isOverkilled(): boolean
+getSkullAmount(): int
+addDamage(shooters: List<Players>)
+score(): int
AmmoBox
FrenzyPlayerBoard
PlayerBoard
+damage: List<Player>
+marks: List<Player>
«Abstract»
AbstractCard
+getResName(): String
«enumeration»
AmmoCube
Blue
Yellow
Red
Any
+countRed(pCount: List<AmmoCube>): int
+countBlue(pCount: List<AmmoCube>): int
+countYellow(pCount: List<AmmoCube>): int
WeaponMarket
Spawn
+color: AmmoCube
+getDamage(): int
AmmoCard
PowerupCard
+SpawnColor: String
AmmoCard
WeaponCard
+cost: List<AmmoCube>
+getPossibleSequences(): List<List<Effect>>
«interface»
EffectInterface
+runEffect(shooter Player, alredyTargeted: List<Damageable>): List<Damageable>
+getAddictionalCost(): List<AmmoCube>
AbstractEffect
+damageQty: int
+marksQty: int
TargetEffect
+onlyVisibleTarget: boolean
+onlyInvisibleTarget: boolean
+minimumDistance: int
+onlyCardinalDirection: boolean
Qty = quantity
TargetSquareEffect
+squareDamageQty: int
+squareMarksQty: int
gives damageQty to the target and then the squareDamageQty to everyone in the same square
TargetChainEffect
target must be seen by precedent effect's target
MultipleTargetEffect
+sameDirection: boolean
MoveSelfEffect
+distance: int
PowerGloveEffect
+rocketFistMode: boolean
Does call the Move BasicAction? Put basic action as effect?
MoveTargetEffect
+onlyVisibleDestination: boolean
+maximumDistance: int
+onlyFromShooterSquare: boolean
+onlySameDirection: boolean
VortexEffect
The vortex is the position of the first target, the current target can be moved based on the distance from it
RoomEffect
+allowShooterRoom: boolean
+distanceFromPrevious: int
SquareEffect
+onlyShooterSquare: boolean
+distance: int
+restrictTargetNumber: int
+onlyVisibleSquare: boolean
restrictTargetNumber = 0 means all the targets in the square
MultipleSquareEffect
+chainDamageQty: int
+onlySameDirection: boolean
+lenght: int
+maxStartingPoints: int
lenght: number of consecutive rooms (flamethrower barbecue = 2)
startingPoints: from the shooter (shockwave)
userInteraction
«interface»
InteractionInterface
+instantiate
InteractionFactory
clientLogic
«interface»
clientControllerInterface
serverLogic
+use
gameController
+has
+use
+use
+use
+use
gameModel
comunication