**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

****

**ĐỒ ÁN TỐT NGHIỆP**

**ĐỊNH VỊ VÀ GIÁM SÁT ĐỐI TƯỢNG**

**DI ĐỘNG VỚI GPS/GPRS**

Giảng viên hướng dẫn:

**ThS. Lê Hoài Nghĩa**

**TS. Chung Tấn Lâm**

Sinh viên thực hiện:

1. **Nguyễn Nguyệt Minh – *06520292***
2. **Đào Văn Thiện – *06520450***

Lớp: **KTMT01**

Khóa: **2006**

***Tp. Hồ Chí Minh, tháng 2 năm 2011***

**LỜI MỞ ĐẦU**

Hiện nay, đất nước ta đang phát triển một cách mạnh mẽ trên mọi lĩnh vực kèm theo đó là sự phát triển của cơ sở vật chất hạ tầng mà ở đây là mạng lưới giao thông. Ở những thành phố như Hà Nội, Đà Nẵng, Tp.Hồ Chí Minh có lượng phương tiện tham gia giao thông lớn đã làm cho mạng lưới giao thông bị quá tải, việc vận chuyển hàng hóa trở nên khó khăn hơn dẫn tới giá cả bị đẩy lên cao để bù lại chi phí vận chuyển.

Để sản phẩm của mình có thể cạnh tranh được khi đưa ra thị trường, các công ty phải giảm thiểu tối đa những chi phí phát sinh trong đó việc vận chuyển và phân phối sản phẩm là một vấn đề đặc biệt quan trọng. Nếu tối ưu hóa được công đoạn này, công ty sẽ dễ dàng quản lý được thị trường của mình cũng như giảm thiểu chi phí vận chuyển kèm theo đó là giá thành sản phẩm sẽ có sức cạnh tranh hơn.

Đó cũng là mục tiêu mà đề tài tốt nghiệp: **Định vị và giám sát đối tượng di động với GPS/GPRS** hướng đến.

**Nguyễn Nguyệt Minh – MSSV: 06520292**

**Đào Văn Thiện – MSSV: 06520450**

Chúng em xin chân thành cảm ơn!

**LỜI CẢM ƠN**

**NHẬN XÉT**

**(Giảng viên hướng dẫn)**

**NHẬN XÉT**

**(Giảng viên phản biện)**

**MỤC LỤC**

**DANH MỤC CÁC BẢNG, SƠ ĐỒ, HÌNH**

# Giới thiệu đề tài

1. Giới thiệu

Vận tải là vấn đề quan trọng bậc nhất trong việc phát triển kinh tế, nhất là trong thời kì hiện nay khi giá nhiên liệu và các chi phí vận tải luôn biến động. Chính vì vậy, việc giám sát và tối ưu được quá trình vận tải hàng hóa là nhiệm vụ hàng đầu của doanh nghiệp nếu muốn tối ưu bộ máy sản xuất hàng hóa của mình.

Đề tài: **Định vị và giám sát đối tượng di động** là một giải pháp nhằm tối ưu hệ thống vận tải cũng như những hệ thống giám sát những đối tượng di động. Việc nghiên cứu và phát triển đề tài sẽ giúp cho doanh nghiệp nâng cao được chất lượng và sự hiệu quả của hệ thống sản xuất.

1. Mục đích, đối tượng và phạm vi nghiên cứu

**Mục đích của đề tài:**

* Xây dựng và phát triển giải pháp cho việc giám sát vận tải cũng như giám sát các đối tượng di động.
* Nghiên cứu và ứng dụng phần cứng GPS và hạ tầng kết nối di động cho giải pháp này.

**Đối tượng nghiên cứu của đề tài:**

* Đề tài tập trung nghiên cứu và phát triển giải pháp định vị và giám sát với công nghệ GPS và GPRS.

**Phạm vi nghiên cứu:**

* Phạm vi của đề tài là giám sát hệ thống vận tải với việc quản lý đội xe của một doanh nghiệp sản xuất hàng hóa.

1. Tổng quan

Hiện nay, những hệ thống định vị và giám sát vận tải đã được áp dụng khá phổ biến tại những nước phát triển. Ở Việt Nam, cũng đã xuất hiện những nhà cung cấp của giải pháp này. Sau đây là một số nhà cung cấp và những giải pháp đã được áp dụng:

1. Trên thế giới:
2. Tại Việt Nam:
3. Nhận xét

# Cơ sở lý thuyết

# Hệ thống định vị toàn cầu (GPS)

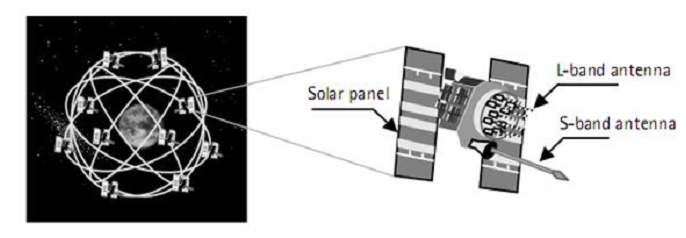
**Hệ thống Định vị Toàn cầu** (tiếng Anh: *Global Positioning System* - **GPS**) là hệ thống xác định vị trí dựa trên vị trí của các vệ tinh nhân tạo, do Bộ Quốc phòng Hoa Kỳ thiết kế, xây dựng, vận hành và quản lý. Trong cùng một thời điểm, ở một vị trí trên mặt đất nếu xác định được khoảng cách đến ba vệ tinh (tối thiểu) thì sẽ tính được tọa độ của vị trí đó.

Tuy được quản lý bởi Bộ Quốc phòng Hoa Kỳ, chính phủ Hoa Kỳ cho phép mọi người trên thế giới sử dụng một số chức năng của GPS miễn phí, bất kể quốc tịch nào.

Ngoài ra, hiện nay Nga đã đưa vào sử dụng hệ thống định vị GLONASS. Liên minh Châu Âu cũng đang phát triển hệ dẫn đường vệ tinh Galileo và Trung Quốc cũng xây dựng hệ thống định vị của riêng mình mang tên Bắc Đẩu.

* 1. Tổng quan

Thông thường GPS bao gồm một chòm sao 24 vệ tinh. Tuy nhiên, để đảm bảo hệ thống vệ tinh này bao phủ khắp toàn bộ trái đất một cách liên tục, những vệ tinh này được sắp xết sao cho mỗi 4 vệ tinh được đặt trong 6 mặt phẳng quỹ đạo. Với sự bố trí này, khoảng từ 4 đến 10 vệ tinh sẽ luôn hiện hữu tại bất cứ nơi nào trên thế giới, nếu góc ngẩng là 100. Ta chỉ cần duy nhất 4 vệ tinh để cung cấp sự định vị.aaa



Hình 1. Chòm sao GPS

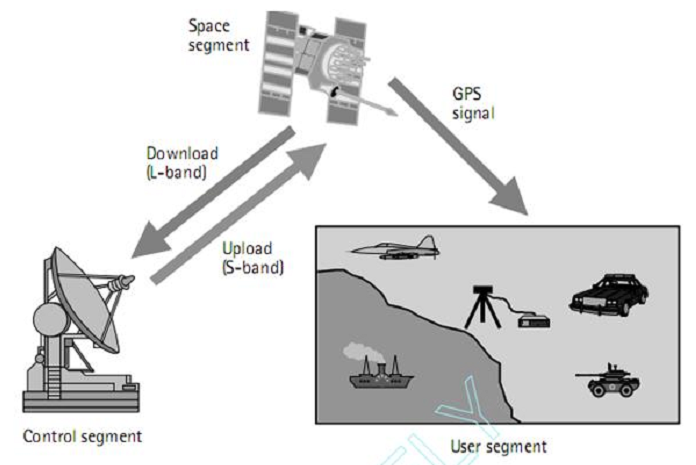
S-band (10cm-radar short-band): 1.55 – 5.2Ghz.

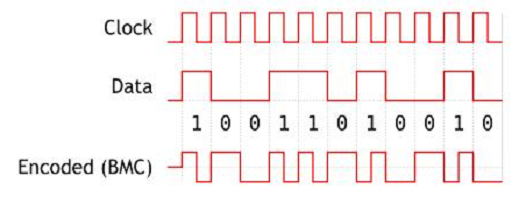
L-band (20cm-radar long-band): 950Mhz – 1450Mhz.

* 1. Những phân đoạn GPS

GPS bao gồm 3 phân đoạn: phân đoạn không gian, phân đoạn điều khiển, phân đoạn người dùng.

* Phân đoạn không gian bao gồm chòm sao 24 vệ tinh. Mỗi vệ tinh GPS phát đi tín hiệu, bao gồm: hai sóng sin(thành phần sóng mang), hai chuỗi dữ liệu số, và một thông điệp điều hướng. Dữ liệu số và thông điệp điều hướng kết hợp với sóng mang bằng cách điều chế nhị phân bitphase. Sóng mang và chuỗi dữ liệu số chủ yếu được sử dụng để xác định khoảng cách từ máy thu của người sử dụng đến những vệ tinh GPS. Thông điệp điều hướng bao gồm tọa độ của vệ tinh, toại độ này biểu diễn dưới dạng hàm biến đổi theo thời gian và một số thông tin cần thiết khác. Tín hiệu phát được điều khiển bởi những đồng hồ nguyên tử (atomic clocks) có độ chính xác cao được gắn vào những vệ tinh.



