

```

<portType name="EmployeeInterface">
  <tài liệu>
    GetEmployeeWeeklyHoursLimit sử dụng giá trị ID nhân viên để truy xuất
    giá trị WeeklyHoursLimit.
    UpdateEmployeeHistory sử dụng giá trị ID nhân viên để cập nhật giá trị Nhận
    xét của Lịch sử nhân viên.
  </documentation> <tên
  hoạt động="GetEmployeeWeeklyHoursLimit">
    <tin nhắn đầu vào=
      "tns:getEmployeeWeeklyHoursRequestMessage"/>
    <tin nhắn đầu ra=
      "tns:getEmployeeWeeklyHoursResponseMessage"/>
  </ hoạt động> <tên
  hoạt động="UpdateEmployeeHistory"> <thông báo đầu vào=
    "tns:updateEmployeeHistoryRequestMessage"/>
  <tin nhắn đầu ra=
    "tns:updateEmployeeHistoryResponseMessage"/>
  </hoạt động> </
portType>

```

Ví dụ 8.5

Hợp đồng dịch vụ, được bổ sung tài liệu siêu dữ liệu bổ sung.

Kiến trúc sự phụ trách thiết kế dịch vụ Nhân viên quyết định điều chỉnh giao diện dịch vụ trừu tượng để áp dụng các tiêu chuẩn thiết kế hiện tại. Cụ thể, các quy ước đặt tên được kết hợp để chuẩn hóa các tên hoạt động, như trong Hình 8.9 và Ví dụ 8.6.



Hình 8.9

Tên hoạt động dịch vụ nhân viên được sửa đổi.

```

< tên hoạt động="GetWeeklyHoursLimit">
    <input message="tns:getWeeklyHoursRequestMessage"/>
    <output message="tns:getWeeklyHoursResponseMessage"/>
</ hoạt động>
< tên hoạt động="UpdateHistory">
    <input message="tns:updateHistoryRequestMessage"/>
    <output message="tns:updateHistoryResponseMessage"/>
</ hoạt động>

```

Ví dụ 8.6

Hai cấu trúc hoạt động có tên mới, được tiêu chuẩn hóa.

Chúng ta hãy xem xét lại hai thao tác đã được thiết kế trong

Dịch vụ nhân viên:

- Giới hạn số giờ hàng tuần
- Cập nhật lịch sử

Việc đầu tiên yêu cầu quyền truy cập vào hồ sơ nhân viên. Tại TLS, thông tin nhân viên được lưu trữ ở hai vị trí:

- Dữ liệu tiền lương được lưu giữ trong kho lưu trữ của hệ thống kế toán cùng với thông tin liên hệ bổ sung của nhân viên.
- Thông tin hồ sơ nhân viên, bao gồm chi tiết lịch sử nhân viên, được lưu trữ trong kho nhân sự.

Khi kiến trúc biểu diễn dữ liệu Lược đồ XML lần đầu tiên được triển khai tại TLS, các lược đồ XML thực thể đã được sử dụng để thu hẹp một số khác biệt hiện có tồn tại trong số nhiều nguồn dữ liệu TLS. Nhận thức được điều này, kiến trúc sư đã điều tra nguồn gốc của lược đồ Staff.xsd được sử dụng như một phần của định nghĩa Staff.wsdl để xác định các yêu cầu xử lý cho hoạt động GetWeeklyHoursLimit.

Người ta phát hiện ra rằng mặc dù lược đồ thể hiện chính xác một thực thể dữ liệu logic, nó đại diện cho một cấu trúc tài liệu bắt nguồn từ hai kho lưu trữ vật lý khác nhau.

