**MỤC LỤC**

[**MỞ ĐẦU** 4](#_Toc294816122)

[**I.** **Cơ sở lí thuyết Remote Execution and Distribution Computing** 5](#_Toc294816123)

[***1.*** ***Telnet*** 5](#_Toc294816124)

[*1.1.* *Telnet Operation* 5](#_Toc294816125)

[*1.2.* *Network Virtual Terminal* 6](#_Toc294816126)

[*1.3.* *Telnet Options* 6](#_Toc294816127)

[*1.4.* *Telnet Command Structure* 7](#_Toc294816128)

[*1.5.* *Option Negotiation* 8](#_Toc294816129)

[*1.6.* *Telnet Basic Commands* 9](#_Toc294816130)

[*1.7.* *Telnet Emulation* 9](#_Toc294816131)

[*1.8.* *TN3270 Enhancements* 11](#_Toc294816132)

[*1.9.* *Device-type Negotiation* 11](#_Toc294816133)

[***2.*** ***Giao thưc dòng lệnh điều khiển truy cập từ xa (REXEC and RSH)*** 13](#_Toc294816134)

[***3.*** ***Môi trường tính toán phân tán (Distributed Computing Enviroment – DCE)*** 13](#_Toc294816135)

[*3.1.* *DCE Directory Service* 14](#_Toc294816136)

[*3.2.* *Authentication Service* 18](#_Toc294816137)

[*3.3.* *DCE Threads* 19](#_Toc294816138)

[*3.4.* *Distributed Time Service* 20](#_Toc294816139)

[***4.*** ***Distributed File Service*** 21](#_Toc294816140)

[*4.1.* *File Naming* 23](#_Toc294816141)

[*4.2.* *Hiệu suất* 23](#_Toc294816142)

[**II.** **Khảo sát thực tế với nền tảng WCF trên Web Service** 24](#_Toc294816143)

[***1.*** ***Khái niệm cơ bản*** 24](#_Toc294816144)

[*1.1.* *WCF là gì?* 25](#_Toc294816145)

[*1.2.* *Tại sao lại sử dụng WCF?* 25](#_Toc294816146)

[***2.*** ***Kiến trúc tổ chức*** 26](#_Toc294816147)

[*2.1.* *Các hiệp nghị (Contracts)* 26](#_Toc294816148)

[*2.2.* *Dịch vụ thực thi (Runtime Service)* 27](#_Toc294816149)

[*2.3.* *Thông điệp/Bản tin (Message)* 28](#_Toc294816150)

[*2.4.* *Lưu trữ và kích hoạt (Host and Activation)* 29](#_Toc294816151)

[***3.*** ***Các tính năng đặc trưng*** 29](#_Toc294816152)

[*3.1.* *Giao dịch (Transaction)* 29](#_Toc294816153)

[*3.2.* *Lưu trữ (Host)* 29](#_Toc294816154)

[*3.3.* *Bảo mật (Security)* 30](#_Toc294816155)

[***4.*** ***Dịch vụ trong WCF (Services)*** 30](#_Toc294816156)

[*4.1.* *Định kiểu (Typed Service)* 30](#_Toc294816157)

[*4.2.* *Không định kiểu (Untyped Service)* 31](#_Toc294816158)

[*4.3.* *Bản tin định kiểu (Typed Message Service)* 31](#_Toc294816159)

[***5.*** ***Kiến trúc Client trong WCF*** 31](#_Toc294816160)

[*5.1.* *Khái niệm cơ bản* 31](#_Toc294816161)

[*5.2.* *Các đối tượng phía Client* 32](#_Toc294816162)

[5.2.1. ICommunicationObject 33](#_Toc294816163)

[5.2.2. IExtensibleObject 34](#_Toc294816164)

[*5.3.* *Các phương thức liên lạc của Client* 35](#_Toc294816165)

[5.3.1. Một chiều (One way) 35](#_Toc294816166)

[5.3.2. Yêu cầu – Hồi đáp (Request – Reply) 36](#_Toc294816167)

[5.3.3. Song công (Duplex) 36](#_Toc294816168)

[5.3.4. Dị bộ (Asynchronous) 37](#_Toc294816169)

[***6.*** ***Mô hình lập trình WCF*** 37](#_Toc294816170)

[*6.1.* *Lựa chọn phương pháp tiếp cận* 37](#_Toc294816171)

[*6.2.* *Mô hình dịch vụ* 38](#_Toc294816172)

[*6.3.* *Các phương pháp lập trình với WCF* 39](#_Toc294816173)

[6.3.1. Phương pháp khai báo (Declarative Programming) 39](#_Toc294816174)

[6.3.2. Phương pháp lập trình trực tiếp (Explicit Programming) 39](#_Toc294816175)

[6.3.3. Phương pháp sử dụng tập tin cấu hình (XML File Config) 40](#_Toc294816176)

[***7.*** ***Bảo mật trong WCF (Security)*** 40](#_Toc294816177)

[*7.1.* *Ý nghĩa bảo mật trong WCF* 40](#_Toc294816178)

[7.1.1. Tích hợp với các kiến trúc bảo mật có sẵn 41](#_Toc294816179)

[7.1.2. Tích hợp với mô hình xác thực có sẵn 41](#_Toc294816180)

[7.1.3. Làm việc liên môi trường 42](#_Toc294816181)

[*7.2.* *Các lĩnh vực bảo mật của WCF* 42](#_Toc294816182)

[7.2.1. Transfer Security 42](#_Toc294816183)

[7.2.2. Các chế độ bảo mật vận chuyển và bản tin 43](#_Toc294816184)

[7.2.3. Điều khiển truy nhập 43](#_Toc294816185)

[7.2.4. Auditing 43](#_Toc294816186)

[**III.** **Ứng dụng kiểm thử** 44](#_Toc294816187)

[***1.*** ***Mô tả*** 44](#_Toc294816188)

[***2.*** ***Các chức năng*** 44](#_Toc294816189)

[*2.1.* *Quản lí môn học* 44](#_Toc294816190)

[*2.2.* *Quản lí lớp học* 45](#_Toc294816191)

[*2.3.* *Quản lí tuyển sinh* 46](#_Toc294816192)

[*2.4.* *Quản lí đào tạo* 47](#_Toc294816193)

[*2.5.* *Quản lí công việc/lịch hẹn* 48](#_Toc294816194)

[*2.6.* *Quản lí báo cáo định kì* 49](#_Toc294816195)

[*2.7.* *Quản lí điều hành* 50](#_Toc294816196)

[**KẾT LUẬN** 52](#_Toc294816197)

[**TÀI LIỆU THAM KHẢO** 53](#_Toc294816198)

[**PHÂN CÔNG CÔNG VIỆC** 54](#_Toc294816199)

**MỞ ĐẦU**

Ngày nay, mạng Internet phát triển ngày càng mạnh mẽ, với một tốc độ nhanh chóng và phạm vi rộng lớn. Cơ sở hạ tầng mạng được cải tiến, nâng cấp cả về số lượng và chất lượng. Đồng thời, các ứng dụng hoạt động trên mạng cũng được những nhà phát triển tập trung cải thiện, mở rộng chức năng, nâng cao hiệu suất và đặc biệt là tiêu tốn chi phí tài nguyên tối thiểu.

Việc phát triển qui mô mạng Internet nói chung cũng kéo theo một mạng lưới rất rộng lớn các thiết bị, thành phần liên kết với nhau. Tương ứng, đó là những ứng dụng trao đổi thông tin liên tục với lượng dữ liệu rất lớn. Tuy nhiên, khi lượng dữ liệu được trao đổi càng cao thì việc đảm bảo đường truyền lại là một bài toán hết sức nan giải. Đối với các nhà phát triển, việc định nghĩa ra một chuẩn giao tiếp giữa các ứng dụng trên mạng hết sức cần thiết. Điều đó giúp tạo ra một qui trình chuẩn để xây dựng các ứng dụng mạng, giảm bớt rắc rối, rườm ra, tiết kiệm chi phí trong việc tổ chức cơ sở hạ tầng cũng như tăng cường tính tương tác giữa nhiều hệ thống khác nhau. Sau khi tìm hiểu về vấn đề này, nhóm chúng em quyết định chọn đề tài: Các giao thức điều khiển, giao tiếp và tính toán trên mạng phân tán – **remote execution and distribution computing**. Toàn bộ nội dung của bài tiểu luận sẽ tập trung trình bày các phương thức trao đổi, giao tiếp giữa các ứng dụng mạng thông qua các giao diện chuẩn được xây dựng. Ngoài ra, nhóm cũng đề xuất một công nghệ mới nhằm hiện thực hóa ý tưởng xây dựng chuẩn giao tiếp trên mạng, đó là nền tảng WCF (Windows Communication Foundation) hoạt động dựa trên Web Service. Đây là mảng kiến thức hết sức thú vị và có ý nghĩa thực tế đối với sinh viên chuyên ngành truyền thông và mạng máy tính.

Trong suốt quá trình thực hiện bài tiểu luận, nhóm đã hết sức nỗ lực để tìm hiểu, trau dồi các kiến thức chuyên môn cần thiết. Tuy nhiên, báo cáo không thể tránh khỏi những thiếu xót và cần tiếp tục cải thiện. Nhóm xin chân thành gửi lời cảm ơn TS. Phạm Huy Hoàng đã tận tình hướng dẫn, giúp đỡ nhóm hoàn thành tốt bài tiểu luận này.

*Nhóm sinh viên*

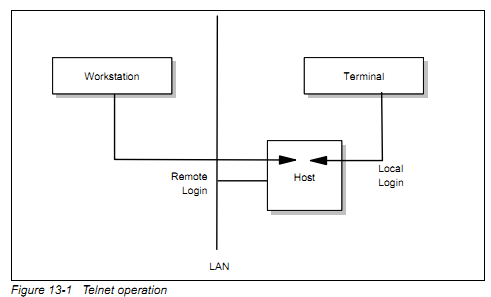
1. **Cơ sở lí thuyết Remote Execution and Distribution Computing**

Một trong những cơ chế cơ bản nhất của mạng máy tính là khả năng điều khiển thực thi từ xa (remote execute), hay nói cách khác, người dùng có thể gọi một ứng dụng từ một máy khác.

1. ***Telnet***

Telnet là giao thức tiêu chuẩn với số STD là 8. Trạng thái của giao thức là khuyến nghị. Telnet được mô tả trong RFC 854 – Telnet protocol specifications và trong RFC 855 – Telnet Option Specifications.

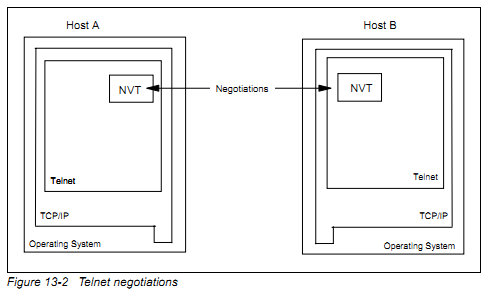
Telnet cung cấp một giao diện chuẩn mà thông qua đó, chương trình trên máy client (telnet client) có thể truy cập được các tài nguyên từ một máy chủ khác (telnet server). Telnet được dùng trên cả mạng LAN và WAN



* 1. *Telnet Operation*

Phương thức Telnel dựa trên 3 ý tưởng:

* Khái niệm mạng đầu cuối ảo – Network Virtual Terminal. Đây là một thiết bị ảo gồm một cấu trúc cơ bản chung cho các thiết bị đầu cuối thực. Mỗi một máy chủ lưu trữ các bản đồ các đặc trưng của thiết bị đầu cuối cho những người dùng NVT và giả định rằng mỗi máy chủ khác cũng sẽ làm như vậy.
* Quan điểm đối xứng của các thiết bị đầu cuối và các tiến trình.
* Thoả thuận về các tuỳ chọn của thiết bị đầu cuối. Sự lựa chọn các nguyên tắc thoả thuận sẽ được sử dụng bởi giao thức telnet, bởi ngoài những lựa chọn có sẵn trong NVT, rất nhiều các máy chủ khác cũng muốn cung cấp các dịch vụ bổ sung. Client và server sẽ sử dụng một tập hợp các quy ước để thiết lập kết nối telnet thông qua các cơ chế "làm, không làm, sẽ, sẽ không". Hai máy chủ bắt đầu bằng việc kiểm tra sự hiểu biết lẫn nhau của mỗi máy. Sau khi những thoả thuận tối thiểu hoàn tất, mỗi máy có thể làm việc ở mức tối thiểu với NVT. Ngoài ra, để phán ánh chính xác hơn về các phần cứng thực tế được sử dụng cũng như mở rộng khả năng của NVT, cả hai máy có thể tuỳ chọn các bổ sung. Theo quan điểm đối xứng, cả máy chủ (host) và máy client đều có thể đề xuất bổ sung các tuỳ chọn.



* 1. *Network Virtual Terminal*

Một NVT có một máy in và một bàn phím, những lệnh được gõ từ bàn phím sẽ được gửi tới kết nối telnet và máy in nhận những lệnh đó

* 1. *Telnet Options*

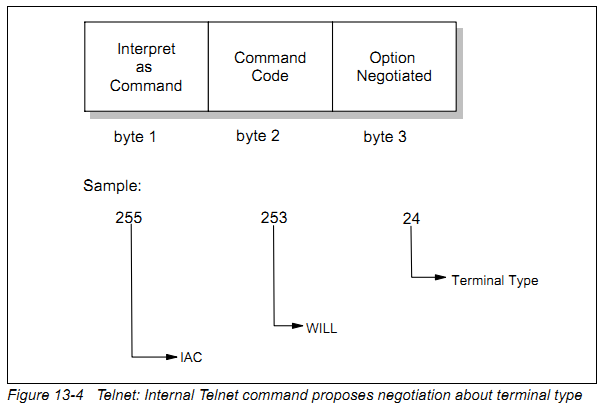
Có rất nhiều các tuỳ chọn cho telnet, được mô tả kỹ càng ở trong STD1. Một số tuỳ chọn thông dụng như:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Số hiệu** | **Tên** | **Trạng thái** | **RFC** | **STD** |
| 0 | Binary transmission | Standard | 856 | 27 |
| 1 | Echo | Standard | 857 | 28 |
| 3 | Suppress Go Ahead | Standard | 858 | 29 |
| 5 | Status | Standard | 859 | 30 |
| 6 | Timing mark | Standard | 860 | 31 |
| 255 | Extended options list | Standard | 861 | 32 |
| 34 | Linemode | Draft | 1184 |  |
| 2 | Reconnection | Proposed |  |  |
| 4 | Approximate, message size negotiation | Proposed |  |  |
| 7 | Remote controlled trans and echo | Proposed | 726 |  |
| 8 | Output line width | Proposed |  |  |
| 9 | Output page size | Proposed |  |  |
| 10 | Output carriage-return dispostion | Proposed | 652 |  |
| 11 | Output horizontal tab stops | Proposed | 653 |  |
| 12 | Output horizontal tab disposition | Proposed | 654 |  |
| 13 | Ouput from feed disposition | Proposed | 655 |  |
| 14 | Output vertical tab stops | Proposed | 656 |  |
| 15 | Output vertical tab disposition | Proposed | 657 |  |
| 16 | Output line feed disposition | Proposed | 658 |  |
| 17 | Extended ASCII | Proposed | 698 |  |
| 18 | Logout | Proposed | 727 |  |
| 19 | Byte macro | Proposed | 735 |  |
| 20 | Data entry terminal | Proposed | 1043 |  |
| 21 | SUPDUP | Proposed | 736 |  |
| 22 | SUPDUP output | Proposed | 749 |  |
| 23 | Send location | Proposed | 779 |  |
| 24 | Terminal type | Proposed | 1091 |  |
| 25 | End of record | Proposed | 885 |  |
| 26 | TACACS user identification | Proposed | 927 |  |
| 27 | Output marking | Proposed | 933 |  |
| 28 | Terminal location number | Proposed | 946 |  |
| 29 | Telnet 3270 regime | Proposed | 1041 |  |
| 30 | X.3 PAD | Proposed | 1053 |  |
| 31 | Negotiate window size | Proposed | 1073 |  |
| 32 | Terminal speed | Proposed | 1079 |  |
| 33 | Remote flow control | Proposed | 1372 |  |
| 35 | X Display location | Proposed | 1096 |  |
| 39 | Telnet environment option | Proposed | 1572 |  |
| 40 | TN3270 enhancements | Proposed | 1647 |  |
| 37 | Telnet authentication option | Experimental | 1416 |  |
| 41 | Telnet xauth | Experimental |  |  |
| 42 | Telnet charset | Experimental | 2066 |  |
| 43 | Telnet remote serial port | Experimental |  |  |
| 44 | Telnet com port control | Experimental | 2217 |  |

* 1. *Telnet Command Structure*

Việc liên lạc giữa máy client và máy chủ (server) được thực hiện bởi các lệnh nội tại, người dùng không thể truy cập các lệnh này. Các lệnh nội tại là một chuỗi gồm 2 tới 3 byte tuỳ thuộc vào lệnh.

