PC網路聊天室(TCP/IP)

## 第一章 緒論

## 1.1研究背景與動機

隨著資訊科技的飛速發展與網際網路的普及，即時通訊（Instant Messaging, IM）已成為現代社會不可或缺的溝通方式。從個人社交、團隊協作到客戶服務，即時通訊軟體憑藉其高效、便捷的特性，深刻地改變了人們的交流習慣和工作模式。PC平台的網路聊天室作為即時通訊的一種早期且經典的形式，其底層的網路通訊協定和程式設計邏輯，對於理解更複雜的分散式系統和網路應用具有重要的學習價值。

本研究動機是對於對網路通訊技術， TCP/IP協定在實際應用中的運作方式。TCP/IP作為網際網路的基石，支援了絕大多數的網路資料傳輸。透過親手設計並實作一個基於C#語言和Windows Forms介面的PC網路聊天室，能夠更深入理解客戶端/伺服器（Client/Server）架構、多執行緒處理等核心技術，同時將理論知識與實際操作相結合，提升軟體開發的實踐能力。

C#語言以其強大的.NET框架支援、物件導向特性以及在Windows平台開發上的便利性，成為了實現此類桌面應用程式的理想選擇。Windows Forms則提供了一套成熟的圖形化使用者介面（GUI）開發工具，有助於快速建構互動友善的聊天室介面。因此，本專題旨在透過C#和Windows Forms技術，結合TCP/IP協定，建構一個功能性的PC網路聊天室，藉此深化對相關技術的理解與應用。

## 1.2研究目的

本研究旨在設計與實作一個基於TCP/IP協定的PC網路聊天室。

具體研究目的如下：

1. 掌握TCP/IP通訊原理：深入理解TCP協定的連線導向、可靠傳輸特性，進行網路資料的傳送與接收。
2. 實現客戶端/伺服器架構：設計並開發聊天室的伺服器端應用程式，負責管理客戶端連線、訊息轉發與使用者狀態維護；同時開發客戶端應用程式，提供使用者登入、訊息收發與顯示等功能。
3. 應用多執行緒技術：在伺服器端採用多執行緒或非同步處理機制，以有效處理多個客戶端的併發連線與請求，確保聊天室的即時性與穩定性。
4. 開發圖形化使用者介面：利用C# Windows Forms技術，為客戶端設計一個直觀、易用的圖形化使用者介面，提升使用者體驗。
5. 實現基本聊天功能：包括但不限於使用者登入/登出、使用者列表顯示等核心聊天功能。

## 1.3問題陳述

在開發PC網路聊天室的過程中，預期將面臨以下主要技術問題與挑戰：

1. 可靠的連線建立與維護：如何在客戶端與伺服器之間建立穩定可靠的TCP連線，並在網路不穩定或意外斷線時進行有效處理？
2. 併發使用者處理：伺服器端如何高效地處理來自多個客戶端的同時連線請求和訊息傳輸，避免阻塞和資源競爭？
3. 訊息同步與即時性：如何確保訊息能夠即時、準確地從傳送端傳遞到所有指定的接收端，並在客戶端介面即時更新？
4. 資料格式與序列化：如何定義客戶端與伺服器之間交換訊息的資料格式，並進行有效的序列化與反序列化，以確保資料的正確解析？
5. 使用者介面回應與互動：如何在進行網路通訊的同時，保持客戶端GUI的流暢回應，避免因網路操作阻塞UI執行緒？
6. 錯誤處理與異常管理：如何設計健壯的錯誤處理機制，以應對網路錯誤、資料格式錯誤、使用者操作不當等各種預期及非預期的異常情況？