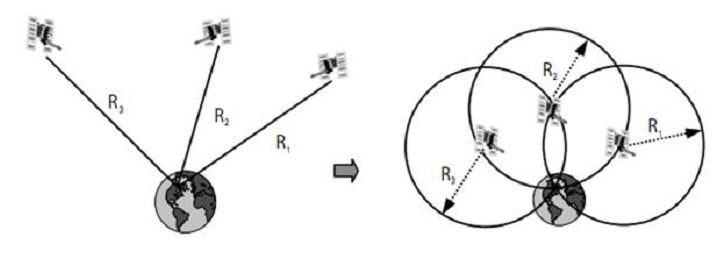


* Phân đoạn điều khiển của hệ thống GPS bao gồm một mạng lướng rộng khắp những trạm theo dõi (tracking station), với một trạm điều khiển chính. Nhiệm vụ của phân đoạn này là theo dõi dấu vết của những vệ tinh GPS để định vị và tiên đoán vị trí vệ tinh, tình trạng hệ thống…. Thông tin sau đó được đóng gói và upload lên những vệ tinh GPS thông qua đường link bằng S.
* Phân đoạn người sử dụng bao gồm quân sự và dân sự. Với một bộ thu GPS kết nối với một antenna GPS, một người dùng có thể thu được tín hiệu GPS, tín hiệu này sau đó được sử dụng để xác định vị trí của người dùng.
  1. Nguyên tắc hoạt động cơ bản

Hệ thống GPS hoạt động như sau: Nếu như khoảng cách từ một điểm trên trái đất tới ba vệ tinh được xác đình cùng với thông tin về vị trí vệ tinh thì chúng ta có thể xác định được vị trí của người dùng.

Mỗi vệ tinh GPS phát liên tục một tín hiệu vô tuyến tạo thành tổng thể từ hai song mang, hai mã và một thông điệp điều hướng. Khi thiết bị thu GPS hoạt động, no sẽ thu lấy tín hiệu thông qua antenna. Một khi thiết bị thu nhận được tín hiệu GPS, nó sẽ sử lý nhờ vào những phần mềm được tích hợp bên trong. Kết quả xử lý bao gồm các khoảng cách tới những vệ tinh GPS và tọa độ vệ tinh được xác định thông qua thông điệp điều hướng.

Theo lý thuyết chỉ cần xác định khoảng cách tới 3 vệ tinh , thiết bị thu sẽ được định vị tại chỗ giao nhau của ba hình cầu, mỗi hình cầu này là một bán kính tương ứng với khoảng cách vệ tinh – thiết bị thu và tâm vệ tinh đó (hình 6). Tuy nhiên, thực tế phải cần đến 4 vệ tinh để mô tả độ lệch giữa đồng hồ của thiết bị thu và đồng hồ vệ tinh.

Hình 6. Nguyên tắc cơ bản định vị GPS

* 1. Hoạt động của GPS có thể bị ảnh hưởng bởi các yếu tố sau:
* Khi các vệ tinh ở quá gần nhau, chúng sẽ khiến việc xác định vị trí chính xác trở nên khó khăn hơn.
* Vì tín hiệu radio đi từ vệ tinh xuyên qua tầng điện ly và tầng đối lưu, tốc độ cần thiết để tín hiệu truyền tới thiết bị nhận sẽ bị chậm đi. Hệ thống GPS có dự phòng điều đó bằng cách tính thêm khoảng thời gian chậm trễ trung bình, nhưng cũng không được hoàn toàn chính xác.
* Chướng ngại vật lớn như các dãy núi hay các tòa nhà cao tầng cũng là cho thông tin bị sai lệch.
* Giữa thiết bị nhận (nhất là của người dùng cá nhân) với vệ tinh có thể không hoàn toàn trùng khớp về mặt thời gian, các vệ tinh đôi khi chạy lệch quỹ đạo.
  1. Tín hiệu GPS:

Các vệ tinh GPS phát hai tín hiệu vô tuyến công suất thấp dải L1 và L2 (dải L là phần sóng cực ngắn của phổ điện trải rộng từ 0,39 tới 1,55GHz). GPS dân sự dùng tần số L1 = 1575.42 MHz trong dải UHF. Tín hiệu truyền trực thị, có nghĩa chúng sẽ xuyên qua mây,thủy tinh và nhựa nhưng không qua phần lớn các đối tượng cứng như núi và nhà.

L1 chứa hai mã ngẫu nhiên đó là mã Protected (P) và mã Coarse/Acquisition (C/A). Mỗi một vệ tinh có một mã truyền dẫn nhất định, cho phép máy thu GPS nhận dạng được vệ tinh thông qua tín hiệu. Mục đích của các mã tín hiệu này là để tính toán khoảng cách từ vệ tinh đến thiết bị thu GPS.

Tín hiệu GPS chứa ba thành phần thông tin khác nhau – mã giả ngẫu nhiên, dữ liệu thiên văn và dữ liệu lịch. Mã giả ngẫu nhiên đơn giản chỉ là mã định danh để xác định được vệ tinh nào đã và đang phát thông tin nào.Dữ liệu thiên văn cho thiết bị thu GPS biết vệ tinh ở đâu trên quỹ đạo ở thời điểm trong ngày. Mỗi vệ tinh phát dữ liệu thiên văn chỉ ra thông tin quỹ đạo của vệ tinh đó và các vệ tinh khác trong hệ thống.

Dữ liệu lịch được phát đều đặn bởi mỗi quả vệ tinh, chứa thông tin quan trọng về trạng thái cảu vệ tinh (tốt hay không tốt), ngày giờ hiện tại. Phần này của tín hiệu là cốt lõi để tiền hành định vị.

# GPRS/EDGE TCP

Dịch vụ vô tuyên gói tổng hợp (General Packet Radio Service-GPRS) là một dịch vụ dữ liệu di động dạng gói dành cho những người dùng Hệ thống thông tin di động toàn cầu (GSM) và điện thoại di động IS-136. Nó cung cấp dữ liệu tốc độ từ 56 đến 114 kbps.

GPRS có thể được dùng cho những dịch vụ như truy cập Giao thức Ứng dụng không dây (WAP), dịch vụ tin nhắn (SMS, MMS), và các dịch vụ liên lạc Internet như email và truy cập World Wire Web.

**Dịch vụ cung cấp GPRS.**

GPRS mở rộng khả năng chuyển mạch dữ liệu GSM và có thể tạo nên những dịch vụ sau:

* Truy cập Internet một cách liên tục.
* Tin nhắn đa phương tiện (MMS).
* Push to talk qua di động (PoC/PTT).
* Tin nhắn tức thời và mạng không dây.
* Ứng dụng Internent cho thiết bị thông ming qua giao thức ứng dụng không dây (WAP).
* Dịch vụ Point-to-Pont (P2P): mạng nội bộ với Internet.

Nếu SMS qua GPRS được sử dụng, tốc độ truyền có thể đạt được khoảng 30 tin nhắn/ phút. Điều này nhanh hơn nhiều so với sử dụng tin nhắn SMS thông thường trên GSM, có tốc độ truyền tải SMS là khoảng 6 đến 10 tin nhắn SMS/ phút.

**Giao thức hỗ trợ**

GPRS hỗ trợ những giao thức sau:

* Internet Protocol (IP). Trong thực tế, tích hợp trình duyệt di động sử dụng IPv4 và IPv6 chưa phổ biến.
* Point-to-point Protocol (PPP). Trong chế độ này, PPP thường không được hỗ trợ bởi các nhà điều hành điện thoại nhưng nếu điện thoại di động sử dụng như một modem kết nối với máy tính, PPP được sử dụng để tạo IP tới điện thoại. Điều này cho phép một địa chỉ IP được gán cho các thiêt bị di động.
* X.25 Connection. Giao thức này thường được sử dụng cho các ứng dụng như thiết bị đầu cuối thanh toán không dây mặc dù nó bị xóa khỏi tiêu chuẩn.

Khi TCP/IP được sử dụng, mỗi điện thoại có thể có một hoặc nhiều địa chỉ IP được phân bổ. GPRS sẽ lưu trữ và chuyển tiếp các gói tin IP với điện thoại ngay cả trong cả khi cầm tay. TCP xử lý bất kỳ gói dữ liệu nào bị mất mát.

**Địa chỉ**

Một kết nối GPRS được thiết lập bằng cách tham chiếu đến tên điểm truy cập của nó (APN). APN định nghĩa các dịch vụ như WAP, SMS, MMS, Email…

Để thiết lập một kết nối GPRS cho một modem không dây, người dùng phải xác định một APN, tùy chọn một tên đăng nhập và mật khẩu, và rất hiếm khi có một địa chỉ IP, tất cả được cung cấp phụ thuộc vào nhà điều hành mạng.

**Tính khả dụng**

Tốc độ tối đa của một kết nối GPRS được cung cấp trong năm 2003 là tương đượng với một kết nối modem trong một mạng điện thoại cố định có dây, khoảng 32-40kbit.s, tùy thuộc vào các điện thoại sử dụng. Độ trễ là rất cao, thời gian khứ hồi (RTT) khoảng 600 – 700ms và thường đạt 1s.

Hiện nay, tốc độ của GPRS được cải tiến không ngừng nhờ sự nâng cấp cơ sở hạ tầng của các nhà các mạng di động.

