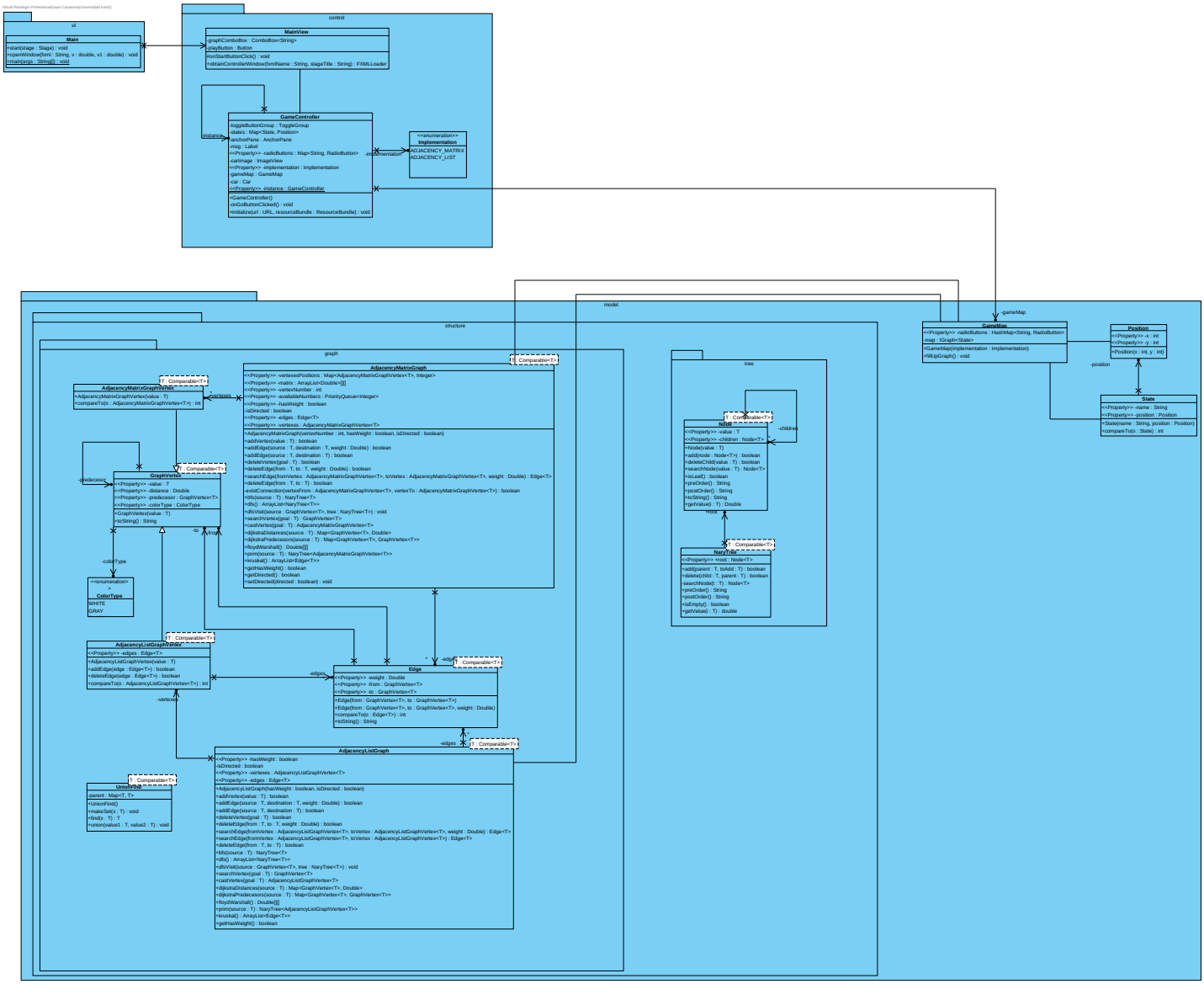
Visual Paradigm Professional(Juan Catanoval(Universidad Icesi))
ui
Main
+start(stage : Stage) : void
+openWindow(text : String, y : double, x1 : double) : void
+main(args : String[]) : void
control
MainView
-graphComboBox : ComboBox<String>
-playButton : Button
#onStartButtonClick() : void
+loMainControllerWindowItemName : String, stageTitle : String) : FXMLLoader
GameController
-toggleButtonGroup : ToggleGroup
-states : Map<State, Position>
-anchorPane : AnchorPane
-msg : Label
<<Property>> -radioButtons : Map<String, RadioButton>
-carImage : ImageView
<<Property>> -implementation : Implementation
-gameMap : GameMap
-car : Car
<<Property>> -instance : GameController
+GameController()
-onButtonClicked() : void
+initialize(url : URL, resourceBundle : ResourceBundle) : void
-instance
-implementation
<<enumeration>>
Implementation
ADJACENCY_MATRIX
ADJACENCY_LIST
model
-gameMap
GameMap
<<Property>> -radioButtons : HashMap<String, RadioButton>
-map : IGraph<State>
+GameMap(implementation : Implementation)
#fillUpGraph() : void
Position
<<Property>> -x : int
<<Property>> -y : int
+Position(x : int, y : int)
-position
State
<<Property>> -name : String
<<Property>> -position : Position
+State(name : String, position : Position)
+compareTo(o : State) : int
structure
graph
AdjacencyMatrixGraph
T : Comparable<T>
<<Property>> -vertexesPositions : Map<AdjacencyMatrixGraphVertex<T>, Integer>
<<Property>> -matrix : ArrayList<Double>[][]
<<Property>> -vertexNumber : int
<<Property>> -availableNumberis : PriorityQueue<Integer>
<<Property>> -hasWeight : boolean
-isDirected : boolean
<<Property>> -edges : Edge<T>
<<Property>> -vertexes : AdjacencyMatrixGraphVertex<T>
+AdjacencyMatrixGraph(vertexNumber : int, hasWeight : boolean, isDirected : boolean)
+addVertex(value : T) : boolean
+addEdge(source : T, destination : T, weight : Double) : boolean
+addEdge(source : T, destination : T) : boolean
+deleteVertex(goal : T) : boolean
+deleteEdge(from : T, to : T, weight : Double) : boolean
+searchEdge(fromVertex : AdjacencyMatrixGraphVertex<T>, toVertex : AdjacencyMatrixGraphVertex<T>, weight : Double) : Edge<T>
+deleteEdge(from : T, to : T) : boolean
-existConnection(vertexFrom : AdjacencyMatrixGraphVertex<T>, vertexTo : AdjacencyMatrixGraphVertex<T>) : boolean
+dfs(source : T) : NaryTree<T>
+dfs() : ArrayList<NaryTree<T>>
+dfsVisit(source : GraphVertex<T>, tree : NaryTree<T>) : void
+searchVertex(goal : T) : GraphVertex<T>
+castVertex(goal : T) : AdjacencyMatrixGraphVertex<T>
+dijkstraDistances(source : T) : Map<GraphVertex<T>, Double>
+dijkstraPredecessors(source : T) : Map<GraphVertex<T>, GraphVertex<T>>
+floydWarshall() : Double[][]
+prim(source : T) : NaryTree<AdjacencyMatrixGraphVertex<T>>
+kruskal() : ArrayList<Edge<T>>
+getHasWeight() : boolean
+getDirected() : boolean
+setDirected(directed : boolean) : void
AdjacencyMatrixGraphVertex
