ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

Tel. (84-511) 3736949, Fax. (84-511) 3842771

Website: itf.dut.udn.vn, E-mail: cntt@dut.udn.vn

❧•❧



**BÁO CÁO ĐỒ ÁN**

**CƠ SỞ NGÀNH MẠNG**

Giáo viên hướng dẫn: PGS.TS. Huỳnh Công Pháp

Sinh viên thực hiện : Nguyễn Thị Phương Dung

Mã sinh viên : 102130062

Nhóm : 13.11

*Đà Nẵng, tháng 5/2017*

**LỜI MỞ ĐẦU**

Ngày nay, với sự phát triển của khoa học kỹ thuật, công nghệ thông tin ngày càng biết đến nhiều hơn. Đặc biệt là vấn đề kết nối các máy trên mạng đề truyền và nhận thông tin là một nhu cầu thiết yếu của mỗi cá nhân.

“Lập trình mạng” là môn học giúp nắm rõ cơ chế hoạt động cũng như cách thức để kết nối các máy tính lại với nhau, đồng thời thực hiện việc trao đổi dữ liệu qua lại giữa nhiều máy. Dựa vào các kiến thức đã học, sự đa dạng của các dịch vụ mạng với nhiều tính năng và đòi hỏi ngày càng cao. Từ ý tưởng xây dựng một chương trình có khả năng tương tác với máng tính từ xa thông qua hệ thống mạng, chương trình Quản lý các tiến trình từ xa được xây dưng, dựa theo mô hình Client/Server.

Do kiến thức và kinh nghiệm của bản thân em còn hạn chế nên việc thực hiện không thể tránh khỏi những thiếu sót. Rất mong được sự đóng góp ý kiến của thầy cô để báo cáo của em được hoàn thiện hơn.

Cuối cùng, em xin chân thành cảm ơn các thầy cô trong khoa Công nghệ thông tin trường đại học Bách khoa Đà Nẵng, đặc biệt là thầy thầy **Huỳnh Công Pháp** -, người đã trực tiếp hướng dẫn và tận tình giúp đỡ, chỉ bảo em trong xuốt thời gian thực hiện đề tài.

Em xin chân thành cảm ơn.

Sinh viên : Nguyễn Thị Phương Dung

MỤC LỤC

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 4](#_30j0zll)

[**1.**](#_1fob9te) **Tìm hiểu về Hệ Điều Hành** 4

[**1.1**](#_3znysh7) **Định nghĩa Hệ điều hành** 4

[**1.2**](#_2et92p0) **Chức năng Hệ điều hành** 4

[**1.3**](#_tyjcwt) **Tìm hiểu về tiến trình** 4

[**2**](#_3dy6vkm) **Lập trình mạng** 7

[**2.1**](#_1t3h5sf) **Tổng quan về ngôn ngữ lập trình C#** 7

[**2.2**](#_4d34og8) **. Giới thiệu về lập trình Socket** 8

[CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 13](#_2s8eyo1)

[**1.**](#_17dp8vu) **Phân tích yêu cầu** 13

[**1.1.**](#_3rdcrjn) **Đề tài** 13

[**1.2.**](#_26in1rg) **Yêu cầu** 13

[**1.3.**](#_lnxbz9) **Triển khai** 13

[**2.**](#_35nkun2) **Phân tích hệ thống** 13

[**2.1.**](#_1ksv4uv) **Sơ đồ Use case** 13

[CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 16](#_44sinio)

[**1.**](#_2jxsxqh) **Môi trường làm việc** 16

[**2.**](#_z337ya) **Triển khai** 16

[**2.1**](#_3j2qqm3) **Server** 16

[**2.2**](#_1y810tw) **Client** 16

[CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 17](#_4i7ojhp)

[**1.**](#_2xcytpi) **Kết luận** 17

[**2.**](#_1ci93xb) **Hướng phát triển** 17

**ĐỀ TÀI: XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ TIẾN TRÌNH TỪ XA**

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1. **Tìm hiểu về Hệ Điều Hành**
   1. **Định nghĩa Hệ điều hành**

Hệ điều hành (HĐH) là một phần quan trọng của mỗi hệ thống thông tin. Mỗi hệ thống thông tin gồm 4 thành phần quan trọng: Phần cứng, hệ điều hành, chương trình ứng dụng và người sử dụng.

Đối với người sử dụng: HĐH là tập hợp các chương trình, phục vụ khai thác hệ thống tính toán một cách dễ dàng, thuận tiện.