Vì vậy, với một hệ thống yêu cầu dữ liệu truyền từ Client về là không quá lớn, thì GPRS là rất hiệu quả và phù hợp.

1. Windows Service

Trong hệ điều hành Windows, một Windows Service là một thực thi lâu dài thực hiện chức năng cụ thể và được thiết kế không yêu cầu người cùng can thiệp. Windows Service có thể được cấu hình để bắt đầu khi hệ điều hành được khởi động và chạy nền khi Windows đang chạy, hoặc cũng cũng có thể được bắt đầu bằng tay khi cần thiết.

**Quản lý Service**

Một khi service được cài đặt, nó sẽ cho phép người dùng có thể:

* Khởi động, dừng, tạm dừng hoặc khởi động lại một Service.
* Xác định tham số của service.
* Thay đổi kiểu khởi động bao gồm: Automatic, Manual, Disabled.
  + Automatic: khởi động service lúc hệ thống khởi động
  + Manual: Khởi động service khi được yêu cầu hoặc được gọi từ một ứng dụng (theo định nghĩa, nhưng chỉ có một số thời gian trong thực tế phụ thuộc vào service).
  + Disabled: vô hiệu hóa hoàn toàn service , ngăn chặn việc cho phép service chạy.
  + Automatic (delayed) là một kiểu khởi động mới được giới thiệu trong Windows Vista, nó sẽ bắt đầu service sau khoảng thời gian ngắn sau khi hệ thống kết thúc việc khởi động và các thiết đặt bận rộn ban đầu để hệ thống khởi động nhanh hơn.
* Thay đổi tài khoản đăng nhập của service.
* Cấu hình tùy chọn phục hồ sau khi service bị thất bại
* Xuất ra file text hoặc CSV danh sách những service có trong hệ thống.

**Phát triển một Windows Service.**

Một windows service được tạo bởi những công cụ phát triển như Microsoft Visual Studo hặc Embarcadero Delphi. Windows cung cấp một interface gọi là Service Control Manager để quản lý khởi động và kết thúc của service. Một ứng dụng muốn là một service thì đầu tiên cần viết cách mà nó có thể xử lý thông điệp khởi động, kết thúc và tạm dừng từ Service Control Manager. Sau đó, một hoặc nhiều API lời gọi, tên của service hoặc các thuộc tính khác được đăng ký với Service Control Manager. Mặc dù các service thường không có một giao diện người dùng, nhưng chúng ta có thể tạo các forms và những thành phần giao diện khác. Trong trường hợp này, “Allow service to interact with desktop” nên được check trong tab đăng nhập tại hộp thoại thuộc tính Service.

1. WPF và Design Pattern trong thiết kế phần mềm
   1. WPF là gì?

WPF, viết tắt của Windows Presentation Foundation là hệ thống API mới hỗ trợ việc xây dựng giao diện đồ họa trên nền Windows. Được xem như thế hệ kế tiếp của Winforms, WPF tăng cường khả năng lập trình giao diện của lập trình viên bằng cách cung cấp cái API cho phép tận dụng những lợi thế về đa phương tiện hiện đại. Là một bộ phận của .Net Framework, WPF sẵn có trong Windows Vista, Windows 7 và Windows Server 2008. Đồng thời, WPF cũng có thể hoạt động trên nền Windows XP SP2 hoặc mới hơn và cả Windows Server 2003.

WPF được xây dựng nhằm vào ba mục tiêu cơ bản:

* Cung cấp một nền tảng thống nhất để xây dựng giao diện người dùng;
* Cho phép người lập trình và người thiết kế giao diện làm việc cùng nhau một cách dễ dàng.
* Cung cấp một công n ghệ chung để xây dựng giao diện người dùng trên cả Windows và trình duyệt Web.

WPF cung cấp đầy đủ các tính năng lập trình giao diện trong cùng một công nghệ đơn nhất. Điều này giúp cho quá trình tạo giao diện người dùng trở nên dễ dàng hơn đáng kể.

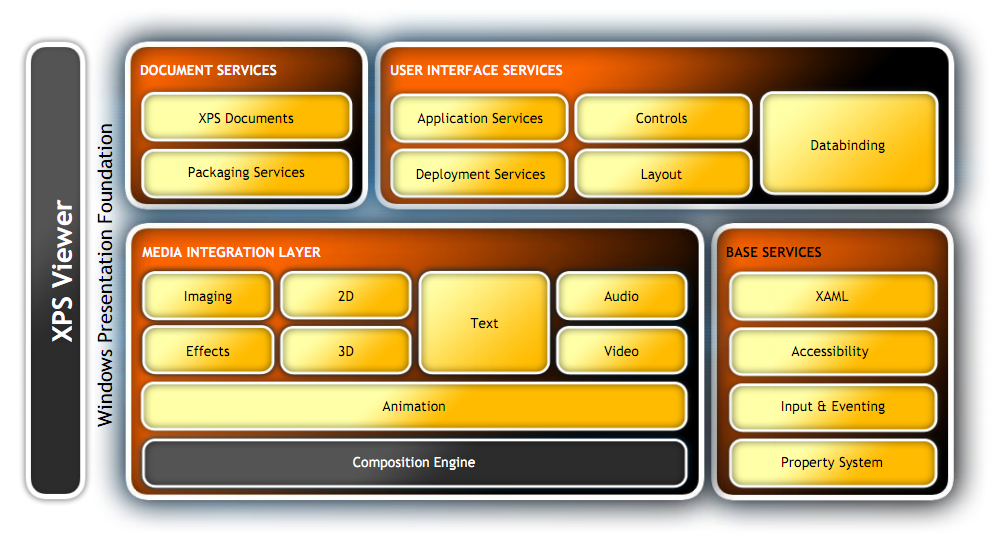


Hình. Giao diện quản lý bệnh nhân bằng WPF

**Các thành phần của WPF**

Giống như các thành phần khác của .Net Framework, WPF tổ chức các chức năng theo một nhóm namespace cùng trực thuộc namespace System.Windows. Bất kể chức năng nào được sử dụng, cấu trúc cở bản của mọi ứng dụng WPF đều gần như nhau.

Mặc dù WPF cung cấp một nền tảng thống nhất để tạo giao diện người dùng, những công nghệ mà WPF chứa đựng có thể phân chia thành những thành phần độc lập. Nhân của WPF là cơ chế tạo sinh đồ họa dựa trên vector và độc lập với độ phân giản nhằm tận dụng những lợi thế của phần cứng đồ họa hiện đại. WPF được mở rộng với các tập tính năng phát triển ứng dụng bao gồm XAML, các control, cơ chế móc nối dữ liệu, layout, đồ họa 2 chiều, hoạt họa,, style, khuông dạng mẫu, văn bản, media, text và in ấn. WPF nằm trong .Net Framework, nên ngoài ra, ứng dụng WPF có thể kết hợp các thành phần khác có trong thư viện lớp của .NET Framework.



Hình. Các thành phần cơ bản của WPF

* 1. Mẫu thiết kế (Design Pattern)

Trong công nghệ phần mềm, một mẫu thiết kế là một giải pháp tổng thể cho các vấn đề chung trong thiết kế phần mềm. Một mẫu thiết kế không phải là một thiết kế hoàn thiện để mà có thể được chuyển đổi trực tiếp thành mã, nó chỉ là một mô tả hay là mẫu mô tả cách giải quyết một vấn đề mà có thể được dùng trong nhiều tình huống khác nhau. Các mẫu thiết kế hướng đối tượng thường cho thấy mối quan hệ và sự tương tác giữa các lớp hay các đối tượng, mà không cần chỉ rõ các lớp hay đối tượng của từng ứng dựng cụ thể. Các giải thuật không được xem là các mẫu thiết kế, vì chúng giải quyết các vấn đề về tính toán hơn là các vấn đề thiết kế.

Sau đây là hai mô hình thiết kế được nhóm kết hợp trong việc xây dựng phần mềm quản lý đội xe.

**Mô hình PureMVC**

PureMVC (là một dạng khác của mô hình MVC cổ điển) được xây dựng nhằm mục đích giúp người thiết kế phần mềm tách biệt ứng dụng thành ba tầng riêng biệt: Model, View, Controller.

Trong việc thực thi của mô hình MVC cổ điển, các tầng ứng dụng sẽ được biểu diễn bở ba Singletons (một lớn chỉ được tạo ra một lần).

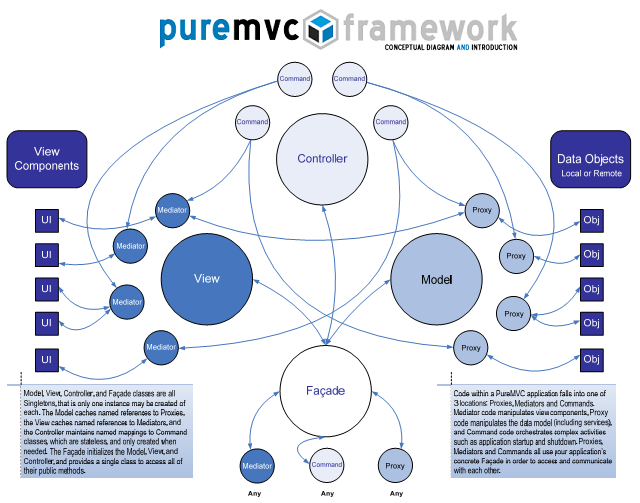
Trong pureMVC, mô hình này có thêm một Singleton thứ tư là Façade, đơn giản hóa việc phát triển bằng cách cung cấp một interface duy nhất cho việc liên lạc trong ứng dụng.