T : Comparable<T>
+AdjacencyMatrixGraphVertex(value : T)
+compareTo(o : AdjacencyMatrixGraphVertex<T>) : int
-vertexes
GraphVertex
T : Comparable<T>
<<Property>> -value : T
<<Property>> -distance : Double
<<Property>> -predecessor : GraphVertex<T>
<<Property>> -colorType : ColorType
+GraphVertex(value : T)
+toString() : String
-predecessor
-colorType
-to
-from
<<enumeration>>
ColorType
WHITE
GRAY
AdjacencyListGraphVertex
T : Comparable<T>
<<Property>> -edges : Edge<T>
+AdjacencyListGraphVertex(value : T)
+addEdge(edge : Edge<T>) : boolean
+deleteEdge(edge : Edge<T>) : boolean
+compareTo(o : AdjacencyListGraphVertex<T>) : int
-edges
-vertex
Edge
T : Comparable<T>
<<Property>> -weight : Double
<<Property>> -from : GraphVertex<T>
<<Property>> -to : GraphVertex<T>
+Edge(from : GraphVertex<T>, to : GraphVertex<T>)
+Edge(from : GraphVertex<T>, to : GraphVertex<T>, weight : Double)
+compareTo(o : Edge<T>) : int
+toString() : String
-edges
AdjacencyListGraph
T : Comparable<T>
<<Property>> -hasWeight : boolean
-isDirected : boolean
<<Property>> -vertexes : AdjacencyListGraphVertex<T>
<<Property>> -edges : Edge<T>
+AdjacencyListGraph(hasWeight : boolean, isDirected : boolean)
+addVertex(value : T) : boolean
+addEdge(source : T, destination : T, weight : Double) : boolean
+addEdge(source : T, destination : T) : boolean
+deleteVertex(goal : T) : boolean
+deleteEdge(from : T, to : T, weight : Double) : boolean
+searchEdge(fromVertex : AdjacencyListGraphVertex<T>, toVertex : AdjacencyListGraphVertex<T>, weight : Double) : Edge<T>
+searchEdge(fromVertex : AdjacencyListGraphVertex<T>, toVertex : AdjacencyListGraphVertex<T>) : Edge<T>
+deleteEdge(from : T, to : T) : boolean
+dfs(source : T) : NaryTree<T>
+dfs() : ArrayList<NaryTree<T>>
+dfsVisit(source : GraphVertex<T>, tree : NaryTree<T>) : void
+searchVertex(goal : T) : GraphVertex<T>
+castVertex(goal : T) : AdjacencyListGraphVertex<T>
+dijkstraDistances(source : T) : Map<GraphVertex<T>, Double>
+dijkstraPredecessors(source : T) : Map<GraphVertex<T>, GraphVertex<T>>
+floydWarshall() : Double[][]
+prim(source : T) : NaryTree<AdjacencyListGraphVertex<T>>
+kruskal() : ArrayList<Edge<T>>
+getHasWeight() : boolean
UnionFind
T : Comparable<T>
-parint : Map<T, T>
+UnionFind()
+makeSet(x : T) : void
+find(x : T) : T
+union(value1 : T, value2 : T) : void
tree
Node
T : Comparable<T>
<<Property>> -value : T
<<Property>> -children : Node<T>
+Node(value : T)
+add(node : Node<T>) : boolean
+deleteChild(value : T) : boolean
+searchNode(value : T) : Node<T>
+isLeaf() : boolean
+preOrder() : String
+postOrder() : String
+toString() : String
+getValue() : T) : Double
-children
-root
NaryTree
T : Comparable<T>
<<Property>> -root : Node<T>
+add(parent : T, toAdd : T) : boolean
+delete(child : T, parent : T) : boolean
-searchNode() : T) : Node<T>
+preOrder() : String
+postOrder() : String
+isEmpty() : boolean
+getValue() : T) : double