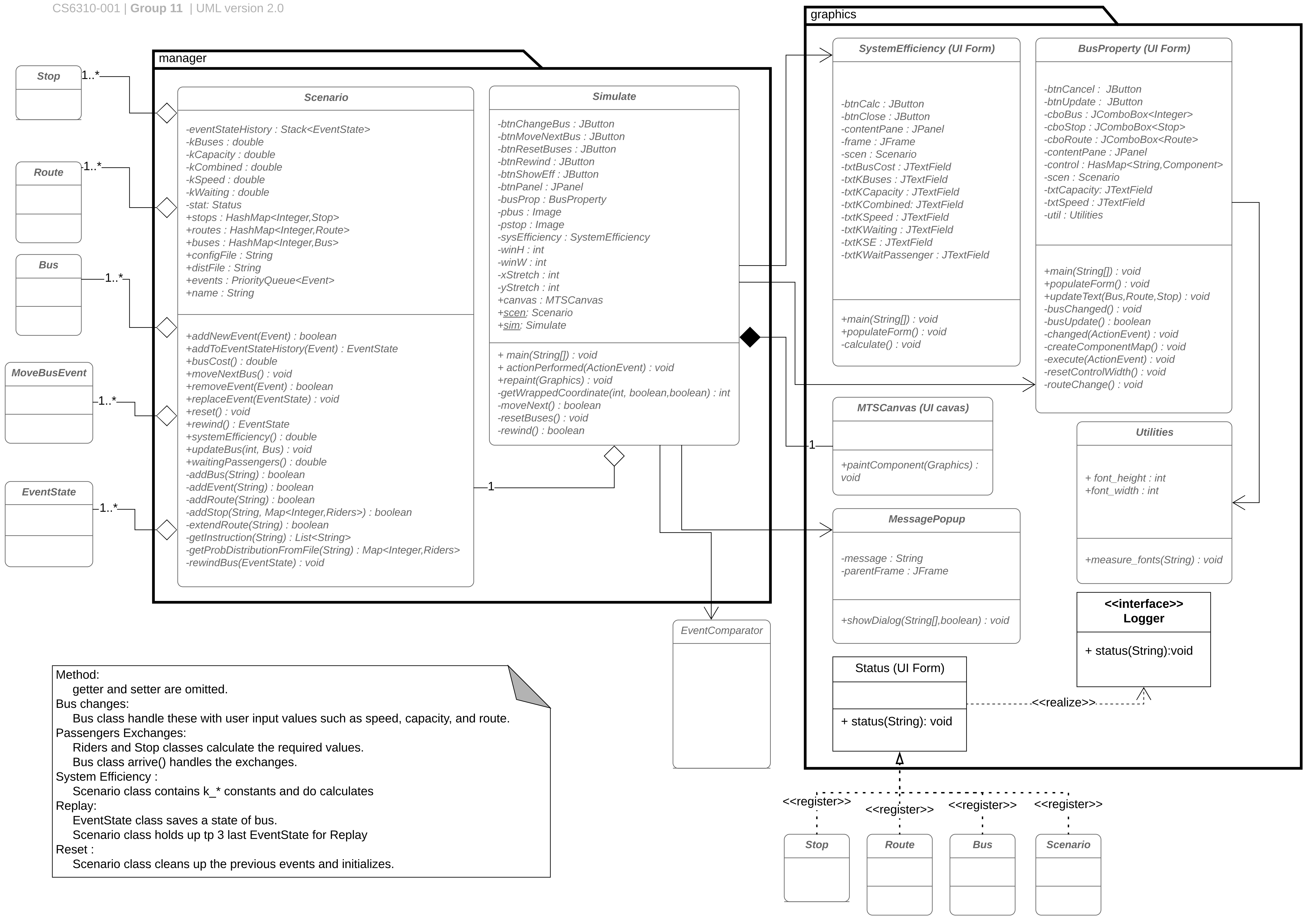
Table of contents

This design document contains the system's architectural documents and the source code documentation (Javadocs). These resources were created to simplify the Mass Simulation System's maintenance and further software updates.

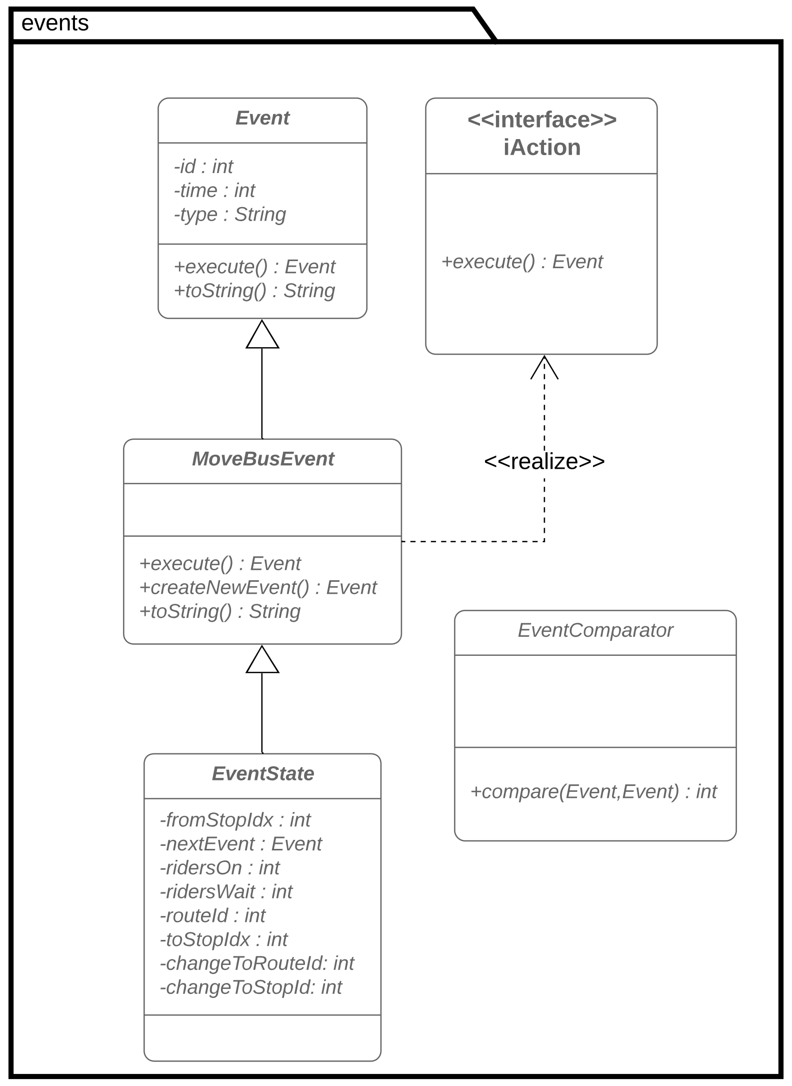
The document is structured as follows:

UML Class Diagram-Part 1

CS6310-001 | **Group 11** | UML version 2.0

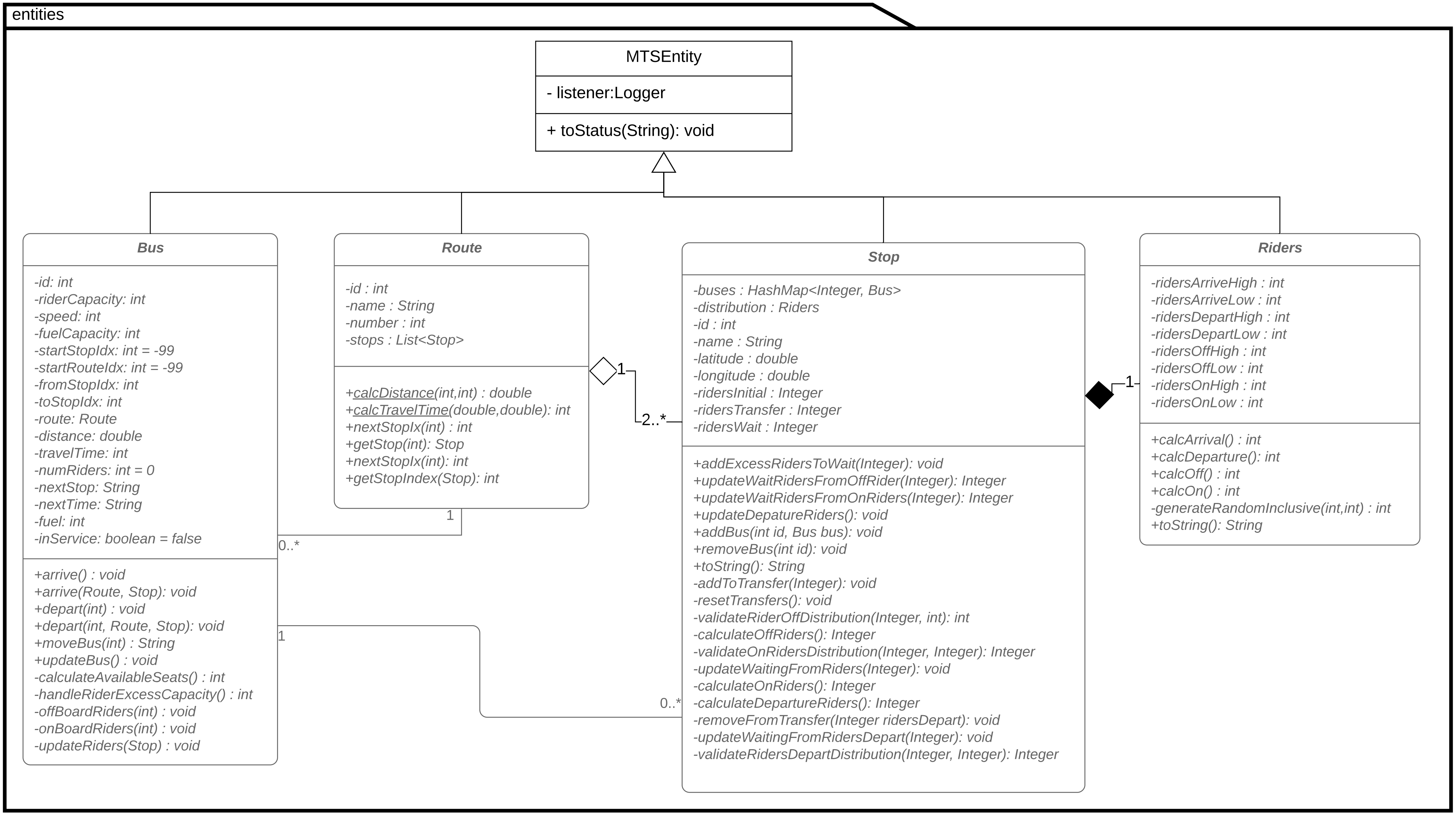


Method:
getter and setter are omitted.
Bus changes:
Bus class handle these with user input values such as speed, capacity, and route.
Passengers Exchanges:
Riders and Stop classes calculate the required values.
Bus class arrive() handles the exchanges.
System Efficiency :
Scenario class contains k_* constants and do calculates
Replay:
EventState class saves a state of bus.
Scenario class holds up tp 3 last EventState for Replay
Reset :
Scenario class cleans up the previous events and initializes.