Người sử dụng khi thực hiện một chương trình nào đó trên máy tính thì chỉ quan tâm đến việc hệ thống có đáp ứng được nhu cầu của hộ hay không? Họ không quan tâm đến việc hệ điều hành làm như thế nào, nhằm mục đích gì, có cấu trúc như thế nào?

Đối với người làm công tác quản lý: HĐH là một tập các chương trình phục vụ quản lý chặt chẽ và sử dụng tối ưu các tài nguyên của hệ thống.

* 1. **Chức năng Hệ điều hành**

Hệ điều hành là một chương trình đóng vai trò trung gian trong việc giao tiếp giữa người sử dụng và phần cứng của máy tính. Mục tiêu của hệ điều hành là cung cấp một môi trường cho phép người sử dụng phát triển và thực hiện các ứng dụng của họ một cách dễ dàng và hiệu quả. Theo nguyên tắc, một hệ điều hành cần thoả mãn chức năng Quản lý, chia sẻ tài nguyên: Tài nguyên hệ thống, đặc biệt các tài nguyên phần cứng như CPU, bộ nhớ, thiết bị ngoại vi,.. hệ điều hành cần có các cơ chế và chiến lược thích hợp để quản lý việc phân phối tài nguyên.

* 1. **Tìm hiểu về tiến trình**
     1. **Tiến trình là gì?**

Tất cả các máy tính hiện đại đều có thể thực hiện nhiều việc cùng một lúc. Trong khi thực hiện chương trình của người sử dụng, máy tính có thể đọc dữ liệu từ đĩa và đưa ra màn hình hoặc máy in. Trong môi trường đa chương trình (multiprogramming system), một CPU có thể chuyển từ chương trình này sang chương trình khác, thực hiện mỗi chương trình trong khoảng 1% hoặc 1/10 mili giây. Nếu nói chính xác, thì tại một thời điểm, CPU chỉ thực hiện được một chương trình. Nhưng nếu xét trong khoảng thời gian phần trăm giây thì CPU có thể thực hiện nhiều công việc.

Để hỗ trợ hoạt động đa nhiệm, hệ thống máy tính cần phải có khả năng thực hiện nhiều tác vụ xử lý đồng thời nhưng việc điều khiển hoạt động song hành ở cấp độ phần cứng là rất khó khăn. Vì vậy, các nhà thiết kế hệ điều hành đề xuất một mô hình song hành giả lập bằng cách chuyển đổi bộ xử lý qua lại giữa các chương trình để duy trì hoạt động của nhiều chương trình phải cùng một thời điểm. Trong mô hình này các chương trình của hệ thống được tổ chức thành các tiến trình (process).

Như vậy có thể coi tiến trình là một chương trình đang xử lý, nó sử dụng một con trỏ lệnh, tập các thanh ghi và các biến. Để hoàn thành nhiệm vụ của mình, các tiến trình còn có thể yêu cầu một số tài nguyên hệ thống như: CPU, bộ nhớ và các thiết bị.

Chúng ta cần phân biệt rõ tiến trình và chương trình. Chương trình là một thực thể thụ động chưa các chỉ thị điều khiển máy tính thi hành một tác vụ cụ thể nào đó. Khi thực hiện các chỉ thị này, chương trình được chuyển thành các tiến trình là một thực thể hoạt động, với con trỏ lệnh xác định kèm thêm tài nguyên phục vụ cho hoạt động.

Vậy tóm lại, tiến trình là một bộ phận của một chương trình đang thực hiện, đơn vị thực hiện tiến trình là processer.

* + 1. **Các trạng thái của tiến trình**

Từ khi đưa vào hệ thống cho đến khi kết thúc tiến trình tồn tại ở các trạng thái khác nhau.Trạng thái của tiến trình tại một thời điểm được xác định bởi hoạt động hiện thời của tiến trình tại thời điểm đó. Trong quá trình sống, một tiến trình thay đổi trạng thái do nhiều nguyên nhân như : phải chờ một sự kiện nào đó xảy ra, hay đợi một thao tác nhập/xuất hoàn tất, buộc phải dừng hoạt động do đã hết thời gian xử lý …

Tại một thời điểm, một tiến trình có thể nhận trong một các trạng thái sau đây :

New: tiến trình vừa được tạo lập;

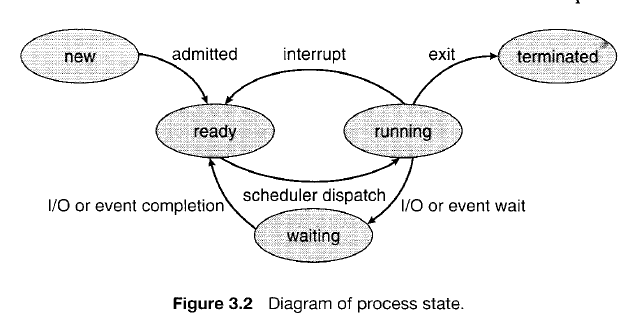
Running : các chỉ thị của tiến trình đang được thực thi;.

