Revised UML

<<interface>> Monster
+ getName()
+ getHealth()
+ reduceHealth()

<<interface>> Location
+ getX()
+ setX()

<<interface>> Treasure
+ getName()

Otyugh
-name: String
-health: int

+ getName()
+ getHealth()
+ reduceHealth()

AbstractLocation
- northDoor:boolean
- southDoor:boolean
- eastDoor:boolean
- westDoor:boolean
-arrowList: List<Arrow>

+ getX()
+ setX()

AbstractTreasire
- name:String

+ getName()

Sapphire

Diamond

Ruby

<<interface>> Arrow
+ getPosition()
+ setPosition()

Cave
treasureList:List<Treasure>
monsterList: List<Monster>

+ getTreasure()
+ addTreasure()
+ removeTreasure()
+ getMonster()
+ addMonster()
+ removeMonster()

Tunnel

DungeonModel
-start: Point2D
-end: Point2D
-player: Player
-locations[][]: Location

+ getStartPosition()
+ getEndPosition()
+ getLocations()
+getDungeon()
+ move()
+ pickTreasure()
+ getPlayerDescription()
+ getLocationDescription()
+ getSmellLevel()
+getPlayer()
+ shoot()
+ pickArrow()
+ combat()
+ isGameOver()
+resetPlayer()
+resetStartCave()
+resetEndCave()
+resetDungeon()

Point2D
- x: int
-y: int

+ getX()
+ setX()

<<interface>> Player
+ getX()
+ setX()

CrookedArrow
-position: Point2D

+ getPosition()
+ setPosition()

DungeonPlayer
-life: int
- name:String
-position: Point2D
-bag: List<Treasure>
-quiver: List<Arrow>

+ getX()
+ setX()

<<interface>> ReadonlyDungeonModel
+ getStartPosition()
+ getEndPosition()
+ getLocations()
+ getDungeon()
+ getPlayerDescription()
+ getLocationDescription()
+ getSmellLevel()
+ getPlayer()

<<interface>> Dungeon
+ move()
+ pickTreasure()
+ shoot()
+ pickArrow()
+ combat()
+resetPlayer()
+resetStartCave()
+resetEndCave()
+resetDungeon()

<<interface>> DungeonController
+ playGame()

DungeonConsoleController
- out: Appendable
-scan: Scanner

+ playGame()

DungeonViewControllerImpl
- model: Dungeon
-view: DungeonView
- isRPressed: boolean
- reuseLocation: Location[][]
- reuseStart: Point2D
- reuseEnd: Point2D

+ playGame()
+handlePanelClick()
+actionPerformed()
+keyPressed()
+keyReleased()

<<interface>> DungeonView
+ refresh()
+ makeVisible()
+ setListeners()
+addClickListener()
+ resetFocus()
+ setOperationdescription()
+ setInformation()
createSettings()
+addDungeon()
+updateDungeon()
+restartGame()

<<interface>> DungeonViewController
+ playGame()
+ handlePanelClick()

DungeonSwingView
- model: ReadonlyDungeonModel
- settings: JMenuItem
- restart: JMenuItem
- quit: JMenuItem
- dungeonPanel: DungeonPanel
- descriptionPanel: DescriptionPanel
- buttonPanel: ButtonPanel

+ refresh()
+ makeVisible()
+ setListeners()
+addClickListener()
+ resetFocus()
+ setOperationdescription()
+ setInformation()
createSettings()
+addDungeon()
+updateDungeon()
+restartGame()

SettingsDialog
- txtRows: JTextField
- txtCols: JTextField
- txtWrapping: JTextField
- txtConnectivity: JTextField
- txtPercent: JTextField
- txtMonster: JTextField
-rowLabel: JLabel
-colLabel: JLabel
-wrappingLabel: JLabel
-connectivityLabel: JLabel
-percentLabel: JLabel
-monsterLabel: JLabel
-startButton: JButton
-input: String

~getTextValue()

DungeonPanel
- model: ReadonlyDungeonModel
-visitedLocations: Location[][]
labels: JLabel[][]

paintComponent()
~ addNewLocation()
~ updateVisitedLocation()
~resetPanel()

DescriptionPanel
- model: ReadonlyDungeonModel
-opResult:String
labels: JLabel[][]

paintComponent()
~ updateOpResult()

ButtonPanel
- playerButton: JButton
-playerDescription:String

paintComponent()
~ addActionListener()
~ updateDescription()