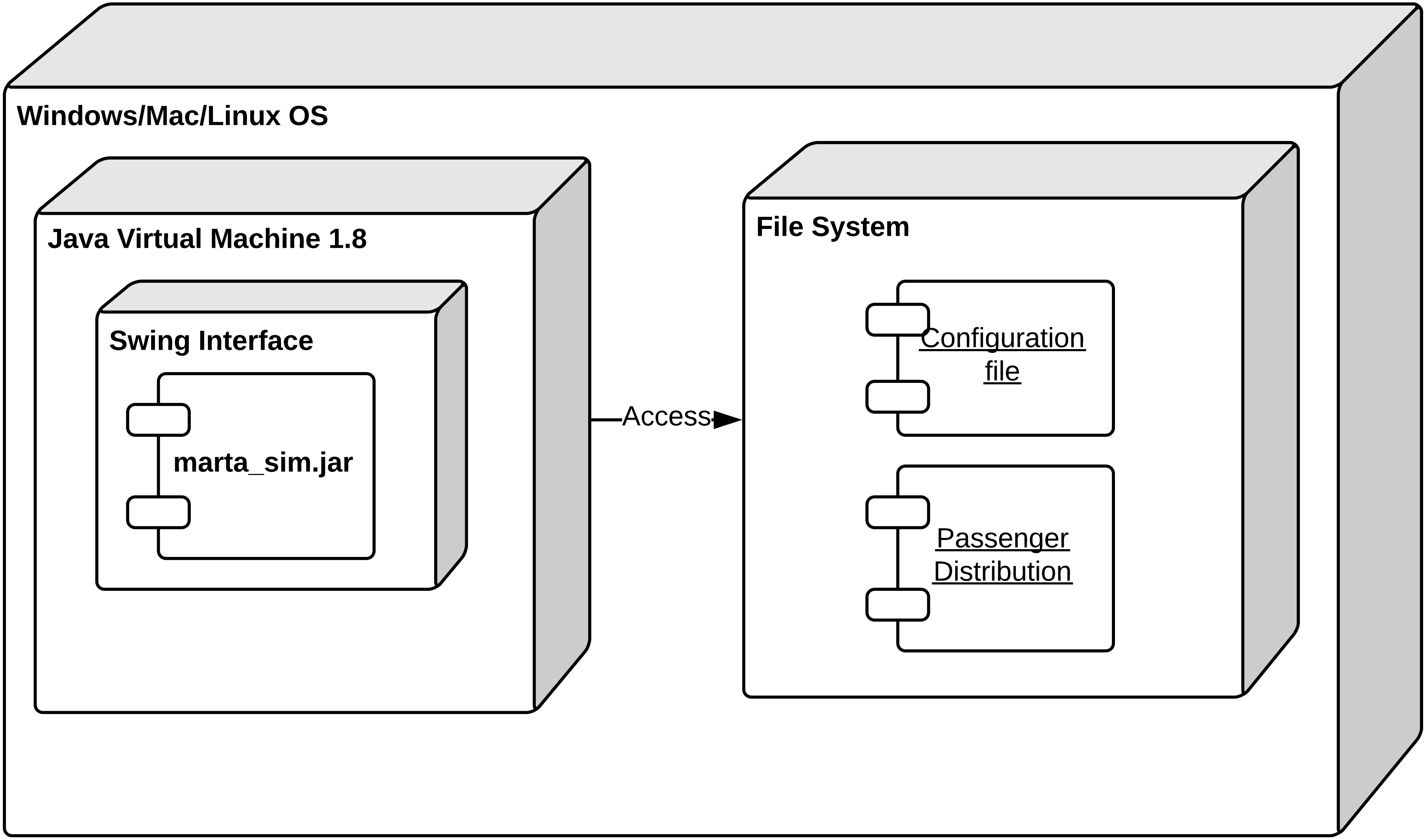


UML Class Diagram-Part 2

CS6310-001 | **Group 11** | UML version 2.0

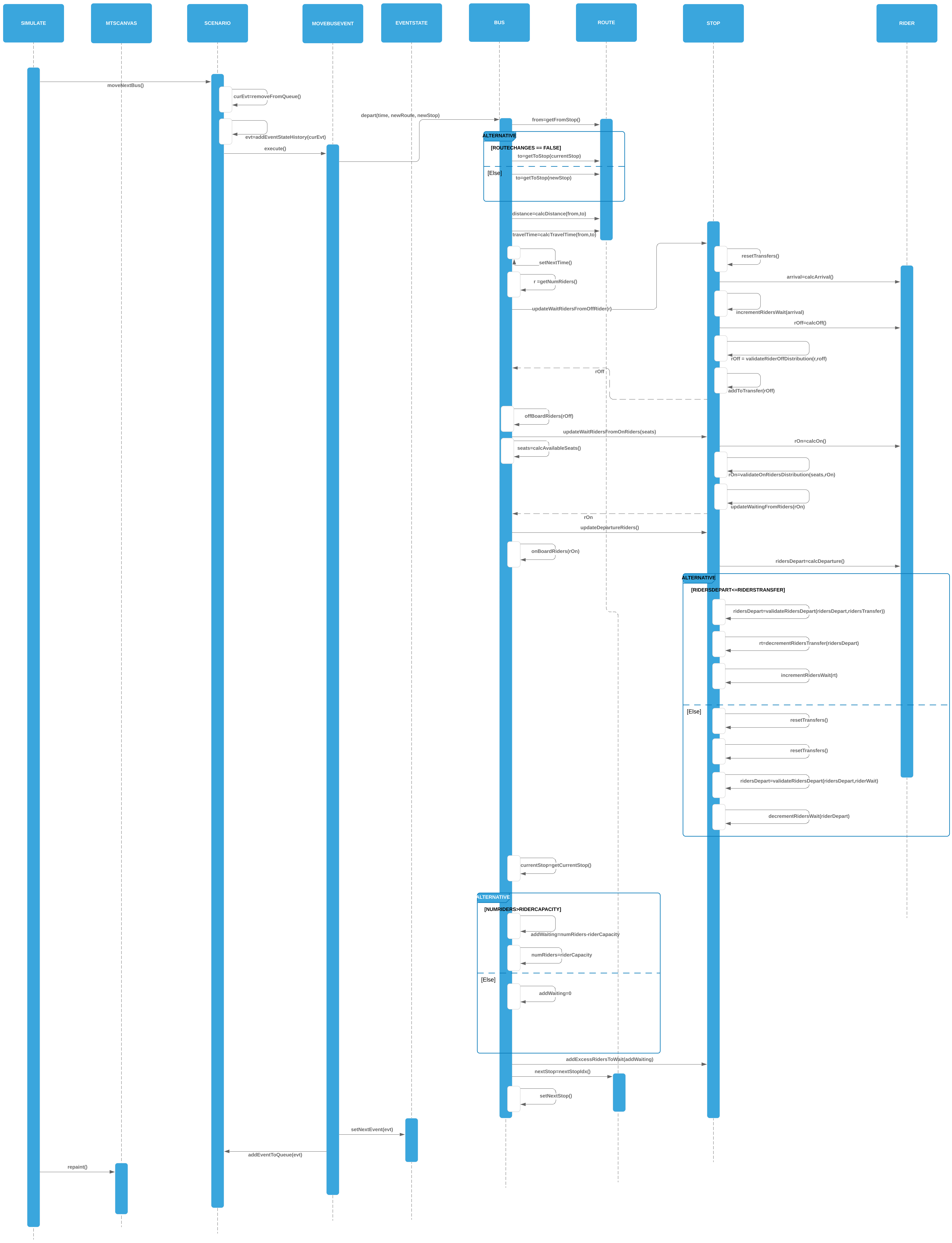

entities
MTSEntity
- listener:Logger
+ toStatus(String): void
Bus
-id: int
-riderCapacity: int
-speed: int
-fuelCapacity: int
-startStopIdx: int = -99
-startRouteIdx: int = -99
-fromStopIdx: int
-toStopIdx: int
-route: Route
-distance: double
-travelTime: int
-numRiders: int = 0
-nextStop: String
-nextTime: String
-fuel: int
-inService: boolean = false
+arrive() : void
+arrive(Route, Stop): void
+depart(int) : void
+depart(int, Route, Stop): void
+moveBus(int) : String
+updateBus() : void
-calculateAvailableSeats() : int
-handleRiderExcessCapacity() : int
-offBoardRiders(int) : void
-onBoardRiders(int) : void
-updateRiders(Stop) : void
Route
-id : int
-name : String
-number : int
-stops : List<Stop>
+calcDistance(int,int) : double
+calcTravelTime(double,double): int
+nextStopIx(int) : int
+getStop(int): Stop
+nextStopIx(int): int
+getStopIndex(Stop): int
1
0..*
1
2..*
1
0..*
Stop
-buses : HashMap<Integer, Bus>
-distribution : Riders
-id : int
-name : String
-latitude : double
-longitude : double
-ridersInitial : Integer
-ridersTransfer : Integer
-ridersWait : Integer
+addExcessRidersToWait(Integer): void
+updateWaitRidersFromOffRider(Integer): Integer
+updateWaitRidersFromOnRiders(Integer): Integer
+updateDepatureRiders(): void
+addBus(int id, Bus bus): void
+removeBus(int id): void
+toString(): String
-addToTransfer(Integer): void
-resetTransfers(): void
-validateRiderOffDistribution(Integer, int): int
-calculateOffRiders(): Integer
-validateOnRidersDistribution(Integer, Integer): Integer
-updateWaitingFromRiders(Integer): void
-calculateOnRiders(): Integer
-calculateDepartureRiders(): Integer
-removeFromTransfer(Integer ridersDepart): void
-updateWaitingFromRidersDepart(Integer): void
-validateRidersDepartDistribution(Integer, Integer): Integer
1
Riders
-ridersArriveHigh : int
-ridersArriveLow : int
-ridersDepartHigh : int