Model lưu tham chiếu được đặt tên Proxy, trong đó biểu diễn một API cho thao tác với mô hình dữ liệu (bao gồm cả các dữ liệu lấy từ các service ngoài).

View: chủ yếu lưu tham chiếu được đặt tên Mediator, nhằm quản lý các thành phần View tạo nên giao diện người dùng.

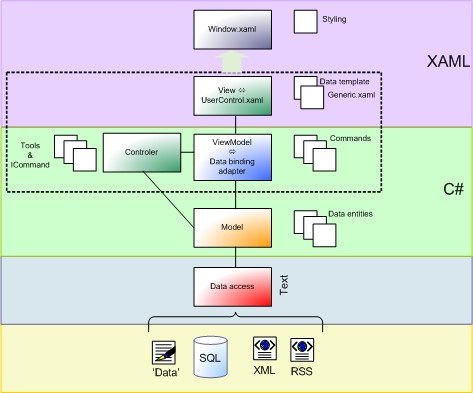
Controller: duy trì những tên ánh xạ lên lớp Command, và chỉ tạo khi cần.

Façade: khởi tạo và lưu trữ các Core (Model, View và Controller) và cung cấp một nơi duy nhất để truy cập vào tất cả các thương thức public.

****

**Mô hình MVVM (Model – View – View Model)**

MVVM khác với MVP (Model-View-Presenter) và MVC (Model-View-Controller). Đây là design pattern đặc trưng và mạnh mẽ nhất của WPF.



Hình. Mô hình thiết kế chuẩn của MVVM

Trong mô hình này:

* Model: là dữ liệu hay có thể là phiên bản của dữ liệu, dữ liệu nên có thể được binding vì binding là một trong những thành phần quan trọng nhất của WPF.
* View: Giao diện người dùng dưới dạng XAML.
* View Model: Model của View giữ vai trò chủ yếu là cung cấp Model cho View để binding và cấp Command cho phiá View.

Trong MVVM các Business logic thường được để ở các Command ví dụ delete, cut, copy, paste. Sẽ có built-in command tương ứng Application.Delete, Application.Copy, ApplicationCommands.Cut….

Tùy vào thành phần của ứng dụng mà Model có thể là Wrapper hay Data của Data.

* Dữ liệu liên kết cơ sở hạ tầng WPF rất mạnh mẽ - với WPF, các ràng buộc giữa view và “ViewModel” rất đơn giản để xây dựng vì một đối tượng “ViewModel” được thiết lập như là DataContext;
* Nếu giá trị trong ViewModel thay đổi, sự thay đổi đó sẽ tự động truyền cho chế độ xem dữ liệu thông qua ràng buộc. Khi người dùng nhấn một nút trong View, một lệnh trên các ViewModel thực thi để thực hiện các hành động được yêu cầu. Các ViewModel, không bao giờ là View, thực hiện tất cả các thay đổi được thực hiện cho các mô hình dữ liệu.

# Nội dung

1. Tổng quan về thiết kế hệ thống
   1. Yêu cầu của hệ thống:

* Đối tượng (xe) có khả năng tự xác định được vị trí hiện tại của mình.
* Đối tượng (xe) có khả năng gửi thông tin về trung tâm điều khiển theo một chu kỳ thời gian hoặc khi có yêu cầu từ trung tâm.
* Trung tâm điều khiển giữ nhiệm vụ thu nhận và phân tích thông tin mà đối tượng gửi về để giám sát đối tượng
  1. Phân tích:
  2. Để xác định được vị trí của đối tượng chúng ta có những phương pháp sau:
* Sử dụng hệ thống định vị toàn cầu để xác định tọa độ của đối tượng qua vệ tinh: Hiện nay, có một số hệ thống định vị toàn cầu đang được xây dựng cũng như đã đưa vào sử dụng, trong đó phổ biến nhất hiện nay là GPS (Mỹ). Sắp tới đây Nga cũng sẽ đưa hệ thống GLONASS của mình vào sử dụng. Châu Âu và Trung Quốc cũng đang phát triển hai hệ thống của mình là Galileo và Bắc Đẩu.

Ưu điểm: Hoàn toàn miễn phí, hoạt động trong mọi thời tiết, mọi nơi trên trái đất, 24h một ngày.

Nhược điểm: Gặp khó khăn trong việc xác định tọa độ khi đối tượng ở trong nhà, đường hầm hoặc khu có nhiều nhà cao tầng che khuất tín hiệu đến vệ tinh.

* Sử dụng hệ thống định vị dựa trên các trạm phát song di động (BTS). Hệ thống này sẽ xác định khoảng cách của thuê bao tới tối thiểu 3 trạm BTS gần nhất để từ đó xác định được tọa độ của thuê bao dựa trên tọa độ sẵn có của các trạm.

Ưu điểm: Hoạt động được ở những nơi mà GPS bị giới hạn

Nhược điểm: Những nơi nằm ngoài vùng phủ sóng sẽ rất khó để xác định đủ 3 trạm BTS tới thuê bao. Bên cạnh đó, ở nước ta hiện nay, hệ thống này chưa được các nhà mạng chú trọng (duy nhất Mobifone có dịch vụ SMS Locator)

* Giải pháp tối ưu nhất là việc kết hợp hai giải pháp trên để duy trì được sự ổn định trong việc xác định tọa độ của đối tượng ngay cả khi đối tượng ở trong nhà, trong hầm hay khu nhà cao tầng. Giải pháp này là A-GPS.
  1. Để đối tượng có khả năng gửi thông tin về trung tâm điều khiển, chúng ta cần thiết kế một module có khả năng kết nối mạng và có vùng phủ sóng rộng.

Trong trường hợp này, Mobile Internet là phương án khả thi nhất với vùng phủ sóng rộng và khả năng kết nối tốt.

Nhóm đã chọn dịch vụ GPRS (General Packet Radio Service) để thực hiện việc kết nối và truyền dữ liệu giữa trung tâm và đối tượng thông qua giao thức TCP. Lý do là vì giá thành của những thiết bị GPRS hay EDGE thường rẻ hơn so với những thiết bị 3G. Hơn nữa, với việc chỉ truyền dữ liệu dạng gói chứa thông tin nên GPS/EDGE (tốc độ 144kbit/s khi di chuyển với tốc độ cao) là hoàn toàn phù hợp và đạt được hiệu quả cao nhất.

* 1. Trung tâm điều khiển:

Nhiệm vụ của trung tâm là thu nhận và phân tích thông tin từ đối tượng để giám sát và quản lý hoạt động của đối tượng.

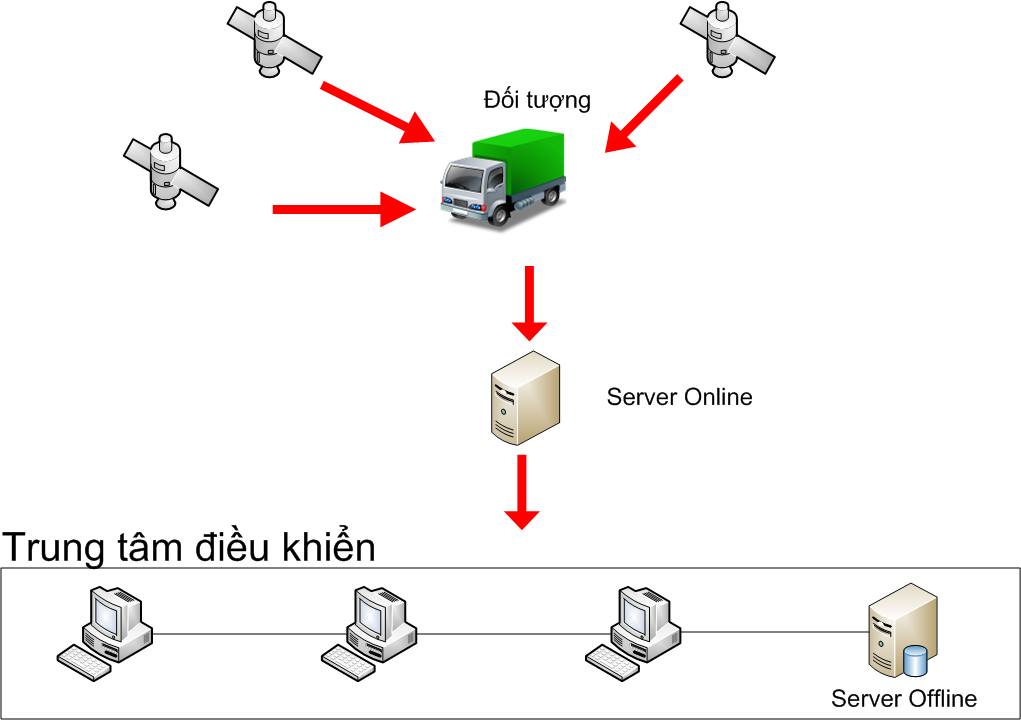
Để kết nối và thu nhận dữ liệu từ đối tượng, trung tâm sẽ có một server dùng để lắng nghe những dữ liệu được trả về. Nhằm tối ưu hóa tốc độ và sự hiệu quả của hệ thống, nhóm đã tách riêng Server Online và Server Offline.

