edu.monash.fit2099.engine

Location

Exit

MoveActorAction

Item

game

Actor

Player

game.dinosaurs

game.behaviours

1..4

<<Interface>>
Behaviour

FeedAction

Dinosaur

1

AttackBehaviour

HungryBehaviour

HornyBehaviour

WanderBehaviour

FollowBehaviour

Egg

BabyDino

Stegosaur

Brachiosaur

Allosaur

BabyStegosaur

BabyBrachiosaur

BabyAllosaur

edu.monash.fit2099

engine

WeaponItem

Item

Action

Ground

game.actions

BuyAction

PickFruitAction

game.items

LaserGun

PortableItem

game.grounds

VendingMachine

Fruit

Tree

Bush

Dirt

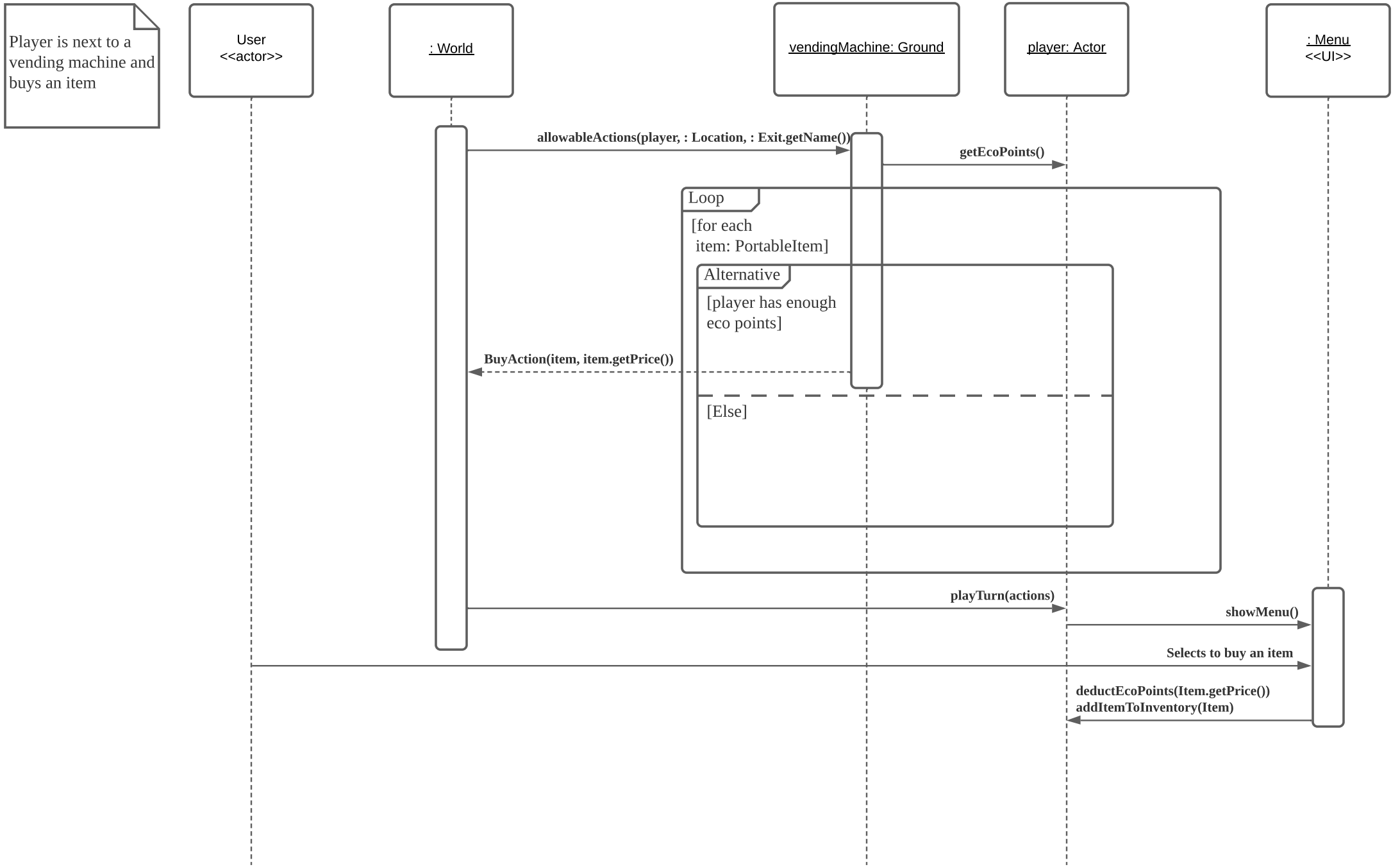
Egg

MealKit

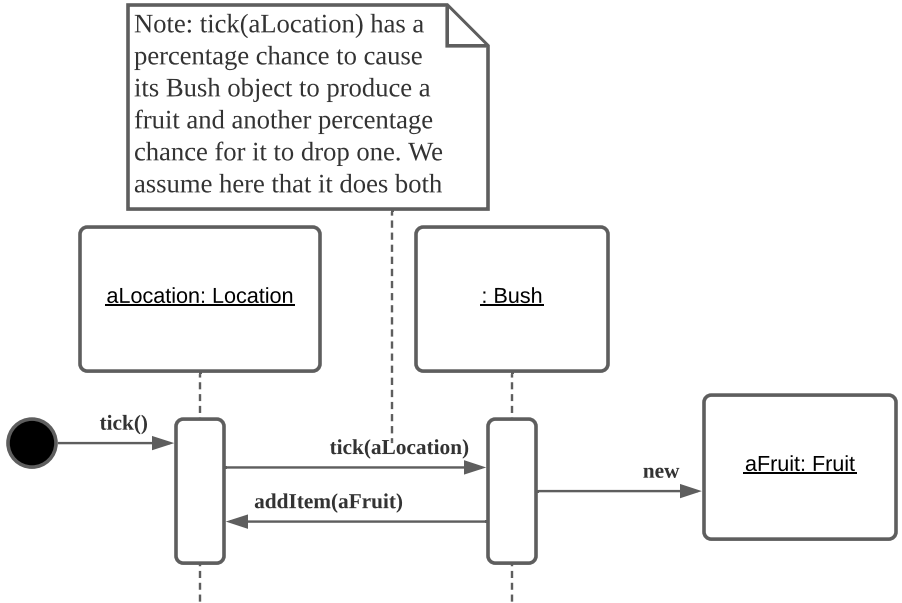
game.dinosaurs

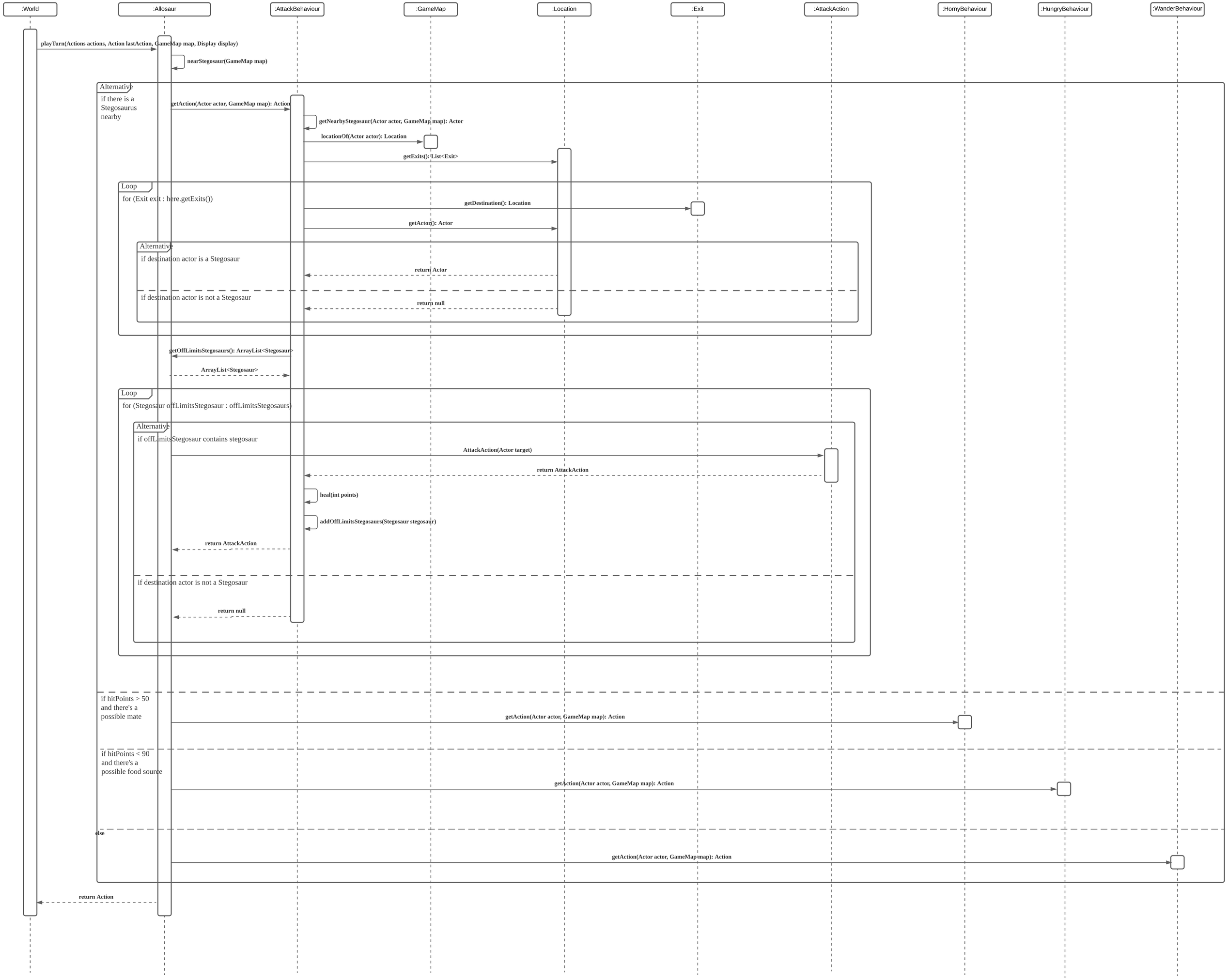
Branchiosaur

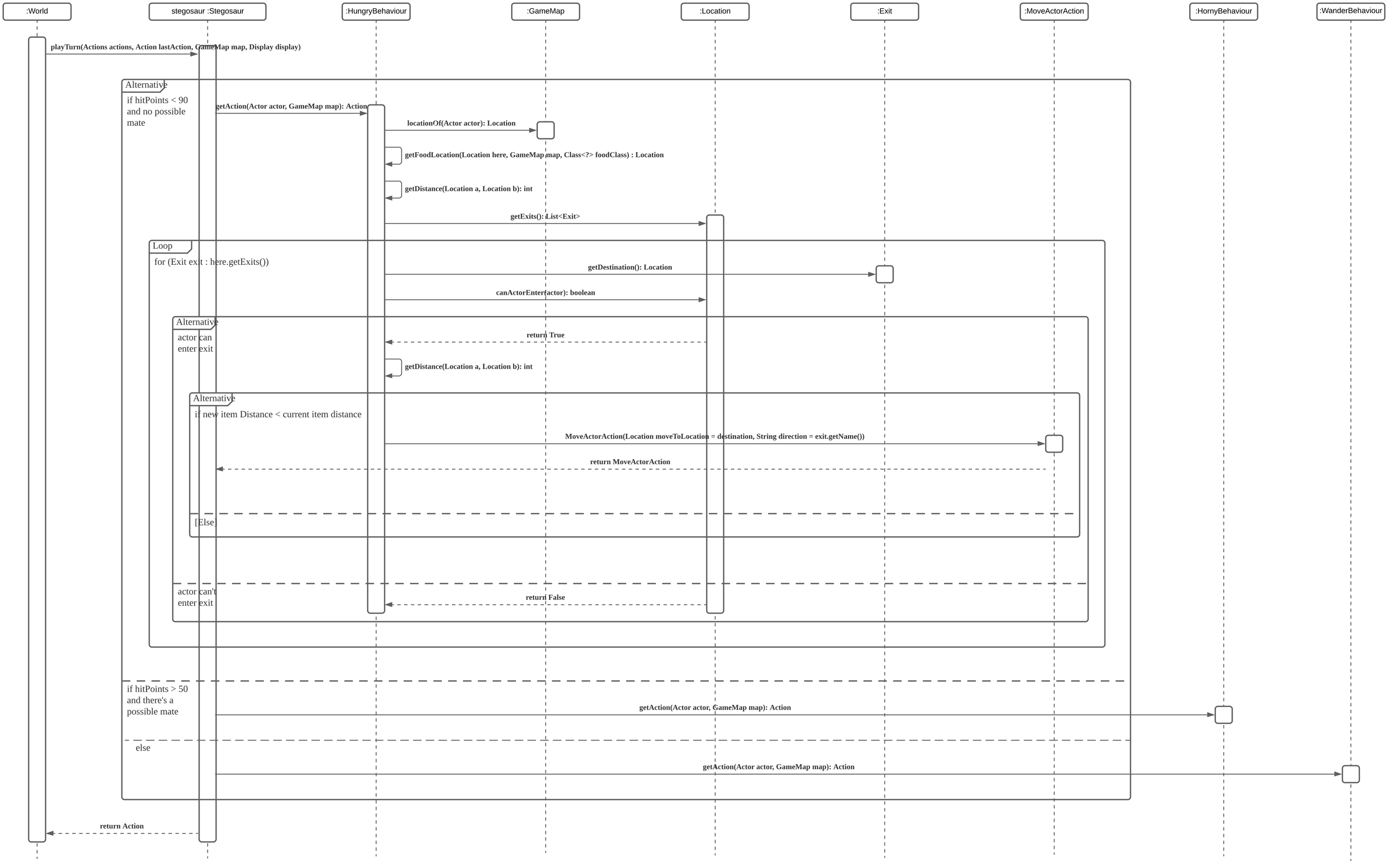
Stegosaur


Player is next to a vending machine and buys an item
User
<<actor>>
: World
vendingMachine: Ground
player: Actor
: Menu