Mỗi một Interpret As Command đều được gắn kèm một mã lệnh Command Code. Nếu có thêm lựa chọn thoả thuận (Option Negotiated), lựa chọn này sẽ được gắn vào cuối khối – byte thứ ba.



Một số mã lệnh thông dụng:

|  |  |  |
| --- | --- | --- |
| **Lệnh** | **Mã** | **Mô tả** |
| SE | 240 | Kết thúc các tham số của sub-negotiation. |
| NOP | 241 | Không có lệnh nào. |
| Data Mark | 242 | Đồng bộ một phần dòng dữ liệu. Đi kèm với một thông báo TCP khẩn cấp. |
| Break | 243 |  |
| Go Ahead | 249 | Tín hiệu GA. |
| SB | 250 | Bắt đầu một tuỳ chọn sub-negotiation được chỉ ra ngay ở dòng lệnh sau. |
| WILL | 251 | Yêu cầu sử dụng, xác nhận việc sử dụng, tuỳ chọn được chỉ định bởi mã ngay sau lệnh này. |
| WON | 252 | Hiển thị từ chối sử dụng hoặc tiếp tục sử dụng các tuỳ chọn. |
| DO | 253 | Yêu cầu sử dụng bên kia, hoặc xác nhận đã sẵn sàng cho yêu cầu. |
| DONT | 254 | Yêu cầu các bên ngừng sử dụng, hoặc xác nhận không đồng ý việc tiếp tục cho các bên sử dụng. |
| IAC | 255 | Interpret as Command, chỉ ra ngay phía sau là lệnh (command) chứ không phải dữ liệu. |

* 1. *Option Negotiation*

Telnet có thể thoả thuận việc sử dụng các tuỳ chọn cho mỗi máy chủ bằng việc sử dụng các lệnh nội tại. Cơ sở đầu tiên của việc thoả thuận là mỗi host đều phải kết nối và đồng ý với các thoả thuận tối thiểu, các lệnh được sử dụng trong thoả thuận là: WILL, WON, DO, DON’T. Ngoài ra, một số các tuỳ chọn phụ sử dụng các câu lệnh SB, SE để quản lí các thương lượng phụ. Các thoả thuận được mô tả như sau:

|  |  |  |
| --- | --- | --- |
| **Gửi** | **Trả lời** | **Ý nghĩa** |
| DO transmit binary | WILL transmit binary |  |
| DO window size | WILL window size | Thoả thuận về kích cỡ cửa sổ. |
| SB window size 0 80 0 24 SE |  | Xác định kích cỡ cửa sổ. |
| DO terminal type | WILL terminal type | Thoả thuận về kiểu của thiết bị đầu cuối. |
| SB terminal type SE |  | Yêu cầu gửi đặc trưng (mã) của thiết bị đầu cuối. |
|  | SB terminal type SE  IBM = 3278-2 SE | Mã của thiết bị là 3278-2 |
| DO echo | WONT echo |  |

* 1. *Telnet Basic Commands*

Mục tiêu chính của các giao thức Telnet là cung cấp một giao diện chuẩn cho các máy thực hiện. Để bắt đầu thiết lập kết nối, giao thức Telnet định dạng một số hàm sau:

* IP: Interrupt process – tiến trình interrupt
* AO: Abort output – từ chối đầu ra
* AYT: Are you there
* EC: Erase character – xoá ký tự
* EL: Erase line – xoá dòng
* SYNCH: Synchronize – đồng bộ
  1. *Telnet Emulation*

Telnet có thể được sử dụng để thực hiện một kết nối TCP/IP tới máy chủ SNA. Tuy nhiên, telnet 3270 được sử dụng để cung cấp 3270 telnet emulation. Telnet 3270 bổ sung thêm các tuỳ chọn đặc biệt:

* 3270 terminal emulation sử dụng chế độ khối hơn chế độ dòng
* 3270 terminal emulation sử dụng EBCDIC hơn ASCII
* 3270 terminal emulation sử dụng các khoá đặc biệt, như ATTN và SYSREQ

Các kết nối telnet 3270 được thực hiện thông qua các ba tuỳ chọn thoả thuận:

* Kiểu thiết bị đầu cuối
* Truyền nhị phân
* Kết thúc bản ghi

Một máy chủ 3270 phải hỗ trỡ các đặc điểm này trong suốt thời gian phiên làm việc đầu tiên. Các tuỳ chọn truyền nhị phân hoặc kết thúc bản ghi tuỳ chọn có thể được gửi theo thứ tự trong suốt quãng thời gian thoả thuận. Sau khi một kết nối TN3270 được thiết lập, hai bên có thể thực hiện được các tuỳ chọn bổ sung. Các tuỳ chọn này là TELNET-REGIME, SUPPRESS-GO-AHEAD, ECHO, TIMING-MARK

Thiết bị đầu cuối được đặc trưng bởi một chuỗi ký tự, ví dụ như IBM 3278-3. -3 phía sau 3278 thể hiện việc sử dụng màn hình thay thế khác so với màn hình tiêu chuẩn 24x80. Tuỳ chọn truyền nhị phân cũng khác với chế độ NVT ban đầu, client và server có thể chuyển đổi giữa hai phương thức bằng cách gửi một lệnh để vô hiệu hoá việc truyền nhị phân. Một dòng dữ liệu 3270 bao gồm lệnh và dữ liệu liên quan, độ dài của dữ liệu có thể khác nhau, lệnh và dữ liệu được phân tách bởi IAC EOR. Các tuỳ chọn EOR telnet được sử dụng trong suốt quá trình thoả thuận.

Một vấn đề quan trọng khác của TN3270 là việc xử lý chính xác các chức năng trong ATTN và SYSREQ. Khoá 3270 ATTN được sử dụng trong SNA để gián đoạn các tiến trình còn khoá 3270 SYSREQ sử dụng để chấm dứt phiên làm việc mà không cần đóng kết nối. Lệnh SYSREQ và ATTN được gửi trực tiếp tới máy chủ TN3270 thông qua kết nối telnet. Hầu hết các máy chủ TN3270 đều chuyển lệnh BREAK thành một yêu cầu ATTN tới host thông qua mạng SNA. Ở phía client, một phím hoặc tổ hợp phím sẽ được liên kết với BREAK. Còn với lệnh SYSREQ, một lệnh ngắt tiến trình – telnet interrupt process sẽ được gửi hoặc lệnh SYSREQ được đính kèm vào dữ liệu của TN3270. Tương tự như trên, ở phía client cũng có một phím hoặc một tổ hợp phím cho lệnh SYSREQ.

Dưới đây là một số vấn đề về các chức năng mà TN3270 không xử lý được:

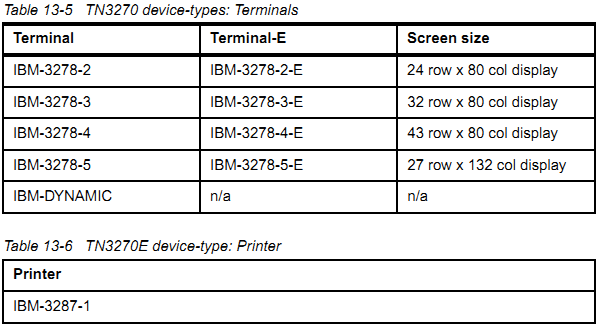
* TN3270 không hỗ trợ máy in 328x.
* TN3270 không thể xử lý được thông tin SNA BIND.
* Không hỗ trợ các cho các tiến trình đáp ứng SNA positive/negative.
* TN3270 không hỗ trợ liên kết phiên làm việc với tên thiết bị SNA.
  1. *TN3270 Enhancements*

Cấu trúc của 3270 cho phép những dữ liệu không theo chuẩn 3270 được tương tích với chuẩn dữ liệu 3270. Trường có cấu trúc bao gồm một hoặc nhiều khối lệnh. Tuy nhiên, không phải mọi client TN3270 đều hỗ trợ mọi loại dữ liệu, vì vậy, để client có thể sử dụng tất cả chức năng, cần xác định phạm vi của các loại dữ liệu khi mà kết nối telnet được thành lập. Để khắc phục những điều đó, TN3270 mở rộng các thuộc tính được xác định – TN3270E.

Nếu muốn sử dụng các mở rộng, cả client và server đều phải được hỗ trợ TN3270E. Nếu một trong hai bên không hỗ trợ TN3270E thì TN3270 sẽ được sử dụng. Sau khi cả hai bên đồng ý sử dụng TN3270E, cả hai máy bắt đầu các thoả thuận. Các tuỳ chọn này có trong một vài kiểu thiết bị:

* Kiểu dòng dữ liệu máy in.
* Thông tin trạng thái của thiết bị.
* Thông qua thông tin BIND từ server tới client.
* Các đáp ứng chủ động/bị động.
  1. *Device-type Negotiation*

Tên của thiết bị là một chuỗi ký tự ASCII gồm cả chữ thường và chữ hoa. Khi máy chủ TN3720E nhận lệnh DEVICE-TYPE REQUEST từ máy client, máy chủ trả lời bằng kiểu thiết bị, tên thiết bị, tên tài nguyên. Một số kiểu thiết bị và tên như sau:



Với 3278 và 3287sử dụng các thiết bị cơ bản, kiểu thiết bị giới hạn trong 3278 và 3287kèm các loại máy in để đơn giản cho các thoả thuận. Nhưng điều này không có nghĩa là các loại thiết bị khác không sử dụng được. Các loại đầu cuối có những thoả thuận chung của thiết bị 3270 vẫn tương thích.

Hậu tố -E cho phép sử dụng các thuộc tính mở rộng như phân vùng, đồ hoạ, màu sắc và các bộ ký tự thay thế. Nếu cả server và client đều đồng ý sử dụng các thuộc tính mở rộng đồng thời đạt được thoả thuận trong việc sử dụng các thiết bị có hầu tố -E, cả hai bên cần phải có khả năng xử lý các trường 3270 có cấu trúc. Các trường này cho phép các máy client sử dụng 3270 sử dụng các ứng dụng chứa trên máy server 3270.

Không phải lúc nào máy client TN3270E cũng có thể dễ dàng để biết tên của thiết bị có sẵn trong mạng. Để giải quyết vấn đề này, các máy server TN3270E phải gán các thiết bị thích hợp cho các client bằng cách sử dụng một tập các thiết bị được định nghĩa ngay trên server. Cơ bản, các tập các thiết bị chứa các thiết bị của mạng SNA, các thiết bị đầu cuối, máy in. Ngoài ra, tập các thiết bị còn cung cấp các chức năng quan trọng nhất cho một phiên làm việc:

* Cho phép các thiết bị dùng chung một máy in được xác định sẵn.
* Chỉ định một nhóm các thiết bị cho một tổ chức cụ thể.
* Định nghĩa quyền truy cập cho một số ứng dụng trên máy chủ.

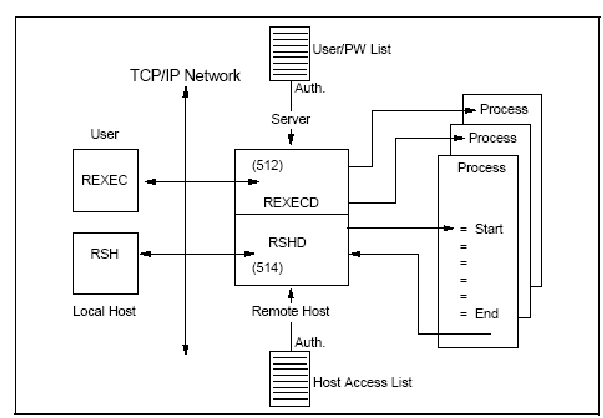
1. ***Giao thưc dòng lệnh điều khiển truy cập từ xa (REXEC and RSH)***

REXECD là một server mà cho phép thực thi các công việc đăng nhập từ một Host ở xa thông qua giao thức TCP/IP.Client thường sử dụng các câu lệnh REXEC hay RSH để truy nhập vào hệ thống.Bất cứ một kết quả truy nhập nào hay lỗi trong quá trình truy nhập cũng được gửi trở lại cho máy khác hoặc thực hiện cho tới khi thành công.

Nguyên tắc hoạt động:

REXECD là một server.Nó tiếp quản lý các câu lệnh được cung cấp cho Host bên ngoài và chuyển các câu lệnh cho các máy ảo thứ cấp của nó để thực thi.Server tự đông thực hiện đăng nhập cho người dùng và xác thực người dùng khi họ nhập vào ID và password.

Các câu lênh REXEC thường định nghĩa một ID,password,địa chỉ Host và yêu cầu xử lý để bắt đầu truy cập từ xa vào hệ thống.Tuy nhiên RSH không cần ta gửi username,pass mà nó thay thế bằng file của host truy cập(Trong đó có chứa đầy đủ các thông tin ).Cả Server và Client kết nối với nhau thông qua giao thức TCP/IP.REXEC sử dụng TCP qua cổng 512 RSH sử dụng cổng 514, được mô tả ở hình sau:



Nguyên lý hoạt động của REXEC,RSH và REXECD server.

1. ***Môi trường tính toán phân tán (Distributed Computing Enviroment – DCE)***

DCE là một kiến trúc ,là một tập các chuẩn service mở và giao diện API mà hỗ trợ cho người lập trình và các nhà phát triển dự án trong các ứng dụng phân tán trong môi trường đa nền tảng,đa người dùng.

DCE là kết quả làm việc của OSF(Open Group) là sự cộng tác của nhiều nhà cung cấp phần cứng,phần mềm,khách hàng,các chuyên viên cố vấn.OSF bắt đầu thực hiện vào năm 1988 với mục đích hỗ trợ nghiên cứu,phát triển ,phân phối các công nghệ neural và các chuẩn công nghiệp.DCE version 1 được công bố năm 1992.

Các dịch vụ chính của DCE bao gồm:

-Thông tin các đối tượng.

