TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
VÀ TRUYỀN THÔNG VIỆT - HÀN
KHOA KỸ THUẬT MÁY TÍNH VÀ ĐIỆN TỬ

BÁO CÁO
LẬP TRÌNH MẠNG

ĐỀ TÀI:

XÂY DỰNG CHƯƠNG TRÌNH GIÁM SÁT MẠNG LAN THEO MÔ HÌNH CLIENT – SERVER

Ngành **: AN TOÀN THÔNG TIN**
Sinh viên thực hiện **: Đinh Hữu Đức**
Lớp **: 20NS**
Giảng viên hướng dẫn **: ThS. Nguyễn Thanh Cẩm**

Đà Nẵng, tháng 12 năm 2022

LỜI CẢM ƠN

Lời đầu tiên chúng em xin trân trọng cảm ơn và bày tỏ lòng biết ơn sâu sắc nhất tới thầy ThS. Nguyễn Thanh Cẩm – Giảng viên Khoa Kỹ thuật máy tính và Điện tử, Trường Đại học CNTT & TT Việt – Hàn, giáo viên hướng dẫn môn học Lập trình mạng đã nhiệt tình hướng dẫn, chỉ bảo trong môn học lần này.

Chúng em xin chân thành cảm ơn các thầy cô giáo đang giảng dạy tại Trường Đại học CNTT & TT Việt – Hàn đã nhiệt tình ủng hộ, cung cấp tài liệu và đưa ra những ý kiến đóng góp quý báu!

Cuối cùng, chúng tôi xin chân thành giành lời cảm ơn chân thành tới bạn bè đã động viên, khuyến khích và tạo điều kiện cho chúng tôi hoàn thành tốt đề tài của mình.

Xin chân thành cảm ơn!

Đà Nẵng, tháng 12 năm 2022

Sinh viên thực hiện

Đinh Hữu Đức

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Giảng viên hướng dẫn

MỤC LỤC

DANH MỤC BẢNG BIỂU

DANH MỤC HÌNH ẢNH

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI & CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về đề tài

1.1.1. Lý do chọn đề tài

Với sự phát triển nhanh chóng của khoa học công nghệ và đặc biệt là ngành công nghệ thông tin, làm từng bước thay đổi cuộc sống của con người bởi các ứng dụng sản phẩm của công nghệ. Đi kèm với đó là ngày càng có nhiều doanh nghiệp phát triển, số lượng máy tính dùng trong các doanh nghiệp, công ty ngày càng tăng dẫn tới việc điều khiển, theo dõi hoạt động trên nhiều máy tính của các doanh nghiệp, công ty đang ngày một cấp thiết. Vì vậy em đã lựa chọn đề tài "**Xây dựng chương trình giám sát mạng LAN theo mô hình client - server**" để làm đồ án với ý nghĩa mang tính thực tế. Với mong muốn có thể giúp các cơ quan doanh nghiệp dễ dàng hơn trong việc quản lý các máy tính trong mạng LAN của mình.

1.1.2. Mục tiêu đề tài

- Xây dựng giao diện trực quan, sinh động, dễ nhìn
- Thiết kế giao diện cho phép xử lý các chức năng chính trong chương trình.
- Thiết kế giao diện chương trình chính thực hiện các công việc:
- Chat giữa máy server và máy client
- Gửi thông điệp cho máy client bất kì
- Truyền tập tin
- Theo dõi hoặc điều khiển client
- Chụp hình client
- Gửi lệnh shell
- Xem lịch sử

1.1.3. Giới hạn phạm vi nghiêm cứu

- Sử dụng ngôn ngữ Java, phần mền Eclipse, Apache NetBeans
- Nghiên cứu trong phạm vi khoa học, nội dung, cách thực hiện và phát triển một phần mềm có thể quản lý nhiều máy tính cùng lúc.

1.2. Cơ sở lý thuyết

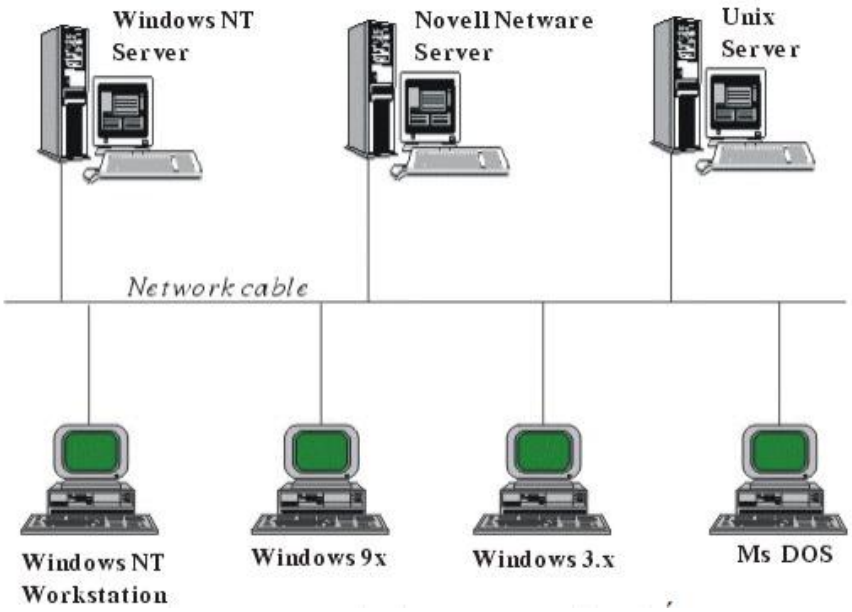
1.2.1. Tổng quan về lập trình mạng

1.2.1.1. Định nghĩa mạng máy tính

Mạng máy tính là một tập hợp các máy tính được nối với nhau bởi đường truyền theo một cấu trúc nào đó và thông qua đó các máy tính trao đổi thông tin qua lại cho

nhau.

Đường truyền là hệ thống các thiết bị truyền dẫn có dây hay không dây dùng để chuyển các tín hiệu điện tử từ máy tính này đến máy tính khác. Các tín hiệu điện tử đó biểu thị các giá trị dữ liệu dưới dạng các xung nhị phân (on - off). Tất cả các tín hiệu được truyền giữa các máy tính đều thuộc một dạng sóng điện từ. Tùy theo tần số của sóng điện từ có thể dùng các đường truyền vật lý khác nhau để truyền các tín hiệu. Ở đây đường truyền được kết nối có thể là dây cáp đồng trục, cáp xoắn, cáp quang, dây điện thoại, sóng vô tuyến ... Các đường truyền dữ liệu tạo nên cấu trúc của mạng. Hai khái niệm đường truyền và cấu trúc là những đặc trưng cơ bản của mạng máy tính.

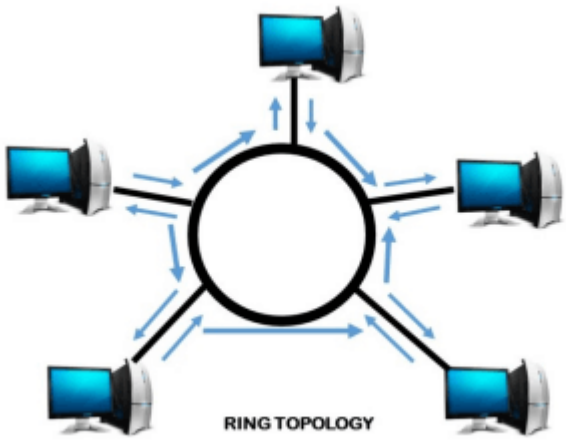


Hình 1.1: Một mô hình các máy tính liên kết trong mạng

1.2.1.2. Một số topo mạng thông dụng

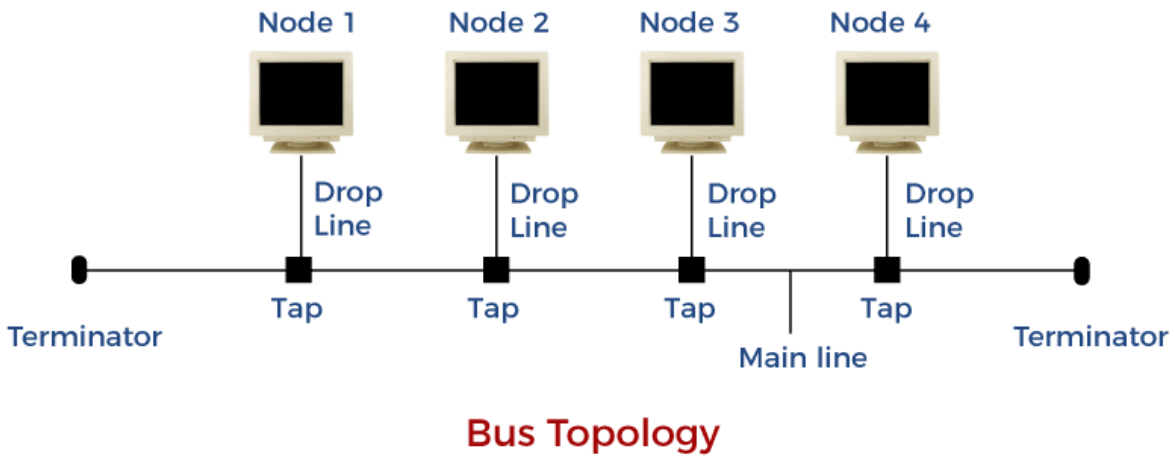
Theo định nghĩa về mạng máy tính, các máy tính được nối với nhau bởi các đường truyền vật lý theo một kiến trúc nào đó, các kiến trúc đó gọi là Topology. Thông thường mạng có ba loại kiến trúc đó là: mạng hình sao (Star Topology), mạng dạng tuyến (Bus Topology), mạng dạng vòng (Ring Topology).

- Ring Topology: Mạng được bố trí vòng tròn, đường dây cáp được thiết kế làm thành một vòng khép kín, tín hiệu chạy theo một chiều nào đó. Các nút truyền tín hiệu cho nhau tại một thời điểm được một nút mà thôi. Mạng dạng vòng có thuận lợi là có thể nới rộng ra xa nhưng đường dây phải khép kín, nếu bị ngắt ở một nơi nào đó thì toàn bộ hệ thống cũng bị ngưng.



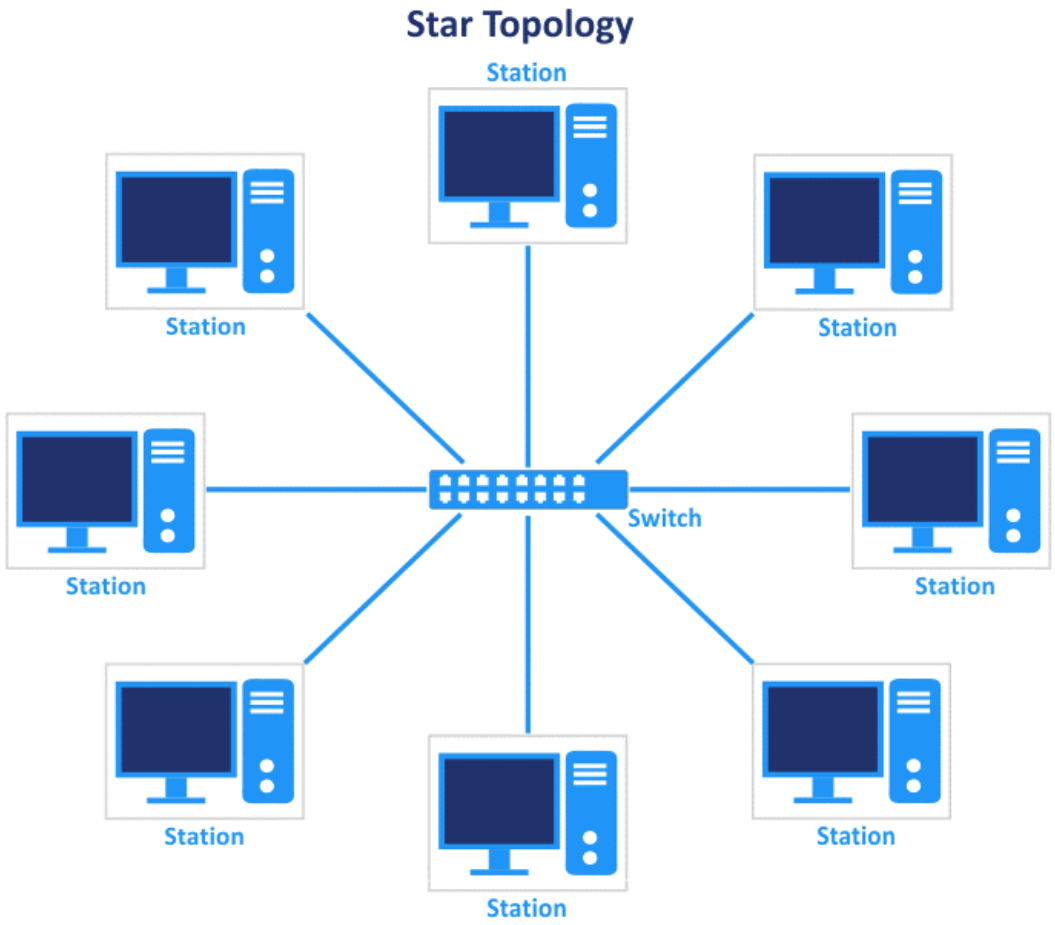
Hình 1.2: Ring Topology

- Bus Topology: Ở dạng Bus tất cả các nút được phân chia một đường truyền chính (bus). Đường truyền này được giới hạn hai đầu bởi một loại đầu nối đặc biệt gọi là Terminator. Khi một nút truyền dữ liệu, tín hiệu được quảng bá trên hai chiều của bus, mọi nút còn lại đều được nhận tín hiệu trực tiếp. Loại mạng này dùng dây cáp ít, dễ lắp đặt. Tuy vậy cũng có những bất lợi đó là sẽ có sự ùn tắc giao thông khi di chuyển với lưu lượng lớn và khi có sự hỏng hóc ở đoạn nào đó thì rất khó phát hiện, nếu một nút ngừng hoạt động sẽ ảnh hưởng tới toàn bộ hệ thống.



Hình 1.3: Bus Topology

- Star Topology: Mạng hình sao bao gồm một bộ tập trung và các nút thông tin. Các nút thông tin có thể là các trạm cuối, các máy tính hay các thiết bị khác của mạng. Mạng hoạt động theo nguyên lý nối song song nên nếu có một nút bị hỏng mạng vẫn hoạt động bình thường. Mạng có thể mở rộng hoặc thu hẹp tùy theo yêu cầu của người sử dụng, tuy nhiên mở rộng phụ thuộc và khả năng của trung tâm.



Hình 1.4: Star Topology

1.2.1.3. Giao thức mạng

Giao thức mạng là một tập các quy tắc, quy ước để trao đổi thông tin giữa hai hệ thống máy tính hoặc hai thiết bị máy tính với nhau. Nói một cách hình thức thì giao thức mạng là một ngôn ngữ được các máy tính trong mạng sử dụng để trao đổi dữ liệu với nhau. Có nhiều loại giao thức được sử dụng trong mạng máy tính như: Apple Talk, DLC, NetBEUI,… nhưng hiện nay giao thức được sử dụng phổ biến nhất trong mạng máy tính là giao thức TCP/IP.

