**Homework 4 Design Doc**

Dylan Whitehead

**Components:**

1. Cache Spawner
2. Cache
3. Home
4. AI controller
5. AI player
6. AI Behavior Tree

**1. Cache Spawner Component**

Name: CacheController

Unreal Type: Actor

Description: The actor responsible for creating/respawning all instances of the cache during the runtime of the game. This component is designed as to take responsibility away from the individual caches regarding behavior such as spawn location/amount/timing.

Data Structures:

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Default | Name | Description |
| int | 0 | cacheAmtMin | The minimum amount of value each cache should be spawned with. |
| int | 0 | cacheAmtMax | The maximum amount of value each cache should be spawned with. |
| Int | 0 | spawnAmtMin | The minimum amount of caches that should be spawned. |
| Int | 0 | spawnAmtMax | The maximum amount of caches that should be spawned. |
| Vector Array | 5 set locations on map | spawnLocations | An array of vectors holding all locations for caches to be spawned at. Should never be changed. |
| Vector  Array | empty | spawnLocationsTemp | An array of vectors holding all locations for caches to be spawned at. Will be subtracted from to avoid spawn duplication. |
|  |  |  |  |



Algorithms:

void SpawnLocation()

spawnLocationsTemp = spawnLocations

for(int i = 0; rand (between spawnAmtMin and spawnAmtMax); i++)

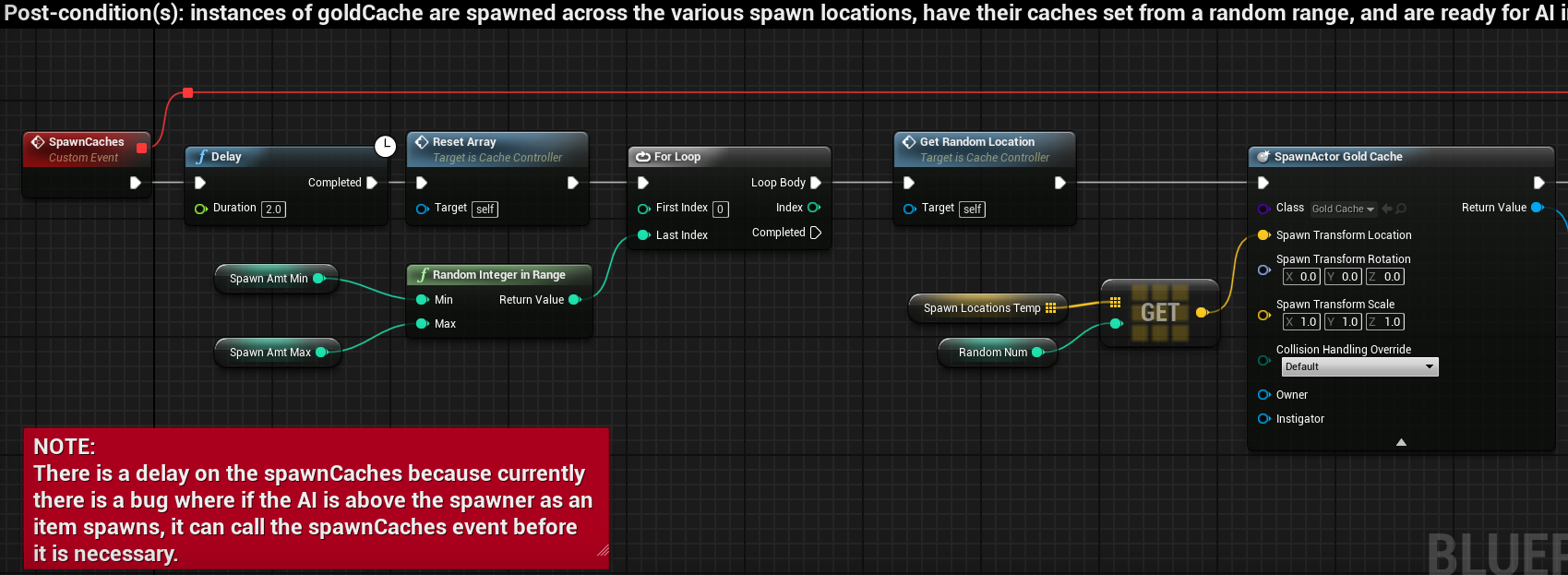
spawnActor(set rand number for the cache amount, set scale size)