Waiting: tiến trình chờ được cấp phát một tài nguyên, hay chờ một sự kiện xảy ra .

Ready : tiến trình chờ được cấp phát CPU để xử lý.

Terminated: tiến trình hoàn tất xử lý.

Tên gọi của các trạng thái này không nhất thiết phải luôn như vậy, mỗi hệ điều hành khác nhau sẽ có cách đặt tên khác nhau. Tất nhiên là hệ điều hành cần phải xác định rõ ràng trạng thái của một tiến trình. Biểu đồ trạng thái của các tiến trình được giới thiệu ở hình bên dưới.



* + 1. **Các thuộc tính của tiến trình**

Trong các hệ thống, thông tin về mỗi tiến trình là rất cần thiết cho công tác quản lý của hệ điều hành, các thông tin này có thể trong khố quản lý tiên trình (PCB: process control block). Các hệ điều hành khác nhau sẽ có cách tổ chức PCB khác nhau, ở đây chúng ta khảo sát một trường hợp chung nhất.

Process state: các trạng thái có thể gồm có new, ready, running, waiting, halted…

Program counter: bộ đếm chương trình sẽ chỉ ra địa chỉ của chỉ thị tiếp theo cần được thực thi của tiến trình.

CPU registers: các thanh ghi khác nhau ở số lượng và loại, phụ thuộc vào kiến trúc máy tính. Chúng bao gồm thanh ghi tích lũy, thanh ghi chỉ số, stack pointers và các thanh ghi mục đích chung (general-purpose registers). Cùng với program counter, thông tin trạng thái mày phải được lưu lại khi có một gián đoạn xảy ra, việc này giúp cho tiến trình có thể tiếp tục được thực thi sau khi kết thúc gián đoạn đó.

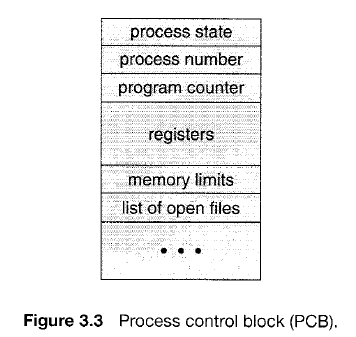
CPU-scheduling information: thông tin này bao gồm độ ưu tiên của tiến trình, con trỏ trỏ đến scheduling queues, và bất kì các tham số định thời nào khác.

Memor-management information

Accounting information: thông tin này chứa đựng thời gian thực sử dụng CPU, thời gian giới hạn, số lượng các account, số lượng job hoặc process…

I/O status information: thông tin này bao gồm danh sách các thiết bị I/O được cấp cho tiến trình, một danh sách chứa các file sử dụng…

Một cách ngắn gọn, PCB hoạt động như một nơi chứa lưu trữ các thông tin của process, mà dựa vào đó ta có thể phân biệt được các process với nhau.



**- Các thao tác điều khiển tiến trình**

- Khi khởi tạo tiên trình hệ điều hành thực hiện các thao tác sau:

+ Hệ điều hành gán PID cho tiến trình mới và đưa tiến trình vào danh sách quản lý của hệ thống.

+ Cấp phát không gian bộ nhớ cho tiến trình. ở đây hệ điều hành cần phải xác định được kích thước của tiến trình, bao gồm code, data và stack. Giá trị kích thước này có thể được gán mặt định dựa theo loại của tiến trình hoặc được gán theo yêu cầu của người sử dụng khi có một công việc được tạo. Nếu một tiến trình được sinh ra bởi một tiến trình khác thì tiến trình cha có thể chuyên kích thức cuat nó cho hệ điều hành trong yêu cầu tạo tiến trình.

+ Khởi tạo các thông tin cần thiết cho khối điều khiển các tiến trình như PID, thông tin trạng thái, độ ưu tiên của tiến trình..

+ Cung cấp đầy đủ các tài nguyên cần thiết nhất để tiến trình có thể vào trạng thái ready hoặc bắt đầu hoạt động.

Đưa tiến trình vào một danh sách tiến trình nào đó: ready list, waiting list, runnin list sao cho phù hợp với chiến lược điều phối của hệ điều hành.