❖ Giao Thức TCP

Định nghĩa: TCP (Transmission Control Protocol) là giao thức hướng kết nối, nó cung cấp một đường truyền dữ liệu tin cậy giữa hai máy tính. Tính tin cậy thể hiện ở việc nó đảm bảo dữ liệu được gửi sẽ đến được đích và theo đúng thứ tự như khi nó được gửi. Khi hai ứng dụng muốn giao tiếp với nhau một cách tin cậy, chúng sẽ tạo ra đường kết nối giữa chúng và gửi dữ liệu thông qua đường này. Cách trao đổi dữ liệu này tương tự như cách chúng ta gọi điện thoại. Hãy lấy ví dụ khi bạn nhấc điện thoại lên và quay số của người họ hàng này, lúc đó một kết nối sẽ được tạo ra giữa điện

thoại của bạn và người họ hàng, sau đó bạn gửi và nhận dữ liệu (dưới dạng âm thanh) bằng cách nói và nghe qua điện thoại của bạn. Toàn bộ việc thực hiện kết nối và truyền dữ liệu giữa hai máy điện thoại được thực hiện bởi công ty điện thoại thông qua các trạm và đường dây điện thoại, nhiệm vụ duy nhất của bạn là quay số để cung cấp cho nhà cung cấp dịch vụ điện thoại biết số điện thoại bạn muốn liên lạc. Giống như vậy, trong việc truyền dữ liệu qua mạng thì TCP đóng vai trò như nhà cung cấp dịch vụ điện thoại ở ví dụ trên, nó làm nhiệm vụ tạo kết nối và truyền dữ liệu giữa hai điểm giao tiếp để đảm bảo dữ liệu không bị mất và tới đích theo đúng trật tự như khi chúng ta gửi. Tính tin cậy của đường truyền được thể hiện ở hai điểm sau:

- Mọi gói tin cần gửi sẽ đến được đích. Để làm được điều này thì mỗi lần phía gửi gửi xong một gói tin nó sẽ chờ nhận một xác nhận từ bên nhận rằng đã nhận được gói tin. Nếu sau một khoảng thời gian mà phía gửi không nhận được thông tin xác nhận phản hồi thì nó sẽ phát lại gói tin. Việc phát lại sẽ được tiến hành cho đến khi việc truyền tin thành công, tuy nhiên sau một số lần phát lại max nào đó mà vẫn chưa thành công thì phía gửi có thể suy ra là không thể truyền tin được và sẽ dừng việc phát tin.

- Các gói tin sẽ được trình ứng dụng nhận được theo đúng thứ tự như chúng được gửi. Bởi các gói tin có thể được dẫn đi trên mạng theo nhiều đường khác nhau trước khi tới đích nên thứ tự khi tới đích của chúng có thể không giống như khi chúng được phát. Do đó để đảm bảo có thể sắp xếp lại gói tin ở phía nhận theo đúng thứ tự như khi chúng được gửi, giao thức TCP sẽ gắn vào mỗi gói tin một thông tin cho biết thứ tự của chúng trong cả khối tin chung được phát nhờ vậy bên nhận có thể sắp xếp lại các gói tin theo đúng thứ tự của chúng.

Như vậy có thể thấy TCP cung cấp cho chúng ta một kênh truyền thông điểm-điểm phục vụ cho các ứng dụng đòi hỏi giao tiếp tin cậy như HTTP (HyperText Transfer Protocol), FTP (File Tranfer Protocol), Telnet…. Các ứng dụng này đòi hỏi một kênh giao tiếp tin cậy bởi thứ tự của dữ liệu được gửi và nhận là yếu tố quyết định đến sự thành công hay thất bại của chúng. Hãy lấy ví dụ khi HTTP được sử dụng để đọc thông tin từ một địa chỉ URL, dữ liệu phải được nhận theo đúng thứ tự mà chúng được gửi nếu không thứ mà bạn nhận được có thể là một trang HTML với nội dung lộn xộn hoặc một file zip bị lỗi và không giải nén….

❖ Giao Thức UDP

Định nghĩa: UDP (User Datagram Protocol) là giao thức không hướng kết nối, nó gửi các gói dữ liệu độc lập gọi là datagram từ máy tính này đến máy tính khác mà không đảm bảo việc dữ liệu sẽ tới đích.

Ở phần trước chúng ta đã thấy trong giao thức TCP khi hai chương trình muốn giao tiếp với nhau qua mạng chúng tạo ra một kết nối liên kết hai ứng dụng và trao đổi dữ liệu qua kết nối đó. Trái lại ở giao thức UDP khi hai ứng dụng muốn giao tiếp với nhau chúng không tạo ra kết nối mà chỉ đơn thuần gửi các gói tin một cách độc lập từ máy này tới máy khác. Các gói tin như vậy gọi là các datagram. Việc gửi các gói tin như vậy tương tự như việc chúng ta gửi thư qua đường bưu điện. Các bức thư bạn gửi độc lập với nhau, thứ tự các thư là không quan trọng và không có gì đảm bảo là thư sẽ đến được đích. Trong truyền thông bằng UDP thì các datagram giống như các lá thư, chúng chứa thông tin cần gửi đi cùng thông tin về địa chỉ đích mà chúng phải đến, tuy nhiên chúng khác với các lá thư ở một điểm là nếu như trong việc gửi thư, nếu lá thư của bạn không đến được đích thì nó sẽ được gửi trả lại nơi gửi nêu trên lá thư đó bạn có đề địa chỉ gửi còn UDP sẽ không thông báo gì cho phía gửi về việc lá thư đó có tới được đích hay không.

Vậy nếu UDP là một giao thức không đảm bảo giao tiếp tin cậy thì tại sao người ta lại dùng chúng. Điều đó là bởi nếu như giao thức TCP đảm bảo một kết nối tin cậy giữa các ứng dụng thì chúng cũng đòi hỏi nhiều thời gian để truyền tin do chúng phải kiểm tra các header các gói tin để đảm bảo thứ tự các gói tin cũng như để phát lại các gói tin không đến được đích do đó trong một số trường hợp thì điều này không cần thiết. Dưới đây là một số trường hợp trong đó giao thức không hướng kết nối là thích hợp hơn so với giao thức hướng kết nối:

Khi chỉ một gói dữ liệu cần truyền đi và việc đó đến được đích hay không là không quan trọng, sử dụng giao thức UDP sẽ loại bỏ được các thủ tục tạo và hủy kết nối. So sánh một chút chúng ta sẽ thấy giao thức hướng kết nối TCP phải dùng đến bẩy gói tin để gửi một gói tin do nó cần phát và nhận các gói tin yêu cầu và chấp nhận kết nối cũng như các gói tin yêu cầu và xác nhận việc hủy kết nối, trong khi đó giao thức không kết nối UDP chỉ sử dụng duy nhất một gói tin chính là gói tin chứa dữ liệu cần chuyển đi.

Chúng ta lấy ví dụ về một server đồng hồ, nhiệm vụ của nó là gửi thời gian hiện tại của nó cho các ứng dụng trên client khi có yêu cầu. Nếu gói tin chứa thời gian bị thất lạc trên đường truyền và không tới được đích thì client cũng sẽ không đòi hỏi server phải gửi lại gói tin đó bởi khi gói tin đó được phát lại lần hai và tới được client thì thông tin thời gian chứa trong nó đã không còn đúng nữa. Nếu client tạo ra hai yêu cầu và nhận được các gói tin trả lời không theo đúng thứ tự mà server đã gửi thì client cũng không gặp phải vấn đề gì bởi nó hoàn toàn có thể suy ra được rằng các gói đã không được chuyển đến đúng thứ tự bằng cách tính thời gian được chứa trong các gói. Trong trường hợp này tính tin cậy của TCP là không cần thiết bởi nó làm giảm hiệu suất và có thể cản trở hoạt động của server.

Trường hợp thứ hai chúng ta xem xét việc sử dụng giao thức UDP là các ứng dụng đòi hỏi chặt chẽ về thời gian như các ứng dụng nghe audio thời gian thực. Trong trường hợp này việc hướng tới một kênh giao tiếp tin cậy không phải là ưu điểm mà ngược lại đó là một nhược điểm bởi nếu việc phải chờ cho khi một gói tin bị mất được nhận có thể gây ra những tác động dễ nhận thấy hoặc khiến chương trình phải tạm ngừng. Với các ứng dụng này giao thức không hướng kết nối đã được phát triển và chúng làm việc tốt hơn hẳn. Chúng ta có thể tham khảo ứng dụng RealAudio, trong đó người ta sử dụng một giao thức không hướng kết nối để truyền các dữ liệu âm thanh qua mạng.

Bảng sau so sánh sự khác biệt giữa hai chế độ giao tiếp hướng kết nối và không hướng kết nối.

Bảng 1.1: So sánh sự khác nhau giữa TCP & UDP

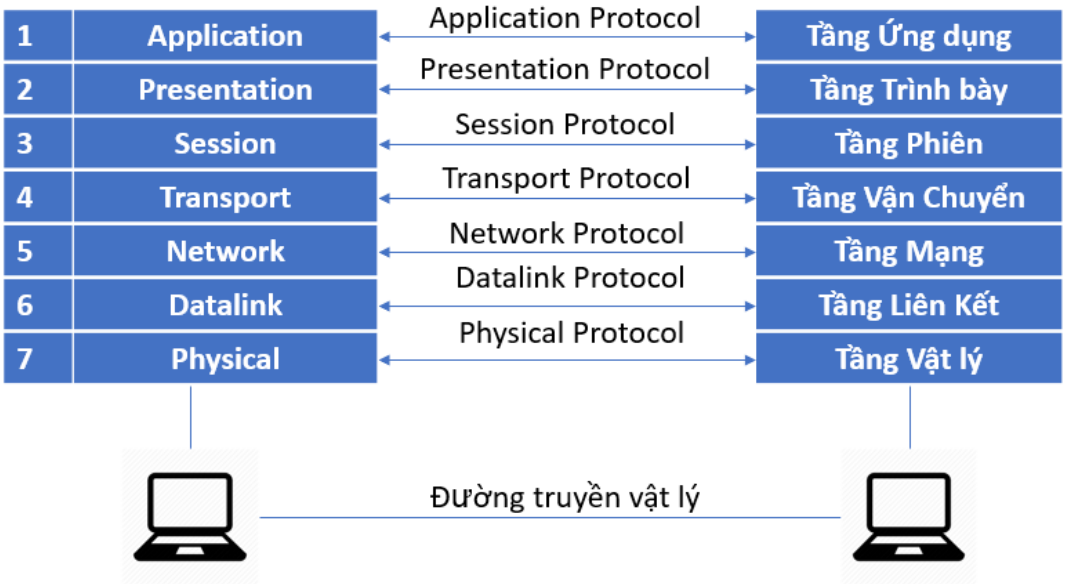
Chế độ có hướng kết nối (TCP)	**Chế độ không hướng kết nối(UDP)**
Tồn tại kênh giao tiếp ảo giữa hai bên giao tiếp	Không tồn tại kênh giao tiếp ảo giữa hai bên giao tiếp
Dữ liệu được gửi đi theo chế độ bảo đảm: Có kiểm tra lỗi truyền lại gói tin lỗi hay mất bảo đảm thứ tự đến của các gói tin…	Dữ liệu được gửi đi theo chế độ không bảo đảm: Không kiểm tra lỗi, không phát hiện không truyền lại gói tin bị lỗi hay mất, không bảo đảm thứ tự đến của các gói tin…
Dữ liệu chính xác, tốc độ truyền chậm	Dữ liệu không chính xác, tốc độ truyền nhanh.
	Thích hợp cho các ứng dụng cần tốc độ,

Chế độ có hướng kết nối (TCP)	Chế độ không hướng kết nối(UDP)
	không cần chính xác cao: Truyền âm thanh, hình ảnh.

1.2.1.4. Các mô hình hệ thống truyền thông

❖ **Mô hình tham chiếu OSI**

Mô hình kết nối các hệ thống mở OSI là mô hình căn bản về các tiến trình truyền thông, thiết lập các tiêu chuẩn kiến trúc mạng ở mức Quốc tế, là cơ sở chung để các hệ thống khác nhau có thể liên kết và truyền thông được với nhau. Mô hình OSI tổ chức các giao thức truyền thông thành 7 tầng, mỗi một tầng giải quyết một phần hẹp của tiến trình truyền thông, chia tiến trình truyền thông thành nhiều tầng và trong mỗi tầng có thể có nhiều giao thức khác nhau thực hiện các nhu cầu truyền thông cụ thể.



Hình 1.5: Mô hình OSI

Mô hình OSI tuân theo các nguyên tắc phân tầng như sau:

- Mô hình gồm N = 7 tầng. OSI là hệ thống mở, phải có khả năng kết nối với các hệ thống khác nhau, tương thích với các chuẩn OSI.

- Quá trình xử lý các ứng dụng được thực hiện trong các hệ thống mở, trong khi vẫn duy trì được các hoạt động kết nối giữa các hệ thống.

- Thiết lập kênh logic nhằm mục đích thực hiện việc trao đổi thông tin giữa các thực thể.

Các giao thức trong mô hình OSI:

- Trong mô hình OSI có hai loại giao thức được sử dụng: giao thức hướng liên kết (Connection – Oriented) và giao thức không liên kết (Connectionless).

Giao thức hướng liên kết:

- Trước khi truyền dữ liệu, các thực thể đồng tầng trong hai hệ thống cần phải thiết lập một liên kết logic. Chúng thương lượng với nhau về tập các tham số sẽ sử dụng trong giai đoạn truyền dữ liệu, cắt/hợp dữ liệu, liên kết sẽ được hủy bỏ. Thiết lập liên kết logic sẽ nâng cao độ tin cậy và an toàn trong quá trình trao đổi dữ liệu.

Giao thức không liên kết:

- Dữ liệu được truyền độc lập trên các tuyến khác nhau. Với các giao thức không liên kết chỉ có giai đoạn duy nhất truyền dữ liệu.

Tóm tắt chức năng các tầng giao thức trong OSI:

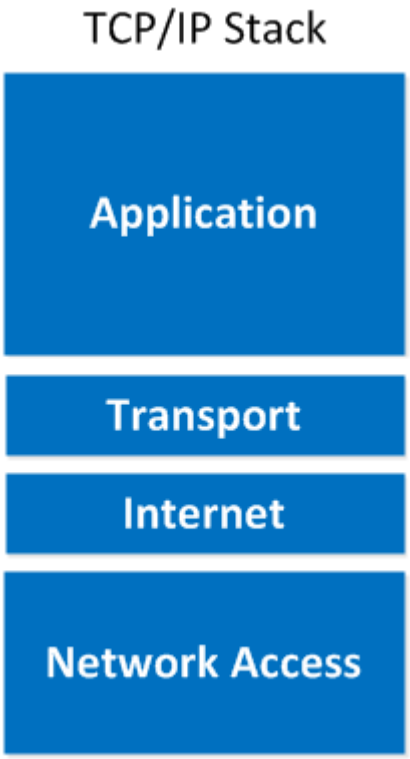
Bảng 1.2: Chức năng các tầng trong mô hình OSI

Tầng	Chức năng chủ yếu	Giao thức
7 – Application	Giao tiếp người và môi trường mạng	Ứng dụng
6 – Presentation	Chuyển đổi cú pháp dữ liệu để đáp ứng yêu cầu truyền thông của các ứng dụng	Giao thức Biến đổi mã
5 - Session	Quản lý các cuộc liên lạc giữa các thực thể bằng cách thiết lập, duy trì, đồng bộ hóa và hủy bỏ các phiên truyền thông giữa các ứng dụng	Giao thức phiên
4 – Transpost	Vận chuyển thông tin giữa các máy chủ (End to End). Kiểm soát lỗi và luồng dữ liệu	Giao thức Giao vận
3 – Network	Thực hiện chọn đường và đảm bảo trao đổi thông tin trong liên mạng với công nghệ chuyển mạch thích hợp.	Giao thức mạng
2 – Data Link	Tạo/gỡ bỏ khung thông tin (Frames), kiểm soát luồng và kiểm soát lỗi.	Thủ tục kiểm soát
1 - Physical	Đảm bảo các yêu cầu truyền/nhận các chuỗi bit qua các phương tiện vật lý.	Giao diện DTE - DCE

❖ **Mô hình giao thức TCP/IP**

TCP/ IP (Transmission Control Protocol/ Internet Protocol - Giao thức điều khiển truyền nhận/ Giao thức liên mạng), là một bộ giao thức trao đổi thông tin được sử

dụng để truyền tải và kết nối các thiết bị trong mạng Internet. TCP/IP được phát triển để mạng được tin cậy hơn cùng với khả năng phục hồi tự động.



