



MSc Software Engineering

School of Mathematical & Computer Sciences

Clients for a massively multi-player on-line historical role-playing game

Author:

Yuanqi Zhu

HW ID:

H00117534

Supervisor:

Dr Hans-Wolfgang LOIDL

Second Reader:

Prof. Sven-Bodo Scholz

Abstract:

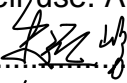
JominiEngine¹ is historical Massively Multiplayer Online Role-Playing Game which designed and created in 2015. After three years of developing with three different developers, JominiEngine has a relatively well-developed server and three text-based clients. This project aims to develop a Unity-based graphical client for a historically accurate middle ages massively multiplayer online role-playing game to help the learning and understand this phase of history. This client helps to more easily interact with players and help players learn about history. This Game engine is called JominiEngine. The concrete instance of the game engine is called “Age of Magna Carta” in the mainland of Britain in the time period of 1194-1225. As an Overlord, the player needs to compete with other overloads and conquer them to win the game. To reach this purpose, you will need to build up your strengths by developing your fiefs, reinforcing your armies, managing your family and even use some dirty moves. One of the goals of the project is to assess how the JominiEngine will help find out the historical game will help the people to learning history.

1 <http://www.macs.hw.ac.uk/~hwloidl/Projects/JominiEngine/>

Declaration

I, Yuanqi Zhu,

confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: 

Date: 14 — 08 — 19

Table of Contents

Abstract:	1
Declaration	2
Table of Contents	3
List of Figures	6
List of Abbreviations	10
1 Introduction:	11
1.1 Background:.....	11
1.2 AIM:.....	11
1.3 Objectives:.....	11
1.4 Evaluations:.....	12
2 Literature Review:	13
2.1 JominiEngine:	13
2.1.1 JominiEngine design:	13
2.1.2 Existing Client:	16
2.2 Game UI Design:	19
2.3 The programming languages and Tools:.....	21
2.3.1 .NET:.....	21
2.3.2 C#:.....	22
2.3.3 Unity:	23
2.4 Design Pattern.....	24
3 Professional, Legal, Ethical, and Social Issues	25
3.1 Professional.....	25
3.2 Legal.....	26
3.3 Ethical.....	26
3.4 Social	26
4.1 Model	27

4.2	View.....	29
4.3	Control.....	36
4.3.1	Controller.cs.....	36
5	Evaluation.....	41
5.1	Function Evaluation.....	41
	Case 1: login test.....	41
	Case 8: Check Current fief detail test	41
	Case 3: move test.....	42
	Case 4: check the user profile	42
	Case 5: check army list	43
	Case 6: hire 30 troops.....	43
	Case 7: maintain army 4	44
	Case 8: disbanded army 1.....	44
	Case 9: appoint a leader for the army	45
	Case 10: propose marriage	45
	Case11: accept and reject marriage	46
	Case 12: try for a child.....	46
	Case 13: kidnapping a character	46
	Case 14: ransom/ release/ execute the captive	47
	Case 15: siege current fief	47
	Case 16: siege around the storm/ negotiate/ negotiate.....	47
	Case 17: spy a fief	48
	Case 18: spy a character	48
	Case 19: spy an army.....	49
5.2	User Evaluation.....	49
5.2.1	Background	50

5.2.2	interface.....	53
5.2.3	Gameplay.....	56
6	Conclusion.....	60
6.1	Finish function in the client-side	61
6.2	future development.....	61
6.2.1	server.....	61
6.2.2	client.....	62
7	Appendix.....	63
8	Reference.....	66

List of Figures

Figure 1 Basic game model.....	13
Figure 2 System architecture	14
Figure 3 FSM of Terminal Client	16
Figure 4 Structure diagram of Gtk client.....	17
Figure 5 Gtk Client screen shoot.....	18
Figure 6 Structure diagram of Android client	18
Figure 7 relation between difficulty and ability	21
Figure 8 The CLR, CTS and CLS relationship	21
Figure 9 Unity IDE screen shoot	24
Figure 10 MVC-architecture	25
Figure 11 Template setting.....	27
Figure 12 Tuple class	28
Figure 13 login scene	29
Figure 14 Detail scene	30
Figure 15 army & army results scenes	31
Figure 16 profile scene.....	32
Figure 17 map scene.....	33
Figure 18 siege and siege result scenes	34
Figure 19 family scene	35
Figure 20 spy scene	35
Figure 21 kidnap scene	36
Figure 22 profile control.....	40
Figure 23 test case 1 result	41
Figure 24 test case 2 result	41
Figure 25 diagram for the travel path	42
Figure 26 test case 3 result	42
Figure 27 test case 4 result	43
Figure 28 test case 5 result	43
Figure 29 test case 6 result	44

Figure 30 test case 7 result	44
Figure 31 test case 8 result	45
Figure 32 test case 10 result	46
Figure 33 test case 11 result	46
Figure 34 test case 12 result	46
Figure 35 test case 15 result	47
Figure 36 test case 16 result	48
Figure 37 test case 17 result	48
Figure 38 test case 18 result	49
Figure 39 Test case 19 result	49

List of Table

Table 1 Client commands.....	17
Table 2library list	28
Table 3 function list of controller.cs	38
Table 4 introduction of used function.....	39
Table 5 comparison table of scenes and controller	40
Table 6	50
Table 7	50
Table 8	51
Table 9	51
Table 10	51
Table 11	52
Table 12	52
Table 13	53
Table 14	53
Table 15	53
Table 16	54
Table 17	54
Table 18	54
Table 19	55
Table 20	56
Table 21	56
Table 22	57
Table 23	57
Table 24	57

How long would you play this game each time last



(12 条回复)

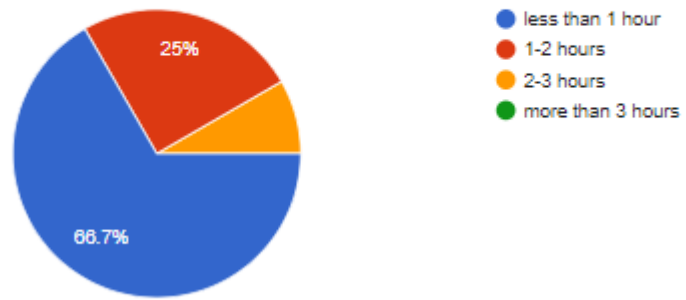


Table 25 58

List of Abbreviations

RPG: Role-Playing Game

MMORPG: Massively Multiplayer Online Role-Playing Game

DBMS: DataBase Management System

FLOSS: Free/Libre/Open-source software

CLR: Common Language Runtime

CTS: Common Type System

CLS: Common Language Specification

CLI: Common Language Infrastructure

IDE: Integrated Development Environment

FSM: Finite-State Machine

GUI: Graphical User Interface

UI: User Interface

CPU: Central Processing Unit

MVC: Model View Control

PC: Player Character

NPC: None-Player Character

1 Introduction:

1.1 Background:

Games create a new, virtual world for the players interacting with them. Players can become anyone in the world of the game, they can become the hero of the fantasy land, become athletic in the Olympics even become part of history. Unlike the early age of Role-player-game player travel around the fantasy world, nowadays there are lots of game is based on real history. This attracts attention in the learning community to convey messages about a historical time period, does the historical game will help people learn history. In this project, one game will use to help me find out the answer - JominiEngine.

The JominiEngine is a historical Massively Multiplayer Online Role-Playing Game (MMORPG), design and Implemented by David Bond, Hans-Wolfgang Loidl and Sandy Louchart in 2015. JominiEngine is made as an educational tool for history education, to provide an experience to interact with history. This project will be working on the existing server and client implementation. The server is based on the server named RepairHist_mmo which was created by Helen RANKIN in 2016 and the client is based on the Rory Malcolm created the text-based terminal client in 2017.

1.2 AIM:

The purpose of this project is to develop graphical, Unity-based client to the server and to research how an MMORPG can help in learning about history. To research this, a new client should be built, this client is built in unity¹. Unity is the most popular low-level 3D game engine; it is a great 3D game engine to have an inexperienced programmer. it has all the functions of gameplay from the server-side, and it must have the graph interface which matching with the characteristics of the history.

1.3 Objectives:

This part will list the objectives will be completed in this project.

- Build a graphical client for JominiEngine by unity

1 www.unity3d.com/

- The client should have all the function for gameplay form server and the client needs to establish communication to the server in a portable way.
- The graphical interface should match the historical background.
Find out if an in which way the game helps players to learn about history.

1.4 Evaluations:

The evaluation has 2 parts, function evaluation and user evaluation.

The function evaluation is aimed at testing functions on the client to prove they can communicate to the server correctly, send the right request to the server and parse reply from server and interact with the server to perform the intended operation. The user evaluation is to test the usability for the graphic user interface and to assess whether the game is an effective tool of learning about the historical context.

2 Literature Review:

2.1 JominiEngine:

The JominiEngine is an emerging, distributed, scalability game engine for historical massive multiplayer online role-playing games (MMORPGs). It can simulate different historical time periods. In this case, JominiEngine will model the Age of Magna Carta on the mainland of Britain, this model has high historical accuracy because it underlying the engine is a model of core interactions of nobles of that time and an accurate database of nobles that can be used as PCs or NPCs.

2.1.1 JominiEngine design:

The JominiEngine has 3 main components - character management, Resource management and the combat resolution. In the fief, it will generate wealth, produce money and people, so that the player can hire armies and NPC in the fief. Armies protect the player's fiefs and plunder new fiefs from his enemies by combat system. The combat system can let player attack, siege or pillage other players. If the player wants to survive in this game, the player must use the Household. Player marries and sire heirs with NPC. Player also can find talented NPCs to serve the players. (Bond, 2015)

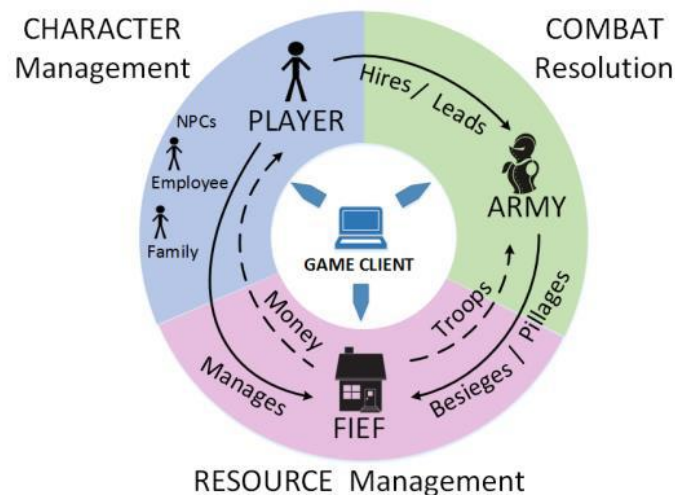


Figure 1 Basic game model¹

¹ From Design and Implementation of the Jomini Engine: Towards a historical Massively Multiplayer Online Role-Playing Game, page 4

Architecture:

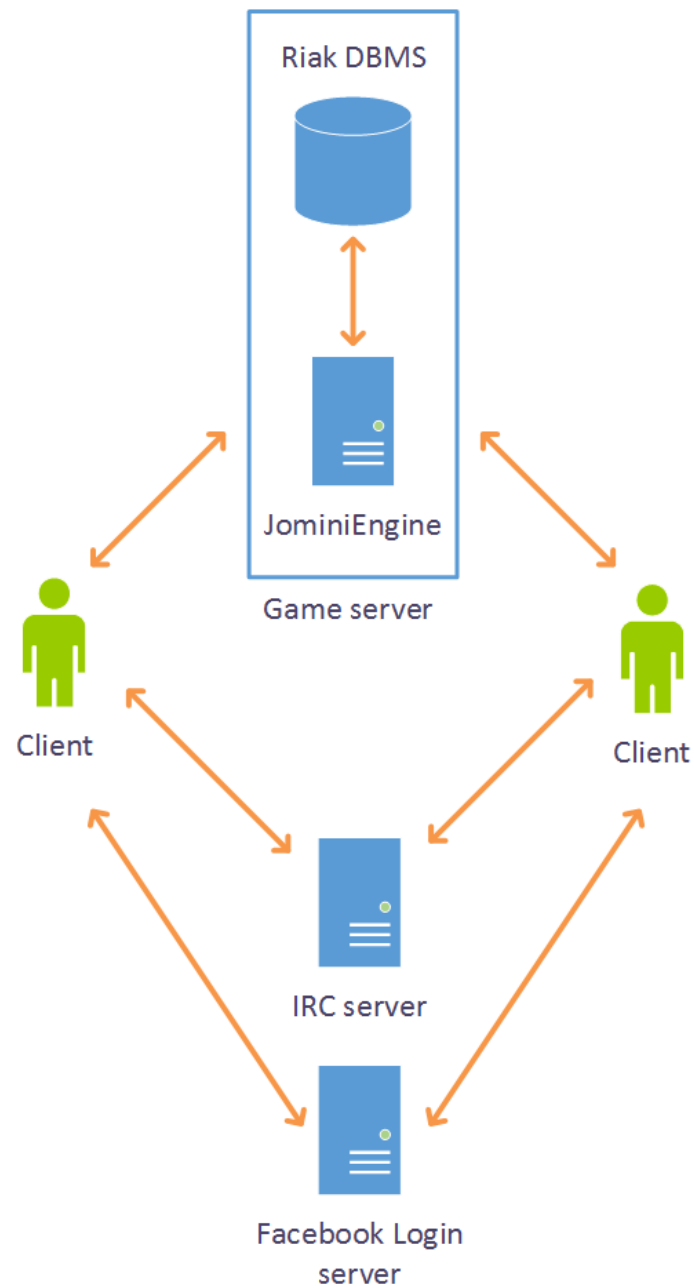


Figure 2 System architecture¹

The JominiEngine uses basic client-server architecture. Using this architecture has a fellow advantage:

- This architecture is very easy to implement.

¹ From Design and Implementation of the Jomini Engine: Towards a historical Massively Multiplayer Online Role-Playing Game, page 4

- Because the JominiEngine is stored in server this achieve a certain extent anti-cheating.
- A dedicated server has the ability to host the game.
- All the clients connect to one server the client is apt to centralised control

2.1.1.1 DBMS:

The current server is used Riak as the backend DBMS to store the game map and objects. Riak is a key-value, NoSQL (non-relational) DBMS. For an MMO game engine, its reduced reliability (NoSQL DBMS's are not ACID compliant) is more than compensated for by its inherent advantages.

The Riak has advantages when a fast input for the small operations than the MySQL. And the Riak does not need to change the program code when the database structural changes. Riak is a distributed database, it allows adding nodes and clusters and close to users. Riak also provides the redundancy when component failure. (Bond, 2015) In Bond's project, it verified performance (time) used by using the MySQL and Riak to insert, fetch, delete the data in a different number of threads. The Riak use less time than MySQL while executing the query.

2.1.1.2 Communication Protocol:

The Lidgren.Network C# library is based on the UDP. Lidgren.The network uses a single UDP socket to send a simple API for connecting a client to a server, reading and sending messages. This library supports mainstream platforms Linux, Mac and OSX. But this library cannot support the latest .NET 4.6. And it is not tested on the Android, iPhone, Xamarin and Unity 2018.1. This library has a limitation of working on different platforms.