* Khi kêt thúc tiến trình hệ điều hành thực hiện các thao tác sau: Khi tiến trình kết thúc xử lý, hoàn thành chỉ thị cuối cùng của hệ điều hành sẽ thực hiện các thao tác sau:

+ Thu hồi tài nguyên đã cấp phát cho tiến trình;

+ Loại bỏ tiến trình ra khỏi danh sách quản lý của hệ thống;

+ Hủy bỏ khối điều khiển tiến trình.

Hầu hết các hệ điều hành đều không cho phép tiến trình con hoạt động khi tiến trình cha đã kết thúc. Trong những trường hợp như thế, hệ điều hành sẽ chủ động việc kết thúc tiên trình con khi tiến trình cha vừa kết thúc.

* Khi thay đổi trạng thái tiến trình hệ điều hành thực hiện các bước sau: Khi một tiến trình đang ở trạng thái running bị chuyển sang trạng thái khác (ready, blocked..) thì hệ điều hành phải tạo ra sự thay đổi trong môi trường làm việc của nó. Sau đây là các bước mà hệ điều hành phải thực hiện đầy đủ khi thay đổi trạng thái của tiến trình:

+ Lưu ngữ cảnh của processor, bao gồm thanh ghi bộ đệm chương trình(PC: program counter) và các thanh ghi khác;

+ Cập nhật PCB của tiên trình, sao cho phù hợp với trạng thái mới của tiến trình, bao gồm trạng thái mới của tiến trình, các thông tin…

+ Di chuyển PCB của tiến trình đến một hàng đợi thích hợp, để đáp ứng được các yêu cầu của công tác điều phối tiến trình;

+Chọn một tiến trình khác để cho phép nó thực hiện;

+Cập nhât PCB của tiến trình được chọn ở trên, chủ yếu thay đổi trạng thái đến running;

+Cập nhật các thông tin liên quan đến quản lý bộ nhớ. Bước này phụ thuộc vào các yêu cầu chuyển đổi địa chỉ bộ nhớ đang được sử dụng.

+Khôi phục (Restore) lại ngữ cảnh của processor và thay đổi giá trị của bộ đếm chương trình và các thanh ghi khác sao cho phù hợp với tiến trình được chọn ở trên, để tiến trình này có thể bắt đầu hoạt động được.

Như vậy, khi hệ điều hành chuyển một tiến trình từ trạng thái running sang một trạng thái nào đó, thì hệ điều hành phải lưu các thông tin cần thiết để sau này hệ điều hành có thể cho tiến trình tiếp tục hoạt động trở lại được. Đồng thời hệ điều hành phải chọn một tiến trình nào đó đang ở trạng thái ready để cho tiến trình này chạy( chuyển tiến trình sang trạng thái running).

1. **Lập trình mạng**
   1. **Tổng quan về ngôn ngữ lập trình C#**

C # là một ngôn ngữ lập trình hiện đại được phát triển bởi Microsoft được ra đời cùng với .NET

C#, theo một hướng nào đó, là ngôn ngữ lập trình phản ánh trực tiếp nhất đến.NET Framework mà tất cả các chương trình.NET chạy, và nó phụ thuộc mạnh mẽ vào Framework này. Mọi dữ liệu cơ sở đều là đối tượng, được cấp phát và hủy bỏ bởi trình dọn rác Garbage-Collector (GC), và nhiều kiểu trừu tượng khác chẳng hạn như class, delegate, interface, exception, v.v, phản ánh rõ ràng những đặc trưng của.NET runtime.

So sánh với C và C++, ngôn ngữ này bị giới hạn và được nâng cao ở một vài đặc điểm nào đó, nhưng không bao gồm các giới hạn sau đây:

+Các con trỏ chỉ có thể được sử dụng trong chế độ không an toàn. Hầu hết các đối tượng được tham chiếu an toàn, và các phép tính đều được kiểm tra tràn bộ đệm. Các con trỏ chỉ được sử dụng để gọi các loại kiểu giá trị; còn những đối tượng thuộc bộ thu rác (garbage-collector) thì chỉ được gọi bằng cách tham chiếu.

+Các đối tượng không thể được giải phóng tường minh.

+Chỉ có đơn kế thừa, nhưng có thể cài đặt nhiều interface trừu tượng (abstract interfaces). Chức năng này làm đơn giản hóa sự thực thi của thời gian thực thi.

+C# thì an-toàn-kiểu (typesafe) hơn C++.

+Cú pháp khai báo mảng khác nhau("int[] a = new int[5]" thay vì "int a[5]").

+Kiểu thứ tự được thay thế bằng tên miền không gian (namespace).

+C# không có tiêu bản.

