## Technical Design Document

Oscar Smith



1. **– Version History**

|  |  |
| --- | --- |
| **Version** | **Description** |
| AI update | 30-07-2020 Ai initial push |
| AI update2 | 31-07-2020 Ai updated and edited push |
| AI commit | 12-08-2020 reinstating the AI document after it broke |
| AI commit 02 | 13-08-2020 more work going towards AI and pushed |
| AI commit 03 | 14-08-2020 another push updating work |
| AI commit 04 | 20-08-2020 another push updating work |
| AI final | 16-09-2020 final push to github with completed AI task |

1. **– Development Environment**

**2.1** - Visual studio 2019

**2.2** - Visual studio built in

**2.3** - Uploading to github at the end of each day,

**2.4** - <https://kenney.nl/assets> using a creative commons license

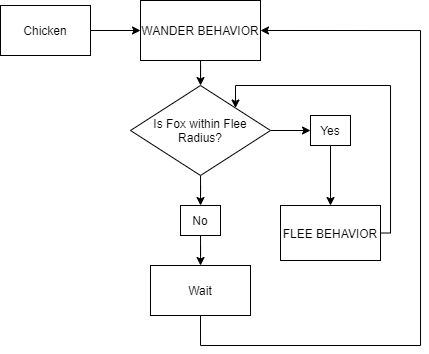
**2.5** – Tiled

**3.0** – I am to create a small AI program that features several chickens and a fox. The fix will periodically chase the closest chicken around when the chicken must flee from the fox. If a fox is not chasing a chicken it is to wait in its area and wander for a certain amount of random time. The chickens must wander around they’re coop stopping to eat food. Once a fox gets within a certain radius of a chicken the chicken must switch to its flee behavior when it picks a location in the opposite direction of the fox to pathfind to and travel to. Once the chicken arrives at that location it must either switch back to its Wander Behavior and travel back to the coop or continue to flee from the fox. The use of TAB will also bring up a debug screen where the borders of each chicken can be found and the paths that are formed.

**4.0 -** Entities

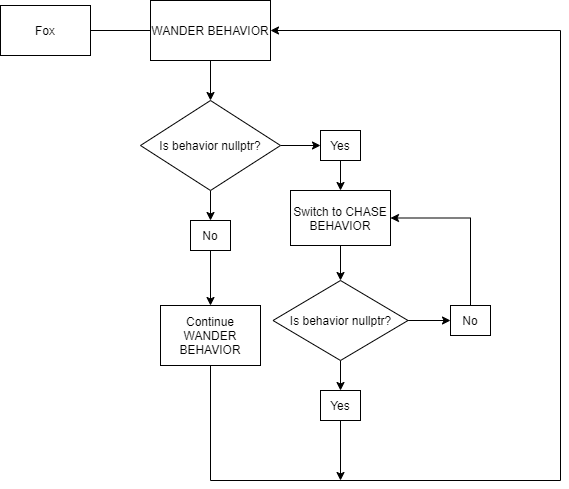
Chickens –

* Wander – around the chicken coop
* Flee – away from the fox



Fox –

* Wander – around the fox’s area
* Chase – The chickens for food

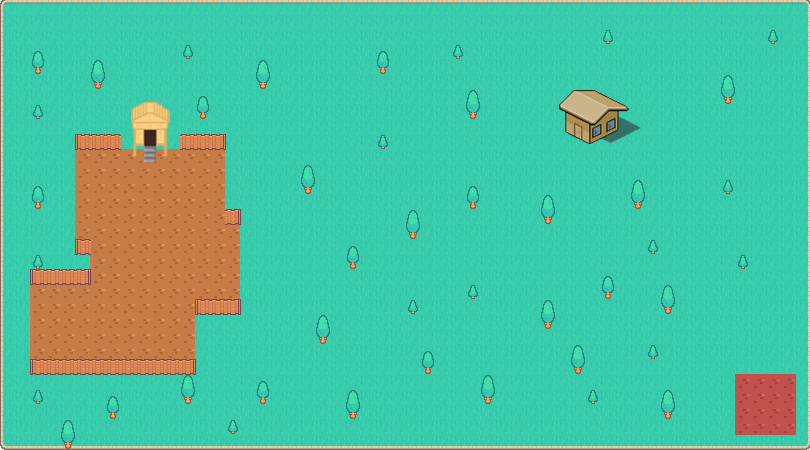


**5.0** – Item

Chicken Coop

****

Fox Area



**6.0** – Levels

1 level where everything is set and where all the characters move and interact.



**7.0** – Asset List

|  |  |
| --- | --- |
| **Asset** | **Description** |
| Chicken | Chicken has several images that are used for the walking and eating animations as well as any flipped sprites |
| Fox | Like the chicken the fix uses different sprites for different animations as well as flipped sprites |
| Background | The background is one solid image that uses a graph over the top to interact with the characters pathfinding and AI code. |

**Evaluation**

I am quite happy with how the project turned out as a whole following my general initial plan for the project which does feature several chickens and a fox that behave as intended. Though beginning AI in c++ was a challenge I feel I overcame my issues in a way that was appropriate to my level of skill I had at the time of completing the project. Though looking back I am aware of better ways to complete certain sections that could have been done more effectively, I am still happy with the final product.

I am also happy with the way I dealt with both the Wander and Flee behaviors. Learning about pathfinding and the maths behinding each behavior and learning about dealing with behaviors through classes and switch statements was quite interesting to me and though it was challenging I did enjoy it.

There are some parts of my AI code though that is heavy on the system resources. Though c++ as a minimum amount of code is easy for a modern pc to run I do have some maths that takes a second for the pc to run. I tried to optimize the code as much as possible and I made the chicken pathfinding practically instant and there is no lag. But for the foxes “picking closest chicken” and pathfinding to it is a bit more intense. The fox must path-find across the board to a chicken which is why it takes so long. Though I did remove statements that took more time, it's just the continuous finding the next node wrapped in a while statement that leaves the Fox thinking of just a second. Spending more time looking into the maths and implementation of both DFS and BFS and which would have been more optimal for each behavior would have also benefited. Though I did have some trouble with all the maths and how to implement them all I think it was wiser to resolve to just one working solution as opposed to spending more time on working towards a more optimal solution, taking a more ‘if it ain’t broke’ approach.

In my initial ideas and in my image files I also did include the use of a Farmer which would chase the Fox with the goal of taking it out to stop it from hunting his chickens. I also planned to include a process that allowed the chickens to repopulate over a night cycle and be taken out when the foxes hunt. Almost like a more ecosystem kind of approach. As the day cycle rolled over to night the chickens repopulated and created more chickens. When Fox went to Chase the chickens they would actively eat chickens and once the fox got enough chickens and was full it would resume its Wander behavior. Again with my lack of understanding or rather with the difficulty I had with a project that I did not have enough time to complete and implement these Ideas.

I did have some issues with the fox and Chickens running to the corners and getting stuck there. I also had an issue where each entity would use the furthest node closest to them to start a followpath which was a bug and overall affected how the behaviors worked. To fix both these issues I edited the Wander behavior to pick a closer node by getting it to pick the back of a closest node list which solved the problem mostly. Though using a function to get the singular closest node to the entity, it would've been more resource intensive so for optimisation sake I left it to picking the last of a list. One of the corners of the map still has chasing issues where both the fox and chickens get stuck while fleeing and chasing though it was something that I was willing to deal with, the chickens and the fox all steer themselves away from every other corner.

Overall I am happy with the project overall though on a second run there are definitely some things I would have done more effectively and farsighted to do more effectively.