Trong đó, Server Online sẽ có nhiệm vụ thu nhận hoặc truyền dữ liệu với đối tượng thông qua giao thức TCP. Dữ liệu trên Server Online chỉ bao gồm thông tin tọa độ và những thông tin cơ bản mà đối tượng gửi về.

Server Offline sẽ bao gồm thông tin của toàn bộ hệ thống: Từ quản lý đối tượng đến giám sát lịch sử hoạt động của đối tượng. Mặt khác, với một doanh nghiệp, server này sẽ có khả năng kết nối đến các dữ liệu khác của doanh nghiệp như sản phẩm, khách hàng, đội xe… nhằm tối ưu hóa khả năng quản lý và giám sát hoạt động vận tải.

Một chức năng khác của Server Offline là lưu trữ dữ liệu của Server Online để đề phòng trường hợp hệ thống mạng gặp trục trặc.

* 1. Sơ đồ



1. Module GPS/GPRS
2. **Chức năng của thiết bị:**

* Thiết bị có nhiệm vụ xác định tọa độ của xe bằng GPS, xác định các thông số khác của xe qua các cảm biến gắn trên xe (cảm biến nhiệt độ, tốc độ, cảm biến đo mức nhiên liệu…)
* Thiết bị kết nối và truyền nhận dữ liệu với Server qua giao thức TCP/IP sử dụng công nghệ GPRS cho mạng di động.

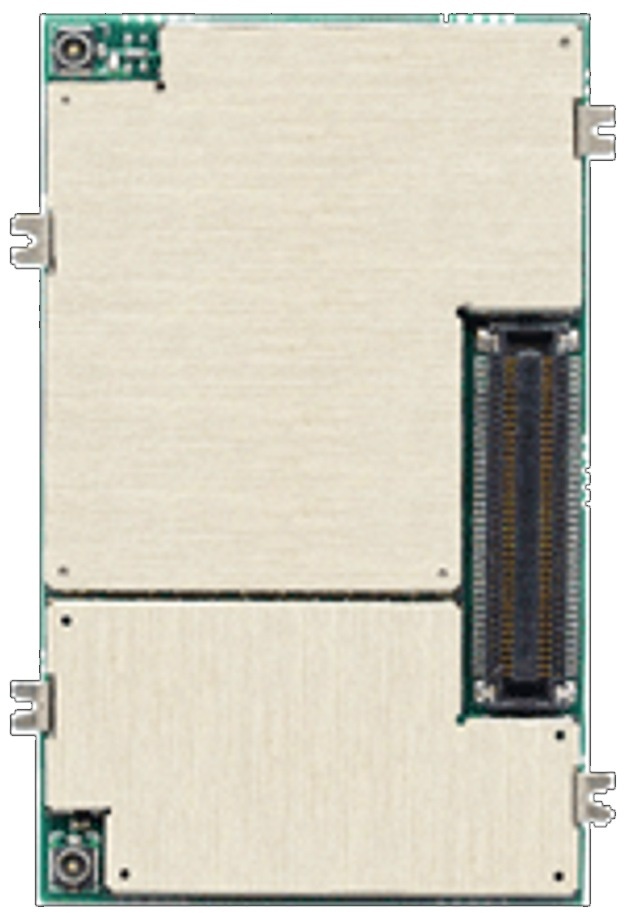
1. **Linh kiện chính của thiết bị:**

Chip SIM548 – Chip chuyên dụng GPS/GPRS

Là sản phẩm được thiết kế bởi SIMCOM tích hợp sẵn 2 module GPS và GPRS.

*Những tính năng chính:*

* Kích thước: 34mm x 55mm x 3 mm, nặng 12g
* Hoạt động trong -20oC đến +55oC
* Hỗ trợ chuẩn GPRS class 10, tốc độ tối đa 85,6Kbps
* Băng tần GSM/GPRS 850/900/1800/1900MHz
* Hoạt động với điện thế 3,4 – 4,5 V
* Tích hợp TCP/IP
* Được gắn sẵn 2 ăngten riêng biệt cho GSM/GPRS và GPS.
* Hỗ trợ giao diện Dual – Serial cho GPS
* GPS với kiểu tọa độ WGS-84
* Hỗ trợ FAX, SMS, Voice
* Hỗ trợ PBCCH
* Đóng gói với 80 chân.
* Bộ nhớ bao gồm 4Mbit Flash và 1Mbit SDRAM



Hình 13: SIM548

## Phần mềm hệ thống quản lý đội xe

* 1. Xây dựng cơ sở dữ liệu
     1. Yêu cầu thiết kế:

Hệ thống cần lưu trữ những thông tin cơ bản sau:

* Xe, đội xe, tài xế
* Tọa độ và trạng thái xe
* Thông tin hoạt động của Server Online (System Log)
* Thông tin người điều khiển
* Thông tin hàng hóa, nhóm hàng hóa, khách hàng
* Thông tin lịch sử hoạt động vận chuyển hàng hóa
  + 1. Thiết kế cơ sở dữ liệu

Để đảm bảo tốc độ của hệ thống, phù hợp với việc phân chia Server thành Server Online và Server Offline, cơ sở dữ liệu cũng được tách biệt để phù hợp với từng Server.

* + - 1. Server Online

Có nhiệm vụ chính là định vị và theo dõi xe, do đó, cơ sở dữ liệu được thiết kế như sau:



**Bảng Module**

Bảng này sẽ lưu thông tin của module phần cứng gắn trên xe tải. Trong đó, ModuleNumber là số của SIM GPRS, ModuleType sẽ lưu chủng loại của Module này.

Để module có thể hoạt động trong hệ thống, module phải được kích hoạt nhằm đảm bảo sự an toàn của hệ thống.

**Bảng Tracking**

Bảng này sẽ lưu thông tin tọa độ cũng như trạng thái của xe nhằm nhiệm vụ giám sát.

Nó bao gồm dữ liệu kinh độ (Longtitude), vĩ độ (Latitude), tốc độ(Speed) và mức nhiên liệu(FuelLevel) của xe.

Thời gian server nhận được thông tin từ module cũng được lưu lại nhằm quản lý tốc độ và sự hoạt động của hệ thống.

**Bảng Session**

Hoạt động của xe được quản lý dựa trên những Session (phiên làm việc).

Một session bắt đầu từ lúc xe khởi hành và kết thúc khi xe tới đích.

Trong suốt quá trình hoạt động trong 1 session, xe sẽ gửi liên tục dữ liệu tọa độ của mình hoặc sẽ gửi khi được yêu cầu. Một session chứa tối thiểu 2 dữ liệu tracking và có thể chứa rất nhiều tracking.

**Bảng System Log**

Bảng này sẽ lưu lại hoạt động của Server, từ việc server được khởi động, kích hoạt chức năng lắng nghe dữ liệu từ client đến những lỗi đã xảy ra trong quá trình hoạt động. Tất cả nhằm mục đích giúp người quản trị hệ thống theo dõi được hoạt động của hệ thống từ đó đưa ra hướng giải quyết khắc phục hệ thống khi xảy ra lỗi.

* + - 1. Server Offline

Là server chứa những thông tin nội bộ của doanh nghiệp và quản lý hoạt động vận tải của doanh nghiệp, nó sẽ được lưu trữ và quản lý bởi đội ngũ giám sát của doanh nghiệp. Do đó, cơ sở dữ liệu được thiết kế như sau:



**Bảng Vehicle Tracking**

Đây là bảng chính của cơ sở dữ liệu, nó nắm giữ toàn bộ thông tin để giám sát hoạt động của một xe hay một phiên làm việc (Session) bao gồm: Thông tin xe, tài xế, sản phầm mà xe vận chuyển, khách hàng của sản phẩm và thời gian của hoạt động này.

Hệ thống sẽ cho phép doanh nghiệp lên lịch trước những Vehicle Tracking này, sau đó dựa trên sự giám sát của mình để đảm bảo được tính chính xác và tối ưu trong hoạt động vận chuyển.

Với những Tracking đã hoàn thành, IsCompleted sẽ được đặt là true.

**Bảng Fleet**

Trong một doanh nghiệp vận tải, việc quản lý đội ngũ của mình theo đội xe là một mô hình khá phổ biến hiện nay, nó có tác dụng giúp doanh nghiệp tiếp cận được nhanh hơn với đội ngũ của mình, từ đó có thể giám sát và quản lý một cách linh hoạt đội ngũ của mình.

Bảng này sẽ lưu trữ thông tin cơ bản của một đội xe như: Tên (Name), Mô tả (Description) và ngày thành lập.

Thông tin về đội trưởng, đội phó sẽ được xác định thông qua bảng Driver.

**Bảng Driver**

Bảng này sẽ lưu trữ thông tin về tài xế bao gồm:

* Tên (Firstname - LastName)
* Ngày sinh (Birthday)
* Số CMND (IDCard)
* Loại bằng lái (LicenceType)
* Hình ảnh(Image)

Bảng này cũng giữ quan hệ với bảng Fleet nhằm xác định tài xế thuộc đội xe nào và tài xế có phải đội trưởng hay đội phó của đội xe đó hay không.

**Bảng Vehicle**

Nhằm quản lý phương tiện vận tải, cơ sở dữ liệu sẽ lưu trữ những thông tin cơ bản của phương tiện thông qua bảng Vehicle.