+Có thêm Properties, các phương pháp có thể gọi các Properties để truy cập dữ liệu.

+Có reflection.Ngôn ngữ C# khác đơn giản với ít từ khóa và kiểu dữ liệu.

* 1. **. Giới thiệu về lập trình Socket với TCPClient**

**+ Socket :** Socket định nghĩa những đặc trưng sau:

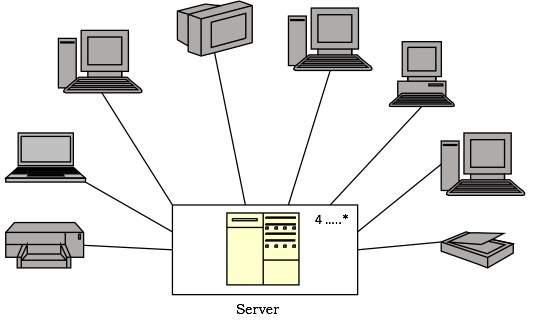
Một kết nối mạng hay một đường ống dẫn để truyền tải dữ liệu.

Một kiểu truyền thông như stream hay datagram.

Một giao thức như TCP hay UDP

* + 1. **Mô hình Client/Server**

Mô hình Client/Server là một mô hình nổi tiếng trong mạng máy tính, được áp dụng rất rộng rãi và là mô hình của mọi trang web hiện có. Ý tưởng của mô hình này là máy con (đóng vài trò là máy khách) gửi một yêu cầu (request) để máy chủ (đóng vai trò người cung ứng dịch vụ), máy chủ sẽ xử lý và trả kết quả về cho máy khách.



Mô hình client/server cung cấp một cách tiếp cận tổng quát để chia sẻ tài nguyên trong các hệ thống phân tán.

+ Cả tiến trình client và tiến trình server đều có thể chạy trên cùng một máy tính.

+ Một tiến trình server có thể sử dụng dịch vụ của một server khác.

+ Mô hình truyền tin client/serverhướng tới việc cung cấp dịch vụ

Quá trình trao đổi dữ liệu bao gồm:

+Bước 1 : Truyền một yêu cầu từ tiến trình client tới tiến trình server

+ Bước 2 : Yêu cầu được server xử lý

+ Bước 3 : Truyền đáp ứng cho client1.

Khi được chạy, server cần được xác định rõ địa chỉ IP và sẽ “lắng nghe” trên một port cụ thể. Server sẽ nằm trong trạng thái này cho đến khi client gửi đến một yêu cầu kết nối. Sau khi được server chấp nhận, một connection sẽ hình thành cho phép server và client giao tiếp với nhau.

Cụ thể hơn, các bước tiến hành trên server và client mà ta cần thực hiện sử dụng giao thức TCP/IP trong C#

Server

+Tạo một đối tượng System.Net.Sockets.TcpListener để bắt đầu “lắng nghe” trên một cổng cục bộ.

+Đợi và chấp nhận kết nối từ client với phương thức AccepSocket(). Phương thức này trả về một đối tượng System.Net.Sockets.Socket dùng để gửi và nhận dữ liệu.

+Thực hiện giao tiếp với client.

+Đóng Socket.

Thông thường quy trình này sẽ được đặt trong một vòng lặp (lặp lại bước 2) để chấp nhận nhiều kết nối cùng lúc (sử dụng Thread) hoặc các kết nối lần lượt.

Client:

+Tạo một đối tượng System.Net.Sockets.TcpClient

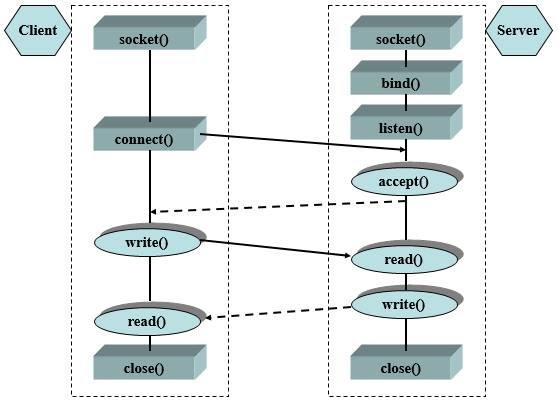
+Kết nối đến server với địa chỉ và port xác định với phương thức TcpClient.Connect()

+Lấy luồng (stream) giao tiếp bằng phương thức TcpClient.GetStream().

+Thực hiện giao tiếp với server.

+Đóng luồng và socket.

