逢　甲　大　學

資 訊 工 程 學 系 碩 士 班

專題研究報告

蛇梯棋

Snakes & Ladders

指 導 教 授：　劉宗傑

學 生：　林季暉 M0302091

程國智 D0475117

中 華 民 國 一 百 零 五 年 五 月

摘要

蛇梯棋遊戲(Snakes & Ladders)，可為多人制的遊戲，因此此專題設計了可多人同時遊玩的蛇梯棋網路遊戲。此網路遊戲的通訊架構，為 Java 所提供的 Remote Procedure Call—Remote Method Invocation(RMI)。使用 RMI 的好處為可以省去許多處理 Socket 通訊問題的時間，因此可使專題的時間成本消費於功能的開發。此專題與其他的蛇梯棋比較，差別在於可動態置放蛇與梯子，也就是說可依照程式需求，輸入置放位置的座標後，蛇與梯子可於圖形化的棋盤介面顯示於相對應的位置。

關鍵詞：Remote Procedure Call、Remote Method Invocation、Snakes & Ladders

目錄

圖目錄

表目錄

第一章 緒論

1.1 蛇梯棋簡介

如下圖所示，遊戲介面為棋盤，大小 10*10，共 100 格，編號從 1~100，棋盤上有兩種特殊物件，分別是蛇與梯子。

蛇梯棋為回合制遊戲，可以多人遊玩，每個人皆會在同一輪中擁有甩骰子的權力，例如當玩家一甩完骰子則換玩家 2 甩骰子，直到所有玩家皆甩完骰子則進行下一輪。甩骰子可決定基本步伐，當玩家走完基本步伐，假若走上特殊物件，則可進行相對應的位移。遇見蛇頭，順流蛇尾；走上梯子，高升梯頭。

而此遊戲的獲勝者，為最先走到或超越格子編號 100 的玩家。當某位玩家達成此獲勝條件，則不再進行其他玩家的回合，且同時遊戲結束。

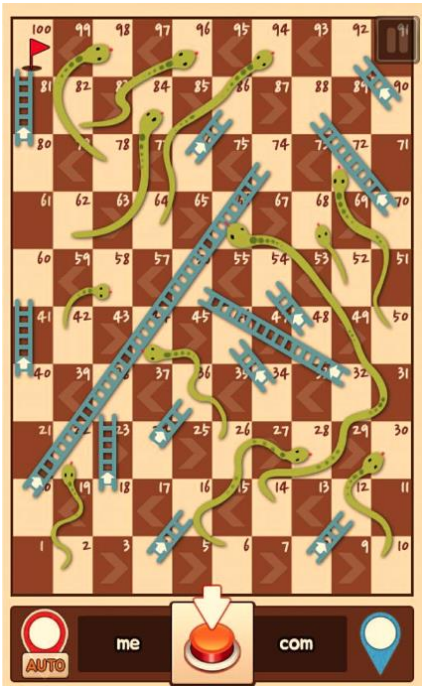

圖 1.1 蛇梯棋示意圖

1.2 研究動機

蛇梯棋可為多人制遊戲，因此我認為是適合設計為網路遊戲。假若只是單機

遊戲或實體的遊戲，則想要多人玩樂時，將時常因為人數不夠而始人興致缺缺。假若是網路遊戲則可以彌補此種缺憾，網路遊戲可連接處在相異位置卻有相同興致的玩家。

蛇梯棋上的物件多數是固定位置的，也就是不論你何時玩樂，棋盤上的物件皆固定在同一區塊。對於此種機率為重的遊戲，棋盤物件卻固定在同一區塊會造成遊戲一成不變。因此此次實作也將會實作隨機產生棋盤之蛇梯棋，以增加遊戲的不確定性，進而增加興致。

1.2 研究目標

以 JAVA 實作蛇梯棋介面，並以 Remote Method Invocation (RMI)時作網路的溝通。

Server 端可允許多個 Clients 同時註冊，多多個 Clients 同時遊樂。

遊戲為房間制，使用者可決定是否開房，或者進入已經開啟但尚未開始遊戲的房間。每個房間滿四人，即可開始遊玩。

遊戲棋盤可動態置放梯子與蛇。

1.3 工作分配及甘特圖

表 1.1 工作分配表

工作項目	工作分配
繪圖	林季暉
通訊	程國智
遊戲設計及程式開發	林季暉、程國智

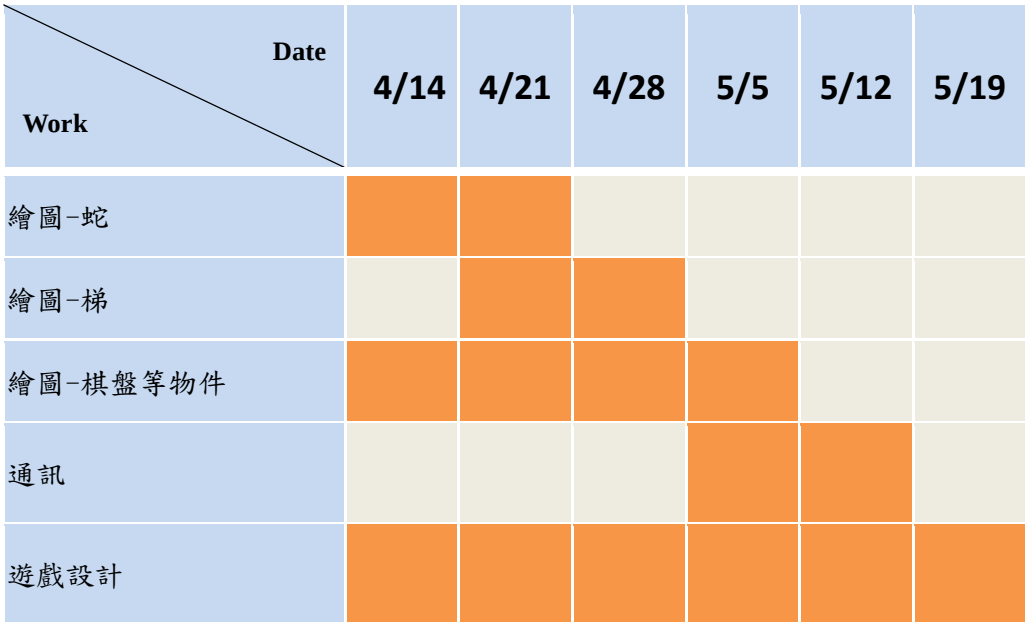

圖 1.2 甘特圖

第二章 文獻回顧與探討

2.1 Remote Procedure Call

Remote Procedure Call (RPC) 常被應用於開發網路程式的通訊方法。假若使用 Socket 介面撰寫網路應用程式，例如實際操作 Transmission Control Protocol (TCP)或 User Datagram Protocol (UDP)的網路應用程式，時常需要考慮各種通訊的細節，比如說資料型式與結構的轉換以及連線的管理等等。而 RPC 將簡化了這些細節，使程式設計者可將省去的時間用於開發應用系統的需求。