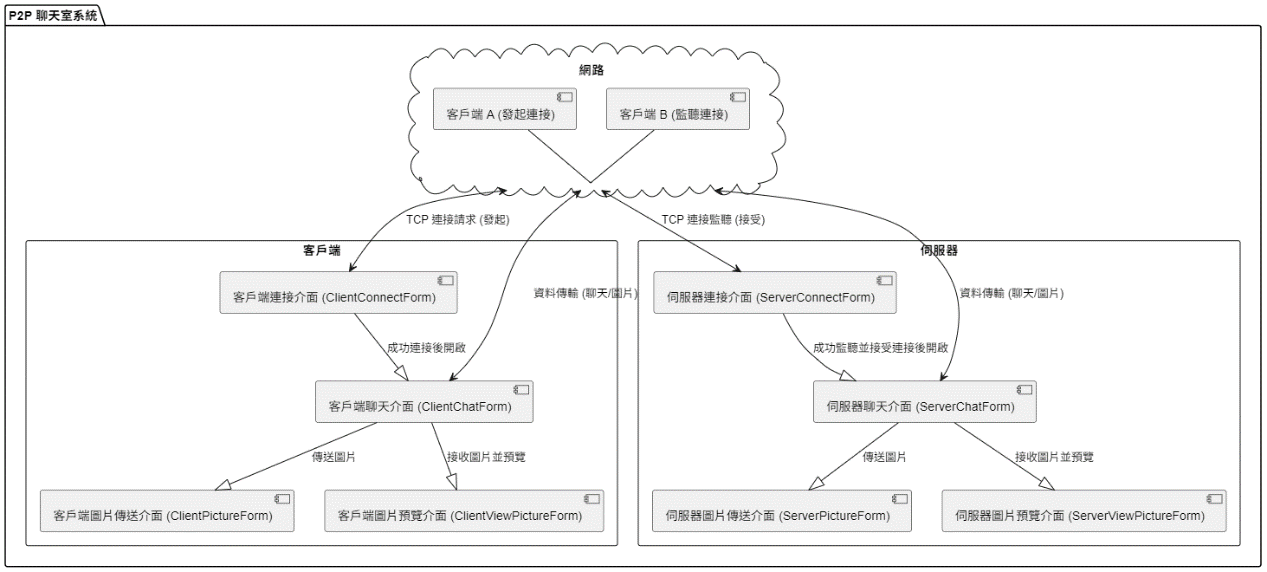
## 1.4預期貢獻

本研究的完成，預期將達成以下幾方面的貢獻：

1. 技術實踐能力的提升：對於研究者本身，將顯著提升在C#程式設計、Windows Forms介面開發、TCP/IP網路編程以及多執行緒應用等方面的綜合實踐能力。
2. 一個功能性的聊天室應用程式：產出一個可在PC環境下運行的、具備基本聊天功能的網路聊天室原型系統，可作為學習和演示TCP/IP通訊的實例。
3. 網路編程知識的深化：透過專案的完整開發週期，加深對客戶端/伺服器模型、資料封包處理等網路核心概念的理解。
4. 解決方案的探索與驗證：針對在開發過程中遇到的技術挑戰（如併發處理、UI回應等），所採用的解決方案可為類似專案提供參考。

# 第二章 系統設計

## 2.1系統架構圖



### 2.1.1架構概述

網路層 (Network): 負責處理底層的 TCP/IP 通訊，實現點對點的數據交換。

A客戶端(Client):

* ClientConnectForm: 用於讓客戶端輸入目標 IP 和端口，發起連接請求。
* ClientChatForm: 核心聊天介面，處理文字訊息的收發、表情符號的插入、圖片傳送與接收的介面操作。
* ClientPictureForm: 允許使用者選擇本地圖片或從 URL 載入圖片並傳送。
* ClientViewPictureForm: 接收圖片後顯示，並提供縮放、平移和儲存功能。

B伺服器(Server):

* ServerConnectForm: 用於讓伺服器設定監聽端口，等待客戶端連接。
* ServerChatForm: 核心聊天介面，功能與 ClientChatForm 類似，處理文字訊息的收發、表情符號的插入、圖片傳送與接收的介面操作。
* ServerPictureForm: 允許使用者選擇本地圖片或從 URL 載入圖片並傳送。
* ServerViewPictureForm: 接收圖片後顯示，並提供縮放、平移和儲存功能。