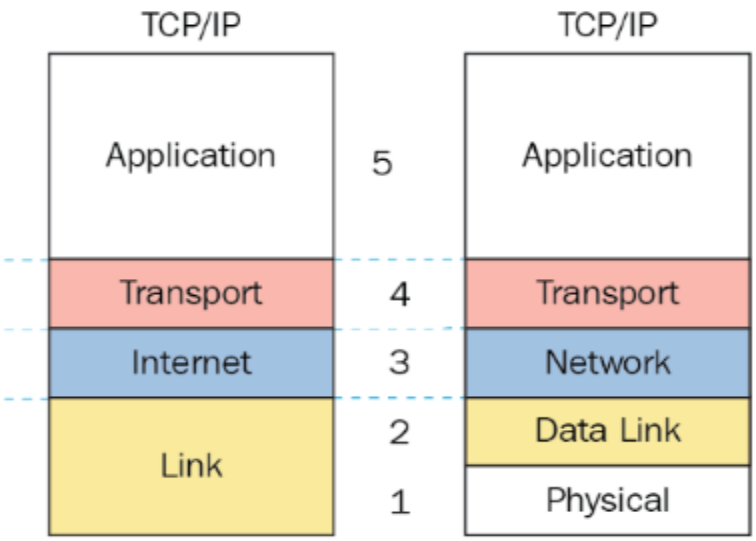
Hình 1.6: Mô hình TCP/IP

Cách thức hoạt động của mô hình TCP/IP:

- Phân tích từ tên gọi, TCP/IP là sự kết hợp giữa 2 giao thức. Trong đó IP (Giao thức liên mạng) cho phép các gói tin được gửi đến đích đã định sẵn, bằng cách thêm các thông tin dẫn đường vào các gói tin để các gói tin được đến đúng đích đã định sẵn ban đầu. Và giao thức TCP (Giao thức truyền vận) đóng vai trò kiểm tra và đảm bảo sự an toàn cho mỗi gói tin khi đi qua mỗi trạm. Trong quá trình này, nếu giao thức TCP nhận thấy gói tin bị lỗi, một tín hiệu sẽ được truyền đi và yêu cầu hệ thống gửi lại một gói tin khác. Quá trình hoạt động này sẽ được làm rõ hơn ở chức năng của mỗi tầng trong mô hình TCP/IP.

Chức năng của các tầng trong mô hình TCP/IP:

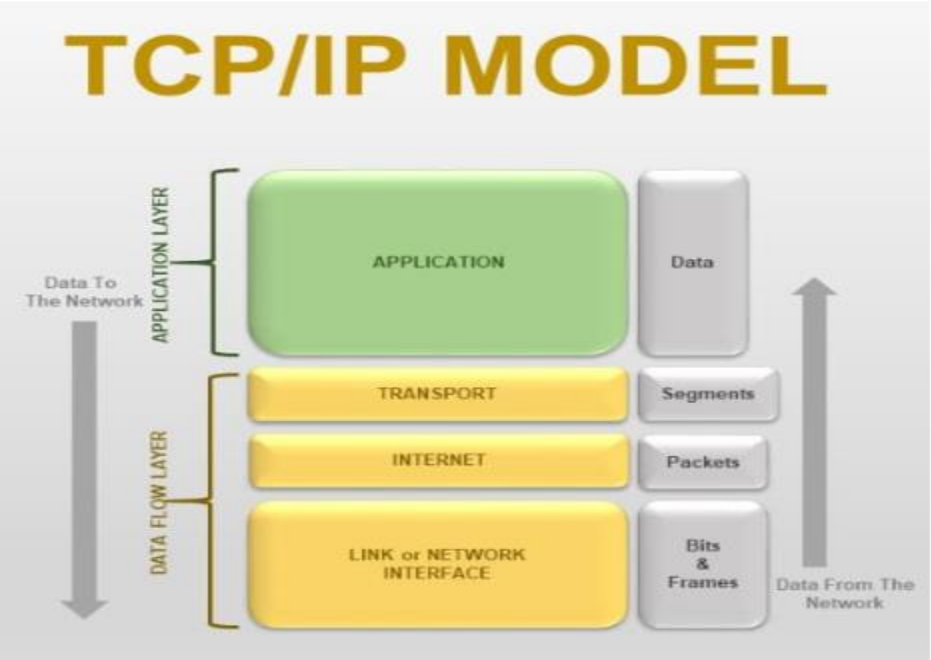
- Một mô hình TCP/IP tiêu chuẩn bao gồm 4 lớp được chồng lên nhau, bắt đầu từ tầng thấp nhất là Tầng vật lý (Physical) → Tầng mạng (Network) → Tầng giao vận (Transport) và cuối cùng là Tầng ứng dụng (Application).



Hình 1.7: Chức năng các tầng trong mô hình TCP/IP

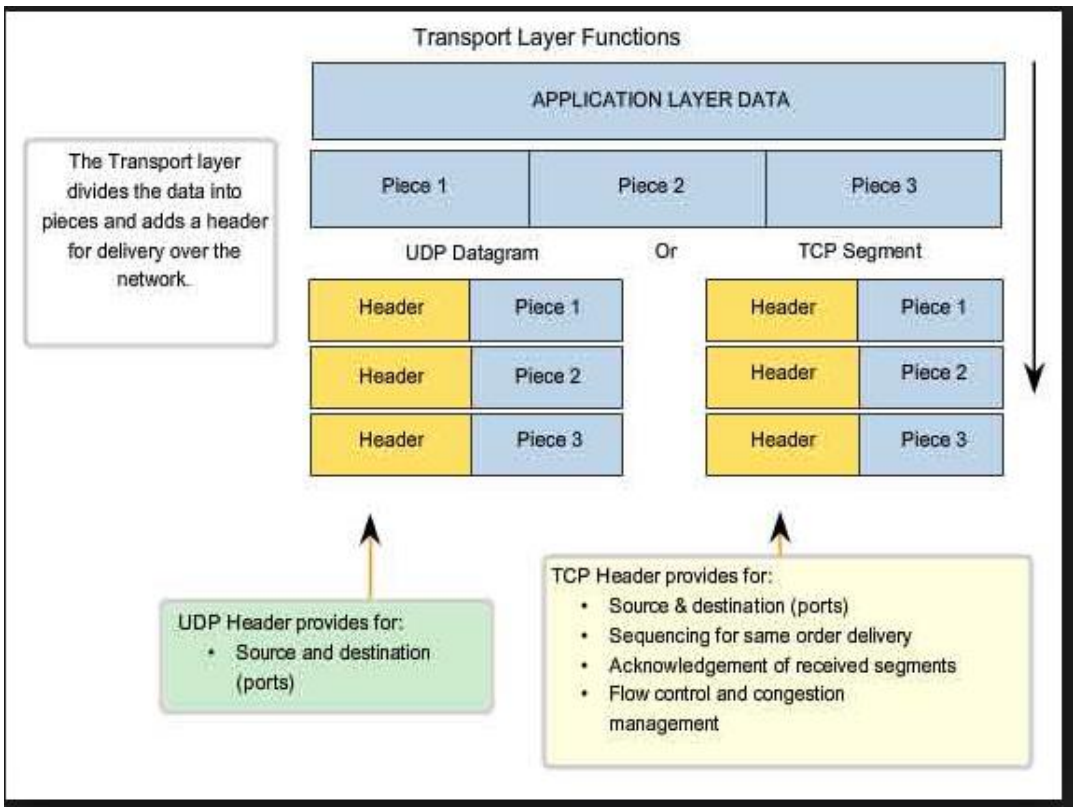
- Tuy nhiên, một số ý kiến lại cho rằng mô hình TCP/IP là 5 tầng, tức các tầng 4 đến 2 đều được giữ nguyên, nhưng tầng Datalink sẽ được tách riêng và là tầng nằm trên so với tầng vật lý.

- **Tầng 4 - Tầng Ứng dụng (Application):** Đây là lớp giao tiếp trên cùng của mô hình. Đúng với tên gọi, tầng Ứng dụng đảm nhận vai trò giao tiếp dữ liệu giữa 2 máy khác nhau thông qua các dịch vụ mạng khác nhau (duyệt web, chat, gửi email, một số giao thức trao đổi dữ liệu: SMTP, SSH, FTP,...). Dữ liệu khi đến đây sẽ được định dạng theo kiểu Byte nối Byte, cùng với đó là các thông tin định tuyến giúp xác định đường đi đúng của một gói tin.



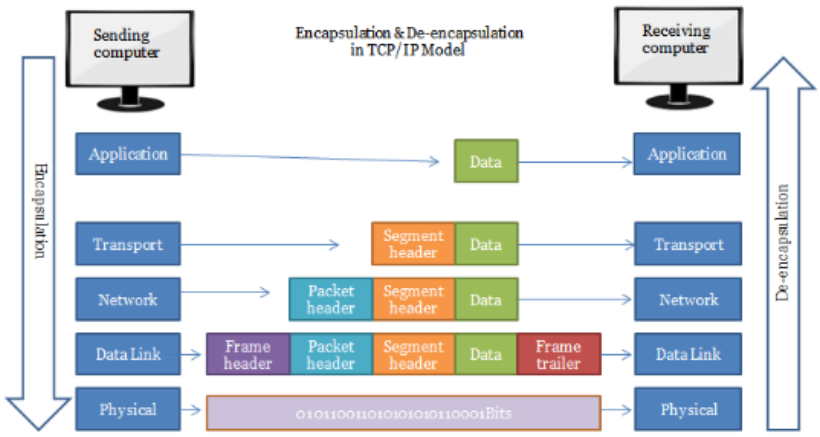
Hình 1.8: Tầng ứng dụng (Aplication)

- **Tầng 3 - Tầng Giao vận (Transport):** Chức năng chính của tầng 3 là xử lý vấn đề giao tiếp giữa các máy chủ trong cùng một mạng hoặc khác mạng được kết nối với nhau thông qua bộ định tuyến. Tại đây dữ liệu sẽ được phân đoạn, mỗi đoạn sẽ không bằng nhau nhưng kích thước phải nhỏ hơn 64KB. Cấu trúc đầy đủ của một Segment lúc này là Header chứa thông tin điều khiển và sau đó là dữ liệu. Trong tầng này còn bao gồm 2 giao thức cốt lõi là TCP và UDP. Trong đó, TCP đảm bảo chất lượng gói tin nhưng tiêu tốn thời gian khá lâu để kiểm tra đầy đủ thông tin từ thứ tự dữ liệu cho đến việc kiểm soát vấn đề tắc nghẽn lưu lượng dữ liệu. Trái với điều đó, UDP cho thấy tốc độ truyền tải nhanh hơn nhưng lại không đảm bảo được chất lượng dữ liệu được gửi đi.



Hình 1.9: Tầng giao vận (Transport)

- **Tầng 2 - Tầng mạng (Internet):** Gần giống như tầng mạng của mô hình OSI. Tại đây, nó cũng được định nghĩa là một giao thức chịu trách nhiệm truyền tải dữ liệu một cách logic trong mạng. Các phân đoạn dữ liệu sẽ được đóng gói (Packets) với kích thước mỗi gói phù hợp với mạng chuyển mạch mà nó dùng để truyền dữ liệu. Lúc này, các gói tin được chèn thêm phần Header chứa thông tin của tầng mạng và tiếp tục được chuyển đến tầng tiếp theo. Các giao thức chính trong tầng là IP, ICMP và ARP.



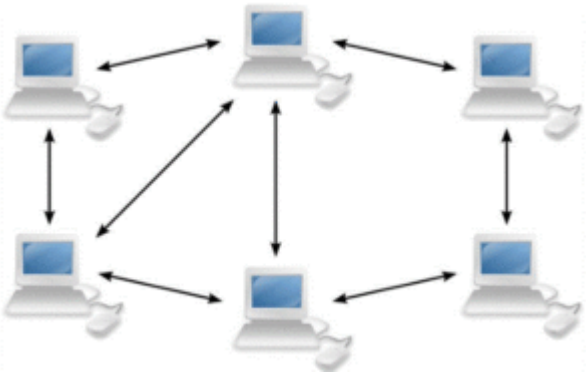
Hình 1.10: Tầng mạng (Internet)

- **Tầng 1 - Tầng Vật lý (Physical):** Là sự kết hợp giữa tầng Vật lý và tầng liên kết dữ liệu của mô hình OSI. Chịu trách nhiệm truyền dữ liệu giữa hai thiết bị trong cùng một mạng. Tại đây, các gói dữ liệu được đóng vào khung (gọi là Frame) và được định tuyến đi đến đích đã được chỉ định ban đầu.

1.2.1.5. Các mô hình hoạt động của mạng máy tính

❖ **Mô hình hoạt động peer to peer**

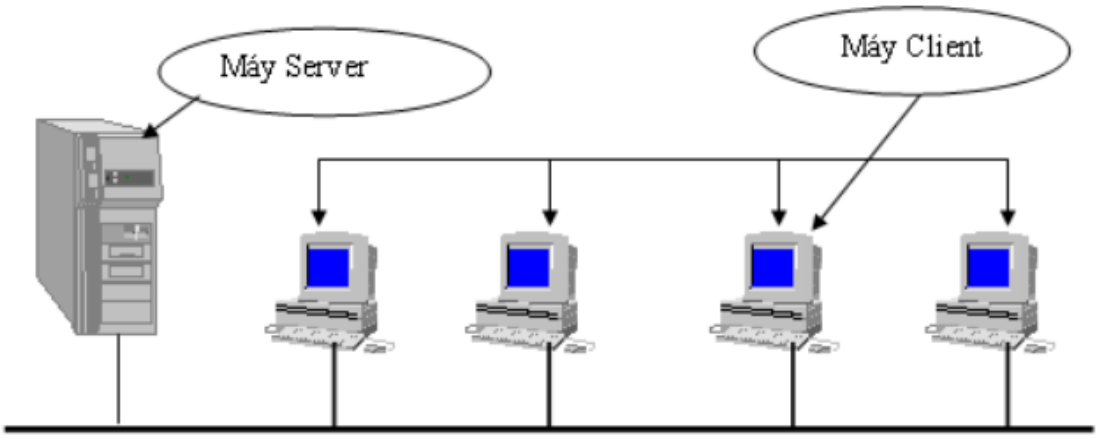
Trong mạng hoạt động theo mô hình Clients/Server có một hoặc nhiều máy có nhiệm vụ cung cấp một số dịch vụ cho các máy khác ở trong mạng. Các máy này được gọi là server còn các máy tính được phục vụ gọi là máy clients.