-ridersDepartLow : int
-ridersOffHigh : int
-ridersOffLow : int
-ridersOnHigh : int
-ridersOnLow : int
+calcArrival() : int
+calcDeparture(): int
+calcOff() : int
+calcOn() : int
-generateRandomInclusive(int,int) : int
+toString(): String

GROUP: A9-11 - CS-6310-O01 - ASSIGNMENT 09
DEPLOYMENT DIAGRAM (UML 2.0)



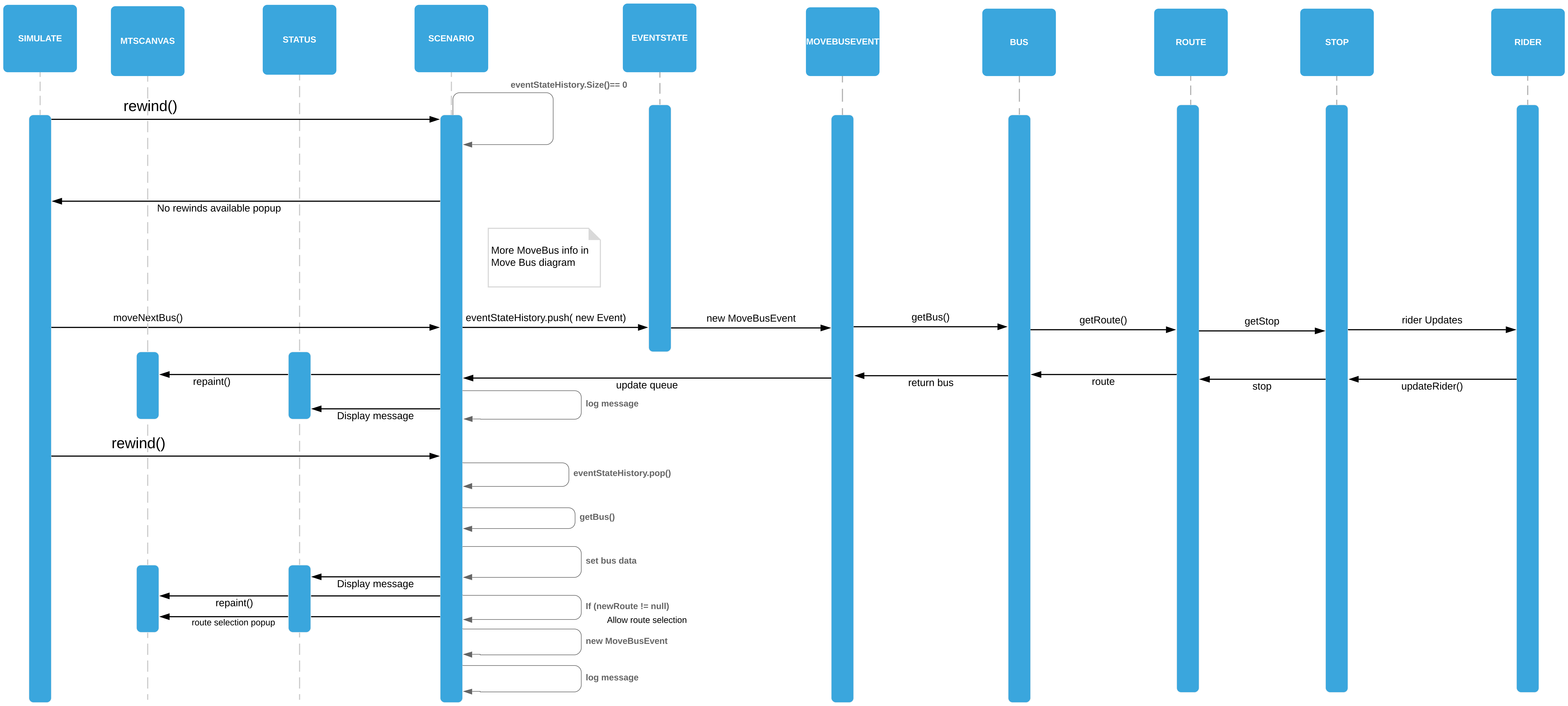
GROUP: A9-11 - CS-6310-O01 - ASSIGNMENT 09
SEQUENCE DIAGRAM MOVE BUS (UML 2.0)

THIS DIAGRAM OMMITED MOST OF THE SYSTEM'S VISUAL ELEMENTS, WE ONLY REPRENSENTED A FEW FOR CLARITY


SIMULATE
MTSCANVAS
SCENARIO
MOVEBUSEVENT
EVENTSTATE
BUS
ROUTE
STOP
RIDER
moveNextBus()
curEvt=removeFromQueue()
evt=addEventStateHistory(curEvt)
execute()
depart(time, newRoute, newStop)
from=getFromStop()
ALTERNATIVE
[ROUTECHANGES == FALSE]
to=getToStop(currentStop)
[Else]
to=getToStop(newStop)
distance=calcDistance(from,to)
travelTime=calcTravelTime(from,to)
setNextTime()
r =getNumRiders()
updateWaitRidersFromOffRider(r)
resetTransfers()
arrival=calcArrival()
incrementRidersWait(arrival)
rOff=calcOff()
rOff = validateRiderOffDistribution(r,roff)
rOff
addToTransfer(rOff)
offBoardRiders(rOff)
updateWaitRidersFromOnRiders(seats)
seats=calcAvailableSeats()
rOn=calcOn()
rOn=validateOnRidersDistribution(seats,rOn)
updateWaitingFromRiders(rOn)
rOn
updateDepartureRiders()
onBoardRiders(rOn)
ridersDepart=calcDeparture()
ALTERNATIVE
[RIDERSDEPART<=RIDERSTRANSFER]
ridersDepart=validateRidersDepart(ridersDepart,ridersTransfer))
rt=decrementRidersTransfer(ridersDepart)
incrementRidersWait(rt)
[Else]
resetTransfers()
resetTransfers()
ridersDepart=validateRidersDepart(ridersDepart,riderWait)
decrementRidersWait(riderDepart)
currentStop=getCurrentStop()
ALTERNATIVE
[NUMRIDERS>RIDERCAPACITY]
addWaiting=numRiders-riderCapacity
numRiders=riderCapacity
[Else]
addWaiting=0
addExcessRidersToWait(addWaiting)
nextStop=nextStopIdx()
setNextStop()
setNextEvent(evt)
addEventToQueue(evt)
repaint()

GROUP: A9-11 - CS-6310-O01 - ASSIGNMENT 09
SEQUENCE DIAGRAM REWIND BUS (UML 2.0)