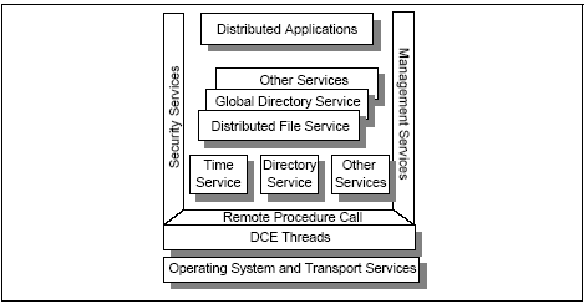
-Dịch vụ an ninh.

-Dịch vụ thời gian phân tán.

-Dịch vụ file phân tán.

-Các luồng.

-RPC(dịch vụ truy cập từ xa).



Kiến trúc các bộ phân của DCE

* 1. *DCE Directory Service*

Khi làm việc trong một khu vực rộng lớn ,nhất là trong môi trường mạng phức hợp,một điều quan trọng là giữ thông tin về các vị trí ,tên ,và các dịch vụ (và nhiều chi tiết khác) của những người tham gia hệ thống cũng như các tài nguyên trên mạng.Điều này rất quan trọng để ta có thể truy cập thông tin một cách dễ dàng.Để thực hiện được điều này thông tin cần được lưu trữ trên một vị trí trung tâm logic và có thể truy cập thông tin qua các giao diện chuẩn.DCE Cell Directory Service cho phép thực hiện việc này.

DCE Directory bao gồm các bộ phận sau:

-Cell Directory Service (CDS)

-Global Directory Service (GDS)

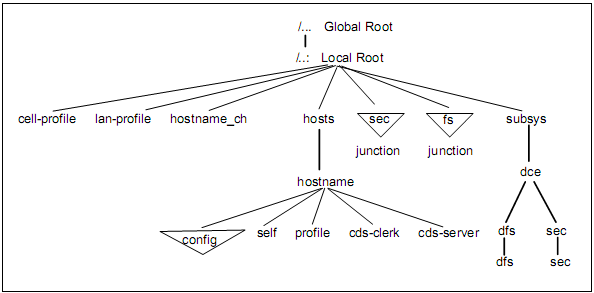
-Global Directory Agent (GDA)

-Application program interface (API)

Cell Directory Service:Quản lý cơ sở dữ liệu lưu trữ các thông tin về tài nguyên trong một nhóm các Host làm việc cùng nhau mà được gọi là các cell.Một DCE cell có tính co giãn và có thể bao gồm hàng trăm thực thể.Tiêu biểu là các tập đoàn có nhiều tổ chức trong một cell.Các cơ sở dữ liệu bao gồm cây phân cấp các tên tượng trưng cho các máy logic ,các ứng dụng ,các ứng dụng,người sử dụng và các tài nguyên trong cell.Các tên này thường nằm trong các thư mục bộ phận.Cây phân cấp tên này cũng thường gọi là không gian têm.Tất cả các Cell chứa ít nhất một Directory Service.

CDS gồm hai đặc tính quan trọng là:nó có thể được phân tán cũng có thể được nhân bản.Phân tán có nghĩa là các thực thể dữ liêu không được cư trú trên một máy nào trong một cell.Các dữ liệu có thể chia thành nhiều đoạn và cu trú trên nhiều máy riêng.

CDS chỉ có thể được sử dụng khi nó xác thực với DCE security service để ngăn chặn sự giả mạo.



Cấu trúc phân cấp của không gian tên CDS

Không phải các DCE name đều được lưu trữ trên DCE directory service.Một số tài nguyên có thể được quản lý trên vài service ví dụ như Security service và được phân tán trên các file hệ thống kết nối với CDS qua một bộ phận đặc biệt gọi là Junction.Một junction bao gồm nhiều các thông tin ẩn giấu và một Client có thể kết nối tới nó thông qua directory Service

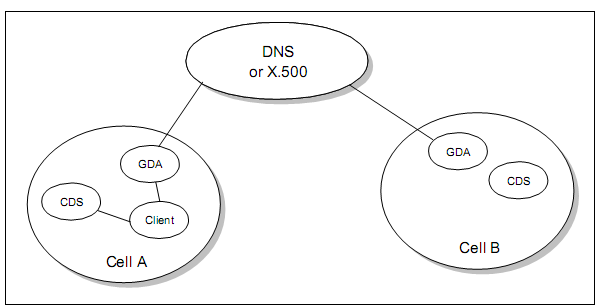
Global Directory Service and Agent.

CDS chỉ trả lời vị trí các tài nguyên trong cell mà nó quản lý.Tuy nhiên trong mạng có thể có rất nhiều cell.Vì thế người ta đưa ra Global Directory Namespace mà là cây phân cấp của các cell name. Global Directory Service (GDS) có khả năng phân giải cho chúng ta các tài nguyên mà nằm ngoài hệ thống.Ví dụ như trong trường hợp một công ty muốn kết nối với các hệ thống ngoài phạm vi của nó trên mạng internet.

Để tìm kiếm các cell khác cần có sự trao đổi thông tin giữa các cell.hiện nay có hai cách để thực hiện điều này:  
-CCITT X.500

-Internet Domain Name Services (DNS)

Để nhiều cell có thể giao tiếp với nhau một bộ phận khác cần được đưa ra Global Directory Agent.GDA là trung gian giữa local cell và GDS.Trong ví dụ trong hình dưới CDS không biết vị trí của tài nguyên nào đó bên ngoài nó yêu cầu client hỏi GDA.GDA kết nối tới không gian tên GDS và trả về kết quả.

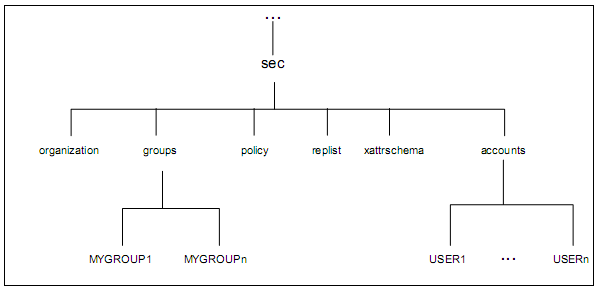


DCE GDA(Global Directory Agent)

DCE Security Service.

Vấn đề an ninh luôn là một vấn đề trở ngại trong môi trường mạng.Nói một cách rộng hơn đó là môi trường hệ phân tán,vấn đề chủ yếu là đảm bảo tất cả các bên tham gia trong hệ thống phải là những người dùng hợp lệ và có quyền truy cập đối với dữ liệu.Hai mối quan ngại chính là vấn đề xác thực và ủy quyền.Xác thực tức là nhận dạng người dùng và dịch vụ khi anh ta truy cập vào hệ thống.Còn ủy quyền tức là sau khi anh ta đăng nhập vào hệ thống thì anh ta có thực hiện đúng các quyền được cấp hay không

DCE Security đảm bảo rằng môi trường truyền thông cũng như quá trình truy cập dữ liệu trên mạng là an toàn.DCE Security cũng có một Database mà lưu trữ các thông tin về người sử dụng hệ thống như Account,nhóm,tổ chức,chính sách…Database nay được gọi là Registry được lưu trữ trên các server riêng.Registry thường được biểu diễn dưới dạng cây như sau:



Registry Directory Structure

DCE security thường bao gồm các bộ phận sau:

Authentication service:điều khiển quá trình xác thực(nhận dạng chính xác người sử dụng).Nó cũng bao gồm ticket granting service mà cho phép truyền thông an toàn.

Privilege Service cung cấp các quyền cho người sử dụng và gửi các thông tin đó lên DCE server .

Registry service lưu trữ các thông tin của người sử dụng gồm account,group…

Access control list facility cung cấp các cơ chế để phối hợp các truy nhập tài nguyên

Login facility cung cấp môi trường cho người sử dụng login và khởi tạo môi trường làm việc an toàn.

* 1. *Authentication Service*

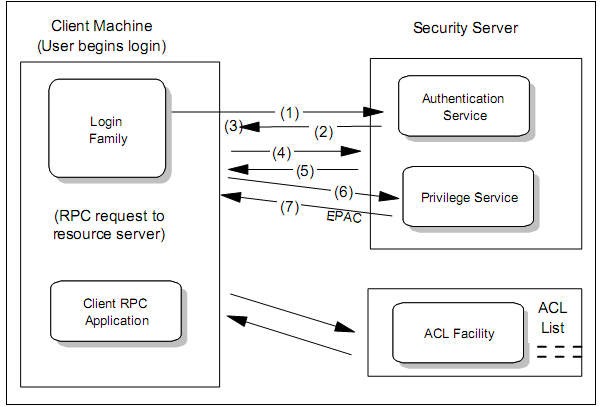
Nhiệm vụ chủ yếu của AS là cho phép nhận dạng các bên tham gia trong mạng DCE.Bao gồm cả người sử dụng,các xác thực dịch vụ trong trong môt trường DCE,không giống như trong hầu hết các mô hìn client/server nơi mà chỉ người dùng được xác thực.Có hai giai đoạn xác thực.Trong giai đoạn khởi tạo Kerberos third-party protocol được sử dụng để xác nhận yêu cầu của Client .Tiếp theo ta trả lại kết quả gồm giấy ủy quyền phiên làm việc với DCE Server khi mà người sử dụng muốn truy cập vào tài nguyên.

Trong DCE Version 1.1 một ý tưởng được đưa ra là ngoài sử dụng giao thức xác thực Kerberos.Còn có quá trình xác thực lại nhằm đảm bảo an toàn cho dịch vụ khi mà kẻ tấn công cố tình đoán ID của người dùng để tấn công hệ thống.Trong DCE 1.1 có 3 giao thức dùng để xác thực lại gồm:

No preauthentication dùng cho các phiên bản trước 1.1

Timestamps được sử dụng trong DCE Client 1.1 mà không dùng giao thức xác thực ba bên.Một nhãn thời gian được mã hóa và gừi cho Server bảo mật.Nhãn thời gian được giải mã nếu trong thời gian 5 phút thì ngươi sử dụng được xác thực lại

Third-party nó tương tự như nhãn thời gian nhưng thêm các thông tin Client được mã hóa với một khóa khác.



Quá trình xác thực theo Third-party

* 1. *DCE Threads*

Trong các ứng dụng truyền thống (được viết bằng các ngôn ngữ như C,cobol …) có rất nhiều đoạn chương trình được thực hiện một cách tuần tự.Tại một thời điểm chỉ có một điểm được thực hiện.Đây có thể coi như một Single Thread.Một luồng là một dòng thực thi của một đơn vị trong một tiến trình.Để tăng hiệu năng cho ứng dụng ta có thể thực hiện nhiều vùng trong cùng một thời gian điều này được coi như là đa luồng trong ứng dụng.Khả năng thực hiện đa luồng phụ thuộc vào hệ điều hành.

Trong môi trường tính toán phân tán dựa trên mô hình Client/Server nhiều luồng được cung cấp thêm khả năng thực hiện nhiều thủ tục trong một thời điểm.Công việc được tiến hành một cách liên tục tại luồng hiện tại trong khi một luồng khác bị block.Một server có thể cung cấp nhiều thủ tục có thể thực hiện đồng thời gọi là các tiến trình.Khi một luồng đợi để kết thúc quá trình vào ra một tiến trình khác có thể thực hiện.

Để chức năng này có thể thực hiện tốt hơn,luồng phải được tích họp trong hệ điều hành,nếu một luồng được thực hiện trong ứng dụng ở mức cao thì hiệu năng của nó sẽ thấp hơn khi được tiến hành trong hên điều hành.

*DCE Remote Procedure Call*

DCE Remote Procedure Call là một nền tảng của quá trình truyền thông Client/Server trên môi trường DCE

RPC cung cấp khả năng cho các ứng dụng phân tán có thể truy cập vào nhiều hệ thống tại bất cứ nơi nào trên mạng.

Một ứng dụng viết bằng DCE RPC được cấp cho một Client mà có thể đưa ra các RPC request và một Server mà nhận RPC request,xử lý chúng sau đó gửi lại kết quả cho Client.RPC có 3 thành phần chính:

-Interface Definition Language (IDL) và trình biên dịch của nó.Từ file được chỉ ra nó sinh ra các header file,Client stub,và Server Stub.Điều này cho phép một ứng dụng có thể cung cấp các RPC giống như nó cung cấp các thủ tục trên máy local.

-Thành phần trình bày dữ liệu trên mạng,mà định nghĩa các định dạng dữ liệu như các tham số đầu vào và đầu ra.Điều này đảm bảo chắc chắn rằng các bit có thể được sắp xếp và có Platform mà máy nhận có thể chuyển sang định dạng của nó.Quá trình sử lý định dạng dữ liệu cho RPC được gọi là marshalling .

-Runtime library che dấu đi các chi tiết của ứng dụng khi chúng được truyền thông từ Client đến Server.

Người lập trình ứng dụng có thể sử dụng đa luồng khi thực hiện RPC.

RPC có thể được sử dụng để xây dựng các ứng dụng như sử dụng các tiện ích DCE khác như CDS hay security service.CDS có thể được sử dụng để tìm kiếm hoặc quảng bá các địa chỉ của service cho client truy nhập.Trong khi security service được sử dụng để xác thực RPC và mà hóa dữ liêh trên đường truyền hay thực hiện ủy quyền.

* 1. *Distributed Time Service*

Đồng bộ thời gian trên các host khác nhau là rất khó khăn vì có sự sai khác giữa các đồng hồ vật lý.Vấn đề hiện nay trong các ứng phân tán là phụ thuộc vào thứ tự thời gian của các sự kiện trong suốt quá trình thực thi.Ví dụ khi một lập trình viên biên dịch code của anh ta trên máy trạm và gòi tới một vài file trên máy chủ nếu máy trạm và máy chủ không được đồng bộ với nhau có thể quá trình biên dịch sẽ bị ngừng mãi mãi vì thời gian giữa hai máy là khác nhau.Vấn đề này cacngf trở lên quan trọng khi trong hệ thống phân tán có hàng trăm máy tính khác nhau.

DCE Distributed Time Service (DTS) cung cấp một phần mềm chuẩn cho phép đồng bộ giữa các Host khác nhau trong môi trường hệ phân tán.Nó cũng cung cấp cách thức để giữ thời gian trên các host gần với thời gian tuyệt đối .

Distributed Time Service gồm các thành phần sau:

-Local time server

-Global time server

-Courier and backup courier time server

Local time server chịu trách nhiệm phản hồi các request về thời gian của các Time Clerk trong LAN.Nó giúp đồng bộ trên một LAN.Nếu Time Clerk không được phản hồi từ LTS nó sẽ kết nối với global time server thông qua CDS lookup.

Global time server (GTS) quảng bá chính nó trong các CDS namespace cho tất cả các hệ thống có thể biết nó một cách dễ dàng.Một đối tượng tham gia GTS cũng sử dụng GTS như trên local LAN nhưng rõ ràng khả năng cung cấp thông tin của GTS là cao hơn rất nhiều.

Courier roles :local hay global time server cũng có thể có courier role.Chúng có thể là các courier, backup couriers hay non-couriers. Courier hoạt động như local time server hay như các bên tham gia trong tiến trình đồng bộ thời gian.Tuy nhiên các courier không có đồng hồ thời gian cho chính nó.Nó request thời gian từ global time server về thời gian của các LAN khác ,hay của các bộ phận khác trong cell bởi vì thời gian trên các LAN là khác nhau nên nó phải remote vào các LAN segment và đồng bộ thời gian của chúng.