2.1.1.3 Security:

The current client is able to authenticate the server in order to prevent attackers from masquerading as a trusted server. This is achieved through X509 certificate verification. X509 certificate verification is a broadly used and well-recognised technique for authentication. The server's certificate is stored on the client while the game client is installing, enabling the client to validate the server's certificate when the connection in the process. The X509 certificate verification is been evaluated in Helen's project and it will use in this project to keep the data safety. (Rankin, 2016)

2.1.2 Existing Client:

Rory Malcolm developed 3 game clients for JominiEngine in 2017. There is a text-based Terminal Client, Gtk Client and Android Client. All these clients are experimental. They aim to make demonstrators agile communicate with the server. The new should be easier to use and provide more of an immersive experience to the player.

2.1.2.1 Terminal Client:

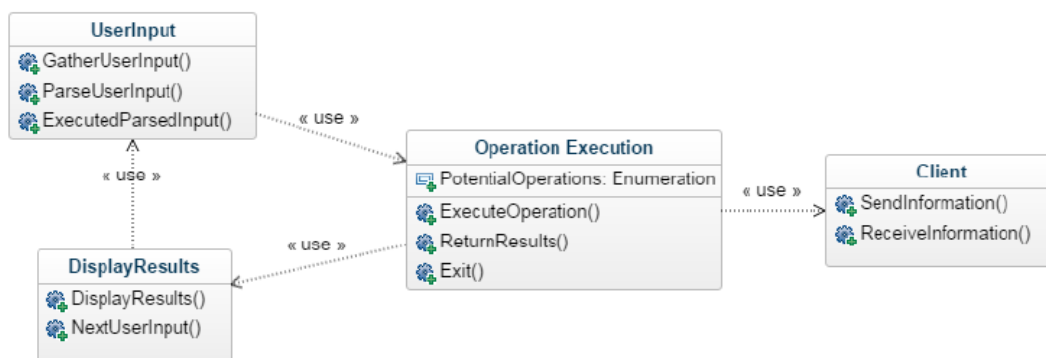


Figure 3 FSM of Terminal Client¹

The terminal client interaction with play and client by using a finite-state machine (FSM). FSM is a mathematical model of computation. It is an abstract machine which means this client only can be in exactly one of a finite number of states at any given time. When the terminal client is running, the client will run in the loop. The client waits for user input after the user sends input the command, the operation execution is used, it uses the client to send the information to the server and receive the feedback from the server. Then operation execution uses the DisplayResults to display the feedback. Finally, the client waits for the next command. This loop will keep running until the player exit the game. (Malcolm, 2017)

All the commands of the terminal Client:

¹ From an Exploration into Building Three Clients for JominiEngine, page 28

Commands	Descriptions
Help	Display all the commands available.
Siege	Siege current fief.
Army	Check the army information.
Check	Display the player's all the Fiefs' name.
Profile	Display all the information about the player. Include player ID, player Name, owned fiefs, current location, army name and player's purse.
Fief	Display current location's fief information. Include owner detail and armies in fief.
Move [direction]	Move to the select direction. (Possible choice nw, ne, e, w, n, s, se, sw)
Hire [amount]	Hire the recruits to the army from the current location.

Table 1 Client commands

2.1.2.2 Gtk Client:

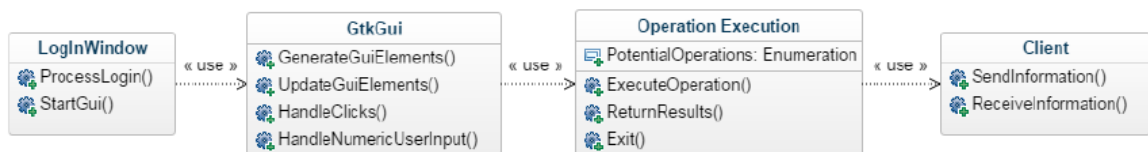


Figure 4 Structure diagram of Gtk client¹

This client uses the same FSM as the terminal client. The difference is Gtk client uses Gtk framework. Gtk framework will detect the player click the button and send a command to the server. The Gtk framework will receive feedback from the server and display it in the window. (Malcolm, 2017)

There are two windows in this client in GTK client. The first window is the login window; the player will use this window to log in the game. Next window is the main window. In this client, the fief profile and player profile will directly display on the

¹ From an Exploration into Building Three Clients for JominiEngine, page 38

window, unlike the terminal client player need to use the command to check the fief profile and player profile. Gtk client this design change has a very significant effect, in the later questionnaires, most of the testers think the Gtk client is more succinct and easier to use than the terminal client. This design will be kept in the Unity client in the future.

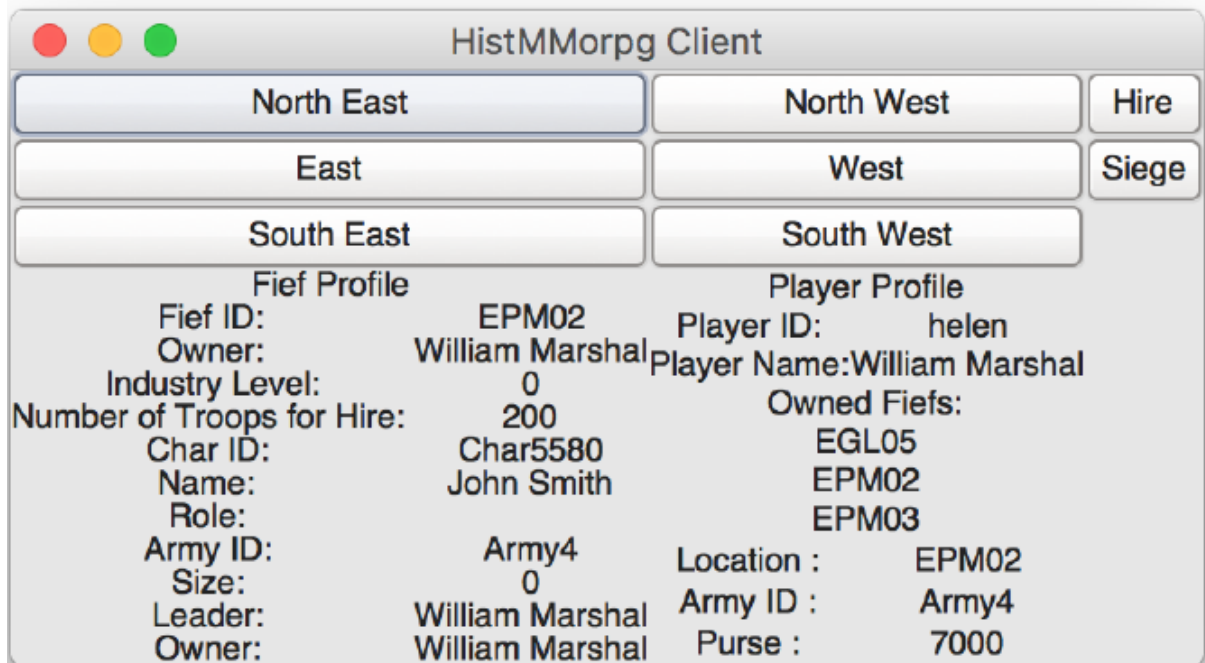


Figure 5 Gtk Client screen shoot

2.1.2.3 Android Client:



Figure 6 Structure diagram of Android client¹

The android client is the third client developed by Rory. This client has a similar layout and structure as the Gtk client. But this client is incomplete. The reason causes this client unfinished is one of important reason in this project will change data communication protocol. Because data communication protocol current used is

¹ From an Exploration into Building Three Clients for JominiEngine, page 45

Lidgren. The network cannot support the Java-based Android applications. (Malcolm, 2017)

2.2 Game UI Design:

The Unity client UI design will follow the principles below (RareSloth Games, 2018):

A fast hook

The client should immerse the player by a few clicking. This game should hook the player immediately in the game world. Ideally, the player should enter the game world as soon as possible after login the game.

Keep it light

The game UI should keep light and simple. Every element should be prudently added to UI. Each additional element adds to UI will become visual noise, hence players will have a more cognitive load. The background of the game should be an introduction in a few lines, if it is too long major of the player will NOT read it. The game should not have a manual for players to understand how to play it.

Bake the tutorial in the game

The player should learn how to play the game as natural as possible. This will improve the engagement of the game.

THEME your UI elements

The game UI's theme should accord with JominiEngine 's historical background. If the game doesn't meet the historical background will cause the player to confuse.

Create consistency

The client's all elements (such as style, navigation, button sizes, etc.) should keep consistency. The client will create a design pattern and colour palette and keep using them in this project.

Plan ahead

The Unity client in this project could not finish all David designed functions for the JominiEngine, even this project can finish all the functions, the JominiEngine may be

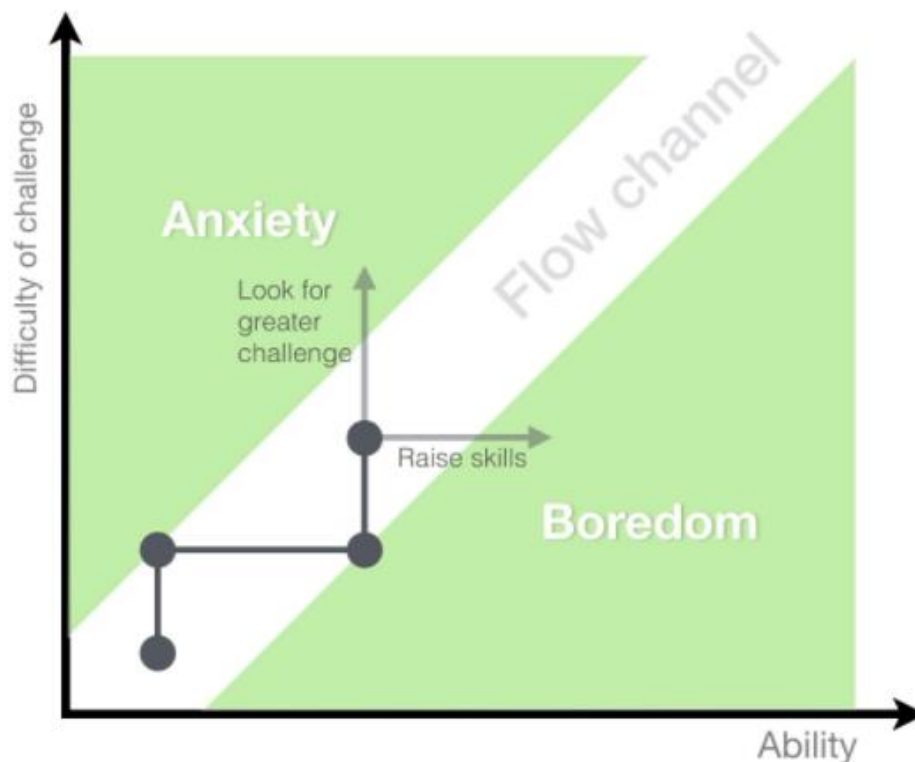
going to be adding other feature in the future. The game UI must schedule empty spot for future developing.

Focus on content

The game UI should focus on the game content and keep the game maintain flow. In this project, it will focus on historical educational elements. On the existing game client, they are more focus on achieving the game functions. There are not any historical contents in the current server. Players do not any background on this game while playing. This deviates from the purpose of JominiEngine. The player should know they are in the Age of Magna Carta on the mainland of Britain in the new client by adding new elements to the UI.

Maintain flow

The difficulty of the current game is too easy. Thus Player will feel bored easily while playing. To attract the player to keep playing the game the JominiEngine should find the balance of difficulty of challenge and ability.



2.3 The programming languages and Tools:

2.3.1 .NET:

The .NET framework is the Common Language Infrastructure (CLI). It is Microsoft's Internet strategy. It has major benefits like provides interoperability with existing code, supports many programming languages, uses a common runtime engine by all.NET-ware language, language integration, a comprehensive base class library and a simplifies deployment model. There is three building block to make these benefits possible – CLR (common language runtime), CTS (common type system) and CLS (common language specification).

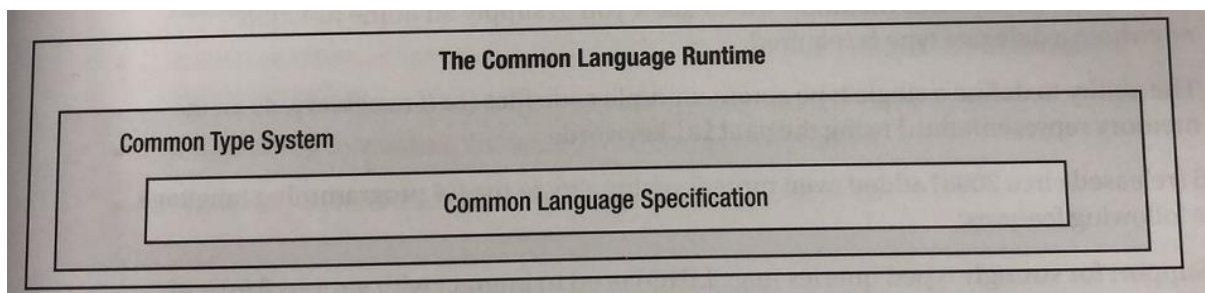


Figure 8 The CLR, CTS and CLS relationship²

Locating, loading and managing .NET objects on programmer behalf are the main characters of the. “CLR will take care of the low-level detail such as memory management, application hosting, coordinating threads and performing a basic security check (among other low-level details). CTS specification fully describes all possible data type and all programming constructs support by running time specifies how these entities can interact with each other and details how they are represented in the .NET metadata format. CLS is a related specification that defined a subset of common types and program constructs that all .NET programming language agrees on.” (Troelsen and Japikse, n.d.) Because of these features, the .NET support more 40 different languages and they share the same libraries.

¹ From <http://ui-patterns.com/patterns/Appropriate-challenge/examples/17571>

² From C# 6.0 and the .NET 4.6 Framework, Seventh Edition. Page 5

2.3.2 C#:

The current JominiEngine is developed programming language C#. This project will use C# in the Implementation part. C# was developed by Microsoft and released by Microsoft in July 2000. C# is part of its .NET Framework initiative. The C# is a programming language that its syntax is very similar to java. The C# standard provides support for getting at the minimum CLI features required. The Ecma Technical Committee produce a standard for C# in September 2000(C# Language Specification, 2017):

- C# is designed as a simple, modern, general-purpose, object-oriented programming language.
- C# is intended for use in developing software components suitable for deployment in distributed environments.
- C# provides strong type checking, array bounds checking, detection of attempts to use uninitialized variables, and automatic garbage collection to support the software engineering principles.
- The C# source code is portability, particularly for the programmers who are familiar with C and C++. C# is suitable for programming for embedded and hosted systems.
- C# applications are designed to not use too much regard to memory and processing power requirements.
- C# supports for internationalization

2.3.2.1 Mono:

Mono is an open-source implementation of Microsoft's .NET Framework for C# and the Common Language Runtime. And mono is a cross-platform application. Linux system cannot execute the .exe file, so the Linux system needs mono to execute the .exe file. Also, the mono provides a command called xbuild for Linux to compile C# project to the .exe file. The JominiEngine is developing by the C# and should able to running on the Linux system, so mono will use to test JomniEngine on the Linux system. (Mono-project.com, 2018)

2.3.2.2 Visual Studio 2017:

Microsoft Visual Studio is an integrated development environment (IDE). It is developed by Microsoft. Visual Studio support all the windows platform. Visual Studio included a large number of libraries. Also, Visual Studio can connect to the GitHub to do the version control for this project. Using visual studio can reliably improve programming speed. So that JomniEngine will use the Visual Studio to finish the programming part of the window then test program on the Linux system.