Cụ thể, nó sẽ lưu trữ những thông tin sau:

* Biển số xe (NumPlate) – *khóa chính*
* Số khung (VIN)
* Số máy (EIN)
* Nhà sản xuất(Manufacturer)
* Năm sản xuất (Year)
* Loại xe (Type)
* Trọng tải (Tonage)
* Dung tích bình xăng (Capacity)
* Đội xe (FleetID) – nhằm quản lý xe theo đội xe.

**Bảng Group Product**

Bảng này sẽ lưu trữ thông tin về những nhóm sản phẩm (đối với doanh nghiệp quản lý sản phẩm của mình theo nhóm). Bao gồm:

* Tên nhóm (Name)
* Chi tiết thông tin nhóm (Description)

**Bảng Product**

Bảng này lưu trữ thông tin về sản phẩm, nhằm giúp doanh nghiệp quản lý được hệ thống sản phẩm và chủng loại mình sản xuất.

Bảng này sẽ được kết hợp với dữ liệu quản lý sản phẩm của hệ thống sản xuất (với những doanh nghiệp áp dụng mô hình quản lý này) nhằm theo dõi được hoạt động sản xuất và doanh thu cua từng sản phẩm.

Bảng này bao gồm:

* Tên sản phẩm (Name)
* Chi tiết thông tin sản phẩm (Description)
* Nhóm của sản phẩm (GroupID) – khóa này liên kết với bảng Group Product đối với những doanh nghiệp quản lý sản phẩm theo mô hình nhóm.

**Bảng Customer**

Đây cũng là bảng có thể kết hợp với dữ liệu quản lý của doanh nghiệp nhằm theo dõi được hoạt động của mình đối với một khách hàng nhất định.

Thông tin của khách hàng bao gồm:

* Tên (Name)
* Tên đầy đủ (FullName)
* Tên viết tắt của khách hàng (Initials)
* Địa chỉ (Address)
* Số điện thoại liên lạc (Phone)
* Tọa độ của khách hàng (Longtitude, Latitude)
  + - 1. Khả năng mở rộng và tính linh hoạt của cơ sở dữ liệu

Với một hệ thống có thể áp dụng cho hoạt động của một doanh nghiệp, việc mở rộng và kết nối với cơ sở dữ liệu sẵn có của doanh nghiệp là một yêu cầu nhằm nâng cao chất lượng hệ thống. Chính vì vậy, khi thiết kế cơ sở dữ liệu, áp dụng phương pháp chia cơ sở dữ liệu làm 2 nhóm dữ liệu: Dữ liệu Server Online và dữ liệu Server Offline. Điều này đảm bảo dữ liệu của việc xác giám sát hoạt động vận tải có thể tách biệt với hệ thống quản lý nội bộ của doanh nghiệp nhằm đảm bảo sự ổn định khi doanh nghiệp có nhu cầu mở rộng hệ thống của mình.

Việc mở rộng và kết nối với cơ sở dữ liệu sẵn có của doanh nghiệp sẽ được thông qua cơ sở dữ liệu của Server Offline. Chính vì vậy,nhóm đã cố gắng tối ưu hóa cơ sở dữ liệu này theo tiêu chí: đơn giản và quan hệ nội bộ rõ ràng. Việc thêm các trường hay các bảng trong cơ sở (nếu có) cũng sẽ không ảnh hưởng với dữ liệu sẵn có của Server Offline.

* 1. Server Online
     1. Tổng quan

Nhiệm vụ của Server Online là lắng nghe và thu nhận dữ liệu được đối tượng gửi về. Sau đó phân tích dữ liệu này theo một định dạng cho trước thành dữ liệu quản lý thông tin của đối tượng.

*Ví dụ:*

Dữ liệu gửi về: 0166663870210,67827896104,589657580,539,1

Với định dạng cho trước, server sẽ phân tích dữ liệu thành:

1. Số Module (số SIM) – 11 ký tự

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 6 | 6 | 6 | 6 | 3 | 8 | 7 | 0 | 2 |

1. Kinh độ - 11 ký tự

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | , | 6 | 7 | 8 | 2 | 7 | 8 | 9 | 6 |

1. Vĩ độ - 11 ký tự

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 4 | , | 5 | 8 | 9 | 6 | 5 | 7 | 5 |

1. Tốc độ - 4 ký tự

|  |  |  |  |
| --- | --- | --- | --- |
| 8 | 0 | , | 5 |

1. Mức nhiên liệu

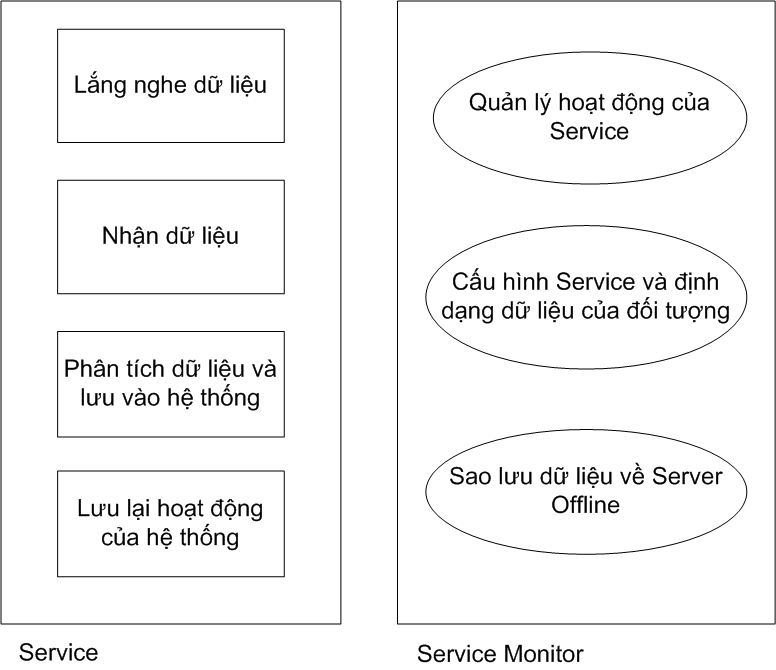
|  |  |  |  |
| --- | --- | --- | --- |
| 3 | 9 | , | 1 |

Để thực hiện nhiệm vụ này, nhóm đã xây dựng một Windows Service cho Server để có thể kết nối và lắng nghe dữ liệu qua giao thức TCP.

***Chức năng của VMS Server Service:***

* Lắng nghe dữ liệu từ đối tượng (Module GPS/GPRS)
* Đảm bảo được tính toàn vẹn của dữ liệu và hoạt động của hệ thống khi cùng lúc có nhiều đối tượng gửi dữ liệu về.
* Phân tích dữ liệu và lưu vào Database của hệ thống
* Lưu thông tin của thiết bị GPS đã kết nối để liên lạc khi cần thiết.

***Sơ đồ khối của VMS Server Service:***

******

***Thiết kế VMS Server Service:***

Để đảm bảo Server có thể cho phép nhiều đối tượng kết nối đồng thời. Trước hết, server sẽ phải thực hiện đa luồng và đảm bảo sự an toàn của luồng bởi nếu một kết nối bị gián đoạn cũng có thể làm các kết nối khác ngưng hoạt động. Thứ hai, chúng ta phải xem xét hiệu quả và khả năng mở rộng. Thứ ba, chúng ta cần phải đảm bảo rằng những kết nối bị đứt hoặc gián đoạn sẽ không ảnh hưởng tới kết nối giữa server và một đối tượng khác đang hoạt động. Hơn nữa, chúng ta cần xây dựng chức năng hạn chế tổng số đối tượng và từ chối những kết nối không tin cậy.

Lớp socket trong .Net cho phép chúng ta thực hiện những hoạt động bất đồng bộ. Về cơ bản, việc gọi một thành phần chức năng(ví dụ BeginReceive và BeginSend) là để bắt đầu hoạt động bất đồng bộ của socket. Thành phần chức năng sẽ không ngăn chặn, chúng có thể chờ đợi cho các hoạt động socket khác hoàn thành hoặc xác định một delegate (hàm con trỏ) của chức năng phản hồi và nó sẽ được gọi tự động bằng một luồng từ .Net thread pool khi các hoạt động của socket được hoàn thành. Tuy nhiên, các lớp socket thường không đảm bảo sự toàn vẹn ở mức thấp vì rất khó để xác định thời điểm, vị trí, hoạt động tạo nên lời gọi bất đồng bộ.

Để giải quyết vấn đề này, nhóm sử dụng một thành phần .Net TCP socket, trong đó có một lớp Server dựa trên các lớp socket .Net và sử dụng những lớp thread pool riêng, lớp này sẽ không chứa lời gọi tới thành phần bất đồng bộ của socket.

* + 1. XYNetServer

**Lớp Server**

XYNetServer là tên của lớp socket server mà nhóm sử dụng. Thực thể của lớp này là một thread an toàn nghĩa là có thể sử dụng cùng một thực thể từ nhiều luồng mà không gặp vấn đề gì. Cách thức hoạt động của nó như sau:

Để tạo một thực thể của lớp server, người dùng cần xác định địa chỉ IP và số của port (để kết nối tới server), cũng như số lượng tối thiểu và tối đa của những luồng sẽ được sử dụng bởi server. Khi phương thức StartServer được gọi, thực thể server sẽ bắt đầu lắng nghe yêu cầu của client trên địa chỉ IP và số port xác định. Người dùng có thể gọi phương thức SetConnectionFilter để xác định một chức năng xử lý các yêu cầu kết nối, tất cả các yêu cầu kết nối sẽ được chấp nhận nếu không có bất bỳ bộ lọc kết nối nào được thiết lập.

