Thuật ngữ viết tắt

WSDL – Web Services Description Language

CSDL – Cơ sở dữ liệu

XML – Extensible Markup Language

SEI – service endpoint interface

# Phần mở đầu

# Phần nội dung

# Chương 1: Tổng quan về kiến trúc hướng dịch vụ

## Công nghệ Java Web Services

### Tổng quan về Web Services

Web service là một hệ thống phần mềm được thiết kế để hỗ trợ khả năng tương tác giữa các ứng dụng trên các máy tính khác nhau thông qua mạng Internet, giao diện chung và sự gắn kết của nó được mô tả bằng XML[[1]](#footnote-1). Web service là tài nguyên phần mềm có thể xác định bằng địa chỉ URL, thực hiện các chức năng và đưa ra các thông tin người dùng yêu cầu. Một Web Service được tạo nên bằng cách lấy các chức năng và đóng gói chúng sao cho các ứng dụng khác dễ dàng nhìn thấy và có thể truy cập đến những dịch vụ mà nó thực hiện, đồng thời có thể yêu cầu thông tin từ Web Service khác. Nó bao gồm các mô đun độc lập cho hoạt động của khách hàng và doanh nghiệp và bản thân nó được thực thi trên Server.

Web Services cho phép các ứng dụng khác nhau từ các nguồn khác nhau có thể giao tiếp với các ứng dụng khác mà không đòi hỏi nhiều thời gian coding, do tất cả các quá trình giao tiếp đều tuân theo định dạng XML, cho nên Web Service không bị phụ thuộc vào bất kì hệ điều hành hay ngôn ngữ lập trình nào. Web service cho phép client và server có thể tương tác được với nhau trên các nền tảng khác nhau mà không cần bất cứ thay đổi hay yêu cầu đặc biệt nào. Ví dụ, chương trình viết bằng ngôn ngữ Java cũng có thể trao đổi dữ liệu với các chương trình viết bằng Perl, các ứng dụng chạy trên nền Windows cũng có thể trao đổi dữ liệu với các ứng dụng chạy trên nền Linux. Công nghệ Web Service không yêu cầu phải sử dụng trình duyệt và ngôn ngữ HTML.

Web Services cũng có thể được nói một cách khác là các khối cơ bản được xây dựng để di chuyển trong hệ thống máy tính phân tán trên Internet. Các chuẩn mở và việc tập trung vào giao tiếp và làm việc cộng tác giữa con người và các ứng dụng đã tạo nên một môi trường nơi mà Web service đang trở thành nền tảng cho việc tích hợp ứng dụng. Các ứng dụng được xây dựng sử dụng các Web service các loại từ nhiều nguồn khác nhau làm việc cùng với nhau bất kể là chúng ở đâu hoặc chúng đã được triển khai như thế nào. Có thể có các định nghĩa khác nhau về Web service khi các công ty xây dựng chúng, nhưng hầu hết tất cả các định nghĩa đều có chung các điểm sau:

* Thứ nhất, Web service đưa ra chức năng hữu dụng cho người sử dụng Web thông qua một giao thức chuẩn Web. Trong hầu hết các trường hợp, giao thức được sử dụng đó là SOAP.
* Thứ hai, Web service đưa ra cách mô tả các giao diện của chúng một cách đủ chi tiết nhằm cho phép người sử dụng xây dựng một ứng dụng máy trạm để giao tiếp được với chúng. Mô tả này thường được cung cấp ở dạng một tài liệu XML gọi là một tài liệu về ngôn ngữ mô tả Web service – WSDL (Web service Description Language).
* Thứ ba, Web service được đăng ký sao cho các khách hàng tiềm năng là người sử dụng có thể tìm thấy chúng một cách dễ dàng. Điều này được thực hiện với UDDI (Universal Discovery Description and Integration).



Hình 1.1 Cơ chế hoạt động của Web Services

Web service như một dịch vụ phần mềm được trình bày trên Web thông qua giao thức SOAP, được mô tả bằng một tệp WSDL và được đăng ký trong UDDI. Các dịch vụ Web service là nguồn thông tin mà ta có thể dễ dàng kết hợp vào các ứng dụng. Dễ dàng nhận ra toàn bộ lớp ứng dụng có thể được xây dựng để phân tích và tích hợp thông tin ta quan tâm và trình bày nó theo nhiều cách khác nhau.

Việc trình bày các ứng dụng đang có như các dịch vụ Web service cho phép người sử dụng xây dựng các ứng dụng có các tính năng mạnh hơn thông qua việc sử dụng Web service như những block được xây sẵn. Ví dụ, người sử dụng có thể phát triển một ứng dụng mua bán để tự động lấy các thông tin về giá cả từ nhiều nhà cung cấp khác nhau, cho phép người dùng chọn một nhà cung cấp, chuyển đơn hàng và sau đó theo dõi việc chuyển hàng cho tới khi nhận được hàng. Ứng dụng của nhà cung cấp, khi trình bày các dịch vụ của họ trên Web, có thể quay ra sử dụng các dịch vụ Web service để kiểm tra tín dụng của khách hàng, lấy tiền từ tài khoản của khách hàng và thiết lập việc chuyển hàng với một công ty vận tải.