Backup courier role cung cấp các trợ giúp trong trường hợp Primary couter cho một LAN nào đó không hoạt động. Backup courier sẽ giàn xếp để chỉ định một courier mới cho LAN đó và đồng bộ với GTS.

1. ***Distributed File Service***

Distributed File Service (DFS) không thực sự là một thành phần lõi của DCE nhưng nó lại là một ứng dụng được tích hợp và sử dụng bởi các dịch vụ DCE khác.DFS cung cấp việc chia sẻ file toàn cục.Việc truy nhập vào các file ở bất kỳ đâu được liên kết qua cell DCE là trong suốt với người sử dụng.với người sử dụng,nó xuất hiện như những tập tin được đặt trên một ổ đĩa cục bộ .DFS server và các client có thể chạy trên các máy tính khác nhau với hệ điều hành khác nhau.Nguồn gốc của DFS xuất phát từ công ty Transarc thực thi giống như AFS từ đại học Carnegie-Mellon.DFS phù hợp với POSIX 1003.1 cho ngữ nghĩa của file hệ thống và POSIX 1003.6 cho việc truy cập điều khiển bảo mật .DFS được xây dựng và tích hợp với tất cả các dịch vụ khác của DCE và phát triển trên hệ thống file phân tán với các tính chất sau :

+ Trong suốt về định vị

+Uniform naming

+Hiệu suất cao

+Bảo mật

+Tính sẵn sàng cao

+File điều khiển tính nhất quán

+Tương thích với NFS

DFS cho phép thực hiện mô hình client-server và nó mở rộng khái niệm cell DCE bởi cung cấp DFS administrative domains,điều này cho phép quản lý một tập hợp của DFS server và các client hệ thống trong một cell DCE

Ở đây có nhiều DFS file trong một cell . Mỗi máy chủ file DFS chạy một dịch vụ xuất file để đảm bảo rằng các file luôn sẵn sàng khi DFS client cần đến. Bộ xuất file còn được biết đến với tên gọi bộ xuất giao thức.DFS chạy trình quản lý bộ nhớ cache, nó đóng vai trò trung gian giữa các ứng dụng yêu cầu file từ DFS server. Trình quản lý bộ nhớ cache đóng vai trò dịch các yêu cầu RPC tới file xuất ở trên file hệ thống server và lưu file dữ liệu trên đĩa hay trên bộ nhớ. Nó đảm bảo rằng client luôn cập nhật bản copy lên file. DFS file server có thể phục vụ hai loại file hệ thống khác nhau :

+File hệ thống cục bộ (LFS) được biết đến như phần giữa của file hệ thống

+Một vài các file hệ thống khác, giống như file hệ thống UNIX (UFS)

Chức năng DFS chỉ có tính sẵn sàng với LFS bao gồm:  
 +Hiệu suất cao

+Log-based, fast restarting file sets for quick recovery from failure

+Hiệu suất cao với các bản sao, tự động hóa cập nhật, có khả năng chịu lỗi file server

+Tính bảo mật mạnh với việc tích hợp với dịch vụ bảo mật DCE cung cấp chứng thực ACL.

* 1. *File Naming*

DFS sử dụng dịch vụ Cell Directory để quản lý không gian tên miền. DFS được xậy dựng như hệ thống file của mỗi cell. Thư mục và file có thể được truy cập bởi người sử dụng ở bất kỳ đâu ở trên mạng mà người sử dụng không biết nơi lưu trữ thực sự của nó.Tất cả các tài nguyên của DCE đều thuộc không gian tên miền toàn cầu. Một ví dụ của DFS file naming, để truy cập tới một file trong một cell một người sử dụng có thể sử dụng đường dẫn sau:

Nếu truy cập từ ngoài cell, sử dụng định dạng GDS (X 500) có thể truy cập theo đường dẫn:



Hay sử dụng định dạng DNS:  


* 1. *Hiệu suất*

Hiệu suất là một trong những mục tiêu chính của DFS và có thể đạt được nhờ những điều sau:

+ Bộ quản lý bộ nhớ cache: File yêu cầu từ server được lưu trữ trong bộ nhớ cache của client đểclient không cần phải gửi các yêu cầu cho dữ liệu thông qua mạng mỗi khi người sử dụng cần một file.Điều này làm giảm tải ở trên hệ thống file server, giảm lưu lượng của mạng tới mức tối thiểu do đó có thể tăng được hiệu suất.

+ Server đa luồng: DFS server sử dụng các luồng DCE hỗ trợ nhiều yêu cầu từ cac client.

+ Đường ống RPC: là cơ chế có thể giúp vận chuyển số lượng lớn dữ liệu hiệu quả.

+ Bản sao: Bản sao cho phép đảm bảo được cân bằng tải, và tăng hiệu suất khi các client có thể truy cập dữ liệu từ nhiều bản sao.

**Tính nhất quán file**

DFS sử dụng cơ chế thẻ bài để đồng bộ các file được truy cập bởi nhiều người sử dụng và đảm bảo rằng mỗi người sử dụng sẽ luôn làm việc với phiên bản mới nhất của file.Quá trình sử lý này là hoàn toàn trong suốt với người sử dụng.

**Tính sẵn sàng**

LFS file có thể được nhân bản trên nhiều server để có thể tăng tính sẵn sàng tốt nhất cho hệ thống .Mỗi file có thể là một file đơn với quyền đọc và viết và quyền chỉ đọc trên các bản sao.Phiên bản có cả quyền đọc và viết chỉ có thể được sửa đổi bởi một người .Mỗi sự thay đổi trên phiên bản đó đều được lan truyền đến các bản sao .Nếu có sự hỏng hóc của một máy chủ thì công việc không bị ngừng trệ ,các yêu cầu của máy client sẽ được tự động chuyển sang một bản sao

**DFS Security**

DCE security cung cấp DFS với việc chứng thực định danh người sử dụng ,kiểm tra các quyền của người sử dụng ,chứng thực điều khiển.Sử dụng cơ chế DCE security ACL.DFS cung cấp khả năng mềm dẻo và điều khiển truy nhập mạnh hơn các mô hình đặc trưng cung cấp bởi hệ điều hành

**Sự tương thích giữa DFS và NFS**

DFS có thể xuất ra file dạng NFS để cho các client của NFS có thể truy cập chúng.

1. **Khảo sát thực tế với nền tảng WCF trên Web Service**
2. ***Khái niệm cơ bản***
   1. *WCF là gì?*

WCF (Windows Communication Foundation) là công nghệ nền tảng nhằm thống nhất nhiều mô hình lập trình giao tiếp được hỗ trợ trong .NET 2.0 thành một mô hình duy nhất. Vào tháng 11 năm 2005, .NET 2.0 được Microsoft phát hành trong đó có cung cấp các hàm API riêng biệt cho các liên lạc dựa trên SOAP để tối đa hoá sự làm việc giữa các nền tảng sử dụng Web Services, đồng thời .NET 2.0 còn cung cấp các API để tối ưu việc liên lạc dựa trên mã nhị phân giữa các ứng dụng chạy trên hệ thống Windows gọi là .NET Remoting, các API cho các giao dịch phân tán, và API cho liên lạc dị bộ. WCF thống nhất các API này thành một mô hình duy nhất nhằm đáp ứng mô hình lập trình hướng dịch vụ.

WCF có thể sử dụng các bản tin SOAP giữa hai tiến trình, do đó làm cho các ứng dụng dựa trên WCF có thể làm việc với các tiến trình khác thông qua việc giao tiếp sử dụng bản tin SOAP. Khi một tiến trình WCF liên lạc với một tiến trình không là WCF, các bản tin SOAP được mã hoá trên cơ sở XML, nhưng khi nó liên lạc với một tiến trình WCF khác, bản tin SOAP có thể được tối ưu hoá dựa trên mã hoá nhị phân.

* 1. *Tại sao lại sử dụng WCF?*

.NET 2.0 hỗ trợ rất nhiều phương pháp liên lạc giữa các ứng dụng khác nhau nhằm vào các mục tiêu khác nhau. Các phương pháp liên lạc này khá phức tạp và phải mất nhiều thời gian để làm chủ được công nghệ. Tuy nhiên kiến thức thu được từ việc triển khai một phương pháp ít có khả năng dùng được khi làm việc với phương pháp khác.

Với việc ra đời của WCF, mọi phương pháp liên lạc trước kia đều có thể thực hiện trên WCF. Do vậy nhà phát triển chỉ cần làm chủ được công nghệ WCF là có thể xây dựng các ứng dụng một cách nhanh chóng.

WCF là một mô hình lập trình cho phép nhà phát triển xây dựng các giải pháp dịch vụ đảm bảo tính ổn định, và bảo mật và thậm chí là đảm bảo giao dịch. Nó làm đơn giản hoá việc phát triển các ứng dụng nối kết và đưa ra cho nhà phát triển những giá trị mà có thể họ chưa nhận ra ngay, đó là cách tiếp cận phát triển hệ thống phân tán thống nhất, đơn giản, và quản lý được.

Do WCF được xây dựng trên cơ sở của .NET Framework 2.0 CLR, nó là tập các lớp cho phép các nhà phát triển xây dựng các ứng dụng hướng dịch vụ bằng môi trường lập trình quen thuộc của họ như VB.NET hay C#.

1. ***Kiến trúc tổ chức***

Hình sau mô tả các lớp chủ yếu trong kiến trúc của Windows Communication Foundation (WCF):



Hình 1 – Mô hình tổng quan về kiến trúc WCF

* 1. *Các hiệp nghị (Contracts)*

Các contract trong WCF cũng giống như các hợp đồng/hiệp định mà bạn ký trong đời sống thật. Một hợp đồng bạn ký có thể chứa các thông tin như kiểu công việc bạn sẽ làm, và những thông tin mà bạn muốn đưa ra cho các bên khác. WCF contract cũng chứa các thông tin tương tự như vậy. Contract định nghĩa các đặc tả trong hệ thống bản tin. Thông thường có các loại contract sau:

* Contract dữ liệu mô tả các tham số cho các bản tin mà một dịch vụ có thể tạo ra hay sử dụng. Các tham số bản tin được định nghĩa bằng các tài liệu sử dụng ngôn ngữ đặc tả XML Schema (XSD), điều này cho phép các hệ thống hiểu XML có thể xử lý tài liệu dễ dàng. Các dịch vụ khi liên lạc với nhau có thể không cần đồng ý với nhau về các kiểu, nhưng cần đồng ý về contract dữ liệu, nghĩa là đồng ý về các tham số và các kiểu trả về.
* Contract bản tin định nghĩa các phần có trong bản tin sử dụng các giao thức SOAP, và nó cho phép điều khiển sâu hơn tới các phần trong bản tin khi có yêu cầu sự chính xác như vậy.
* Contract dịch vụ đặc tả chi tiết các phương thức của dịch vụ, và được phân phối như là một giao diện trong các ngôn ngữ lập trình như Visual Basic hay Visual C#. Có thể hình dung về contract dịch vụ một cách gián tiếp như sau: „Đây là các kiểu dữ liệu của các bản tin của tôi, đây là nơi tôi cung cấp, và đây là các giao thức mà tôi có thể liên lạc”
* Các chính sách và các kết nối (bindings) mô tả các điều kiện cần có để giao tiếp với một dịch vụ. Các chính sách sẽ bao gồm cả các yêu cầu về bảo mật và các điều kiện khác cần phải có khi kết nối với một dịch vụ.
  1. *Dịch vụ thực thi (Runtime Service)*

Lớp dịch vụ thực thi chứa các hành xử sẽ xảy ra trong quá trình thực hiện của dịch vụ, nghĩa là các hành xử thực thi của dịch vụ. Ta sẽ thấy một số các hành xử như sau:

* **Throttling behavior:** Điều khiển luồng nhằm quy định xem có bao nhiêu bản tin được xử lý.
* **Error behavior:** Hành xử lỗi quy định những hành động khi lỗi xảy ra trong hệ thống.
* **Metadata behavior:** Hành xử với các siêu dữ liệu quy định xem làm thế nào và khi nào thì các siêu dữ liệu được đưa ra bên ngoài dịch vụ.
* **Instance behavior:** Hành xử thực thể quy định xem có bao nhiêu thực thể của dịch vụ đó được chạy.
* **Transaction behavior:** Hành xử giao dịch cho phép việc rollback các giao dịch nếu xảy ra lỗi.
* **Message inspection:** Kiểm tra bản tin đem lại cho dịch vụ khả năng kiểm tra tất cả hay một số phần của bản tin.
* **Dispatch behavior:** Khi một bản tin được xử lý bởi nền tảng WCF, dịch vụ Dispatch behavior xác định xem bản tin được xử lý như thế nào.
* **Concurrency behavior:** Hành xử đồng thời xác định xem việc xử lý thế nào với việc đa luồng của mỗi dịch vụ hay mỗi thực thể của dịch vụ. Hành xử này giúp cho việc điều khiển số lượng luồng có thể truy nhập tới một thực thể của dịch vụ.
* **Parameter filtering:** Khi một bản tin được đưa tới một dịch vụ, sẽ xảy ra một số hành động dựa trên nội dung phần đầu đề của bản tin. Phần lọc tham số sẽ thực hiện lọc các đầu đề bản tin và thực hiện các hành động đặt sẵn dựa trên việc lọc đầu đề bản tin.
  1. *Thông điệp/Bản tin (Message)*

Lớp bản tin là tập hợp các kênh. Mỗi kênh là một thành phần xử lý bản tin theo một cách nào đó. Một tập các kênh thường được gọi là ngăn xếp kênh. Các kênh làm việc trên bản tin và trên đầu đề của bản tin. Lớp này khác với lớp thực thi dịch vụ chủ yếu bởi sự khác nhau trong việc xử lý nội dung bản tin.

Có hai kênh khác nhau là kênh vận chuyển (transport channel) và kênh điều khiển (control channel):

* Kênh vận chuyển phụ trách việc đọc và ghi các bản tin từ mạng (network) hoặc từ một số điểm giao dịch bên ngoài).
* Kênh điều khiển thực hiện xử lý bản tin theo giao thức, thông thường làm việc bằng cách đọc và ghi thêm các đầu đề cho bản tin.
  1. *Lưu trữ và kích hoạt (Host and Activation)*

Nhìn một cách tổng thể thì một dịch vụ thực chất là một chương trình. Cũng giống như các chương trình khác, một dịch vụ cần phải chạy trong một tệp thực thi. Dịch vụ này thường được gọi là dịch vụ tự chứa.

Các dịch vụ còn có thể được chứa, hoặc chạy trong một tệp thực thi được quản lý bởi một agent bên ngoài như IIS hay Windows Activation Services (WAS). WAS cho phép WCF được kích hoạt một cách tự động khi phân phối tới một máy tính có chạy WAS.

1. ***Các tính năng đặc trưng***
   1. *Giao dịch (Transaction)*

Một giao dịch là một đơn vị của công việc. Một giao dịch đảm bảo chắc chắn rằng mọi thứ diễn ra trong giao dịch thành công hay thất bại đều là kết quả tổng thể. Ví dụ, nếu một giao dịch chứa ba mục công việc cần thực hiện, trong quá trình thực hiện giao dịch, một trong số các mục đó bị thất bại, khi đó cả ba mục sẽ là thất bại. Giao dịch chỉ thành công khi cả ba mục công việc đều thành công. Giao dịch thường thấy trong các thao tác với cơ sở dữ liệu.