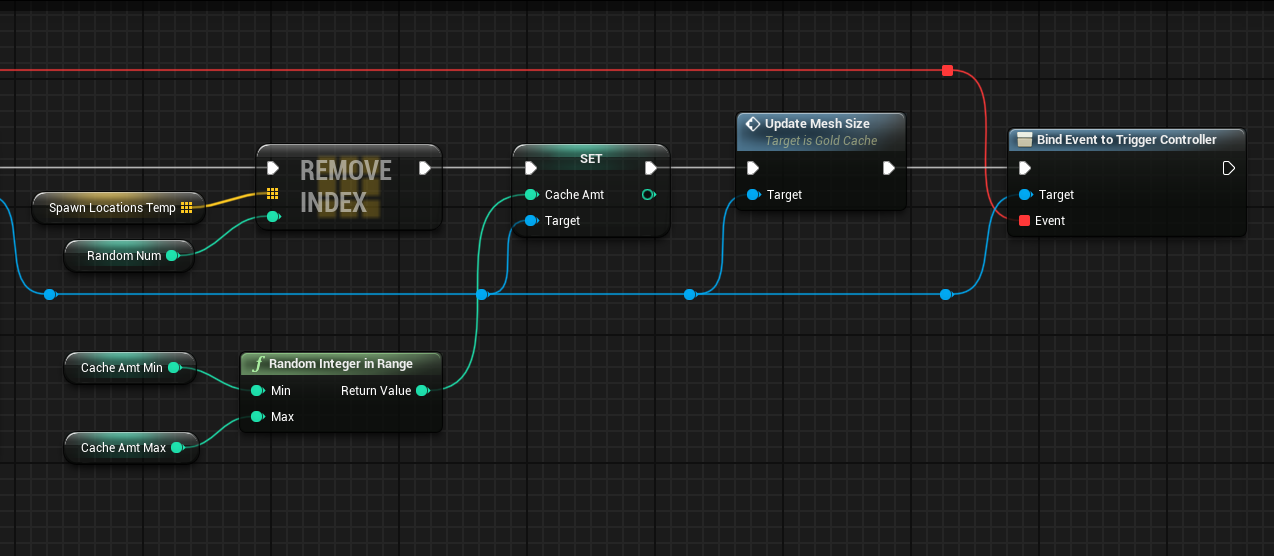
randomNum = rand(between 0 and spawnLocationsTemp.len-1)

SetActorLocation(spawnLocationsTemp[rand(randomNum)])

SpawnLocations.RemoveIndex(randomNum)

In Unreal:





Relation to other components:

Spawns the caches, attaches the cache object’s dispatcher node to the spawn event. All computation for if the spawn should be called in the first place takes place in the cache calling the dispatcher, not this component.

**2. Cache Component**

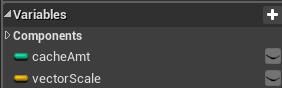
Name: Cache

Unreal Type: Actor

Description: The actor responsible for transferring “points” to the AI player. Does very little besides get interacted with upon colliding with the AI player and then get destroyed when all of the cache is “mined”.

Data Structures:

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Default | Name | Description |
| int | 0 | cacheAmt | The amount of value the cache has at any given point. |
| vector | (1,1,1) | vectorScaleBy | The vector in which the mesh is scaled by that will be multiplied by the cacheAmt upon creation and every time the cacheAmt changes. |



Algorithms:

Void ActorBeginOverlap(actorType actor\*)

Cast actor to AIPlayer

Player.hasgold(true)

CacheAmt -= 1

If(CacheAmt == 0)

If(Any other cache exists)