RPC 將網路通訊視為程式中的程序(Procedure)。當程式需要遠端程序(Remote Procedure)提供某種服務或接受訊息時，可直接呼叫(Call)一個程序，而此程序有相對應的遠端程序，可以於呼叫後於遠端的電腦上被執行。

而此專題所使用的 RMI 為 Java 所提供的 RPC，將於下個章節有詳細的解釋。

RPC 程式設計方面，首先可先將多個服務程序置放在本地端實作唯一單機的程式，接著再將此些服務依照以下步驟實作為遠端程序，而最終可為 RPC 程式。首先得將服務程序的框架以介面定義語言(IDL, Interface Definition Language) 定義，定義其服務程序名稱，及服務程序的輸入輸出。接著將此 interface 實作為可用的服務程序。Server Program 則將此服務程序註冊於 RMI Registry，並且當被遠端呼叫時，執行被呼叫的程序。Client Program 則 RMI Registry 要求呼叫被註冊的遠端程序。

第三章 研究方法

此專題為以 RMI 通訊為基礎的蛇梯棋網路遊戲，分為三個重點，RMI 的架構、Client 端 GUI 介面及其運算、Server 端的運算功能以及共同資源的管理，將於下三個小章節敘述其細節。

3.1 Remote Method Invocation 的實作

選擇 RMI 實作網路應用程式，是因為想將有限的時間成本投資於服務程序的開發，而不適 Socket 通訊的細節問題。

如上章節所言 RMI 為 Java 所提供的 RPC。其實作的程序如下圖所示，與 RPC 的實作雷同，但實作語言將固定為 Java。首先，得先以 Java 的 interface 定義所遠端程序的程序名稱以及資料結構等等。接著將此 interface 實作後，將之編譯(Compiler)為 Byte Code 後，再以 RMI Compiler 編譯後得到 RMI Skeletons 以及 RMI Stubs。RMI Skeletons 將予以 Server 端執行此些將被呼叫的程序。RMI Stubs 將予以 Client 端為依據而呼叫遠端的程序[1]。

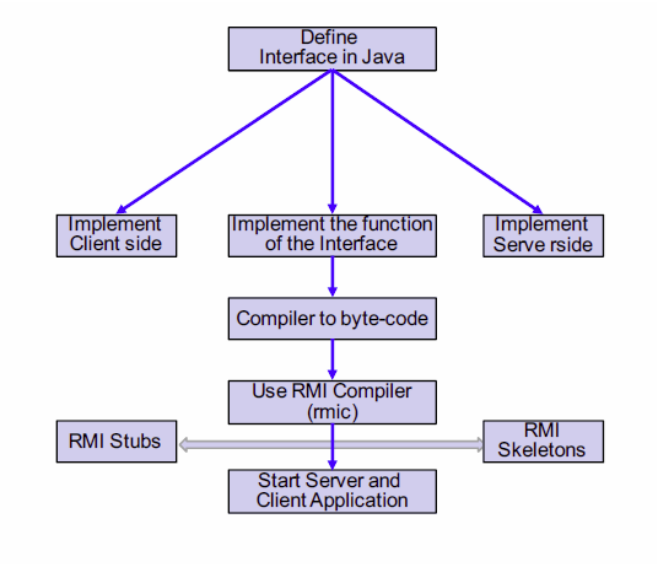

圖 3.1 Remote Method Invocation 實作程序圖

此專題以 RMI 為通訊基礎的應用程式架構如下圖所示，SnakeLaddersServer

會向 RMI Registry 註冊 SnakeLaddersRMIImpl 所實作的 SnakeLaddersInterface 之程序服務。而 SnakeLaddersClient 可藉由 RMI Registry 與 SnakeLaddersServer 連線，而後即可呼叫其所註冊之程序服務，並得到相對應的輸出，已於使用者介面顯示結果。此處提供的遠端程序服務有三；一為使用者服務，包括註冊帳號、登入以及登出帳號的服務；次為使用者遊戲房服務，包括產生房間、顯示房名、進出房間的服務；最後為遊戲服務，包括遊戲地圖以及遊戲回合管理的服務。