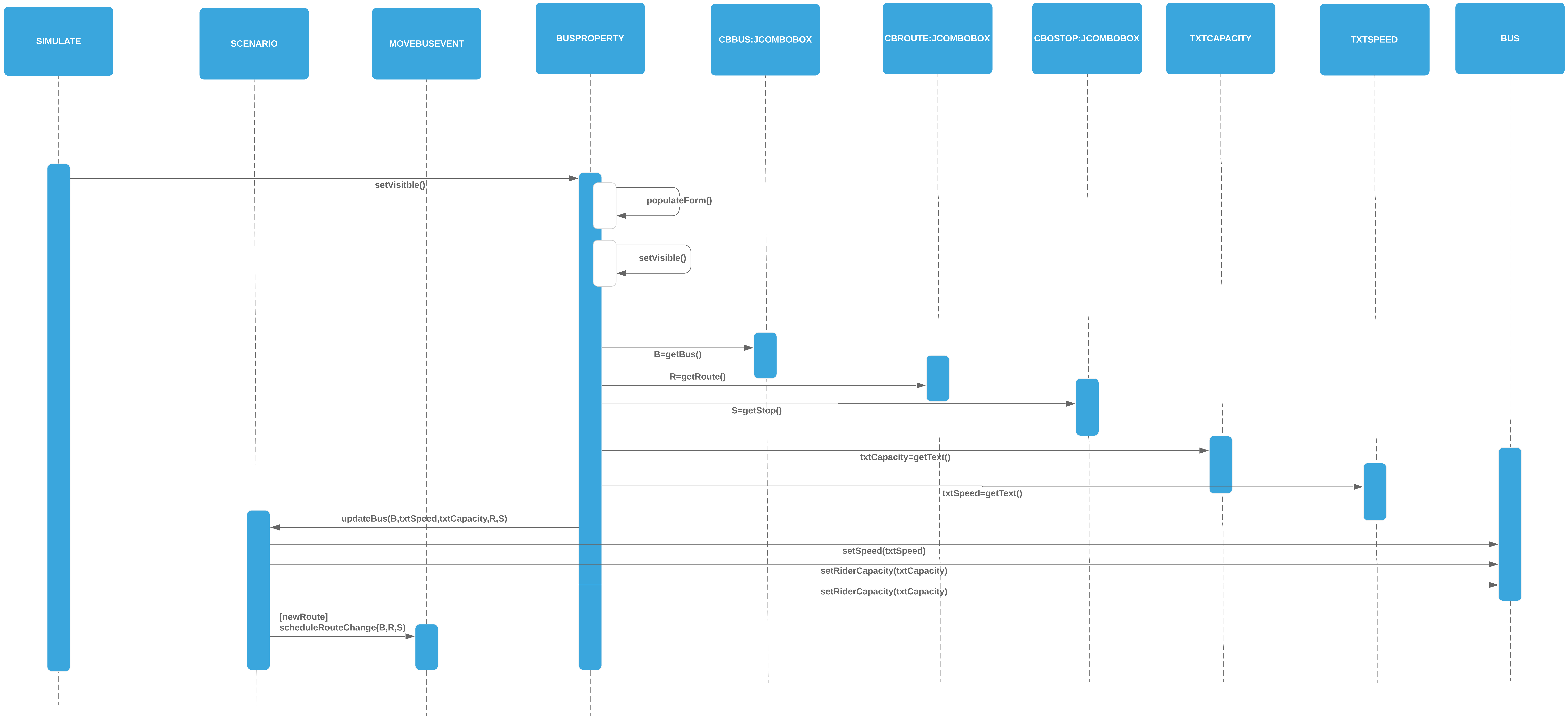
THIS DIAGRAM OMMITED MOST OF THE SYSTEM'S VISUAL ELEMENTS, WE ONLY REPRENSENTED A FEW FOR CLARITY



GROUP: A9-11 - CS-6310-O01 - ASSIGNMENT 09
SEQUENCE CHANGE BUS (UML 2.0)

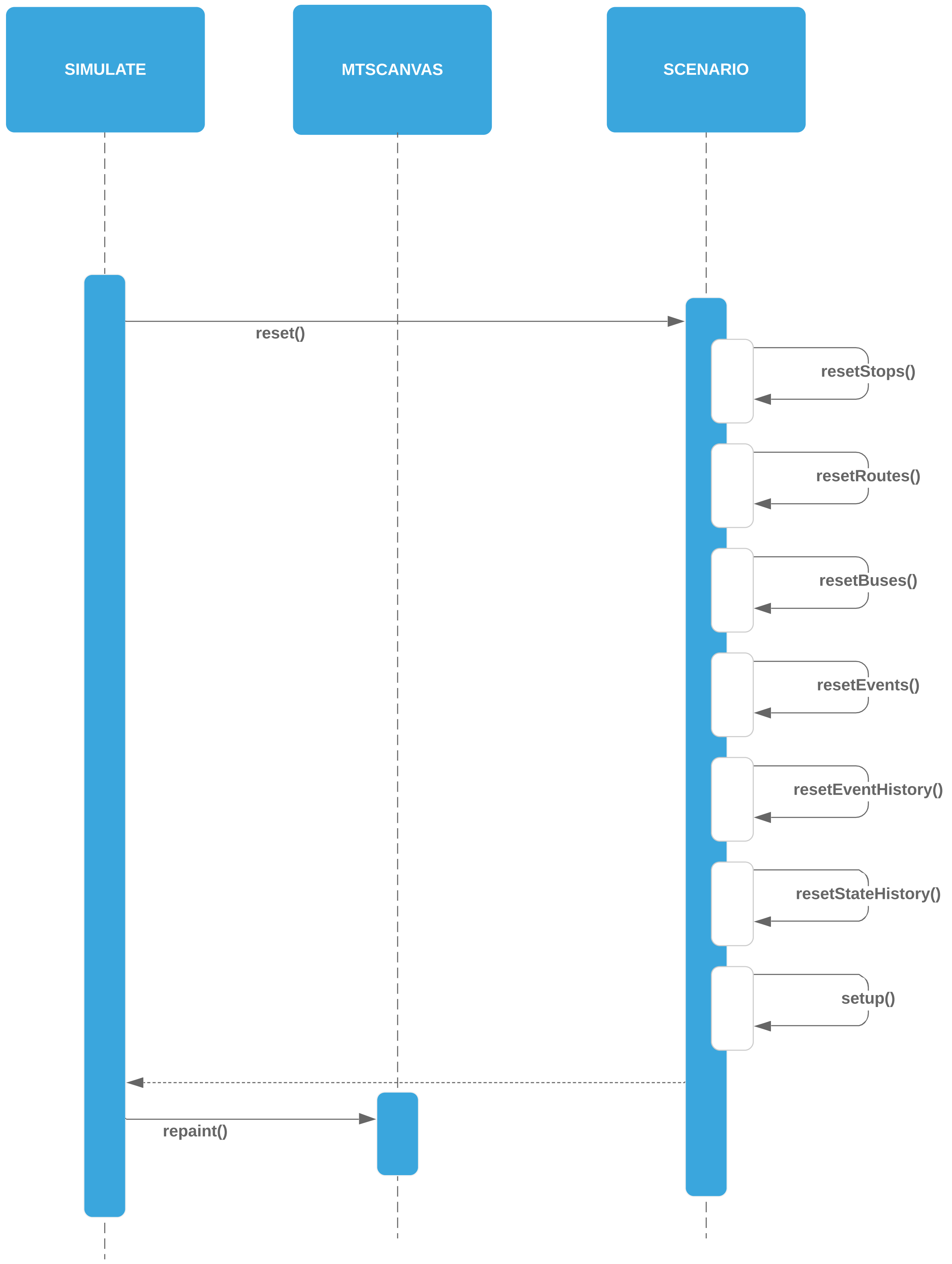
THIS DIAGRAM OMMITED MOST OF THE SYSTEM'S VISUAL ELEMENTS, WE ONLY REPRENSENTED A FEW FOR CLARITY

BEFORE SETTING ANY OF THE BUS CHANGE (CAPACITY, SPEED, ROUTE) WE ALWAYS CHECK IF THERE WAS REALLY A CHANGE BY COMPARING THE OLD VALUE WITH THE ENTERED ELEMENT IN THE COMBOBOXES AND TEXT AREAS. WE DID NOT SHOW THIS BEHAVIOR IN THE DIAGRAM, BUT IT IS IMPLEMENTED IN THE CODE