Phân tích tiếp theo cho thấy giá trị giới hạn số giờ hàng tuần được lưu trữ trong cơ sở dữ liệu kế toán. Sau đó, yêu cầu xử lý cho hoạt động GetWeeklyHoursLimit được viết như sau:

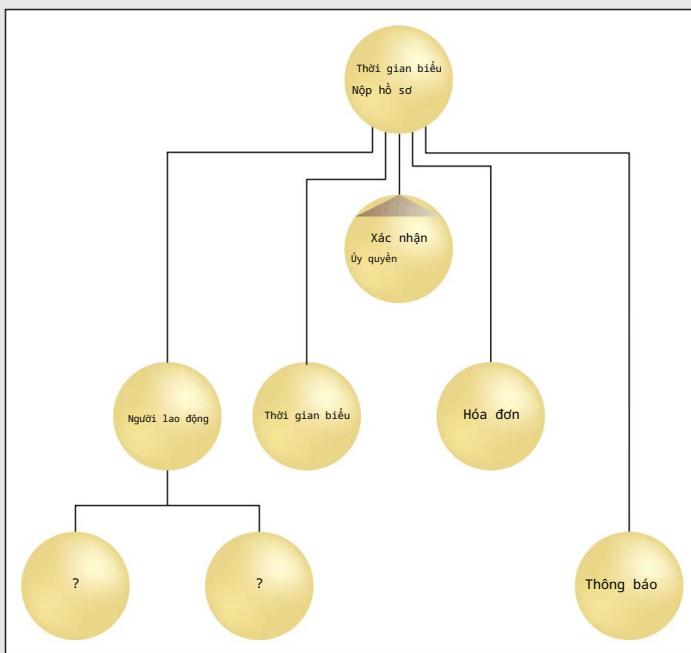
Chức năng cấp dịch vụ tiện ích có khả năng đưa ra truy vấn sau đối với cơ sở dữ liệu kế toán: Trả về giới hạn số giờ hàng tuần của nhân viên bằng cách sử dụng ID nhân viên làm Tìm kiếm duy nhất Tiêu chuẩn

Tiếp theo, các chi tiết đằng sau thao tác UpdateHistory sẽ được nghiên cứu. Lần này có hơi chút dễ dàng hơn vì lược đồ MemberHistory.xsd được liên kết với một nguồn dữ liệu duy nhất-kho lưu trữ hồ sơ nhân viên nhân sự. Nhìn lại tài liệu phân tích ban đầu, kiến trúc sư xác định một phần thông tin mà giải pháp cụ thể này

sẽ cần cập nhật trong kho lưu trữ này. Vì vậy yêu cầu xử lý định nghĩa vượt xa các yêu cầu trước mắt của giải pháp, như sau:

Chức năng cấp độ dịch vụ tiện ích có khả năng đưa ra bản cập nhật cho cột “nhận xét” của bảng lịch sử nhân viên trong cơ sở dữ liệu hồ sơ nhân viên nhân sự, sử dụng giá trị ID nhân viên làm tiêu chí duy nhất.

Thoạt nhìn, có vẻ như giải pháp Gửi bằng châm công có thẻ yêu cầu các dịch vụ tiện ích mới để hỗ trợ các yêu cầu xử lý dịch vụ của Nhân viên, như được minh họa trong thành phần mở rộng được thể hiện trong Hình 8.10. Những yêu cầu mới được xác định này sẽ cần phải tuân theo quy trình mô hình hóa dịch vụ được mô tả trong Chương 6.



Hình 8.10

Hệ thống phân cấp thành phần được sửa đổi xác định các dịch vụ tiện ích tiềm năng mới.