(Webster, 2018)

2.3.3Unity:

The new client will use Unity as the graphics engine. Unity is a Customizable IDE environment. The Unity is script developed base on the mono. So the Unity is fully supporting the C#. Using Unity can keep the language consistency between client and server.

One of an important feature of the Unity is what you see is what you get. There is an IDE for Unity. The Unity is not only can see results immediately after finish programming but also can check game effect while programming in progress.

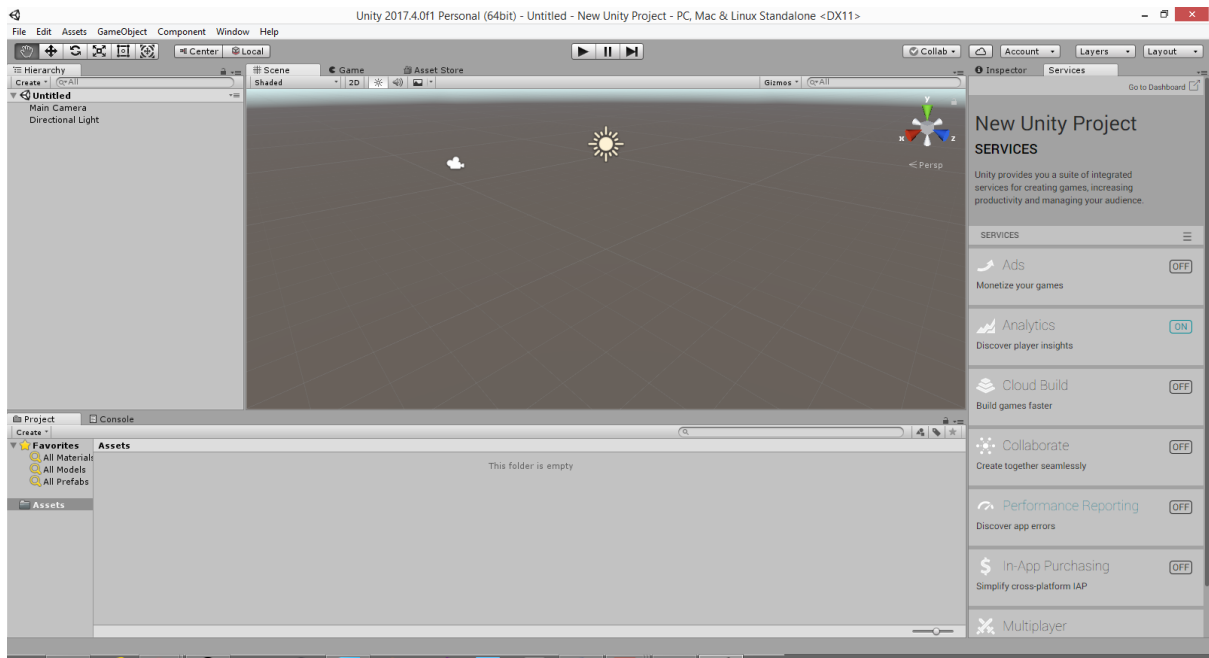


Figure 9 Unity IDE screen shoot

The Unity is supported multi-platforms. Unity client is easy to transplant to another platform such as Android. Unity is a Code driven design pattern. Unity enables us to quickly build a prototype. Developers can assign their objects capabilities of the assigned to different script component by capabilities of the objects, then hook up according to the demand of the object. (Alxiangfeng, 2018)

2.4 Design Pattern

The first come up with the design pattern is by an architect and also the founder of the Center for Environmental Structures Christopher Alexander in 1977. The design pattern is first used in homes and offices design. Ten years later, Panel on design methodology believes the design is also useful for computer programming. The design pattern is a standard solution to a common programming problem. It helps the programmer faster and more effective to design the program and make the code more flexible.

2.4.1MVC

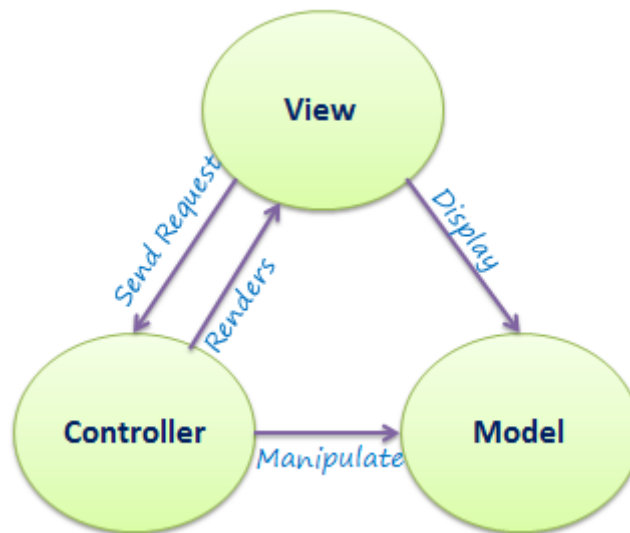


Figure 10 MVC-architecture¹

The MVC is a classic design pattern which will be used in this project. The MVC design pattern is made up three parts – Model, View and Controller. The model contains data, structure and the algorithm for the program. The normally the view is the graphical user interface to display the data form the model. And the view also collects the user requests and send the request to the controller. The controller handles the requests and manipulates data in the model by these requests. The MVC will help the project increase the extensibility, maintainability and testability.

3 Professional, Legal, Ethical, and Social Issues

3.1 Professional

All the code will under the Professional mention BCS Code of Conduct and software coding standards in this project. This project builds on existing research across multiple disciplines and sources and credits these in the References section. The original creation of the JominiEngine is the sole work of Dave Bond in 2015 and developed by Helen RANKIN and Rory Malcolm in 2016 and 2017. I am doing some development on the JominiEngine.

¹ <https://www.tutorialsteacher.com/Content/images/mvc/mvc-architecture.png>

3.2 Legal

Microsoft Office Word 2016 will use to write a document in this project and Microsoft Visual Studio community 2017 will use to program the C# programming. They are both use my personal license. And I will Strictly abide by the Microsoft software licensing terms and I will not exceed Microsoft software licensing terms in this project. And I will follow its terms and will not exceed it. All the library using in this project are open source and I will not change the source file without the permission of the owner. The Unity will use the free Personal license and I will follow the Unity and will not exceed it. In the Unity terms of service that I can publish the code produced through Unity as own product without paying license fees. The copyright on this project's code is using an open-source licence.

3.3 Ethical

User evaluation will use the questionnaire to evaluate the research questions. The tester will do a demo only after they are agreeing and signing a liability waiver agreement. All tester to be recruited are over 16, able to give informed consent, and have no known impediment that might affect their ability to participate in the study. All questions in the questionnaire are only designed for user evaluation. These questions are not involving any Personal data identifiable with living people and Sensitive personal data. And all the experiments data will be preserved carefully and safely. The questionnaire will collect some personal information only for studying the usability of the interface of the unity client. The email address will also be collected in the user evaluation. The email address is only used for further study and use as one of the evidence for delete questionnaire. And email address will not use for commercial use such as sending the advertisement. All the questionnaire will be anonymized and testers can delete their questionnaire any time you want and there do not need to give any reasons.

3.4 Social

Because JominiEngine is an MMORPG, character's language will be filtered, to keep the player away from the bullying through the game. The JominiEngine is for historical educational. Therefore, this game will accurately restore history and will not include the personal orientation of the value of this history.

4 Implementation

4.1 Model

The model of the JominiEngine was defined by David A. Bond on the server-side. In the client, David A. Bond specialized data structure used for the data commination between server and client. And the whole David A. Bond project is based on the .NET framework 4.5 and the unity only supports the .NET framework 3.5. So this part aims to make the old model compatible with the Unity.

4.1.1 Communication Format and Protocol: ProtoMessage

The data structure is defined in the c# project under the namespace ProtoMessage. For example, type `ProtoMessage` for the general commination, type `ProtoSiegeOverview` is for the siege display and type `ProtoPlayerCharacter` is for the profile display...In order to transplant data structure into unity client, ProtoMessage needs to be compiled to a Dynamic Link Library (DLL). To create a DLL file, the first step is creating a new C# project use following template: Templates -> visual C# -> class Library and select .NET Framework 3.5.

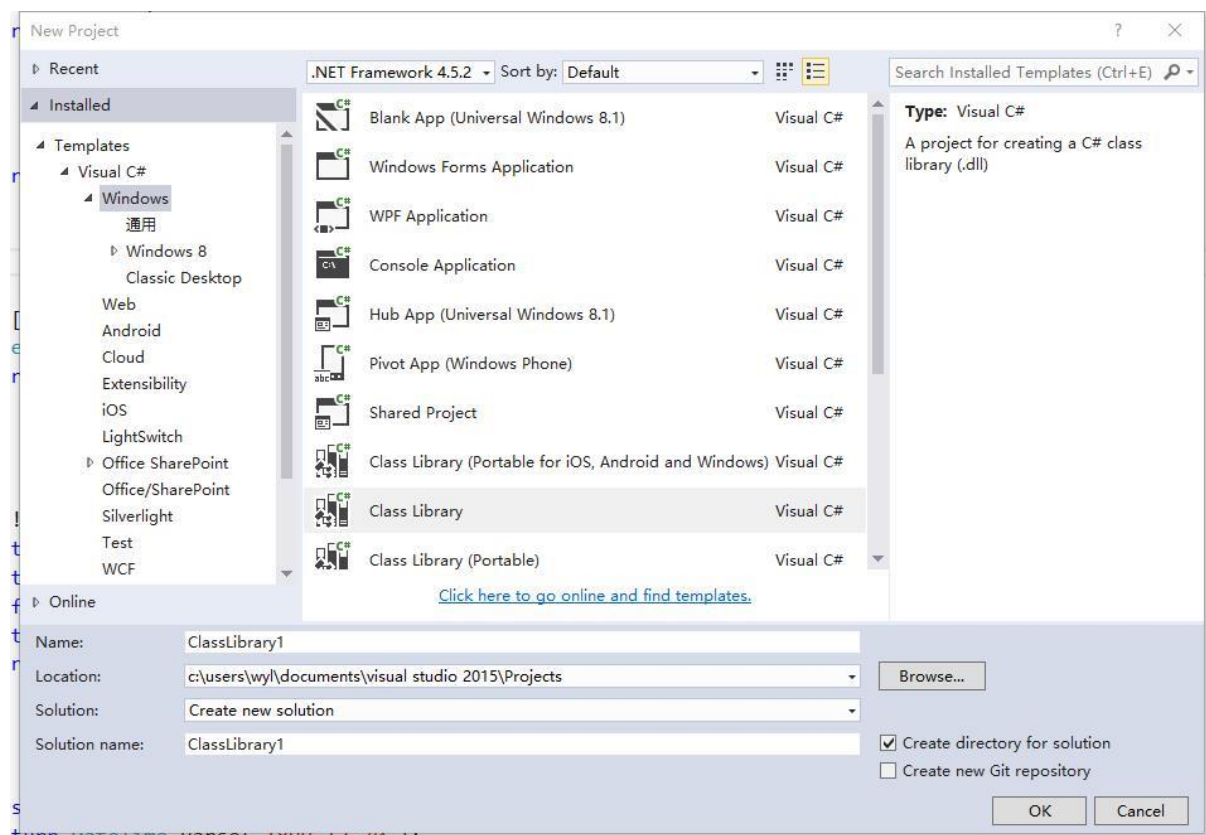


Figure 11 Template setting

Then copy the original ProtoMessage in and make fix errors. In the new project is missing there are 3 different types of problem in the new project.

The first problem is missing the library. Some library is useful in this project and these libraries and their purposes are lists in the table below.

Library name	Purpose
protobuf-net.dll	This library is for class protoMessage to help define the buffer data structure.
System.Threading.dll	The System.Threading.Tasks are not available in .Net 3.5. System.Threading.dll is using to import the library System.Threading.Tasks
Lidgren.Network.dll	This library is the protocol for the server and client data communication.

Table 2library list

The second problem is the type tuple is not defined in .Net 3.5. The solution is simply to create a new tuple class.

```

4      [using System.Linq,
5
6      namespace ProtoMessage
7      {
8          public class Tuple<T1, T2>
9          {
10             public T1 Item1 { get; private set; }
11             public T2 Item2 { get; private set; }
12             internal Tuple(T1 first, T2 second)
13             {
14                 Item1 = first;
15                 Item2 = second;
16             }
17         }
18     }
19
20     public static class Tuple
21     {
22         public static Tuple<T1, T2> New<T1, T2>(T1 first, T2 second)
23         {
24             var tuple = new Tuple<T1, T2>(first, second);
25             return tuple;
26         }
27     }
28 }
29

```

Figure 12Tuple class

The last part is replacing the different function between two frameworks, for example, the function to check string is Null or just white space in .Net 4.5 is

`IsNullOrWhiteSpace` and in .Net 3.5 `IsNullOrEmpty`

4.2 View

In this project will use a total of nine scenes interactive with players and display results for server reply.

4.2.1 Login scene

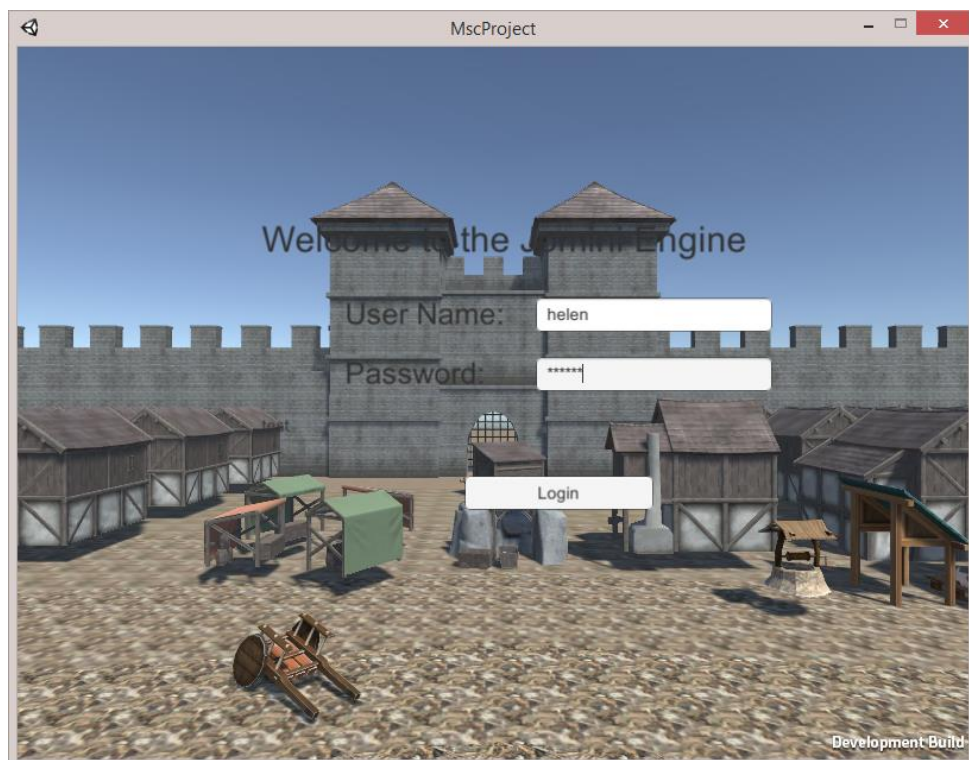


Figure 13 login scene

In this scene has two input field. They use to enter the user name and password. After the user fills the 2 input field then click the login button to log in.