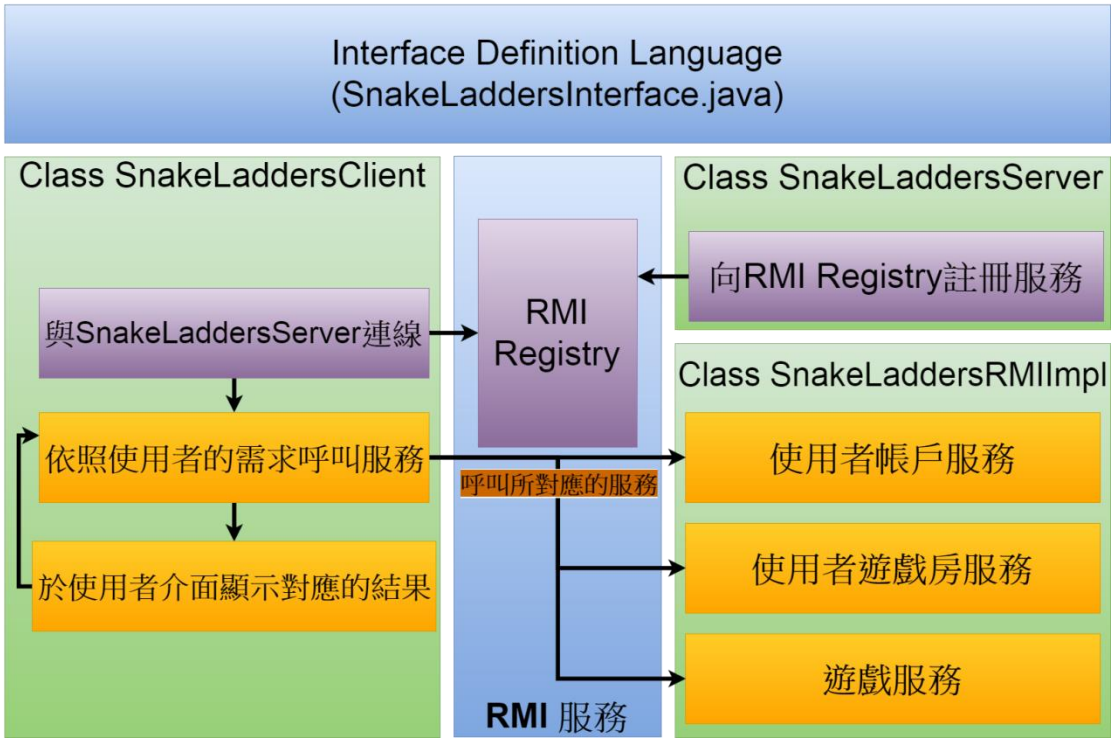

圖 3.2 Remote Method Invocation 應用程式架構圖

3.2 Server 端以及此遊戲系統的訊息訊列

Server 端所提供的程序服務，如上小節所述，主要有三，使用者帳戶服務、使用者遊戲房服務、遊戲服務。

使用者帳戶服務，有註冊帳號及帳號的登入登出。其系統服務的訊息序列下圖，註冊帳戶如下圖左，Client 端呼叫程序服務時，須輸入帳號(Account)及密碼(Password)，而 Server 執行完帳戶註冊程序後，假若成功會回傳會員的 ID (Member

Index)，假若失敗會回傳失敗原因的對應碼(Checking Number)；而登入服務程序的系統序列圖，與註冊的服務雷同，假若登入成功也是回傳會員的 ID，而失敗也是會傳失敗原因的對應碼。登出服務程序的系統序列圖則如下圖右所示，只需輸入 Member Index，而 Server 將會改變此帳戶的狀態為離線模式，但並不會回傳值給 Cient。

以上服務皆可以提供多人同時使用，因此當使用者註冊時，Server 端將會記錄於會員名單中，以確保不會有相同的帳戶名稱以及 Member Index 同時存在的問題。

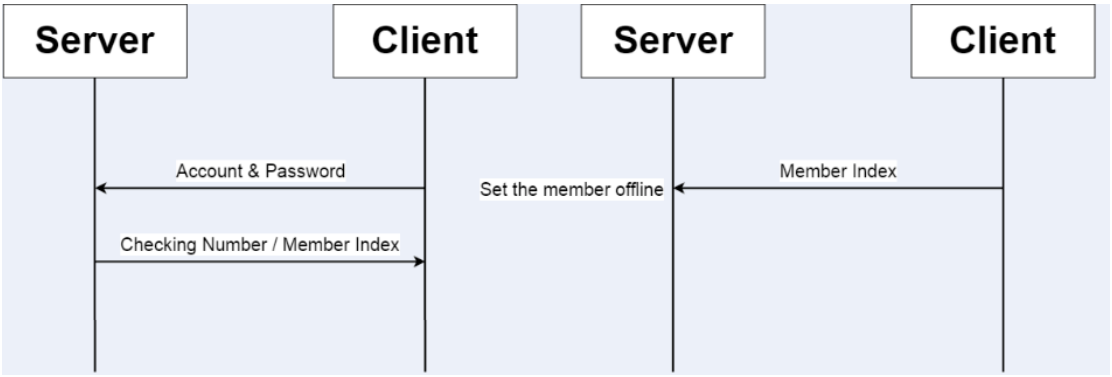

圖 3.3 使用者帳戶服務的系統訊息序列圖(左一:註冊、登入；右一: 登出)

使用者遊戲房服務，包括要求房間名單、產生房間以及進出房間。

要求房間名單的系統訊息序列如下圖所示，Client 可直接呼叫 Server 端的名單程序，而 Server 端收到呼叫後，會輸出房間名單並將之回傳。

以上服務皆可以提供多人同時使用，因此當使用者呼叫產生房間的程序時，會於執行程序時，記錄此新房間於房間名單中，以確保不會有同名房間的出現。而進出房間也會紀錄於使用者狀態欄位以及房間狀態，是否於房間中，因此也不會出現相同帳號同時出現於兩個房間中的狀況，以及人數超過房間人數上限的問題。

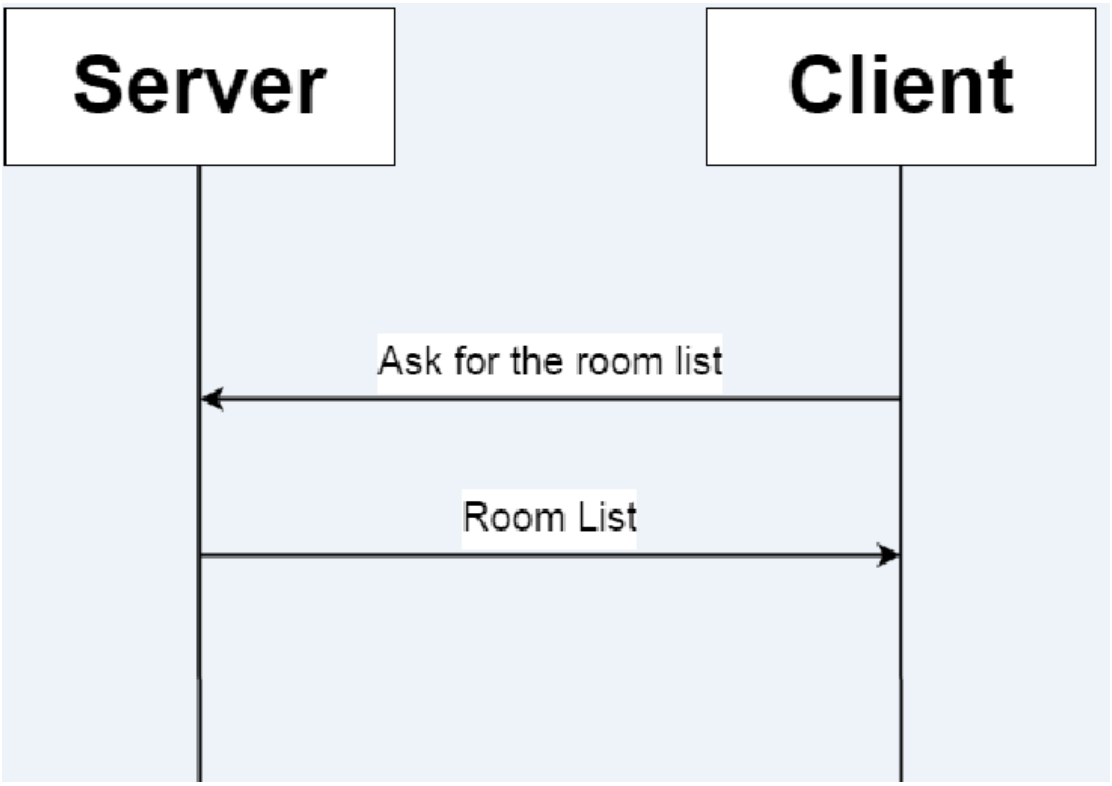

圖 3.4 遊戲房服務的系統訊息序列圖(使用者要求房間名單)