Cuối cùng, người ta tiết lộ rằng chỉ cần một dịch vụ tiện ích mới để đáp ứng dịch vụ Nhân viên – dịch vụ bao bọc Nhân sự cũng có thể tạo điều kiện thuận lợi cho dịch vụ Bảng chấm công. Ví dụ 8.7 chứa phiên bản cuối cùng của định nghĩa dịch vụ Nhân viên, kết hợp các thay đổi về tên thành phần và các phiên bản trước đó.

```
<định nghĩa name="Nhân viên"
    targetNamespace="http://www.example.org/tls/employee/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:act= "http://www.example.org/tls/employee/schema/accounting/"
    xmlns:hr="http://www.example.org/tls/employee/schema/hr/" xmlns:soap="http://schemas.xmlsoap.org/
    wsdl/soap/" xmlns:tns="http://www.example.org/tls/employee/wsdl/" xmlns:xsd="http://
    www.w3.org/2001/XMLSchema"> <types> <xml:schema targetNamespace= "http:// www.example.org/
    tls/employee/schema/"> <xml:import namespace= "http://www.example.org/tls/employee/schema/ Accounting/"

    SchemaLocation="Employee.xsd"/>
    <xml:nhập không gian tên=
        "http://www.example.org/tls/employee/schema/hr/" SchemaLocation="EmployeeHistory.xsd" />
    </xml:lực dò>
</types>
<message name="getWeeklyHoursRequestMessage">
    <part name="RequestParameter"
        element="act:EmployeeHoursRequestType"/>
</message>
<message name="getWeeklyHoursResponseMessage">
    <part name="ResponseParameter"
        element="act:EmployeeHoursResponseType"/>
</message>
<message name="updateHistoryRequestMessage">
    <part name="RequestParameter"
        element="hr:EmployeeUpdateHistoryRequestType"/>
</message>
<message name="updateHistoryResponseMessage">
    <part name="ResponseParameter"
        element="hr:EmployeeUpdateHistoryResponseType"/>
</message>
<portType name="EmployeeInterface"> <documentation>
```

```

GetWeeklyHoursLimit sử dụng giá trị ID nhân viên để truy xuất giá trị
WeeklyHoursLimit.

UpdateHistory sử dụng giá trị ID nhân viên để cập nhật giá trị Nhận xét
của Lịch sử nhân viên.

</tài liệu>

<tên hoạt động="GetWeeklyHoursLimit"> <tin nhắn đầu vào =
"tns:getWeeklyHoursRequestMessage"/>
<thông báo đầu ra =
"tns:getWeeklyHoursResponseMessage"/>
</ hoạt động> <tên
hoạt động="UpdateHistory"> <input message= "tns:updateHistoryRequestMessage"/>
<thông báo đầu ra =
"tns:updateHistoryResponseMessage"/>
</hoạt động> </
portType>
...
</định nghĩa>
```

Ví dụ 8.7

Định nghĩa dịch vụ trừu tượng cuối cùng cho hợp đồng dịch vụ Nhân viên. Bước tiếp theo cho dịch vụ này sẽ là tiến hành định nghĩa dịch vụ cụ thể và logic dịch vụ của nó.

8.2 Nguyên tắc thiết kế dịch vụ web

Phần này cung cấp một tập hợp các hướng dẫn chung cho việc thiết kế các hợp đồng dịch vụ Web. Một số hướng dẫn này có thể trở thành cơ sở của các tiêu chuẩn thiết kế tùy chỉnh chính thức.

Áp dụng tiêu chuẩn đặt tên

Dán nhãn các dịch vụ Web tương đương với việc dán nhãn cơ sở hạ tầng CNTT. Do đó, điều cần thiết là các API dịch vụ phải có khả năng tự mô tả nhất quán nhất có thể.

Do đó, tiêu chuẩn đặt tên cần được xác định và áp dụng cho:

- Tên điểm cuối dịch vụ
- Tên hoạt động dịch vụ
- Giá trị tin nhắn

Đây là một vài gợi ý:

- Các ứng viên dịch vụ có tiềm năng tái sử dụng cao phải luôn được loại bỏ mọi đặc điểm đặt tên gợi ý về quy trình kinh doanh mà chúng được xây dựng ban đầu. Ví dụ: thay vì đặt tên một thao tác là GetTimesheetSubmissionID, nó có thể được rút gọn thành GetTimesheetID hoặc thậm chí chỉ là GetID.