4.2.2 Top-level actions scene

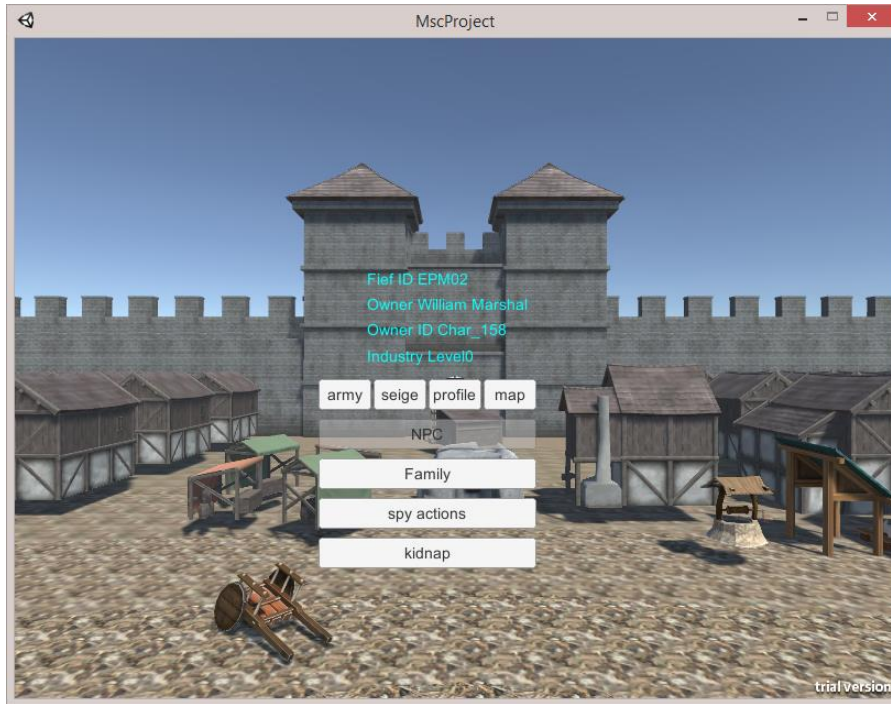


Figure 14 Detail scene

After login to the server, the client will jump into this scene. This scene is like an index link to other scenes. And this scene display fief detail the player currently in.

4.2.3 Army management scene

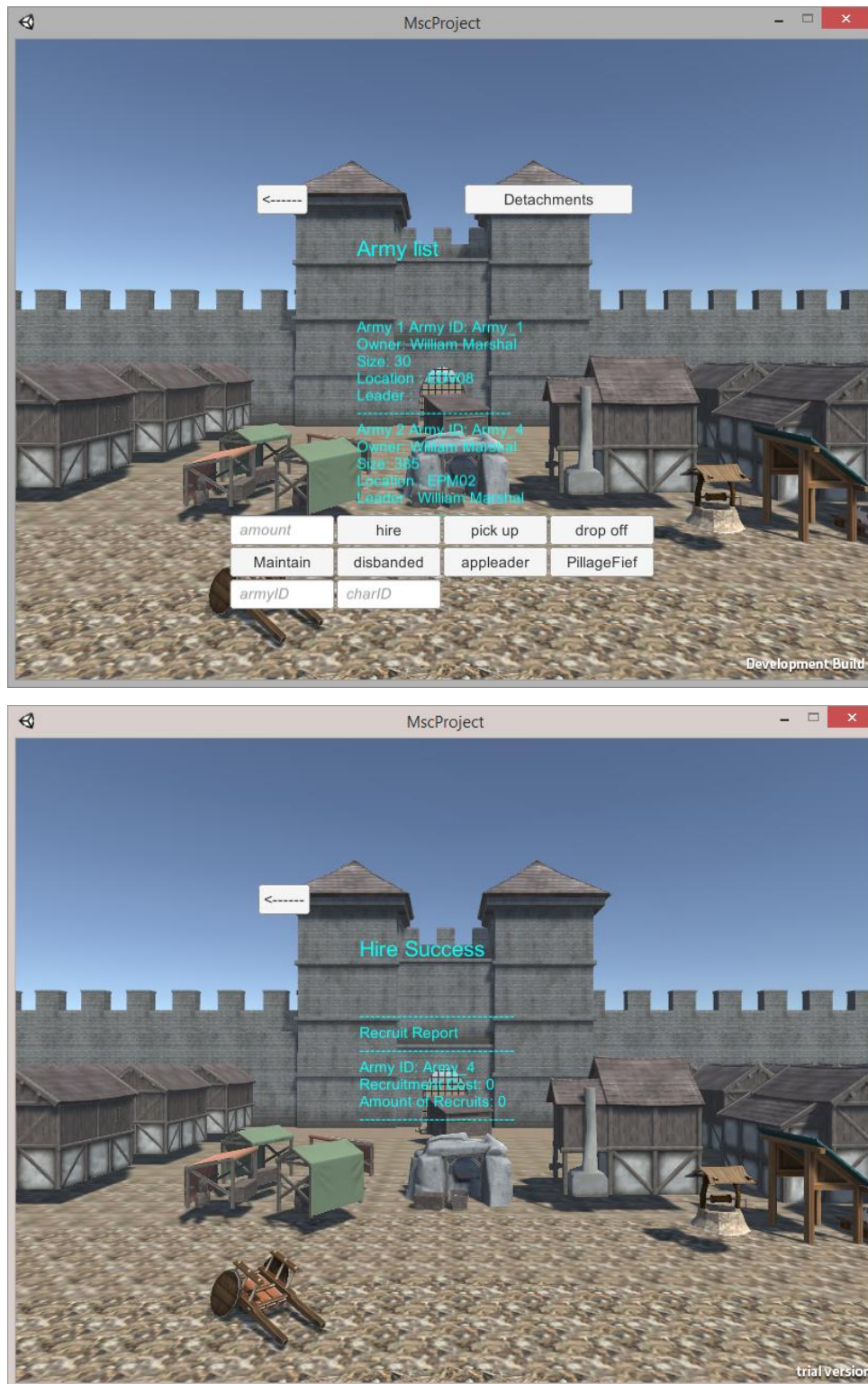


Figure 15 army & army results scenes

In the detail, scene click the army button to enter this army scene. In the army scene, it displays all the army information the player has. On the left top, there is a back button to back to the previous scene. The same button will use in most scenes. Pick

up, drop off, detachments and pillage fief button is not available for now. Before click the hire button need player need enter the number in the account input field. The maintain and disbanded button need to enter the army ID in the army ID input field. The appointed leader needs to enter the character ID and army ID. All the button in the army scene will lead to the army result scene. This scene will display the result of player action.

4.2.4 Profile scene

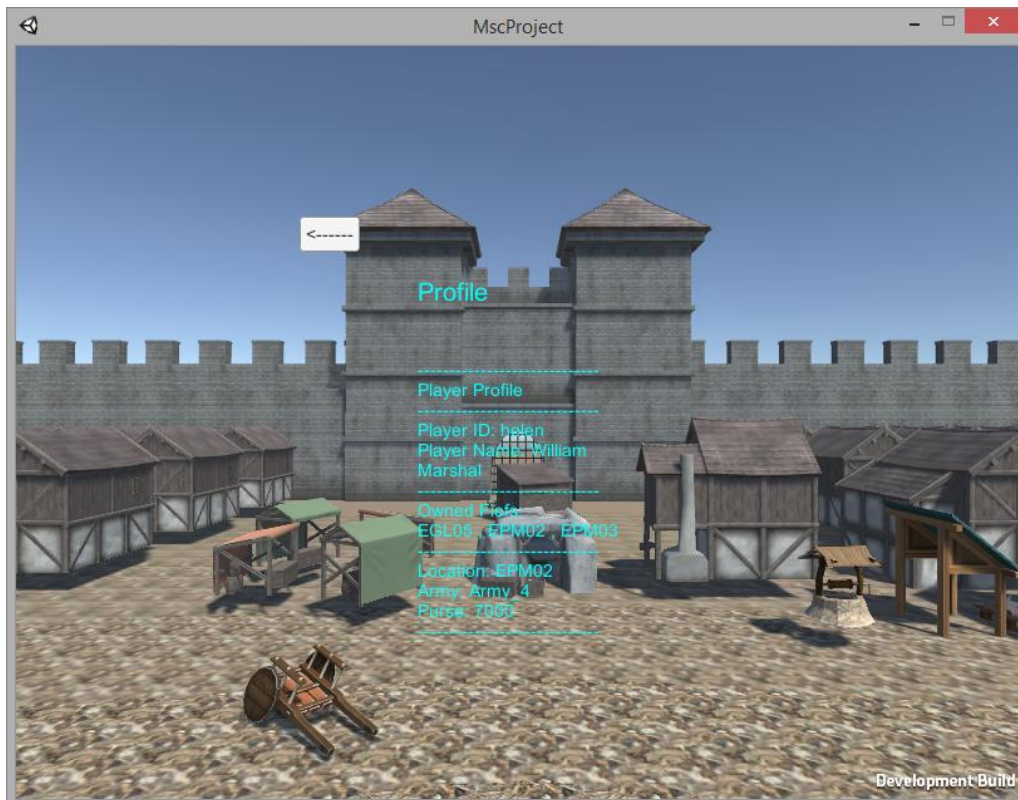


Figure 16 profile scene

This scene is used for display the player profile in the game

4.2.5 Map scene

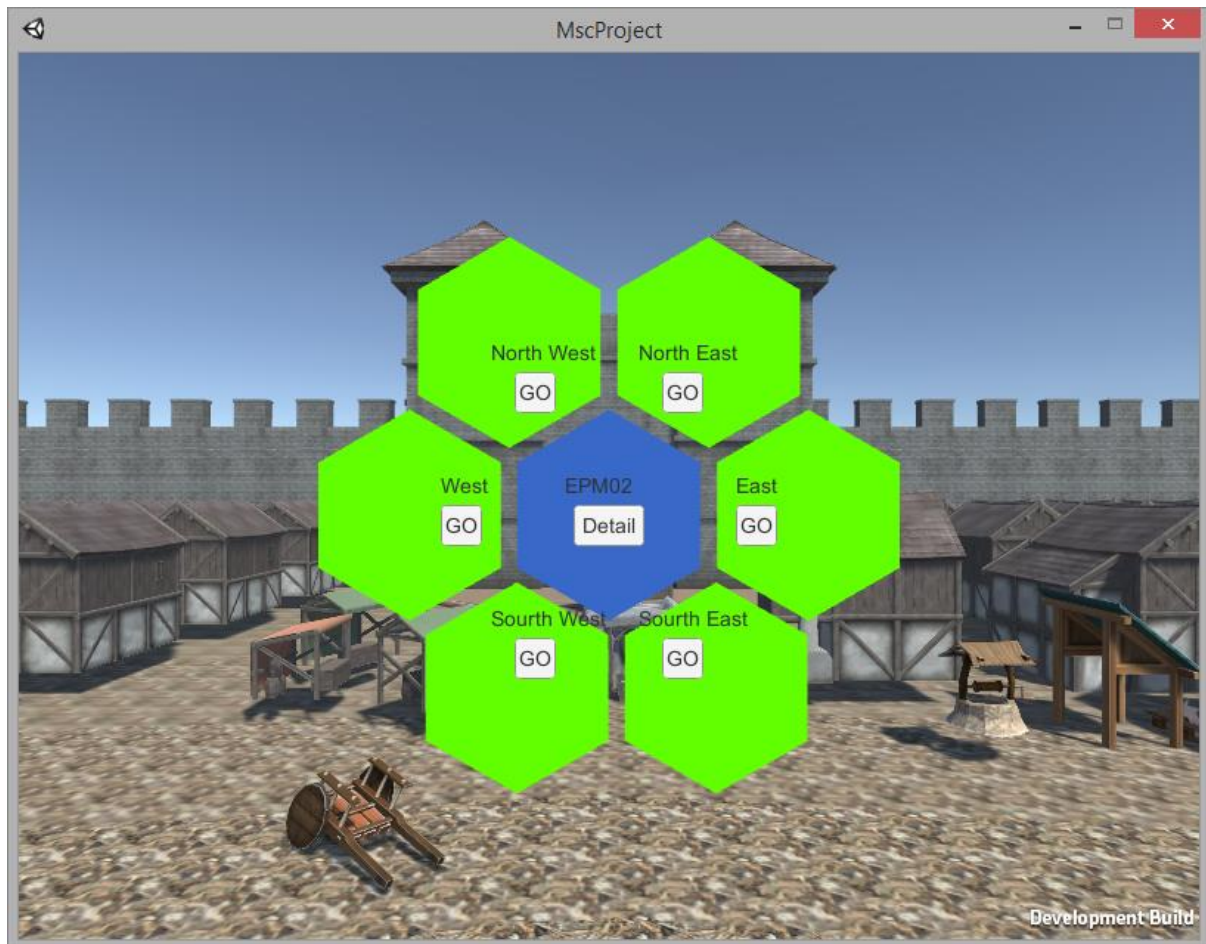


Figure 17 map scene

England is represented as a hex-map in the JominiEngine. And this scene has six go button and a detail button. The texts in the green hexagons are point direction of go below. The text in the blue hexagon is the fief ID the player is currently in. And the detail button will back to the detail scene. The text above detail button is the fief ID of the current fief.