產生房間與進入房間的系統訊息序列圖如下圖左所示，皆是 Client 端於呼叫程序時輸入 Member Index 以及 Room Name，而 Server 端執行完程序後後會回傳相對應的 Checking Number 或 Room Index。假若產生房間或進入房間時失敗，會回傳相對應的 Checking Number，而假若成功，則皆會回應 Room Index。

離開房間的系統訊息序列圖如下圖右所示，Client 呼叫此程序時需輸入 Member Index 以及 Room Index，而 Server 執行完此程序時會輸出 Checking Number，已使使用者確認是否有成功離開房間。

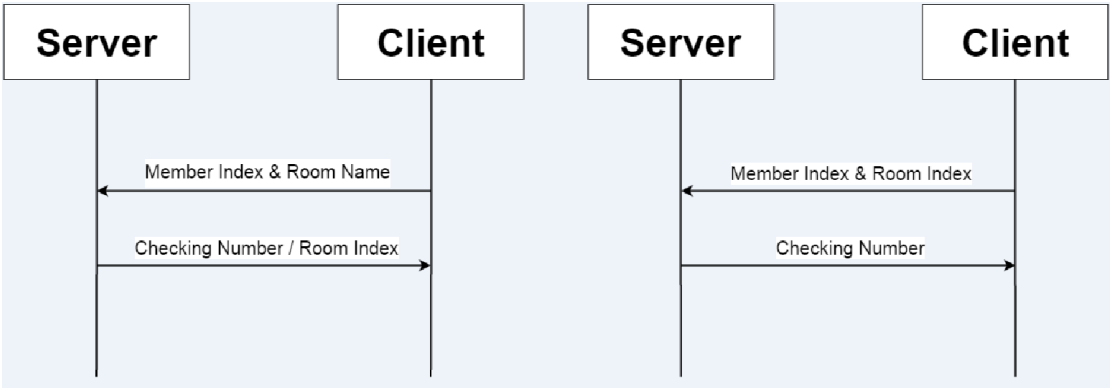

圖 3.5 遊戲房服務的系統訊息序列圖(左: 產生房間與進入房間；右: 離開房間)

遊戲服務，有著回傳棋盤物件資料、以及目前共同玩樂的玩家於棋盤上的位置、是否輪到己方回合的控制、甩骰子的程序等等。

Client 進入遊戲時會需要棋盤介面，而棋盤介面有棋盤物件以及玩家的棋子。

取得棋盤物件首先需知道目前棋盤物件有幾個，而此動作的系統序列圖如下圖左所示，輸入 Room Index 呼叫此程序，Server 執行完後即刻回傳物件的數量。依照所知道的數量執行迴圈，一個個的將物件的資訊回傳，物件資訊包括物件位置以及其種類，種類可能是梯子或者是蛇。而要得知物件資訊的系統訊息序列圖如下圖右所示，Client 需要輸入 Room Index 以及棋盤物件的 ID (Map Object Index)，而後 Server 會回傳棋盤的資訊。

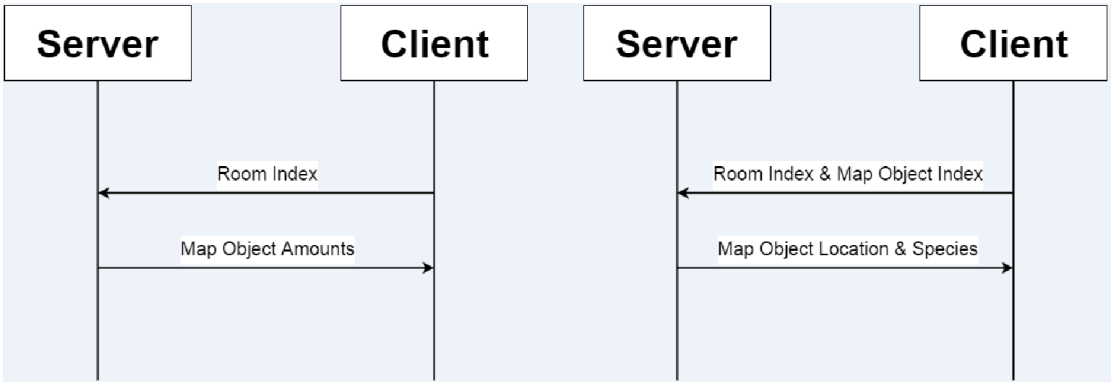

圖 3.5 遊戲服務的系統訊息序列圖(左: 取得棋盤物件的數量;右: 取得物件資訊)

取得玩家物件首先需知道目前棋盤物件有幾個，而此動作的系統序列圖如下圖左所示，輸入 Room Index 呼叫此程序，Server 執行完後即刻回傳玩家的數量。依照所知道的數量執行迴圈，一個個的將玩家的資訊回傳，玩家資訊包括玩家的位置以及其 Member Index。而要得知玩家資訊的系統訊息序列圖如下圖右所示，Client 需要輸入 Room Index 以及玩家順為的代碼(Player Index)，而後 Server 會回傳玩家的資訊。