Hình 1.11: Mô hình hoạt động P2P

❖ **Mô hình hoạt động clients/ server**

Trong mạng hoạt động theo mô hình Clients/Server có một hoặc nhiều máy có nhiệm vụ cung cấp một số dịch vụ cho các máy khác ở trong mạng. Các máy này được gọi là server còn các máy tính được phục vụ gọi là máy clients.



Hình 1.12: Mô hình hoạt động clients/ server

Đây là mô hình tổng quát, trên thực tế server có thể được nối với nhiều server khác để tăng hiệu quả làm việc. Khi nhận được yêu cầu từ client, server có thể xử lý yêu cầu đó hoặc gửi tiếp yêu cầu vừa nhận được cho một server khác. Máy server sẽ thi hành các nhiệm vụ do máy client yêu cầu. Có rất nhiều dịch vụ trên mạng hoạt động theo nguyên lý nhận các yêu cầu từ client sau đó xử lý và trả lại các kết quả cho client yêu cầu.

1.2.2. Tổng quan về ngôn ngữ lập trình Java

1.2.2.1. Giới thiệu về ngôn ngữ Java

Java là một ngôn ngữ lập trình được Sun Microsystems giới thiệu vào tháng 6 năm 1995. Từ đó, nó đã trở thành một công cụ lập trình của các lập trình viên chuyên nghiệp. Java được xây dựng trên nền tảng của C và C++. Do vậy nó sử dụng các cú pháp của C và các đặc trưng hướng đối tượng của C++. Vào năm 1991, một nhóm các kỹ sư của Sun Microsystems có ý định thiết kế một ngôn ngữ lập trình để điều khiển các thiết bị điện tử như Tivi, máy giặt, lò nướng, … Mặc dù C và C++ có khả năng làm việc này nhưng trình biên dịch lại phụ thuộc vào từng loại CPU. Trình biên dịch thường phải tốn nhiều thời gian để xây dựng nên rất đắt. Vì vậy để mỗi loại CPU có một trình biên dịch riêng là rất tốn kém. Do đó nhu cầu thực tế đòi hỏi một ngôn ngữ chạy nhanh, gọn, hiệu quả và độc lập thiết bị tức là có thể chạy trên nhiều loại CPU khác nhau, dưới các môi trường khác nhau. “Oak” đã ra đời và vào năm 1995 được đổi tên thành Java. Mặc dù mục tiêu ban đầu không phải cho Internet nhưng do đặc trưng không phụ thuộc thiết bị nên Java đã trở thành ngôn ngữ lập trình cho Internet.

1.2.2.2. Một số tính chất của ngôn ngữ Java

❖ **Đơn giản**

Những người thiết kế mong muốn phát triển một ngôn ngữ dễ học và quen thuộc với đa số người lập trình. Do vậy Java loại bỏ các đặc trưng phức tạp của C và C++ như:

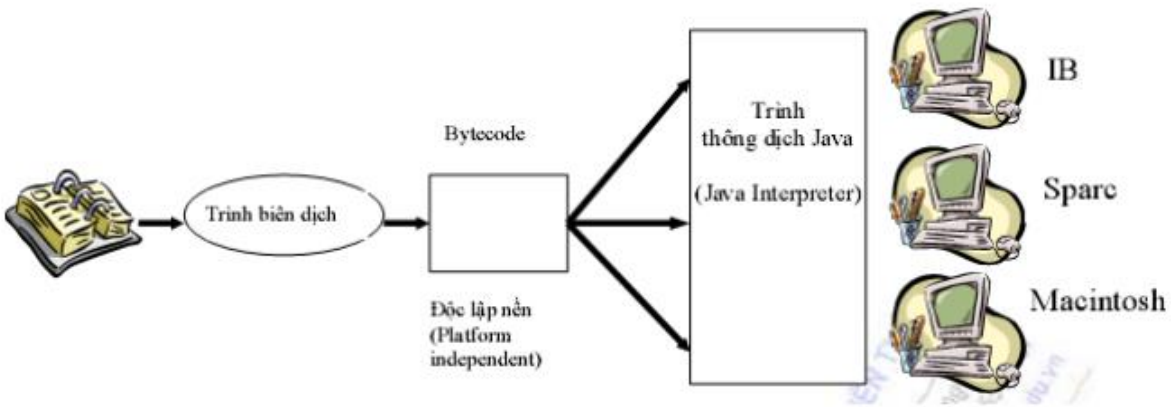
- Loại bỏ thao tác con trỏ, thao tác định nghĩa chồng toán tử
- Không cho phép đa kế thừa mà sử dụng các giao diện
- Không sử dụng lệnh "goto" cũng như file header (.h)
- Loại bỏ cấu trúc "struct" và "union"

❖ **Hướng đối tượng**

Java là ngôn ngữ lập trình thuần hướng đối tượng. Mọi chương trình viết trên Java đều phải được xây dựng trên các đối tượng. Nếu trong C/ C++ ta có thể tạo ra các hàm (không gắn với đối tượng nào) thì trong Java ta chỉ có thể tạo ra các hàm (phương thức) gắn liền với một đối tượng. Trong Java không cho phép các đối tượng có tính năng đa kế thừa mà thay bằng các giao tiếp

❖ **Độc lập phần cứng và hệ điều hành**

Với mỗi nền phần cứng khác nhau, có một trình biên dịch khác nhau để biên dịch mã nguồn chương trình cho phù hợp với nền phần cứng ấy. Do vậy, khi chạy trên một nền phần cứng khác bắt buộc phải biên dịch lại mã nguồn Đối với các chương trình viết bằng Java, trình biên dịch Java sẽ biên dịch mã nguồn thành dạng bytecode. Sau đó, khi chạy chương trình trên các nền phần cứng khác nhau, máy ảo Java dùng trình thông dịch Java để chuyển mã bytecode thành dạng chạy được trên các nền phần cứng tương ứng. Do vậy, khi thay đổi nền phần cứng, không phải biên dịch lại mã nguồn Java.



Hình 1.13: Dịch chương trình Java

❖ **Mạnh**

Java là ngôn ngữ yêu cầu chặt chẽ về kiểu dữ liệu.

- Kiểu dữ liệu phải khai báo tường minh.

- Java không sử dụng con trỏ và các phép toán con trỏ.

- Java kiểm tra tất cả các truy nhập đến mảng, chuỗi khi thực thi để đảm bảo rằng các truy nhập đó không ra ngoài giới hạn kích thước

- Trong các môi trường lập trình truyền thống, lập trình viên phải tự mình cấp phát bộ nhớ. Trước khi chương trình kết thúc thì phải tự giải phóng bộ nhớ đã cấp. Vấn đề nảy sinh khi lập trình viên quên giải phóng bộ nhớ đã xin cấp trước đó. Trong chương trình java, lập trình viên không phải bận tâm đến việc cấp phát bộ nhớ. Quá trình cấp phát, giải phóng được thực hiện tự động, nhờ dịch vụ thu nhặt những đối tượng không còn sử dụng nữa (garbage collection).

- Cơ chế bẫy lỗi của java giúp đơn giản hóa quá trình xử lý lỗi và hồi phục sau lỗi

❖ Bảo mật

Java cung cấp một môi trường quản lý thực thi chương trình với nhiều mức để kiểm soát tính an toàn:

- Ở mức thứ nhất, dữ liệu và các phương thức được đóng gói bên trong lớp. Chúng chỉ được truy xuất thông qua các giao diện mà lớp cung cấp

- Ở mức thứ hai, trình biên dịch kiểm soát để đảm bảo mã là an toàn, và tuân theo các nguyên tắc của java

- Mức thứ ba được đảm bảo bởi trình thông dịch. Chúng kiểm soát xem bytecode có đảm các quy tắc an toàn trước khi thực thi

- Mức thứ tư kiểm soát việc nạp các lớp vào bộ nhớ để giám sát việc vi phạm giới hạn truy xuất trước khi nạp vào hệ thống.

❖ Phân tán

Java được thiết kế để hỗ trợ các ứng dụng chạy trên mạng bằng các lớp mạng (java.net). Hơn nữa, java hỗ trợ nhiều nền chạy khác nhau nên chúng được sử dụng rộng rãi như là công cụ phát triển trên Internet, nơi sử dụng nhiều nền khác nhau

❖ Đa luồng

Chương trình java cung cấp giải pháp đa luồng(Multithreading) để thực thi các công việc đồng thời. Chúng cũng cung cấp giải pháp đồng bộ giữa các luồng. Đặc tính hỗ trợ đa luồng này cho phép xây dựng các ứng dụng trên mạng chạy hiệu quả

❖ Động

Java được thiết kế như một ngôn ngữ động để đáp ứng cho những môi trường mở. Các chương trình Java chứa rất nhiều thông tin thực thi nhằm kiểm soát và truy nhập đối tượng lúc chạy. Điều này cho phép khả năng liên kết động mã.

1.2.2.3. Cấu trúc của tệp chương trình Java

Tệp chương trình java có thể có các phần được đặc tả như sau:

- Định nghĩa một gói là tùy chọn thông qua định danh của gói (package). Tất cả các lớp, các interface được định nghĩa trong tệp chứa gói này đều thuộc gói đó. Nếu bỏ qua định nghĩa gói thì các định nghĩa ở tệp này sẽ thuộc vào gói mặc định.

- Một số lệnh nhập import

- Một số định nghĩa lớp và interface có thể định nghĩa theo thứ tự bất kỳ. Trong đó thường là lớp public

Như vậy, cấu trúc của một tệp chương trình Java có thể khái quát như sau:

```
// Filename: New.java
// Phần 1: tùy chọn
// Định nghĩa gói
package Têngói;
// Phần 2: 0 hoặc nhiều hơn
// các gói cần sử dụng
import java.io.*;
// Phần 3: 0 hoặc nhiều hơn
// Định nghĩa các lớp và các interface
public class New{...}
class C1 {...}
interface I1 {...}
// ...
class Cn {...}
interface Im {...}
```

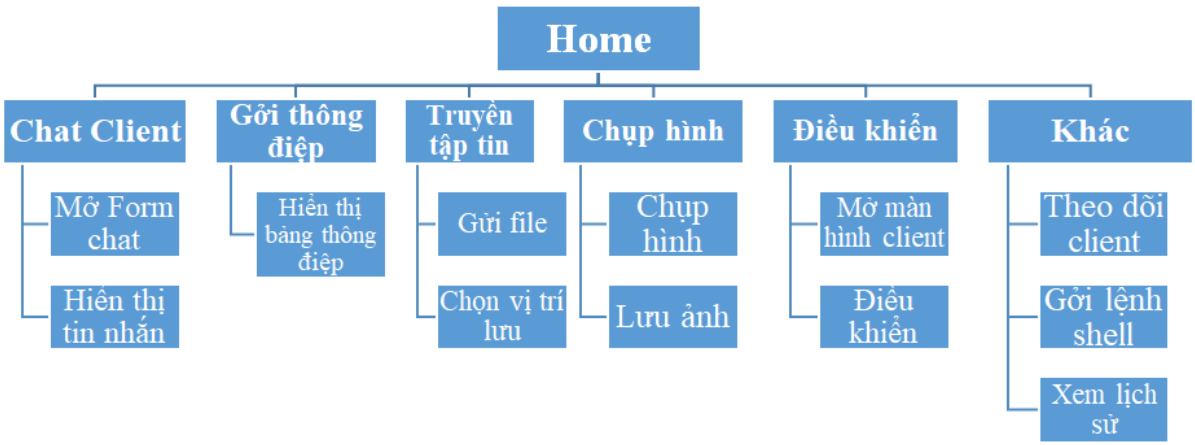
CHƯƠNG 2: PHÂN TÍCH & THIẾT KẾ HỆ THỐNG

2.1. Giới thiệu

Trao đổi dữ liệu giữa hai máy tính trong mạng thực chất là sự trao đổi dữ liệu giữa hai chương trình ứng dụng chạy trên hai máy tính đó. Trong đó, một chương trình được gán nhãn là server và một chương trình được gán nhãn là client, có nhiều phương pháp để xây dựng chương trình ứng dụng mạng nhưng phương pháp sử dụng phổ biến là lập trình ứng mạng dựa trên cơ chế socket. Trong chương này sẽ trình bày một ứng dụng của lập trình socket TCP là xây dựng chương trình quản lý mạng LAN giữa nhiều máy tính bằng Java Socket TCP.

2.2. Sơ đồ phân cấp chức năng

2.2.1. Sơ đồ chức năng



Hình 2.1: Sơ đồ phân cấp chức năng

2.2.2. Chi tiết chức năng

1. Chat Client: Mở form chat của máy chủ server, sau ghi máy chủ gửi tin nhắn thì sẽ hiện 1 form chat bên phía client, từ đấy 2 bên có thể tiến hành trao đổi tin nhắn với nhau.

2. Gởi thông điệp: Hiển thị form gửi thông điệp, máy chủ tạo thông điệp và gửi tới cho máy client, bên phía client không thể phản hồi tin nhắn lại cho máy server.

3. Truyền tập tin: Bên máy server chọn file để gửi cho máy client, bắt đàu truyền file, máy client sẽ tiến hành chọn chỗ lưu file và sau đó sẽ bắt đầu lưu.

4. Chụp hình: Hiển thị màn hình của máy client được chọn, có thể tiến hành chụp và chọn lưu hình hay không, hoặc tiến hành thoát.

5. Điều khiển: Hiển thị màn hình máy client mà con trỏ chuột, người dùng máy server có thể tiến hành điều khiển máy client.