4.2.6 Siege management scene

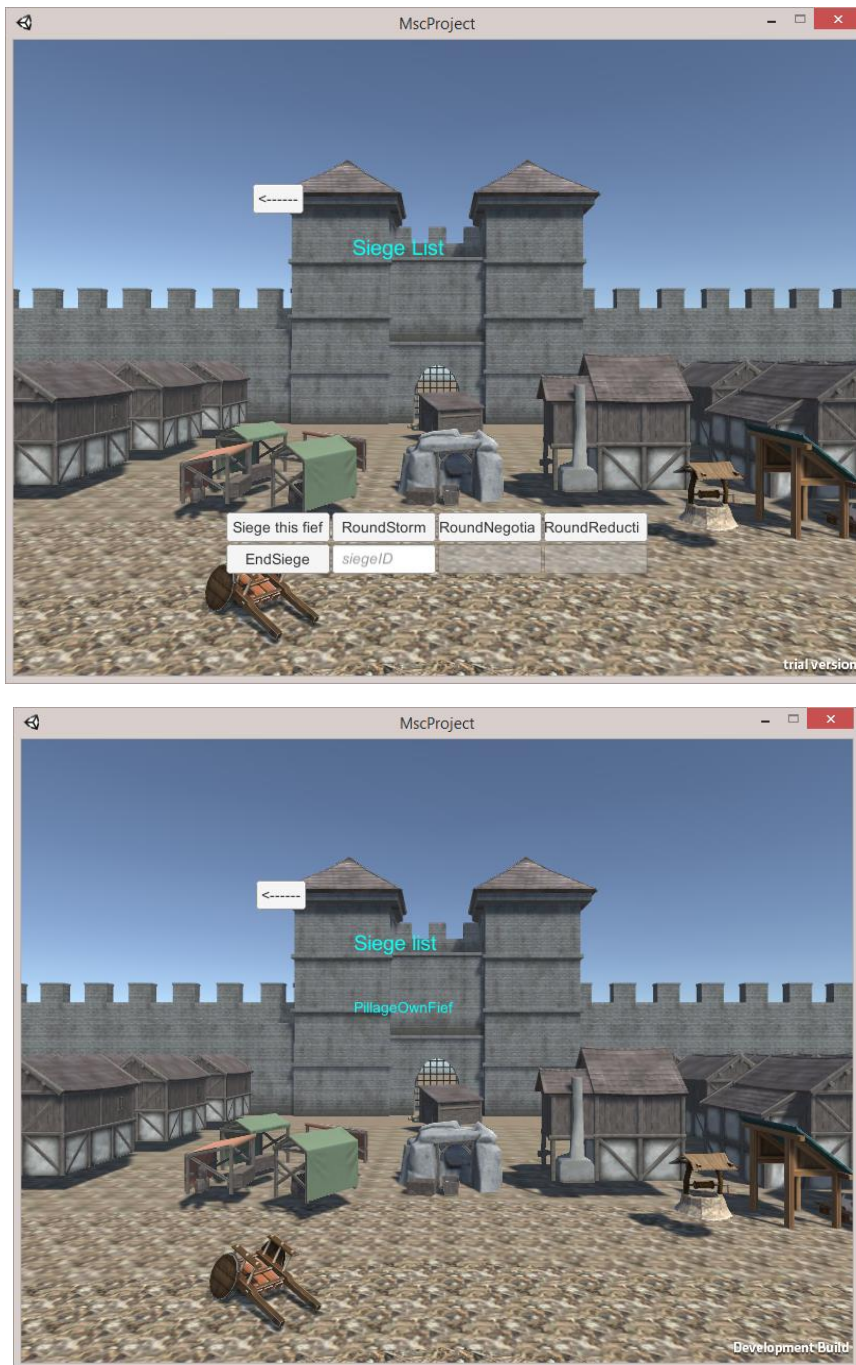


Figure 18 siege and siege result scenes

The army scene will display the all siege information The siege this fief button will siege the button the player currently in. The other four buttons will need the player to enter the siege ID in the input field before clicking them. All the button will lead to the siege result scene. The siege result scene will displayer the result of siege action.

4.2.7 Family Management scene

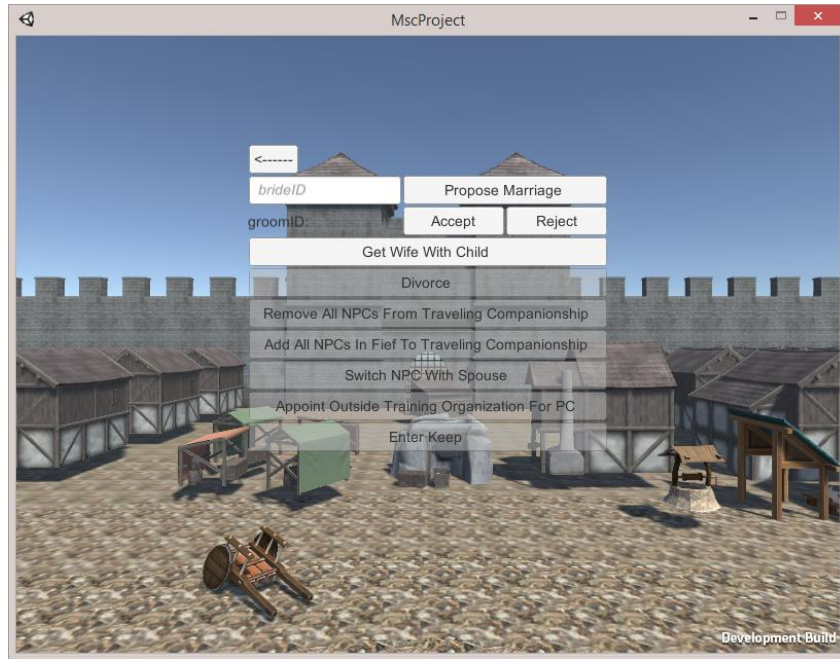


Figure 19 family scene

The Propose marriage needs to enter the bride ID first. The accept and reject button the groom ID. And get a wife with child button is used to try for a child with the player wife.

4.2.8 Spy scene

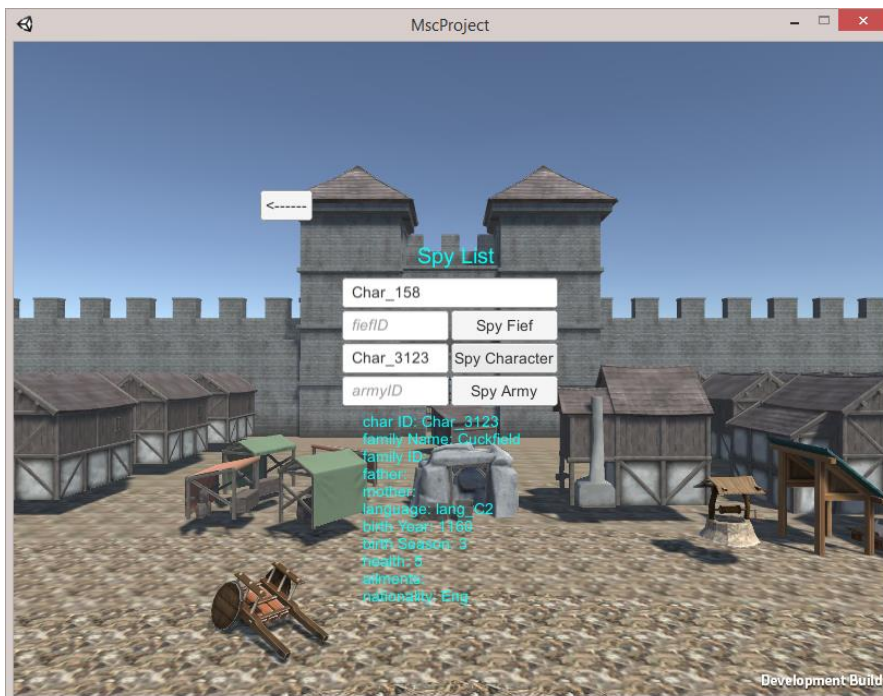


Figure 20 spy scene

This scene has 4 buttons and 4 input fields. It will display the result in this scene as well. Three spy buttons will need to enter the spy ID in the text field above them and then enter the other information in the text front them.

4.2.9 Kidnap scene



Figure 21 kidnap scene

This scene has 5 buttons and 3 input fields. It will display the captives detail. The kidnap button needs to enter the target and kidnapper's character IDs in the left 2 input fields in the same line before clicking. The ransom, release and execute button need enter captive character ID the in the input field in the same line before the click.

4.3 Control

4.3.1 Controller.cs

This class is the interface for the other controller. It is including the buffer and functions.

Function name	Arguments	Reply message type	Is used in the GUI
GetActionReply	Request action type	protoMessage	Yes
Login	Username and password	Void	Yes
GetArmyID	Character ID	ProtoPlayerCharacter	Yes
GetPlayers	None	ProtoPlayerCharacter	Yes

SiegeList	None	ProtoGenericArray <ProtoSiegeOverview>	Yes
Profile	None	ProtoPlayerCharacter	Yes
GetNPCList	None	ProtoPlayerCharacter	Yes
FiefDetails	None	FiefDetails	Yes
ViewMyFiefs	None	ProtoGenericArray <ProtoFief>	Yes
ArmyStatus	None	ProtoGenericArray <ProtoArmyOverview>	Yes
MaintainArmy	armyID	ProtoMessage	Yes
AppointLeader	armyID	ProtoMessage	Yes
ListDetachments	None	ProtoGenericArray <ProtoDetachment>	No
DropOffTroops	troops	ProtoMessage	NO
PickUpTroops	arm	ProtoMessage	NO
HireTroops	Number of Troops	ProtoMessage	Yes
PillageFief	None	ProtoMessage	No
DisbandArmy	None	ProtoMessage	Yes
Marry	Bride's character ID	ProtoMessage	Yes
AcceptRejectProposal	Boolean	ProtoMessage	Yes
TryForChild	None	ProtoMessage	Yes
SiegeCurrentFief	None	ProtoMessage	Yes
ViewSiege	Siege ID	ProtoMessage	Yes
SiegeRoundStorm	Siege ID	ProtoSiegeDisplay	Yes
SiegeRoundReduction	Siege ID	ProtoSiegeDisplay	Yes
SiegeRoundNegotiate	Siege ID	ProtoSiegeDisplay	Yes
EndSiege	Siege ID	ProtoMessage	Yes
SpyFief	Character ID of spy and target fief ID	ProtoMessage	Yes
SpyCharacter	Character ID of spy and target ID	ProtoMessage	Yes
SpyArmy	Character ID of spy and target army ID	ProtoMessage	Yes
ViewJournalEntries		ProtoGenericArray <ProtoJournalEntry>	No
ViewJournalEntry	Journal ID	ProtoJournalEntry	No
Kidnap	Character ID of target and Kidnapper	ProtoMessage	Yes
ViewCaptives	None	ProtoMessage	Yes
RansomCaptive	Character ID of captive	ProtoMessage	Yes
ReleaseCaptive	Character ID of captive	ProtoMessage	Yes
ExecuteCaptive	Character ID of captive	ProtoMessage	Yes
RespondRansom	Character ID of captive	ProtoMessage	Yes

UseChar	None	ProtoMessage	No
hireNPC	Character ID of NPC	ProtoMessage	No
fireNPC	Character ID of NPC	ProtoMessage	No
AppointBailiff	Fief ID	ProtoMessage	No
RemoveBailiff	Fief ID	ProtoMessage	No
BarCharacters	Fief ID and Character IDs	ProtoMessage	No
UnbarCharacters	Fief ID and Character IDs	ProtoMessage	No
BarNationalities	Fief ID and List of nationality IDs to unbar	ProtoMessage	No
UnbarNationalities	Fief ID and List of nationality IDs to unbar	ProtoMessage	No
GrantFiefTitle	Fief ID	ProtoMessage	No
AdjustExpenditure	Fief ID	ProtoMessage	No
AutoAdjustExpenditure	Fief ID	ProtoMessage	No
TransferFunds	Fief ID of the sender and receiver and amount	ProtoMessage	No
TransferFundsToPlayer	Fief ID of and character ID and amount	ProtoMessage	No
AdjustCombatValues		ProtoMessage	No
Attack	Army ID of attacker and target	ProtoMessage	No
EnterExitKeep	Character ID	ProtoMessage	No
ListCharsInMeetingPlace		ProtoMessage	No
Camp	Character ID and days	ProtoMessage	No
AddRemoveEntourage	Character ID	ProtoMessage	No

Table 3 function list of controller.cs

Function name	introduction
login	Player login to the server and server IP is changed in this function.
SiegeList	List all the sieges and their status.
Profile	Give the player profile.
FiefDetails	Show the current fief details
ViewMyFiefs	List all the fief player has.
ArmyStatus	List all the player's army and their details.
MaintainArmy	Maintain the army and this army and player must in the same fief.

AppointLeader	Appoint a leader for an army. Leader, army and player must in the same fief
HireTroops	Hire troops for an army, this army and player must in the same fief.
DisbandArmy	Disband an army, this army and player must in the same fief.
Marry	Propose marriage between two characters.
AcceptRejectProposal	Accept or reject marriage proposal.
TryForChild	Player character tries for a child with his wife.
SiegeCurrentFief	Siege the fief the player currently in and this fief must be other players.
ViewSiege	List all the siege and all their information
SiegeRoundStorm	The storm round consists of a reduction round.
SiegeRoundReduction	The Reduction round consists of a reduction round.
SiegeRoundNegotiate	The negotiation round consists of a reduction round.
EndSiege	End the siege.
SpyFief	Player request to spy on another player's fief.
SpyCharacter	Player request to spy on another player's character.
SpyArmy	Player request to spy on another player's army.
Kidnap	Kidnap a target character. The player must own kidnapper, and both target ID and kidnapper ID must be valid Character IDs.
ViewCaptives	List the player's all captives.
RansomCaptive	Player request to ransom captive to his owner.
ReleaseCaptive	Player request to release a captive.
ExecuteCaptive	Player request to kill a captive.

Table 4 introduction of used function

4.3.2 Other control Classes

Each scene has its own control class. This control is the sub-class of the controller and each class set up button listener for each scene and control their input and output.

Scenes	Controller
army	armyControl.cs

armyRes	armResControl.cs
Family	familyControl.cs
FiefDetail	DetailController.cs
kidnap	kidnapControl.cs
LoginScreen	loginControl.cs
map	mapControl.cs
profile	profileControl.cs
siege	siegeResControl.cs
siegeOptions	siegeControl.cs
spy	spyControl.cs

Table 5 comparison table of scenes and controller

For example, in the profileControl.cs objects back, title and report are the corresponding objects back, title and report on the scene. The Start() is a function to initialize the scene(set up the text and add button listener) while loading the profile scene. backListener() is the script of the back button.

```

public class profileControl : Controller
{
    // Use this for initialization
    public Text title;
    public Text report;

    public Button back;

    void Start()
    {
        back.onClick.AddListener(backListener);
        report.text += ("-----\n");
        report.text += ("Player Profile\n");
        report.text += ("-----\n");
        report.text += ("Player ID: " + c.playerID+"\n");
        report.text += ("Player Name: " + c.firstName + " " + c.familyName+"\n");
        report.text += ("-----\n");
        report.text += ("Owned Fiefs: \n");
        bool written = false;
        foreach (var fief in c.ownedFiefs)
        {
            if (written == false)
            {
                report.text += (fief);
                written = true;
            }
            else
                report.text += (" , " + fief);
        }
        report.text += ("\n");
        report.text += ("-----\n");
        report.text += ("Location: " + c.location + "\n");
        report.text += ("Army: " + c.armyID + "\n");
        report.text += ("Purse: " + c.purse + "\n");
        report.text += ("-----\n");
    }

    void backListener()
    {
        SceneManager.LoadScene(1);
    }
}

```

Figure 22 profile control

5 Evaluation

5.1 Function Evaluation

The function evaluation will be a series of test cases; these test case will evaluate the functionality of the client. Each test case will set up a test process and result from expectation. And the test result will be recorded after the test. If the test result is the success that means the test result the same as the expectation, so that function reaches the objective. If the test result is different from the expectation, I will analyse the causes of failure.

Case 1: login test

Test process: In the login scene enter the login name helen and the password potato. Then press the login button.

Expectation: Server receives the data message and sends reply message success. Then client switch Detail scene

Result: Success

```
Accepted connection from helen
SERUER: recieved data message
helen logs in from 127.0.0.1:61245
SERUER: recieved data message
From hist_mmorpg.Client: request = ViewChar, reply = Success
SERUER: recieved data message
```

Figure 23 test case 1 result

Case 8: Check Current fief detail test

Test process: After login, the current fief detail will display in the screen.