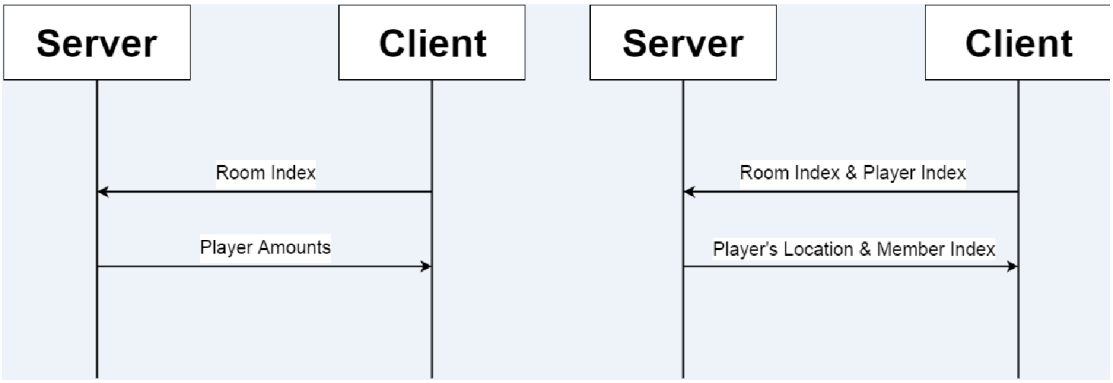

圖 3.6 遊戲服務的系統訊息序列圖(左: 取得棋盤玩家的數量;右: 取得玩家資訊)

當 Clients 已經獲得目前棋盤的物件以及玩家的資訊後，且玩家已經達到房間遊戲人數的條件後，即可開始遊戲。遊戲時，Clients 需先向 Server 詢問是否輪到己方回合，假若是則可甩骰子前行。

Clients 詢問是否為己方回合的系統訊息序列圖如下圖左所示，需輸入 Room Index 以 Member Index 而後呼叫程序，當 Server 執行完程序後會回傳 Checking Signal，Clients 可從此訊號知道是否輪到己方回合。假若是，則可呼叫甩骰子的程序；假若否，則不行呼叫。

Clients 呼叫甩骰子程序時如下圖右所示，需給予 Room Index 以及 Member Index，已使 Server 執行甩骰子程序時，可先行驗證是否為 Client 的回合。而驗證成功，可甩骰子，Server 會依照骰子點數使此 Client 前行，而後視其是否走上

特殊物件，如梯子或者蛇，而做出相對應的位移，並依照最後的位置更改地圖上相對的玩家的資訊。

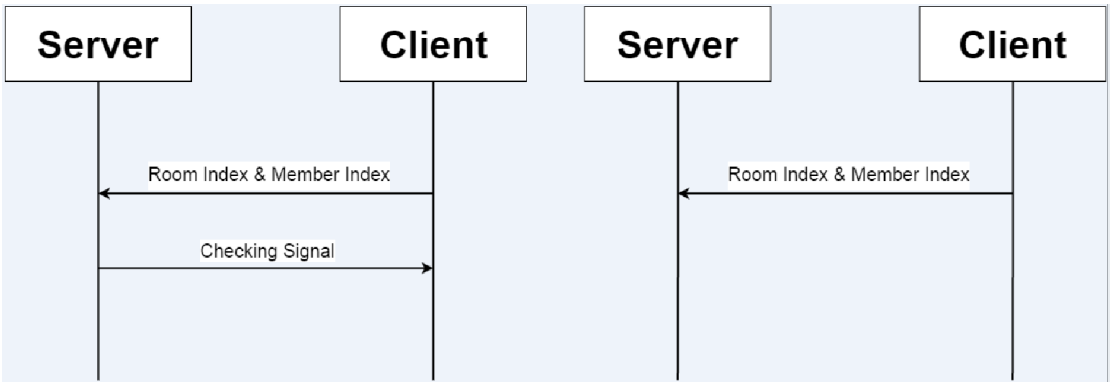

圖 3.7 遊戲服務的系統訊息序列圖(左: 是否為己方回合的確認；右: 甩骰子)

3.3 Client 端以及使用者流程圖

上小節敘述各個程序的系統訊息序列圖，而使用者於使用介面使用此些服務程序的流程圖如下圖所示，此些服務會依照下述三個使用者介面而有所不同，帳戶介面、選擇房間介面以及遊戲棋盤介面。

使用者帳戶服務、使用者遊戲房服務、遊戲服務，上個小節詳述的三種服務於下圖中依序各別為紅色、綠色、藍色的方格所提供的服務。

各個使用者介面，皆有使用者於介面可呼叫的服務程序；帳戶介面，有登入及註冊兩種服務;選擇房間介面,有產生房間、進入房間以及登出帳戶三種服務；遊戲棋盤介面，有離開房間、以及甩骰子兩種服務。由上述可知並非所有的服務程序，皆會由使用者於介面上直接呼叫，這是因為有些服務，直接於程式中於觸發事件產生時呼叫或每隔些許時間即會循環呼叫。

使用者操作流程如下圖所示。

啟動遊戲後，即進入如圖 3.9 的帳戶介面，而此時先於 Account 及 Password 的輸入框中輸入使用者想註冊或登入的帳號及密碼,後依照使用者需求按下按鈕，假若是登入，按下 Login，假若是註冊，按下 Reg。

而後登入或註冊成功，則進入如圖 3.10 的選擇房間介面，使用者可於右側的房間清單找尋想進入的房間，而後於輸入框中輸入房間後按下 In 的按鈕，假若輸入無誤，則進入房間；假若沒有想進入的房間，則可於輸入框中輸入沒有重複於房間清單中名稱的房間名，若輸入無誤，則進入房間。於此介面假若想要登出也可以直接按下右下角的 LogOut，則可登出成功。

進入房間後，印入眼簾的即是如圖 3.11 的遊戲棋盤介面，使用者此時需要等待進房人數抵達遊戲人數的標準，而後開始遊戲；當然假若使用者想要離開房間，也可以按下右下角的 Leave 按鈕，於開始遊戲前皆可以就此離開房間。當遊戲開始後，當使用者已經輪到己方回合，則可以按下 Dice 按鈕及可甩骰子，並且使用者的棋盤介面會有著相對應的位移，並且其他房內的使用者的棋盤介面也會有相對應的位移。