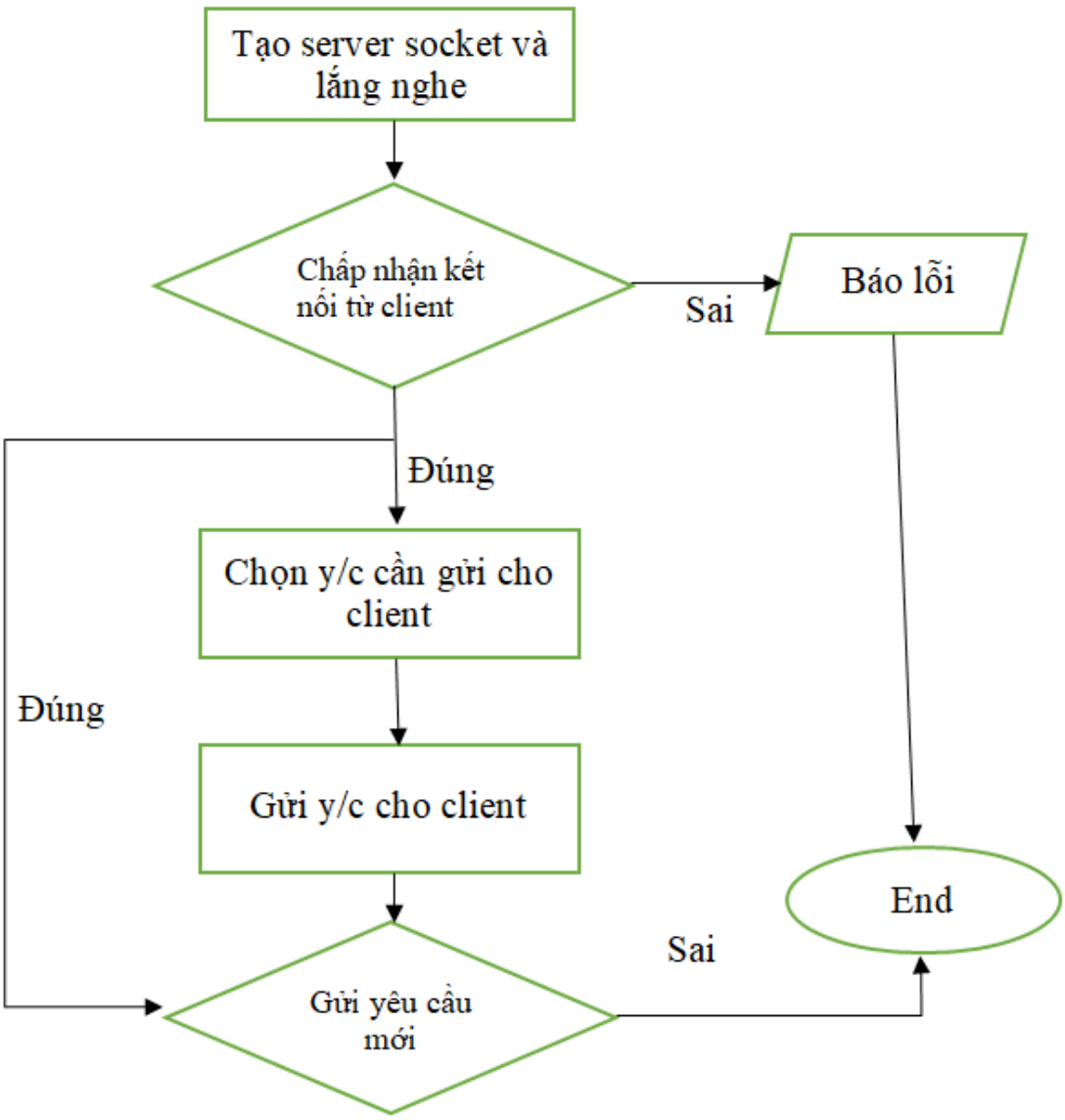
6. Khác: Có thể tiến hành các thao tác khác nhau như

- Theo dõi client
- Gởi lệnh Shell
- Xem lịch sử trình duyệt của client

2.3. Phân tích chương trình

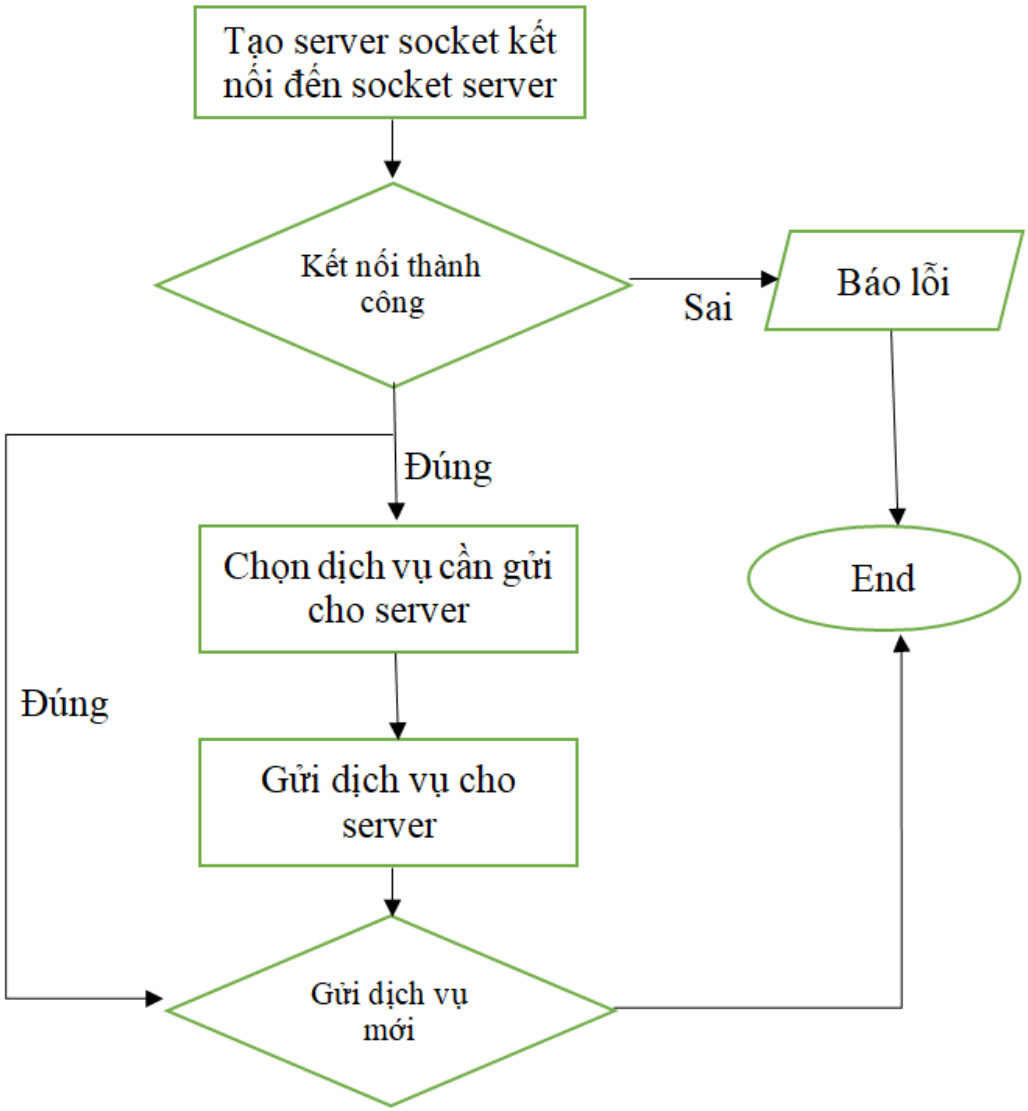
Chương trình ứng dụng được xây dựng theo mô hình clients/server. Chương trình bao gồm hai mô đun server và client. Người sử dụng có thể truyền yêu cầu từ phía client cho server hoặc ngược lại.

Mô đun phía server



Hình 2.2: Sơ đồ modul phía server

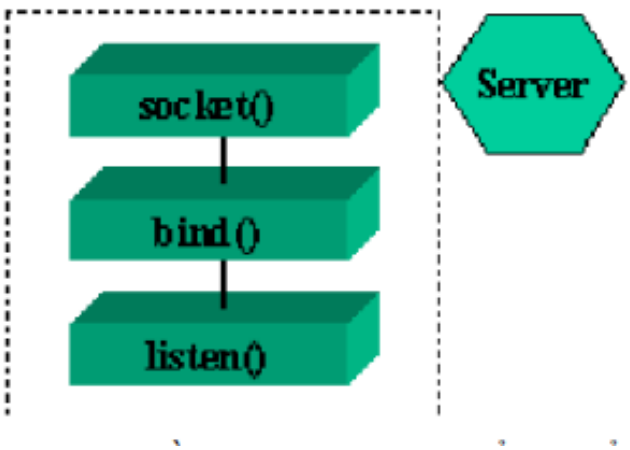
Mô đun phía client



Hình 2.3: Sơ đồ modul phía client

2.4. Phân tích giai đoạn hoạt động

Giai đoạn 1: Server tạo socket, gán số hiệu cổng và lắng nghe yêu cầu kết nối.



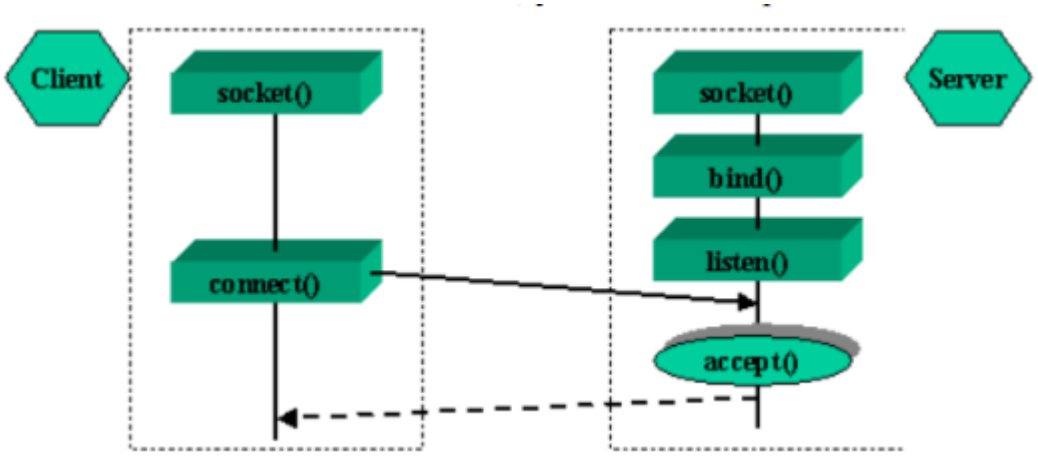
Hình 2.4: Giai đoạn 1 – Server tạo socket

Socket(): Server yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng vận chuyển.

Bind(): Server yêu cầu gán số hiệu cổng (port) cho socket.

Listen(): Server lắng nghe các yêu cầu nối kết từ các client trên cổng đã được gán

Giai đoạn 2: Client tạo socket, yêu cầu thiết lập một nối kết với Server.



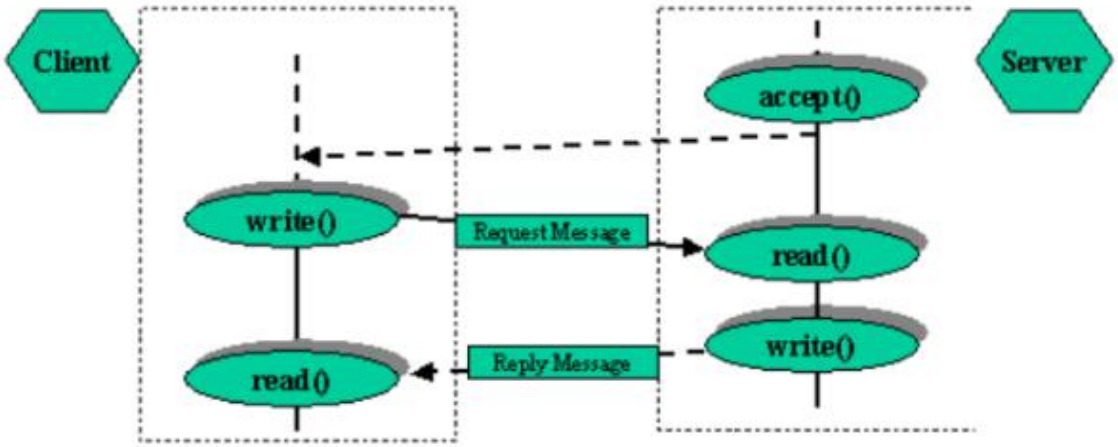
Hình 2.5: Giai đoạn 2 – Client tạo socket

Socket(): Client yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng vận chuyển, thông thường hệ thống tự động gán một số hiệu cổng chưa sử dụng cho socket của Client.

Connect(): Client gửi yêu cầu nối kết đến server có địa chỉ IP và Port xác định.

Accept(): Server chấp nhận nối kết của client, khi đó một kênh giao tiếp ảo được hình thành, client và server có thể trao đổi thông tin với nhau thông qua kênh ảo này

Giai đoạn 3: Trao đổi thông tin giữa Client và Server



Hình 2.6: Trao đổi thông tin giữa Client và Server

Sau khi chấp nhận yêu cầu nối kết, thông thường server thực hiện lệnh read() và nghẽ cho đến khi có thông điệp yêu cầu(Reaquest Message) từ client gửi đến.

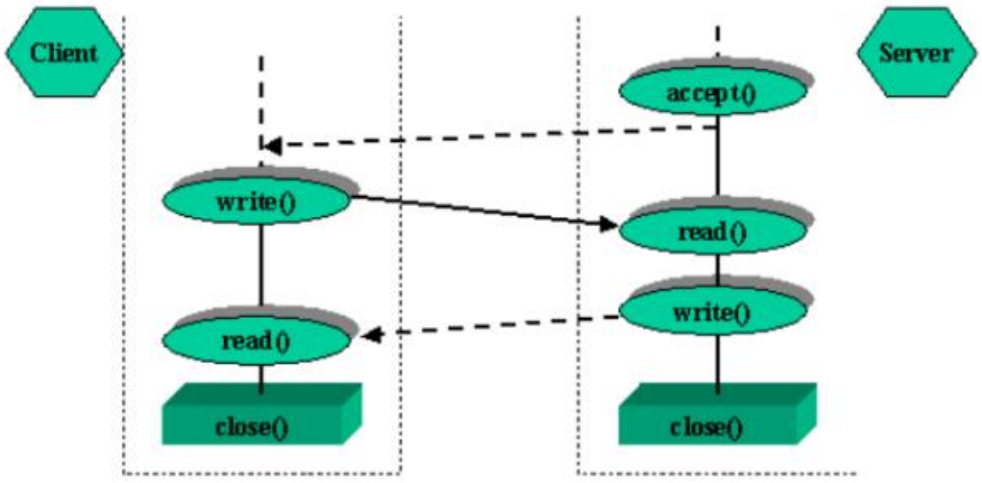
Server phân tích và thực thi yêu cầu. Kết quả sẽ được gửi về client bằng lệnh write().

Sau khi gửi yêu cầu bằng lệnh write(), client chờ nhận thông điệp kết quả (Reply Message) từ server bằng lệnh read().

Trong giai đoạn này, việc trao đổi thông tin giữa client và server phải tuân thủ giao thức của ứng dụng (Dạng thức và ý nghĩa các thông điệp, quy tắc bắt tay, đồng bộ hóa…). Thông thường client sẽ là người gửi yêu cầu đến server trước.

Nếu chúng ta phát triển ứng dụng theo các protocol đã định nghĩa sẵn, chúng ta phải tham khảo và tuân thủ đúng những quy định của giao thức. Ngược lại, nếu chúng ta phát triển một ứng dụng clients/server riêng của mình, thì công việc đầu tiên chúng ta phải thực hiện là đi xây dựng protocol cho ứng dụng.