Expectation: Base on the CSV file on the server, the client should display the detail of fief EPM02. And the server should receive the request about view fief.

Result: Success



```
SERUER: recieved data message
From hist_mmorpg.Client: request = ViewFief, reply = None
SERUER: recieved data message
```

Figure 24 test case 2 result

Case 3: move test

Test process: This test is testing the function of travelling between different fief. In the detail, scene clicks map button and enter the map scene then go east twice and go north-east once. The go back in a different way go west twice then go south-west once.

Expectation: The server receives the request of travel to, reply success and the client displays fief ID in this order:

EPM02->ESW07->ESW06->ESW03->ESW04->ESW05->EPS02.

Result: Success

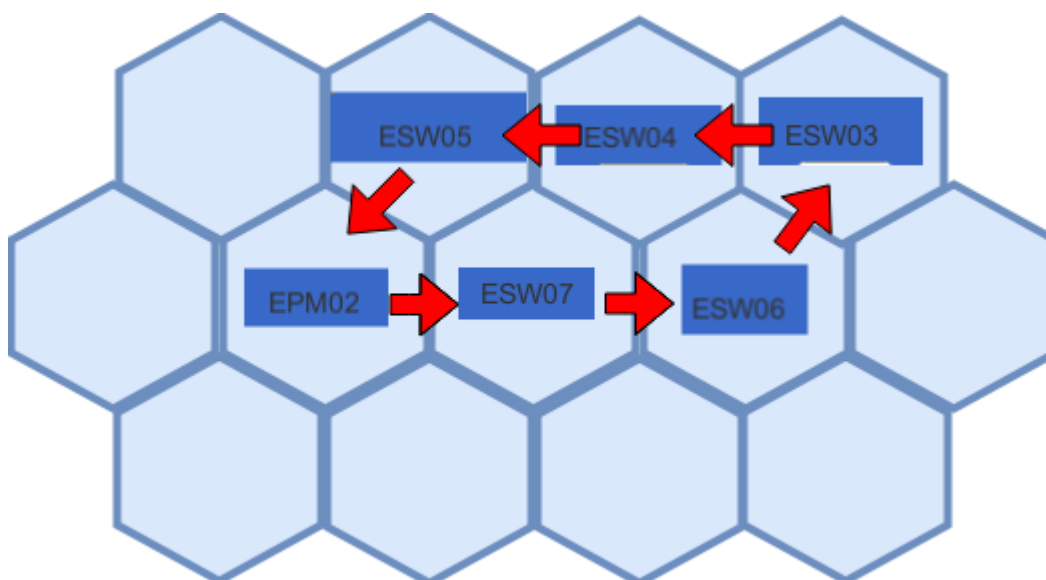


Figure 25 diagram for the travel path

```
SERVER: recieved data message
From hist_mmorpg.Client: request = TravelTo, reply = Success
```

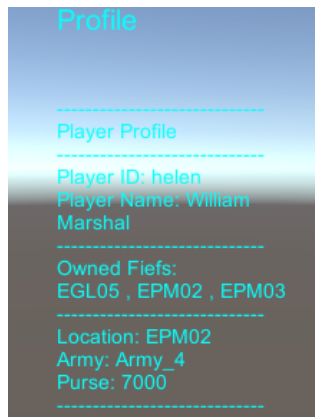
Figure 26 test case 3 result

Case 4: check the user profile

Test process: In the detail scene, click the profile button and enter the profile scene and check the detail.

Expectation: The server receives the request to view character and replies success. on the client-side display the character information.

Result: Success



```
SERVER: recieved data message  
From hist_mmorpg.Client: request = ViewChar, reply = Success
```

Figure 27 test case 4 result

Case 5: check army list

Test process: In the detail scene, click the army button.

Expectation: After click button, the client scene changes to the army scene and list the army. On the server-side receive the request and reply to the message.

Result: Success



```
SERVER: recieved data message  
From hist_mmorpg.Client: request = ListArmies, reply = None
```

Figure 28 test case 5 result

Case 6: hire 30 troops

Test process: In this test case is have 2 parts the first part is trying hire troop on another player's fief, go to the fief ESW07 and go to army scene and enter the amount 30 then click the hire button. The second part is back to the Fief and hires 30 troops again.

Expectation: The first part client will receive the request recruit troops and reply character recruit own and client display fails message. Then the second part the server will replay success message and the client will display the successful message and hire detail.

Result: success.



Figure 29 test case 6 result

Case 7: maintain army 4

Test process: In the army scene enter the arm ID Army_4 and press maintain button.

Expectation: The server will receive the maintain army request and reply success. And the client and display the success message.

Result: Partly succeeded. The server receives the right request but because funds shortage cannot carry out maintenance. In the game design maintain the army needs the funds and on the server-side the funds are incomplete. So that all the player funds are 0 and player have no funds maintain an army.

```
SERVER: recieved data message
From hist_mmorpg.Client: request = MaintainArmy, reply = ArmyMaintainInsufficientFunds
```

Figure 30 test case 7 result

Case 8: disbanded army 1

Test process: In the army scene enter the army ID Army_1 and click the disband button.

Expectation: The server received disband army request and in there is only Army_4 in the army list in the client.

Result: Success.

```
Sending update ArmyDisband to helen  
From hist_mmorpg.Client: request = DisbandArmy, reply = Success
```

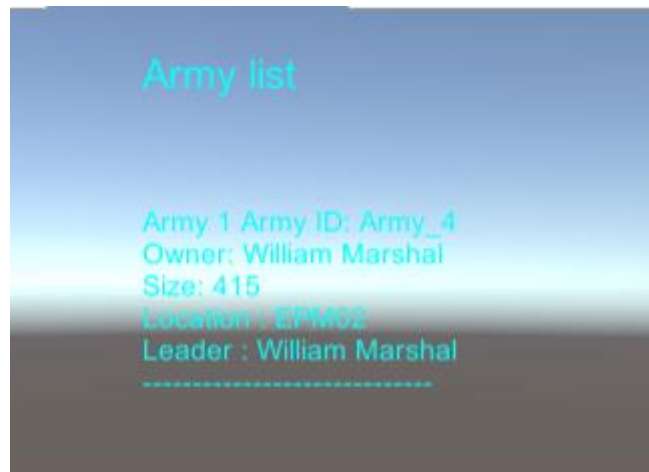


Figure 31 test case 8 result

Case 9: appoint a leader for the army

Test process: Enter the army ID army_1 and the character ID Char_123456 in the army scene and click the hire button.

Expectation: In Army Scene, Army_1 leader displays new leader name, W Marshal. In the server, side receives appoint leader request and reply success message.

Result: The client successful send the request to the server. But the server stops working after receiving the message because of the array's index out of range inline 849, Army.cs. To fix this problem.

Case 10: propose marriage

Test process: In the detail scene press, the family button then enters the bride character ID. Then click the proposed marriage.

Expectation: The server will receive the propose marriage request and reply success. And the client and display the success message.

Result: Partly succeeded. The server receives the request, but I check all the female NPC from the CSV file of the game object, none of them is available for marriage. In the future development need to add more game object.

```
From hist_mmorpg.Client: request = ProposeMarriage, reply = CharacterProposalMarried
```

Figure 32 test case 10 result

Case11: accept and reject marriage

Test process: In the family, click accept or reject button.

Expectation: Because there is only one available player account on the server and there is not register function on the server-side, no other player can send the propose marriage request to the test account. This test case is only tested client can send the right type request to the server.

Result: Succeed. The server needs to add more player account to take the complete test.

```
SERVER: recieved data message  
From hist_mmorpg.Client: request = AcceptRejectProposal, reply = ErrorGenericMessageInvalid
```

Figure 33 test case 11result

Case 12: try for a child

Test process: In the family scene click the try for a child.

Expectation: The server will receive the try for child request and send back the result.

Result: Success the player character's Spouse pregnant.

```
SERVER: recieved data message  
From hist_mmorpg.Client: request = TryForChild, reply = CharacterSpousePregnant
```

Figure 34test case 12 result

Case 13: kidnapping a character

Test process: In the kidnap scene, enter target character ID Char971, Kidnapper character ID and click kidnap button.

Expectation: The server will receive the kidnap request and reply kidnap result. And the client display result.

Result: Partly succeeded. The server receives the right request from the client. But because the player 's character is dead in the action. The game

```

SERVER: recieved data message
From hist_mmorpg.Client: request = Kidnap, reply = KidnapNoPlayer
SERVER: recieved data message
Kidnap success: -4.692,SuccessChance: 35.8347625242708,EscapeChance: 43.7836471566854
Escape chance: 43.7836471566854
Detected
Killed
Char_158 dies; Char_971 inherits
Debug: role is : Son & Heir
Update helen: YouDied
Sending update YouDied to helen
Update helen: newEvent
Sending update newEvent to helen
Update helen: newEvent
Sending update newEvent to helen
Update helen: KidnapFailDead
Sending update KidnapFailDead to helen

```

Case 14: ransom/ release/ execute the captive

Test process: In the kidnap scene, enter the target character ID and click ransom/ release/ execute button.

Expectation: The server will receive the ransom/ release/ execute request and reply ransom/ release/ execute result. And client display ransom/ release/ execute result.

Result: Partly succeeded, Server receives the right request from the client. But because on the server side I did not find NPC meet the conditions to kidnap, the server only can reply error message cannot find captive.

Case 15: siege current fief

Test process: In this test case is have 2 parts. The first part is siege your own fief. In the siege scene and click siege current fief button. The second part will try to siege the fief.

Expectation: The server will receive the besiege fief request and the first part will reply success and second will reply pillage own fief. And client display r relevant result.

Result: Success

```

Sending update newEvent to helen
From hist_mmorpg.Client: request = BesiegeFief, reply = Success
SERVER: recieved data message
From hist_mmorpg.Client: request = BesiegeFief, reply = PillageOwnFief

```

Figure 35 test case 15 result

Case 16: siege around the storm/ negotiate/ negotiate

Test process: In the siege scene, enter siege ID and click siege around the storm/ negotiate/ negotiate button.

Expectation: The server will receive the click siege around the storm/ negotiate/ request and reply ransom/ release/ execute the result. And client display clicks siege around the storm/ negotiate/ result.

Result: Success.

```
update helen: newEvent
Sending update newEvent to helen
From hist_mmorpg.Client: request = SiegeRoundStorm, reply = None

Sending update newEvent to helen
From hist_mmorpg.Client: request = SiegeRoundNegotiate, reply = None
SERVER: recieved data message
From hist_mmorpg.Client: request = SiegeList, reply = None
```

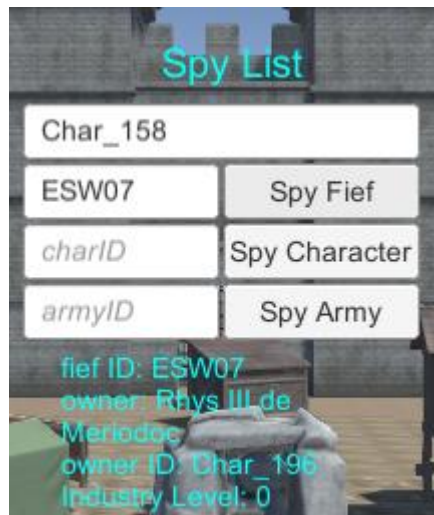
Figure 36 test case 16 result

Case 17: spy a fief

Test process: In the siege scene, enter fief ID, spy character ID and click spy fief button.

Expectation: The server will receive the spy fief request and reply success result. And client display target detail.

Result: Success.



```
Sending update SpySuccess to helen
From hist_mmorpg.Client: request = SpyFief, reply = Success
```

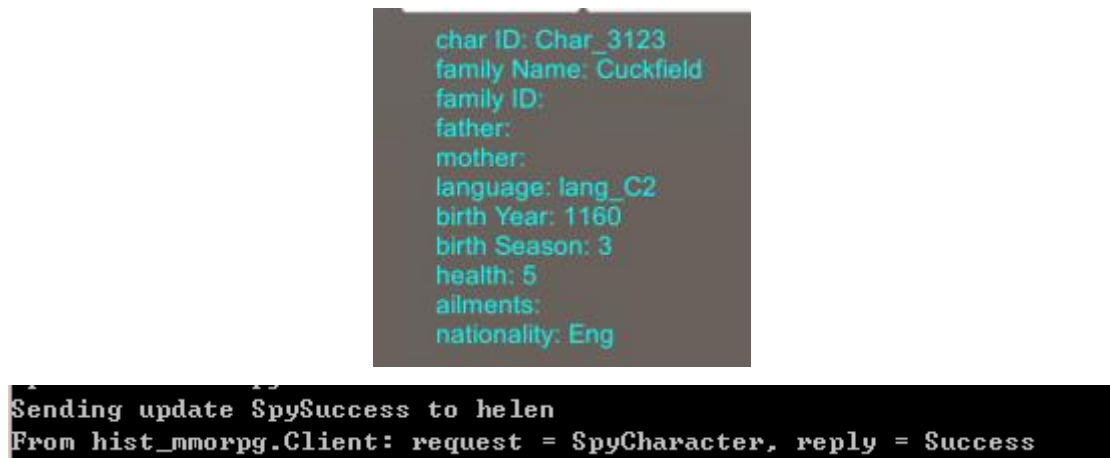
Figure 37 test case 17 result

Case 18: spy a character

Test process: In the siege scene, enter fief ID, spy character ID and click spy character ID.

Expectation: The server will receive the character ID request and reply success result. And client display target detail.

Result: Success.



```
char ID: Char_3123
family Name: Cuckfield
family ID:
father:
mother:
language: lang_C2
birth Year: 1160
birth Season: 3
health: 5
ailments:
nationality: Eng

Sending update SpySuccess to helen
From hist_mmorpg.Client: request = SpyCharacter, reply = Success
```

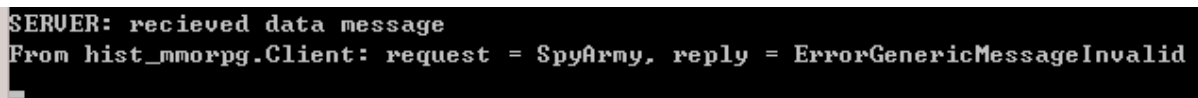
Figure 38 test case 18 result

Case 19: spy an army

Test process: In the siege scene, enter fief ID, spy army ID and click the spy army button.

Expectation: The server will receive the spy character request and reply success result. And client display target detail.

Result: Partly succeeded. The server receives the right request. But there are not emery army in the server so will reply error message.



```
SERVER: recieved data message
From hist_mmorpg.Client: request = SpyArmy, reply = ErrorGenericMessageInvalid
```

Figure 39 Test case 19 result

5.2 User Evaluation

The user evaluation is will let tester finish a few tasks by using unity client and finish a questionnaire. This questionnaire is used to test the usability of the unity client and evaluation hypothesis the player will interest in history after playing Historical game. The questionnaire has three parts the background, interface and gameplay. Each part has a few question need to answer testers. The tester will answer the question for background before playing the game and answer the next 2 parts after playing the game. Most of the testers are students from the school of mathematics and computer science at Heriot-Watt University. In the questionnaire, some questions will require the tester to use a linear scale to answer them. For all the Linear scale 1

means strongly disagree or strongly dislike, 2 means disagree or dislike, 3 means not sure, 4 means agree or like and 5 mean strongly agree or strongly like.