WCF cho phép đưa vào việc xử lý giao dịch như trên với các liên lạc. Nhà phát triển có thể nhóm các liên lạc với nhau thành các giao dịch. Ở mức doanh nghiệp, tính năng này cho phép bạn thực hiện các công việc giao dịch qua các nền tảng khác nhau.

* 1. *Lưu trữ (Host)*

WCF cho phép các dịch vụ được chứa trong một số lớn các môi trường khác nhau, như Windows NT Services, Windows Forms, và ứng dụng console, cũng như ở trên IIS (Internet Information Server) và WAS (Windows Activation Services).

Chứa ứng dụng trên IIS còn có thêm các lợi điểm khác là dịch vụ có thể nhận các ưu điểm của rất nhiều tính năng có sẵn trên IIS, ví dụ IIS có thể điều khiển một cách tự động việc bắt đầu hay kết thúc một dịch vụ.

* 1. *Bảo mật (Security)*

Bảo mật là tính năng không thể thiếu trong WCF nói riêng và trong liên lạc nói chung. Trong WCF, tất cả mọi thứ từ các bản tin tới các client hay server đều phải xác thực và WCF có tính năng để đảm bảo rằng các bản tin không bị lẫn trong quá trình vận chuyển. WCF bao gồm việc đảm bảo tính toàn vẹn và bảo mật của bản tin.

WCF còn cho phép bạn tích hợp ứng dụng của bạn với cơ sở hạ tầng bảo mật sẵn có, bao gồm cả các chuẩn bên ngoài môi trường Windows bằng cách sử dụng các bản tin SOAP bảo mật.

1. ***Dịch vụ trong WCF (Services)***

Có ba kiểu dịch vụ trong WCF là: Định kiểu, Không định kiểu, và Bản tin định kiểu. Sau đây chúng ta sẽ xem xét chi tiết các kiểu dịch vụ của WCF.

* 1. *Định kiểu (Typed Service)*

Dịch vụ có định kiểu là loại dịnh vụ đơn giản nhất trong số ba kiểu dịch vụ. Tuy vậy nó cung cấp phần lớn các tính năng cần thiết để bạn phát triển các dịch vụ WCF.

Một thuật ngữ hay dùng cho dịch vụ định kiểu là “mô hình tham số”, mô hình này định nghĩa một cách chính xác kiểu dịch vụ này làm gì. Các dịch vụ định kiểu không giới hạn ở việc định kiểu của các tham số hay kết quả trả về. Các dịnh vụ kiểu này có thể chấp nhận các kiểu đơn giản cũng như phức hợp. Tuy nhiên khi truyền các tham số hay trả về giá trị kiểu phức hợp, ta cần phải định nghĩa data contract cho các kiểu đó.

Dịch vụ định kiểu trong Windows Communication Foundation xử lý tất cả các bản tin, bạn không cần phải làm việc trực tiếp ở mức bản tin. Ví dụ sau đây mô tả một dịch vụ định kiểu. Contract dịch vụ định nghĩa 2 thao tác. Thao tác thứ nhất không cần tham số, thao tác thứ 2 chấp nhận 2 tham số kiểu là int (số nguyên):

*[ServiceContract]*

*public interface IService1*

*{*

*[OperationContract]*

*string SayHello();*

*[OperationContract]*

*string GetData(int firstValue, int secondValue);*

*}*

Dịch vụ định kiểu còn hỗ trợ các kiểu tham số với từ khoá ref và out.

* 1. *Không định kiểu (Untyped Service)*

Các dịch vụ không định kiểu hơi phức tạp hơn so với dịch vụ định kiểu do chúng làm việc trực tiếp với bản tin. Trong kiểu dịch vụ này, bạn định nghĩa các bản tin và nội dung của chúng. Nghĩa là với kiểu dịch vụ này, bạn làm việc ở mức bản tin, các đối tượng bản tin được chuyển qua lại giữa client và dịch vụ, và dịch vụ có thể trả về một đối tượng bản tin nếu được yêu cầu. Hơn nữa, kiểu dịch vụ này cho bạn khả năng truy nhập tới nội dung bản tin.

* 1. *Bản tin định kiểu (Typed Message Service)*

Trong một dịch vụ bản tin có định kiểu (typed message service) còn được gọi là “mô hình contract bản tin” bạn định nghĩa các bản tin và nội dung của chúng. Ở đây, các bản tin được mô tả với thuộc tính MessageContract, là các contract bản tin thường được sử dụng để định nghĩa các lớp bản tin. Các bản tin này sẽ được sử dụng để gửi qua lại trong các thao tác của dịch vụ.

1. ***Kiến trúc Client trong WCF***
   1. *Khái niệm cơ bản*

Một client trong Windows Communication Foundation là một chương trình sử dụng các chức năng cung cấp bởi một dịch vụ WCF. Chương trình client sẽ liên lạc với dịch vụ thông qua điểm cuối dịch vụ. Để làm được việc này client cần phải biết một số thông tin về dịch vụ như địa chỉ của điểm cuối, binding mà dịch vụ sử dụng, và contract dịch vụ. Các thành phần này đã được thảo luận ở các bài trước.

Một trong những thứ bạn có thể thấy trong kiến trúc của client là kênh thông tin được xây dựng dựa trên các cấu hình binding, những cấu hình này được quy định trong tệp tin cấu hình. Những thông tin cấu hình này chính là các thông tin đã được giới thiệu trong phần nói về bindings. Những bindings này cho phép client và dịch vụ liên lạc với nhau một cách hiệu quả.

Điểm thứ hai bạn có thể thấy là cài đặt của giao diện IClientChannel. Giao diện này định nghĩa các thao tác cho phép nhà phát triển điều khiển các chức năng của kênh, như đóng phiên làm việc của client và huỷ một kênh (để thu hồi tài nguyên). Nó đưa ra các phương thức và hàm của lớp được chỉ ra sau đây: System.ServiceModel.ChannelFactory.

Cuối cùng là bạn có thể thấy contract dịch vụ được tạo ra tự động, contract dịch vụ sẽ cung cấp tính năng chuyển lời gọi hàm ở phía client thành các bản tin đi, và chuyển các bản tin tới thành thông tin mà chương trình client có thể sử dụng dưới dạng giá trị trả về hay tham số đầu ra của hàm.

Các client liên lạc với điểm cuối dịch vụ thông qua một proxy, như hình dưới. Sự liên lạc được thực hiện thông qua một kênh. Sau khi proxy và kênh được tạo ra, client có thể truy xuất các phương thức có ở điểm cuối đó.



Hình 2 – Mô hình giao tiếp giữa Client và dịch vụ của nhà cung cấp

* 1. *Các đối tượng phía Client*
     1. ICommunicationObject

Giao diện ICommunicationObject là một trong những thành phần lõi để định nghĩa chức năng liên lạc. Nhiệm vụ của đối tượng này là định nghĩa contract cho trạng thái cơ bản của tất cả các đối tượng trong hệ thống; ví dụ, trạng thái đối tượng liên lạc đóng hay mở, hoặc trong trạng thái đang mở hoặc đang đóng. Những đối tượng này bao gồm các kênh, các đầu nghe (listeners), các dispatchers, factories, và chứa dịch vụ (service host).

Một chuyển trạng thái là việc chuyển từ trạng thái này sang trạng thái khác; ví dụ kênh liên lạc chuyển từ trạng thái “đang mở” sang trạng thái “mở”.

Giao diện này định nghĩa các phương thức cho việc khởi tạo chuyển trạng thái:

* **Open**: Làm cho đối tượng liên lạc chuyển trạng thái từ “Created” sang “Opened”.
* **Close**: Làm cho đối tượng liên lạc chuyển từ trạng thái hiện tại sang trang thái “Opened”.
* **Abort**: Làm cho đối tượng liên lạc chuyển ngay lập tức từ trạng thái hiện tại sang trạng thái “Closed”.

Ngoài các phương thức khởi tạo ở trên, giao diện ICommunicationObject còn định nghĩa các sự kiện để thông báo việc thay đổi trạng thái:

* **Opening**: Sự kiện này được kích hoạt khi đối tượng chuyển trạng thái từ Created sang Opened, sự kiện này xảy ra khi hàm Open hoặc BeginOpen được gọi.
* **Closing**: Sự kiện này được kích hoạt khi đối tượng chuyển trạng thái từ Opened sang Closed, sự kiện này xảy ra khi hàm Close hoặc BeginClose được gọi.
* **Opened**: Sự kiện này được kích hoạt khi đối tượng kết thúc việc chuyển trạng thái từ Opening sang Opened
* **Closed**: Sự kiện này được kích hoạt khi đối tượng kết thúc việc chuyển trạng thái từ Closing sang Closed
* **Faulted**: Sự kiện này được kích hoạt khi đối tượng chuyển sang trạng thái Faulted.

Ngoài ra còn có một tập phương thức dị bộ cho các phương thức Open và Close:

* **BeginOpen**: Bắt đầu thao tác dị bộ để mở một đối tượng liên lạc
* **BeginClose**: Bắt đầu thao tác dị bộ để đóng một đối tượng liên lạc.
* **EndOpen**: Hoàn tất thao tác dị bộ để mở một đối tượng liên lạc
* **EndClose**: Hoàn tất thao tác dị bộ để đóng một đối tượng liên lạc

Giao diện ICommunicationObject có một thuộc tính trạng thái State, với kiểu là CommunicationState, thuộc tính này được dùng để chỉ ra trạng thái hiện tại của đối tượng.

Khi một đối tượng cài đặt ICommunicationObject được khởi tạo, trạng thái mặc định là Created chứ không phải là trạng thái Opened. Trong khi đang ở trạng thái Created, ta chỉ có thể thiết lập cấu hình cho đối tượng ICommunicationObject chứ không thể dùng nó để gửi hay nhận bản tin. Để có thể gửi hay nhận bản tin, đối tượng cần phải chuyển sang trạng thái Opened, nhưng khi ở trạng thái này, ta không thể thiết lập cấu hình cho nó nữa.

Để chuyển đối tượng sang trạng thái Opened, ta gọi hàm Open. Đối tượng sẽ giữ trạng thái này cho tới khi nó chuyển hoàn toàn sang trạng thái Closed. Hàm Close được dùng để chuyển đối tượng sang trạng thái Closed, tuy nhiên nó đợi cho các thao tác hiện tại của đối tượng hoàn tất mới thực hiện chuyển trạng thái. Hàm Abort thì không như vậy, nghĩa là những thao tác chưa hoàn tất sẽ bị huỷ bỏ, và đối tượng được chuyển ngay sang trạng thái Closed.

* + 1. IExtensibleObject

Giao diện IExtensibleObject cung cấp cách mở rộng cho client. Trong WCF, cách mở rộng đối tượng được sử dụng để thêm các tính năng mới cho các lớp thực thi đã có, qua đó mở rộng các thành phần hiện tại cũng như thêm mới các trạng thái cho một đối tượng.

Giao diện này được đưa ra duy nhất một thuộc tính để cung cấp tính năng này, Extensions, với kiểu là IExtensionCollection. Thuộc tính này được sử dụng để trả về một tập hợp các đối tượng mở rộng có thể được sử dụng để mở rộng các lớp thực thi đã có.

* 1. *Các phương thức liên lạc của Client*
     1. Một chiều (One way)

Liên lạc một chiều là liên lạc chỉ theo một hướng duy nhất. Là hướng từ phía client tới dịch vụ. Không có trả lời từ phía dịch vụ và client hoàn toàn không trông mong nhận được phản hồi. Theo cách này, client gửi đi một bản tin và tiếp tục thao tác của mình.

Do không có phản hồi từ dịch vụ trong liên lạc một chiều, client sẽ không biết được liệu có lỗi xảy ra trong quá trình liên lạc hay không, và nó cũng hoàn toàn không biết liệu yêu cầu có thành công hay không. Để tạo ra dịch vụ cho liên lạc một chiều, ta đặt tham số IsOneWay của thuộc tính mô tả OperationContract là True. Điều này sẽ thông báo cho dịch vụ biết là không cần có phản hồi. Đoạn mã nguồn sau đây biểu diễn cách thiết lập một liên lạc một chiều, thao tác AddPerson, EditPerson, và DeletePerson:

*[ServiceContract]*

*public interface IStaffInformation*

*{*

*[OperationContract]*

*bool HasPerson(int personId);*

*[OperationContract]*

*Person GetPerson(int personId);*

*[OperationContract]*

*Person[] GetAll();*

*[OperationContract(IsOneWay=true]*

*void AddPerson(Person person);*

*[OperationContract(IsOneWay=true]*

*void EditPerson(int personId, Person person);*

*[OperationContract(IsOneWay=true]*

*void DeletePerson(int personId);*

*}*

* + 1. Yêu cầu – Hồi đáp (Request – Reply)

Liên lạc theo kiểu yêu cầu-trả lời thực hiện như sau: client gửi đi một bản tin cho dịch vụ, nó sẽ chờ để nhận một phản hồi từ dịch vụ. Kiểu liên lạc này đồng thời còn có nghĩa là khi client gửi đi bản tin, nó sẽ không thực hiện thao tác khác cho tới khi nó nhận được một phản hồi từ phía dịch vụ.

Trong Windows Communication Foundation, có hai cách để quy định kiểu liên lạc yêu cầu-trả lời. Cách thứ nhất là đặt giá trị false cho tham số IsOneWay của OperationContract. Thực chất giá trị mặc định của tham số IsOneWay là false, do vậy cách thứ hai để quy định kiểu liên lạc yêu cầu-trả lời là không dùng tham số IsOneWay nữa. Do vậy các thao tác trên dịch vụ của WCF mặc định là liên lạc kiểu hỏi-trả lời.

Kiểu liên lạc hỏi-trả lời là kiểu liên lạc mặc định của WCF nên thông thường bạn cài đặt dịch vụ WCF hay sử dụng dịch vụ WCF, bạn liên lạc theo kiểu hỏi-trả lời. Các ví dụ ở các bài trước đều theo kiểu làm việc này. Hai kiểu liên lạc tiếp theo đây là liên lạc song công và liên lạc dị bộ ta ít gặp hơn.

* + 1. Song công (Duplex)

Liên lạc song công là khả năng mà cả client và dịch vụ đều có thể khởi tạo liên lạc, cũng như phản hồi các bản tin đến; nói cách khác đây là liên lạc hai chiều. Với liên lạc song công, dịch vụ không chỉ có thể trả lời các bản tin đến mà còn có thể khởi tạo liên lạc với client bằng cách gửi bản tin yêu cầu một phản hồi từ phía client.

Để cấu hình một liên lạc song công, cần có thay đổi từ cả hai phía: dịch vụ và client.

1. Phía dịch vụ

Để thực hiện liên lạc song công, phía dịch vụ cần có hai giao diện. Mục tiêu của giao diện thứ nhất là sử dụng trong liên lạc từ client tới dịch vụ, nghĩa là nó được sử dụng để nhận các bản tin từ phía client, giống như các ví dụ từ trước tới nay. Giao diện thứ hai, còn gọi là callback interface, được sử dụng trong liên lạc từ dịch vụ tới client để gửi bản tin từ phía dịch vụ tới client. Điều quan trọng là các thao tác ở trong các giao diện phải được định nghĩa là các thao tác một chiều.

1. Phía Client

Để thực hiện liên lạc song công, phía client cũng phải nhận một phần trách nhiệm trong việc cài đặt, và cũng phải cài đặt một callback contract. Client thực hiện bằng cách cài đặt callback interface của duplex contract.