- Các dịch vụ thực thể cần phải mang tính đại diện cho các mô hình thực thể mà từ đó các ứng cử viên dịch vụ tương ứng của chúng được tạo ra. Do đó, các quy ước đặt tên được sử dụng phải phản ánh những quy ước được thiết lập trong các mô hình thực thể ban đầu của tổ chức. Thông thường, loại dịch vụ này sử dụng cấu trúc đặt tên chỉ danh từ. Ví dụ về tên dịch vụ thực thể phù hợp là Hóa đơn, Khách hàng và Nhân viên.

- Hoạt động dịch vụ cho các dịch vụ thực thể phải dựa trên động từ và không được lặp lại tên thực thể. Ví dụ: một dịch vụ thực thể có tên là Invoice không được có thao tác có tên AddInvoice.

- Các dịch vụ tiện ích cần được đặt tên theo ngữ cảnh xử lý trong hoạt động của họ được nhóm lại. Cả quy ước động từ+danh từ hoặc chỉ danh từ đều có thể được sử dụng. Ví dụ đơn giản về tên dịch vụ tiện ích phù hợp là CustomerDataAccess, SalesReporting và GetStatistics.

- Hoạt động dịch vụ tiện ích cần truyền đạt rõ ràng bản chất chức năng riêng lẻ của chúng. Ví dụ về tên hoạt động dịch vụ tiện ích phù hợp là GetReport, ConvertCurrency và VerifyData.

- Mặc dù các dịch vụ vi mô không phải lúc nào cũng phải tuân theo các tiêu chuẩn thiết kế giống như các dịch vụ bất khả tri, nhưng chúng tôi vẫn khuyến nghị rằng các quy ước về tên dịch vụ và hoạt động nên được áp dụng nhất quán ở mọi mức độ có thể.

Dù tiêu chuẩn đặt tên nào được chọn, điều quan trọng là chúng phải được áp dụng nhất quán trên tất cả các dịch vụ trong kho dịch vụ nhất định.

MẪU SOA

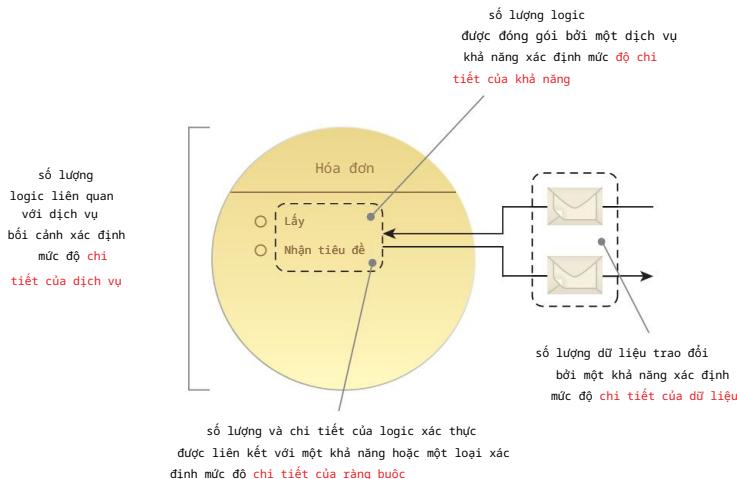
Mẫu Canonical Expression [325] chính thức hóa việc sử dụng các quy ước đặt tên cho mục đích tiêu chuẩn hóa.

Áp dụng mức độ chi tiết API hợp đồng phù hợp

Khi thiết kế dịch vụ, cần phải thực hiện các mức độ chi tiết khác nhau được xem xét như sau:

- **Mức độ chi tiết của dịch vụ** - Điều này thể hiện phạm vi chức năng của một dịch vụ. Ví dụ, Mức độ chi tiết của dịch vụ được phân loại chi tiết cho thấy rằng có một lượng nhỏ logic liên quan đến bối cảnh chức năng tổng thể của dịch vụ.
- **Mức độ chi tiết về năng lực** - Phạm vi chức năng của các khả năng dịch vụ riêng lẻ là được thể hiện bằng mức độ chi tiết này. Ví dụ: khả năng GetDetail sẽ có xu hướng để có mức độ chi tiết tốt hơn khả năng GetDocument.
- **Mức độ chi tiết của ràng buộc** - Mức độ chi tiết logic xác thực được đo bằng ràng buộc độ chi tiết. Ví dụ, độ chi tiết của ràng buộc càng thô thì càng ít ràng buộc (hoặc lượng logic xác thực dữ liệu nhỏ hơn) mà một khả năng nhất định sẽ có.
- **Độ chi tiết của dữ liệu** - Mức độ chi tiết này thể hiện số lượng dữ liệu được xử lý. Ví dụ: mức độ chi tiết của dữ liệu tốt tương đương với một lượng nhỏ dữ liệu.

Vì mức độ chi tiết của dịch vụ xác định phạm vi chức năng của dịch vụ nên thường được xác định trong giai đoạn phân tích và lập mô hình trước khi thiết kế hợp đồng dịch vụ. Khi phạm vi chức năng của dịch vụ đã được thiết lập, các loại chi tiết khác sẽ phát huy tác dụng và ảnh hưởng đến cả mô hình hóa và thiết kế vật lý của hợp đồng dịch vụ (Hình 8.11).



Hình 8.11

Bốn cấp độ chi tiết thể hiện các đặc điểm khác nhau của dịch vụ và hợp đồng của dịch vụ đó. Lưu ý rằng các loại chi tiết này phần lớn độc lập với nhau.

Độ chi tiết thường được đo ở mức độ mịn và thô. Cần phải thừa nhận rằng việc sử dụng các thuật ngữ hạt mịn và hạt thô mang tính chủ quan cao. Những gì có thể chi tiết trong trường hợp này có thể không có trong trường hợp khác. Vấn đề là phải hiểu cách áp dụng các thuật ngữ này khi so sánh các bộ phận của một dịch vụ hoặc khi so sánh các dịch vụ với nhau.

GHI CHÚ

Thuật ngữ “độ chi tiết của ràng buộc” không liên quan đến ràng buộc thuật ngữ vì nó liên quan đến REST.

Mặc dù mức độ chi tiết mà các dịch vụ có thể được thiết kế có thể khác nhau, nhưng có một nguyên tắc chung là tạo các API cho các dịch vụ Web ở mức độ thô để tận dụng tối đa mỗi lần trao đổi tin nhắn. Tất nhiên, hiệu suất là yếu tố quan trọng đối với sự thành công và sự phát triển cuối cùng của các giải pháp hướng dịch vụ. Tuy nhiên, những cân nhắc khác cũng cần được đưa vào tài khoản.

Mức độ chi tiết của hợp đồng dịch vụ càng thô thì khả năng tái sử dụng nó càng ít. Nếu như nhiều chức năng được nhóm lại thành một hoạt động duy nhất, điều này có thể không mong muốn đối với người tiêu dùng chỉ yêu cầu sử dụng một trong các chức năng đó. Ngoài ra, một số API chi tiết thực sự có thể áp đặt việc xử lý hoặc trao đổi dữ liệu dư thừa bằng cách buộc người tiêu dùng gửi dữ liệu không liên quan đến một hoạt động cụ thể.

Mức độ chi tiết của hợp đồng dịch vụ là điểm quyết định chiến lược quan trọng đáng được chú ý trong giai đoạn thiết kế hướng dịch vụ. Dưới đây là một số hướng dẫn để giải quyết vấn đề này:

- Hiểu đầy đủ các hạn chế về hiệu suất của môi trường triển khai mục tiêu tư vấn và khám phá các công nghệ hỗ trợ thay thế, nếu cần.
- Khám phá khả năng cung cấp nguyên liệu thay thế (thô và ít thô hơn)