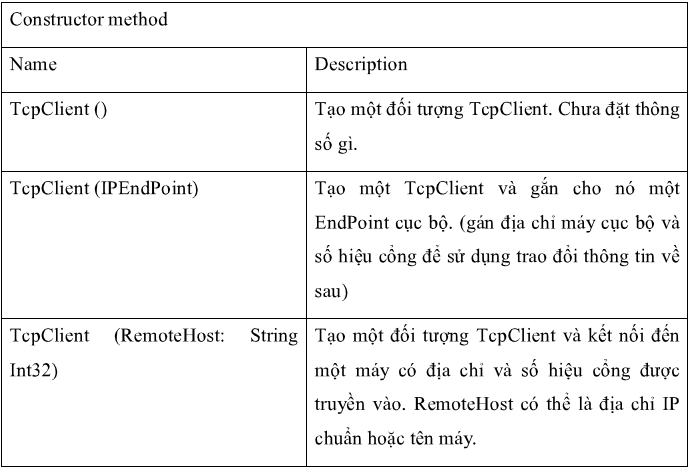
Quy trình này có thể được minh họa theo mô hình sau:



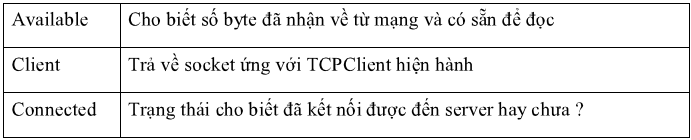
* + 1. **Lớp TCPClient**

Các thành phần

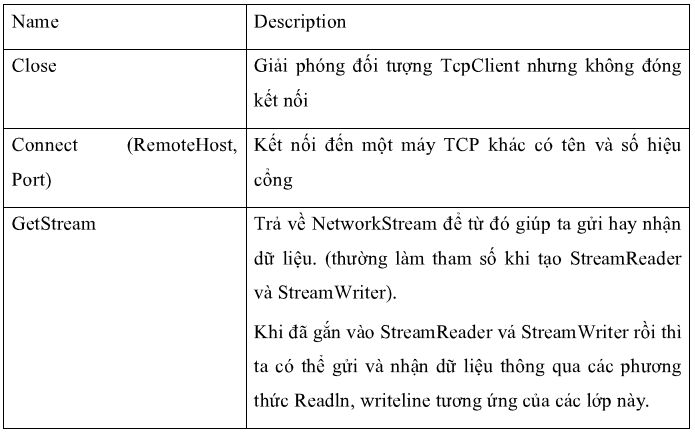
Phương thức khởi tạo



Một số thuộc tính



Một số phương thức:



* + 1. **Lớp TCPListener**

TCPListener là một lớp cho phép người lập trình có thể xây dựng các ứng dụng server.

Các thành phần của lớp TcpListener

Phương thức khởi tạo:

|  |  |
| --- | --- |
| Name Description |  |
| TcpListener (IPAddress, Int32)Tạo một TcpListener và lắng nghe tại cổng chỉ địnhTcpListener (Port: Int32) | Tạo một TcpListener và lắng nghe các kết nối đến tại địa chỉ IP và cổng chỉ định |
| TcpListener (IPEndPoint) | Tạo một TcpListener với giá trị EndPoint truyền vào. |

Các phương thức khác:

|  |  |
| --- | --- |
| Name Description |  |
| AcceptTcpClient | Chấp nhận một yêu cầu kết nối đang chờ. (ứng dụng sẽ dừng tại câu lệnh này cho đến khi nào có một kết nối đến) |
| AcceptSocket | Chấp nhận một yêu cầu kết nối đang chờ. |
| Pending | Cho biết liệu có kết nối nào đang chờ đợi không? ( True = có). |
| Start | Bắt đầu lắng nghe các yêu cầu kết nối |
| Stop | Dừng việc nghe |

# CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

1. **Phân tích yêu cầu**
   1. **Đề tài**

“Xây dựng chương trình quản lý tiến trình từ xa”

* 1. **Yêu cầu**

Kết nối giữa client và server qua mạng LAN;

Hiển thị thông tin của client;

Server quản lý được thông tin của client, hiển thị các tiến trình hiện có, thêm mới và kết thúc tiến trình.