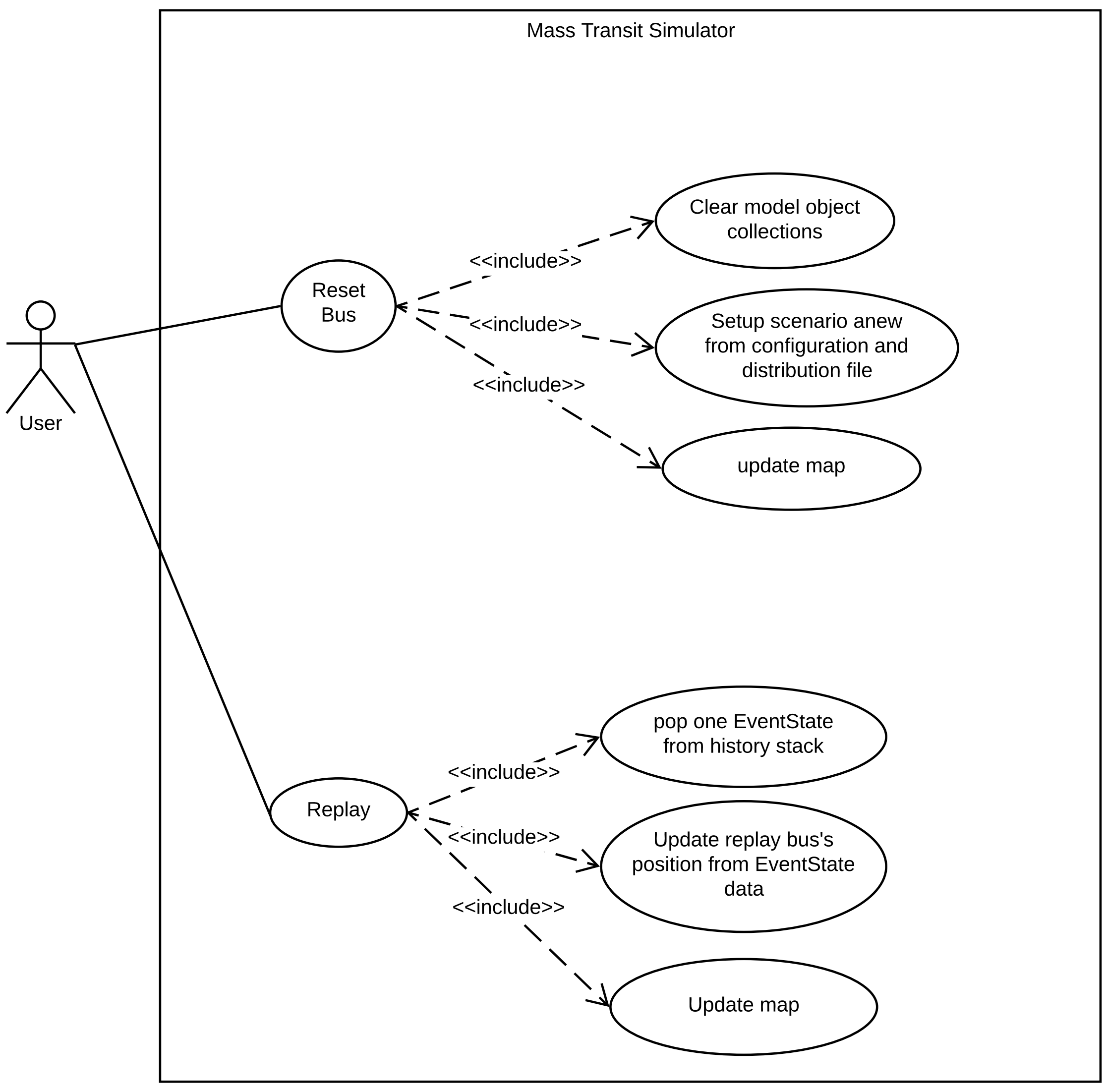


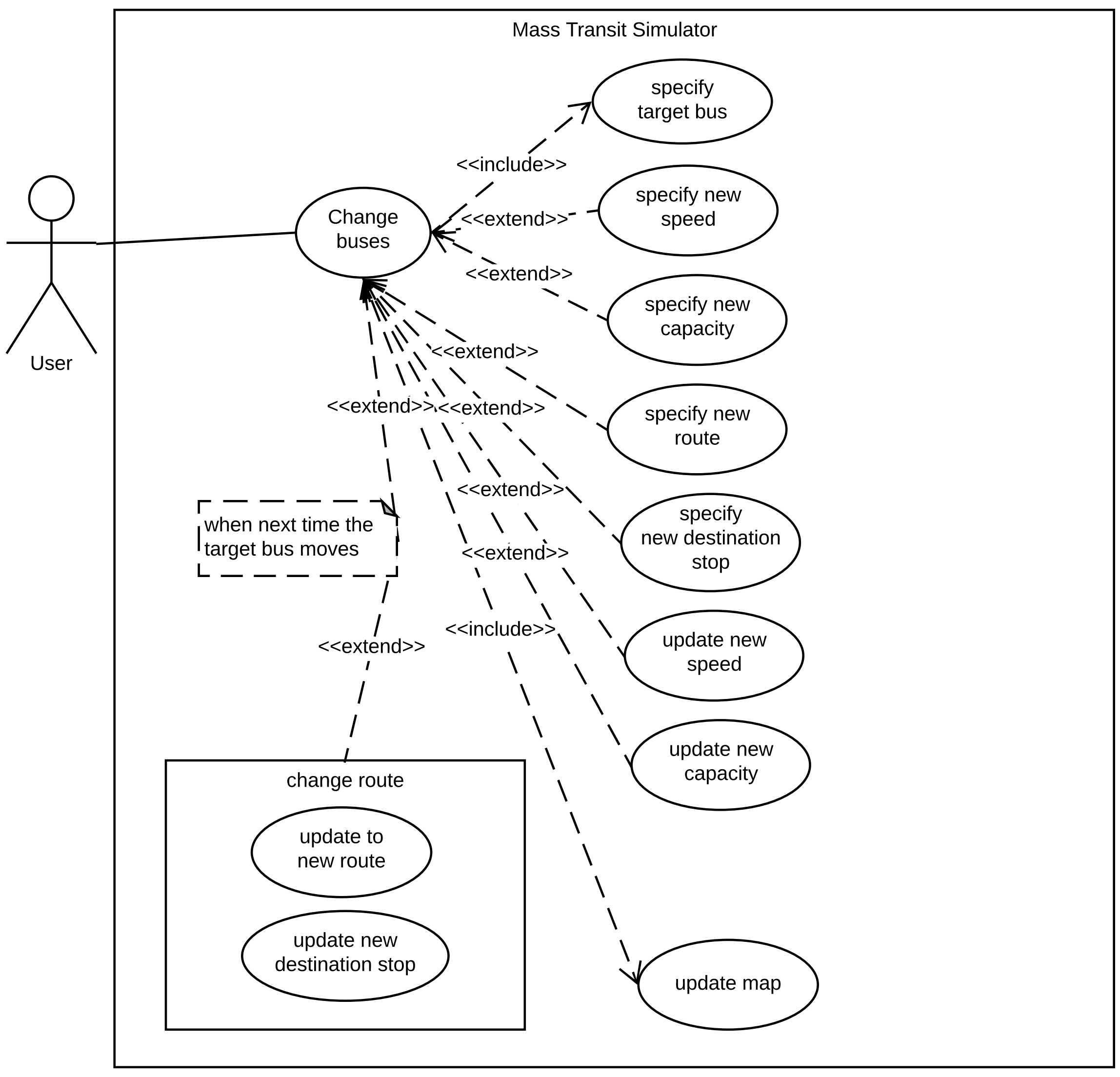
GROUP: A9-11 - CS-6310-O01 - ASSIGNMENT 09
SEQUENCE DIAGRAM RESET BUS (UML 2.0)

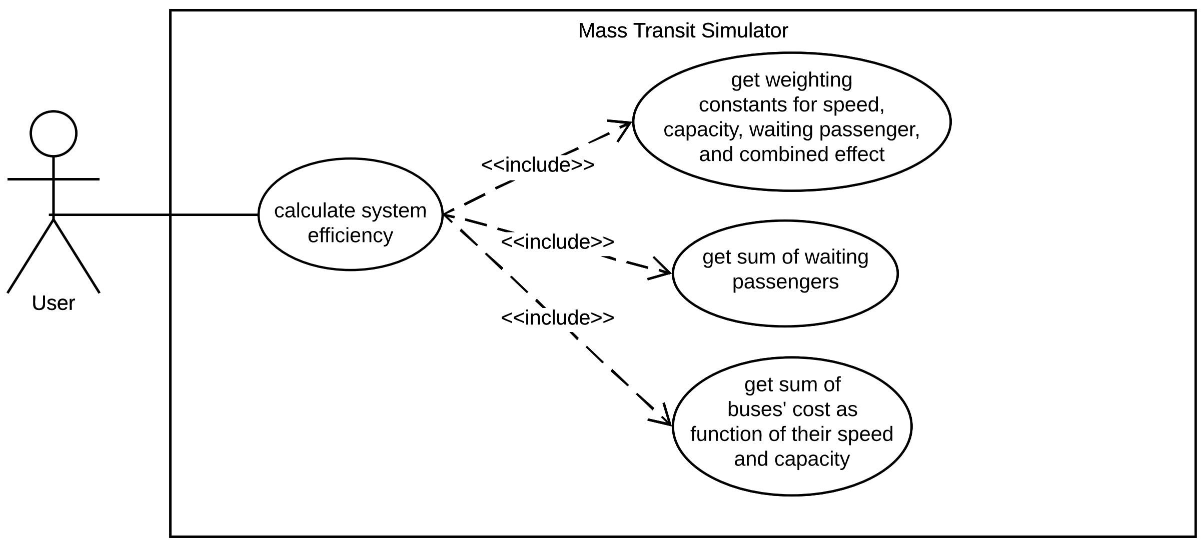
THIS DIAGRAM OMMITED MOST OF THE SYSTEM'S VISUAL ELEMENTS, WE ONLY REPRENSENTED A FEW FOR CLARITY

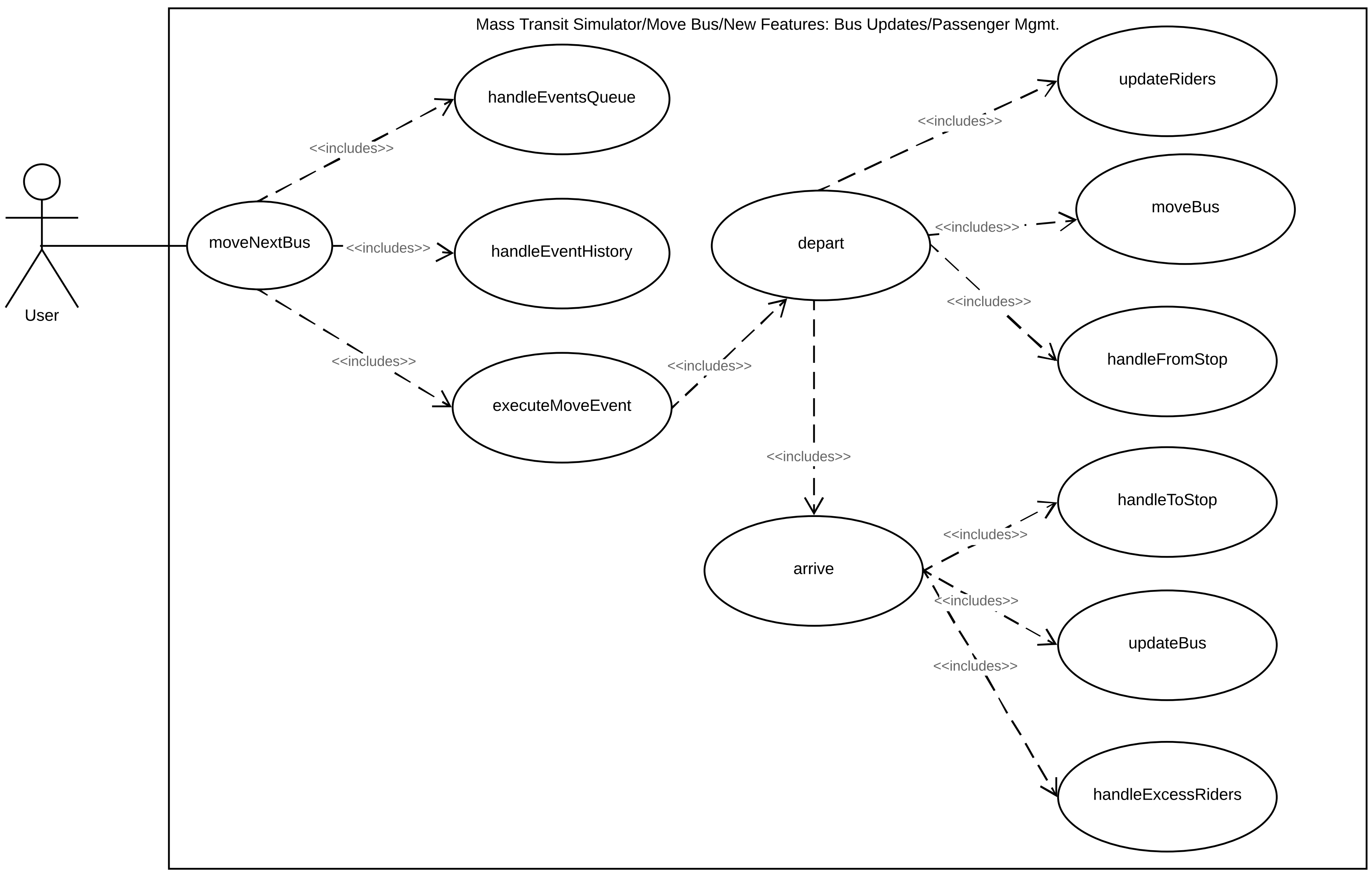


GROUP: A9-11 - CS-6310-O01 - ASSIGNMENT 09 USE CASE DIAGRAMS (UML 2.0)