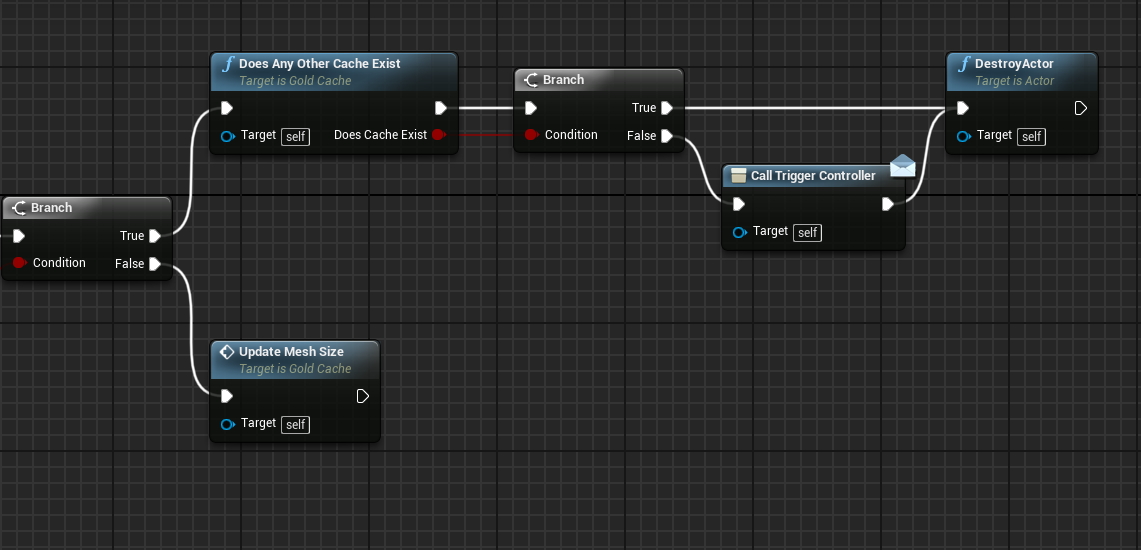
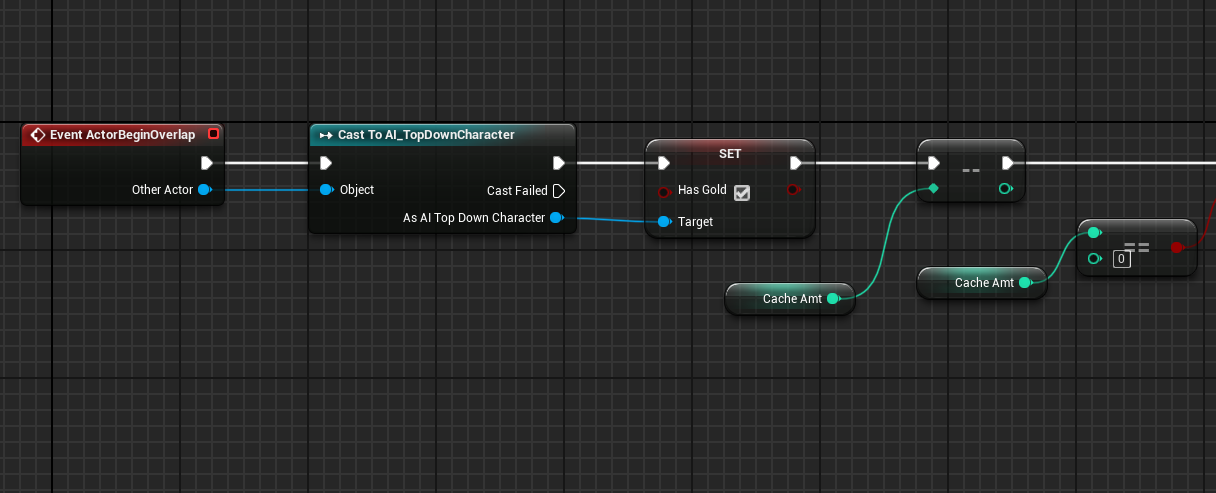
DispatchCacheControllerSpawn()

Destroy

Else

Mesh.Transform(CacheAmt \* vectorScaleBy)

In Unreal:



Relation to other components:

CACHE CONTROLLER: The cache uses the functionality of the cache controller via a dispatcher that is triggered only if the cache is empty and no other caches exist.

AI PLAYER: The cache checks for collision with the AI player. If a collision occurs, the object is cast to the AI player and if successful the player is now “holding” a value correlating to the cache to return to its home.

**3. Home Component**

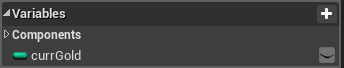
Name: AIHome

Unreal Type: Actor

Description: The actor that acts as a hub for the AI player. The AI player should return here every time they have taken an item from one of the caches. Every time an AI player returns, if they are carrying an item, a visual representation should be made that an increase in total value was made.