Server sẽ nhận hai loại thông điệp từ Client là thông điệp dạng nhị phân và thông điệp dạng chuỗi. Để xác định chức năng xử lý thông điệp gửi đến, chúng ta sẽ gọi hai phương thức SetBinaryInputHandler và SetStringInputHandler. Để gửi thông điệp, chúng ta sẽ gọi phương thức SendBinaryData hoặc SendStringData.

Trong khuôn khổ đề tài này, nhóm sử dụng phương thức truyền, nhận thông điệp dạng chuỗi.

Để chỉ định một chức năng xử lý các lỗi của server, chúng ta sẽ gọi phương thức SetExceptionHandler. Nếu không sử dụng phương thức này, tất cả các lỗi sẽ bị bỏ qua.

Sau đây là định nghĩa hàm constructor, StartServer và StopServer:

public XYNetServer(string sAddress, int nPort, int nMinThreadCount, int nMaxThreadCount);

public bool StartServer();

public void StopServer();

Tham số sAddress cho phép định địa chỉ IP của server sẽ sử dụng trong trường hợp có nhiều địa chỉ IP được gán cho máy. Nếu là null, server sẽ sử dụng địa chỉ IP mặc định của máy. sAddress và nPort sẽ xác định một server socket nội bộ. Sau khi server được bắt đầu, số lượng các thread được sử dụng bởi server sẽ nằm trong khoảng nMinThreadCount và nMaxThreadCount, con số chính xác phụ thuộc vào sức tải của server ở từng thời điểm. Nếu server không bận, một vài thread sẽ được kết thúc nhưng ít nhất nMinThreadCount vẫn sẽ hoạt động.

Phương thức StartServer thông báo cho server để bắt đầu chấp nhận và xử lý dữ liệu client gửi đến. Phương thức StopServer sẽ tắt hoàn toàn server (bao gồm tất cả những thread). Khi StartServer trả về false, server không thể được bắt đầu, trong trường hợp này chúng ta có thể gọi GetLastException để lấy đối tượng ngoại lệ được đưa ra.

Khi server socket chạy lần đầu, nó sẽ sử dụng 2 threads, một để chấp nhận kết nối của client, thread khác sẽ quét dữ liệu gửi đến. Nếu dữ liệu từ một client gửi đến hoặc xảy ra lỗi, server sẽ sử dụng những thread nội để xử lý dữ liệu hoặc xử lý các lỗi. Cần lưu ý rằng hai tham số nMinThreadCount và nMaxThreadCount xác định trong constructor không bao gồm hai thread này.

**Sử dụng những Delegates được chỉ định. (Events Handlers)**

Server socket sẽ sử dụng những delegate được chỉ định để xử lý các yêu cầu kết nối, thông báo nhận được dữ liệu và xử lý dữ liệu. Nó cũng hỗ trợ việc tùy chỉnh việc xử lý các lỗi xảy ra trong quá trình lắng nghe dữ liệu.

Server socket có 4 loại delegate là:

**Delegate Xử lý ngoại lệ**

public delegate void ExceptionHandlerDelegate(Exception oBug)

Định nghĩa delegate này nhằm mục đích xử lý những ngoại lệ xảy ra, nó sẽ được thiết lập bởi phương thức SetExceptionHandler. Tham số oBug là một đối tượng ngoại lệ biểu diễn lỗi của máy chủ. Trong khi thực hiện việc xử lý lỗi chũng ta có thể gọi phương thức GetThreadCount để lấy số lượng threads hoạt động trong vùng chứa những thread nội. Nếu số lượng thread này bằng với số lượng thread tối đa cho phép, server có thể bị deadlock trên một vài vùng tài nguyên, trong trường hợp này ta có thể chọn cách khởi động lại máy chủ ứng dụng.

Để bảo vệ server, một kết nối từ client sẽ tự động bị đóng khi xảy ra lỗi kết nối socket.

Điểm đặc biệt của delegate này là: bộ xử lý lỗi có thể được gọi đồng thời bởi những threads khác nhau từ vùng thread nội, do đó code sẽ được chạy trên những thread an toàn.

**Delegate Lọc kết nối**

Public delegate void ConnectionFilterDelegate(string sRemoteAddress, int nRemotePort, Socket sock);

Bộ xử lý này được gọi bởi bởi một thread riêng biệt để xử lý yêu cầu kết nối, nó được thiết lập bởi phương thức SetConnectionFilter. Hai tham số nRemoteAddress và nRemotePort sẽ xác định một kết nối từ client. Để từ chối một yêu cầu kết nói, chúng ta chỉ cần gọi phương thức Shutdown và sau đó gọi phương thức Close() của tham số sock, ngược lại kết nối từ client sẽ được thành lập.

Bên trong bộ xử lý này, nhóm lưu lại địa chỉ IP từ tham số sRemoteAddress vào SystemLog để có thể quản lý những client đã gửi yêu cầu tới server của hệ thống. Phương thức GetClientCount cũng được gọi để lấy tổng số Client đã kết nối, nếu số kết nối lớn hơn số kết nối tối đa cho phép, các yêu cầu kết nối tiếp theo sẽ bị từ chối.

Để tăng tính bảo mật cho hệ thống, chúng ta cũng có thể xác định một danh sách những IP được phép kết nối thông qua việc xác định địa chỉ IP từ Client.Tuy nhiên, với hệ thống mạng di động tại Việt Nam hiện nay, việc này gần như là không thể bởi địa chỉ IP qua mạng GPRS luôn thay đổi.

Giải pháp mà nhóm sử dụng là gọi phương thức Thread.Sleep() để hạn chế việc server chấp nhận quá nhiều kết nối trong một khoảng thời gian quá nhanh nhằm chống lại những cuộc tấn công từ chối dịch vụ. Bên cạnh đó, chúng ta cũng có thể yêu cầu chứng thực client thông qua tham số sock. Client có thể sử dụng tên truy cập và mật khẩu từ đối tượng sock và server sẽ kiểm tra trước khi chấp nhận kết nối của client đó. Cũng cần lưu ý rằng, việc sử lý này có thể làm chậm việc xử lý các yêu cầu kết nối khác.

**Delegate Xử lý dữ liệu dạng nhị phân**

*Public delegate void BinaryInputHandlerDelegate(string sRemoteAddress, int nRemotePort, byte[] data);*

Để thiết lập bộ xử lý này, ta sẽ gọi phương thức SetBinaryInputHandler. Khi một thông điệp nhị phân từ Client được nhận, bộ xử lý sẽ được gọi tự động bởi vùng thread nội. Vì nó có thể được thực thi đồng thời bởi nhiều thread, việc xử lý của bộ xử lý này sẽ được đảm bảo. Hai tham số sRemoteAddress và nRemotePort xác định thông số của client gửi thông điệp hiện tại. thông điệp dữ liệu pData sẽ có dạng mảng byte.

Trong trường hợp server muốn gửi một thông điệp trả lời tới Client, chúng ta có thể gọi phương thức SendBinaryData hoặc SendStringData bên trong bộ xử lý này thông qua với các tham số sRemoteAddress, nRemotePort và dữ liệu thông điệp muốn gửi.

**Delegate Xử lý dữ liệu dạng chuỗi**

*Public delegate void StringInputHandlerDelegate(string sRemoteAddress, int nRemotePort, string sData);*

Để xử lý bộ xử lý này, chúng ta sẽ gọi phương thức SetStringInputHandler. Khi một thông điệp dạng chuỗi từ Client được nhận, bộ xử lý sẽ được gọi tự động bởi một thread từ vùng thread nội. Cũng giống như Delegate Xử lý chuỗi dạng nhị phân, việc xử lý của bộ xử lý này cũng được đảm bảo và chúng ta có thể xác định được thông số của Client hiện tại qua sRemoteAddress và nRemotePort.

Dữ liệu nhận được từ phương thức này là một chuỗi sData.

Tương tự, chúng ta cũng có thể gửi thông điệp trả lời qua phương thức SendBinaryData hoặc SendStringData.

*Lưu ý: Bốn bộ xử lý này chỉ có thể được thiết lập một lần cho mỗi thực thể của server socket.*

**Thông điệp, kích thước tối đa của dữ liệu và thời gian chờ đợi**

Để gửi thông điệp tới client ta có thể sử dụng hai phương thức sau:

*Public bool SendBinaryData(string sRemoteAddress, int nRemotePort, byte[] pData);*

*Public bool SendStringData(string sRemoteAddress, int nRemotePort, string sData);*

Hai phương thức này sẽ trả về giá trị là false nếu:

* Client không được kết nối
* Kết nối socket bị gãy
* Socket đang bận (khi đang xử lý dữ liệu nhận được).

Phương thức SetMaxDataSize có thể được gọi để hạn chế kích thước của các thông điệp được gửi đến. Nếu một client gửi một kích thước dữ liệu quá lớn, kết nối sẽ bị đóng để bảo vệ server. Theo mặc định, kích thước tối đa mà server nhận được phải nhỏ hơn 4 megabytes. Với những thông điệp là chuỗi, kích thước của dữ liệu bằng hai lần chiều dài của chuỗi đó (unicode).