Các định nghĩa WSDL cho cùng các dịch vụ Web. Hoặc khám phá các tùy chọn cung cấp các hoạt động thô dư thừa và ít thô hơn trong cùng một định nghĩa WSDL. Những cách tiếp cận này phi chuẩn hóa các hợp đồng dịch vụ nhưng có thể giải quyết các vấn đề về hiệu suất và đáp ứng nhiều đối tượng người tiêu dùng.
- Chỉ định các API chi tiết cho các dịch vụ được chỉ định làm điểm cuối giải pháp và cho phép các API chi tiết hơn cho các dịch vụ được giới hạn trong các ranh giới được xác định trước. Cái này của Tất nhiên, hơi trái ngược với các nguyên tắc định hướng dịch vụ và đặc điểm của SOA nhằm thúc đẩy việc tái sử dụng và khả năng tương tác trong các dịch vụ. Khả năng tương tác

được thúc đẩy trong các dịch vụ chi tiết thô và khả năng sử dụng lại được thúc đẩy nhiều hơn trong các dịch vụ chi tiết hơn.

- Xem xét việc sử dụng các hợp đồng dịch vụ thử cấp hỗ trợ thay thế, hơn thế nữa giao thức truyền thông hiệu quả. Mặc dù nó làm tăng thêm gánh nặng quản trị, nhưng nó có thể hỗ trợ phương tiện truyền thông thứ hai trong kho dịch vụ.
Ví dụ: có thể được đảm bảo cung cấp hỗ trợ cho các dịch vụ REST cùng với Các dịch vụ Web dựa trên SOAP.

Bất kể cách tiếp cận của bạn là gì, hãy đảm bảo rằng nó nhất quán và có thể dự đoán được để SOA có thể đáp ứng nhu cầu hiệu suất trong khi vẫn được tiêu chuẩn hóa.

VÍ DỤ NGHIÊN CỨU TRƯỜNG HỢP

TLS đã chọn một cách tiếp cận để thu xếp mức độ chi tiết của API trong đó các dịch vụ được định vị để người tiêu dùng bên ngoài TLS sử dụng sẽ cung cấp các API chi tiết thô một cách nhất quán. Hoạt động trên các dịch vụ này sẽ chấp nhận tất cả dữ liệu cần thiết để xử lý một hoạt động cụ thể. Các chuyển đổi khứ hồi tiếp theo giữa người tiêu dùng bên ngoài và dịch vụ sẽ chỉ được yêu cầu nếu thực sự cần thiết hoặc nếu các chính sách nội bộ yêu cầu điều đó. Các dịch vụ được sử dụng trong TLS có thể cung cấp các hoạt động ít chi tiết hơn để tạo điều kiện tái sử dụng và phạm vi rộng hơn cho người tiêu dùng tiềm năng (nội bộ), miễn là chi phí xử lý được áp đặt bởi các hoạt động ít thô hơn có thể chấp nhận được.

MẪU SOA

Việc cung cấp các hợp đồng thay thế cho cùng một dịch vụ được đề cập trong mẫu Hợp đồng thuê đồng thời [332]. Việc thêm các hoạt động dư thừa trong cùng một hợp đồng dịch vụ Web được chính thức thông qua mẫu Không chuẩn hóa hợp đồng [335].

Hỗ trợ cho hai giao thức liên lạc trong cùng một kho dịch vụ được mô tả trong mẫu Giao thức kép [339].

Thiết kế các hoạt động dịch vụ web để có thể mở rộng được

Bất kể các dịch vụ được thiết kế tốt như thế nào khi triển khai lần đầu tiên, chúng không bao giờ có thể được sử dụng. chuẩn bị đầy đủ cho những gì tương lai sẽ xảy ra. Một số loại thay đổi quy trình kinh doanh dẫn đến nhu cầu mở rộng phạm vi của các thực thể. Kết quả là, hoạt động kinh doanh tương ứng

dịch vụ có thể cần được mở rộng. Mặc dù ứng dụng Khả năng sử dụng lại dịch vụ (295) và Khả năng kết hợp dịch vụ (302) được xem xét kỹ lưỡng khi phân vùng logic như một phần của quy trình mô hình hóa dịch vụ, khả năng mở rộng thiên về chất lượng thiết kế vật lý hơn cần được xem xét trong quá trình thiết kế.