除了上述所述的使用者介面的按鈕，假若想快速離開房間且離線，則只要按下右上方的視窗關閉鈕即可。

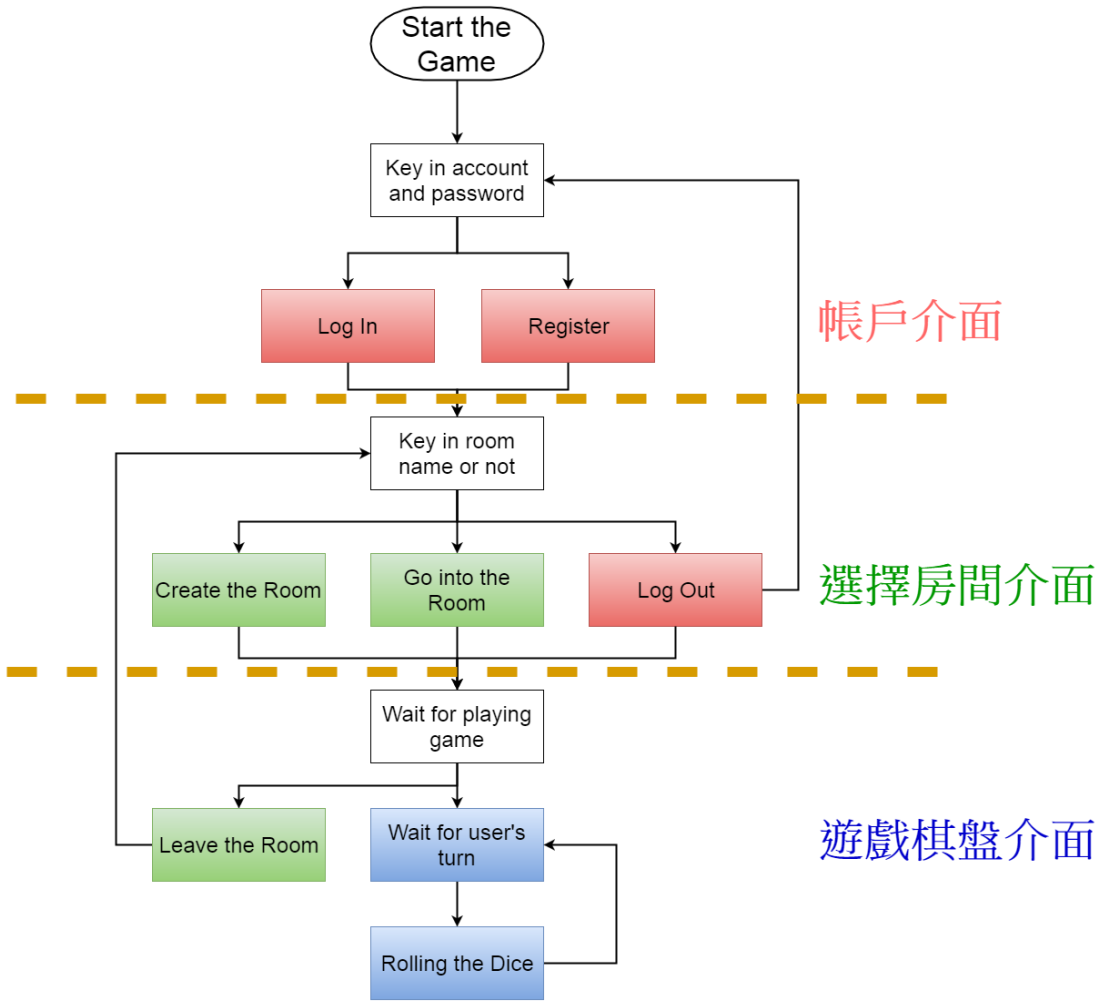
Start the Game
Key in account and password
Log In
Register
帳戶介面
Key in room name or not
Create the Room
Go into the Room
Log Out
選擇房間介面
Wait for playing game
Leave the Room
Wait for user's turn
Rolling the Dice
遊戲棋盤介面