* + 1. Dị bộ (Asynchronous)

Liên lạc dị bộ được thực hiện bằng cách gọi các hàm dị bộ. Việc liên lạc dị bộ cho phép chương trình tiếp tục thực hiện những việc khác trong khi hàm được gọi vẫn đang thực hiện.

Cũng giống như trong liên lạc song công, các thao tác dị bộ yêu cầu một số thay đổi về phía client và dịch vụ.

1. ***Mô hình lập trình WCF***
   1. *Lựa chọn phương pháp tiếp cận*

Nếu các bạn đã học qua môn học về các phương pháp lập trình hẳn sẽ thấy có 2 phương pháp chính là hướng thủ tục và hướng đối tượng. Và phương pháp hướng đối tượng trong thời gian gần đây được phát triển rất mạnh và được hỗ trợ ngay trong các ngôn ngữ lập trình như C# hay VB.NET. Khi làm việc với các dịch vụ web bạn đã làm quen với một phương pháp nữa là lập trình hướng dịch vụ (Service-oriented programming). Microsoft đã cung cấp nhiều công cụ trên .NET Framework để hỗ trợ phương pháp lập trình này thông qua các lớp trong không gian tên: System.Web.Services.

Như vậy khi làm việc với WCF bạn có hai lựa chọn là hướng đối tượng và hướng dịch vụ. Vậy ta nên sử dụng phương pháp nào? Câu trả lời là dùng cả hai. Nói một cách đơn giản là phương pháp hướng đối tượng được sử dụng để phát triển các ứng dụng trên desktop, còn phương pháp hướng dịch vụ được sử dụng để kết nối các ứng dụng đó với nhau. Điều quan trọng ở đây là làm sao để hiểu được sự khác nhau giữa hai phương pháp và hiểu được khi nào chúng được sử dụng và sử dụng như thế nào đồng thời cũng phải hiểu về các lợi ích chúng cung cấp.

Về hướng đối tượng có thể hiểu như sau. Các ứng dụng hướng đối tượng là hai hay nhiều lớp phụ thuộc lẫn nhau và chia sẻ chung các kiểu dữ liệu. Những lớp này liên lạc với nhau thông qua các lời gọi các hàm mà lớp đối tượng cung cấp.

Các ứng dụng hướng dịch vụ là các chương trình không biết gì về nhau. Mỗi ứng dụng liên lạc với ứng dụng khác thông qua các bản tin. Điểm đặc biệt là các bản tin này được gửi từ một ứng dụng sang ứng dụng khác mà không quan tâm tới nền tảng mà dịch vụ đang chạy.

Khi phát triển các dịch vụ WCF, điều quan trọng là cần hiểu sự liên kết giữa hướng đối tượng và hướng dịch vụ. Khi làm việc với .NET Framework bạn chắc chắn rất quen thuộc với thuật ngữ lớp (class) và giao diện (interface). Các thuật ngữ này vẫn được sử dụng khi phát triển dịch vụ WCF. Các lớp và giao diện là phần hướng đối tượng trong WCF, còn phần hướng dịch vụ trong WCF sẽ được thấy khi bạn đưa vào các thuộc tính WCF để định nghĩa các thực thể.

* 1. *Mô hình dịch vụ*

Nếu bạn đã từng làm việc với dịch vụ web, bạn sẽ thấy mô hình này quen thuộc với bạn theo một cách nào đó. Khi bạn tạo một dịch vụ web, bạn thực sự tạo ra một dịch vụ (service). Dịch vụ web chứa một tài liệu XML để mô tả tất cả mọi thứ cần biết về dịch vụ đó. Tài liệu này được mô tả bằng ngôn ngữ Web Service Description Language (ngôn ngữ mô tả dịch vụ web). Nó chứa ba phần:

* **Service (dịch vụ)**: Chứa thông tin về vị trí của dịch vụ
* **Binding**: Chứa thông tin về cách liên lạc với dịch vụ, như dịch vụ sử dụng giao thức gì, vv.
* **PortType (kiểu cổng)**: Giải thích về dịch vụ sẽ làm gì

Mô hình dịch vụ trên WCF cũng tương tự như với mô hình dịch vụ web. Điểm khác biệt là ở cách đặt tên. Trong WCF các phần không được gọi là service, binding, và portType mà được gọi tương ứng là address (địa chỉ), binding, và contract.

Mô hình dịch vụ WCF được cung cấp trong không gian tên System.ServiceModel. Không gian tên này chứa rất nhiều lớp, nhưng bạn hoàn toàn không cần biết toàn bộ chúng.

* 1. *Các phương pháp lập trình với WCF*
     1. Phương pháp khai báo (Declarative Programming)

Lập trình khai báo đạt được thông qua các thuộc tính. Những thuộc tính này được sử dụng để định nghĩa các contract và xác định hành xử của dịch vụ. Chúng được sử dụng để xác định thêm các tham số để thay đổi các chi tiết của contract và hành xử dịch vụ.

Thuộc tính ServiceContract dùng để quy định là giao diện này định nghĩa các chức năng của một dịch vụ. Thuộc tính OperationContract được sử dụng ở các hàm để quy định rằng hàm này được khai báo là một phần của dịch vụ. Đó là tất cả những gì cần để tạo ra một dịch vụ WCF.

Thêm nữa, bạn không nhất thiết phải sử dụng các giao diện (interface) khi cài đặt một dịch vụ, điều này cũng giống như việc bạn không cần phải sử dụng giao diện để định nghĩa một lớp. Tuy vậy bạn nhất thiết phải quy định phần nào thuộc về dịch vụ. Bạn có thể định nghĩa những phần khác cần cho giao diện, nhưng chỉ những hàm (phương thức) có gắn thuộc tính [OperationContract].

* + 1. Phương pháp lập trình trực tiếp (Explicit Programming)

Là phương pháp lập trình hướng đối tượng, bạn làm việc trực tiếp với các lớp và giao diện cung cấp bởi mô hình đối tượng của WCF. Làm việc trực tiếp với mô hình đối tượng cho phép nhà phát triển tính linh hoạt cao hơn và khả năng điều khiển tốt hơn thông qua mã nguồn của họ. Thêm nữa nó cho phép điều khiển sâu hơn rất nhiều so với phương pháp khai báo và phương pháp sử dụng tập tin cấu hình.

* + 1. Phương pháp sử dụng tập tin cấu hình (XML File Config)

Cũng giống như phương pháp khai báo, có rất nhiều thứ mà bạn có thể quy định liên quan đến hành xử của một dịch vụ thông qua tập tin cấu hình của dịch vụ. Điều hay trong cách tiếp cận này là những thay đổi ở tập tin cấu hình hoàn toàn không cần phải biên dịch lại dịch vụ mới sử dụng được.

1. ***Bảo mật trong WCF (Security)***
   1. *Ý nghĩa bảo mật trong WCF*

Do WCF là một nền tảng lập trình phân tán dựa trên các bản tin SOAP, với việc sử dụng WCF, các ứng dụng bạn tạo ra có thể tạo và xử lý các bản tin từ một số không hạn chế các dịch vụ và client khác. Trong các ứng dụng này, các bản tin có thể được truyền từ điểm này sang điểm khác, thông qua các thiết bị như tường lửa, router, switch, hay qua Internet, và qua một loạt các điểm trung chuyển SOAP. Điều này tạo ra không ít mối đe doạ tới an ninh của bản tin. Những ví dụ sau đây cho ta thấy một số các mối đe doạ thường thấy khi trao đổi bản tin giữa các ứng dụng, những mối đe doạ này hoàn toàn có thể loại bỏ được nhờ vào sử dụng tính năng bảo mật trong WCF:

* Quan sát các bản tin trên mạng để lấy ra các thông tin nhạy cảm. Ví dụ ở máy client thực hiện login vào một hệ thống sử dụng chế độ gửi tên tài khoản và mật khẩu dạng text không mã hoá. Hacker hoàn toàn có thể bắt được bản tin đó và trích ra thông tin về tài khoản cùng với mật khẩu.
* Đóng giả một dịch vụ mà client không hề biết. Việc này cũng tương tự như web phishing, nghĩa là làm giả một trang web giống như trang web mà người dùng quen thuộc (như trang web yahoo hay trang web ngân hàng). Người dùng sẽ nhập thông tin về tài khoản cùng với mật khẩu đề đăng nhập vào trang giả đó. Khi đó hacker sẽ có được các thông tin này.
* Thay đổi nội dung bản tin. Hacker hoàn toàn có thể thay đổi nội dung của một bản tin mà client lẫn dịch vụ không biết.

Việc đảm bảo bảo mật cho các bản tin trao đổi giữa client với dịch vụ WCF cần phải chú trọng ở những điểm sau:

* Xác thực điểm cuối dịch vụ
* Xác thực client
* Tính nhất quán của bản tin
* Tính bảo mật của bản tin
* Phát hiện replay (hiện tượng lặp lại yêu cầu của client hoặc dịch vụ mà thực chất client/dịch vụ không đưa ra)
  + 1. Tích hợp với các kiến trúc bảo mật có sẵn

WCF hoàn toàn có thể làm việc với các giải pháp bảo mật có sẵn như Secure Sockets Layer (SSL) hoặc giao thức Kerbeos. Ngoài ra nó cũng có thể làm việc với kiến trúc bảo mật đang sử dụng như domain trên Windows sử dụng Active Directory. Ngoài việc hỗ trợ các giải pháp bảo mật thường thấy, WCF còn tích hợp với các mô hình bảo mật sẵn có ở tầng vận chuyển và có thể chuyển hạ tầng sẵn có sang các mô hình mới hơn dựa trên bảo mật các bản tin SOAP.

* + 1. Tích hợp với mô hình xác thực có sẵn

Một phần quan trọng trong bất kỳ mô hình bảo mật truyền tin nào là khả năng xác định và xác thực các thực thể trong quá trình trao đổi dữ liệu. Các thực thể trong quá trình này sử dụng các “định danh điện tử”, hay còn gọi là credentials, để xác thực chúng với đối tượng đang trao đổi. Với sự phát triển của các nền tảng phân tán, có rất nhiều cách để xác thực các credentials được tạo ra. Lấy ví dụ, trên Internet cách thông dụng nhất là sử dụng tên tài khoản cùng với mật khẩu để xác thực người dùng. Trong mạng intranet, mô hình thông dụng là sử dụng Kerberos domain controller để xác thực người dùng cùng với dịch vụ. Do vậy với cùng một dịch vụ, ta có thể có các cách xác thực khác nhau cho dịch vụ đó tuỳ thuộc vào việc dịch vụ đó được sử dụng ở môi trường nào. WCF hỗ trợ rất nhiều các mô hình xác thực khác nhau:

* Anonymous caller (Người gọi vô danh)
* Username client credential
* Certificate client credential
* Windows (Kerberos và NT LanMan – NTML)
  + 1. Làm việc liên môi trường

Trong thế giới với nhiều nền tảng như hiện nay, các nền tảng tính toán/truyền thông phân tán cần phải làm việc liên thông với các công nghệ khác nhau mà các nhà cung cấp hỗ trợ. Cũng vì vậy mà bảo mật cũng cần phải làm việc liên môi trường (interoperable).

Để thực hiện các hệ thống bảo mật interoperable, các công ty sử dụng dịch vụ Web với một loạt các chuẩn khác nhau. Về bảo mật thì có thể kể ra một số chuẩn như sau: WS-Security, SOAP Message Security, WS-Trust, WS-SecureConversation, và WS-SecurityPolicy.

Với các dịch vụ WCF ta có thể sử dụng WSHttpBinding để hỗ trợ WS-Security 1.1 và WS-SecureConversation.

* 1. *Các lĩnh vực bảo mật của WCF*

Bảo mật trong WCF chia ra thành ba vùng chức năng: transfer security (bảo mật truyền thông), (access control) điều khiển truy nhập, và auditing (ghi vết).

* + 1. Transfer Security

Bảo mật truyền thông bao gồm ba chức năng chính: sự nhất quán, sự bảo mật, và sự xác thực. Sự nhất quán là khả năng phát hiện liệu bản tin có bị thay đổi hay không. Sự bảo mật là khả năng giữ cho bản tin không đọc được bởi những người không đủ thẩm quyền; điều này có được nhờ vào cryptography (mã mật – mật mã). Sự xác thực là khả năng xác minh được một định danh có thực đúng hay không. Kết hợp ba chức năng này cho ta đảm bảo rằng các bản tin được gửi đi một cách an toàn, đến đúng nơi cần đến.

* + 1. Các chế độ bảo mật vận chuyển và bản tin

Có hai phương pháp chính dùng để thực hiện bảo mật truyền thông trong WCF là chế độ bảo mật ở tầng vận chuyển (transport security mode) và chế độ bảo mật ở bản tin (message security mode):

* *Transport security mode* sử dụng các giao thức ở tầng vận chuyển như HTTPS để đảm bảo bảo mật. Chế độ này có ưu điểm là được sử dụng rất nhiều ở các nền tảng khác nhau, và độ phức tạp tính toán ít hơn. Tuy vậy nhược điểm là chỉ đảm bảo bảo mật các bản tin từ điểm-tới-điểm.
* *Message security mode* sử dụng chuẩn WS-Security để đảm bảo bảo mật. Do bảo mật bản tin được áp dụng trực tiếp lên các bản tin SOAP và được chứa trong các vỏ (envelope) SOAP, cùng với dữ liệu của chương trình, nó có ưu điểm là không phụ thuộc vào giao thức vận chuyển, dễ mở rộng, đảm bảo bảo mật từ đầu cuối-tới-đầu cuối (thay vì điểm-tới-điểm). Nhược điểm của nó là chậm hơn so với chế độ transport security do nó phải làm việc với XML trong bản tin SOAP.
  + 1. Điều khiển truy nhập

Điều khiển truy nhập còn được biết tới như là authorization (nhận thực). Authorization cho phép những người dùng khác nhau có các quyền khác nhau để xem dữ liệu. Trong WCF, các tính năng điều khiển truy nhập được cung cấp dựa vào sự tích hợp với CLR (common language runtime) thông qua lớp thuộc tính PrincipalPermissionAttribute và qua một loạt các hàm API.