Data Structures:

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Default | Name | Description |
| int | 0 | currGold | The value pertaining to the AI player’s current gold. This value is incremented by 1 every time the AI player returns and is holding a gold from one of the caches. |



Algorithms:

Void ActorBeginOverlap(actorType actor\*)

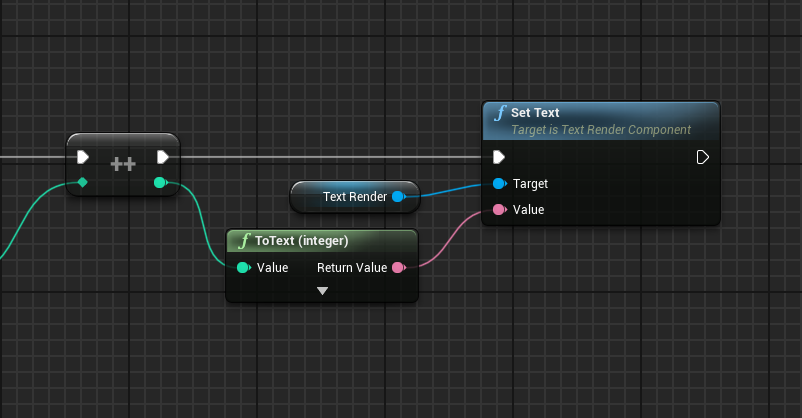
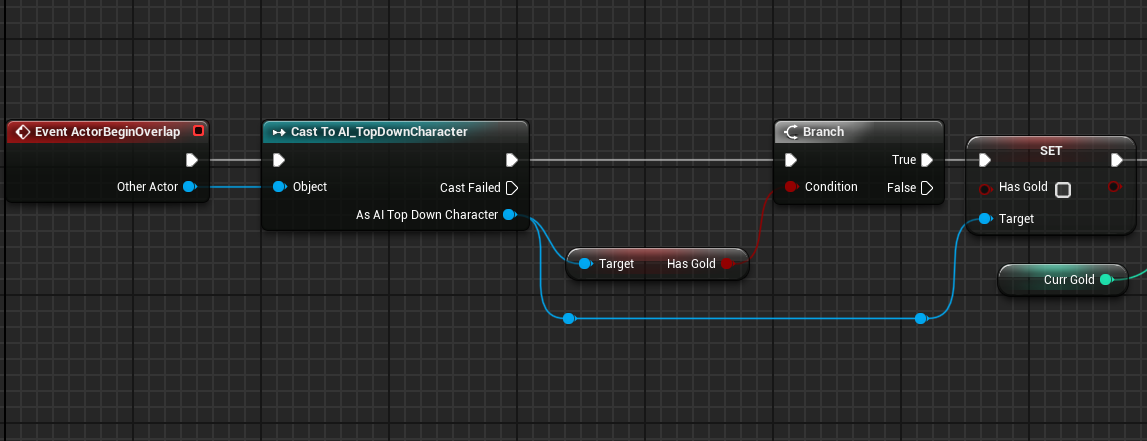
Cast actor to AIPlayer

If(AIPlayer has gold)

AiPlayer.SetHasGold(false)

currGold += 1

textRender.SetText(currGold)

In Unreal:

Relation to other components:

AI PLAYER: Checks for collision and sets the AI Player’s hasGold variable to false if collision is made and AIPlayer has gold

**4. AI Controller Component**

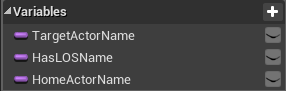
Name: AIC\_TopDownCharacter

Unreal Type: AIController

Description: Controller class for the AI Player. Acts as an intermediary between the behavior tree’s blackboard and the AI Player.

Data Structures:

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Default | Name | Description |
| Name | TargetActor | TargetActorName | Name of the blackboard key used to set the targetActor |
| Name | HasLOS | HasLOSName | Name of the blackboard key used to set hasLOS |
| Name | HomeActor | HomeActorName | Name of the blackboard key used to set homeActor |



Algorithms:

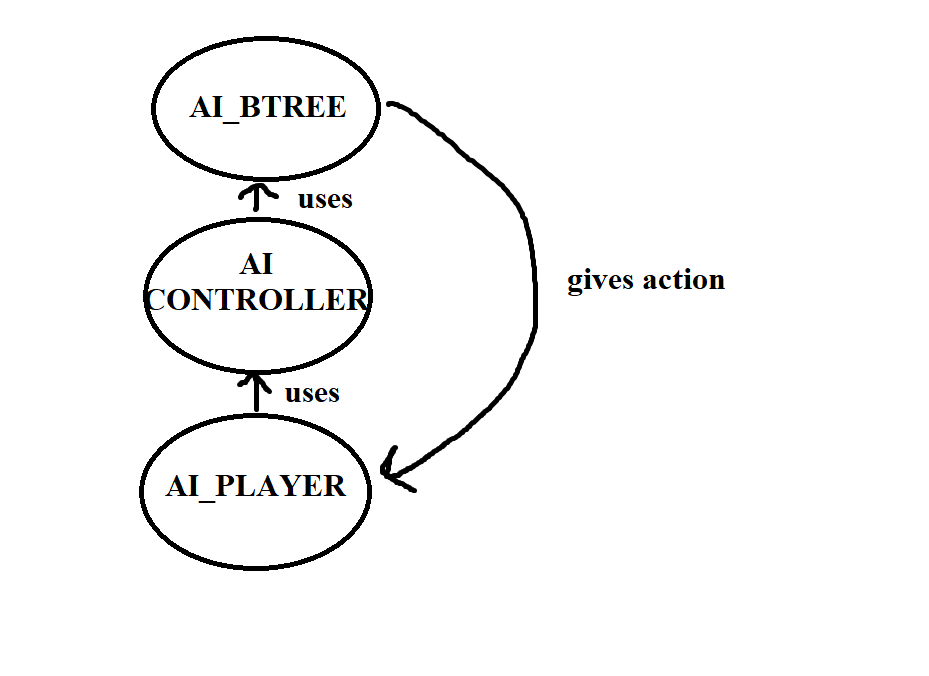
All behavior in the AIC\_TopDownCharacter is defined in the following way:

Take input from function header, set variable name in the blackboard with the input and the key listed above.

Relation to other components:

AI PLAYER: The AI Player uses the various functions in the AIC to transfer information to the blackboard.

BEHAVIOR TREE: The AI Controller transfers data over to the blackboard to be used for defining the AI player’s current behavior.



**5. AI Player Component**

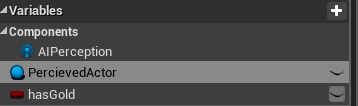
Name: AI\_TopDownCharacter

Unreal Type: Character

Description: The primary way in which the AI player is defined. All behavior regarding the visualization of the character model, the sight perception, and the hitbox are defined here. This should also include all variables that need to be kept for specific instances in case there should be any additional ai agents developed.

Data Structures:

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Default | Name | Description |
| bool | false | hasGold | Simple Boolean flag to determine if the AI is currently holding any gold. This is the primary and only way to determine what the AI is holding. |
| Actor | null | perceivedActor | The actor colliding with the sight perception. |



Algorithms:

Void OnTargetPerceptionUpdated(actorType actor\*)

Cast actor to cache

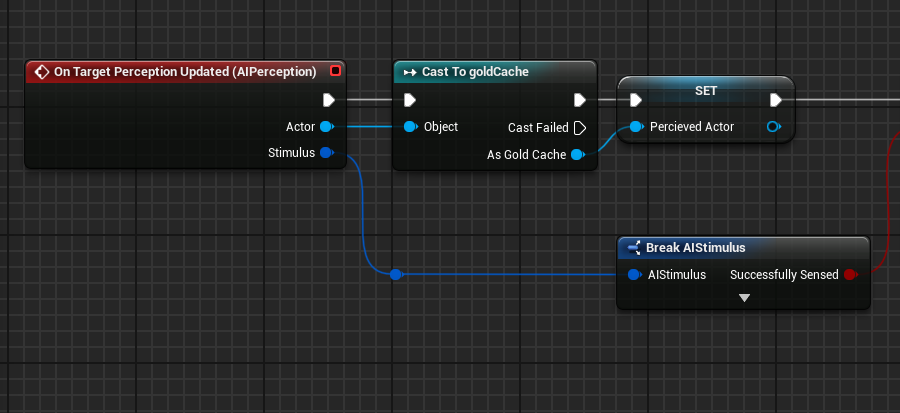
Set perceivedActor to cache

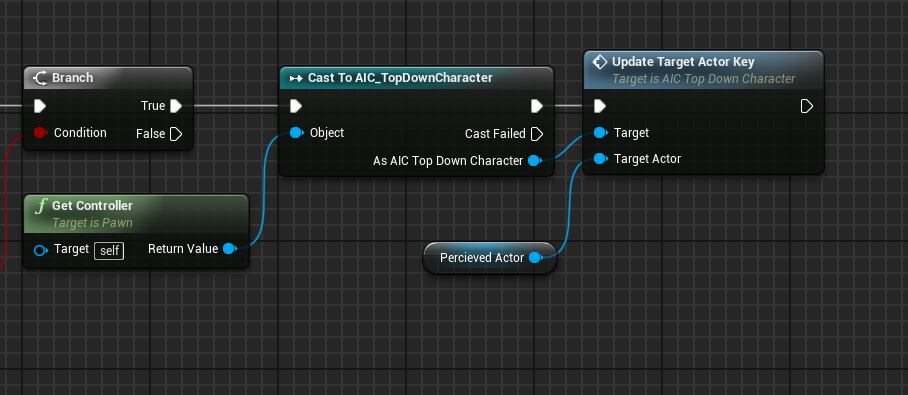
If(sense successful)