Mass Transit Simulator
User
Reset Bus
<<include>>
<<include>>
<<include>>
Clear model object collections
Setup scenario anew from configuration and distribution file
update map
Replay
<<include>>
<<include>>
<<include>>
pop one EventState from history stack
Update replay bus's position from EventState data
Update map




Mass Transit Simulator
User
calculate system efficiency
<<include>>
<<include>>
<<include>>
get weighting constants for speed, capacity, waiting passenger, and combined effect
get sum of waiting passengers
get sum of buses' cost as function of their speed and capacity



Package manager

Class Summary

Scenario

Scenario class represents a simulation system of a particular bus, route, and stop configuration It holds, organizes, manages all model elements, and associated event collections in the system

Simulate

Simulate is the main managing class to launch the graphic user interface that contains map canvas and command buttons to perform the following main functions:

- Reset buses
- Change buses
- Move buses
- Calculate system efficiency

---

manager

Class Scenario

```
java.lang.Object
   |
   +--manager.Scenario
```

< Fields > < Constructors > < Methods >

---

public class **Scenario**
extends java.lang.Object

Scenario class represents a simulation system of a particular bus, route, and stop configuration It holds, organizes, manages all model elements, and associated event collections in the system

Author:

Team A9-11

Fields

buses

`public java.util.HashMap buses`

---

configFile

`public java.lang.String configFile`

---

depots

```
public java.util.HashMap depots
```

distFile

```
public java.lang.String distFile
```

events

```
public java.util.PriorityQueue events
```

name

```
public java.lang.String name
```

routes

```
public java.util.HashMap routes
```

stops

```
public java.util.HashMap stops
```

verbose

```
public boolean verbose
```

Constructors

Scenario

```
public  Scenario()
```

Methods

addNewEvent

```
public boolean addNewEvent(events.Event evt)
```

Add a new event to the event listing of the system

Parameters:

evt -

Returns:

---

addToEventHistory

```
public void addToEventHistory(events.Event evt)
```

---

addToEventStateHistory

```
public events.EventState addToEventStateHistory(events.Event evt)
```

create and add an EventState onto the event history stack The EventState is created based on a scheduled event object and additional bus and route information

Parameters:

evt - an event object

Returns:

the newly create EventState history object

---

busCost

```
public double busCost()
```

Return bus cost factor as a function of buses' speed and capacity

---

findBusEvent

```
public events.Event findBusEvent(entities.Bus b)
```

Find the next schedule event involves a given bus

Parameters:

b - the target Bus object

Returns:

the next scheduled Event object

---

getKBuses

```
public double getKBuses()
```

---

getKCapacity

```
public double getKCapacity()
```

---

getKCombined

```
public double getKCombined()
```

---

getKSpeed

```
public double getKSpeed()
```

---

getKWaiting

```
public double getKWaiting()
```

---

getStatus

```
public graphics.Status getStatus()
```

---

moveNextBus

```
public void moveNextBus()
```

Perform move bus event, one event at a time

---

removeEvent

```
public boolean removeEvent(events.Event evt)
```

Remove an event from system's event listing

Parameters:

evt - an event to be removed

Returns:

true/false for success or failure

---

replay

```
public events.EventState replay()
```

Perform rewind function by restoring bus position to previous stop and reset previous bus stop's waiting passenger pool

Returns:

the last used EventState object

---

reset

```
public void reset()
```

Reset the simulation environment

---

rewind

```
public void rewind()
```

---

setKBuses

```
public void setKBuses(double kBuses)
```

---

setKCombined

```
public void setKCombined(double kCombined)
```

---

setKSpeed

```
public void setKSpeed(double kSpeed)
```

---

setKWaiting

```
public void setKWaiting(double kWaiting)
```

---

setKcapacity

```
public void setKcapacity(double kCapacity)
```

---

setup

```
public boolean setup(java.lang.String configFile,
                     java.lang.String distFile)
```

setup creates the collection of model objects from configuration file

Parameters:

configFile - contains instructions for setting up the simulation environment
distFile - contains stop specific passenger distribution parameters

Returns:

true if setup is successful and false if not

---

systemEfficiency

```
public double systemEfficiency()
```

Return system efficiency composite index

---

updateBus

```
public java.lang.String updateBus(entities.Bus b,
                                  int newSpeed,
                                  int newCapacity,
                                  entities.Route newRoute,
                                  entities.Stop newStop)
```

Update bus information via bus argument

Parameters:

b - target Bus object
newSpeed - new speed value
newCapacity - new passenger capacity value
newRoute - new Route for the bus
newStop - new destination Stop

Returns:

update status message string

---

updateBus

```
public void updateBus(int id,
                      entities.Bus newBus)
```

Replaces original bus with a new bus object after route change so as to start fresh

Parameters:

id - original bus id
newBus - new bus object on new route

---

updateMoveBusEvent

```
public java.lang.String updateMoveBusEvent(entities.Bus b,
                                           entities.Route newRoute,
                                           entities.Stop newStop)
```

Update next schedule event involving target bus with new route and stop information

Parameters:

b - target Bus object
newRoute - new bus Route
newStop - new bus destination Stop

Returns:

update status message string

---

waitingPassengers

```
public double waitingPassengers()
```

Return sum of waiting passengers at all stops

---

manager

Class Simulate

```
java.lang.Object
    |
    +--manager.Simulate
```

All Implemented Interfaces:

java.awt.event.ActionListener

< Fields > < Constructors > < Methods >

public class **Simulate**
extends java.lang.Object
implements java.awt.event.ActionListener

Simulate is the main managing class to launch the graphic user interface that contains map canvas and command buttons to perform the following main functions:

- Reset buses
- Change buses
- Move buses
- Calculate system efficiency

Author:

Team A9-11

Fields

canvas

```
public graphics.MTSCanvas canvas
```

scen

```
public static Scenario scen
```

sim

```
public static Simulate sim
```

Constructors

Simulate

```
public  Simulate()
```

Methods

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent e)
```

action responder

---

main

```
public static void main(java.lang.String[] args)
```

---

repaint

```
public void repaint(java.awt.Graphics g)
```

main painter function to update map canvas

Parameters:

g - a Graphic object

Class BusProperty

```
java.lang.Object
   |
   +--java.awt.Component
         |
         +--java.awt.Container
               |
               +--java.awt.Window
                     |
                     +--java.awt.Frame
                           |
                           +--javax.swing.JFrame
                                 |
                                 +--graphics.BusProperty
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.TransferHandler.HasGetTransferHandler, javax.swing.WindowConstants

< Constructors > < Methods >

public class **BusProperty**
extends javax.swing.JFrame

BusProperty is a property editor form for changing the following bus attributes:

- speed
- capacity
- route
- next destination stop

Author:

Team A9-11

Constructors

BusProperty

```
public  BusProperty()
```

Create the frame.

Methods

getComponentByName

```
public java.awt.Component getComponentByName(java.lang.String name)
```

main

`public static void **main**(java.lang.String[] args)`

Launch the application.

populateForm

`public void **populateForm**()`

populateForm retrieves listing of buses, routes, and stops in the system to setup dropdown lists for buses, routes, and stops.