圖 3.8 使用者操作流程


Snakes
&
Ladders
Account
Password
Login
Reg

圖 3.9 帳戶介面

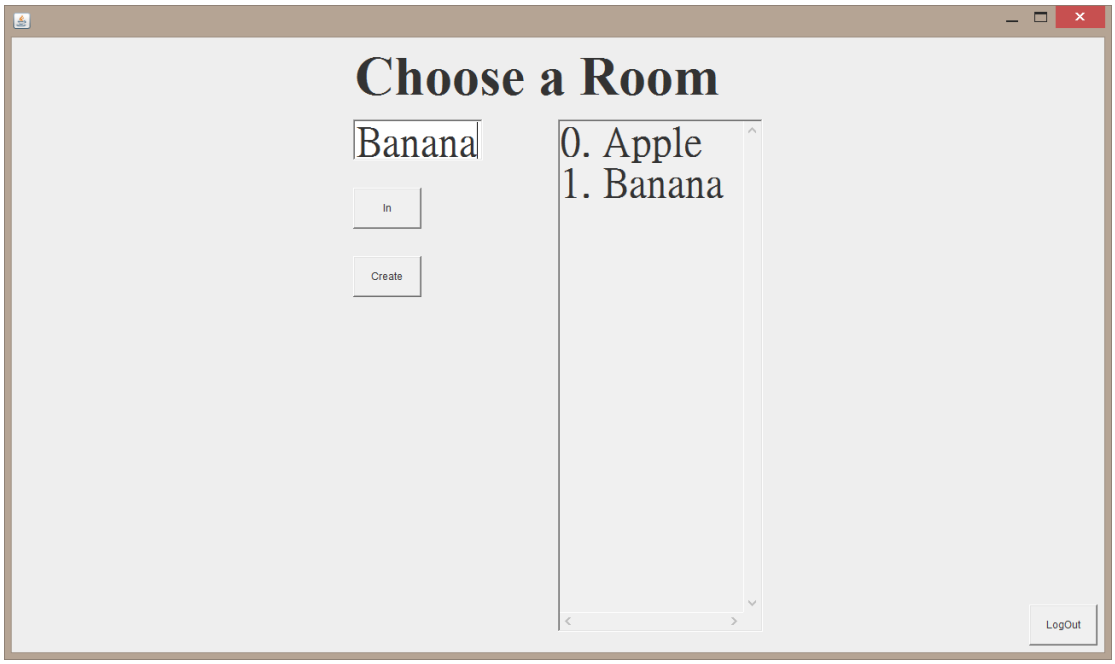

圖 3.10 選擇房間介面

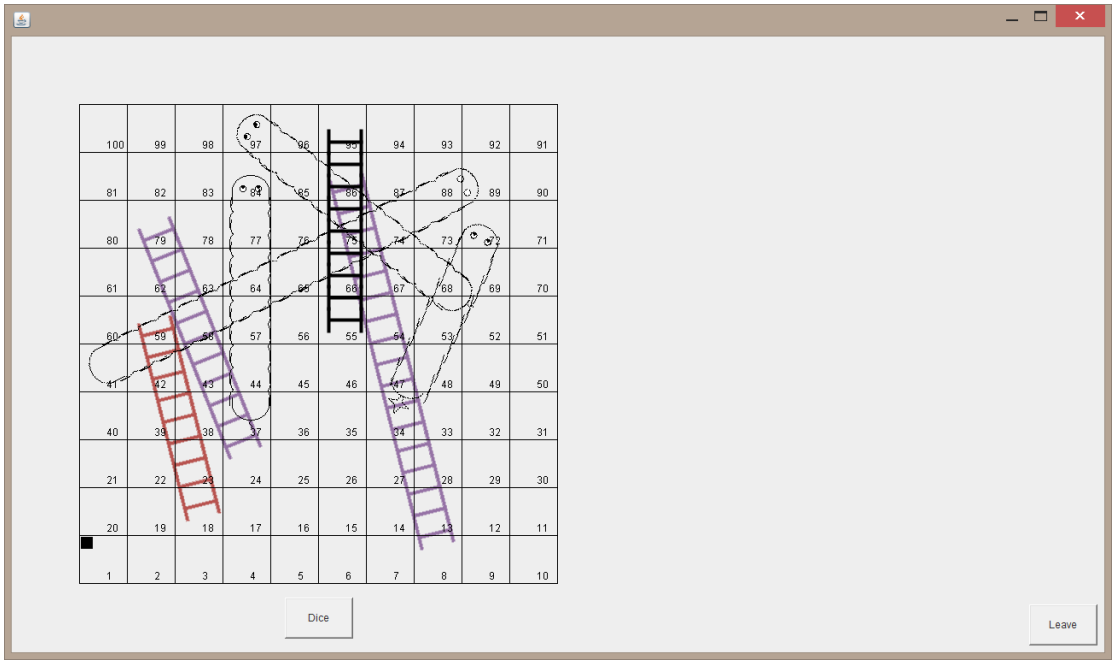

圖 3.11 遊戲棋盤介面

Client 端的圖形化使用者介面，實作方面主要是以 Java 的 Swing、Awt 搭配

繪製，此觀念可參考網路資源[2][3]。以 JFrame 為框架，而上述不同的三種使用者介面，則各以三個 JPanel 實作，當要切換介面時，不必置換 JFrame，只需置換 JFrame 中的 JPanel 為正確對應的介面。

此些介面時做用到了 Awt 中的許多物件，Label 可為文字表述，TextField 可為單行輸入框，TextField 搭配 Scroll Bar 可為多行多列的動態顯示框架，Button 可為按鈕，而 Canvas 則可依照遊戲設計者的需求設計所需要的圖樣。

使用者圖形化介面的實作中，最複雜之處莫過於遊戲棋盤介面中棋盤的設計。此部分以 Canvas 物件為底，先畫上許 100 個正方形而成棋盤貌，後依照物件及玩家棋子資訊畫上正確位置的物件及玩家棋子。繪畫物件，分為兩個步驟，先畫出完整的物件，而後置放在正確位置。

要畫出完整的物件，得先將物件的區塊相連而後為完整的物件；畫梯子物件時，首先得將圖 3.12 中的梯子區塊，相連為所需要的長度；畫蛇時，則略為不同，首先得先畫蛇頭，後畫所需要的蛇身長度，而後接上蛇尾。依照上述步驟則可德所需長度的物件。

圖 3.12 梯子的區塊

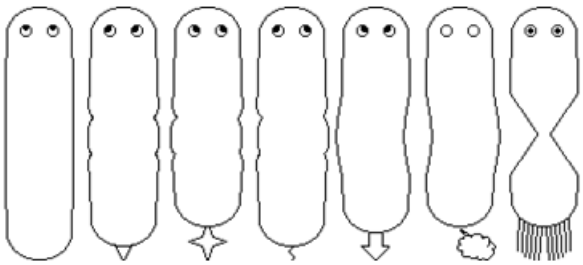

圖 3.13 蛇的示意圖

要將物件置放在正確的位置，首先需知道畫圖片時，於此是控制左上角的點為起始位置，而後依照使用者需求轉角度而得最終顯示的圖。物件起始位置的設

定需要如下圖所示，假若左斜的圖的左上角點須放在紅點上，正直的圖的左上角點須放在綠點上，右斜的圖的左上角點須放在藍點上，才不至於將圖轉角度後超出原先設置的區塊。轉角度只需呼叫三角函數推敲即可。

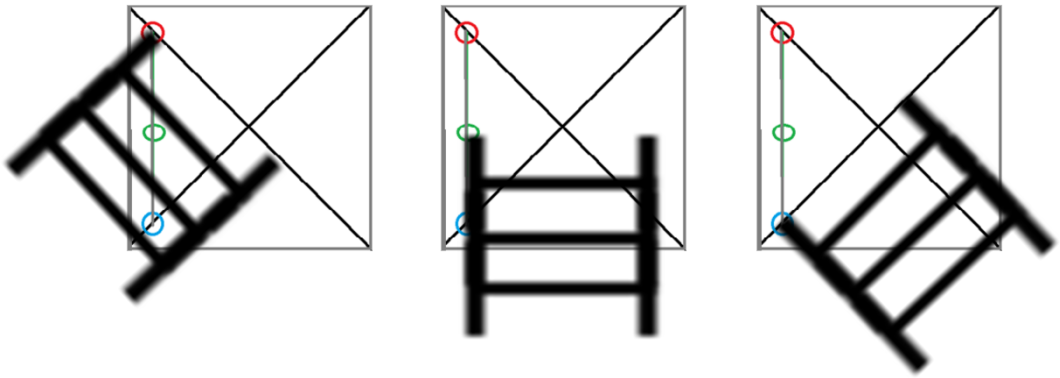

圖 3.13 物件置放位置的示意圖

第四章 實驗結果

此網路遊戲是設計給多人同時遊樂的，因此此專題做了多人使用的測試；此外也實驗了使用者遊戲時，是否如我方所設計，按下圖形化介面的按鈕而有相對應的動作。

4.1 多人使用的測試

多人使用的部分，包括了多人同時註冊以及多人同時進房、多人同時遊玩，而此三種狀況於此專題接是利用 RMI 程序呼叫的特性，使 Server 開啟多個執行序，並且於共用同筆資料時，搭配 Java 中 synchronized 的屬性的函數，而於不同執行序呼叫同筆資料時，給予某一執行序使用權，並且 lock 住，使得其他使用者須待至其使用完。因為皆是使用同種方法處理之，所以此專題指測試多人註冊的部分。

測試時，因為圖形化介面不易以模擬多人同時使用，因此另外寫了專門註冊帳號的 Client 端程式，如此即可批次執行。此程式可依照輸入參數決定要註冊幾組帳號以及所註冊的帳密內容，而此實驗，模擬三人同時註冊各 2000 組帳戶，因此最終假若皆註冊成功，則可得 6000 組帳號。(此實驗確保障密不會重複)