兩個應用程式（客戶端和伺服器）在邏輯上是獨立的，但它們都可以扮演P2P通訊中的任何一方角色，即一個應用程式可以作為伺服器等待連接，另一個作為客戶端發起連接。一旦連接建立，雙方就可以交換訊息和圖片。

## 2.2功能模組說明

### 2.2.1本系統主要分為以下幾個功能模組：

連接管理模組 (Connection Management Module):

* 功能: 負責建立和管理點對點的 TCP 連接。
* 子模組:
  + 伺服器端連接 (ServerConnectForm): 啟動 TCP 監聽器，等待客戶端連接請求，並處理連接的接受、超時判斷及狀態顯示。
  + 客戶端連接 (ClientConnectForm): 發起 TCP 連接請求至指定的伺服器 IP 和端口，並處理連接結果、超時判斷及狀態顯示。
* 相關檔案: ClientConnectForm.cs, ServerConnectForm.cs。

### 2.2.2聊天通訊模組 (Chat Communication Module):

* 功能: 實現即時文字訊息的收發和顯示。
* 子模組:
  + 訊息傳送: 將使用者輸入的文字訊息封裝並透過網路串流傳送給對方。
  + 訊息接收與顯示: 監聽網路串流，接收來自對方的文字訊息並解析，顯示在聊天介面中。
  + 表情符號功能: 提供表情符號選單，使用者可點擊插入表情符號至訊息。
* 相關檔案: ClientChatForm.cs, ServerChatForm.cs。

### 2.2.3圖片傳輸模組 (Image Transfer Module):

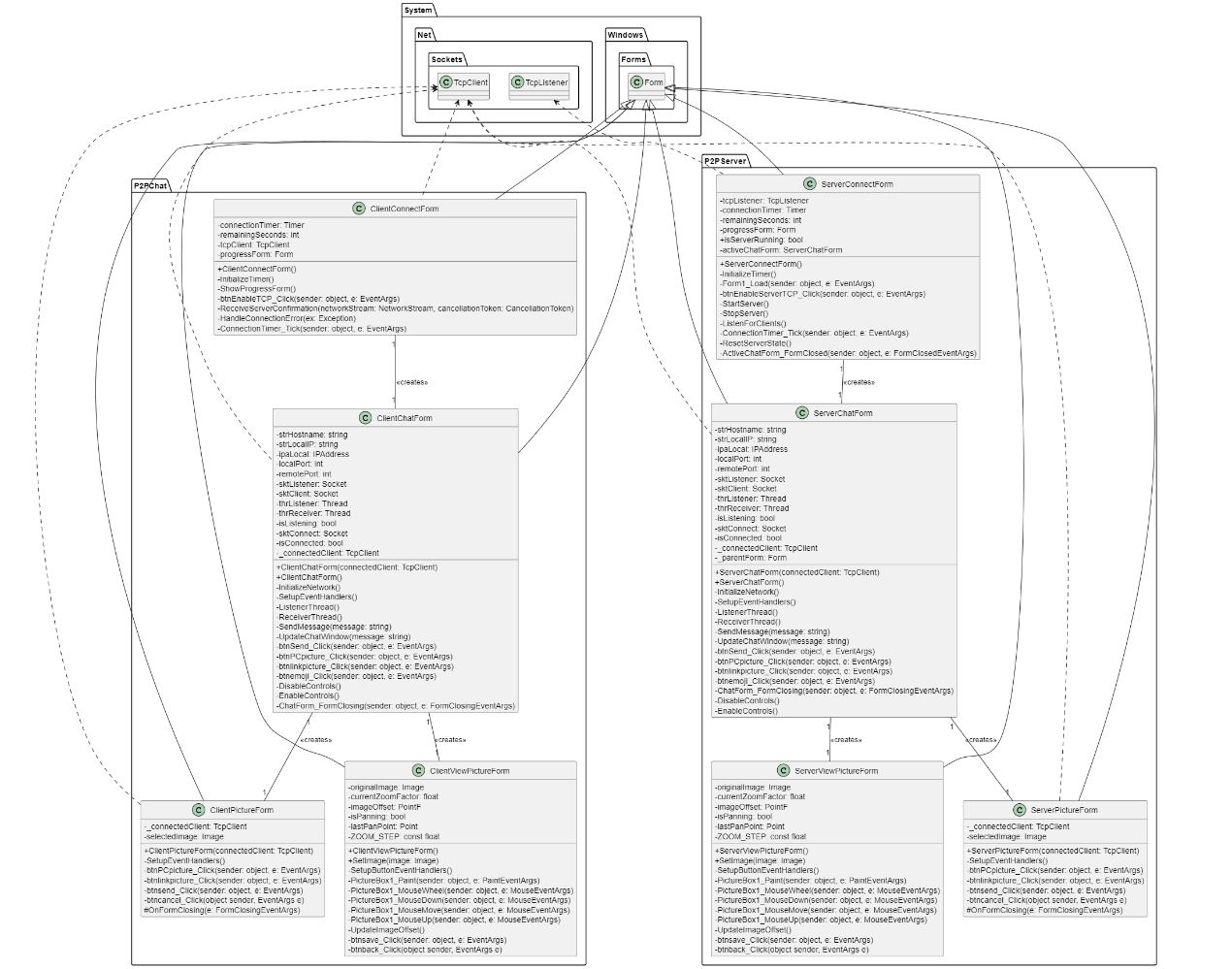
* 功能: 支援圖片檔案的傳送與接收。
* 子模組:
  + 圖片選擇與載入: 允許使用者從本地電腦選擇圖片檔案，或輸入圖片 URL 載入圖片。
  + 圖片編碼與傳送: 將選定的圖片轉換為 Base64 字串或位元組陣列，透過網路串流傳送。
  + 圖片接收與解碼: 接收圖片數據，將其解碼為圖片物件。
  + 圖片預覽與儲存 (ClientViewPictureForm / ServerViewPictureForm): 提供獨立的圖片預覽視窗，支援圖片的縮放、平移功能，並可將接收到的圖片儲存至本地。
* 相關檔案: ClientPictureForm.cs, ServerPictureForm.cs, ClientViewPictureForm.cs, ServerViewPictureForm.cs。