When a bus is selected, then its route and corresponding stops will be listed When a route is selected, then its stops will be listed accordingly

Class MTSCanvas

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--javax.swing.JComponent
                    |
                    +--javax.swing.JPanel
                          |
                          +--graphics.MTSCanvas
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.TransferHandler.HasGetTransferHandler

< Constructors > < Methods >

public class **MTSCanvas**
extends javax.swing.JPanel

MTSCanvas is the main map canvas

Author:

Team A9-11

Constructors

MTSCanvas

```
public  MTSCanvas()
```

Methods

paintComponent

```
public void paintComponent(java.awt.Graphics g)
```

Overrides:

paintComponent in class javax.swing.JComponent

Class SystemEfficiency

```
java.lang.Object
   |
   +--java.awt.Component
         |
         +--java.awt.Container
               |
               +--java.awt.Window
                     |
                     +--java.awt.Frame
                           |
                           +--javax.swing.JFrame
                                 |
                                 +--graphics.SystemEfficiency
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.TransferHandler.HasGetTransferHandler, javax.swing.WindowConstants

< Constructors > < Methods >

public class **SystemEfficiency**
extends javax.swing.JFrame

SystemEfficiency form allows user to enter new K constants for the calculation of system efficiency

- k_bus_speed
- k_bus_capacity
- k_waiting_passengers
- k_buses
- k_combined

Users can change the above constants, then click the 'Calculate' button to calculate the system efficiency as a function of weighted sum of bus costs and waiting passengers.

Author:

Team A9-11

Constructors

SystemEfficiency

```
public  SystemEfficiency()
```

Create the frame.

Methods

main

```
public static void main(java.lang.String[] args)
```

Launch the application.

---

populateForm

```
public void populateForm()
```

Retrieve the K constants from the system and populate corresponding text fields

Package events

Interface Summary

iAction

Interface for event execution

Class Summary

Event

Event is the action the system will take in a given situation

- time
- type
- id

EventComparator

Used to compare events by time

EventState

Event State stores the state of each event

- routeId
- fromStopIdx
- ridersOn
- ridersWait

MoveBusEvent

Event that tracks bus movements

events

Class Event

```
java.lang.Object
  |
  +--events.Event
```

All Implemented Interfaces:

iAction

Direct Known Subclasses:

EventState, MoveBusEvent

< Constructors > < Methods >

public class **Event**
extends java.lang.Object
implements iAction

Event is the action the system will take in a given situation

- time
- type
- id

Author:

Team A9-11

Constructors

Event

```
public  Event()
```

Default Constructor to set time, type and id to default values

---

Event

```
public  Event(int time,
              java.lang.String type,
              int id)
```

Parameters:

time -
type -
id -

Methods

execute

```
public Event execute()
```

Default event execution

---

getId

```
public int getId()
```

---

getTime

```
public int getTime()
```

---

getType

```
public java.lang.String getType()
```

---

setId

```
public void setId(int id)
```

---

setTime

```
public void setTime(int time)
```

---

setType

```
public void setType(java.lang.String type)
```

---

toString

```
public java.lang.String toString()
```

The string of the event attributes

Overrides:

toString in class java.lang.Object

---

events

Class EventComparator

```
java.lang.Object
    |
    +--events.EventComparator
```

All Implemented Interfaces:

java.util.Comparator

< Constructors > < Methods >

---

public class **EventComparator**
extends java.lang.Object
implements java.util.Comparator

Used to compare events by time

Author:

Team A9-11

Constructors

EventComparator

```
public  EventComparator()
```

Methods

compare

```
public int compare(Event e1,
                   Event e2)
```

Overriding compare() method of Comparator for ascending order of time

---

events

Class EventState

```
java.lang.Object
    |
    +--Event
        |
        +--events.EventState
```

All Implemented Interfaces:

iAction

< Constructors > < Methods >

---

public class **EventState**
extends Event

Event State stores the state of each event

- routeId
- fromStopIdx
- ridersOn
- ridersWait

Author:

Team A9-11

Constructors

EventState

```
public  EventState(Event evt,
                   int routeId,
                   int fromStopIdx,
                   int ridersOn,
                   int ridersWait)
```

Parameters:

evt -
routeId -
fromStopIdx -
ridersOn -
ridersWait -

EventState

```
public  EventState(int time,
                   java.lang.String type,
                   int id,
                   int routeId,
                   int fromStopIdx,
                   int ridersOn,
                   int ridersWait)
```

Parameters:

time -
type -
id -
routeId -
fromStopIdx -
ridersOn -
ridersWait -

Methods

getChangeToRouteId

```
public int getChangeToRouteId()
```

---

getChangeToStopId

```
public int getChangeToStopId()
```

getFromStopIdx

```
public int getFromStopIdx()
```

getNextEvent

```
public Event getNextEvent()
```

getRidersOn

```
public int getRidersOn()
```

getRidersWait

```
public int getRidersWait()
```

getRouteId

```
public int getRouteId()
```

setChangeToRouteId

```
public void setChangeToRouteId(int routeId)
```

setChangeToStopId

```
public void setChangeToStopId(int stopId)
```

setFromStopIdx

```
public void setFromStopIdx(int fromStopIdx)
```

setNextEvent

```
public void setNextEvent(Event nextEvent)
```

set the next scheduled event

Parameters:

nextEvent -

setRidersOn

```
public void setRidersOn(int ridersOn)
```

setRidersWait

```
public void setRidersWait(int ridersWait)
```

setRouteId

```
public void setRouteId(int routeId)
```

events

Class MoveBusEvent

```
java.lang.Object
    |
    +--Event
        |
        +--events.MoveBusEvent
```

All Implemented Interfaces:

iAction

< Fields > < Constructors > < Methods >

public class **MoveBusEvent**
extends Event
implements iAction

Event that tracks bus movements

Author:

Team A9-11

Fields

bus

```
public entities.Bus bus
```

Constructors

MoveBusEvent

```
public  MoveBusEvent()
```

Default bus execution

MoveBusEvent

```
public  MoveBusEvent(int time,
                     java.lang.String type,
                     int id,
                     entities.Bus bus)
```

Parameters:

time -
type -
id -
bus -

Methods

createNewEvent

```
public Event createNewEvent()
```

---

execute

```
public Event execute()
```

Execute move bus event

Overrides:

execute in class Event

getChangeToRoute

```
public entities.Route getChangeToRoute()
```

getChangeToStop

```
public entities.Stop getChangeToStop()
```

setChangeToRoute

```
public void setChangeToRoute(entities.Route changeToRoute)
```

setChangeToStop

```
public void setChangeToStop(entities.Stop changeToStop)
```

toString

```
public java.lang.String toString()
```

Move Bus Event string with attributes

Overrides:

toString in class Event

events

Interface iAction

< Methods >

public interface **iAction**

Interface for event execution

Author:

Team A9-11

Methods

execute

```
public Event execute()
```

Execute event

Returns:

event

Package entities

Class Summary

Bus

This class implements the behavior of a Bus in the MASS Transit Simulation

MTSEntity

Implements the base behavior for Mass Transit Simulation Entity Classes

Riders

This class stores the probability distributions per stop

Route

Defines a route a bus must take

- id
- name
- number
- stops

Stop

This class represents the Stop entity in the Mass Transit Simulation

entities

Class Bus

```
java.lang.Object
  |
  +--MTSEntity
        |
        +--entities.Bus
```

< Fields > < Constructors > < Methods >

public class **Bus**
extends MTSEntity

This class implements the behavior of a Bus in the MASS Transit Simulation

Author:

Team A9-11

Fields

fuel

```
public int fuel
```

Constructors

Bus

```
public  Bus()
```

Default constructor

Bus

```
public  Bus(int busId,
            Route route,
            int loc,
            int passengerCount,
            int riderCapacity,
            int fuel,
            int fuelCapacity,
            int speed)
```

Parameters:

busId -
route -
loc -
passengerCount -
riderCapacity -
fuel -
fuelCapacity -
speed -

Methods

arrive

```
public void arrive()
```

This method implements the behavior when a bus arrives at it's next stop. The bus should handle the excess riders, in case a capacity change was executed Finally the next stop for the Bus should be set

---

arrive

```
public void arrive(Route newRoute,
                   Stop newStop)
```

This method implements the behavior when a bus arrives at it's next stop. The bus should handle the excess riders, in case a capacity change was executed Finally the next stop for the Bus should be set This methods accounts for the bus and stop changes

Parameters:

newRoute -
newStop -

---

depart

```
public void depart(int current_time)
```

Implements the behavior of the depart functionality. Executes the moveBus, which will calculate based on the stops navigated on the Route: the next stops, the passengers management, the logical time.

Parameters:

current_time -

---

depart

```
public void depart(int current_time,
                   Route newRoute,
                   Stop newStop)
```

Implements the behavior of the depart functionality. Executes the moveBus, which will calculate based on the stops navigated on the Route: the next stops, the passengers management, the logical time. This methods accounts for route and stop changes

Parameters:

current_time -
newRoute -
newStop -

---

getCurrentStopObj

```
public Stop getCurrentStopObj()
```

Returns:

---

getFromStopIdx

`public int getFromStopIdx()`

Returns:

---

getFuelCapacity

`public int getFuelCapacity()`

Returns:

---

getId

`public int getId()`

Returns:

---

getNextStop

`public java.lang.String getNextStop()`

Returns:

---

getNextTime

`public java.lang.String getNextTime()`

Returns:

---

getNumRiders

`public int getNumRiders()`

Returns:

---

getRiderCapacity

```
public int getRiderCapacity()
```

Returns:

getRidersWait

```
public int getRidersWait()
```

Returns:

getRoute

```
public Route getRoute()
```

Returns:

getSpeed

```
public int getSpeed()
```

Returns:

getStartRouteIdx

```
public int getStartRouteIdx()
```

Returns:

getStartStopIdx

```
public int getStartStopIdx()
```

Returns:

getStop

```
public Stop getStop(int idx)
```

Get the stop from the route

Parameters:

idx -

Returns:

---

getToStopIdx

```
public int getToStopIdx()
```

Returns:

---

getTravelTime

```
public int getTravelTime()
```

Returns:

---

isInService

```
public boolean isInService()
```

---

moveBus

```
public java.lang.String moveBus(int currentTime,
                                Route newRoute,
                                Stop newStop)
```

Calculates the distance, travel time, executes the rider management based on the stop from which the bus will be moving, and prints the bus status after move event is completed

Parameters:

currentTime -
newRoute -
newStop -

Returns:

---

setFromStopIdx

```
public void setFromStopIdx(int fromStopIdx)
```

Parameters:

fromStopIdx -

---

setFuelCapacity

```
public void setFuelCapacity(int capacityFuelMiles)
```

Parameters:

capacityFuelMiles -

---

setId

```
public void setId(int id)
```

Parameters:

id -

---

setNextStop

```
public void setNextStop(java.lang.String nextStop)
```

Parameters:

nextStop -

---

setNextTime