Giai đoạn 4: Kết thúc phiên làm việc

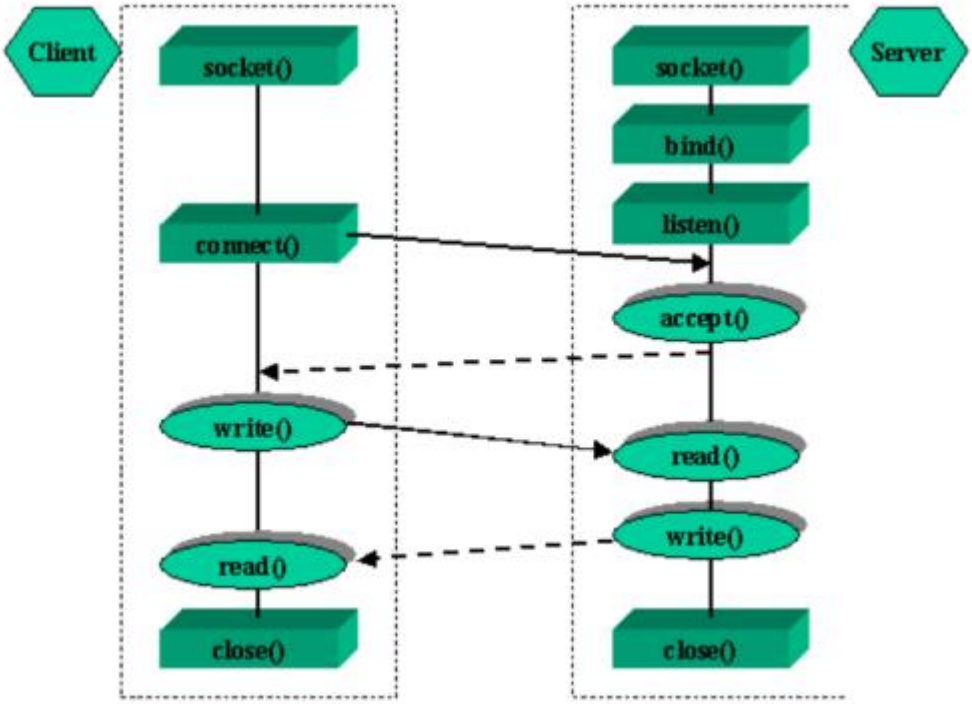


Hình 2.7: Kết thúc phiên làm việc

Các câu lệnh read(), write() có thể được thực hiện nhiều lần(ký hiệu bằng hình ellipse).

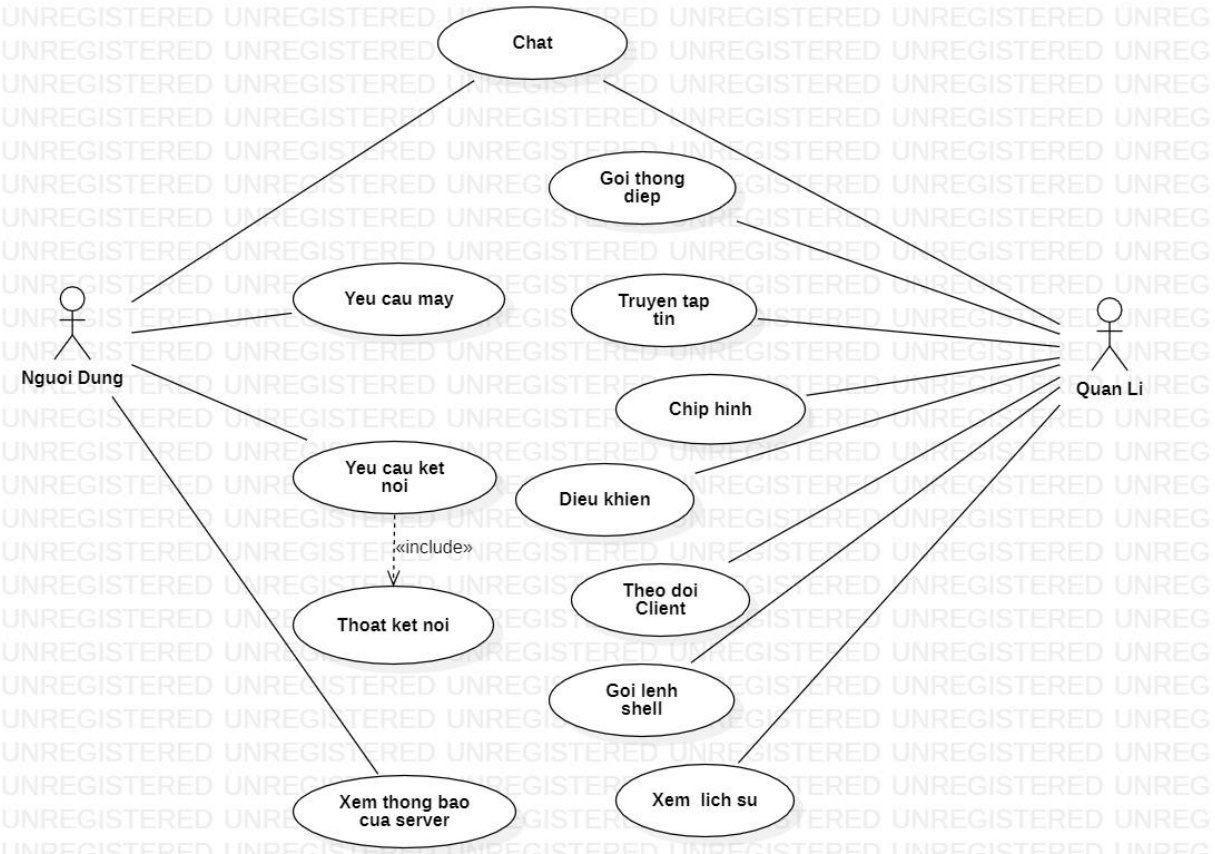
Kênh ảo sẽ bị xóa khi server hoặc client đóng socket bằng lệnh close().

Như vậy toàn bộ tiến trình diễn ra như sau:



Hình 2.8: Toàn bộ tiến trình hoạt động

2.5. Biểu đồ Use



Hình 2.4: Sơ đồ Usecase

Đặc tả UseCase Nguoi Dung

Bảng 2.1: Đặc tả UseCase người dùng

Tác nhân	Người dùng
Mô tả: tác nhân sử dụng usecase này để thực hiện chức năng của chương trình như kết nối với máy server, tiến hành chat để trao đổi với máy server, hay thoát khỏi chương trình.	
Dòng sự kiện chính: 1. Dòng thứ 1: Tác nhân chọn kết nối với máy server 2. Dòng thứ 2: Tác nhân có thể tiến hành thoát khỏi kết nối với máy server nếu muốn.	
Dòng sự kiện phụ: 1. Dòng thứ 1: Tác nhân tiến hành chat trao đổi với với máy server 2. Dòng thứ 2: Xem thông báo do máy server gửi tới	
Các yêu cầu đặc biệt:	Phải kết nối chung một mạng LAN
➢ **Trạng thái hệ thống trước khi UseCase được sử dụng:** Chưa có gì xảy ra ➢ **Trạng thái hệ thống sau khi UseCase được sử dụng:** Server có thể thực hiện các dịch vụ có trong hệ thống và tương tác với clients **Nếu thành công:** Tiến hành tương tác. **Nếu thất bại:** Clients kết nối thất bại	
Điểm mở rộng	
Tần suất sử dụng	Thường xuyên

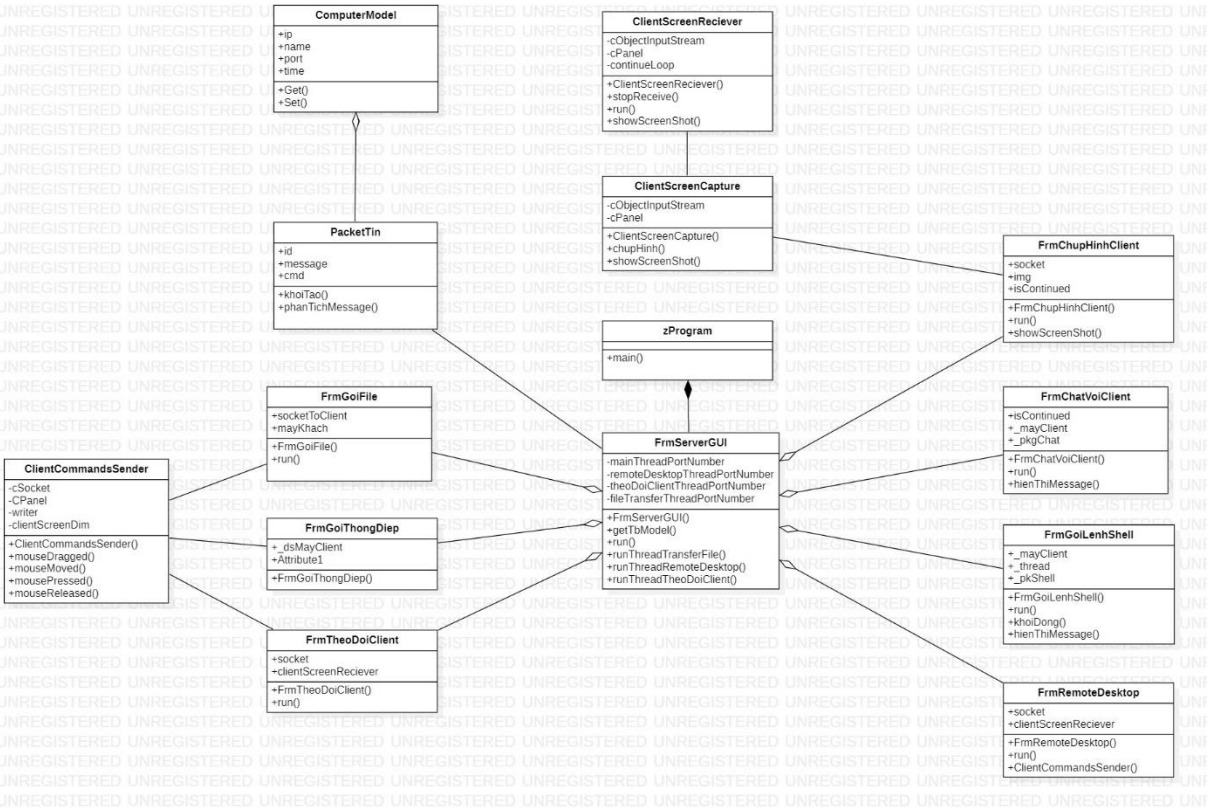
Đặc tả UseCase Quan Li

Bảng 2.2: Đặc tả UseCase quản lí

Tác nhân	Quản lí
Mô tả: Quản lí có thể sử dụng các dịch vụ để tương tác với người dùng khi người dùng đăng nhập thành công như truyền file, điều khiển clients, gửi lệnh shell, chat…	
Dòng sự kiện chính: 1. Dòng thứ 1: Tác nhân xem danh sách và có thể chọn máy client cần thao tác 2. Tác nhân có thể tiến hành các dịch vụ có trong hệ thống như truyền file, điều khiển clients, gửi lệnh shell, chat…	
Dòng sự kiện phụ: Không có	

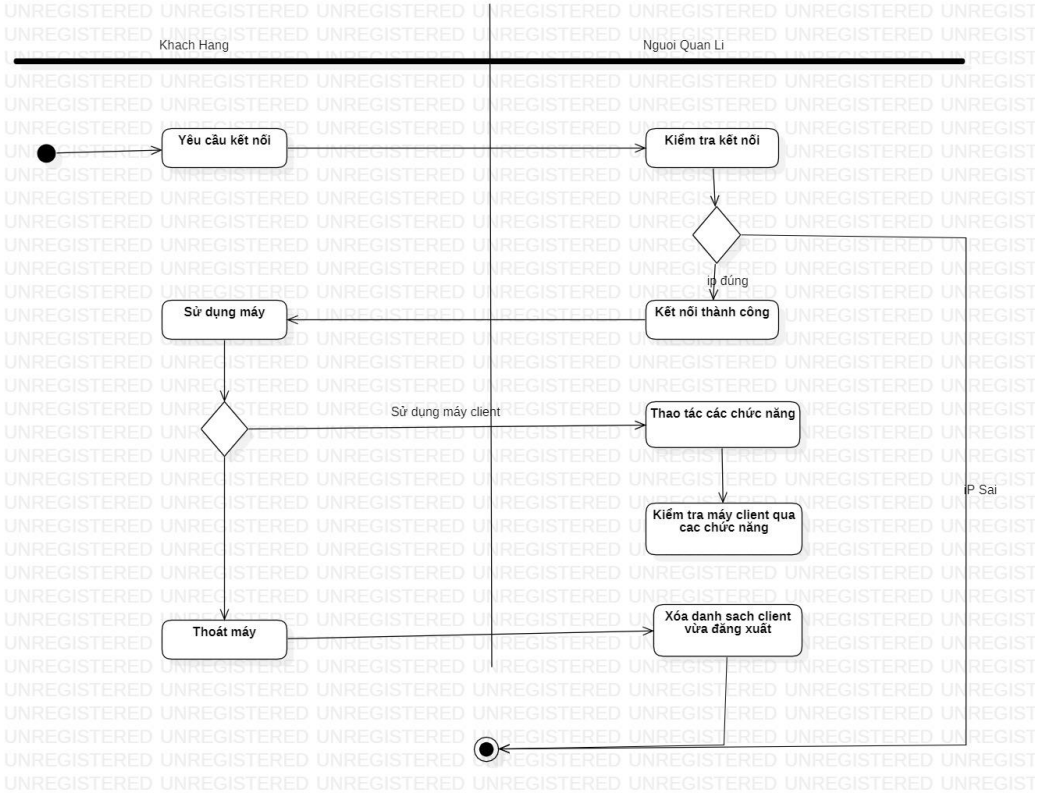
Các yêu cầu đặc biệt:	Clients phải kết nối với Server trong cùng một mạng LAN
➢ **Trạng thái hệ thống trước khi UseCase được sử dụng:** Chưa có gì xảy ra ➢ **Trạng thái hệ thống sau khi UseCase được sử dụng:** Quản lí có thể sử dụng các dịch vụ của hệ thống **Nếu thành công:** Sử dụng hệ thống. **Nếu thất bại:** Clients không kết nối được với server	
Điểm mở rộng	
Tần suất sử dụng	Thường xuyên

2.6. Biểu đồ lớp



Hình 2.5: Sơ đồ Class Diagram

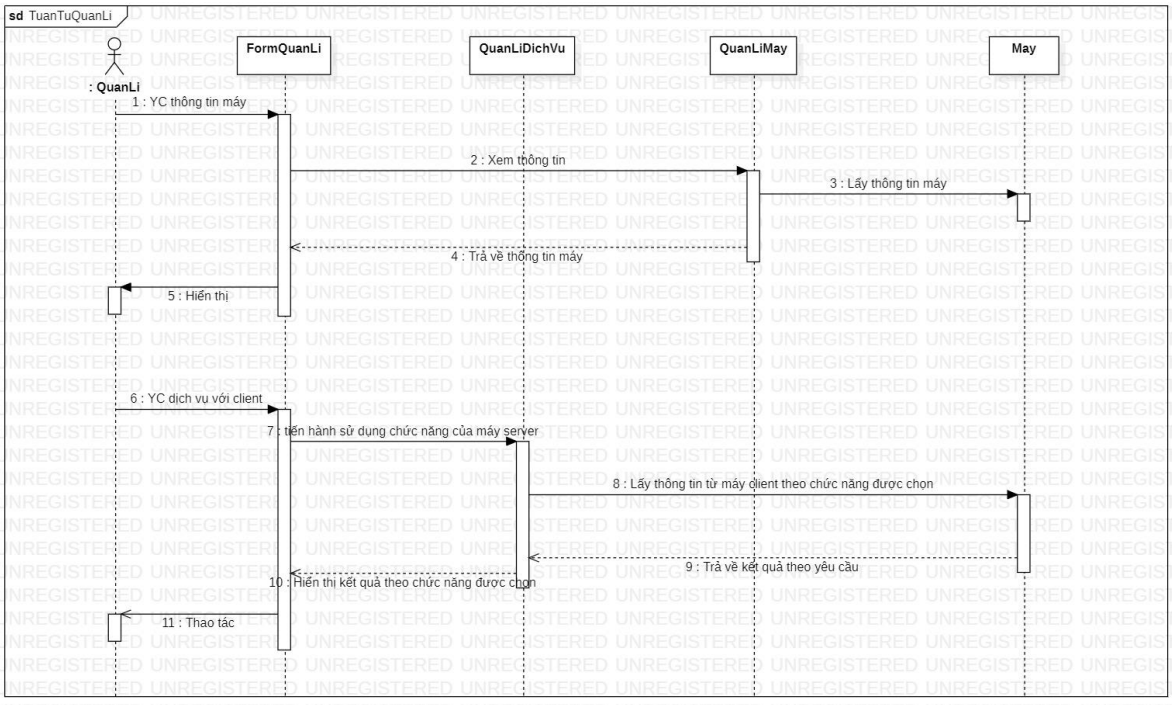
2.7. Biểu đồ hoạt động



Hình 2.6: Sơ đồ Activity Diagram

2.8. Biểu đồ tuần tự

Biểu đồ tuần tự của người quản lí máy server



Hình 2.7: Sơ đồ Sequence Diagram

CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

3.1. Cơ chế hoạt động

Chương trình gồm hai mô đun. Phía server là file chương trình có tên là server_da.jar, phía client là file chương trình có tên là client_da.jar.