There are three complement questions for further study after the test (table 19, table 24, table 25). These questions are emailed to the same testers who doing the questionnaire before.

5.2.1 Background

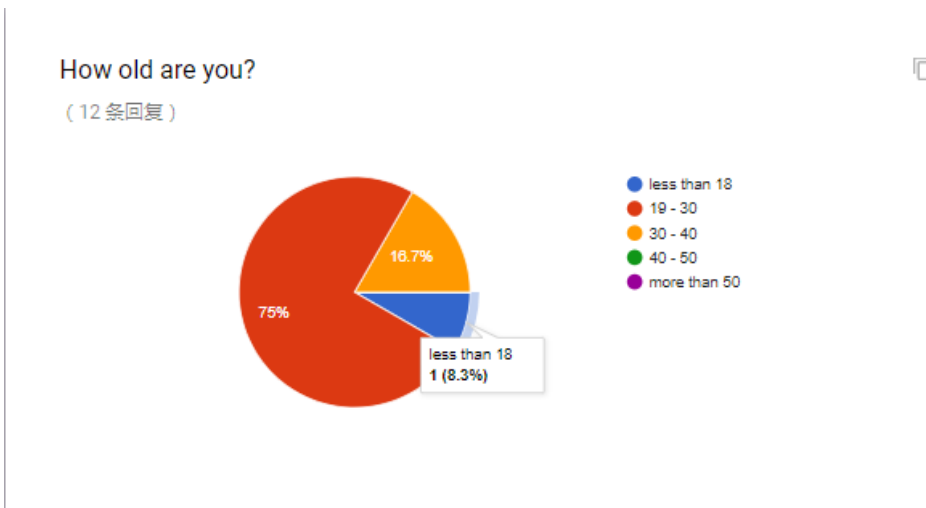


Table 6

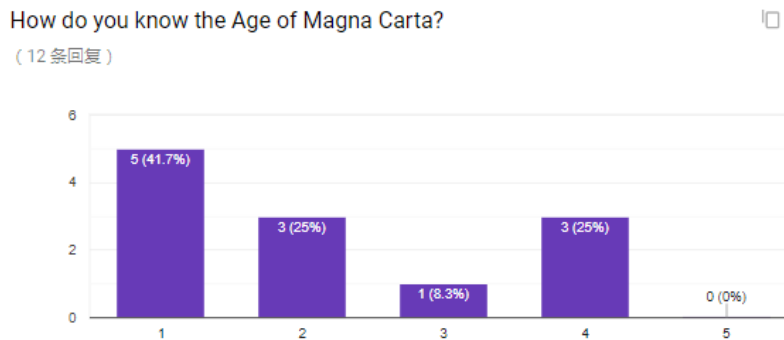


Table 7

How often you use computer every week

(12 条回复)

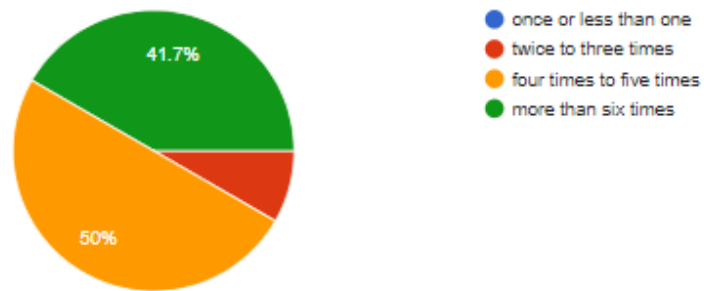


Table 8

How often you play computer game every week?

(12 条回复)

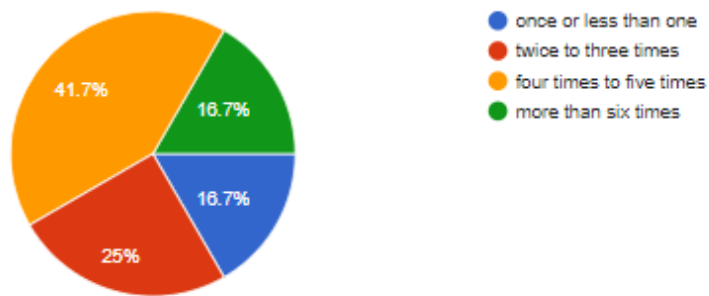


Table 9

Do you think people can learn something through computer game?

(12 条回复)

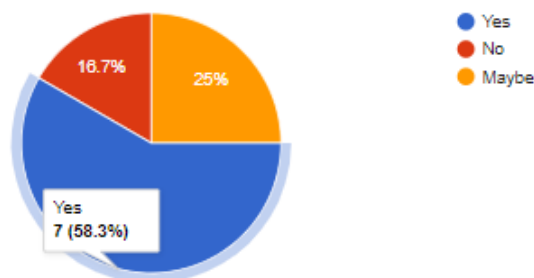


Table 10

Do you like historical computer games?

(12 条回复)

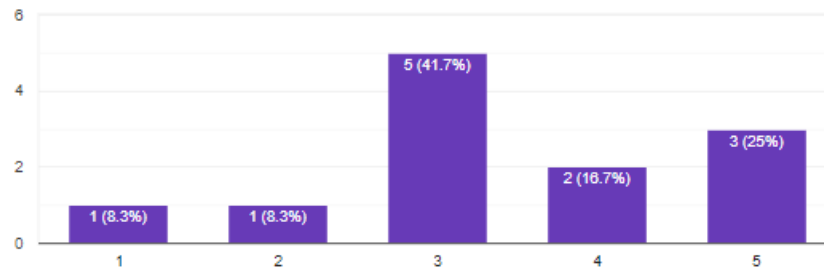


Table 11

In the background, the question is focus on tester's background. Because there are lots of students in the test, most people are in the age group 18 to 30. And Heriot-Watt is an international university, most of the student is from outside of the UK. So that only a few people think they know the Age of Magna Carta, most of the people never hear Age of Magna Carta before the test. From table 7, table 8 and table 10, we can find for most people computer and computer game is become an important part of our daily life, but 5 testers do not care of the topic of the game and 5 testers like historical games. These historical game lovers all believe they can learn from the game (table 11). And before the playing the game nearly half tester not sure or do not believe they can learn from the game.

Summary: Testers are young and most of the tester are experience, computer user or game player. And tester who likes historical games they always can learn from the game.

F	G
Do you think people can learn something through computer game?	Do you like historical computer games?
Yes	5
Yes	5
Yes	4
Yes	4
Yes	5

Table 12

5.2.2 interface

Do you think this client interface is intuitive?



(12 条回复)

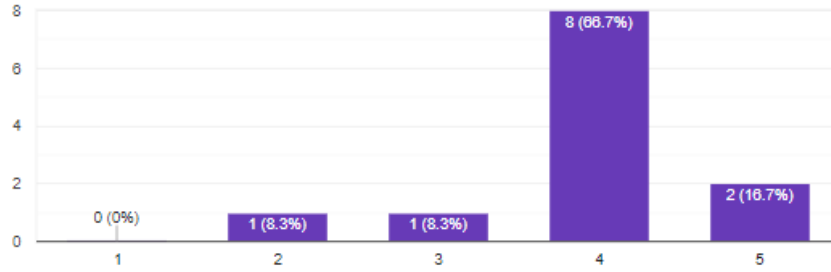


Table 13

Do you think this client interface given enough information to help you to play game?



(12 条回复)

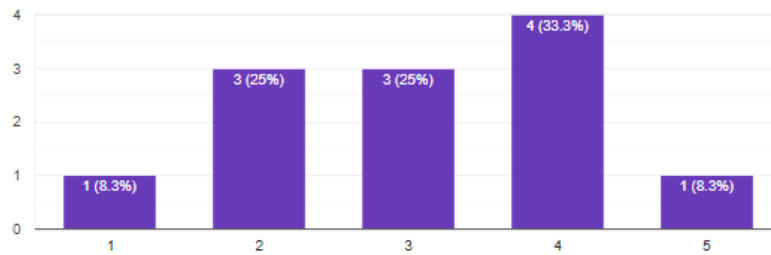


Table 14

Do you think the interface is unnecessarily complex?



(12 条回复)

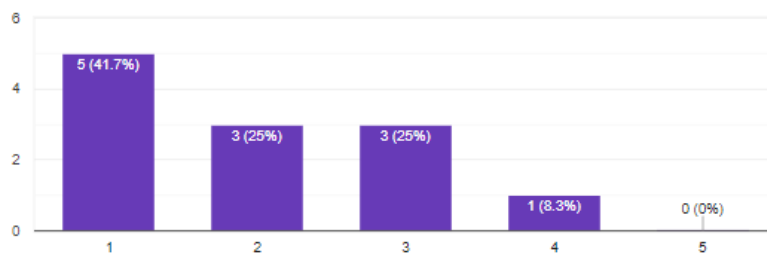


Table 15

Do you think the client background match the game theme?

(12 条回复)

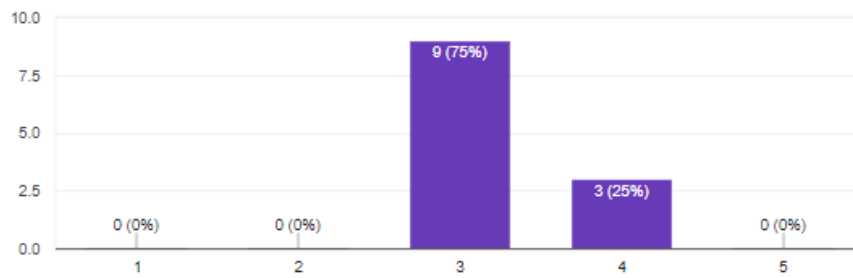


Table 16

Do you think the text clear to read in the game?



(12 条回复)

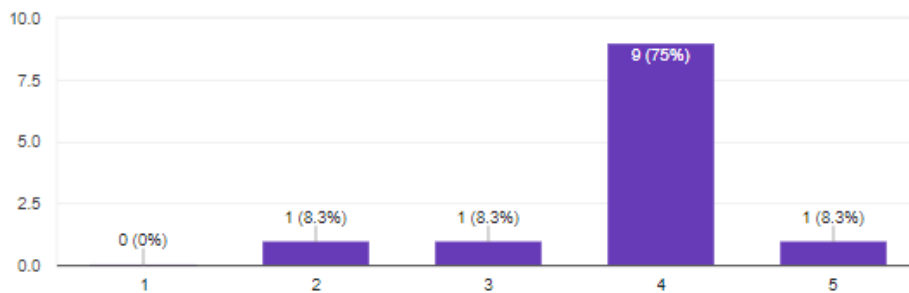


Table 17

Do you think the information easy to understand in the game?



(12 条回复)

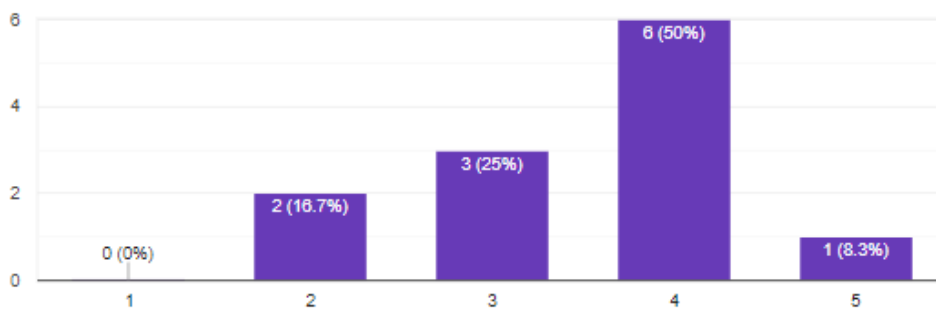


Table 18

For the number display in the game(e.g army size)Do you prefer touse
grpahical representation rather than text?

(12 条回复)

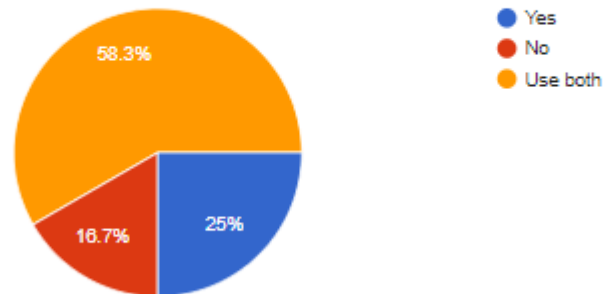


Table 19

For the interface part, most of the features of the client reach the expectation, but there are still few points need to be noticed. The first problem is in the question “Do you think this client interface given enough information to help you to play a game?” (table 13). The number of disagreeing and agree are 4 and 5. It almost half-and-half. Base on the discussion after the test, most of the testers who disagree have a problem are using functions which need enter information, sometimes they do not know which function need enter the information in the army, siege and kidnap scenes. The next problem is text colour one of the testers suggest player should customize the text colour because the current colour is not clear to read in few places and this colour is not friendly for tritanopia (blue blindness) people. Then is about “Do you think the client background match the game theme?” (table 15). Because of the background of the testers (table 6), most of them are not familiar with the Age of Magna Cart. Therefore, they do not know the background matches the theme of the game. Last but not least, some testers advised the different should use a different background, the current background is not suited for the army, siege, Kidnap and family scene. In table 18, more than half testers believe that the best way to display the number in the game is Combined with graphical representation and text

Summary: User guide and customize text colour should add to the game client in future development. About background more study of the Age of Magna Cart is

needed, the method for the study is not limited to consult new literature, field trips and consult a specialist. Add the chart for the number displaying in the client.

5.2.3 Gameplay

Did you learn something new about this time period?

(12 条回复)

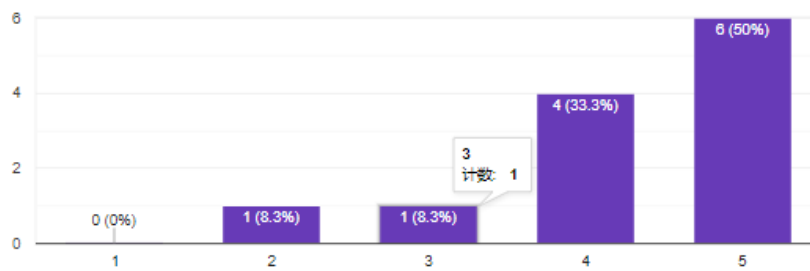


Table 20

Would picking a (historical) starting PC make the game more interesting to you?

(12 条回复)

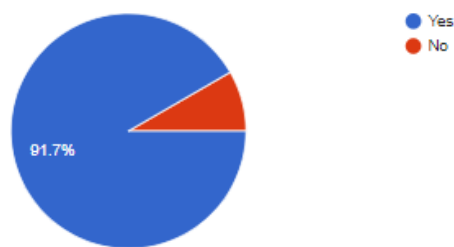


Table 21

After having played the game, would you be interested to read-up on this historical time period?

(12 条回复)

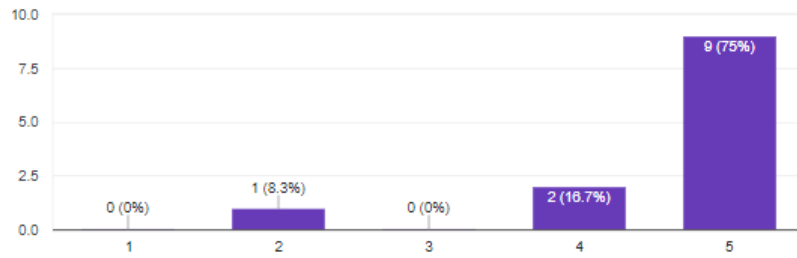


Table 22

Do you think that the available in-game actions (such as sieging, fief mgmt etc) are a good reflection of the historical time period?