Thời gian chờ (timeout) có thể xảy ra trong khi nhận một thông điệp, các kết nối socket sẽ được đóng khi nó xảy ra. Để thiết lập giá trị thời gian chờ, chúng ta cần gọi phương thức SetReadTimeout. Theo mặc định, giá trị thời gian chờ của server là 30s.

*Trong khuôn khổ đề tài này, nhóm áp dụng cách thức nhận dữ liệu dạng string bởi sự đơn giản và thuận tiện trong xử lý của nó. Tuy nhiên, trong thực tế việc nhận dữ liệu dạng nhị phân sẽ giúp mã hóa được dữ liệu gửi về nhằm nâng cao tính bảo mật của hệ thống.*

* + 1. **Vehicle Management Server (VMS.Server)**

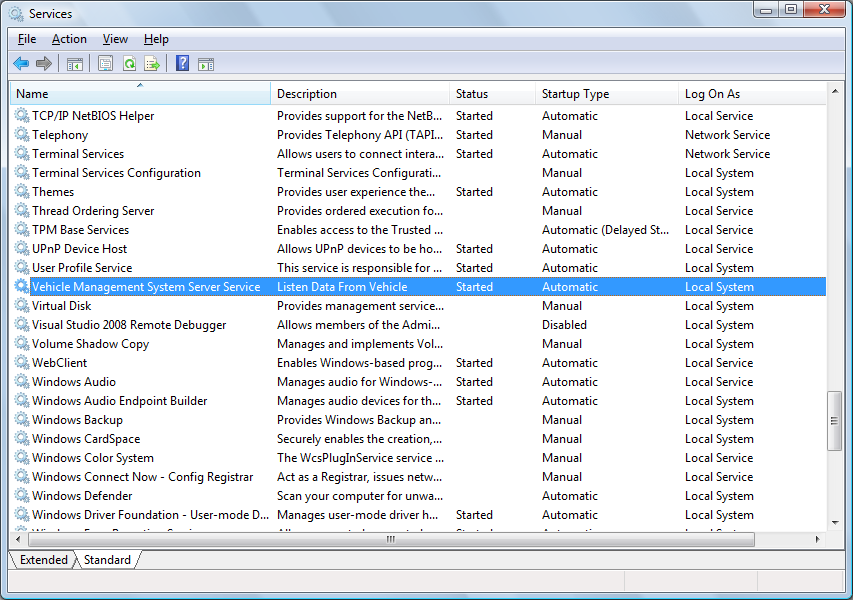
Việc xây dựng phần mềm cho Server Online cần đảm bảo được yêu cầu đề ra đối với Server này. Đó là việc lắng nghe dữ liệu và quản lý hoạt động của hệ thống:

Server bao gồm hai phần:

**Vehicle Management System Server Service:**

Là thành phần giữ vai trò chính của Server Online, nó có chức năng lắng nghe dữ liệu từ client sau đó phân tích, xử lý dữ liệu và lưu và cơ sở dữ liệu của server.

Đối với mỗi hoạt động của Server, service này sẽ lưu vào bảng System Log. Đối với những hoạt động khác của phần mềm, service này sẽ lưu vào một file text có tên là servicle.log nằm trong thư mục của chương trình



Để cấu hình các thông số của Service, nhóm sử dụng một file dạng xml để lưu các thông số này



Trong đó,

ServerIP là địa chỉ IP của server trong trường hợp server có nhiều địa chỉ IP.

Port là số của port TCP được sử dụng để lắng nghe dữ liệu từ vehicle.

LongLength là chiều dài chuỗi chứa thông tin về kinh độ

LatLength là chiều dài chuỗi chưa thông tin về vĩ độ

SpeedLength là chiều dài chuỗi chứa thông tin về tốc độ xe.

FuelLength là chiều dài chuỗi chứa thông tin về mức nhiên liệu hiện tại.

Chúng ta cần thiết lập những bộ xử lý cho Server (thông số của server là: ServerID, PortNumber, Min Connection, Max Connection) được khởi động bao gồm: Bộ xử lý ngoại lệ (ExceptionHandler), bộ xử lý dữ liệu(String Input Handler) và bộ xử lý kết nối(Connection Filter).

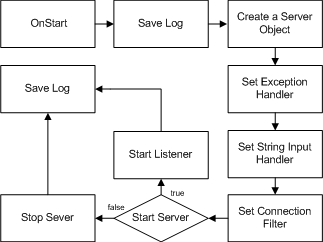
Phương thức ExceptionHandler sẽ lưu những lỗi xảy ra từ việc lắng nghe dữ liệu này tới bảng SystemLog.

Phương thức StringInputHandler sẽ nhận dữ liệu từ client, phân tích dữ liệu theo định dạng đã được cài đặt và lưu vào cơ sở dữ liệu những thông tin liên quan đến hoạt động hiện tại của xe và module GPS.

Phương thức ConnectionFilter có nhiệm vụ lọc những kết nối không an toàn cũng như đảm bảo hoạt động thông suốt của Server khi có quá nhiều yêu cầu kết nối được gửi đến.

Phương trức StartServer() của đối tượng server sẽ thực hiện việc khởi động server. Phương thức này sẽ trả về true khi server khởi động thành công. Ngược lại, server sẽ không thể khởi động hoặc khởi động bị lỗi. Lỗi này sẽ được xử lý trong phương thức ExceptionHandler.

Sơ đồ khối lập trình khởi động server:



**Save Data**

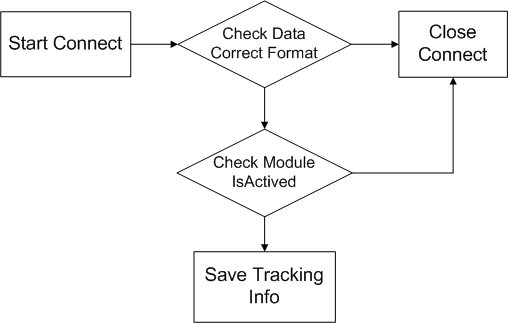
Khi một kết nối đã được khởi tạo, Phương thức StringInputHandler sẽ thực hiện việc kiểm tra và lưu dữ liệu.

Dữ liệu sẽ được phân tích và đối chiếu với định dạng đã được định nghĩa của dữ liệu. Nếu không đúng định dạng do server định nghĩa, kết nối sẽ bị ngắt ngay lập tức.

Dữ liệu sau khi phân tích sẽ cho chúng ta một giá trị là ID của module GPS/GPRS (số SIM). Server sẽ thực hiện việc kiểm tra sự cho phép hoạt động đối với module này thông qua trường IsActived trong cơ sở dữ liệu. Nếu Module không có trong danh sách hoặc không được kích hoạt, kết nối cũng sẽ bị ngắt ngay lập tức.

Cuối cùng, dữ liệu nhận được khi đã qua những bước kiểm tra bao gồm thông tin hiện tại mà xe gửi về như: tọa độ, tốc độ và mức nhiên liệu hiện có.

Sơ đồ khối:

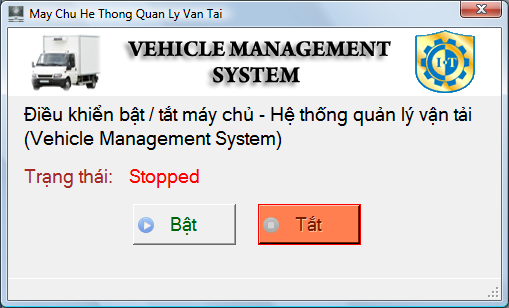


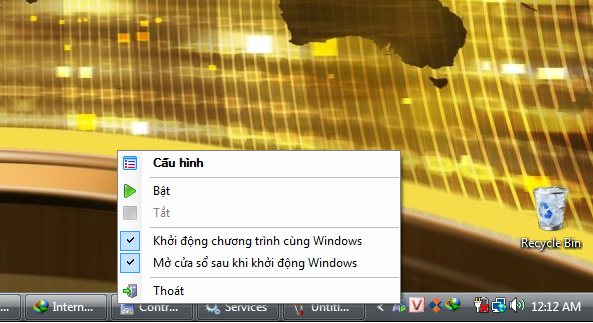
**Vehicle Management System Server Monitor (VMS Monitor)**

Phần mềm này có chức năng chính là quản lý hoạt động của Service, nhằm giúp người quản lý server có thể vận hành hệ thống một cách dễ dàng và trực quan.

Chức năng của VMS Monitor bao gồm:

* Start/Stop Service
* Quản lý Service bởi một ứng dụng tại symtray của Windows
* Cho phép ứng dụng khởi động cùng windows.





Phần mềm sẽ điều khiển Service thông qua một đối tượng ServiceController. Đối tượng này xác định Service bởi thuộc tính Service Name.

*ServiceController.ServiceName = “VMSServerService”;*

Một bộ Timer sẽ được sử dụng để kiểm tra trạng thái của Servier qua thuộc tính Status của đối tượng ServiceControl.

Một đối tượng ServiceController bao gồm các phương thức quản lý Servicer như sau:

* Start() – Khởi động Service
* Stop() – Dừng Service
* Pauser() – Tạm dừng Service
* Close() – Đóng Service
* Continue() – Cho phép Service tiếp tục hoạt động.
  1. Vehicle Management System Manager