Sau khi biên dịch file .jar này ta nhận được các file .class tương ứng.

Chương trình được thực thi tại dấu nhắc hệ thống theo cú pháp

c:\>java FileTransferServer

c:\>java FileTransferClient

Chạy chương trình ở server mode:

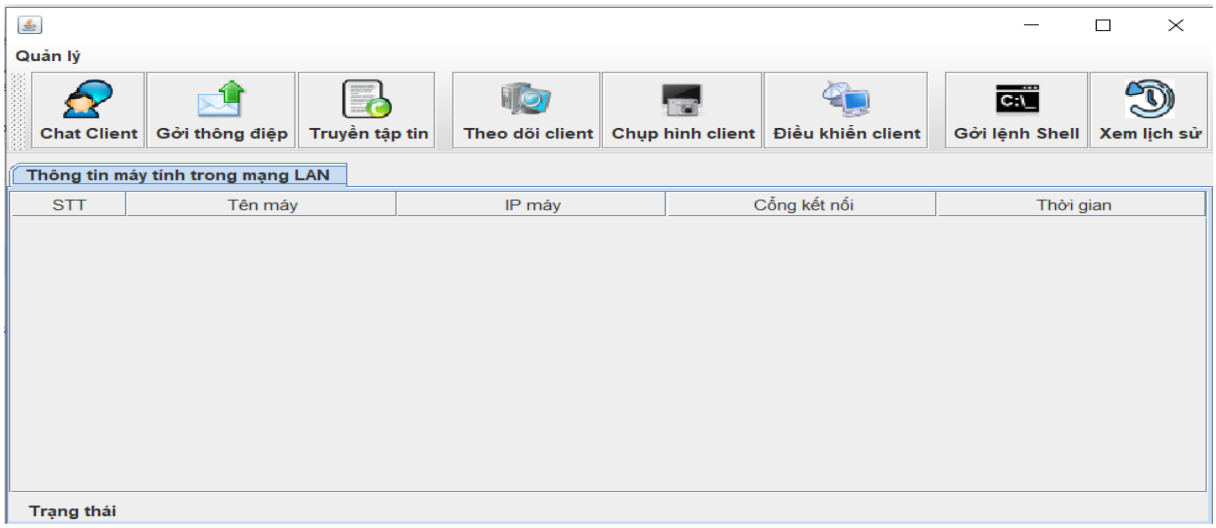
Chương trình chạy phía server đã được chỉ định chạy server mode, sau đó máy tính này sẽ chờ đợi các kết nối từ phía clients đến nó. Ta phải nhập tiếp port number, ứng với server mode này thì ta có thể chọn bất cứ port number nào. Chương trình này cũng có thể thực hiện các dịch vụ giữa một server và nhiều máy clients đồng thời trong một hệ thống mạng nên nó sẽ yêu cầu ta nhập vào số lượng máy client lớn nhất có thể kết nối đến server này.

Chạy chương trình ở client mode:

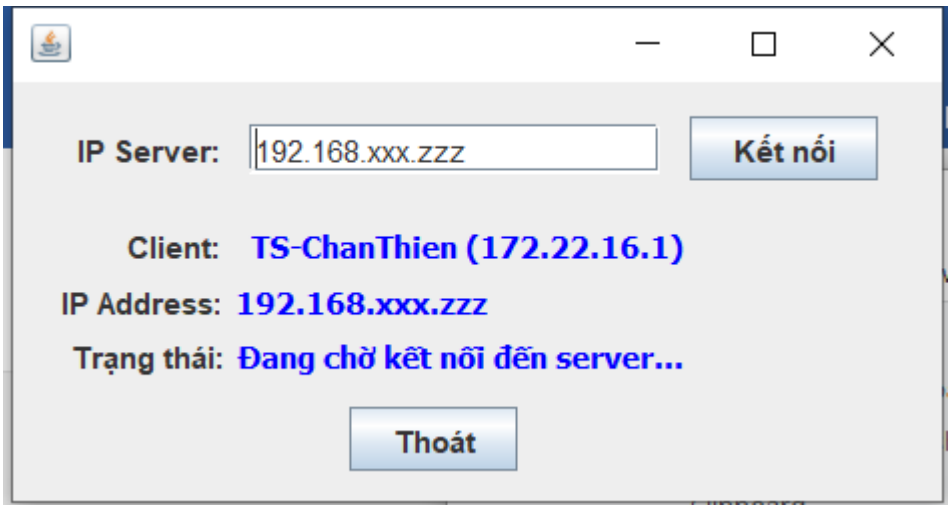
Tương tự cho phía bên client, chương trình sẽ yêu cầu nhập vào địa chỉ của server (host address), ta có thể nhập địa chỉ IP hay nhập vào tên của máy chạy server mode đều được. Tiếp tục ta sẽ nhập port number (số hiệu cổng) của server socket (đã biết) cần kết nối đến.

