Model <<enumeration>> GameState LOGIN IN_GAME Game - players: List<Player> [1...4] - gamestate: GameState Player - maxPlayers: int <<enumeration>> SoloOpponent - developmentCards: DevCardsTray PlayerState - nickname: String - leaderCardsDeck: Deck - blackCross: int board: Board - market: MarketTray host 1..4 _ - state: PlayerState - actionTokens: SoloActionTokens <<interface>> - Iorenzo: SoloOpponent TAKE RESOURCES StartStopStrategy playerOrder: int BUY_ONE_DEV_CARD + increaseBlackCross(int) - isFirstTurn: boolean + setGameStrategy(StartStopStrategy strategy) ACTIVATE PRODUCTION + startMatch() + pickActionTokens() + setGameState(GameState) + endGame() + getGameState(): GameState + compareTo(Player) + setMaxPlayer(int number) + getMaxPlayer(): int SoloStrategy MoreThanOneStrategy + startGame() + startMatch() + startMatch() + endGame() + isNicknameTaken(String): boolean + endGame() + endGame() + getPlayerNickname(int): String Board + startMatch() SoloActionTokens - faithMarker: int Depots popeFavorTiles: boolean[] - stack: SoloActionToken[] - warehouseDepots: Depots layer1: DepotsLayer - strongBox: Resources - layer2: DepotsLayer developmentCards: List<List<DevelopmentCard>> - layer3: DepotsLayer + shuffle() - leaderCards: List<LeaderCard> 1 - extraSpace: Resources Deck - basicProductionPower: ProductionPower DevCardsTray MarketTray - cards: ArrayList + canBeStores(Resources): boolean - tray: DevelopmentCard[4][3][4] + depotsToList: List<List<Integer>> + increaseFaith(int) - columns: List<Color> + getValue() + activateLeaderCard(LeaderCard) + shuffle() - rows: List<Color> + addLeaderCardLayer(LeaderCard) SoloActionToken + buy(int position) + getColorsByLevel(int): List<Color> + pickCard() + tokenDiscardDevCards(Color color) - slideMarble: Color + takeResources(Resources) + buyDevCard(DevelopmentCard, List<DevelopmentCard>): - icon: File + storeResources(Resources) + addToStrongbox(Resources) + takeMarble(int): Color[] - effect: SoloActionEffect + takeResources(List<Color>): Resources - convertColor(Color, Resources) compose Y + activateProduction(): Resources + spendResources(Resources) <<Abstract>> + setPopeFavorTiles(int) DepotsLayer <<enumeration>> Card + getVictoryPoints(): int <<abstract>> ActionType contain - amountStored: int SoloActionEffect - face: File - maxStorable: int DISCARDDEVCARDS - victoryPoints: int type: ActionType component - layerType: StorableResources MOVE2FORWARD + doAction() MOVEANDSHUFFLE + toList(): List<Integer> + takeFromLayer(Resources) extend + addToTotal(): Resources LeaderCard <<enumeration>> LeaderCardsRequirements DevelopmentCard - requirements: Requirements <<abstract>> ConcreteEffect RESOURCES, <<Interface>> - requiresType: LeaderCardsRequirements EffectDecorator - cost: Resources BuyDevCardStrategy specialAbility: SpecialAbility DEVELOPMENTCARDS. type: ActionType - type: Color - isTaken: boolean type: ActionType DEVELOPMENTCARD + doAction() + buyDevCard(DevelopmentCard, Board) - level: int + doAction() - productionPower: ProductionPower - isTaken: boolean + doAction() <<enumeration>> decorates StorableResources store extend requires <<enumeration>> | DiscardDevCardDecorator STONES Color SERVANTS - color: Color GREY SHIELDS BLUE EMPTYLAYER <<Abstract>> <<enumeration>> VIOLET Requirements BuyDevCardStandard 1 YELLOW RED SpecialAbility - ActiveStatus: boolean EXTRADEPOT + buyDevCard(DevelopmentCard, Board) ProductionPower WHITE + canBeActivated() DISCOUNT GREEN + getActiveStatus(): boolean - inputResources: Resources WHITEMARBLE - outputResources: Resources MOREPOWER BuyDevCardDiscount + activate() + buyDevCard(DevelopmentCard, Board) RequiresResources RequiresColors RequiresLevel - required: Resources - requires: Color[] - requiredColor: Color - requiredLevel: int - getRequired(): Color[] + canBeActivated() - getRequired(): Resources Resources - getRequiredColor(): Color + canBeActivated() - getRequiredLevel(): int + canBeActivated() - coins: int stones: int 0..* - servants: int - shields: int - faith: int part of + getValue(): int[] + typeVariety(): int[] + subtractAmount(int, StorableResource) + getBiggerResourceType() part of