```
public void setNextTime(java.lang.String nextTime)
```

Parameters:

nextTime -

---

setRiderCapacity

```
public void setRiderCapacity(int capacity)
```

Parameters:

capacity -

setRiders

```
public void setRiders(int numRiders)
```

Parameters:

numRiders -

setRidersWait

```
public void setRidersWait(int riderWait)
```

Parameters:

riderWait -

setRoute

```
public void setRoute(Route route)
```

Parameters:

route -

setServiceStatus

```
public void setServiceStatus(boolean inService)
```

setSpeed

```
public void setSpeed(int speed)
```

Parameters:

speed -

setStartRouteIdx

```
public void setStartRouteIdx(int idx)
```

Parameters:

idx -

setStartStopIdx

```
public void setStartStopIdx(int idx)
```

Initializes the start stop

Parameters:

idx -

setToStopIdx

```
public void setToStopIdx(int toStopIdx)
```

Parameters:

toStopIdx -

toString

```
public java.lang.String toString()
```

the string of bus attributes

Overrides:

toString in class java.lang.Object

entities

Class MTSEntity

```
java.lang.Object
    |
    +--entities.MTSEntity
```

Direct Known Subclasses:

Bus, Riders, Route, Stop

< Constructors > < Methods >

public class **MTSEntity**
extends java.lang.Object

Implements the base behavior for Mass Transit Simulation Entity Classes

Author:

Team A9-11

Constructors

MTSEntity

```
public  MTSEntity()
```

Methods

getListener

```
public graphics.Logger getListener()
```

Getting logger object

Returns:

setListener

```
public void setListener(graphics.Logger toAdd)
```

Setting logger object

Parameters:

toAdd -

toStatus

```
public void toStatus(java.lang.String msg)
```

Prints the status message

Parameters:

msg -

entities

Class Riders

```
java.lang.Object
    |
    +--MTSEntity
        |
        +--entities.Riders
```

< Constructors > < Methods >

public class **Riders**
extends MTSEntity

This class stores the probability distributions per stop

Author:

Team A9-11

Constructors

Riders

```
public  Riders(int ridersArriveHigh,
               int ridersArriveLow,
               int ridersOffHigh,
               int ridersOffLow,
               int ridersOnHigh,
               int ridersOnLow,
               int ridersDepartHigh,
               int ridersDepartLow)
```

Constructor

Parameters:

ridersArriveHigh -
ridersArriveLow -
ridersOffHigh -
ridersOffLow -
ridersOnHigh -
ridersOnLow -
ridersDepartHigh -
ridersDepartLow -

Methods

calcArrival

```
public int calcArrival()
```

Returns:

calcDeparture

```
public int calcDeparture()
```

Returns:

calcOff

```
public int calcOff()
```

Returns:

calcOn

```
public int calcOn()
```

Returns:

getRidersArriveHigh

```
public int getRidersArriveHigh()
```

Returns:

getRidersArriveLow

```
public int getRidersArriveLow()
```

Returns:

getRidersDepartHigh

```
public int getRidersDepartHigh()
```

Returns:

getRidersDepartLow

```
public int getRidersDepartLow()
```

Returns:

getRidersOffHigh

```
public int getRidersOffHigh()
```

Returns:

getRidersOffLow

```
public int getRidersOffLow()
```

Returns:

getRidersOnHigh

```
public int getRidersOnHigh()
```

Returns:

getRidersOnLow

```
public int getRidersOnLow()
```

Returns:

setRidersArriveHigh

```
public void setRidersArriveHigh(int ridersArriveHigh)
```

---

setRidersArriveLow

```
public void setRidersArriveLow(int ridersArriveLow)
```

Parameters:

ridersArriveLow -

---

setRidersDepartHigh

```
public void setRidersDepartHigh(int ridersDepartHigh)
```

Parameters:

ridersDepartHigh -

---

setRidersDepartLow

```
public void setRidersDepartLow(int ridersDepartLow)
```

Parameters:

ridersDepartLow -

---

setRidersOffHigh

```
public void setRidersOffHigh(int ridersOffHigh)
```

Parameters:

ridersOffHigh -

---

setRidersOffLow

```
public void setRidersOffLow(int ridersOffLow)
```

Parameters:

ridersOffLow -

setRidersOnHigh

```
public void setRidersOnHigh(int ridersOnHigh)
```

Parameters:

ridersOnHigh -

setRidersOnLow

```
public void setRidersOnLow(int ridersOnLow)
```

Parameters:

ridersOnLow -

toString

```
public java.lang.String toString()
```

Overrides:

toString in class java.lang.Object

entities

Class Route

```
java.lang.Object
  |
  +--MTSEntity
        |
        +--entities.Route
```

< Constructors > < Methods >

public class **Route**
extends MTSEntity

Defines a route a bus must take

- id
- name
- number
- stops

Author:

Team A9-11

Constructors

Route

```
public  Route()
```

Route

```
public  Route(int id,
              int number,
              java.lang.String name)
```

Parameters:

id -
number -
name -

Methods

calcDistance

```
public static double calcDistance(Stop stop1,
                                  Stop stop2)
```

Distance calculations between stops

Parameters:

stop1 -
stop2 -

Returns:

---

calcDistance

```
public double calcDistance(int idx1,
                           int idx2)
```

Distance calculation between stops indexes

Parameters:

idx1 -
idx2 -

Returns:

---

calcTravelTime

```
public static int calcTravelTime(double distance,
                                 double speed)
```

Travel time calculations

Parameters:

distance -
speed -

Returns:

---

calcTravelTime

```
public static int calcTravelTime(double distance,
                                 int speed)
```

Travel time calculations

Parameters:

distance -
speed -

Returns:

---

getId

```
public int getId()
```

Returns:

---

getName

```
public java.lang.String getName()
```

Returns:

---

getNumber

```
public int getNumber()
```

Returns:

---

getStop

`public Stop getStop(int idx)`

Get the stop based on stop index

Parameters:

idx -

Returns:

---

getStopIndex

`public int getStopIndex(Stop s)`

Get index based on Stop

Parameters:

`s` -

Returns:

---

getStops

`public java.util.List getStops()`

Returns:

---

nextStopIx

`public int nextStopIx(int idx)`

Get the stop based on stop index

Parameters:

idx -

Returns:

---

setId

`public void setId(int id)`

Parameters:

id -

setName

```
public void setName(java.lang.String name)
```

Parameters:

name -

setNumber

```
public void setNumber(int number)
```

Parameters:

number -

setStops

```
public void setStops(java.util.List stops)
```

Parameters:

stops -

toString

```
public java.lang.String toString()
```

Overrides:

toString in class java.lang.Object

entities

Class Stop

```
java.lang.Object
    |
    +--MTSEntity
        |
        +--entities.Stop
```

< Constructors > < Methods >

public class **Stop**
extends MTSEntity

This class represents the Stop entity in the Mass Transit Simulation

Author:

Team A9-11

Constructors

Stop

```
public  Stop(int id,
             java.lang.String name,
             java.lang.Integer riders,
             java.lang.Double latitude,
             java.lang.Double longitude,
             Riders distLimit)
```

Parameters:

id -
name -
riders -
latitude -
longitude -
distLimit -

Methods

addBus

```
public void addBus(int id,
                   Bus bus)
```

Parameters:

id -
bus -

---

addExcessRidersToWait

```
public void addExcessRidersToWait(java.lang.Integer excessRiders)
```

Add the excess passengers to the wait group at the stop

Parameters:

excessRiders -

---

getBuses

```
public java.util.HashMap getBuses()
```

Returns:

---

getDistribution

```
public Riders getDistribution()
```

Returns:

---

getId

```
public int getId()
```

Returns:

---

getLatitude

```
public double getLatitude()
```

Returns:

---

getLongitude

```
public double getLongitude()
```

Returns:

---

getName

```
public java.lang.String getName()
```

Returns:

---

getRiders

```
public int getRiders()
```

Returns:

---

getRidersTransfer

```
public int getRidersTransfer()
```

Returns:

---

getRidersWait

```
public int getRidersWait()
```

Returns:

---

removeBus

```
public void removeBus(int id)
```

Parameters:

id -

---

setBuses

```
public void setBuses(java.util.HashMap buses)
```

Parameters:

buses -

---

setDistribution

```
public void setDistribution(Riders distribution)
```

Parameters:

distribution -

setId

```
public void setId(int id)
```

Parameters:

id -

setLatitude

```
public void setLatitude(double latitude)
```

Parameters:

latitude -

setLongitude

```
public void setLongitude(double longitude)
```

Parameters:

longitude -

setName

```
public void setName(java.lang.String name)
```

Parameters:

name -

setRiders

```
public void setRiders(java.lang.Integer riders)
```

Parameters:

riders -

setRidersTransfer

```
public void setRidersTransfer(java.lang.Integer ridersTransfer)
```

Parameters:

ridersTransfer -

---

setRidersWait

```
public void setRidersWait(java.lang.Integer ridersWait)
```

Parameters:

ridersWait -

---

toString

```
public java.lang.String toString()
```

The string of the stop attributes

Overrides:

toString in class java.lang.Object

---

updateDepatureRiders

```
public void updateDepatureRiders()
```

Update waiting group based on the riders that either decide to depart the stop or arrived at their destination

---

updateWaitRidersFromOffRider

```
public java.lang.Integer updateWaitRidersFromOffRider(java.lang.Integer
maxRiderCanGetOff)
```

Updates the waiting group and based on the arrivals

Parameters:

maxRiderCanGetOff -

Returns:

Riders that can get off

---

updateWaitRidersFromOnRiders

```
public java.lang.Integer updateWaitRidersFromOnRiders(java.lang.Integer
busAvailableSeats)
```

Update the waiting group, based on the riders that can get on the bus

Parameters:

busAvailableSeats -

Returns:

Riders that can get on the bus