3.2. Giao diện chương trình

3.2.1. Giao diện phía server và client lúc chưa kết nối

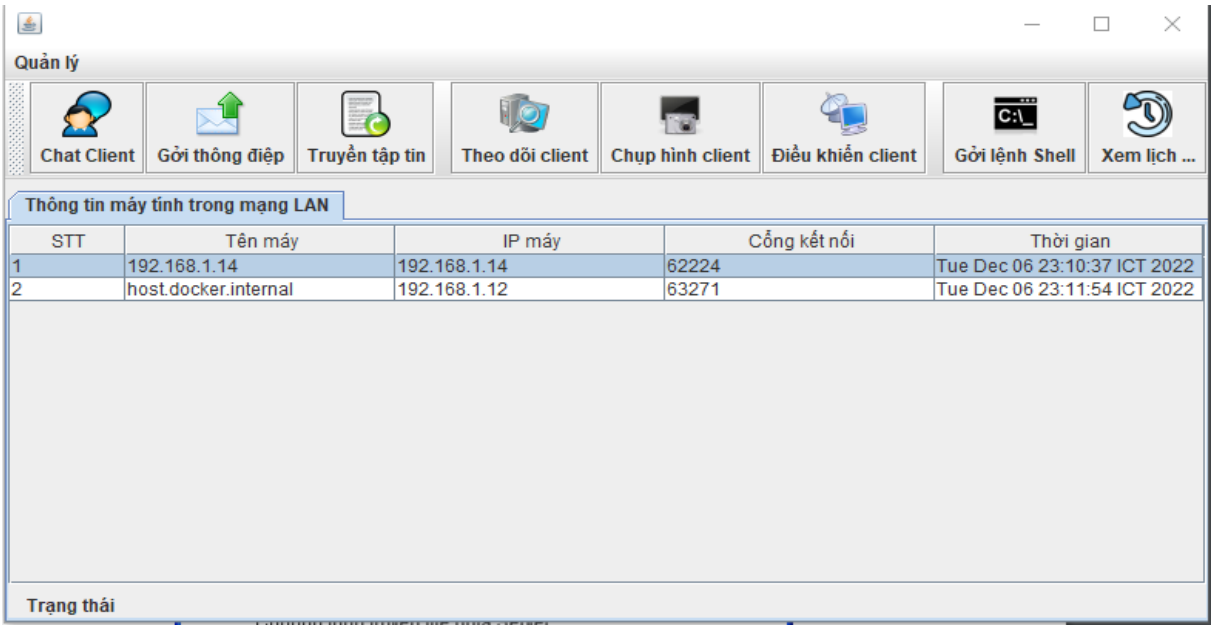


Hình 3.1: Máy server lúc chưa kết nối

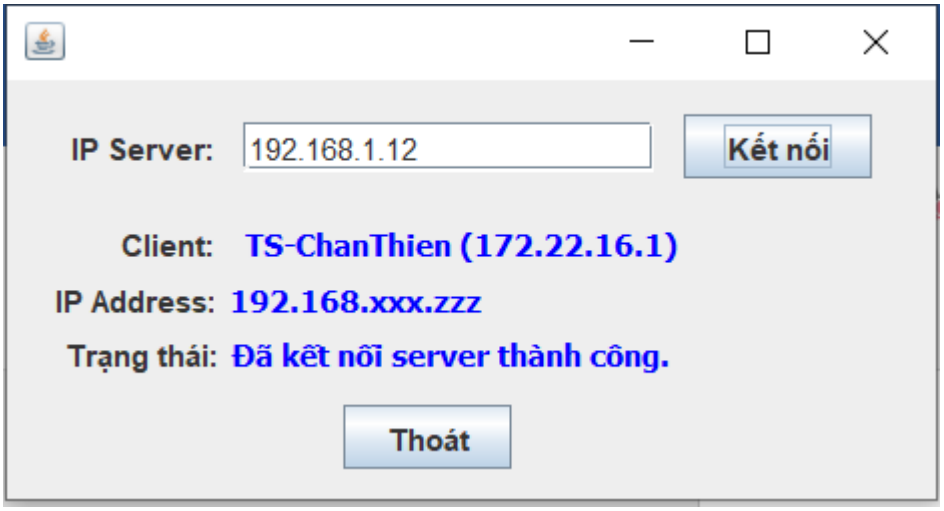


Hình 3.2: Máy client lúc chưa kết nối

3.2.2. Giao diện phía server và client lúc kết nối

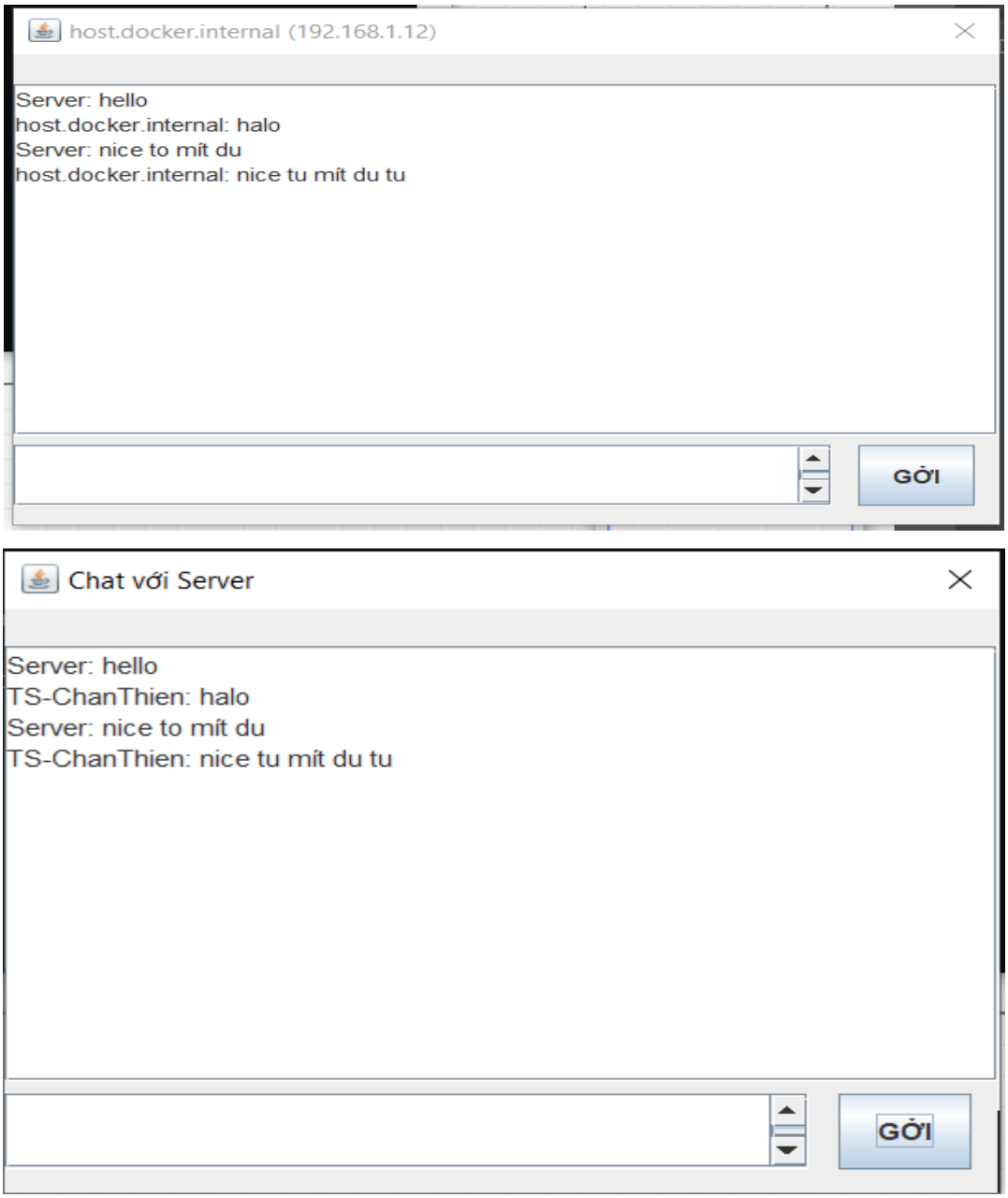


Hình 3.3: Máy server lúc kết nối



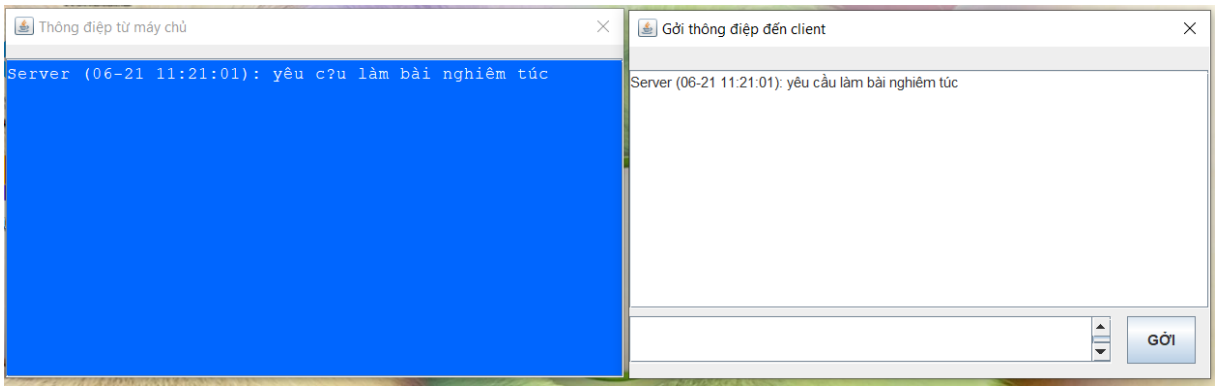
Hình 3.4: Máy Client lúc kết nối

3.2.3. Giao diện Chat Client



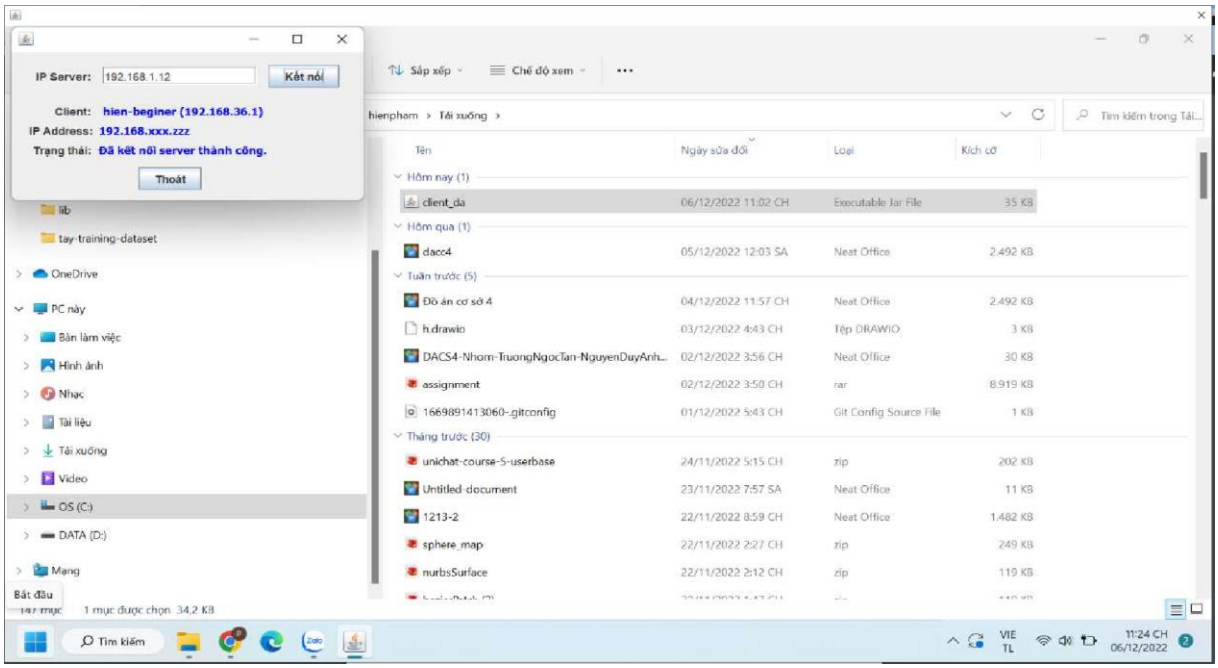
Hình 3.5: Giao diện chat giữa Server và Client

3.2.4. Giao diện gởi thông điệp

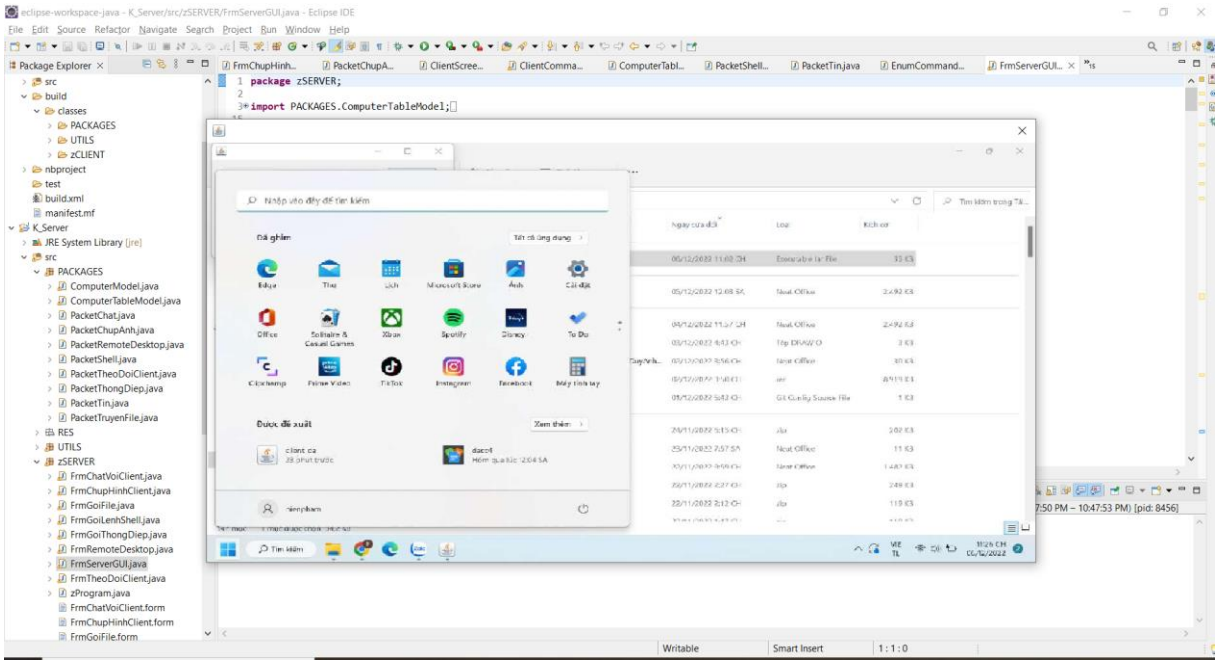


Hình 3.6: Giao diện gởi thông điệp

3.2.5. Giao diện điều khiển client

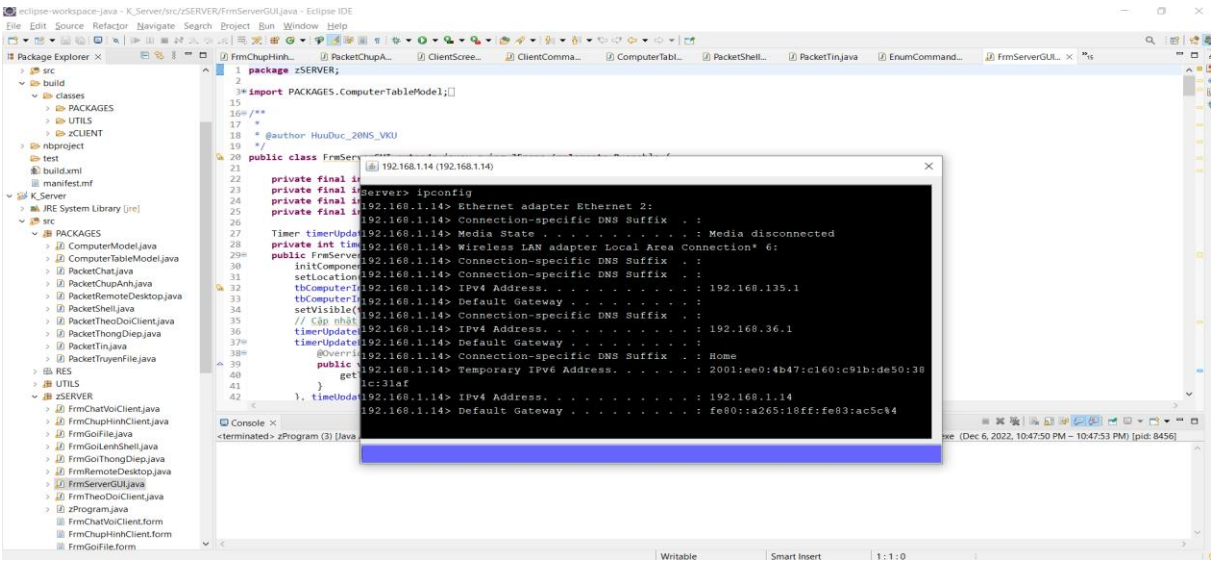


Hình 3.7: Giao diện điều khiển client1



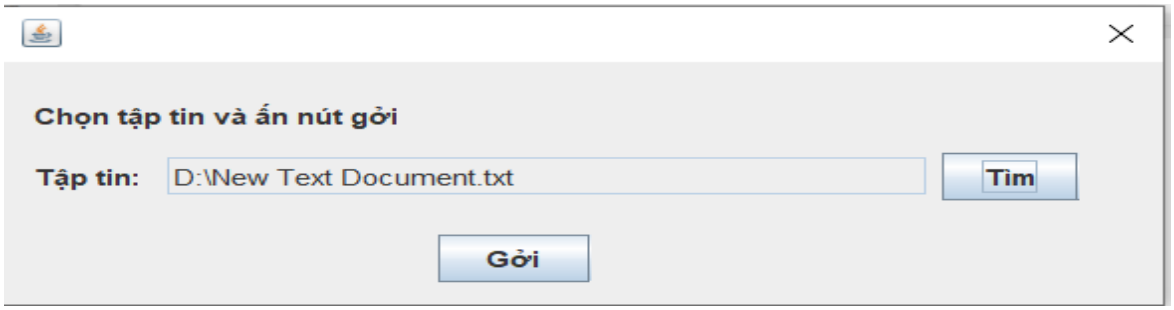
Hình 3.8: Giao diện điều khiển client2

3.2.6. Giao diện gởi lệnh shell

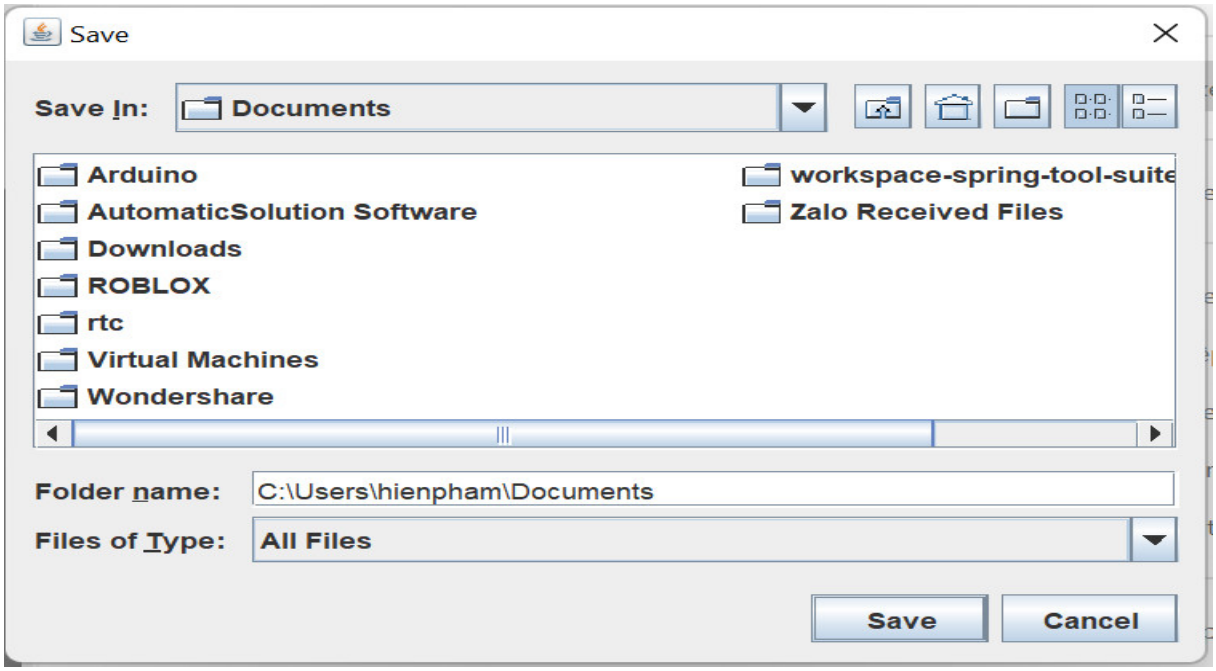


Hình 3.9: Giao diện gởi lệnh shell

3.2.7. Giao diện truyền tập tin



Hình 3.10: Giao diện truyền file phía server



Hình 3.11: Giao diện nhận file phía Client

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Nhận xét

Chương trình ứng dụng được lựa chọn viết bằng ngôn ngữ Java, ta chọn Java vì những lý do sau đây:

Thứ nhất, những ứng dụng mạng kiểu này sẽ gọn gàng hơn khi viết bằng Java, với Java sẽ có ít dòng mã hơn, và mỗi dòng có thể dễ dàng giải thích với cả những người lập trình mới bắt đầu.

Thứ hai, những chương trình ứng dụng mô hình clients/server lập trình bằng Java đã gia tăng ngày càng thông dụng và nó có thể trở thành tiêu chuẩn cho lập trình mạng trong vài năm tới. Java là một ngôn ngữ độc lập nền, nó có cơ chế bẫy lỗi (điều quản các ngoại lệ), có thể giải quyết hầu nó các lỗi xảy ra trong quá trình xuất/nhập và những hoạt động mạng và khả năng phân luồng (thread) mạnh cung cấp những phương pháp đơn giản để xây dựng những server mạnh mẽ.

2. Kết quả đạt được

✓ Hoàn thiện được tính năng cơ bản của một chương trình quản lí mạng LAN.

✓ Giao diện được thiết kế đẹp mắt đơn giản và dễ sử dụng.

✓ Người dung đã có thể quản lý các máy client với khá đầy đủ chức năng

✓ Có thể đáp ưng được các nhu cầu cần thiết của một chương trình quản lí mạng LAN

3. Hướng phát triển

Mặc dù đã cố gắng, tìm hiểu các kiến thức đã học, kết hợp với thực tế và tra cứu các tài liệu liên quan đến đề tài "**Xây dựng chương trình giám sát mạng LAN theo mô hình Client – Server**" do hạn chế về thời gian, khả năng và kinh nghiệm nên đề tài không tránh khỏi những sai sót. Sau đây là hướng phát triển của em trong tương lai:

➢ Tìm hiểu sâu hơn về ngôn ngữ java để có thể đáp ứng nhiều hơn nữa nhu cầu của người sử dụng, phát triển và tối ưu hóa hệ thống.

➢ Tìm hiểu thêm một số ngôn ngữ, các phần mềm ứng dụng để nâng cao giao diện đồ họa đẹp mắt, thân thiện hơn…

➢ Xây dựng ứng dụng quy mô lớn hơn với nhiều ứng dụng, chức năng…

TÀI LIỆU THAM KHẢO

Tài liệu tham khảo trực tuyến

[1]. https://lib.hpu.edu.vn/bitstream/handle/123456789/18380/65_PhamHongThu_CT902.pdf

[2]. https://www.academia.edu/37278581/BG_lap_trinh_mang