Controller = GetController()

Controller.UpdateTargetActorKey(perceivedActor)

In Unreal:





Relation to other components:

AI CONTROLLER: Gives the AIC data based on what it senses via its AIPerception component

**6. AI Behavior Tree Component**

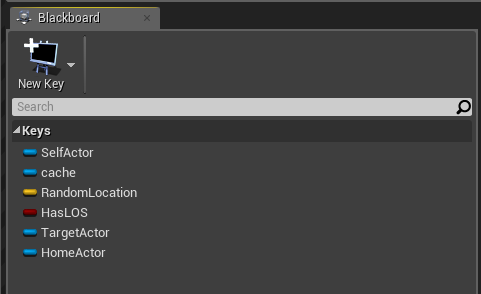
Name: AI\_BTree

Unreal Type: Behavior Tree

Description: The behavior tree defining all behavior to be computed and acted out by the AI player.

Data Structures:

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Default | Name | Description |
| Actor | Null | Cache |  |
| Vector | (0,0,0) | RandomLocation |  |
| Bool | False | HasLOS |  |
| Actor | Null | TargetActor |  |
| Actor | Null | HomeActor |  |



Algorithms:

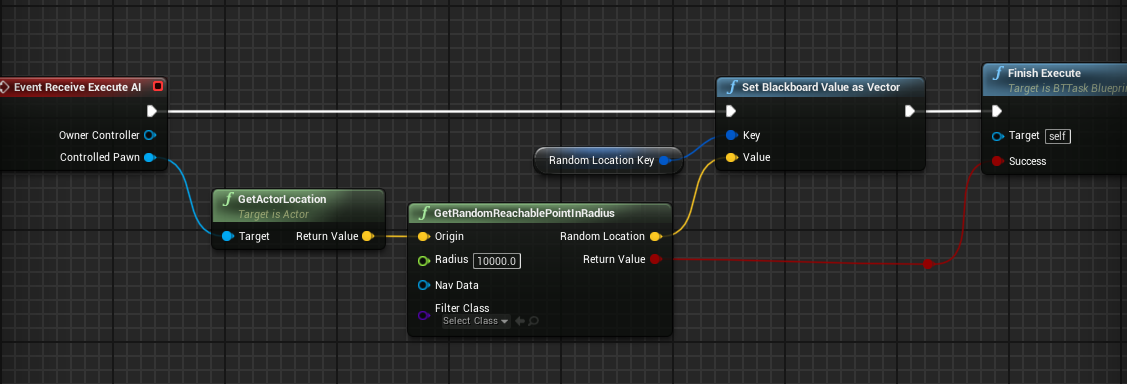
Void FindNavigableLocation(pawnType pawn)

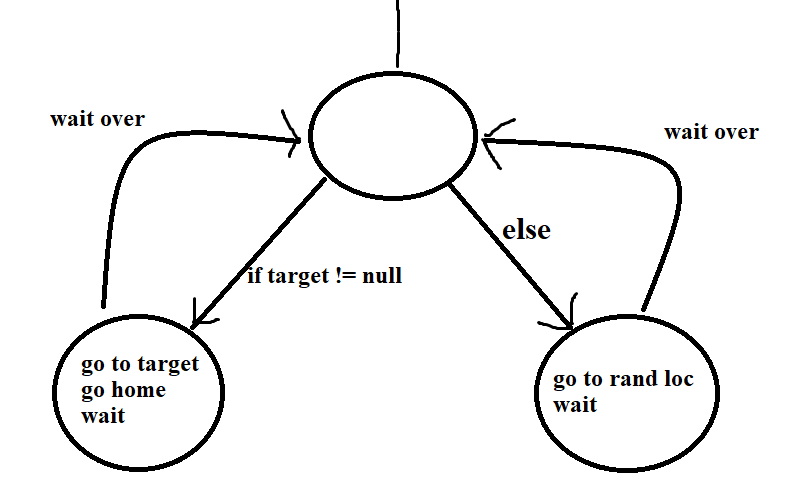
tempVec = GetRandomReachablePointInRadius(Pawn.GetActorLocation())

SetBlackboardValueAsVector(tempVec)

FinishExecute //needed for all blackboard tasks

In Unreal:



Relation to other components: 

In Unreal:

