MỤC LỤC

LỜI CẢM ƠN

Lời đầu tiên em xin gửi lời cảm ơn chân thành nhất tới TS. Phùng Văn Ổn đã dành rất nhiều thời gian quí báu để tận tình trực tiếp hướng dẫn, chỉ bảo và định hướng cho em trong suốt thời gian hoàn thành đồ án này.

Em xin bày tỏ lòng biết ơn tới các thầy cô giáo Khoa Công Nghệ Thông Tin – Trường Đại học Dân Lập Hải Phòng đã truyền đạt cho em những kiến thức, kinh nghiệm quý báu trong suốt thời gian học tập tại trường.

Em xin chân thành cảm ơn bạn bè và gia đình, những người thân yêu luôn luôn ở bên khuyến khích, động viên và ủng hộ em trong học tập cũng như trong cuộc sống.

Do thời gian và trình độ có hạn nên đồ án này không thể tránh khỏi những thiếu sót. Rất mong nhận được sự đóng góp ý kiến của các thầy cô giáo, bạn bè, các quý vị quan tâm tới vấn đề này để đồ án được hoàn thiện hơn.

Trân trọng cảm ơn!

Hải Phòng, tháng 7 năm 2010

Mai Quang Huy

MỞ ĐẦU

Sự phát triển của công nghệ thông tin và việc ứng dụng công nghệ thông tin trong nhiều lĩnh vực của đời sống, kinh tế xã hội trong nhiều năm qua cũng đồng nghĩa với lượng dữ liệu đã được các cơ quan thu thập và lưu trữ ngày một tích lũy nhiều lên. Trong khi khối lượng dữ liệu ngày càng phát triển với tốc độ chóng mặt và phân tán khắp nơi thì mỗi hệ thống chỉ cần một số thông tin, dữ liệu nhất định phù hợp với yêu cầu riêng của hệ thống đó và trong nhiều trường hợp để xây dựng một hệ thống cần đến thông tin, dữ liệu từ nhiều nguồn khác nhau. Xuất phát từ thực tế đó dẫn đến yêu cầu phải có phương pháp tích hợp thông tin, dữ liệu từ các nguồn khác nhau để có thể sử dụng tối ưu thông tin, dữ liệu cần thiết và quan trọng là có thể sử dụng thông tin, dữ liệu giữa các hệ thống khác nhau.

Xuất phát từ vấn đề nêu trên em chọn đề tài: " Nghiên cứu và đề xuất giải pháp tích hợp các CSDL phân tán trên môi trường Internet." với mục đích là xây dựng giải pháp tích hợp dữ liệu giúp ích trong công tác quản lý dữ liệu phục vụ các lĩnh vực quản lý.

Đồ án được chia thành ba chương chính:

Chương 1: Trình bày về các khía cạnh của tích hợp dữ liệu một cách tổng quan nhất, các phương pháp tích hợp dữ liệu

Chương 2: Trình bày các giải pháp tích hợp dữ liệu.

Chương 3: Thử nghiệm tích hợp dữ liệu về các cầu trên quốc lộ.

Cuối cùng, phần kết luận trình bày một số kết quả đạt được và những hạn chế của đồ án.

CHƯƠNG 1: GIỚI THIỆU CHUNG

1.1. ĐẶT VẤN ĐỀ

1.1.1 Bài toán tích hợp dữ liệu:

Vấn đề tổng hợp thông tin, đồng bộ dữ liệu từ các nguồn dữ liệu có sẵn là nhu cầu không thể thiếu của bất kỳ hệ thống thông tin nào.

Xuất phát từ yêu cầu đó em nghiên cứu các công nghệ để thực hiện yêu cầu của đồ án. Với bối cảnh chung về sự phát triển công nghệ thông tin trong nước, mỗi cơ quan thường đầu tư phát triển hạ tầng cơ sở viễn thông và các phần mềm ứng dụng theo nhu cầu của từng cơ quan đó, giai đoạn đầu đáp ứng được yêu cầu đặt ra nhưng đến một lúc cần có sự kết hợp và chia sẻ thông tin thì các hệ thống này không đáp ứng được yêu cầu đó hoặc không đáp ứng đầy đủ. Nếu đầu tư mới từ đầu thì vừa lãng phí, vừa khó có thể khai thác các thông tin tích lũy trong nhiều năm qua. Vậy phải có cách nào đó cho phép các hệ thống hiện có vẫn hoạt động bình thường mà vẫn có thể trao đổi thông tin với các hệ thống cũ và mới khác. Trên thực tế bài toán này đã được giải quyết theo nhiều mức độ khác nhau tùy vào mô hình bài toán và sự hỗ trợ của công nghệ hiện thời. Mỗi công nghệ tại một thời điểm chỉ hỗ trợ giải quyết một lớp bài toán nào đó, sau một thời gian lại trở lên lạc hậu, không đáp ứng được yêu cầu công việc. Thế thì, tiêu chí nào cần bổ sung để từ đó ta có thể lựa chọn giải pháp giải quyết được yêu cầu và ít thay đổi theo thời gian hay cụ thể hơn là ít phụ thuộc vào sự thay đổi của hạ tầng viễn thông.

Qua tổng hợp các công nghệ hiện có, ta cần dựa vào một số tiêu chí chính như sau:

- Yêu cầu thực tế của tổ chức.
- Các công nghệ hiện thời có thể dùng để giải quyết bài toán.
- Tài chính của tổ chức.
- Các chuẩn trao đổi thông tin được chuẩn hóa thành chuẩn quốc tế, được nhiều hãng hỗ trợ.

- Chọn lựa giao thức truyền thông không phụ thuộc vào nền tảng cơ sở viễn thông và các phần mềm nền.

1.1.2. Vấn đề tích hợp

Khi nói đến tích hợp dữ liệu chúng ta hình dung ngay đến việc tổng hợp thông tin từ các nguồn dữ liệu có sẵn của các hệ thống khác nhau hay trên cùng một hệ thống thành một nguồn dữ liệu mới có thể dùng cho một hệ thống mới nào đó hay chỉ là để lưu trữ…

Vậy, các công việc được thực hiện thế nào? Trình tự ra sao? Đối với nội dung của đồ án này em đưa ra một số vấn đề nghiên cứu chính như:

- Cách khai thác và vận chuyển thông tin từ các cơ sở dữ liệu của các cơ quan quản lý chuyên ngành chuyển về Trung tâm tích hợp dữ liệu.
- Xây dựng kho lưu trữ thông tin.

1.2 Tổng quan về tích hợp dữ liệu

1.2.1 Khái niệm về tích hợp dữ liệu

Tích hợp dữ liệu là một khái niệm khá trừu tượng thậm chí là hơi mơ hồ khiến nhiều người không thể định nghĩa được chính xác và cụ thể, thông thường tích hợp dữ liệu có thể được hiểu là quá trình kết hợp dữ liệu từ các nguồn thông tin khác nhau nhằm cung cấp cho người dùng một cái nhìn tổng quan và duy nhất về các dữ liệu này. Các đặc điểm của hệ thống tích hợp dữ liệu bao gồm:

Các nguồn dữ liệu là phân tán. Các nguồn dữ liệu này có thể các CSDL trong các hệ thống khác nhau, cũng có thể là các trang Web ở các địa chỉ khác nhau, hoặc cũng có thể là những con người với các quan điểm khác nhau về một vấn đề nào đó.

Các nguồn dữ liệu là không đồng nhất. Sự không đồng nhất này thể hiện ở các ngôn ngữ biểu diễn và từ vựng biểu diễn dữ liệu. Các nguồn dữ liệu có thể có ngôn ngữ biểu diễn khác nhau, ví dụ CSDL của một nguồn được biểu diễn theo dạng XML

nhưng một nguồn dữ liệu khác lại được biểu diễn theo CSDL quan hệ. Các nguồn dữ liệu cũng có thể sử dụng các từ vựng khác nhau để cùng biểu diễn một dữ liệu.

Một hệ tích hợp dữ liệu thường không cần toàn bộ thông tin dữ liệu trong các nguồn cần tích hợp. Với mỗi nhiệm vụ cụ thể, hệ thống chỉ cần những dữ liệu liên quan đến việc thực hiện nhiệm vụ đó. Như vậy nếu tập hợp toàn bộ các nguồn dữ liệu vào hệ thống trước khi tích hợp thì sẽ rất lãng phí và nhiều khi không thể thực hiện được.

Với các đặc điểm như trên, việc xây dựng các hệ tích hợp dữ liệu yêu cầu kiến thức về nhiều lĩnh vực khác nhau như lý thuyết về CSDL, các phương pháp ước lượng, lý thuyết về ngôn ngữ và biểu diễn thông tin,....

1.2.2 Các mức độ tích hợp dữ liệu

Theo Khaled Bashir Shaban, tích hợp dữ liệu được chia thành ba mức dựa trên đặc điểm đầu vào và đầu ra của quá trình tích hợp như sau:

Mức 1: Tích hợp dữ liệu (Data Fusion). Đây là mức thấp nhất. Trong mức này, đầu vào là các bản ghi dữ liệu. Đầu ra cũng có dạng các bản ghi hoặc một dạng cao hơn nhưng vẫn đóng vai trò là dữ liệu cung cấp cho một ứng dụng nào đó.

Mức 2: Tích hợp thông tin (Information Fusion). Trong mức này, cả đầu vào và đầu ra của quá trình tích hợp đều là thông tin, tức là một cấu trúc đầy đủ, tập hợp từ các bản ghi dữ liệu. Mức này xảy ra với các hệ thống nhiều nguồn dữ liệu mà cấu trúc của các nguồn dữ liệu này là khác nhau và mỗi nguồn thông tin không thể tách ra từ một nguồn khác.

Mức 3: Tích hợp quyết định (Decision Fusion). Đây là mức tích hợp thông tin dữ liệu cao nhất. Đầu vào của một hệ thống này có thể là thông tin, dữ liệu, hoặc các quyết định (được biểu diễn theo một dạng cụ thể nào đó) từ các hệ thống khác nhau. Nhiệm vụ của hệ tích hợp dữ liệu ở mức này là phải đưa ra tập quyết định phục vụ yêu cầu đặt ra của hệ thống. Có thể nói tích hợp quyết định phục vụ yêu cầu đặt ra của hệ thống, tích hợp quyết định ở mức trừu tượng cao hơn hai mức trước, do đó nó bao hàm cả hai mức trên. Một điểm khác nhau nữa, nếu như ở mức 1 và mức 2 vẫn có những

trường hợp quá trình tích hợp thông tin dữ liệu không thực hiện được (do không thỏa mãn các điều kiện nào đó) thì mức 3 sẽ luôn được thực hiện vì nó không phụ thuộc vào bản chất và đặc điểm của các nguồn dữ liệu.

Tuy chia làm ba mức như trên nhưng trên thực tế một hệ tích hợp dữ liệu thường có đủ ba mức. Các mức thấp, do đó, sẽ làm cơ sở cho các mức cao hơn.

1.2.3 Các phương pháp tích hợp dữ liệu

Nhu cầu tích hợp dữ liệu trong các hệ thống, nhất là trên môi trường Internet rất lớn. Nhiều nghiên cứu về tích hợp dữ liệu đã được tiến hành. Các nghiên cứu này đưa ra một loạt các phương pháp tích hợp dữ liệu, mỗi phương pháp lại phù hợp với một dạng hệ thống (và các nguồn dữ liệu) cụ thể nào đó. Trong phần này sẽ trình bày một số phương pháp tích hợp dữ liệu theo cách phân loại dựa trên kỹ thuật tích hợp.

1.2.3.1 Tích hợp dữ liệu dựa trên ước lượng không chắc chắn

Hiểu một cách đơn giản, tích hợp dữ liệu dựa trên ước lượng không chắc chắn là phương pháp tính toán độ phù hợp của các dữ liệu thu thập được với yêu cầu của người dùng hoặc ứng dụng cụ thể, sau đó chọn ra dữ liệu có độ phù hợp cao nhất. Để tính toán độ phù hợp, các phương pháp thuộc dạng này sử dụng các ướng lượng không chắc chắn.

Trong các ứng dụng tìm kiếm truy xuất thông tin dữ liệu trên Web quen thuộc như Yahoo, Google, Alta Vista... độ phù hợp của một thông tin dữ liệu được tính qua hai tham số là độ chính xác (precision) và khả năng thu hồi (recall). Từ yêu cầu tìm kiếm thông tin của người dùng, hai tham số trên sẽ được tính toán. Độ chính xác thay thế cho các văn bản phù hợp nhất với người dùng trong các tập văn bản ban đầu. Khả năng thu hồi thay thế cho phần phù hợp nhất bên trong các văn bản tìm được đó. Kết quả trả về sẽ dựa trên cả hai tham số này.

Một phương pháp tích hợp dữ liệu khác sử dụng hệ đa agent. Với mục đích tích hợp và truy xuất các nguồn thông tin dữ liệu trên Internet nhằm tìm ra thông tin dữ liệu phù hợp nhất với người dùng, hệ tích hợp dữ liệu sẽ được tổ chức thành một nhóm các agent khác nhau, mỗi agent có chức năng thu thập thông tin tại một nguồn nhất định.

Phương pháp tích hợp dữ liệu được đưa ra là tổ chức các agent thành các nhóm đồng hướng (team consensus) bao gồm các agent cùng thu thập dữ liệu cho một yêu cầu của người dùng. Các agent trong mỗi nhóm này sẽ thu thập dữ liệu từ các nguồn của mình sau đó dữ liệu sẽ được ước lượng giá trị theo một phương pháp ước lượng không chắc chắn (ước lượng mờ) dựa trên các điều kiện không chắc chắn của agent đó. Cuối cùng, các giá trị dữ liệu sẽ được tính toán, so sánh và lựu chọn theo một thuật toán tích hợp và hệ thống sẽ đưa ra quyết định lựu chọn dữ liệu phù hợp nhất với người dùng.

Nói chung, các phương pháp tích hợp dữ liệu sử dụng ước lượng không chắc chắn đều cần thuật toán tích hợp dữ liệu phức tạp. Mặt khác, việc tính toán độ phù hợp của dữ liệu chưa tính đến sự không đồng nhất về ngữ nghĩa thông tin dữ liệu. Theo nhận định của Morgan Benton và Benjamin K.Ngugi thì phương pháp tính toán độ phù hợp dựa trên hai độ đo: độ phù hợp và khả năng thu hồi có bản chất là so sánh từng bit, do đó không so sánh được ngữ nghĩa thông tin dữ liệu.

1.2.3.2 Tích hợp dữ liệu dựa trên các ràng buộc dữ liệu

Một dạng phương pháp tích hợp dữ liệu khác là dựa trên các ràng buộc dữ liệu. Các phương pháp thuộc về dạng này được áp dụng cho hệ thống bao gồm các nguồn dữ liệu biểu diễn dưới dạng các hệ CSDL và cấu trúc, ràng buộc trong các hệ CSDL này là có thể biết được. Mục đích của các hệ thống này là trả lời các truy vấn của người dùng về thông tin dữ liệu trong nhiều nguồn khác nhau mà không cấn truy nhập trực tiếp vào tất cả các nguồn thông tin này. Tiêu biểu cho phương pháp tích hợp dữ liệu thuộc loại này là phương pháp dùng cho hệ thống IBIS (Internet_base Information System).

Phương pháp tích hợp dữ liệu được đưa ra dựa trên bộ ba lược đồ (G, S, M) được xây dựng từ các nguồn thông tin dữ liệu cần tích hợp:

Lược đồ toàn cục (global schema) G: giống như lược đồ quan hệ trong lý thuyết về CSDL, mô tả các ràng buộc nhất quán, các ràng buộc khóa và các yêu cầu về tính độc lập giữa các nguồn thông tin dữ liệu.

Lược đồ dữ liệu (source schema) S: Mô tả cấu trúc của tập các nguồn dữ liệu cần tích hợp trong hệ thống.

Các ánh xạ M: bao gồm các ánh xạ được thiết lập giữa lược đồ toàn cục và các lược đồ nguồn dữ liệu.

Trên cơ sở xem xét các ràng buộc được định nghĩa trong G và cấu trúc biểu diễn trong S, người thiết kế hệ thống sẽ xác định các ánh xạ tương ứng giữa các thực thể dữ liệu trong các nguồn dữ liệu (ở đây là các CSDL).

Phương pháp này có ưu điểm là biểu diễn được các ngữ nghĩa thông tin dữ liệu thông qua bộ ba (G, S, M) nhưng nhược điểm là cần biết cấu trúc và ràng buộc của các CSDL trong hệ thống. Điều này không phải lúc nào cũng thực hiện được.

1.2.3.3 Tích hợp dữ liệu tự động dựa trên ontology

Nhiều nghiên cứu khác nhau đã khẳng định phương pháp tích hợp dữ liệu dựa trên ontology có một số ưu điểm so với hai dạng phương pháp đã trình bày ở trên. Thay vì sử dụng các ước lượng không chắc chắn hoặc các lược đồ CSDL, các phương pháp dựa trên ontology sử dụng một cấu trúc phân lớp các khái niệm, thuật ngữ và các quan hệ giữa các khái niệm đó gọi là ontology để biểu diễn các nguồn dữ liệu cần tích hợp (cả nội dung và ngữ nghĩa thông tin dữ liệu). Thông qua tương tác giữa các thành phần dựa trên ontology, dữ liệu từ các nguồn được tích hợp.

Vì ontology biểu diễn ngữ nghĩa thông tin dữ liệu thông qua các khái niệm và câc mối quan hệ giữa các khái niệm nên phương pháp tích hợp dữ liệu dựa trên ontology giải quyết được vấn đề không đồng nhất về ngữ nghĩa thông tin dữ liệu. Quá trình tích hợp dữ liệu sẽ diễn ra một cách tự động thông qua việc xác định các ánh xạ tương đương hoặc không tương đương giữa các khái niệm trong các ontology khác nhau.

Có nhiều nghiên cứu khác nhau về tích hợp dữ liệu dựa trên ontology trong hệ đa agent. Trong các nghiên cứu này, Agustina Buccella và H.Stuckenschmidt xây dựng phương pháp tích hợp dữ liệu sử dụng bộ từ vựng chung (shared vocalbulary) còn Soe-Tsyr Yuan xây dựng phương pháp tích hợp dữ liệu sử dụng agent trung gian.

CHƯƠNG 2: GIẢI PHÁP TÍCH HỢP CÁC CSDL

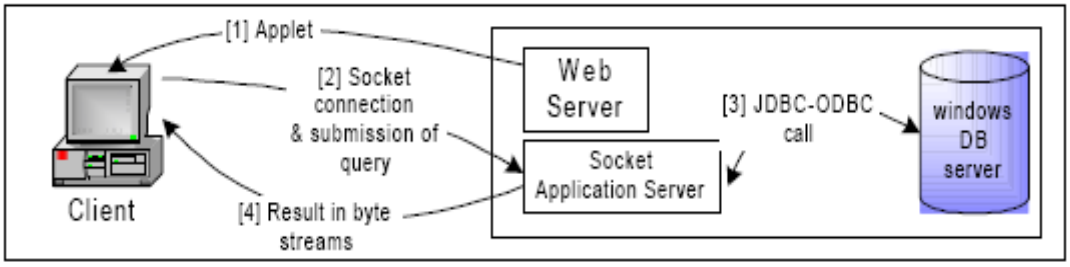
2.1. MỘT SỐ CÔNG NGHỆ XỬ LÝ CSDL TRÊN MÔI TRƯỜNG MẠNG

Hiện nay việc trao đổi thông tin không còn giới hạn về không gian và thời gian, sự phát triển đó là nhờ sự phát triển của khoa học và chính sách chính sách của các quốc gia nói chung và của các tổ chức nói riêng. Xây dựng các hệ thống mạng là không thể thiếu đối với bất kỳ quốc gia hay tổ chức nào trên thế giới, điển hình là mạng Internet. Đồng thời việc xử lý cơ sở dữ liệu trên môi trường mạng là một thách thức cũng là mục tiêu của ngành công nghệ thông tin.

2.1.1 Một số phương pháp truyền thống khai thác dữ liệu dựa trên Web

2.1.1.1 Phương pháp Java Socket

Ngôn ngữ lập trình Java hỗ trợ hai dạng chương trình ứng dụng chính là ứng dụng độc lập (Java application) và ứng dụng nhúng (Java applet). Các Java applet có thể được máy khách tải xuống từ một máy ở xa thông qua trình duyệt Web và thực thi tại máy khách, do tính bảo mật của ngôn ngữ Java nên máy ảo Java sẽ không cho phép các Java applet được quyền truy nhập tài nguyên cục bộ như cơ sở dữ liệu Web đặt trên máy server, vì vậy để bảo đảm được hai yếu tố của phương pháp Java socket là truy nhập cơ sở dữ liệu từ xa thông qua trình duyệt Web và nhận được kết quả trả về cần có thêm thành phần trung gian đứng giữa máy khách và cơ sở dữ liệu do Web trả về. Thành phần trung gian trong phương pháp Java socket là một chương trình ứng dụng độc lập.



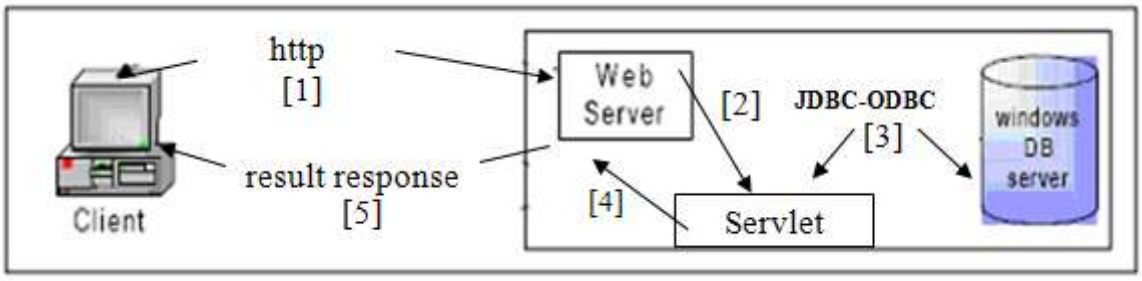
Hình 1: Mô hình truy nhập cơ sở dữ liệu Web bằng Java Socket

Hoạt động của mô hình truy nhập cơ sở dữ liệu thông qua Web bằng phương pháp Java socket thực hiện qua những bước sau :

- Máy khách truy nhập vào máy chủ Web thông qua trình duyệt Web, trang Web và ứng dụng Java applet có chức năng truy nhập cơ sở dữ liệu từ máy chủ Web được tải về máy khách.
- Ứng dụng Java applet truy cập cơ sở dữ liệu được khởi động tại máy khách bởi người dùng và kết nối tới thành phần trung gian trên máy chủ Web, khi kết nối thành công thì máy khách gửi yêu cầu truy cập dữ liệu cho thành phần trung gian trên máy chủ Web.
- Kết nối được chấp nhận thì chương trình trung gian sẽ truy cập vào cơ sở dữ liệu đặt trên máy chủ Web lấy dữ liệu theo yêu cầu của máy khách.
- Thành phần trung gian trả dữ liệu kết quả về cho ứng dụng Java applet ở phía máy khách, sau đó applet chuyển dữ liệu kết quả cho trình duyệt Web để nó hiển thị dữ liệu kết quả lên cho người dùng.

2.1.1.2 Phương pháp Servlets Java

Phương pháp Servlets thường được dùng để tạo ra các trang Web động, mọi thao tác xử lý theo yêu cầu của máy khách được thực hiện tại server như viết mã lệnh để tạo ra trang Web, truy nhập cơ sở dữ liệu... điều này rất có ý nghĩa trong trường hợp các máy khách có năng lực xử lý hạn chế. Một ưu điểm nổi bật của phương pháp Servlet là giúp giảm tải mạng, do không cần phải duy trì một kết nối mạng thường xuyên giữa máy khách và máy chủ trong quá trình máy khách truy cập cơ sở dữ liệu.



Hình 2: Mô hình truy nhập cơ sở dữ liệu bằng Servlet

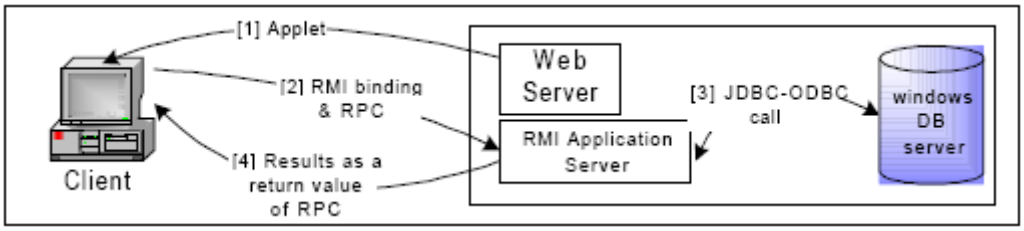
Thành phần trung gian trong phương pháp này là một Servlet, nó là một chương trình Java được thực hiện như là một tiến trình con trong môi trường của một trình chủ Web có hỗ trợ Java. Trình chủ Web có nhiệm vụ định tuyến cho các yêu cầu từ phía máy khách đến được servlet có nhiệm vụ thực thi yêu cầu đó, ngoài ra trình chủ Web còn đảm nhiệm các công việc: nạp, khởi động, chạy và kết thúc các servlet.

Hoạt động của mô hình truy nhập cơ sở dữ liệu bằng Servlet thực hiện theo các bước như sau

- Máy khách truy nhập Web trên máy chủ bằng trình duyệt Web.
- Máy chủ Web gọi servlet tương ứng thực thi yêu cầu từ phía máy khách.
- Chương trình servlet truy nhập vào cơ sở dữ liệu cục bộ lấy dữ liệu theo yêu cầu của máy khách.
- Chương trình servlet chuyển dữ liệu kết quả cho trình chủ Web
- Trình chủ Web trả dữ liệu kết quả cho máy khách. Trình duyệt Web tại máy khách sẽ hiển thị dữ liệu đã yêu cầu lên cho người dùng.

2.1.1.3 Phương pháp RMI

RMI là một giao diện ứng dụng cho phép thực thi các lời gọi phương thức từ xa giữa các đối tượng Java phân tán.



Hình 3: Mô hình truy nhập cơ sở dữ liệu Web bằng RMI

Thành phần trung gian trong phương pháp RMI bao gồm hai đối tượng :

- Chương trình ứng dụng độc lập Java, làm nhiệm vụ cài đặt và thực hiện các phương thức được máy khách triệu gọi từ xa.

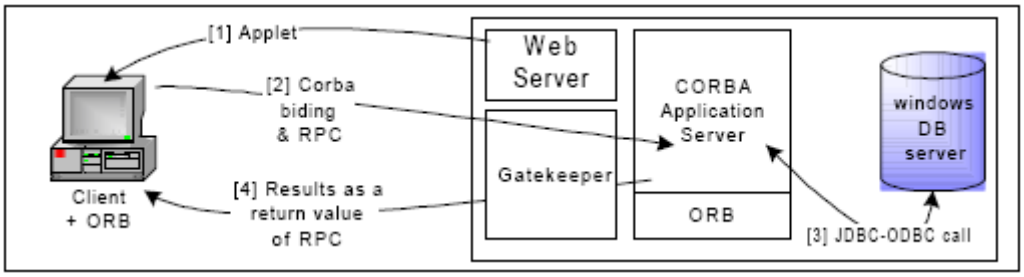
- Ứng dụng nền Rmiregistry.exe đi kèm trong bộ JDK từ phiên bản 1.3 trở lên làm hai nhiệm vụ: Khởi động ứng dụng của máy chủ và đăng ký tên duy nhất cho ứng dụng máy chủ với máy ảo Java chạy trên trình chủ Web.

Hoạt động của mô hình truy nhập cơ sở dữ liệu Web bằng phương pháp RMI thực hiện qua những bước sau :

- Máy khách truy nhập vào máy chủ Web thông quan trình duyệt Web. Java applet có nhiệm vụ truy nhập cơ sở dữ liệu Web bằng lời gọi phương thức từ xa được tải từ máy chủ về máy khách cùng với trang Web của máy chủ Web.
- Applet truy nhập cơ sở dữ liệu Web được người dùng kích hoạt sẽ thực hiện tìm kiếm đối tượng từ xa trên máy chủ Web dựa vào trình đăng ký tên dịch vụ duy nhất Rmiregistry.exe chạy trên máy chủ Web, nếu tìm thấy applet thực hiện lời gọi phương thức từ xa để lấy dữ liệu.
- Ứng dụng của máy chủ đáp ứng yêu cầu được trình đăng ký tên dịch vụ duy nhất chạy trên máy chủ Web khởi động và thực hiện truy nhập cơ sở dữ liệu để lấy dữ liệu theo yêu cầu của máy khách.
- Ứng dụng server trả dữ liệu kết quả về cho máy khách bằng phương thức được gọi từ xa của nó.

2.1.1.4 Phương pháp CORBA

CORBA là một chuẩn đối tượng phân tán, định nghĩa các mối quan hệ khách/chủ (client/server) giữa các đối tượng trong một ngôn ngữ giao diện chung (common interface language). Chương trình RMI chỉ cài đặt có thể thực thi bằng ngôn ngữ lập trình Java nhưng chương trình CORBA có thể được cài đặt và thực thi bằng một ngôn ngữ lập trình bất kỳ.



Hình 4: Mô hình truy nhập cơ sở dữ Web bằng Java CORBA

Đối tượng ứng dụng máy khách CORBA muốn gọi đúng được đối tượng ứng dụng máy chủ CORBA cần có một đối tượng thứ ba có thể cung cấp phương tiện giao tiếp giữa các ứng dụng, dịch vụ và các tiện ích mạng gọi là ORB (Object Request Broker). ORB được quan niệm như là một loại bus mềm hay đường trục sống, cung cấp các giao diện chung giữa nhiều loại đối tượng khác nhau để có thể giao tiếp được với nhau theo mô hình bình đẳng.

Đối tượng máy khách gửi yêu cầu đến ORB, nhiệm vụ của ORB là tìm đối tượng máy chủ hay tìm đối tượng có thể biết các máy chủ, sau đó thiết lập quá trình truyền thông giữa máy khách và máy chủ này. Đối tượng máy chủ gửi đáp ứng cho ORB, nó định dạng lại và chuyển tiếp đáp ứng về cho nơi phát ra yêu cầu. ORB phải được nạp trên cả máy chủ và máy khách. Về vấn đề bảo mật, CORBA chỉ cho phép một applet kết nối trực tiếp từ xa vào đối tượng máy chủ CORBA qua tường lửa gọi là IIOP (Internet Inter ORB Protocol). IIOP là một phần của CORBA, nó cung cấp phương tiện để các đối tượng CORBA có thể tương tác với mạng TCP/IP, bao gồm cả mạng Internet. IIOP kết hợp hoặc thay thế cho HTTP, một giao thức cơ bản trên Internet.

Ngoại trừ giao thức IIOP, thành phần trung gian trong cách tiếp cận CORBA giống như thành phần trung gian trong cách tiếp cận RMI.

Hoạt động của mô hình truy cập cơ sở dữ liệu Web bằng cách tiếp cận CORBA thực hiện theo các bước sau :

- Máy khách truy nhập vào máy chủ Web, applet có chức năng truy nhập cơ sở dữ liệu Web được tải về máy khách từ máy chủ.

- Applet được khởi động từ phía máy khách. Sau khi nạp xong ORB, nó kết nối với ứng dụng của máy chủ CORBA thông qua Gatekeeper bằng cách gọi một phương thức đặc biệt và chuyển tên dịch vụ duy nhất của ứng dụng máy chủ đi giống như tham số của phương thức.
- Ứng dụng CORBA Server thực hiện truy nhập cơ sở dữ liệu Web cục bộ, lấy dữ liệu theo yêu cầu của phía máy khách.

Ứng dụng máy chủ CORBA gửi dữ liệu kết quả về cho phía máy khách giống như giá trị trả về của lời gọi phương thức.

2.1.2 Phương pháp khai thác dữ liệu dựa trên Web service

Web service là phương pháp cho phép trao đổi thông tin giữa các hệ thống dựa trên giao thức HTTP và SOAP, hoàn toàn độc lập với hệ điều hành hoặc ngôn ngữ lập trình được sử dụng trên máy chủ và máy khách. Không như các công nghệ trước kia, Web service không nhất thiết bắt buộc hai đầu kết nối phải cùng hệ điều hành hoặc cùng ngôn ngữ lập trình. Thí dụ, chương trình phía máy chủ có thể viết bằng ngôn ngữ VB.NET cài đặt trên hệ điều hành Window 2000 trong khi chương trình phía máy khách viết bằng ngôn ngữ lập trình khác chạy trên hệ điều hành Unix, hay ngược lại. Nói cách khác, công nghệ cũ yêu cầu các kết nối là kết nối chặt chẽ, thì Web service cho phép máy khách và máy chủ kết nối lỏng lẻo. Máy khách và máy chủ đều nhận được sự hỗ trợ của giao thức chuẩn HTTP, SOAP và XML. HTTP là giao thức được dùng bởi Web, còn SOAP là giao thức hướng đối tượng dựa trên XML lại trở thành chuẩn cho việc định dạng và tổ chức thông tin.

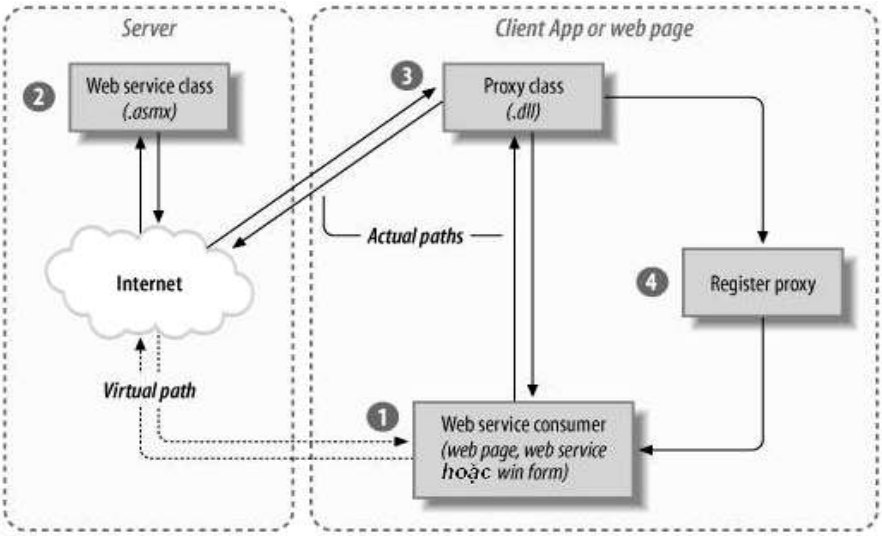
Web service cho phép một đối tượng nằm trên máy chủ có thể đưa ra phần logic chương trình cho các máy khách trên Internet. Các máy khách gọi các phương thức đã trưng ra trên Web service thông qua việc sử dụng các giao thức chuẩn của Internet.

Nền tảng Web service có một số đặc trưng như sau:

- Cả Web service lẫn ứng dụng khách được kết nối trên Internet.
- Dạng dữ liệu mà hai phía liên lạc với nhau cùng tuân theo một chuẩn mở. Chuẩn này thường là giao thức SOAP, các thông điệp SOAP gồm

các tài liệu XML dạng văn bản và tự mô tả. Tuy nhiên nó là kỹ thuật có khả năng liên lạc theo các yêu cầu HTTP-GET và HTTP-POST.

- Hệ thống hai đầu kết nối sẽ được gắn kết một cách lỏng lẻo. Hay nói cách khác là Web service không cần quan tâm mô hình đối tượng, ngôn ngữ lập trình được dùng đến ở hai đầu kết nối là gì, miễn là Web service và ứng dụng tiêu thụ (Consummer Application) có khả năng nhận và gửi các thông điệp tuân thủ theo giao thức chuẩn thích ứng.



Hình 5: Web service nhìn từ trong

Trên hình 5, vị trí ❶, một chương trình khai thác Web service (Web service consumer) đưa ra một lời gọi (vị trí ❷), phía khai thác tưởng rằng mình nói chuyện trực tiếp với Web service thông qua Internet. Thực ra, đây là một lời gọi phương thức từ Proxy (vị trí ❸) nằm ngay trên máy khách, Proxy điều khiển ngay tất cả các cấu trúc phức tạp của việc chuyển các yêu cầu về máy chủ qua Internet, cũng như nhận kết quả từ máy chủ trả về cho máy tiêu thụ. Tất cả việc này có thể thực hiện được là nhờ Proxy trước đó đã đăng ký với ứng dụng tiêu thụ (vị trí ❹), được thực hiện bởi lập trình viên viết ứng dụng tiêu thụ.

Ngoài việc tạo các Web service cũng như ứng dụng tiêu thụ Web service, còn một số vấn đề cần quan tâm:

- **Protocol**

Web service phải liên lạc với máy khách và ngược lại theo một giao thức nào đó mà cả hai phía đều hiểu nhau.

- **Directories**

Các Web service được phát triển bởi hàng ngàn các công ty khác nhau trên thế giới. Directories được tạo ra để liệt kê các dịch vụ này và hiện sẵn dành cho lập trình viên triển khai. Tuy nhiên, muốn cho các thư mục này hữu ích phải có những quy ước liên quan đến khám phá (discovery) và mô tả (description).

Discovery

Các máy khách cần sẽ biết tìm ở đâu những tài liệu mô tả Web service. Như vậy, Web service thường sẽ cung cấp những tài liệu khám phá những tập tin XML chứa thông tin cho phép những khách hàng tiềm năng tìm ra các tập tin khác mô tả Web service.

Description

Một khi Web service được nhận diện, thông qua khám phá hay những phương tiện nào đó, nó phải làm sẵn một tài liệu mô tả những giao thức hỗ trợ và giao diện lập trình cho việc sử dụng Web service. WSDL (Web service Description Language) sẽ được dùng để mô tả Web service, tất cả các phương thức và thuộc tính được trưng ra, bao gồm các kiểu và tham số của phương thức đó.

- **Security**

Phần lớn các máy chủ được kết nối Internet thì sự quan tâm về mặt an toàn lúc nào cũng được đề cập như một phần quan trọng trong hệ thống. Web service phải được đảm bảo về mặt an toàn. Web service không phải là các cổng thông tin cho mọi loại phần mềm và người dùng hỗn độn. Nó chỉ cho phép một số người dùng có quyền truy cập để gọi các phương thức.

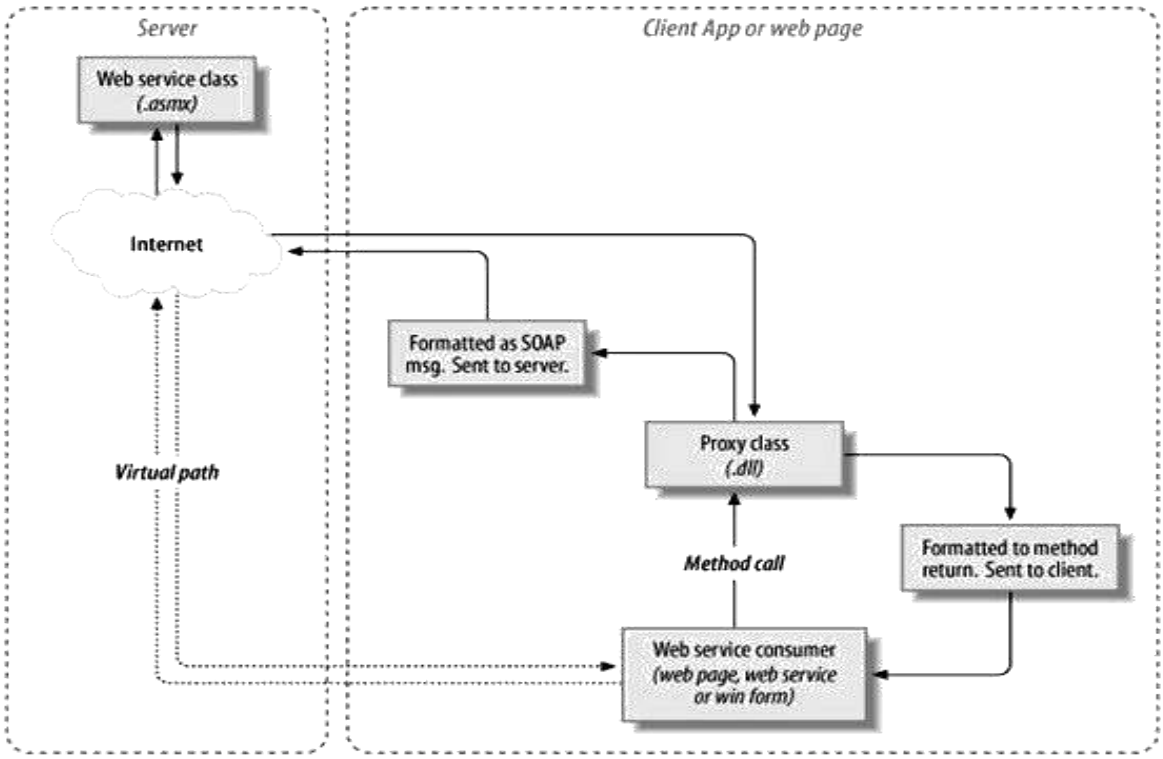
- **State**

Giống như trang Web, Web service sử dụng HTTP, là một giao thức không trạng thái. Do vậy, .NET framework cung cấp các công cụ cho phép duy trì tình trạng nếu các ứng dụng này yêu cầu.

- **Proxy**

Trước khi ứng dụng máy khách có thể dùng được Web service, Proxy phải được tạo. Proxy đóng vai trò thay thế cho các phương thức được gọi. Nó chịu trách nhiệm sắp xếp, dẫn dắt các lời gọi phương thức vượt qua ranh giới các máy tính. Các yêu cầu gọi tới Web service trên máy chủ phải phù hợp với giao thức và định dạng tương ứng, thường là SOAP kết hợp với HTTP.

Proxy phải được đăng ký với ứng dụng máy khách, ứng dụng máy khách tạo các phương thức gọi như gọi các phương thức đó là đối tượng nội bộ. Proxy làm tất cả công việc khi máy khách có lời gọi, gói chúng trong định dạng thích hợp và gửi đi như một yêu cầu SOAP tới máy chủ. Khi máy chủ trả về máy khách gói tin SOAP, Proxy giải mã tất cả và hiển thị chúng trong ứng dụng máy khách như nó được lấy từ đối tượng cục bộ. Tiến trình này được mô tả trong hình sau:



Hình 6: Hoạt động của Proxy

2.1.2.1 HTTP (Hypertext Transfer Protocol)

HTTP là giao thức nằm ở tầng trên cùng của TCP/IP, được dùng để các máy chủ Web và trình duyệt Internet khả năng liên lạc được với nhau. Trình duyệt của máy khách gửi một HTTP request cho máy chủ Web, yêu cầu này được xử lý, sau đó gửi kết quả đã xử lý về cho trình duyệt của máy khách. Trong trường hợp Web service, dữ liệu được trả về là một thông điệp SOAP chứa thông tin kết quả của việc thực thi một lời gọi hàm Web service. HTTP request sẽ trao cặp Name/Value gửi đi một yêu cầu tới máy chủ. Yêu cầu có thể là HTTP-GET hoặc HTTP-POST.

a. HTTP-GET

Trên các GET request, các cặp Name/Value sẽ được ghi nối đuôi trực tiếp trên URL. Dữ liệu không được mã hóa (để nguyên dạng ASCII) được ghi nối đuôi vào URL, phân tách bởi dấu ”?”. Thí dụ:

http://localhost/StockSticker1/Service1.asmx/GetName?StockSymbol=msft

- Dấu hỏi cho biết là một HTTP-GET request
- Tên phương thức GetName
- Tên biến là StockSymbol với giá trị là msft.

GET request chỉ thích hợp khi tất cả các dữ liệu nhỏ chỉ toàn các cặp Name/Value và GET request thích hợp khi an toàn không phải là một vấn đề.

.Net framework cung cấp một lớp *HttpGetClientProtocol* để dùng giao thức HTTP-GET trên các ứng dụng máy khách.

b. HTTP-POST

Trên các POST request các cặp Name/Value cũng không được mã hóa, nhưng thay vì nối đuôi sau URL thì chúng được gửi đi như là thành phần của thông điệp yêu cầu.

POST request thích hợp với lượng thông tin khá nhiều. Ngoài ra vấn đề an toàn là quan trọng thì một POST request sẽ an toàn hơn một GET request vì POST request có thể được mã hóa.

.NET framework cung cấp lớp *HttpPostClientProtocol* để dùng trong giao thức HTTP-POST trong các ứng dụng máy khách.

2.1.2.2. SOAP (Simple Object Access Protocol)

Giao thức SOAP là một đặc tả thông điệp XML để mô tả một khuôn dạng thông báo cùng với một tập các quy tắc cho kiểu dữ liệu cùng với các kiểu cấu trúc và kiểu mảng. Ngoài ra, nó mô tả làm cách nào để sử dụng giao thức HTTP như một sự chuyên trở cho những thông báo như vậy.

Thông điệp SOAP có hiệu quả là gửi các yêu cầu dịch vụ cho kết nối đầu cuối trên mạng. Thiết bị cuối đó có thể thực hiện bất kỳ cách nào trong các cách sau đây Remote Protocol Call (RPC) server, Component Object Model (COM) object, Java servlet, Perl script và có thể chạy trên bất kỳ nền tảng nào (any platform).

Như vậy, SOAP sẽ là thao tác trung gian giữa các ứng dụng chạy trên các nền tảng sử dụng nhiều công nghệ, thi hành trong nhiều ngôn ngữ lập trình khác nhau.

2.1.2.3. XML (eXtensible Markup Language)

a. Giới thiệu

XML (eXtensible Markup Language) được ra đời từ việc giảm thiểu độ phức tạp của SGML (Standard Generalized Markup Language), là ngôn ngữ có kiến trúc gần giống với HTML nhưng XML nhanh chóng trở thành một chuẩn phổ biến trong việc chuyển đổi thông tin qua các trang Web sử dụng giao thức HTTP. Trong khi HTML là ngôn ngữ chủ yếu về hiển thị dữ liệu thì XML lại là ngôn ngữ trung gian trong việc trao đổi dữ liệu giữa các hệ thống khác nhau, trao đổi và thao tác dữ liệu bằng XML. XML đưa ra một định dạng chuẩn cho cấu trúc của dữ liệu hoặc thông tin bằng việc tự định nghĩa định dạng của tài liệu. Bằng cách này, dữ liệu được lưu trữ bằng XML sẽ độc lập với việc xử lý. Vì vậy XML ra đời sẽ đáp ứng được yêu cầu ngày càng cao của các nhà lập trình trong vấn đề trao đổi và xử lý thông tin.

b. Cấu trúc chung của XML

Chúng ta có thể sử dụng trình soạn thảo bất kỳ để soạn thảo tài liệu XML, nhưng phải tuân thủ theo nguyên tắc sau:

```
<root>
      <child>
            <subchild>…..</subchild>
           ....
      </child>
      ....
</root>
```

Theo định dạng trên, chúng ta thấy tuy tài liệu XML rất đơn giản nhưng quy định cũng rất chặt chẽ, tức là các tài liệu XML đều xuất phát từ nút gốc (root), và mỗi phần tử phải có thẻ mở và thẻ đóng "<tên thẻ > … </ tên thẻ>".

c. Lược đồ XML (XML Schema)

- *Cấu trúc lược đồ (Schema structure)*

Một lược đồ là một tập những quy tắc được định nghĩa để mô tả nội dung dữ liệu của một tài liệu XML, nó quy định thành phần nào của XML thuộc kiểu dữ liệu gì.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://tempuri.org/XMLSchema1.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:complexType name="addressType">
  <xs:sequence>
    <xs:element name="street1" type="xs:string"/>
    <xs:element name="street2" type="xs:string"/>
    <xs:element name="city" type="xs:string"/>
    <xs:element name="state" type="xs:string"/>
    <xs:element name="zip" type="xs:integer"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="purchaseOrder">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="shipTo" type="addressType" />
        <xs:element name="billTo" type="addressType" />
        <xs:element name="shipDate" type="xs:date" />
        <xs:element name="item" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

- *Các kiểu dữ liệu trong lược đồ XML*

Khi tập tin XML hoạt động như một cơ sở dữ liệu, và XSL, XPath được sử dụng để truy vấn trên tập tin XML giống như một số ngôn ngữ truy vấn trong SQL, thì lúc này chúng ta cần biết được vị trí của từng phần tử trong tập tin XML được khai báo ở đâu và với kiểu dữ liệu như thế nào.

Có hai loại kiểu dữ liệu trong lược đồ XML đó là kiểu dữ liệu cơ bản và kiểu dữ liệu mở rộng. Kiểu dữ liệu cơ bản là kiểu dữ liệu không bắt nguồn từ kiểu dữ liệu nào ví dụ như kiểu dữ liệu float. Kiểu dữ liệu mở rộng dựa trên những kiểu dữ liệu khác.

d. DOM

XML Document Object Model (DOM) dùng để phân cấp dữ liệu XML thành cấu trúc cây, điều này tạo ra cách thức truy cập vào tài liệu XML, có nghĩa là chỉ xử lý phần văn bản bị thay đổi chúng ta dùng Xpath để truy cập vào cây do DOM tạo ra. Chúng ta dùng các phương thức **Xmlreader**, **Xmlwriter**,…

e. XPath

Để xử lý một tài liệu XML, chương trình ứng dụng phải có cách di chuyển bên trong tài liệu để lấy ra giá trị của các phần tử (Elements) hay thuộc tính (Attributes). Do đó ngôn ngữ **XML Path** được ra đời, mà chúng ta gọi tắt là **XPath**. XPath đóng một vai trò quan trọng trong việc truy vấn dữ liệu cho các chương trình ứng dụng vì nó cho phép ta lựa chọn hay sàng lọc ra những phần tử nào mình muốn để xử lý hay hiển thị.

2.1.2.4. Khai thác các Web Service

a. Discovery

Discovery là quá trình tìm ra các Web service đang sẵn có, các thuộc tính và phương thức được trưng ra bởi một Web service đã xác định. Các tham số đầu vào của các thuộc tính và phương thức được truyền, kiểu dữ liệu mà phương thức Web hay thuộc tính trả về. Tất cả các thông tin này được chứa trong văn bản WSDL (Web Services Description Language). Nếu nhà phát triển ứng dụng tiêu thụ Web service biết URL của tệp tin Web service (*.asmx) thì không cần phải khám phá (discovery). Tuy nhiên, nhà phát triển ứng dụng tiêu thụ Web service thường không biết địa chỉ của

Web service file hoặc văn bản WSDL trên máy chủ đã cho. Trong những trường hợp này, Visual Studio.net cung cấp tiện ích Discovery theo chế độ dòng lệnh là Disco.exe mà coi URL của một Web service là một đối số và tạo ra văn bản Discovery trên máy khách, từ đó nhà phát triển ứng dụng có thể tạo ứng dụng tiêu thụ các Web service.

Lệnh này sẽ tìm kiếm trên URL xác định bất kỳ văn bản Discovery và ghi chúng và thư mục hiện thời của máy cục bộ. Một .wsdl được sinh ra và được ghi lại. Để chỉ định thư mục khác với thư mục hiện thời dùng tham số /out:, hoặc dùng /o: để rút gọn.

disco /out:<output directory name> http://localhost/csStockTicker.asmx

Tiện ích Disco thi hành dòng lệnh trên sau đó ghi 3 tệp tin vào thư mục đầu ra.

Tên tệp tin	Mô tả
service.disco	*Tài liệu discovery tạo ra*
service.wsdl	*Tệp này giống như WSDL đối với Web service được kết sinh khi URL có tệp .asmx cho thêm ?wsdl vào sau.*
results.discomap	*Văn bản discovery chuyển đổi*

Muốn biết toàn bộ thông số của tiện ích Disco.exe ta dùng câu lệnh tại của sổ lệnh:

Disco /?

b. Tạo Proxy

Proxy là phần trung gian xử lý các yêu cầu của ứng dụng máy khách khi muốn gọi các phương thức của Web service, nó được đặt ngay tại máy khách và giữ vai trò đại diện cho Web service trên máy khách. Khi Proxy được tạo ra và được đăng ký cùng với ứng dụng trên máy khách, thì ứng dụng có thể gọi các phương thức hoặc hàm của Web service.

Có hai cách tạo Proxy:

Cách 1: Dùng Visual Studio.Net để tạo

Cách 2: muốn tạo Proxy chúng ta dùng tiện ích wsdl.exe chạy ở chế độ dòng lệnh, trình tiện ích này nhận tệp tin .wsdl làm đầu vào, tệp tin .wsdl có thể được tạo ra từ trước nhờ disco.exe hoặc được kết sinh "on-the-fly" từ bản thân tệp tin Web service. Hai câu lệnh sau cho ra cùng kết quả.

1. wsdl csStockTicker.wsdl

2. wsdl http://localhost/ProgAspNet/csStockTicker.asmx ?wsdl

Phần đầu ra của wsdl.exe là tập tin mã nguồn chứa lớp proxy, chúng ta có thể đem biên dịch thành tập tin *.dll.* Ngôn ngữ mặc định là C#. Muốn thay đổi ngôn ngữ chúng ta dùng khóa chuyển /laguage: (hoặc /l: cho gọn với giá trị là CS, VB, JS).

2.2. XÂY DỰNG KHO DỮ LIỆU

2.2.1 Khái niệm:

Dựa trên lịch sử tính toán của người dùng cuối, người ta đã đưa ra định nghĩa về những thành phần hợp nên kho dữ liệu.

Kho dữ liệu là nơi lưu trữ duy nhất, đầy đủ và nhất quán dữ liệu được lấy từ nhiều nguồn khác nhau và được người dùng cuối sử dụng trong phạm vi nghiệp vụ của mình

Hình 7: Kho dữ liệu

Đạt được sự đầy đủ và nhất quán của dữ liệu trong môi trường hệ thống thông tin ngày nay không đơn giản. Trong phạm vi nghiệp vụ, cần phải hiểu chiến lược nghiệp vụ và những dữ liệu cần thiết để hỗ trợ và kiểm soát quá trình thực hiện. Quá trình này được gọi lập mô hình công ty, đòi hỏi sự tham gia tích cực của của người dùng nghiệp vụ và diễn ra trong thời gian dài. Trong các dự án kho dữ liệu, lập mô hình công ty là bước khởi đầu quan trọng cho việc thiết kế. Xác định được dữ liệu cần thiết chỉ là bước khởi đầu. Dữ liệu có trong nhiều nguồn khác nhau, tồn tại dưới nhiều dạng khác nhau. Dữ liệu phải được kết hợp theo mô hình tổ chức. Để hiểu và sử dụng trong nghiệp vụ, phải chuyển dữ liệu thành thông tin. Các phân tích cần thiết để hoàn thành việc này được thực hiện ở bước mô hình hóa. Yêu cầu của người sử dụng là lập tập chỉ mục và chú giải (catalog) trong phạm vi nghiệp vụ và điều này trợ giúp quá trình tìm kiếm và sử dụng thông tin.

Cuối cùng người dùng cần một bộ công cụ để phân tích và sử dụng các thông tin có sẵn. Công cụ này cung cấp giao diện giữa người dùng với thông tin và đây là bước cuối cùng trong việc chuyển dữ liệu thô thành thông tin hữu ích.

Một số định hướng quản lý dựa trên thông tin:

- *Nguồn thông tin duy nhất*

Các thông tin thô được lấy từ nhiều nguồn khác nhau (có thể cả ở ngoài tổ chức) dưới nhiều hình thức khác nhau từ các dữ liệu có cấu trúc đến các dữ liệu phi cấu trúc. Các dữ liệu này trước khi đưa đến người dùng phải loại bỏ lỗi và và làm tương thích để đảm bảo tính tích hợp của dữ liệu. Các thông tin này sau khi được xử lý trở thành nguồn duy nhất cho hệ quản lý dựa trên thông tin.

- *Khả năng phân tán thông tin*

Hệ quản lý dựa trên thông tin không những được sử dụng tại trụ sở chính mà được phân tán theo tổ chức và theo khu vực. Người ta thường đòi hỏi nơi lưu trữ thông tin phải độc lập nhưng được kết nối với nhau về mặt logic để tăng cường khả năng lưu động, tính hiệu quả và sự bí mật. Không nhất thiết các nơi lưu trữ này phải gắn với nguồn dữ liệu thô.

- *Thông tin trong phạm vi nghiệp vụ*

Người dùng có thể hiểu rõ và sử dụng thông tin nếu nó được đặt trong phạm vi nghiệp vụ mà họ thực hiện. Định nghĩa dữ liệu do các chuyên gia cung cấp phải trở thành chuẩn cũng như các chỉ mục thông tin chứa những định nghĩa này, hướng về người dùng trở thành nguồn cho định nghĩa dữ liệu thậm chí đối với cả bộ phận hệ thống thông tin.

- *Tự động hóa truyền dữ liệu*

Khi dữ liệu chuyển thành thông tin được truyền giữa các tổ chức theo các cách thức ngày càng phức tạp thì rõ ràng cần một cơ chế tự động chuyển. Yêu cầu tự động không chỉ với quá trình chuyển tin thật sự mà còn với việc định nghĩa các phương thức chuyển và biến đổi dữ liệu. Đặc biệt là phải đảm bảo khả năng sử dụng cơ chế tự động hóa với các hệ thống thông tin phân tán.

- *Chất lượng và sở hữu thông tin*

Thông tin là tài sản quan trọng của công ty nên cần được quản lý và bảo vệ như bất kỳ tải sản nào khác. Quyền sở hữu của thông tin là điều tiên quyết để hiểu rõ các giá trị của thông tin.

Các dạng dữ liệu

Dữ liệu thường được định nghĩa là sự biểu diễn thông tin nghiệp vụ trên máy tính. Ở mức cao nhất, dữ liệu được phân chia theo nhiều cách. Ba điểm cần lưu ý khi xác định phạm vi của kho dữ liệu:

- *Ý nghĩa (meaning)*

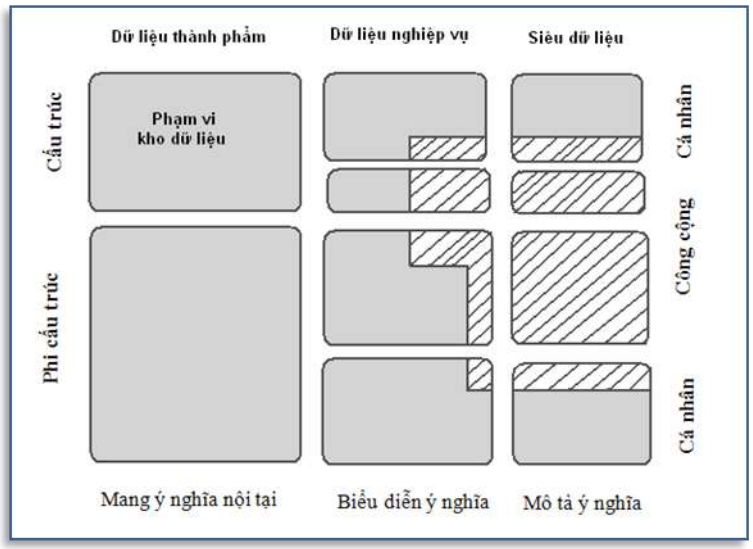
Dữ liệu có thể có nghĩa hay đại diện cho thứ gì có nghĩa. Sự phân biệt này là nguyên tắc cơ bản và cũng là điều khó hiểu nhất. Dữ liệu có ý nghĩa riêng nó và giá trị này nằm trong nội dung hơn là những gì chúng thể hiện. Do đó, chúng được gọi tên là dữ liệu thành phẩm bởi vì chúng được tạo ra và trao đổi giống như sản phẩm. Ví dụ phim, ảnh, sách dưới dạng số hóa,… Dạng cuối cùng là các siêu dữ liệu, chúng mô tả ý nghĩa của dữ liệu. Siêu dữ liệu chỉ định nghĩa hoặc mô tả dữ liệu nghiệp vụ hoặc dữ liệu thành phẩm.

- *Cấu trúc (structure)*

Dữ liệu có thể có cấu trúc ở mức độ cao (trường và các bản ghi) hoặc phi cấu trúc.

- *Phạm vi (scope)*

Dữ liệu có thể mang tính cá nhân nghĩa là chủ sở hữu có thể thay đổi tùy thích hoặc mang tính công cộng – khi đó nó được chia sẻ giữa một nhóm người và bất kỳ thay đổi nào cũng đòi hỏi giám sát chặt chẽ.



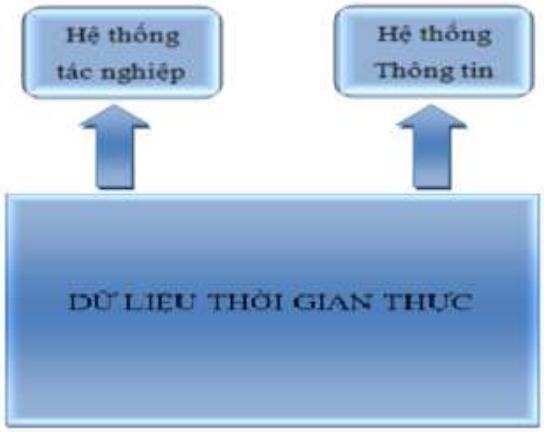
Hình 8: Các kiểu dữ liệu và phạm vi kho dữ liệu

2.2.2 Các kiến trúc dữ liệu nghiệp vụ

Ba mô hình kiến trúc dữ liệu mô tả dưới đây có một điểm chung: Chúng hoàn toàn dựa trên kinh nghiệm thực tế, do phát triển từ thực tế, chúng không rõ ràng như kiến trúc trên lý thuyết. Ba kiến trúc được đặt tên theo số lượng tầng dữ liệu của chúng. Các tầng dữ liệu này ở mức khái niệm hơn là mức vật lý. Do đó trong bất kỳ sự thực hiện nào, tầng được xác định bởi kiểu dữ liệu chứ không phải bởi vị trí vật lý của nó

a. Kiến trúc dữ liệu một tầng:

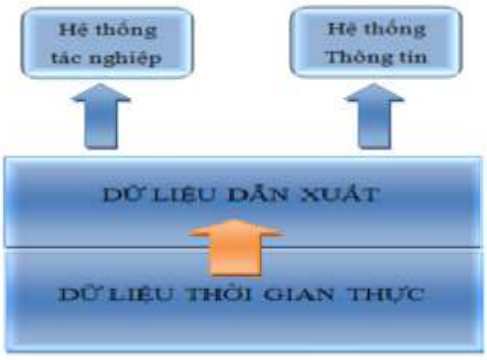
Quan điểm chủ đạo trong kiến trúc một tầng là phần tử dữ liệu bất kỳ được lưu trữ một lần duy nhất, không có sự phân biệt giữa các kiểu dữ liệu, tất cả dữ liệu được đối xử giống nhau.



Hình 9: Kiến trúc dữ liệu một tầng

b. Kiến trúc dữ liệu hai tầng:

Việc cải tiến kiến trúc một tầng dựa trên sự ghi nhận hai nhu cầu sử dụng dữ liệu khác nhau - tác nghiệp và thông tin. Dữ liệu được chi thanh hai tầng, tầng thấp được các ứng dụng tác nghiệp sử dụng trong chế độ đọc/ghi là dữ liệu thời gian thực. Tầng cao được các ứng dụng thông tin sử dụng là dữ liệu dẫn xuất. Dữ liệu dẫn xuất đơn giản có thể là bản sao trực tiếp của dữ liệu thời gian thực hoặc được dẫn xuất từ dữ liệu thời gian thực qua một vài phép tính toán.



Hình 10: Kiến trúc dữ liệu hai tầng

c. Kiến trúc dữ liệu ba tầng:

Điều chủ yếu trong kiến trúc dữ liệu ba tầng là sự ghi nhận quá trình chuyển dữ liệu thời gian thực thành dữ liệu dẫn xuất thực sự đòi hỏi hai bước chứ không phải một bước như trong kiến trúc hai tầng là:

- Tương thích dữ liệu từ nhiều cơ sở dữ liệu trong tầng thời gian thực.

- Dẫn xuất dữ liệu do người sử dụng yêu cầu từ dữ liệu vừa được làm tương thích.

Trong kiến trúc này tầng dưới cùng là dữ liệu thời gian thực, tầng trên cùng là dữ liệu dẫn xuất và tầng giữa là tầng dữ liệu tương thích.

Hình 11: Kiến trúc dữ liệu 3 tầng

Người dùng cuối hiếm khi truy cập trực tiếp vào tầng dữ liệu tương thích vì cấu trúc đã được lập và chuẩn hóa của tầng này nói chung không phù hợp với người dùng cuối. Những lý do nghiệp vụ này hạn chế việc sử dụng trực tiếp tầng dữ liệu tương thích cho các chức năng thông tin quản lý. Hầu hết những người dùng cuối có thể thỏa mãn yêu cầu nghiệp vụ của mình qua tầng dữ liệu dẫn xuất.

2.2.3 Tiêu chuẩn cho phân loại dữ liệu nghiệp vụ

Có bốn tiêu chuẩn để phân loại dữ liệu nghiệp vụ đó là khả năng sử dụng của dữ liệu trong nghiệp vụ, phạm vi dữ liệu, dữ liệu là loại đọc - ghi hay chỉ đọc, và thời gian hiện hành của dữ liệu.

2.2.3.1 Khả năng sử dụng dữ liệu trong nghiệp vụ:

Dữ liệu sử dụng trong nghiệp vụ để thực hiện hai mục tiêu sau:

- *Dữ liệu tác nghiệp*: Dùng để thực hiện nghiệp vụ liên quan đến các hoạt động hoặc quyết định trong tương lai gần.
- *Dữ liệu thông tin*: Dùng để quản lý nghiệp vụ trong thời gian dài hạn hơn.

Dữ liệu tác nghiệp là dữ liệu nghiệp vụ chủ yếu trong tổ chức và là nguồn cho dữ liệu thông tin. Cấu trúc của hai loại dữ liệu này đều phụ thuộc vào cách thức truy cập và nhu cầu sử dụng.

2.2.3.2 Phạm vi dữ liệu:

Dữ liệu có thể biểu thị một khoản mục hay giao dịch, nó có thể tổng hợp nhiều khoản mục hoặc nhiều giao dịch:

- Dữ liệu chi tiết hay dữ liệu nguyên tử: là yếu tố căn bản để thực hiện nghiệp vụ, được sử dụng trong một số tác vụ quản lý nghiệp vụ đơn giản, thường tập trung vào những đối tượng giao dịch như hợp đồng, khách hàng, sản phẩm.
- Dữ liệu tổng hợp: Được sử dụng trong quản lý nghiệp vụ, thể hiện một tầm nhìn bao quát về hoạt động của nghiệp vụ.

2.2.3.3 Dữ liệu đọc - ghi và dữ liệu chỉ đọc:

Dữ liệu đọc – ghi và dữ liệu chỉ đọc có những điểm khác biệt cơ bản.

- Dữ liệu đọc – ghi yêu cầu thiết kế quá trình cập nhật một cách cẩn trọng, bảo đảm các quy tắc nghiệp vụ toàn vẹn. Cấu trúc của nó được tối ưu cho việc ghi vào cở sở dữ liệu hoặc tệp dữ liệu.
- Dữ liệu chỉ đọc: Thường được thiết kế cho các truy vấn ngẫu nhiên và cung cấp một cơ sở ổn định cho việc đọc.

2.2.3.4 Thời gian hiện hành của dữ liệu:

Thời gian hiện hành của dữ liệu phản ánh vị trí của dữ liệu theo thời gian vận hành nghiệp vụ.

- Dữ liệu hiện hành thể hiện tình trạng hiện tại của nghiệp vụ, thường tồn tại trong thời gian ngắn và thay đổi theo thời gian, theo hoạt động của nghiệp vụ. Nó biểu diễn chính xác tình trạng hiện thời của nghiệp vụ.
- Dữ liệu tại một thời điểm là hình ảnh ổn định của của dữ liệu nghiệp vụ tại một thời điểm cụ thể, phản ảnh trạng thái của nghiệp vụ tại thời điểm đó.
- Dữ liệu định kỳ: Xác định hình ảnh chính xác nhất của nghiệp vụ khi thay đổi trong một khoảng thời gian.

2.2.4 Kỹ thuật thiết kế

Thiết kế kho dữ liệu đòi hỏi nhiều kỹ thuật, sự cần thiết của các kỹ thuật này xuất phát từ ba đặc điểm của kho dữ liệu:

- Phạm vi kho dữ liệu bao gồm toàn bộ tổ chức.
- Kho dữ liệu lưu trữ dữ liệu nghiệp vụ có tính chất lịch sử.
- Nguồn của tất cả dữ liệu trong kho là dữ liệu đã có, nó có khả năng phân tán, thay đổi về cấu trúc, nội dung và chất lượng biến đổi.

Các đặc điểm trên trực tiếp dẫn tới các kỹ thuật thiết kế và được mô tả dưới đây:

2.2.4.1 Lập mô hình tổ chức:

Mục đích của phần này là cung cấp hình ảnh chính xác các khía cạnh của thế giới thực trong phạm vi nào đấy. Nó giúp người sử dụng mô hình hiểu rõ các đối tượng được lập mô hình hoạt động như thế nào và khả năng dự đoán kết quả của bất kỳ hoạt động bên trong môi trường này cũng như ảnh hưởng của bất kỳ sự thay đổi nào tới nó.

2.2.4.2 Biểu diễn thời gian trong dữ liệu nghiệp vụ:

Do nghiệp vụ thay đổi theo thời gian nên dữ liệu nghiệp vụ cũng phải thể hiện sự thay đổi đó. Tauzovich (1991) đề nghị đưa vào các bản số *snapshot* và *lifetime* để biểu diễn cách nhìn tĩnh và cách nhìn theo thời gian. Lý do khác biệt giữa *Snapshot* và *lifetime* là sự kiện xảy ra ảnh hưởng đến mối quan hệ của các thực thể. Trong mô hình dữ liệu truyền thống không có vị trí rõ ràng nào thể hiện sự kiện này. Một phương pháp nữa là gắn nhãn thời gian (timestamp), trong thực tế phương pháp này được sử dụng rộng rãi. Thường dữ liệu thay đổi ở mức trường (field), nên có thể biểu diễn thời

gian ở mức đó hay bất kỳ ở mức nào cao hơn trong cấu trúc như bản ghi tùy theo yêu cầu chi tiết.

Cách nhìn dữ liệu thời gian dựa rên trạng thái, hay cơ sở dữ liệu trạng thái bao gồm chuỗi các bản ghi được gắn nhãn thời gian, mỗi bản ghi thể hiện trạng thái của thực thể tại một thời điểm. Thông thường trong các hệ tác nghiệp thời điểm được chọn là thời điểm ngay sau khi xảy ra sự kiện dẫn tới việc cập nhật cơ sở dữ liệu. Thí dụ: Quản lý tài khoản ngân hàng,…

2.2.4.3 Dữ liệu lịch sử:

Yêu cầu truy cập dữ liệu lịch sử là một trong những động cơ chính khi xây dựng kho dữ liệu. Cụ thể hơn, dữ liệu lịch sử đóng vai trò quan trọng trong kho dữ liệu và được dùng để phân tích xu hướng nghiệp vụ hay phân tích các mẫu, thường tập trung vào những phần dữ liệu đặc biệt của tổ chức. Dữ liệu lịch sử ngày càng quan trọng trong kho sữ liệu của tổ chức và nó cung cấp hồ sơ rõ ràng của nghiệp vụ.

Có hai vấn đề đòi hỏi việc duy trì hồ sơ lịch sử của nghiệp vụ:

- *Xem xét nghiệp vụ tại thời điểm bất kỳ:* Người sử dụng cuối cần xem xét nghiệp vụ tại các thời điểm khác nhau, một thời điểm có vai trò quan trọng đối với nghiệp vụ. Ví dụ: Thời điểm đóng thuế, một giao dịch trong ngân hàng,…
- *Phân tích xu hướng nghiệp vụ:* Phân tích xu hướng nghiệp vụ là quá trình nghiên cứu sự khác biệt giữa các chuỗi thời điểm xem xét. Ví dụ: Kết quả kinh doanh hàng tháng được lưu trữ để phân tích xu hướng kinh doanh của công ty.

2.2.4.4 Nhân bản dữ liệu:

Việc nhân bản dữ liệu có các đặc điểm chính sau:

- *Kiểm soát:* Nhân bản dữ liệu phải đảm bảo tính nhất quán của kết quả, bất luận dữ liệu được sao chép hay sử dụng vào mục đích nào.
- *Quản lý:* Cung cấp khả năng xây dựng và sử dụng lại các chức năng.
- *Mềm dẻo:* Cho phép kết hợp các chức năng và kỹ thuật khi cần thiết.
- *Bảo trì dễ dàng:* Cho phép đáp ứng nhanh chóng và có hiệu quả khi có thay đổi trong cấu trúc hoặc vị trí của tập dữ liệu nguồn hoặc đích.

- *Tích hợp siêu dữ liệu:* Cung cấp các kết nối với siêu dữ liệu của cả dữ liệu nguồn và dữ liệu đích, sử dụng hoặc tạo ra những siêu dữ liệu này khi cần thiết.
- *Hiệu quả:* Cung cấp cách thức hỗ trợ nguồn dữ liệu lớn tại nhiều mức đồng bộ.
- *Các nguồn đa dạng:* Hỗ trợ nhiều nguồn dữ liệu, đây là một đặc điểm của hệ thống thông tin ngày nay.
- *Dễ dàng sử dụng:* Người sử dụng với các kỹ năng kỹ thuật khác nhau từ người sử dụng cuối thông thường tới người quản trị cơ sở dữ liệu đều có thể sử dụng công cụ này.
- *Phạm vi nghiệp vụ:* Lưu giữ các mối quan hệ được các tiến trình nghiệp vụ tạo lên khi nhân bản dữ liệu.

Các đặc điểm nêu trên là những yếu tố cơ bản để xây dựng kho dữ liệu. Mỗi kỹ thuật xác định nhu cầu quan trọng trong toàn bộ thiết kế.

Mô hình dữ liệu tổ chức hỗ trợ cách nhìn chung nhất về dữ liệu để hiểu, thực hiện và quản lý nghiệp vụ cả hiện tại và tương lai.

Kỹ thuật biểu diễn dữ liệu theo chiều thời gian dẫn đến khả năng lưu trữ dữ liệu lịch sử, do đó có nhiều tiến trình nghiệp vụ trong tổ chức có thể sử dụng

CHƯƠNG 3 : THỬ NGHIỆM TÍCH HỢP DỮ LIỆU VỀ CÁC CẦU TRÊN QUỐC LỘ

3.1 Mô tả bài toán

Bộ giao thông vận tải là cơ quan đã áp dụng nhiều giải pháp công nghệ thông tin trong quản lý. Với đặc thù của ngành, các hệ thống được phân bố trên nhiều địa phương khác nhau, được xây dựng để phục vụ nhiều mục đích khác nhau. Vấn đề tích hợp dữ liệu, đồng bộ dữ liệu từ các nguồn dữ liệu có sẵn là nhu cầu không thể thiếu của bất kỳ hệ thống thông tin nào. Đối với Bộ giao thông vận tải thì điều này rất quan trọng và cần triển khai càng sớm càng tốt. Bài toán tích hợp dữ liệu về các cầu trên quốc lộ từ các khu quản lý đường bộ về Bộ giao thông vận tải được phát biểu tổng quát như sau: *"Từ các thông tin có sẵn trong hệ thống máy chủ của các khu quản lý đường bộ, cần xây dựng giải pháp tích hợp được các cơ sở dữ liệu đó về trung tâm tích hợp dữ liệu của Bộ giao thông vận tải để phục vụ công tác quản lý"*.

Hiện nay có rất nhiều giải pháp cho bài toán như trên, tuy nhiên lựa chọn phương pháp nào thì tùy thuộc vào hạ tầng kỹ thuật tin học hiện có, quy định về mặt pháp lý và định hướng của cơ quan đó. Trong đồ án này em xin trình bày hướng xây dựng các ứng dụng dùng Web service để trao đổi thông tin, tích hợp dữ liệu để phục vụ công tác quản lý. Để sử dụng Web service xây dựng các ứng dụng, chúng ta cần xem xét Web service dựa trên nền tảng công nghệ nào?

Web service trao đổi thông tin dựa trên giao thức HTTP và SOAP, điều này cho phép chúng ta trao đổi thông tin giữa các trung tâm với nhau thông qua hệ thống Internet rất thuận lợi. Chúng không phụ thuộc vào hạ tầng kỹ thuật, tin học của các đơn vị và không phụ thuộc vào bất kỳ hệ quản trị cơ sở dữ liệu và hệ điều hành nào.

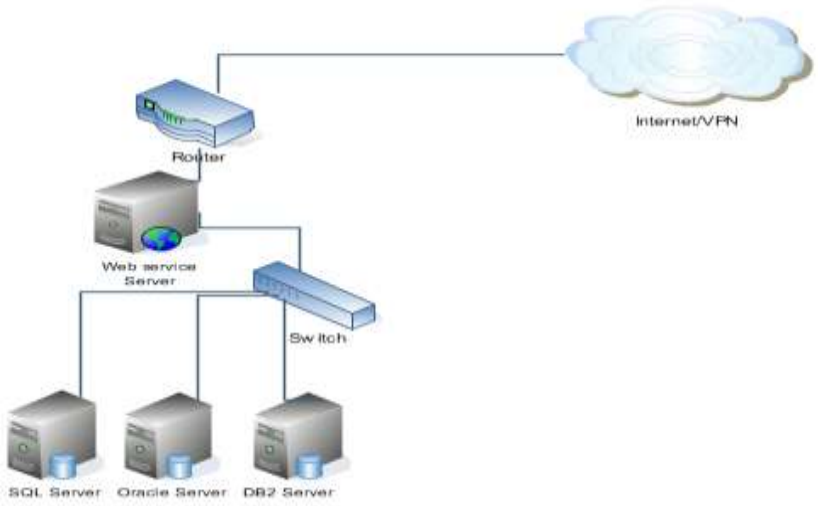
3.2 Truy cập cơ sở dữ liệu tại các khu quản lý đường bộ

Cơ sở dữ liệu Cầu tại các khu quản lý đường bộ được biểu diễn như sau:

Column Name	Data Type	Allow Nulls
[Bridge Number]	nvarchar(15)	☐
[Bridge Ref Code]	nvarchar(16)	☑
[Bridge Name]	nvarchar(50)	☑
[Structure Type]	nvarchar(50)	☑
[Road Classification]	nvarchar(2)	☐
[Road Number]	nvarchar(3)	☑
[Road Name 1]	nvarchar(4)	☑
[Road Name 2]	nvarchar(4)	☑
[Main Obstacle Crossed]	nvarchar(50)	☑
[Road Cross Reference]	nvarchar(30)	☑
[Number of Spans]	smallint	☑
[Maintenance Authority]	nvarchar(250)	☐
[Maintenance Unit]	nvarchar(250)	☑
Owner	nvarchar(30)	☑
[Railway chared]	bit	☐
[Tide influence]	bit	☐
[Inondation influence]	bit	☐
[Bridge Length]	float	☑
Province_Town	nvarchar(50)	☑
[Year exploire]	nvarchar(4)	☑
[Year Built]	nvarchar(4)	☑
[Design Standard]	nvarchar(30)	☑
[Load Standard]	nvarchar(30)	☑
[Load Assesment]	nvarchar(30)	☑
[Span form]	nvarchar(150)	☑
Huyen	nvarchar(30)	☑

Hình 12: CSDL cầu tại các khu quản lý đường bộ.

Tại mỗi khu quản lý đường bộ sẽ xây dựng một Web service truy cập vào cơ sở dữ liệu cục bộ để lấy thông tin. Việc trao đổi thông tin giữa các trung tâm với nhau thông qua HTTP và SOAP do vậy chỉ cần có một định danh cho Web service đó.



Hình 13: Mô hình truy cập dữ liệu tại các khu quản lý đường bộ.

Trên Web service xây dựng các phương thức truy cập và lấy thông tin từ máy chủ cơ sở dữ liệu. Đối với mỗi loại hệ quản trị cơ sở dữ liệu có cách thức riêng để truy cập và mọi hoạt động này được thực hiện trong phạm vi của tổ chức (trong mạng LAN), dữ liệu Web service nhận được là kết quả của các truy vấn vào cơ sở dữ liệu nội bộ sau đó trả về nơi yêu cầu thông qua giao thức HTTP, SOAP.

Phương thức truy cập và lấy dữ liệu các cầu từ các khu quản lý đường bộ:

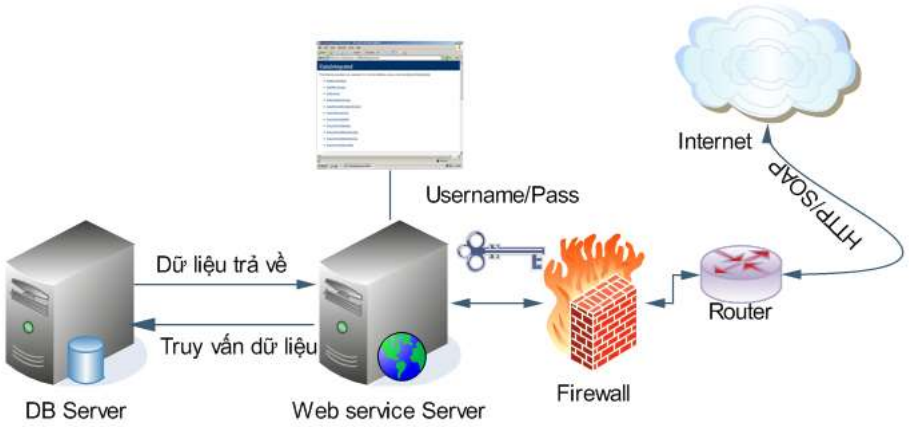
```
[WebMethod]
public DataSet getTable( string servername, string uid, string pass, string databasename)
{
    string source = "server=" + servername + ";uid=" + uid + ";pwd=" + pass + ";database=" + databasename;
    SqlConnection conn = new SqlConnection(source);
    conn.Open();
    SqlDataAdapter sda = new SqlDataAdapter("select * from dmuc_cau", conn);
    DataSet dsTable = new DataSet();
    sda.Fill(dsTable);
    return dsTable;
}
```

Tại Bộ giao thông vận tải, muốn khai thác các Web service này thì cần làm một số thao tác sau:

- Dùng tiện ích Disco, Wsdl để lấy được các thông tin của các Web service.

- Các ứng dụng sẽ tham chiếu thông tin một trong hai tệp này để kích hoạt các phương thức của Web service. Một lời gọi đến phương thức của Web service thực tế là một lời gọi đến Proxy, việc còn lại là do Proxy này thực hiện.

Khi có yêu cầu lấy dữ liệu, một thông điệp được gửi đến Web service của trung tâm cơ sở, sau khi kiểm tra tính hợp lệ của thông điệp một phương thức được chỉ định sẽ thực hiện việc truy nhập vào cơ sở dữ liệu cục bộ để lấy thông tin. Web service gửi thông tin đó dưới định dạng XML về Bộ. Tại đây dữ liệu có thể được xử lý thêm một lần nữa rồi lưu vào cơ sở dữ liệu của Bộ. Toàn bộ thông tin lấy được tích hợp thành cơ sở dữ liệu Cầu đặt tại Trung tâm tích hợp dữ liệu.



Hình 14: Mô hình trao đổi thông tin từ Web service với CSDL cục bộ

Vậy, ta thấy có một số ưu nhược điểm sau:

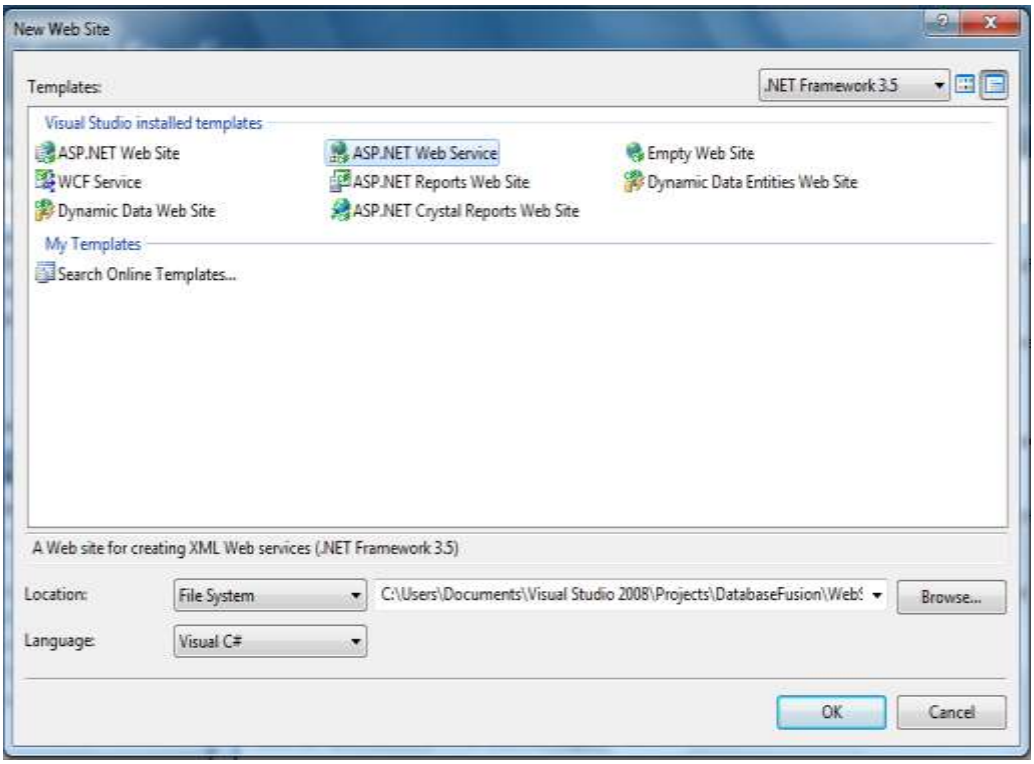
- Ưu điểm:
  - Bên yêu cầu lấy thông tin thông qua giao thức chung HTTP, SOAP nên không phụ thuộc vào hạ tầng hệ thống thông tin của các trung tâm cơ sở (bao gồm hệ quản trị cơ sở dữ liệu và hệ điều hành).
  - Bên yêu cầu chỉ nhận được thông tin do Web service trả về nên ta có thể tinh giản được lượng dữ liệu trước khi truyền trên Internet.
  - Sự an toàn của máy chủ cơ sở dữ liệu được đảm bảo do bên yêu cầu không phải kết nối trực tiếp vào máy chủ cơ sở dữ liệu, dữ liệu đi qua tầng trung gian là Web service và có thêm chế độ bảo mật của chính Web service.
  - Web service được xây dựng dễ dàng bằng nhiều ngôn ngữ lập trình như: PHP, ASP.NET, C#, VB.NET,…
- Hạn chế:
  - Phụ thuộc vào tốc độ đường truyền Internet.
  - Nếu thay đổi yêu cầu nghiệp vụ thì phải thay đổi các phương thức của Web service.

3.3 Xây dựng Web Service

Ở đây, ngôn ngữ lập trình được sử dụng là ASP.NET và C# trong bộ Visual Studio.NET của Microsoft.

Tạo Web service:

1. Khởi động VS.Net
2. Vào Menu File chọn New→Web Site, máy tính xuất hiện hộp thoại:



Hình 15: Hộp thoại tạo web service.

Sau khi hoàn thành việc xây dựng các phương thức để truy vấn dữ liệu. Bước tiếp theo ta cấu hình Web service để nó hoạt động. Việc cấu hình Web service tương tự như việc cấu hình một Web site bình thường.

3.4 Tiêu thụ Web Service

Muốn sử dụng các phương thức của Web service, ta phải lấy được toàn bộ thông tin về Web service này.

Có hai cách để đưa thông tin của Web service vào ứng dụng tiêu thụ (Consumer Application).

Cách 1:

Trong môi trường lập trình DotNet, MicroSoft cung cấp các tiện ích như Disco và Wsdl để lấy thông tin của Web service.

Disco *www.bridge.com/Service.asmx /out: c:*

Trong trường hợp Web service cần khai báo quyền truy nhập thì ta dùng thêm hai tham số */username: tên truy cập* và */password: mật khẩu*. tham số /out để chỉ nơi lưu trữ tệp kết quả được trả về, trong trường hợp trên có hai tệp kết quả được trả về là *Service.diso* và Service.*wsdl*. Tiếp theo ta dùng lệnh *Wsdl* để tạo ra tệp *Service.** phần mở rộng của tệp này tùy thuộc vào việc chỉ định tham số ngôn ngữ trong lệnh Wsdl.

Wsdl *http://www.bridge.com/Service.asmx /o:c:\dataintegrated /language:CS /protocol:SOAP12*

Kết quả của việc thực hiện câu lệnh trên là tạo ra được tệp *Service.cs* tên của tệp kết quả này trùng với tên của lớp được xây dựng bên trong Web service.

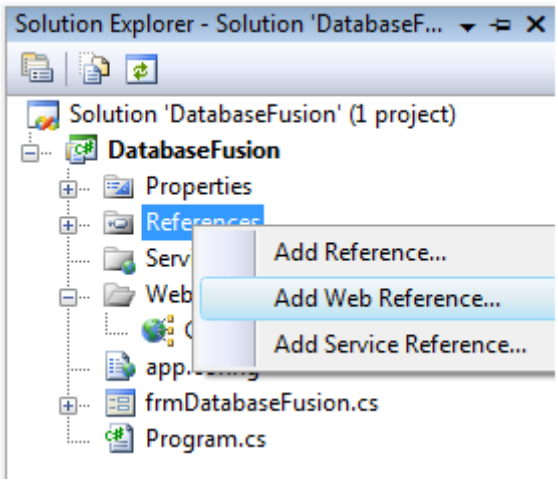
Tham số */language:* có thể nhận một số giá trị như JS, CS, VB,… Tham số */protocol:* chỉ giao thức truyền dữ liệu như SOAP, SOAP12, HttpGet, HttpPost.

Biên dịch tệp *Service.cs* thành Service.*dll*

Csc *c:\Service.cs*

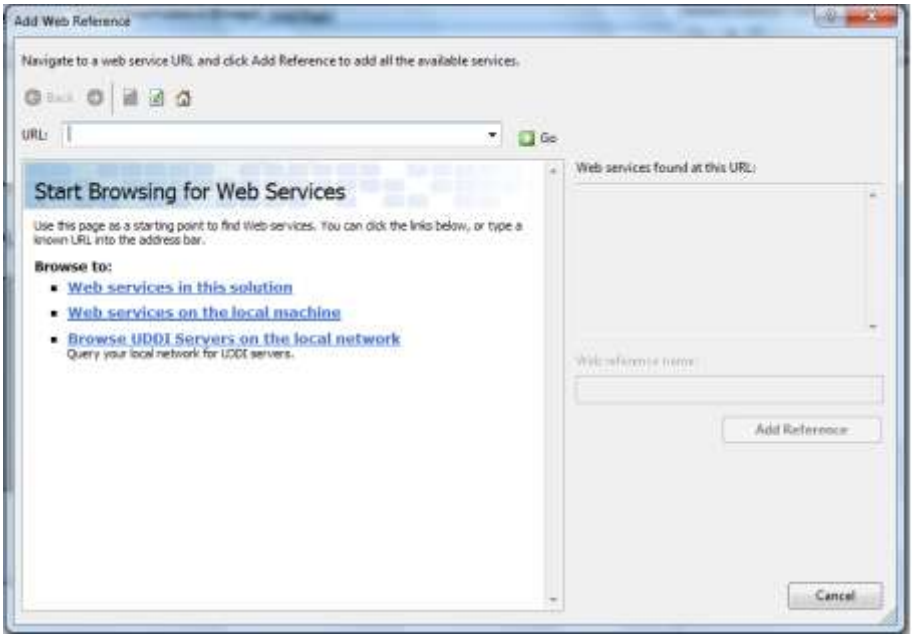
Cách 2:

Khi tạo một ứng dụng tiêu thụ Web service, ta có thể dùng ngay chức năng *Add web reference* của Visual Studio.net



Hình 16: Thêm tham chiếu Web vào chương trình ứng dụng

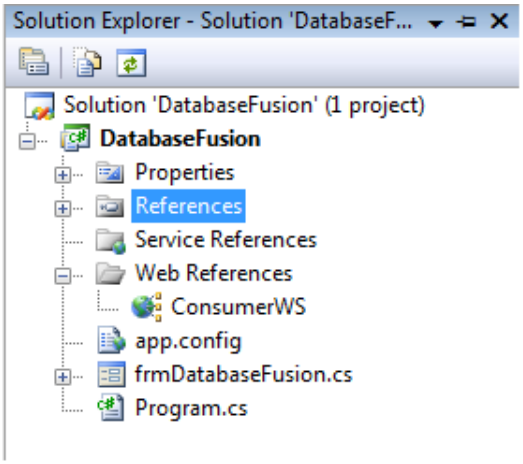
Máy tính xuất hiện hộp thoại:



Hình 17: Giao diện hộp thoại Add web reference

URL: Nhập địa chỉ của Web service

Web reference name: Tên tham chiếu của Web service trong chương trình tiêu thụ Chọn *Add Reference*. Trong chương trình này tên tham chiếu của Web service là: *ConsumerWS* và xuất hiện trong chương trình



Hình 18: Tham chiếu của Web service

Trong các lớp muốn gọi các phương thức của Web service phải khai báo dòng lệnh sau:

```
using DatabaseFusion.ConsumerWS;
```

DatabaseFusion: Đây là namespace của ứng dụng có dùng Web service.

ConsumerWS: Tên tham chiếu của Web service

Sau khi thực hiện phần trên chúng ta đã có toàn bộ thông tin về Web service cần sử dụng. Trong chương trình tiêu thụ tạo đối tượng thuộc Web service đó:

```
Service service = new Service();
```

Truy vấn dữ liệu thành công tại các khu quản lý đường bộ, ta lưu CSDL vừa thu được từ các khu quản lý đường bộ dưới dạng file XML:

```
DataSet ds = service.getTable( servername, uid, pass, databasename);
ds.WriteXml( filename+ ".xml");
```

Có được file XML, ta trích chọn những thông tin cơ bản, đặc trưng về cầu trên quốc lộ để tạo CSDL tích hợp có cấu trúc như sau:


```
DataSet dsRead = new DataSet();
dsRead.ReadXml( filename + ".xml");
DataView dw = new DataView(dsRead.Tables[0]);
DataTable table = dw.ToTable(true, "Bridge Number", "Bridge Ref Code",
                    "Bridge Name", "Structure Type", "Road Classification",
                    "Road Number", "Main Obstacle Crossed",
                    "Road Cross Reference", "Number of Spans",
                    "Maintenance Authority", "Maintenance Unit", "Owner",
                    "Bridge Length", "Province_Town", "Year Built",
                    "Load Standard", "Load Assesment", "Huyen",
                    "Tong chieu rong", "Chieu rong xe chay");
```

Tiếp đó là tạo CSLD tích hợp tại Bộ giao thông đặt tên là *CauQL.xml*:

```
table.WriteXml("CauQL.xml");
```

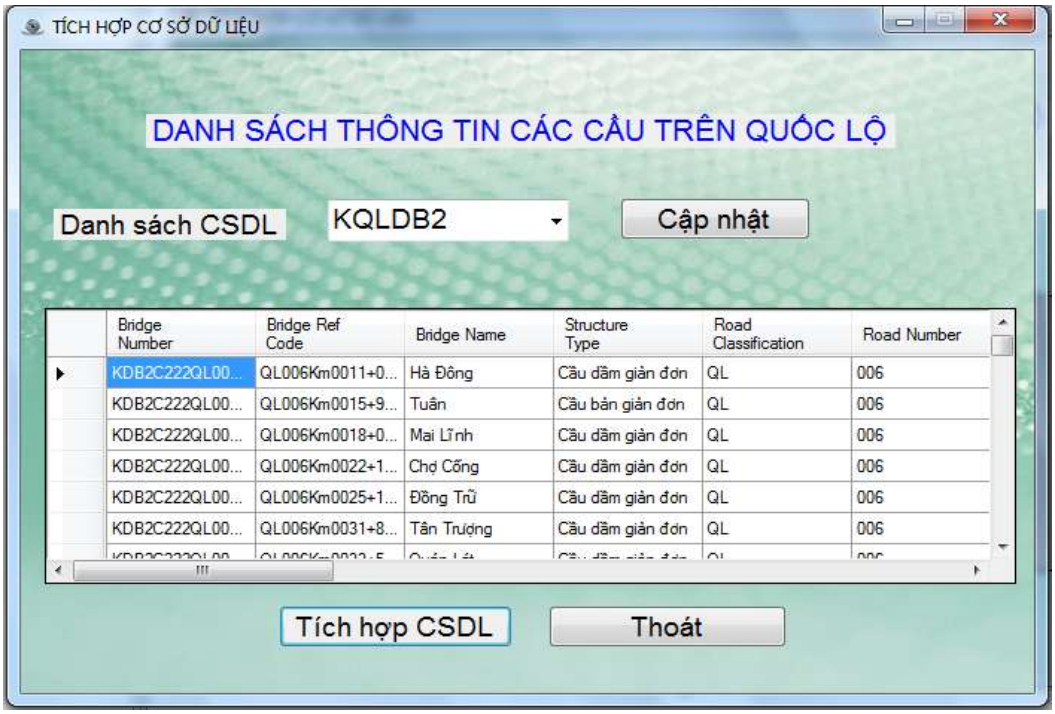
File được tạo ra có dạng như sau:

```
<?xml version="1.0" standalone="yes"?>
<DocumentElement>
  <Table>
    <Bridge_x0020_Number>KDB2C222QL00601</Bridge_x0020_Number>
    <Bridge_x0020_Ref_x0020_Code>QL006Km0011+070</Bridge_x0020_Ref_x0020_Code>
    <Bridge_x0020_Name>HÃ  Ã Ã´ng</Bridge_x0020_Name>
    <Structure_x0020_Type>Cá°§u dá°§m giá°£n Ã‘Æ¡n</Structure_x0020_Type>
    <Road_x0020_Classification>QL</Road_x0020_Classification>
    <Road_x0020_Number>006</Road_x0020_Number>
    <Main_x0020_Obstacle_x0020_Crossed>SÃ´ng</Main_x0020_Obstacle_x0020_Crossed>
    <Road_x0020_Cross_x0020_Reference>Nhuá»‡</Road_x0020_Cross_x0020_Reference>
    <Number_x0020_of_x0020_Spans>3</Number_x0020_of_x0020_Spans>
    <Maintenance_x0020_Authority>Cty QL&amp;amp;SCÃ B 222</Maintenance_x0020_Authority>
    <Maintenance_x0020_Unit>CÃ´ng ty cá°§u 14</Maintenance_x0020_Unit>
    <Owner>Khu QLÃ B 2</Owner>
    <Bridge_x0020_Length>68.7</Bridge_x0020_Length>
    <Province_Town>HÃ  TÃ©y</Province_Town>
    <Year_x0020_Built>-</Year_x0020_Built>
    <Load_x0020_Standard>H30-XB80</Load_x0020_Standard>
    <Load_x0020_Assesment>30T</Load_x0020_Assesment>
    <Huyen>TX. HÃ  Ã Ã´ng</Huyen>
    <Tong_x0020_chieu_x0020_rong>37.58</Tong_x0020_chieu_x0020_rong>
    <Chieu_x0020_rong_x0020_xe_x0020_chay>21.04</Chieu_x0020_rong_x0020_xe_x0020_chay>
  </Table>
  <Table>
    <Bridge_x0020_Number>KDB2C222QL00602</Bridge_x0020_Number>
    <Bridge_x0020_Ref_x0020_Code>QL006Km0015+952</Bridge_x0020_Ref_x0020_Code>
    <Bridge_x0020_Name>TuÃ©n</Bridge_x0020_Name>
    <Structure_x0020_Type>Cá°§u bá°£n giá°£n Ã‘Æ¡n</Structure_x0020_Type>
    <Road_x0020_Classification>QL</Road_x0020_Classification>
    <Road_x0020_Number>006</Road_x0020_Number>
    <Main_x0020_Obstacle_x0020_Crossed>Rá°¡ch(kÃªnh,mÆ°Æ¡ng)</Main_x0020_Obstacle_x0020_Crossed>
    <Road_x0020_Cross_x0020_Reference>MÆ°Æ¡ng thuá»· lá»£i</Road_x0020_Cross_x0020_Reference>
    <Number_x0020_of_x0020_Spans>2</Number_x0020_of_x0020_Spans>
```


Trong chương trình ta cho hiển thị thông tin CSLD lên đối tượng DataGridView:

this.dataGridView.DataSource = table

Kết quả sau khi thực hiện các bước trên:



Hình 19: Hiển thị kết quả nhận từ Web service

KẾT LUẬN

Trong đồ án em đã trình bày nghiên cứu và đề xuất giải pháp tích hợp các cơ sở dữ liệu phân tán trên môi trường Internet. Sau một thời gian nghiên cứu và tìm hiểu tài liệu, các kết quả chính đã đạt được là:

- Tìm hiểu những kiến thức về tích hợp dữ liệu.
- Các phương pháp tích hợp dữ liệu.
- Xây dựng giải pháp tích hợp dữ liệu.
- Trình bày một số công nghệ hiện có dùng trong việc tích hợp dữ liệu từ các hệ thống phân tán.
- Trình bày thử nghiệm tích hợp dữ liệu về các cầu trên quốc lộ dựa trên Web service.
- Giải pháp tích hợp dữ liệu dựa trên Web service hướng đi mới trong việc trao đổi thông tin trên Internet hiện nay.

Đồ án vẫn còn một số hạn chế là chưa xây dựng được một mô hình tích hợp tối ưu và hoàn thiện nhất, phần thực nghiệm mới chỉ xây dựng được chương trình nhỏ mang tính chất minh họa cho quá trình tích hợp dữ liệu từ các nguồn cơ sở dữ liệu phân tán, thu thập những thông tin đơn giản, trên thực tế các nguồn thông tin đa dạng và phức tạp hơn nhiều, cần thực hiện các giải pháp trích chọn thông tin phù hợp rồi mới tiến hành thu thập và tích hợp.

TÀI LIỆU THAM KHẢO

Tiếng Việt

- Dương Quang Thiện, *.NET Toàn Tập - Tập 5: Lập Trình Web Dùng ASP.NET Và C# - Lập Trình Visual C# Thế Nào?,* 2005, 738.
- Dương Quang Thiện, *.NET Toàn Tập - Tập 4: Lập Trình Căn Cứ Dữ Liệu dùng ADO.NET Và C# - Lập Trình Visual C# Thế Nào?,* 2005, 692.

Tiếng Anh

- Donald K. Burleson, Joseph Hudicka,William H. Inmon, Craig Mullins, Fabian Pascal, *The Data Warehouse eBusiness DBA Handbook*, BMC Software and DBAzine, 2003, 220.
- Dan Hurwitz, Jesse Liberty, *Programming ASP.NET,* Third Edition, O'Reilly, 2005, 956.
- Aaron Skonnard, Martin Gudgin, *Essential XML Quick Reference A Programmer's Reference to XML, XPath, XSLT, XML Schema, SOAP, and More,* Addison Wesley, 2007, 429.
- Michael A. Kittel, Geoffrey T. LeBlond, *ASP.NET Cookbook*, 2nd Edition, O'Reilly, 2005, 1014.
- Microsoft, *Developing XML Web Services Using Microsoft® ASP.NET*, 2002, 498.
- Microsoft, *Msdn*, 2005.
- Stavros Papastavrou, Panos Chrysanthis, George Samaras, Evaggelia Pitoura, *An Evaluation of the Java-based Approaches to Web Database Access*, 2005, 15p.