<<UI>>
allowableActions(player, : Location, : Exit.getName())
getEcoPoints()
Loop
[for each item: PortableItem]
Alternative
[player has enough eco points]
BuyAction(item, item.getPrice())
[Else]
playTurn(actions)
showMenu()
Selects to buy an item
deductEcoPoints(Item.getPrice())
addItemToInventory(Item)

Bush produces and drops a fruit
(The process with a Tree is the same, just using different probabilities)




:World
:Allosaur
:AttackBehaviour
:GameMap
:Location
:Exit
:AttackAction
:HornyBehaviour
:HungryBehaviour
:WanderBehaviour
playTurn(Actions actions, Action lastAction, GameMap map, Display display)
nearStegosaur(GameMap map)
Alternative
if there is a Stegosaurus nearby
getAction(Actor actor, GameMap map): Action
getNearbyStegosaur(Actor actor, GameMap map): Actor
locationOf(Actor actor): Location
getExits(): List<Exit>
Loop
for (Exit exit : here.getExits())
getDestination(): Location
getActor(): Actor
Alternative
if destination actor is a Stegosaur
return Actor
if destination actor is not a Stegosaur
return null
getOffLimitsStegosaurs(): ArrayList<Stegosaur>
ArrayList<Stegosaur>
Loop
for (Stegosaur offLimitsStegosaur : offLimitsStegosaurs)
Alternative
if offLimitsStegosaur contains stegosaur
AttackAction(Actor target)
return AttackAction
heal(int points)
addOffLimitsStegosaurs(Stegosaur stegosaur)
return AttackAction
if destination actor is not a Stegosaur
return null
if hitPoints > 50 and there's a possible mate
getAction(Actor actor, GameMap map): Action
if hitPoints < 90 and there's a possible food source
getAction(Actor actor, GameMap map): Action
else
getAction(Actor actor, GameMap map): Action
return Action


:World
stegosaur :Stegosaur
:HungryBehaviour
:GameMap
:Location
:Exit
:MoveActorAction
:HornyBehaviour
:WanderBehaviour
playTurn(Actions actions, Action lastAction, GameMap map, Display display)
Alternative
if hitPoints < 90 and no possible mate
getAction(Actor actor, GameMap map): Action
locationOf(Actor actor): Location
getFoodLocation(Location here, GameMap map, Class<?> foodClass) : Location
getDistance(Location a, Location b): int
getExits(): List<Exit>
Loop
for (Exit exit : here.getExits())
getDestination(): Location
canActorEnter(actor): boolean
Alternative
actor can enter exit
return True
getDistance(Location a, Location b): int
Alternative
if new item Distance < current item distance
MoveActorAction(Location moveToLocation = destination, String direction = exit.getName())
return MoveActorAction
[Else]
actor can't enter exit
return False
if hitPoints > 50 and there's a possible mate
getAction(Actor actor, GameMap map): Action
else
getAction(Actor actor, GameMap map): Action
return Action