(12 条回复)

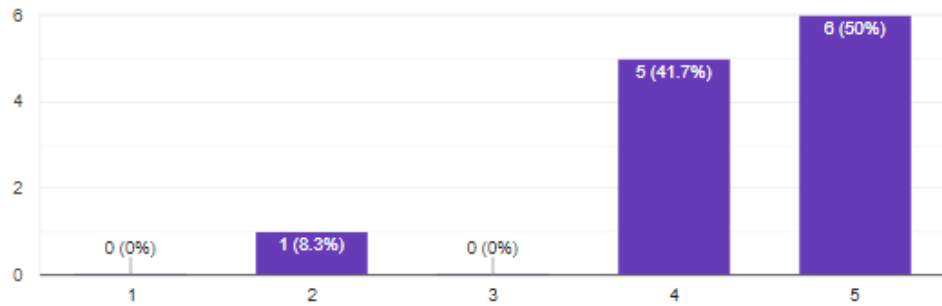


Table 23

When would you play this game?

(12 条回复)

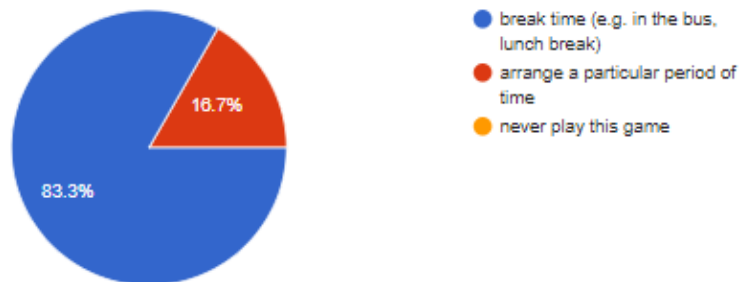


Table 24

How long would you play this game each time last



(12 条回复)

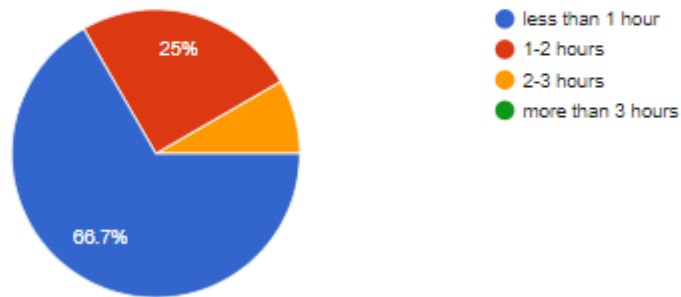


Table 25

The results of the last part of user evaluation are inspiring. Almost all the testers show the interesting about the history of the Age of Magna Cart. And there is only one person will do not want to the future study of the Age of Magna Cart. Almost all the testers think to play as a real historical character is more attractive in the game. Combining tester background, this tester is not playing the game in daily life. So I speculate for a person who does not play the game, this person will not show the interesting about learning history after the player historical game. 10 testers prefer to play this game in the break time rather than arrange a particular period of time. And most of them do not want to spend too much time on this game for each time last. These data recommend that ever game will help actualize the education, but the game still cannot instead of traditional education. The education game is the assistant tool for education.

Summary: The historical game will help most of the people who play learn history. For the people not playing game, more test still needs to take. Because most of the testers are interested in the read-up on this time period after playing the game, some historical documents and a list of recommended readings can be added into the game client. And base on most of the testers is not showing the willingness to play this game for a long time. The text for the historical documents should be terse and informative. For the same reason, it should not take too much time for each new game. Most people want to play it in the break time, the mobile client must be on the agenda. The story mode for the real historical character can be added to the game too. There is three advantage for story mode, the most of tester wants to play as a real historical character, so story mode should be attractive for these testers; the

player can learn this character life and the important events for a historical time period; historical documents can be added into the game more naturally.

6 Conclusion

The historical game – JominiEngine can help people to learn the history of the Age of Magna Carta. People generally believe game and education is so opposed that they just cancel each other out. Students will neglect their studies while they are playing the game. And good students never play games. However, through this project, it gives insights into the educational value of a game designed to teach about the historical context in a medieval time period. This context is significantly different from a modern context to make it an interesting object of study, and the study helps to challenge pre-conceptions and ideas implanted by popular histories such as movies and novels. As such it can be seen as a supporting tool alongside more traditional forms of history education. If people find something interested them in the game, they are willing to learn related information after playing the game. To help the player learn the history better, the historical document and story mode should be added into the game. The contents are very important in the game. Different contests will lead people to learn the different topic. So, for an educational game, the game content needs to be stricter reviewed. Last but not least the educational game still cannot instead of traditional education, but the educational game still a useful tool to assist people in learning.

Unity Is a useful tool-box to build a graphical client. It is good for a none experienced game programmer. It has massive assist from the unity asset store. Programmers can easily understand how control can render the scene in the game. C# as the implementation language provides a range of advanced language constructs to help with the development of the client. C#'s strong library support to help me finish this project more efficiently. And this project has achieved to build an appealing, graphical client, based on a portable communication layer that interacts with the game server in a timely manner. A networked setup has been tested. However, the programmer still needs to note different .Net framework versions brings structural issues. The client-server connection has passed the test by using a local area network between a server and a personal computer. In the server is using the Kylin Linux system (IP:192.168.110.125) and the personal computer is using windows 7 system(IP:192.168.110.123). (These devices are provided by east China institute of computing technology, Shanghai)

New unity client gained a certain level of success. It added a substantial set of more functions known by the server. The background is added for each scene with free 3D models from the unity asset store. But there are still many problems in the unity client. To fix this problem, the first step is modifying the server function and database, In the function evaluation, some test cases are not able to pass because game data is not consolidated in the server. Also, the client still has lots of work to improve and all the objective to improve the game will in the 6.2 future development.

6.1 Finish function in the client-side

- Login to the server;
- Check profile;
- Check fief detail;
- Travel in hexagonal Map;
- Army management;
- Siege management;
- Spy management.

6.2 future development

6.2.1 server

- Fix the error in the appoint leader function;
- Add more data in the game database;
- Add registration functions;
- If possible refactoring the code use .Net framework 3.5;
- Add script for story mode for the historical character;
- Add following function in the server-side:
 - Anathema (players throwing players out of the game)
 - Archbishops (and special tax)
 - The function of Traveling Companions, Seduction
 - Diplomacy
 - Divorce
 - Assassination, Seizing and Cursing
 - English Strategy
 - Financial Activity
 - Fleet, Assemble
 - French Strategy
 - Gamble
 - Household Affairs

- Jousting
- Keep Level
- Law and Order
- Lock Out Option
- Overlord Titles (stripping and transfer)
- Purge
- Scoreboard
- Skill Information
- Skills and Characteristics
- Standing Order
- Strategy Tips
- Winning the Game

Note: all the new functions add to the server also need to add in the Dynamic Link Library ProtoMessage (page. 26) for client use.

6.2.2 client

- Add user guide in the game;
- Add function to customize text colour;
- Do further research for the Age of Magna Carta;
- Add different backgrounds for the different scene;
- Add unused function in table 3 into the client;
- Synchronously update new function from the server;
- Add a short historical document;
- Add story mode for the historical character;
- Mobile client;
- Take more user surveys to optimize the user experience.

7 Appendix

Questionnaire

- The questionnaire is in this page
https://docs.google.com/forms/d/e/1FAIpQLSfuO1BeUOndqXYj8UrIT5n4HAaKsjIDb4JRTCvHFWEVAC_kzw/viewform?usp=sf_link
- Extend questions:
https://docs.google.com/forms/d/e/1FAIpQLSfZd0r5d5Ed8oJiZbHWCwgbrvBRLDIbC6RuSwlo3uD9logoFQ/viewform?usp=sf_link

Code for the server, unity client and ProtoMessage:

<https://drive.google.com/file/d/1BheCTU2Mffpn9LxE8vbl8Bsc9-FqBhYP/view?usp=sharing>

File introduction:

Folder	Introduction
Server	The Rory's server
ProtoMessage	The Dynamic Link Library for client
MscProject	Unity client
MscProject\demo	Unity client demo
MscProject\Assets\Scenes	Scene and C# code for unity

Unity vision is unity 2018.3.8f1 personal/

VS vision is VS community 2015 14.0.25431.01 Update 3

How to execute game:

1. Execute the server from \server\DissertationHistMMORPG-master\RepairHist_mmo\bin\Debug\hist_mmorpg.exe
2. Execute the client from \MscProject\demo\MscProject.exe
3. Username is helen and password is potato

Consent Form

About of project

This project aims to build the available unity client for a historically accurate middle ages massively multiplayer online role-playing game to help the learning and understand this phase of history. This game is called JominiEngine. You will play as one of overlord in the Age of Magna Carta in the mainland of Britain in the time period of 1194-1225. As an overlord, you need to compete with other overlords and conquer them to win the game. To reach this purpose, you will need to build up your strengths by developing your fiefs, reinforcing your armies, managing your family and even use some dirty moves.

About of study

In the coming time, you will take part in this project. You will use the unity client to finish a few steps task and finish and answer a questionnaire to help me to study the usability of the interface of the unity client.

Data Protection

In the questionnaire will collect some personal information only for studying the usability about the interface of the unity client. All the question will be anonymized and you will get a serial number of your questionnaire. This serial number is no one knows but you. And your email address also needs to provide. Your e-mail will only use for further study about this project and request delete your experiment record. You can delete your questionnaire and email any time you want and you do not need to give any reasons. The only thing you need to do to delete your questionnaire and email is sending an email with your serial number or your email address to yz6@hw.ac.uk.

Consent

I agree to volunteer to complete the usability study and understand I can withdraw at any time without giving a reason for doing so during the study or up to 7 days after the study's completion. I agree to allow for the collection of anonymised data during the study which will then be used to report upon the efficacy of the usability of the clients. I agree to allow for information about me to be stored in compliance with the Data Protection Act of 2018 by Yuanqi Zhu.

Signature:

E-mail:

Remote Test Plan

1. Login to JominEngine via the unity client, using the username 'helen', the password 'potato'
2. Check your army information
3. Hire 20 troops from the current fief
4. Move to the northeast
5. Check this fief information
6. Siege the current fief
8. Check the fief result
7. You can play with other function in the unity client

8 Reference

- ShepHertz App42 Blog. (2018). Picking the right communication protocol for your game - ShepHertz App42 Blog. [online] Available at: <http://blogs.shephertz.com/2013/01/28/picking-the-right-communication-protocol-for-your-game/#> [Accessed 26 Mar. 2018].
- Lutz, M. (2017). Learning Python. Sebastopol [etc.]: O'Reilly.
- Baydan, İ. and Baydan, İ. (2018). How To Execute Shell Command with Python - Poftut. [online] Poftut. Available at: <https://www.poftut.com/execute-shell-command-python/> [Accessed 27 Mar. 2018].
- Docs.python.org. (2018). 36.16. commands — Utilities for running commands — Python 2.7.14 documentation. [online] Available at: <https://docs.python.org/2/library/commands.html#commands.getstatus> [Accessed 27 Mar. 2018].
- RANKIN, H. (2016). Improving the Security and Correctness of a Historical Massively Multiplayer Online Role-Playing Game Server. MEng. HERIOT-WATT UNIVERSITY.
- Malcolm, R. (2017). An Exploration into Building Three Clients for JominiEngine. BSc Computer Systems. Heriot-Watt School of Mathematical and Computer Sciences.
- Sowizral, H. (1987). Design Methodology for Object-Oriented Programming OOPSLA Panel Session.
- RANKIN, H. (2016). Improving the Security and Correctness of a Historical Massively Multiplayer Online Role-Playing Game Server. MEng Software Engineering. HERIOT-WATT UNIVERSITY Mathematics and Computer Science.
- Bond, D. (2015). Design and implementation of a massively multi-player online historical role-playing game. MSc Advanced Internet Applications. Heriot-Watt School of Mathematical and Computer Sciences.
- Bond, D., Loidl, H. and Louchart, S. (2015). Design and Implementation of the Jomini Engine: Towards a historical Massively Multiplayer Online Role-

Playing Game. Heriot-Watt University, Riccarton Campus, Digital Design Studio, The Glasgow School of Art, The Hub, Pacific Quay.

- Troelsen, A. and Japikse, P. (n.d.). C# 6.0 and the .NET 4.6 Framework, Seventh Edition.
- GitHub. (2018). lidgren/lidgren-network-gen3. [online] Available at: <https://github.com/lidgren/lidgren-network-gen3> [Accessed 28 Mar. 2018].
- C# Language Specification. (2017). 5th ed.
- Brilliant.org. (2018). Finite State Machines | Brilliant Math & Science Wiki. [online] Available at: <https://brilliant.org/wiki/finite-state-machines/> [Accessed 6 Apr. 2018].
- RareSloth Games. (2018). Game UI design - RareSloth Games. [online] Available at: <https://www.raresloth.com/design/game-ui-design/> [Accessed 8 Apr. 2018].
- Chappell, L. and Tittel, E. (2004). Guide to TCP/IP. Cambridge, Mass.: Course Technology.
- Yadav, S. (2015). User Datagram Protocol.
- Mono-project.com. (2018). Home | Mono. [online] Available at: <https://www.mono-project.com/> [Accessed 11 Apr. 2018].
- Dhruv (2018). Picking the right communication protocol for your game - ShepHertz App42 Blog. [online] ShepHertz App42 Blog. Available at: <http://blogs.shephertz.com/2013/01/28/picking-the-right-communication-protocol-for-your-game/> [Accessed 11 Apr. 2018].
- Alxiangfeng (2018). 用 Unity3d 开发游戏的优点都有哪些？ - CSDN 博客. [online] Blog.csdn.net. Available at: <https://blog.csdn.net/JtNbCOC8N2I9/article/details/78557144> [Accessed 11 Apr. 2018].
- Baydan, İ. (2018). How To Execute Shell Command with Python - Poftut. [online] Poftut. Available at: <https://www.poftut.com/execute-shell-command-python/> [Accessed 11 Apr. 2018].
- Python Software Foundation (2018). 36.16. commands — Utilities for running commands — Python 2.7.14 documentation. [online] Docs.python.org. Available at:

<https://docs.python.org/2/library/commands.html#commands.getstatus>

[Accessed 11 Apr. 2018].

- rendongdeni (2018). python 语言的优点和缺点 - 忍冬的你 - 博客园. [online] Cnblogs.com. Available at: <https://www.cnblogs.com/rourou1/p/6039108.html> [Accessed 11 Apr. 2018].
- Webster, L. (2018). Visual Studio Development Features | Visual Studio. [online] Visual Studio. Available at: <https://www.visualstudio.com/vs/features/> [Accessed 11 Apr. 2018].
- Msdn.microsoft.com. (2018). System.Net Namespaces. [online] Available at: [https://msdn.microsoft.com/en-us/library/mt481553\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/mt481553(v=vs.110).aspx) [Accessed 11 Apr. 2018].