### Kiến trúc của Web Services

Kiến trúc của Web Services bao gồm các tầng như sau:



Hình 1.2 Kiến trúc của Web Services

* Tầng vận chuyển (Transport) với những công nghệ chuẩn là HTTP, SMTP và JMS. Có nhiệm vụ truyền thông điệp giữa các ứng dụng mạng.
* Tầng giao thức tương tác dịch vụ (Service Communication Protocol) với công nghệ chuẩn là SOAP. SOAP là giao thức nằm giữa tầng vận chuyển và tầng mô tả thông tin về dịch vụ, SOAP cho phép người dùng triệu gọi một dịch vụ từ xa thông qua một message XML.
* Tầng mô tả dịch vụ (Service Description) với công nghệ chuẩn là WSDL và XML. WSDL là một ngôn ngữ mô tả giao tiếp và thực thi dựa trên XML. Các Web Services sử dụng ngôn ngữ WSDL để truyền các tham số và các loại dữ liệu cho các thao tác, các chức năng mà các Web Services cung cấp.
* Tầng dịch vụ (Service): cung cấp các chức năng của Services.
* Tầng đăng ký dịch vụ (Service Registry) với công nghệ chuẩn là UDDI. UDDI dùng cho cả người dùng và SOAP server, nó cho phép đăng ký Services để người dùng có thể gọi thực hiện Services từ xa qua mạng, hay nói cách khác một Services cần phải được đăng ký để cho phép các khách hàng có thể gọi thực hiện.
* Bên cạnh đó để cho các Services có tính an toàn, toàn vẹn và bảo mật thông tin trong kiến trúc Web Services chúng ta có thêm các tầng Policy, Security, Transaction, Management giúp tăng cường tính bảo mật, an toàn và toàn vẹn thông tin khi sử dụng Services.

### Các thành phần của Web Services

#### XML

XML được viết tắt của cụm từ Extensible Markup Language – Ngôn ngữ đánh dấu dữ liệu. Là một chuẩn mở do W3C đưa ra cho cách thức mô tả dữ liệu, nó cho phép các máy tính truyền dữ liệu giữa các hệ thống không đồng nhất.

Về hình thức XML có cấu trúc giống với HTML nhưng không tuân theo một đặc tả quy ước như HTML. HTML định nghĩa các thành phần được hiển thị như thế nào, còn XML lại định nghĩa các thành phần chứa cái gì.

Web Services là sự kết hợp của nhiều thành phần khác nhau nên nó sử dụng các tính năng và đặc trưng của các thành phần đó để giao tiếp, XML là công cụ chính để giải quyết vấn đề này và là kiến trúc nền tảng cho việc xây dựng một Web Services.

Một tài liệu XML bao gồm hai thành phần:

Cấu trúc logic : định nghĩa các phần tử, các thuộc tính, các kiểu dữ liệu…

Cấu trúc vật lý : chứa dữ liệu và các phần tử chứa dữ liệu như text, image, media…

Ví dụ về XML:

<?xml version="1.0" encoding="UTF-8"?>

<Employer>

<name>Hồ Nhân</name>

<old>28</old>

</Employer>

#### WSDL – Web Services Description Language

WSDL định nghĩa một tài liệu XML mô tả giao diện của các Web Services. Tài liệu WSDL này được sử dụng cho bên yêu cầu dịch vụ (services requester). Bên yêu cầu dịch vụ sẽ sử dụng thông tin về giao diện định nghĩa trong lược đồ WSDL để triệu gọi (invoke) Web Services.

Một tài liệu WSDL mô tả một Web Service như một tập các đối tượng trừu tượng gọi là các “ports” và “endpoint”. Một tài liệu WSDL cũng định nghĩa bên trong nó các phương thức của web service. Các phương thức tương ứng với “operation” và dữ liệu trao đổi tương ứng với “message”. Một tập các phương thức liên quan được nhóm lại vào trong một “portType”. Một ràng buộc kết nối (binding) chỉ định một giao thức mạng và đặc tả định dạng dữ liệu cho một portType cụ thể. Kế đến một port được định nghĩa bằng cách kết hợp một địa chỉ mạng với một binding. Nếu một client có được một tài liệu WSDL và tìm thấy binding và địa chỉ cho mỗi port, nó có thể gọi các phương thức của dịch vụ theo đúng giao thức và định dạng dữ liệu đã đặc tả.

Phần tử gốc của tất cả các tài liệu WSDL luôn là phần tử <definitions>. Nó chứa bên trong sáu thành phần chia thành hai nhóm:

Thông tin trừu tượng (type, messages, portType)

Thông tin cụ thể (bindings, services)

Cấu trúc của một WSDL :



Hình 1.3 Cấu trúc WSDL

WSDL định nghĩa cách mô tả Web Services theo cú pháp tổng quát của XML, bao gồm các thông tin:

Tên dịch vụ

Giao thức và kiểu mã hóa sẽ được sử dụng khi gọi các hàm của Web Services

Loại thông tin: thao tác, tham số, những kiểu dữ liệu (có thể là giao diện của Web Services cộng với tên cho giao diện này)

Một WSDL hợp lệ gồm hai phần:

Service Inteface mô tả giao diện và giao thức kết nối

Service Implementation mô tả thông tin để truy xuất service



Hình 1.4 Cấu trúc WSDL

##### Các thành phần của WSDL

|  |  |  |
| --- | --- | --- |
| **Thành phần** | **Thông tin** | **Mô tả** |
| Service Interface | <type> | Định nghĩa các kiểu dữ liệu của thông điệp gửi |
| <message> | Mô tả thông điệp được gửi giữa client và server |
| <porttype> | WSDL mô tả cách gửi và nhận thông điệp |
| <binding> | Định nghĩa cách các Web Services kết hợp với nhau |
| Service Implementation | <service> | Nó sẽ thực hiện những gì đã được định nghĩa trong tập tin giao diện và cách gọi web services theo thủ tục và phương thức nào |
| <Port> | là một cổng đầu cuối, nó định nghĩa như một tập hợp của binding và một địa chỉ mạng |

Giải thích ý nghĩa các thành phần:

Type: định nghĩa kiểu dữ liệu được sử dụng cho Web Service để đảm bảo tính không phụ thuộc vào platform hoặc các phần tử XML được sử dụng cho các trao đổi thông báo, WSDL sử dụng cấu trúc của lược đồ XML để định nghĩa kiểu dữ liệu.

<wsdl:definitions .... >

< wsdl:types>

<xs:schema .... />\*

</ wsdl:types>

</ wsdl:definitions>

Message : định nghĩa các thành phần dữ liệu và các thông điệp mà nó được gọi tới. Mỗi thông điệp có thể bao gồm một hoặc nhiều phần, các thành phần này có thể so sánh với các câu lệnh của các lời gọi hàm trong các ngôn ngữ lập trình truyền thống. Những định nghĩa message đư­ợc sử dụng bởi phần tử thi hành dịch vụ. Nhiều thao tác có thể tham chiếu tới cùng định nghĩa message. Thao tác và những message đ­ược mô hình riêng rẽ để hỗ trợ tính linh hoạt và đơn giản hóa việc tái sử dụng lại. Chẳng hạn, hai thao tác với cùng tham số có thể chia sẻ một định nghĩa message.

< wsdl:definitions .... >

< wsdl:message name="nmtoken"> \*

< wsdl:part name="nmtoken" element="qname" /> \*

</ wsdl:message>

</ wsdl:definitions>

PortType : đây là thành phần quan trọng nhất trong một tài liệu WSDL. Nó được sử dụng để mô tả Web Service, các thao tác được thực thi và các lời gọi thông điệp. Thành phần PortType có thể được so sánh với các thư viện hàm (hoặc các module, các lớp ) trong các ngôn ngữ lập trình.

<wsdl:definitions .... >

<wsdl:portType name="nmtoken">

<wsdl:operation name="nmtoken" .... /> \*

</wsdl:portType>

</wsdl:definitions>

Trong thành phần < wsdl:porttype>, ta thường gặp 4 kiểu thao tác được WSDL định nghĩa dưới đây:

|  |  |
| --- | --- |
| **Kiểu thao tác** | **Mô tả** |
| One - way | Thao tác này thể hiện rằng nó chỉ nhận các lời gọi thông điệp nhưng không trả lại thông điệp đáp ứng |
| Request - response | Thao tác này bao gồm việc nhận các thông điệp yêu cầu và trả về các thông điệp đáp ứng |
| Solicit - response | Thao tác này sẽ gửi đi các yêu cầu và đợi các đáp ứng |
| Notification | Thao tác này sẽ gửi đi các yêu cầu nhưng không đợi để nhận các đáp ứng |



Hình 1.5 Bốn kiểu thao tác mà một cổng có thể hỗ trợ

Mỗi kiểu thao tác có cú pháp biến đổi tùy theo: thứ tự của các message nhập, xuất và lỗi.

Ví dụ :

<wsdl:definitions .... >

<wsdl:portType .... > \*

<wsdl:operation name="nmtoken" parameterOrder= "nmtokens">

<wsdl:input name="nmtoken"? message="qname"/>

<wsdl:output name="nmtoken"? message="qname"/>

<wsdl:fault name="nmtoken" message="qname"/>\*

</wsdl:operation>

</wsdl:portType >

</wsdl:definitions>

*Binding:* Định nghĩa cách thức truy cập Web Service thông qua các giao thức bên dưới. Mỗi phần tử Binding sẽ mô tả cách thức liên kết một PortType vào một Protocol nhất định. Web Service hỗ trợ bao nhiêu Protocol thì phải xây dựng bấy nhiêu phần tử Binding

<wsdl:binding name="…" type="ns:…">

<soap:binding transport="… " style="document" />

<wsdl:operation name="…">

<soap:operation soapAction="urn:…" style="document" />

<wsdl:input>

<soap:body use="literal" />

</wsdl:input>

<wsdl:output>

<soap:body use="literal" />

</wsdl:output>

</wsdl:operation>

</wsdl:binding>

Service (dịch vụ) : Nó sẽ thực hiện những gì đã được định nghĩa trong tập tin giao diện và cách gọi Web Service theo thủ tục và phương thức nào:

<wsdl:definitions .... >

<wsdl:service name="nmtoken"> \*

<wsdl:port .... />\*

</wsdl:service>

</wsdl:definitions>

Port (cổng dịch vụ) : Là một cổng đầu cuối, nó định nghiã như­ một tập hợp của binding và một địa chỉ mạng.

<wsdl:definitions .... >

<wsdl:service .... > \*

<wsdl:port name="nmtoken" binding="qname"> \*

</wsdl:port>

</wsdl:service>

</wsdl:definitions>

Ở đây chúng ta thấy rằng thuộc tính kết hợp tên là qname. Nó tham chiếu tới một mối kết hợp. Một cổng chứa đựng chính xác một địa chỉ mạng. Bất kỳ cổng nào trong phần thi hành phải tương ứng chính xác với một tham chiếu trong phần giao diện.

BỔ SUNG VÍ DỤ?????????????????????????

#### UDDI – Universal Description, Discovery, and Integration

##### Tổng quan

Về cơ bản Universal Description, Discovery, and Intergration (UDDI) là một nơi mà các tổ chức đăng ký và tìm kiếm các Web Service. Nó đóng vai trò như service broker cho phép người sử dụng dịch vụ tìm đúng nhà cung cấp dịch vụ cần tìm. UDDI hỗ trợ chức năng:

Thực hiện tìm kiếm, định vị những doanh nghiệp cung cấp dịch vụ hay sản phẩm theo phần loại theo vùng địa lý

Thông tin về một nhà cung cấp dịch vụ bao gồm địa chỉ, thông tin liên lạc và các định danh

Thông tin kỹ thuật (Technical information) về Web service mà doanh nghiệp cung cấp (ví dụ như cách sử dụng dịch vụ được cung cấp)

Để sử dụng đến các dịch vụ của UDDI, bản thân UDDI cung cấp một tập hàm API dưới dạng SOAP Web Service. Tập API được chia làm hai phần: Inquiry API dùng truy vấn và Publisher’s API dùng đăng ký. Phần API dùng để truy vấn bao gồm hai phần con : một phần dùng để tạo ra các chương trình cho phép tìm kiếm và duyệt thông tin trên một UDDI registry, phần còn lại dùng để xử lý lỗi triệu gọi.

Thành phần xử lý chính là bộ đăng ký UDDI, đó là một file XML dùng để mô tả một thực thể kinh doanh (business entity) kèm theo các Web service đi cùng. Sử dụng các dịch vụ của UDDI, các doanh nghiệp đăng ký thông tin về những Web service mà họ định cung cấp. Thông tin này đuợc thêm vào UDDI registry thông qua Web site hoặc sử dụng các công cụ lập trình sử dụng các dịch vụ theo đúng đặc tả UDDI programmer’s API.

##### Các thành phần của UDDI

UDDI gồm 2 thành phần chính:

Phần đăng ký của tất cả các Web Service’s metadata, bao gồm cả việc trỏ đến tài liệu WSDL mô tả dịch vụ

Phần thiết lập WSDL Port type định nghĩa cho các thao tác và tìm kiếm thông tin đăng ký.

UDDI xây dựng dựa trên các giao thức chuẩn Internet được công bố bởi W3C và IETF như XML, HTTP, và DNS. UDDI sử dụng WSDL để mô tả giao diện của Web Service. Thêm nữa tính năng độc lập với nền tảng ngôn ngữ lập trình đã được điều hợp cùng với giao thức SOAP.

##### Mô hình dữ liệu của UDDI

UDDI bao gồm lược đồ XML, mô tả bốn kiểu cấu trúc dữ liệu dưới đây:

businessEntity

businessService

bindingTemplate

tModel

publisherAssertion



Hình 1.6 Mô hình dữ liệu của UDDI

###### businessEntity

Một businessEntity chứa các thông tin về công ty, bao gồm danh sách liên lạc, thông tin, phân biệt các tổ chức thương mại, và danh sách các nhà cung cấp dịch vụ web. Sau khi đăng kí thì mỗi công ty, doanh nghiệp… sẽ nhận được một định danh duy nhất.

###### businessService

Liên kết với mỗi business entity là một danh sách các business service cung cấp bởi business entity đó. Mỗi thành phần chứa thông tin mô tả về dịch vụ, về thông tin phân loại của dịch vụ và danh sách các binding template liên quan đến thông tin kỹ thuật của dịch vụ. Mỗi business service cần có ít nhất một binding template.

###### bindingTemplate

Gắn với mỗi business service là một danh sách các binding template cung cấp thông tin về địa điểm có thể tìm thấy Web Service và làm cách nào để sử dụng nó. Một cấu trúc binding template mô tả thông tin interface của Web Service và các địa chỉ URL. Mỗi bindingTemplate được định danh duy nhất thông qua số phát sinh tự động UUID lưu trong bindingKey.

###### tModel

tModel là lõi trong cùng của kiểu dữ liệu, nhưng rất khó có khả năng để có thể nắm bắt được hết. tModel là chuẩn cho mô hình kĩ thuật.

tModel là phương pháp để mô tả một vài quy trình thương mại, dịch vụ và các cấu trúc mẫu lưu trữ trong UDDI registry. Bất kì một khái niệm trừu tượng nào đều có thể được đăng ký trong UDDI như là một tModel. Ví dụ: chúng ta có thể định nghĩa ra một kiểu cổng (port type) WSDL mới, và đồng nghĩa với đó ta có thể định nghĩa ra một tModel mới mà trình bày kiểu cổng đó trong UDDI. Sau đó, ta có thể chỉ định ra dịch vụ thương mại mà thực thi kiểu cổng đó bằng việc kết hợp với tModel với một business service’s binding template.

Mục đích của tModel là dùng để liên kết đến metadata bên ngoài UDDI. Thành phần quan trọng nhất của tModel là một URL trỏ đến một tài liệu mô tả thông tin metadata. Tài liệu này có thể là tài liệu bất kì HTML, Word, .. tùy ý mô tả một đặc tả kỹ thuật nào đó, ví dụ như giao thức mạng, dạng thức trao đổi hoặc luật tuần tự mà thông thường nhất là file mô tả thông tin service WSDL. Có hai thuộc tính cơ bản bên trong một tModel : tModelKey đóng vai trò định danh duy nhất giữa các tModel với nhau và name dùng cung cấp một tên với đầy đủ ngữ nghĩa cho tModel.

###### publisherAssertion

Đây là một cấu trúc dữ liệu quan hệ mà nó đặt sự kết hợp giữa hai hoặc nhiều cấu trúc dữ liệu businessEntity theo một kiểu quan hệ cụ thể, chẳng hạn như một công ty con hoặc một phòng ban.

Cấu trúc dữ liệu pubisherAssertion bao gồm ba thành phần chính: fromkey (BusinessKey đầu tiên), toKey (bussinesskey thứ hai) và keyedReference. KeyReference thiết kế ra kiểu mỗi quan hệ kết hợp trong cặp thuật ngữ keyName, keyValue trong tModel. Tham chiếu duy nhất bởi tModelkey.

#### SOAP – Simple Object Access Protocol

##### Tổng quan

Làm thế nào chúng ta truy xuất dịch vụ khi tìm thấy ? Câu trả lời là Các Dịch vụ Web có thể truy xuất bằng một giao thức là Simple Object Access Protocol – SOAP. Nói cách khác chúng ta có thể truy xuất đến UDDI registry bằng các lệnh gọi hoàn toàn theo kiểu SOAP.

SOAP là một giao thức dựa trên XML để trao đổi thông tin giữa các máy tính. Mặc dù SOAP có thể sử dụng một loạt các thông điệp hệ thống và có thể được gửi qua giao thức của tầng vận chuyển. Tập trung ban đầu là các thủ tục triệu gọi từ xa được vận chuyển thông qua HTTP. SOAP do đó cho phép các ứng dụng của khách hàng kết nối dễ dàng đến các dịch vụ từ xa và gọi từ xa các phương pháp này.

Hay như định nghĩa của tổ chức W3C thì “SOAP là một giao thức hổ trợ việc trao đổi thông tin trong một môi trường tản quyền và phân tán”.

Khái niệm cơ bản nhất của mô hình SOAP là việc sử dụng các tài liệu XML như những thông điệp trao đổi. Điều này có nhiều ưu điểm hơn các giao thức truyền dữ liệu khác. Các thông điệp XML có thể được tổng hợp và đọc với một bộ soạn thảo text đơn giản, ta có thể làm việc với XML trên hầu hết mọi nền tảng.

##### Đặc trưng

SOAP có những đặc trư­ng sau:

SOAP đư­ợc thiết kế đơn giản và dễ mở rộng

Tất cả các message SOAP đều đư­ợc mã hóa sử dụng XML

SOAP sử dùng giao thức truyền dữ liệu riêng

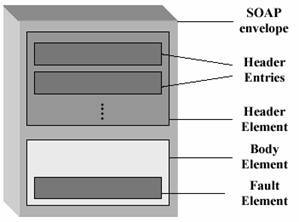
Không có garbage collection phân tán, và cũng không có cơ chế tham chiếu. Vì thế SOAP client không giữ bất kỳ một tham chiếu đầy đủ nào về các đối tượng ở xa

SOAP không bị ràng buộc bởi bất kỳ ngôn ngữ lập trình nào hoắc công nghệ nào

Vì những đặc trư­ng này, nó không quan tâm đến công nghệ gì đư­ợc sử dụng để thực hiện miễn là người dùng sử dụng các message theo định dạng XML. T­ương tự, dịch vụ có thể đư­ợc thực hiện trong bất kỳ ngôn ngữ nào, miễn là nó có thể xử lý được những message theo định dạng XML.

##### Cấu trúc một message theo dạng SOAP

Cấu trúc một message theo dạng SOAP được mô tả như hình dưới đây



Hình 1.7 Cấu trúc message SOAP

Message theo dạng SOAP là một văn bản XML bình thường bao gồm các phần tử sau:

*Phần tử gốc - envelop:* phần từ bao trùm nội dung message, khai báo văn bản XML như là một thông điệp SOAP.

Ví dụ:

<SOAP-ENV:Envelope

xmlns:SOAP-ENV=”<http://schemas.xmlsoap.org/soap/envelope/>”>

Với SOAP 1.1 namespace URI là <http://schemas.xmlsoap.org/soap/envelope/>, đối với SOAP 1.2 namespace URI là <http://www.w3.org/2001/09/soap-envelope>.

*Phần tử đầu trang – header:* chứa các thông tin tiêu đề cho trang, phần tử này không bắt buộc khai báo trong văn bản. Những đầu mục còn có thể mang những dữ liệu chứng thực, những chữ ký số hóa, và thông tin mã hóa, hoặc những cài đặt cho giao tác.

Ví dụ:

<SOAP-ENV:Header>

<ns1:PaymentAccount

xmlns:ns1="urn:ecerami"SOAP-ENV:

mustUnderstand="true">orsenigo473

</ns1:PaymentAccount >

</SOAP-ENV:Header>

*Phần tử khai báo nội dung chính trong thông điệp – body:*chứa các thông tin yêu cầu và phản hồi. Thành phần khai báo nội dung là thành phần bắt buộc đối với tất cả các message dạng SOAP.

Ví dụ:

<env:Body>

<m:GetLastTradePrice Env:encodingStyle="http://www.w3.org/2001/09/soap-encoding"

xmlns:m="http://example.org/2001/06/quotes">

<symbol>DIS</symbol>

</m:GetLastTradePrice>

</env:Body>

*Phần tử phát sinh lỗi – Fault:* cung cấp thông tin lỗi xảy ra trong quá trình xử lý thông điệp.

Ví dụ:

<?xml version='1.0' encoding='UTF-8'?>

<SOAP-ENV:Envelope

xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/1999/XMLSchema">

<SOAP-ENV:Body>

<SOAP-ENV:Fault>

<faultcode xsi:type="xsd:string">

SOAP- ENV:Client</faultcode>

<faultstring xsi:type="xsd:string">

Failed to locate method (ValidateCreditCard) in class (examplesCreditCard) at /usr/local/ActivePerl-5.6/lib/ site\_perl/5.6.0/SOAP/Lite.pm line 1555.

</faultstring>

</SOAP-ENV:Fault>

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>

##### Những kiểu truyền thông

SOAP hỗ trợ hai kiểu truyền thông khác nhau:

Remote Procedure Call (RPC): cho phép gọi hàm hoặc thủ tục qua mạng. Kiểu này đư­ợc khai thác bởi nhiều Các Dịch vụ Web và có nhiều trợ giúp.

Document: đư­ợc biết nh­ư kiểu hư­ớng message, kiểu này cung cấp một lớp thấp của sự trừu tư­ợng hóa, và yêu cầu người lập trình nhiều hơn khi làm việc.

Các định dạng message, tham số, và lời gọi đến các API thì tương ứng trong RPC và document là khác nhau. Nên việc quyết định chọn cái nào tùy thuộc vào thời gian xây dựng và sự phù hợp của dịch vụ cần xây dựng.

##### Mô hình dữ liệu

Mục đích của mô hình dữ liệu SOAP là cung cấp những một sự trừu tượng hóa độc lập ngôn ngữ cho kiểu ngôn ngữ lập trình chung. Nó gồm có:

Những kiểu XSD đơn giản như những kiểu dữ liệu cơ bản trong đa số các ngôn ngữ lập trình như­ int, string, date, …

Những kiểu phức tạp, có hai loại là struct và array.

Tất cả các phần tử và những định danh có trong mô hình dữ liệu SOAP thì được định nghĩa bằng namespace SOAP-ENC.

### Các thư viện dùng để tạo Web Services trong Java

Trong Java có hai thư viện phổ biến nhất thường được sử dụng để xây dựng một Web Service, đó là JAX-WS và Apache Axis2.

#### JAX-WS

##### Tổng quan

JAX-WS (Java API for XML Web Services) là một bộ thư viện được phát triển dưới sự quản lý của Sun Microsystem. Trang chủ của dự án <https://jax-ws.java.net/> là nơi cung cấp các tài liệu hướng dẫn, các bản cập nhật thường xuyên của bộ thư viện này.

JAX-WS giúp tạo ra các dịch vụ web sủ dụng giao thức XML. Với việc được tích hợp sẵn trong bộ Java EE SDK và được đóng gói kèm theo trong các bộ cộng cụ tích hợp dành cho Java EE. nên JAX-WS là bộ thư viện rất phổ biến để tạo ra những dịch vụ web.

JAX-WS làm tăng tốc độ phát triển dịch vụ Web thông qua việc cung cấp một thư viện các chú giải để chuyển các lớp đối tượng Java cũ đơn giản (plain old Java object) (POJO) thành các Web Service. Nó cũng quy định một sự ánh xạ chi tiết từ một dịch vụ được định nghĩa trong WSDL đến các lớp Java thực hiện dịch vụ đó.

Phát triển một Web Service bằng JAX-WS với một trong hai cách tiếp cận:

Theo mục đích: bắt đầu bằng một WSDL và tạo một lớp Java để thực hiện dịch vụ. Tuy nhiên các tiếp cận này đòi hỏi một sự hiểu biết tốt về WSDL và XSD (định nghĩa lược đồ XML) để định nghĩa các định dạng thông điệp

Theo mã: bắt đầu bằng một lớp Java và sử dụng công cụ để tạo cả tệp WSDL và giao diện Java

##### Các bước cơ bản để tạo Web Service với JAX-WS

Thông thường một web service cần có một service endpoint interface (SEI) là một interface dùng để khai báo các phương thức mà client có thể gọi đến. Tuy nhiên khi xây dựng web service với JAX-WS các SEI cần thiết phải có. Sau đó các interface SEI sẽ được cụ thể hóa bằng phiên bản cài đặt của web service (implementation class). Các bước cơ bản gồm có:

Cài đặt các phương thức cho web service

Biên dịch các class của web service thành một tập tin WAR

Cài đặt và triển khai tập tin WAR lên server. Các class thành phần dùng cho việc giao tiếp với client sẽ được tạo ra bởi server trong quá trình cài đặt (deploy)

##### Một số lưu ý khi cài đặt một SEI

Lớp cài đặt của web service phải được đánh dấu là các class javax.jws.WebService hoặc javax.jws.WebServiceProvider (dùng annotation @)

Lớp cài đặt phải được chỉ rõ là cài đặt cho interface SEI nào thông qua thuộc tính endpointInterface của @WebService, nhưng việc này không bắt buộc. Nếu không có endpointInterface nào được chỉ ra, thì class cài đặt đó sẽ được ngầm định là có một SEI

Các phương thức của lớp cài đặt phải là các phương thức public và không được là các phương thức tĩnh static hay final

Các phương thức của web service cho phép client gọi cần phải có khai báo annotation là @javax.jws.WebMethod

Các tham số và các kiểu trả về của các phương thức web service cung cấp cho client phải là các kiểu nằm trong JAX-B-compatible parameters and return types. Xem [Default Data Type Bindings](http://java.sun.com/webservices/docs/2.0/tutorial/doc/JAXBWorks4.html#wp82947)

Lớp cài đặt không được là một lớp final hay abstract

Lớp cài đặt phải có một constructor mặc định

Lớp cài đặt không được phép định nghĩa phương thức finalize

Lớp cài đặt cần sử dụng các annotation là javax.annotation. PostConstruct hoặc javax.annotation.PreDestroy trong các hàm phục vụ việc quản lý vòng đời của các sự kiện trong Web service:một phương thức @PostConstruct được gọi bởi web service container trước khi class cài đặt của web service phản hồi lại client; các phương thức được khai báo với @PreDestroy web service container trước khi web service được gỡ bỏ

#### Apache Axis2

##### Tổng quan

Đây là bộ thư viện mã nguồn mở được phát triển bởi Apache Software Foundation. Trang chủ của dự án [http://axis.apache.org/axis2/java/core/](http://axis.apache.org/axis2/java/core/%20) là nơi cung cấp các tài liệu hướng dẫn, các bản cập nhật thường xuyên của bộ thư viện này. Dự án Axis2 là một sự cài đặt dựa trên ngôn ngữ lập trình Java cho cả phía Client và Server trong mô hình Web Services.

Apache Axis2 cung cấp một mô hình đối tượng hoàn chỉnh và một kiến trúc môđun hóa cho phép dễ dàng việc thêm vào các chức năng và hỗ trợ những đề xuất và các đặc tả mới liên quan đến Web services.

##### Các đặc điểm của Apache Axis2

Axis 2 cung cấp một khung ứng dụng để xử lý các thông điệp SOAP. Khung ứng dụng này có thể được mở rộng trong các xử lý thông điệp SOAP hoặc trong các hoạt động cơ sở. Hơn nữa, Axis2 cho phép mô hình hóa các dạng trao đổi thông điệp khác nhau (MEPs) để được sử dụng ở trong khung xử lý (processing framework)

* Axis2 có khả năng triển khai một dịch vụ Web với sự hỗ trợ của WSDL hoặc không cần WSDL
* Axis2 cung cấp giao diện thao tác người dùng (Client API) sử dụng để gọi các dịch vụ Web. Giao diện thao tác người dùng có thể hỗ trợ cho các hai mô hình đồng bộ và bất đồng bộ
* Axis2 cho phép thay đổi thông tin cấu hình hoặc ở các thành phần của nó trong triển khai dịch vụ Web
* Axis2 cung cấp khả năng nhận và gửi các thông điệp SOAP với nhiều giao thức khác nhau

##### Mô hình xử lý thông điệp SOAP

Nhận và gửi thông điệp SOAP có thể coi là hai trong số những công việc chủ yếu của máy xử lý SOAP. Trong kiến trúc Axis2 cung cấp hai đường ống để thực hiện hai hành động cơ bản là nhận và gửi thông điệp SOAP. Hai đường ống này có tên là InPipe và OutPipe. Các mẫu tin nhắn trao đổi phức tạp ký hiệu MEPs, nó được xây dựng bởi kết nối hai loại ống InPipe và OutPipe. Ngoài ra còn hai đường ống khác có mục đích hỗ trợ cho các thông điệp gửi đến và gửi đi bị lỗi



Hình 1.8 Mô hình xử lý thông điệp SOAP

Khả năng mở rộng của mô hình xử lý SOAP được cung cấp thông qua cơ chế xử lý (handler). Khi một thông điệp đang được xử lý, thì các điều khiển này đã được đăng ký để sẽ thực hiện. Các điều khiển này được đăng ký ở mức độ toàn cục, dịch vụ, hoặc trong một phạm vi hoạt động giới hạn của máy chủ Axis2. Cuối cùng Axis2 kết nối các điều khiển của các phạm vi trên thành một chuỗi điều khiển. Các điều khiển có thể hoạt động như một máy đánh chặn, nghĩa là chúng xử lý một phần thông điệp và sau đó cung cấp các tiện ích dịch vụ (add on services) cho hệ thống khác. Mô tả các bước truyền thông điệp trong mô hình xử lý thông điệp SOAP ở Hình 1.8

Khi một thông điệp được gửi đến thông qua Client API, ống OutPipe trên vùng người dùng được kích hoạt. Ống OutPipe sẽ gọi các xử lý (handlers)

Bộ giao vận gửi thông điệp (Transport sender) trên vùng người dùng sẽ gửi thông điệp SOAP tới ống InPipe trên vùng dịch vụ.Thông điệp SOAP này cuối cùng được bộ giao vận gửi thông điệp (Transport sender) trên vùng dịch vụ chuyển tới cuối điểm mục tiêu (endpoint) trên vùng dịch vụ.

Thông điệp SOAP này sau đó được nhận bởi bộ giao vận nhận (Transport reviever) của vùng người dùng ở điểm mục tiêu (endpoint). Thông điệp SOAP tiếp tục được chuyển vào ống InPipe ở vùng người dùng. Trong ống InPipe này bao gồm các thao các xử lý( handlers) và kết thúc việc nhận thông điệp SOAP. Đồng thời kết thúc quá trình gửi và nhận thông điệp phản hồi SOAP cho Client API.

Quá trình trao đổi thông điệp ở trên xảy ra với mỗi thông điệp tại một thời điểm và với mọi thông điệp. Sau khi xử lý xong một thông điệp, Axis2 có thể quyết định tạo ra một thông điệp SOAP khác. Axis2 luôn luôn xem các thông điệp SOAP ở dạng xử lý đơn. Sự quản lý và kết nối các thông điệp được thực hiện ở lớp trên của khung (framework). Hai ống InPipe và OutPipe không có sự phân biệt khác nhau giữa vùng máy chủ và vùng người dùng.

##### Các phiên bản chính của Apache Axis2

Standard Binary Distribution: là phiên bản hoàn chỉnh của Axis, chứa sample và nhiều script hữu ích. Nó được sử dụng để cài đặt như một server độc lập và dùng trong quá trình phát triển ứng dụng

WAR (Web Archive) Distribution (Axis2.war): là ứng dụng web của Axis2, được triển khai trên hầu hết các trình chứa Servlet Container, ví dụ Tomcat…

## 1.2 Kiến trúc hướng dịch vụ

### 1.2.1 Kiến trúc hướng dịch vụ (SOA) là gì?

Kiến trúc hướng dịch vụ - SOA (Service Oriented Architecture) là một cách tiếp cận hay một phương pháp luận để thiết kế và tích hợp các thành phần khác nhau, bao gồm các phần mềm và các chức năng riêng lẻ lại thành một hệ thống hoàn chỉnh. Kiến trúc SOA rất giống với cấu trúc của các phần mềm hướng đối tượng gồm nhiều module. Tuy nhiên khái niệm module trong SOA không đơn thuần là một gói phần mềm, hay một bộ thư viện nào đó. Thay vào đó, mỗi module trong một ứng dụng SOA là một dịch vụ được cung cấp rải rác ở nhiều nơi khác nhau và có thể truy cập thông qua môi trường mạng. Nói một cách ngắn gọn, một hệ thống SOA là một tập hợp nhiều dịch vụ được cung cấp trên mạng, được tích hợp lại với nhau để cùng cộng tác thực hiện các tác vụ nào đó theo yêu cầu của khác hàng.

Một trong những cách hiểu sai lầm nhất về SOA là coi SOA là một công nghệ. Mặc dù SOA hoạt động được là nhờ công nghệ, nhưng khách hàng cần phải chuyển đổi từ chỗ chỉ việc tích hợp công nghệ SOA sang việc phải điều chỉnh các phương pháp thực hiện dự án, chính sách bảo trì và thay đổi để đạt được các lợi ích về khả năng trưởng thành và đáp ứng.

Dịch vụ (Service) là yếu tố then chốt trong SOA. Có thể hiểu dịch vụ như là một loại module thực hiện một quy trình nghiệp vụ nào đó. Một trong những mục đích của SOA là giúp các ứng dụng có thể “giao tiếp” được với nhau mà không cần biết các chi tiết kỹ thuật bên trong. Để thực hiện điều đó SOA định ra một chuẩn giao tiếp (dùng để gọi hàm dịch vụ) được định nghĩa rõ ràng và độc lập với nền tảng hệ thống, và có thể tái sử dụng. Như vậy, SOA là cấp độ cao hơn của phát triển ứng dụng, chú trọng đến quy trình nghiệp vụ và dùng giao tiếp chuẩn để giúp che đi sự phức tạp kỹ thuật bên dưới. Sự trừu tượng là cốt lõi của khái niệm dịch vụ, nó giúp cho các doanh nghiệp có thể tích hợp các thành phần hiện có vào các ứng dụng mới và các thành phần này có thể được chia sẻ hoặc tái sử dụng trong nhiều lĩnh vực khác nhau của công ty đó mà không cần phải chỉnh sửa mã nguồn hay phải tái cấu trúc lại hệ thống.

Có nhiều cách khác nhau để kết nối các dịch vụ, chẳng hạn dùng các giao thức mạng có sẵn, hoặc tạo một giao thức riêng. Nhưng trong nhiều năm trở lại đây, các dịch vụ web (Web Service) được xây dựng dựa trên nền tảng web toàn cầu, bất cứ nơi nào cũng có, đã trở thành một phương pháp phổ biến cho việc kết nối các thành phần của hệ thống SOA với nhau. Thoạt nhìn SOA và Web Service trông có vẻ giống nhau nhưng chúng không phải là một.



Hình 1.9 Mô hình SOA cơ bản

Trong mô hình SOA cơ bản, nhà cung cấp dịch vụ nhận yêu cầu sử dụng dịch vụ từ người sử dụng, và phản hồi yêu cầu đến người sử dụng

**Find**

**Register**

**Bind,**

**Execute**

Hình 1.10 Mô hình SOA cấp cao

Mô hình SOA cao cấp là mô hình đang được sử dụng hiện nay. Trong mô hình này, nhà cung cấp dịch vụ (Service Provider) cần cung cấp thông tin về dịch vụ của mình cho một dịch vụ lưu trữ thông tin dịch vụ (Service Registry). Người sử dụng dịch vụ (Service Consume) thông qua Service Registry để tìm kiếm thông tin mô tả về dịch vụ cần tìm và sau đó xây dựng kênh giao tiếp với Service Provider

### 1.2.2 Các nguyên tắc chính của hệ thống SOA

### 1.2.3 Các tính chất của một hệ thống SOA

### 1.2.4 Kiến trúc phân tầng chi tiết của SOA

## 1.3 Quy trình xây dựng SOA

## 1.4 Ngôn ngữ thi hành quy trình nghiệp vụ - BPEL

### 1.4.1 Giới thiệu

### 1.4.2 Các khái niệm cơ bản

## 1.5 Tiểu kết chương 1

# Chương 2: Khung ứng dụng hỗ trợ lập trình SOA

## 2.1 Nền tảng Eclipse

### 2.1.1 Giới thiệu

### 2.1.2 Các thành phần và kiến trúc

### 2.1.3 Eclipse và SOA

## 2.2 Kiến trúc SOA trong Eclipse

## 2.3 Kiến trúc plug-in của Eclipse

### 2.3.1 Đặc điểm và cấu hình

### 2.3.2 Tuỳ biến với dịch vụ Web

### 2.3.3 Nhận xét

## 2.4 Tiểu kết chương 2

# Chương 3: Xây dựng ứng dụng trên nền tảng Eclipse

## 3.1 Bài toán điều phối các lời gọi dịch vụ trong kiến trúc SOA

## 3.2 Điều phối dịch vụ Web

## 3.3 Xây dựng Bus dịch vụ

## 3.4 Xây dựng kiến trúc “plug-and-play” dựa trên SOA

# Phần kết luận và hướng phát triển

1. Extensible Markup Language, http://www.w3.org/XML/ [↑](#footnote-ref-1)