而註冊成功時會回傳其對應的 Member Index，而 Member Index 為 Member List 中順位，順位從 0 開始，因此假若皆註冊成功，則必有一 Client 端得到 5999 的 Member Index。如下圖所示，三個視窗為 Client 端的視窗，顯示得為 Member Index，可見中間的視窗最後收到的 Member Index 為 5999，因此實驗成功。

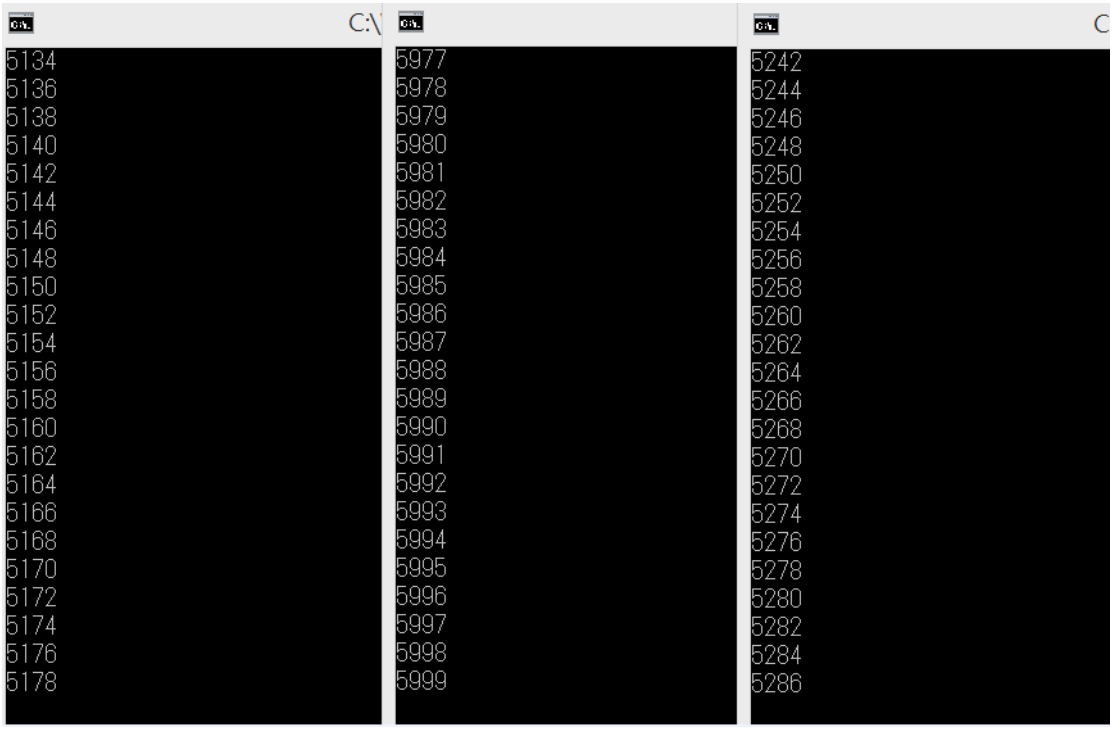

圖 4.1 多人同時註冊的 Client 端顯示

4.2 遊戲狀況的實驗

圖形化介面的實驗，下列為我所實驗過的使用者動作:

表 4. 1 工作分配表

執行動作	應有反應	實驗結果
開啟遊戲→輸入未曾輸入的帳密→按下 Reg	進入房間選擇介面	成功
開啟遊戲→輸入曾輸入的帳密→按下 Reg	毫無反應	成功
開啟遊戲→輸入未曾輸入的帳密→按下 Login	毫無反應	成功
開啟遊戲→輸入曾輸入的帳密→按下 Login	進入房間選擇介面	成功

進入房間介面	顯示目前房間清單於房間清單框	成功
進入房間介面→輸入清單沒有的名稱→按下 Create	進入遊戲棋盤介面	成功
進入房間介面→輸入清單有的名稱→按下 Create	毫無反應	成功
進入房間介面→輸入清單沒有的名稱→按下 In	毫無反應	成功
進入房間介面→輸入清單有的名稱→按下 In	進入遊戲棋盤介面	成功
進入房間介面→按下 LogOut	進入帳戶介面	成功
進入遊戲棋盤介面	視窗左側顯示棋盤及其物件及目前玩家	成功
進入遊戲棋盤介面(棋盤顯示人數尚未滿四人)→按下 Dice	毫無反應	成功
進入遊戲棋盤介面(棋盤顯示人數滿四人，且為己方回合)→按下 Dice	己方棋子有位移	成功
進入遊戲棋盤介面(棋盤顯示人數滿四人，且非己方回合)→按下 Dice	己方棋子無反應，但他方棋子會位移	成功

進入遊戲棋盤介面(棋盤顯示人數未滿四人)→按下 Leave	進入房間選擇介面	成功
進入遊戲棋盤介面(棋盤顯示人數滿四人)→按下 Leave	不會進入房間選擇介面	成功
不論於何種介面按下視窗的右上角的 X	Client 端關閉視窗，且 Server 端紀錄玩家圍離線離房。	成功

第五章 結論

此專題已經達成以 Java 實作，並且以 Remote Method Invocation (RMI)做為網路通訊的介面。且可允許多人註冊及遊樂，並且將此遊戲設計為房間制度管理，且也完善規畫了回合制遊戲的輪循。且最終的棋盤也可動態置放棋盤與蛇。

完成了多項目標，但仍有多處不夠完整。

功能缺陷，包括刪除帳號、刪除房間等功能。

圖形化介面也可以設計得更美化。

通訊方面，因為使用 RMI 的關係，Server 為被動的，因此有些程序呼叫，需要 Client 端以輪循的方法，規律地呼叫 Server 端給予資訊，譬如說，要求房間清單的呼叫程序需要每隔一小段時間就呼叫 Server 執行其程序。而輪循的方法相當耗費 CPU 資源，因此未來如需改善，應更改通訊方法，使得多數功能可以觸發事件的樣貌執行。

參考文獻

[1] RMI: https://docs.oracle.com/javase/tutorial/rmi/

[2] Java Jframe Class:

https://docs.oracle.com/javase/7/docs/api/javax/swing/JFrame.html

[3] Java Canvas Class:

https://docs.oracle.com/javase/7/docs/api/java/awt/Canvas.html