

Computer Games Development CW208

Technical Design Document

Year III

|  |  |
| --- | --- |
| [Paul Nolan]  [C00213034] |  |
| [04/09/20] | |

Contents

[Game Architecture](#_heading=h.gjdgxs) **1**

[UML](#_heading=h.30j0zll) **2**

[Class Diagram: RPG Rules](#_heading=h.pvz2hshsqf4l) 3

[**Features**](#_heading=h.lzpgof58d6lf) **4**

[Feature: Car](#_heading=h.xsql0zx4eq3c) 4

[Feature: Map](#_heading=h.uj49ug2s0czf) 4

[Feature: Town](#_heading=h.cuuc4qccfsvn) 4

[Feature: Hud](#_heading=h.r0y375yvs2fr) 4

[CRC Cards](#_heading=h.3znysh7) **4**

[Class Name : Car](#_heading=h.q719m2it8ih5) 4

[Class Name : Map](#_heading=h.8y259s664sl4) 5

[Class Name : Load Map](#_heading=h.l9rxpaflq8kb) 5

[Class Name : Towns](#_heading=h.3azbqhpimvh8) 5

[Class Name : Roads](#_heading=h.g0gzvetzqrk9) 6

[Class Name : Tiles](#_heading=h.bd1kelu4a1y) 6

[**References**](#_heading=h.5wn98tr8nxyh) **6**

# **Game Architecture**

Game Specific Subsystems and Game Engine Architecture

# **UML**

## ***Class Diagram: RPG Rules***

## 

# 

# **Features**

## ***Feature:*** Car

Tasks:

1. Create a Car class.
2. Have the Car move according to the paths provided by the algorithm located in the map class.

## Feature: Map

Tasks:

1. Create a Map class that handles the data acquired from the yaml file.
2. The data is converted into a series of interconnected series of nodes and arcs which are represented by a town and road class respectively.
3. If a node were to be selected the algorithms are run with the map changed to display the path from the fuel algorithm,
4. If a road were to be selected it will disappear. If this were to occur the fuel algorithm is called and the path will be altered if necessary.

## Feature: Town

Tasks:

1. Create a Town class that represents the nodes of the created map.

## Feature: Hud

Tasks:

1. Create a series of text and shapes to represent the Hud.
2. Use the text to display data created by the algorithms.

## Feature: Dynamic Elements of the algorithm

Tasks:

1. If a road is selected while a path is being run the algorithm is recalled and the roads id is passed to it.
2. It is determined whether the change in the road will affect the path. If it doesnt the algorithm ends early.
3. If it is, the algorithm is rerun starting with the path node closest to the changed road.

# **CRC Cards**

|  |  |
| --- | --- |
| ***Class Name :*** Car | |
| Subclasses : None | |
| Superclasses : None | |
| Responsibilities | Collaborators |
| Run path generated by algorithm. | Map |
| Represent user in the program. |  |
| Visually show the path being travelled. |  |

# 

|  |  |
| --- | --- |
| ***Class Name :*** Map | |
| Subclasses : None | |
| Superclasses : None | |
| Responsibilities | Collaborators |
| Generate path using the algorithm | Load Map |
| Create a map using loaded data using the Town and Road classes. | Car |
| Visually represent data from the algorithm such as time and the towns used. |  |
| Run the base A\* algorithm and output its data such as time taken and path created |  |

# 

|  |  |
| --- | --- |
| Class Name : Load Map | |
| Subclasses : None | |
| Superclasses : None | |
| Responsibilities | Collaborators |
| To load in all data needed to create the map. | Map |

# 

|  |  |
| --- | --- |
| Class Name : Towns | |
| Subclasses : None | |
| Superclasses : None | |
| Responsibilities | Collaborators |
| To represent the nodes used in the algorithms |  |
| Store the ids of connected roads. |  |
| Store the ids of previously visited towns. |  |
| Store the amount of fuel the user had when they passed through the node. |  |

# 

|  |  |
| --- | --- |
| Class Name : Roads | |
| Subclasses : None | |
| Superclasses : None | |
| Responsibilities | Collaborators |
| To represent the arcs used in the algorithms. |  |
| To store the ids of the related roads. |  |
| To store the cost of travelling between the related towns |  |
| To create and store the tiles used in the refined paths. |  |

# 

|  |  |
| --- | --- |
| Class Name : Tiles | |
| Subclasses : None | |
| Superclasses : None | |
| Responsibilities | Collaborators |
|  |  |
|  |  |
|  |  |
|  |  |

# 

# ReferencesAmador, G. P., & Gomes, A. J. P. (2018). XTrek: An Influence-Aware Technique for Dijkstra’s and A Pathfinders. *International Journal of Computer Games Technology*, *2018*. https://doi.org/10.1155/2018/5184605

Carlos Gonçalves Adaixo, M., & Abel João Padrão Gomes, D. (n.d.). *UNIVERSIDADE DA BEIRA INTERIOR Engenharia Influence Map-Based Pathfinding Algorithms in Video Games*.

Dijjkstra, E. W. (1959). A Note on Two Problems in Connexion with Graphs. In *NUMERISCHE MATHEMATIK* (Vol. 1, Issue 1). https://doi.org/10.1016/0042-6989(66)90039-3

Ezra Sidran, by D. (n.d.). *An Analysis of Dimdal’s 1 “An Optimal Pathfinder for Vehicles in Real-World Terrain Maps” Spring 2005*.

Graham Hugh McCabe Stephen Sheridan, R. (2003). Issue 2 Article 6 2003 Part of the Computer and Systems Architecture Commons. *The ITB Journal*, *4*(2). https://doi.org/10.21427/D7ZQ9J

P. E. Hart, N. J. N. and B. R. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In *IEEE Transactions on Systems Science and Cybernetics* (pp. 100–107). IEEE.

Sazaki, Y., Primanita, A., & Syahroyni, M. (2018). Pathfinding car racing game using dynamic pathfinding algorithm and algorithm A∗. *Proceedings - ICWT 2017: 3rd International Conference on Wireless and Telematics 2017*, *2017*-*July*, 164–169. https://doi.org/10.1109/ICWT.2017.8284160

Stuart Russel, P. N. (2010). Artificial intelligence: A Modern Approach. In *2010 The 2nd International Conference on Computer and Automation Engineering, ICCAE 2010* (Vol. 4). https://doi.org/10.1109/ICCAE.2010.5451578

Yan Li, Wenju Zhao, Zhenhua Zhou, C. C. (2013). Hierarchical and Dynamic Pathfinding Algorithms in Game Maps. *International Journal of Advancements in Computing Technology*, *5*(11), 87–98. https://doi.org/10.4156/ijact.vol5.issue11.10

# 