* 1. **Triển khai**

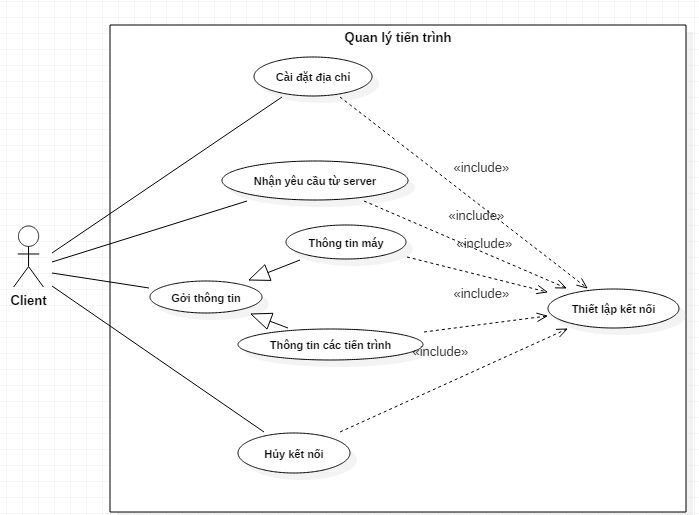
Tìm hiểu mô hình mạng client/server, mô hình OSI và TCP/IP để nắm vững kiến thức cơ bản về mạng máy tính.

Tìm hiểu về giao thức TCP.

Sử dụng ngôn ngữ lập trình C# và kiến thức đã học, xây dựng chương trình theo giao thức TCP trên môi trường Visual Studio 2015.

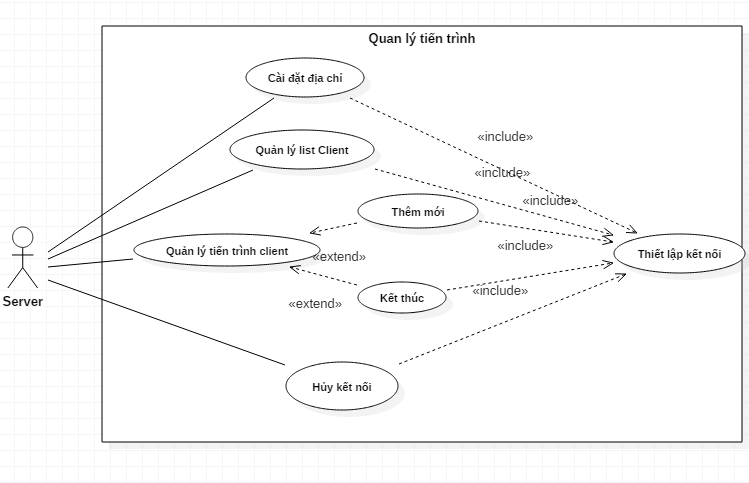
1. **Phân tích hệ thống**
   1. **Sơ đồ Use case**

**Client:**



**Hình 1: Use case Client**

**Server:**



**Hình 2: Use case Server**

# CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

1. **Môi trường làm việc**

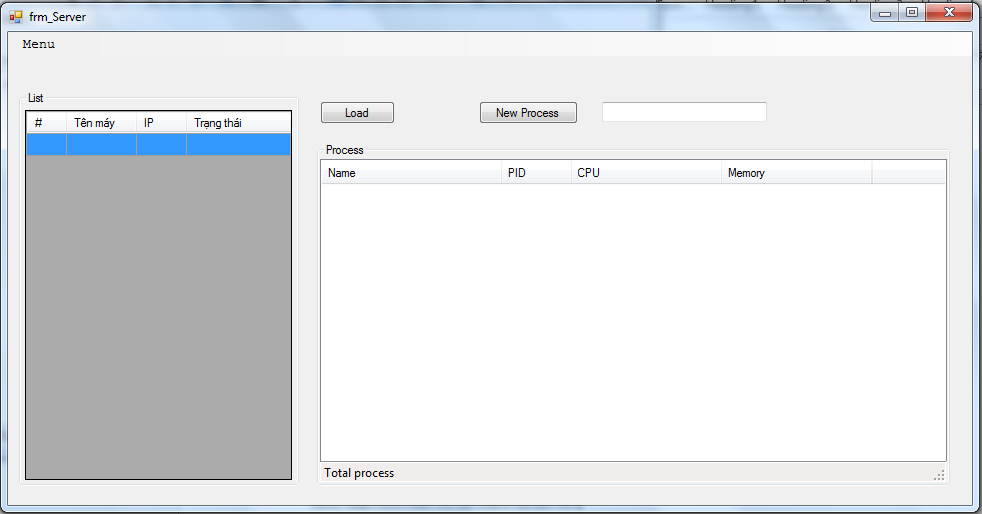
Hệ điều hành: Window.