### 2.2.4使用者介面模組 (User Interface Module):

* 功能: 提供直觀友好的圖形使用者介面 (GUI)，方便使用者操作。
* 子模組:
  + 視窗管理: 控制不同功能視窗的顯示、隱藏和切換。
  + 控制項交互: 處理按鈕點擊、文字輸入、滑鼠事件等使用者交互。
  + 狀態顯示: 顯示連接狀態、聊天記錄、錯誤訊息等。
  + 相關檔案: 所有 .cs 表單檔案。

## 2.3類別圖、循序圖等 UML 圖表

### 2.3.1類別圖:

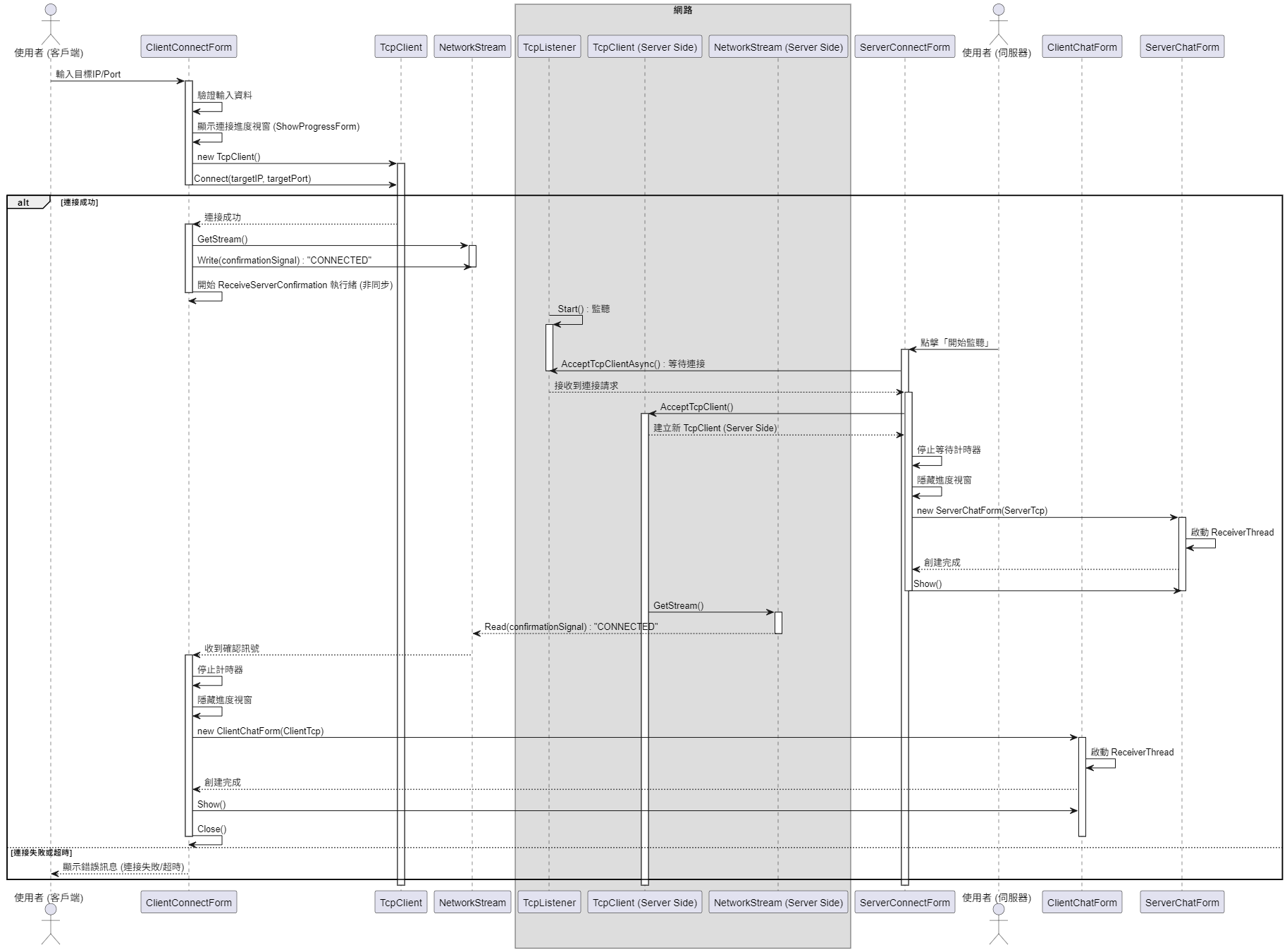


類別圖說明:

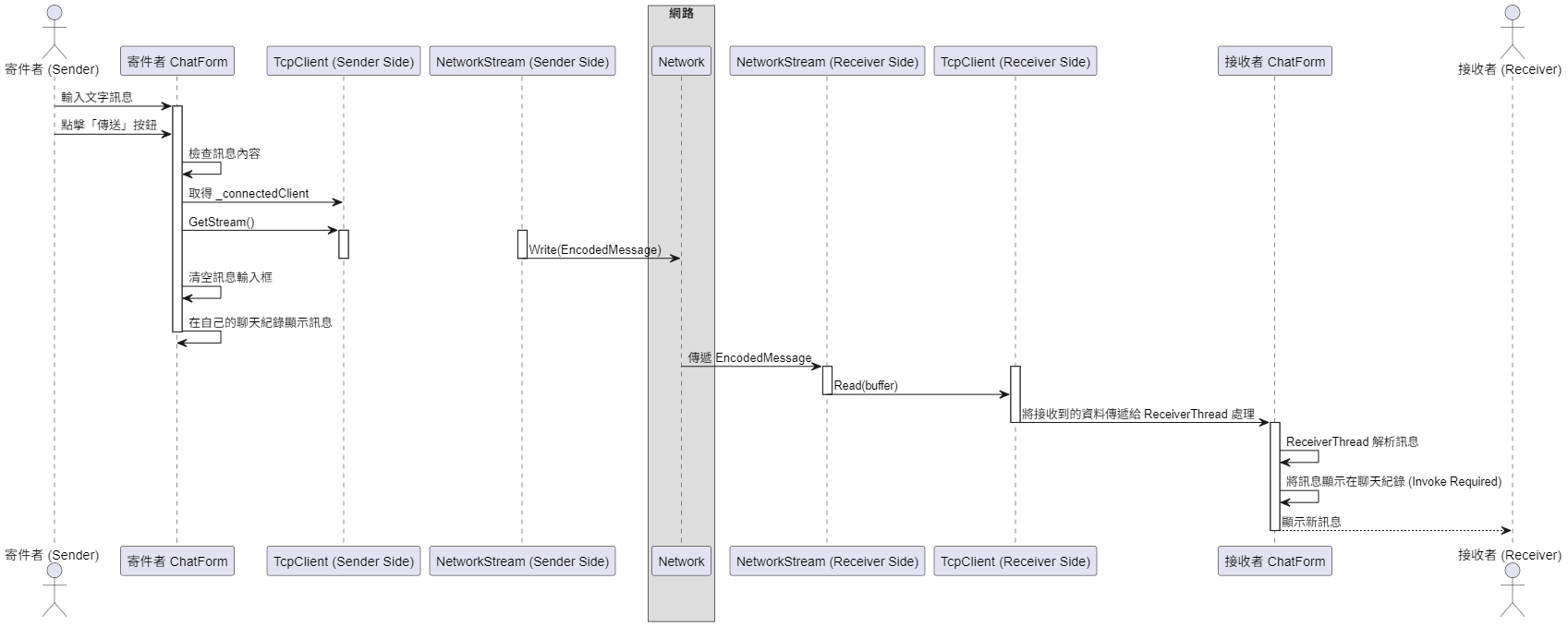
* P2PChat 命名空間 (客戶端): 包含 ClientConnectForm, ClientChatForm, ClientPictureForm, ClientViewPictureForm 等類別，分別處理客戶端的連接、聊天、圖片傳送和圖片預覽功能。
* P2PServer 命名空間 (伺服器): 包含 ServerConnectForm, ServerChatForm, ServerPictureForm, ServerViewPictureForm 等類別，分別處理伺服器的連接、聊天、圖片傳送和圖片預覽功能。
* 繼承關係: 所有表單類別都繼承自 System.Windows.Forms.Form。
* 關聯關係 (Creates): 連接表單在成功連接後會創建聊天表單；聊天表單可以創建圖片傳送表單和圖片預覽表單。
* 依賴關係: 這些類別依賴於 System.Net.Sockets.TcpClient 和 System.Net.Sockets.TcpListener 來實現網路通訊。

### 2.3.2循序圖:

建立連接循序圖 (客戶端發起)



傳送文字訊息循序圖



# 第三章 系統實現

## 3.1開發環境（Visual Studio 版本、.NET 版本等）

Microsoft Visual Studio Community 2022 (64 位元) - Current版本 17.12.4

Microsoft .NET Framework版本4.8.09037

## 3.2核心功能代碼解析

## .3.3關鍵技術實現說明

### 3.3.1界面設計與使用者操作流程

# 第四章 測試與驗證

## 4.1單元測試案例

## 4.2系統整合測試

## 4.3效能評估（如有）

# 第五章 結論與未來工作

## 5.1研究成果總結

## 5.2專題限制

## 5.3未來改進方向

# 附錄. 參考文獻

引用書籍、論文、網路資源

[**.NET Framework API 參考 version 4.8**](https://learn.microsoft.com/zh-tw/dotnet/api/?view=netframework-4.8&preserve-view=true)

### 撰寫建議

- \*\*字數分配\*\*：假設報告約 1000-2500 字，每

個大綱約佔 200-500 字，根據需求調整。