* + 1. Auditing

Auditing là quá trình ghi lại các sự kiện bảo mật vào hệ thống log của hệ điều hành Windows (Windows event log). Bạn có thể ghi lại các sự kiện có liên quan tới bảo mật như là xác thực lỗi hay thành công. Bạn có thể xem thêm trong bài How to: Audit Windows Communication Foundation Security Events

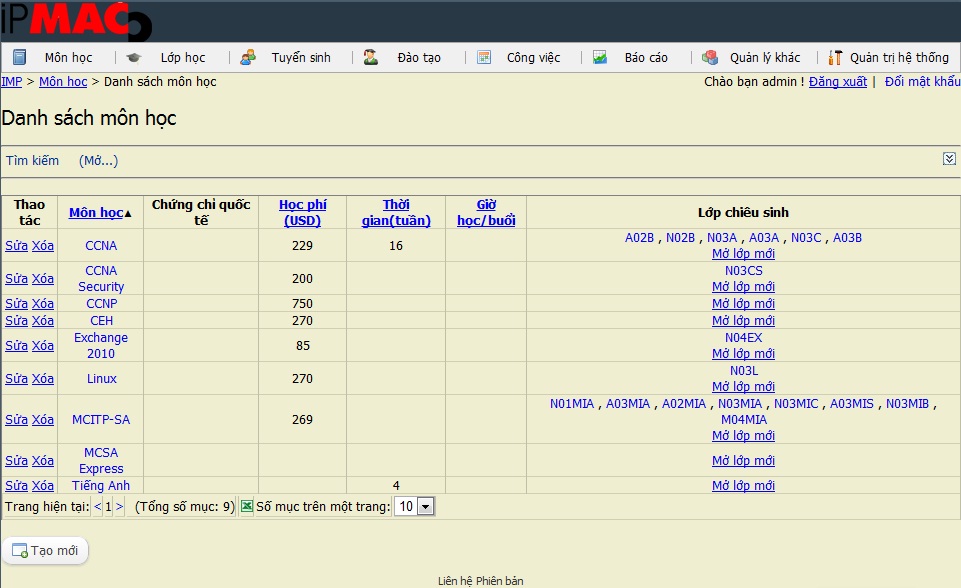
1. **Ứng dụng kiểm thử**
2. ***Mô tả***

Đây là ứng dụng quản lý các nhân viên, học viên trong một công ty: hệ thống quản lí thông tin trực tuyến IMP (iPMAC Management Portal). Đây là cổng thông tin trực tuyến cho phép người dùng có thể sử dụng rất nhiều chức năng hữu ích như: đăng kí học viên, đăng kí khóa học, lớp học, quản lí thông tin tư vấn viên, kiểm tra tình trạng lớp học, cơ sở vật chất, lên lịch hẹn... Ứng dụng này đã được xây dựng trên nền tảng remote execute: Windows Communication Foundation với một máy có nhiệm vụ làm server, cung cấp các dịch vụ cho các máy client khác sử dụng. Yêu cầu cơ bản nhất của ứng dụng đó là xây dựng các dịch vụ web (web service) để tương thích với các phần mềm trên các nền tảng và ngôn ngữ khác nhau, tiện lợi cho việc mở rộng và truy cập tại mọi nơi.

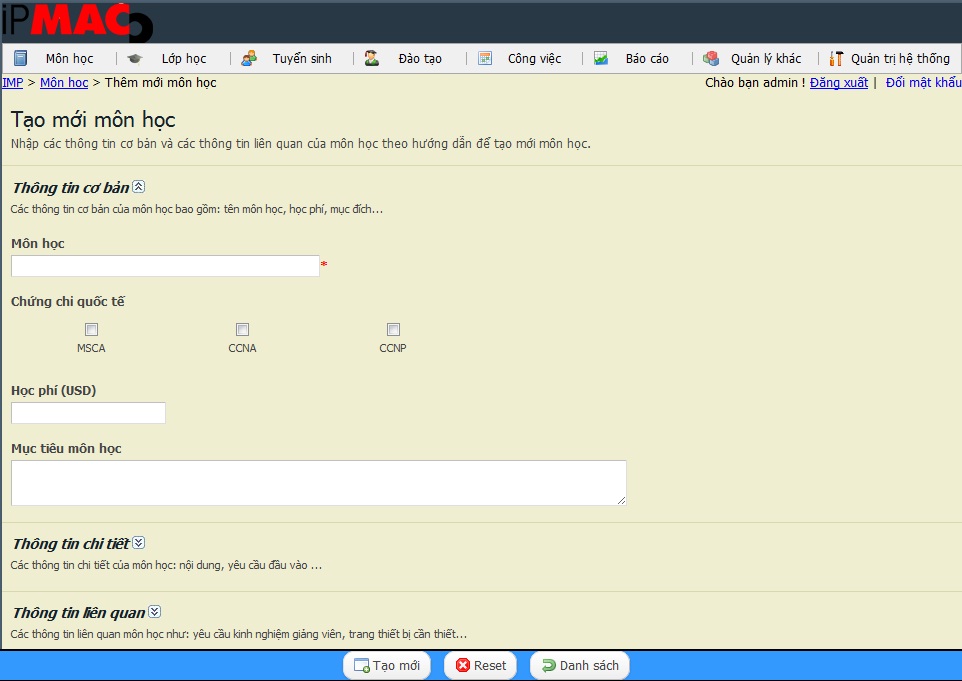


1. ***Các chức năng***
   1. *Quản lí môn học*

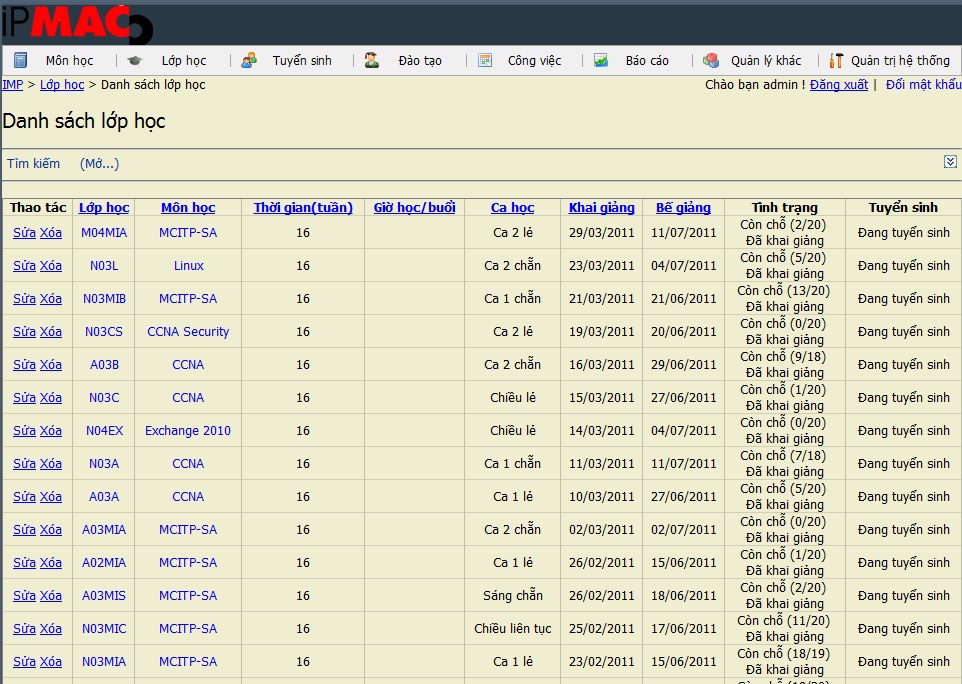
* Cập nhật danh sách Môn học:



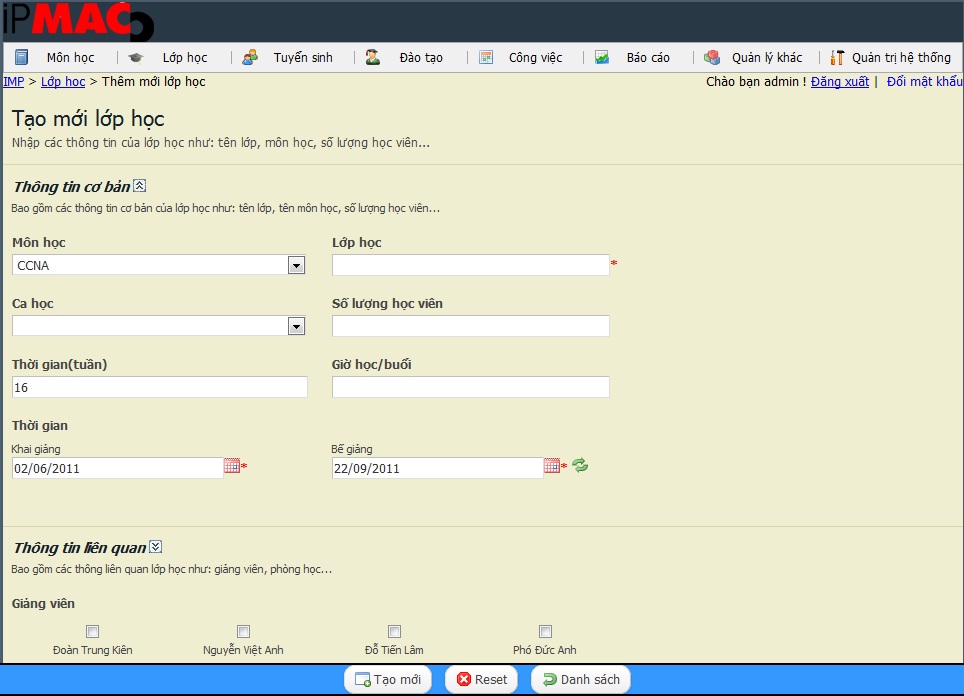
* Thêm mới/Sửa/Xóa thông tin về Môn học:



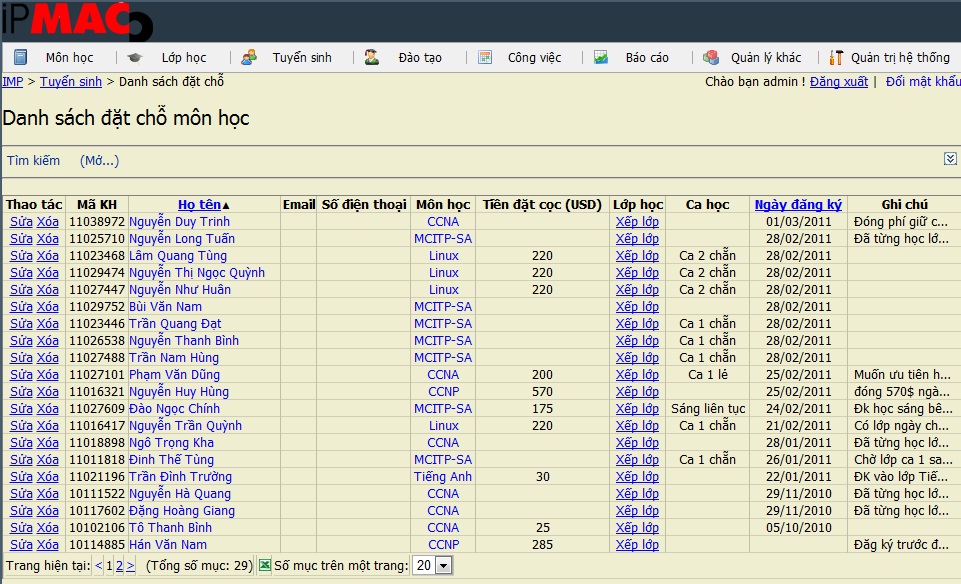
* 1. *Quản lí lớp học*
* Cập nhật danh sách lớp học:



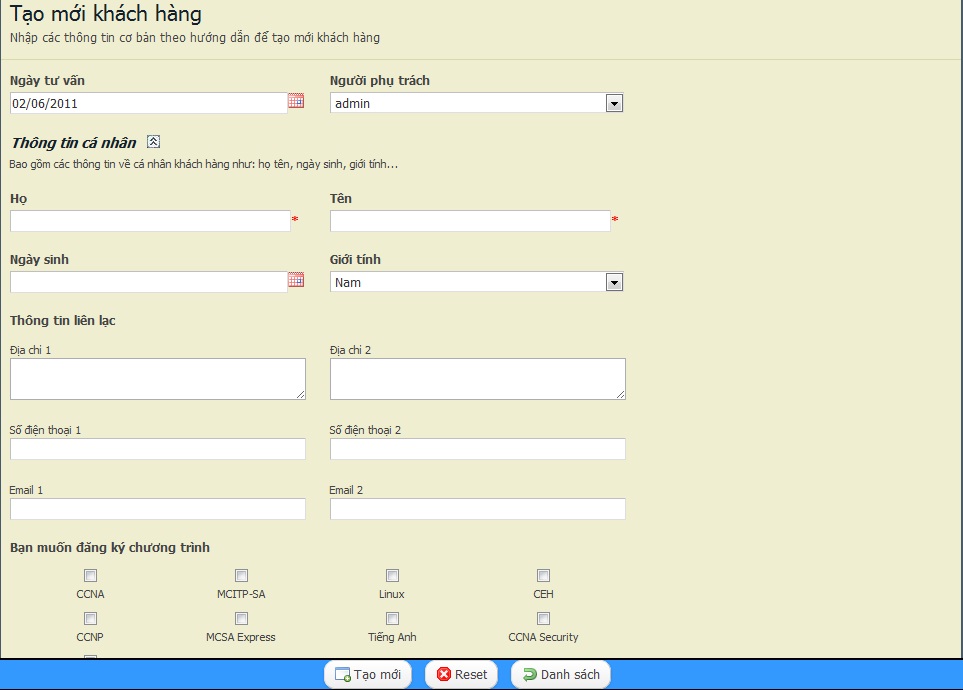
* Thêm mới/Sửa/Xóa thông tin về Lớp học:



* 1. *Quản lí tuyển sinh*
* Cập nhật danh sách khách hàng đặt chỗ môn học/lớp học:



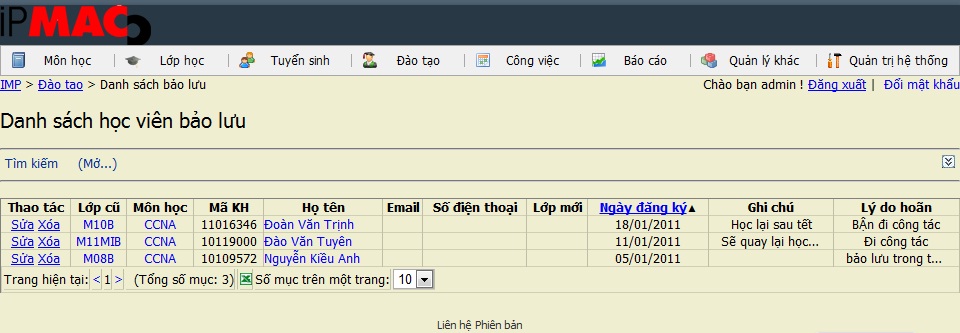
* Thêm mới/Chỉnh sửa/Xóa thông tin về khách hàng:



* 1. *Quản lí đào tạo*
* Cập nhật danh sách xếp lớp cho học viên:



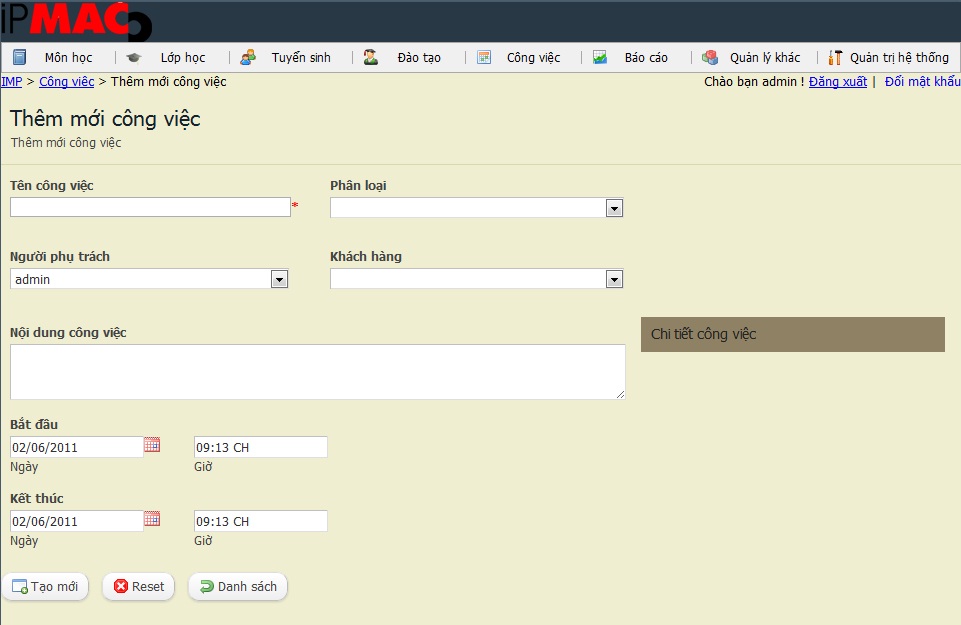
* Cập nhật danh sách bảo lưu lớp của học viên:



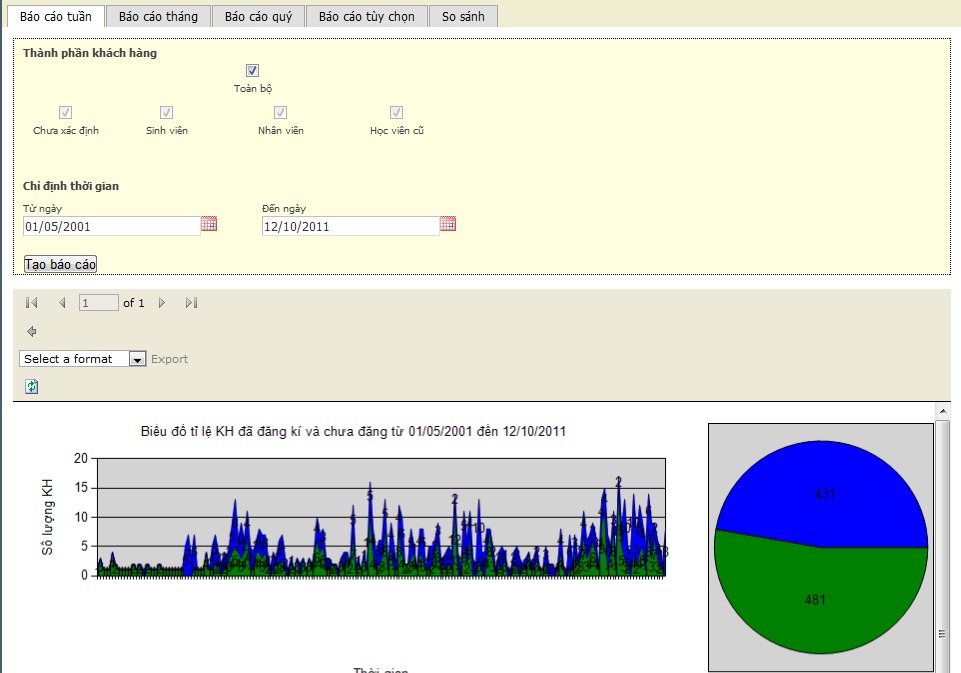
* 1. *Quản lí công việc/lịch hẹn*
* Danh sách công việc/lịch hẹn cho tư vấn viên:



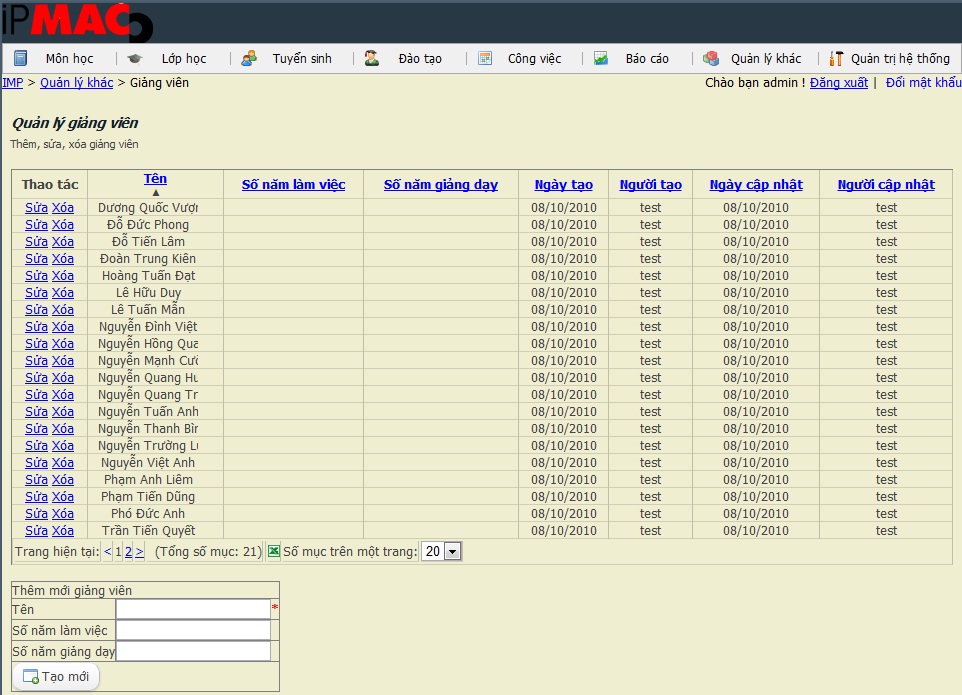
* Thêm mới một công việc/lịch hẹn:



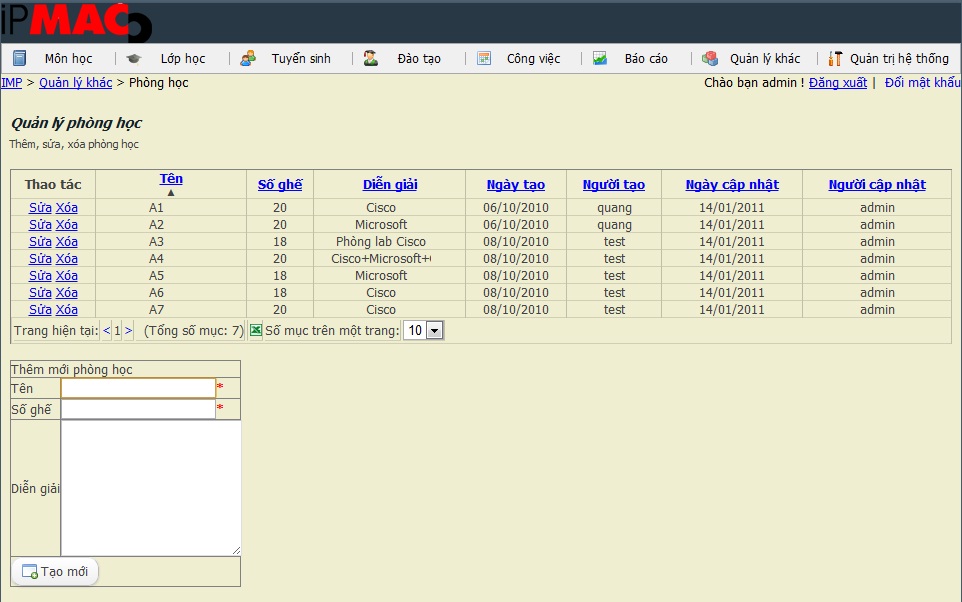
* 1. *Quản lí báo cáo định kì*
* Báo cáo tuyển sinh Khóa học/Lớp học:
* Báo cáo công việc của tư vấn viên:
* Báo cáo theo đối tượng khách hàng:



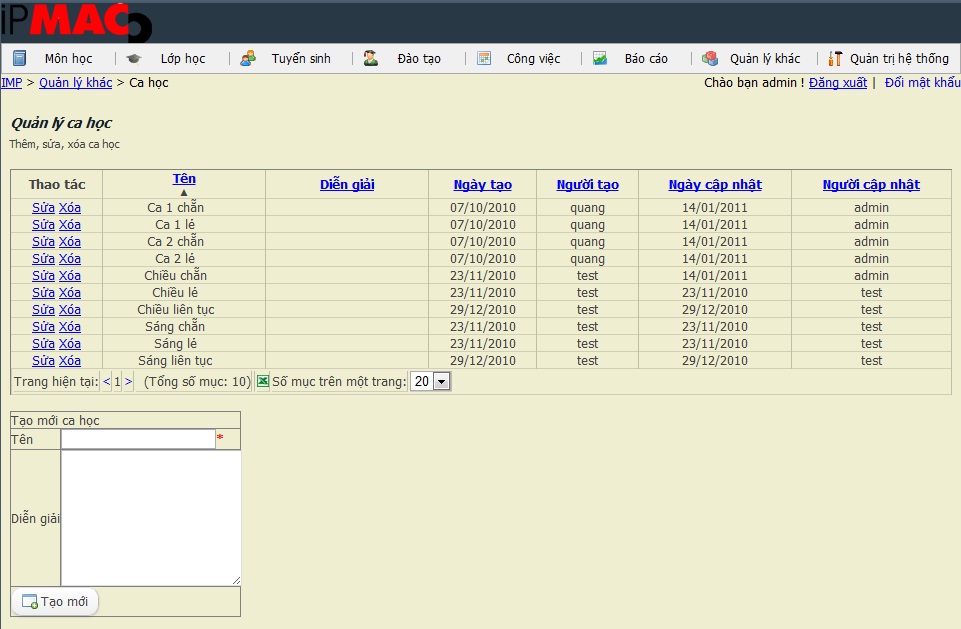
* 1. *Quản lí điều hành*
* Giảng viên:



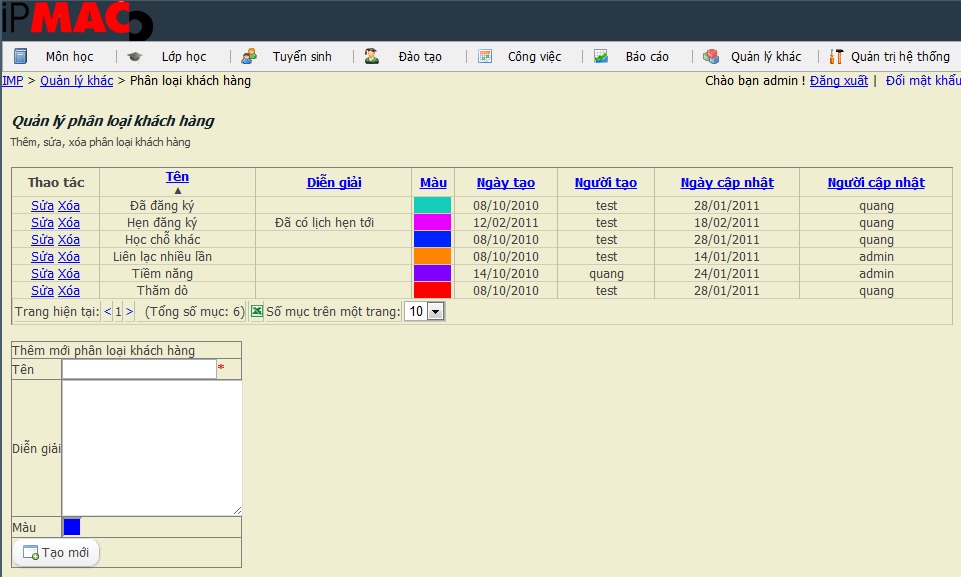
* Phòng học:



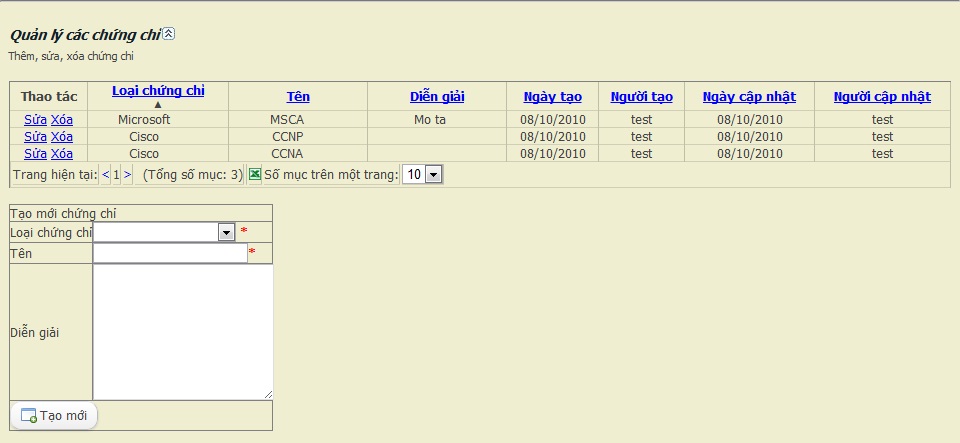
* Ca học:



* Phân loại khách hàng:



* Chứng chỉ của trung tâm:



**KẾT LUẬN**

Các phương thức giao tiếp, truyền thông giữa các ứng dụng trên mạng ngày càng đa dạng phong phú. Ngày nay, với những yêu cầu về hiệu năng, tốc độ, tính ưu việt, các giao diện chuẩn sử dụng cho việc giao tiếp đã được đề xuất, xây dựng thành những chuẩn chung. Điều này giúp cho đội ngũ lập trình, phát triển có thêm nhiều điều kiện để triển khai những ứng dụng phức tạp mà chỉ mất một chi phí nhỏ cho việc tổ chức, viết mã nguồn.

Đối với chuẩn WCF trên nền Web Service, Microsoft đã thực sự thành công trong việc phát triển môi trường lập trình thân thiện, đơn giản, dễ hiểu và hoạt động hết sức hiệu quả, ổn định. WCF không chỉ đáp ứng toàn bộ các yêu cầu nền tảng giao tiếp trên mạng truyền thống mà còn mở rộng, cải tiến thêm rất nhiều những phương thức hoạt động khác như lưu trữ thông điệp dưới dạng tệp tin cấu hình XML... Đây thực sự là một cuộc cải cách mạng trong việc hiện thực hóa chuẩn giao tiếp trên mạng dành cho các ứng dụng.

Trong suốt quá trình tìm hiểu và thực hiện bài tiểu luận này, nhóm đã rất cố gắng hoàn thành tốt nhất những vấn đề cần giải quyết. Tuy vậy, bản báo cáo chắc chắn vẫn còn những thiếu xót cần khắc phục trong thời gian tới. Nhóm xin chân thành cảm ơn sự hướng dẫn, giúp đỡ hết sức tận tình của TS. Phạm Huy Hoàng để thực hiện tốt bài tiểu luận này.

**TÀI LIỆU THAM KHẢO**

[1] TCP/IP Tutorial and Technical Overview – IBM Corp.

[2] Windows Communication Foundation

<http://msdn.microsoft.com/en-au/library/ms735119.aspx>

[3] What Is Windows Communication Foundation?

<http://msdn.microsoft.com/en-us/library/ms731082.aspx>

[4] Windows Communication Foundation

<http://en.wikipedia.org/wiki/Windows_Communication_Foundation>

**PHÂN CÔNG CÔNG VIỆC**

1. Cơ sở lí thuyết
2. Telnet

* Phúc, Khá

1. REXEC và RSH

* Phúc, Khá

1. Distributed Computing Service

* Trung, Thịnh

1. Distributed File Service

* Trung, Thịnh

1. Khảo sát WCF trên Web Service
2. Khái niệm cơ bản

* Khá, Thịnh

1. Kiến trúc tổ chức

* Khá, Thịnh

1. Các tính năng đặc trưng

* Khá, Thịnh

1. Dịch vụ - Service

* Trung, Phúc

1. Kiến trúc Client

* Trung, Phúc

1. Mô hình lập trình

* Trung, Phúc

1. Bảo mật

* Trung, Phúc

1. Ứng dụng kiểm thử

Toàn bộ phần ứng dụng do cả 4 thành viên cùng nhau triển khai (teamwork)