Ngôn ngữ lập trình: C#

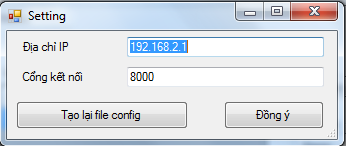
Công cụ: Visual Studio 2015

1. **Triển khai**

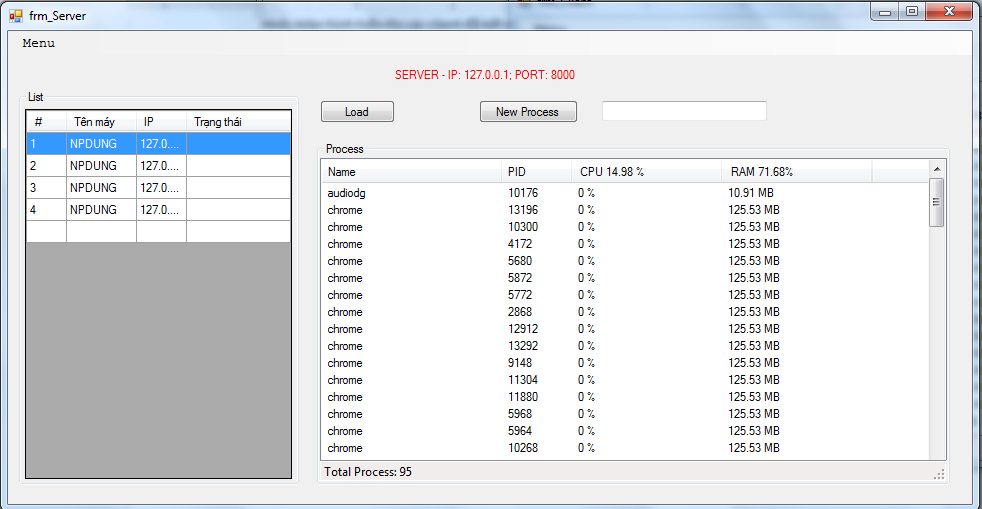
### **2.1 Server**



Hình : Giao diện ban chính của Server



Hình : Người dùng cài đặt địa chỉ ip và cổng giao tiếp ở Server

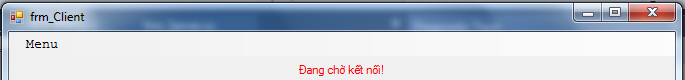


Hình: Màn hình hiển thị các client đã kết nối và process của một máy

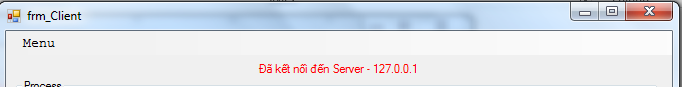
Tại đây, nếu muốn tạo mới một tiến trình bạn có thể điền tên tiến trình và nhấn new process, ở client sẽ nhận được lệnh và tạo tiến trình với tên tương ứng, tương tự với khi kết thúc, bạn nhấn chuột phải chon end task.

Hình:

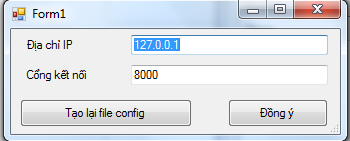
### **Client**



Hình: Màn hình chờ kết nối



*Hình Màn hình kết nối server*



*Hình 9. Màn hình cài đặt ip và cổng giao tiếp*

**Đánh giá kết quả**

* Thực hiện đúng mục tiêu đề ra:

+ Server có thể xem danh sách các client thuộc phạm vi xử lý;

+ Xem các tiến trình, hủy, thêm ở client

+ Ngoài ra bổ xung các tính năng như tắt máy, khởi động lại máy từ server

+ Giao tiếp trong mạng LAN tương đối ổn đinh, tuy nhiên tốc độ còn chậm

# CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. **Kết luận**

Những vấn đề đã đạt được:

Trong quá trình làm đồ án, với những kiến thức đã học và tự tìm hiểu em đã hiểu hơn về lập trình socket

Mặc khác em đã cải thiện được kĩ năng lập trình với ngôn ngữ C# và cách sử dụng Visual Studio 2015

Những vấn đề chưa đạt:

Do thời gian có hạn và kiến thức của em còn hạn chế nên vẫn còn tồn tại một số vấn đề như giao diện chưa đẹp mắt, tốc độ xử lý chưa tối ưu, các thông số về tiến trình chưa đầy đủ

1. **Hướng phát triển**

Tiếp tục nghiên cứu lý thuyết và thực hành.

Tìm hiểu cách để xử lý vấn đề nhanh và tối ưu hơn, giai quyết các vấn đề còn hạn chế

Xây dựng một chương trình hoàn thiện và thân thiện hơn.

**TÀI LIỆU THAM KHẢO**