Tùy thuộc vào bản chất của sự thay đổi, đôi khi có thể đạt được khả năng mở rộng mà không vi phạm hợp đồng dịch vụ hiện có. Điều quan trọng là phải thiết kế các hoạt động và thông báo của dịch vụ Web sao cho càng bất khả thi về hoạt động càng tốt. Điều này hỗ trợ việc xử lý các giá trị và hàm không cụ thể trong tương lai mà vẫn liên quan đến hoạt động hoặc mục đích chung của tin nhắn. Hơn nữa, đó là một thói quen tốt để đáp ứng với quá trình xử lý mới. yêu cầu bằng cách trước tiên điều tra khả năng tạo ra các dịch vụ sẵn có khác (bao gồm cả các dịch vụ có thể mua hoặc thuê). Điều này có thể thành công trong việc đáp ứng các yêu cầu mà không cần phải động tới hợp đồng dịch vụ.

Lưu ý rằng các phần mở rộng cho hợp đồng dịch vụ hiện có thường sẽ ảnh hưởng đến lược đồ XML tương ứng của hợp đồng đó. Những phần mở rộng này có thể được tạo điều kiện thuận lợi bằng cách cung cấp các lược đồ mới cụ thể cho phần mở rộng. Tuy nhiên, trước khi đi theo con đường này, hãy đảm bảo rằng các tiêu chuẩn kiểm soát phiên bản đã được thiết lập được áp dụng một cách chắc chắn.

VÍ DỤ NGHIÊN CỨU TRƯỜNG HỢP

Do quy mô tổ chức của TLS, việc nhân viên làm việc ở vị trí thực tế hoặc tìm kiếm sự thay đổi vị trí theo chiều dọc hoặc chiều ngang không phải là hiếm. Kịch bản thứ hai càng trở nên phổ biến hơn nhờ phương châm “thăng tiến từ bên trong” được khuyến khích bởi nhiều bộ phận. đạo diễn.

Khi một nhân viên thay đổi vị trí hoặc cấp bậc, nhân viên đó phải cập nhật hồ sơ của chính họ bằng cách sử dụng biểu mẫu trên mạng nội bộ. Bởi vì bước này là tự nguyện, nó thường không bao giờ được thực hiện. Điều này, có thể dự đoán được, dẫn đến một ngày càng lão thời tập hợp các hồ sơ. Để chống lại xu hướng này, quy trình Gửi Bảng chấm công TLS đã được thay đổi để bao gồm bước Xác minh Hồ sơ Nhân viên. Khi thực hiện, nó sẽ xác minh thông tin hồ sơ trước khi chấp nhận bảng chấm công. Bảng chấm công được gửi bởi nhân viên có hồ sơ không hợp lệ sẽ bị từ chối.

Để thực hiện yêu cầu mới này, hợp đồng dịch vụ Bảng chấm công không bị thay đổi.

Thay vào đó, logic dịch vụ cơ bản được mở rộng để gọi một dịch vụ tiện ích riêng biệt thực hiện việc xác minh hồ sơ.

MẪU SOA

Một ví dụ về mẫu có thể được áp dụng để hỗ trợ khả năng mở rộng trong tương lai là Trình tương hóa xác thực [365], làm giảm mức độ chi tiết của ràng buộc để hỗ trợ các thay đổi tiềm năng đối với logic xác thực.

Cân nhắc sử dụng tài liệu WSDL mô-đun

Các mô tả dịch vụ WSDL có thể được tập hợp động trong thời gian chạy thông qua việc sử dụng các câu lệnh nhập liên kết đến các tệp riêng biệt chứa các phần của định nghĩa dịch vụ. Điều này cho phép bạn xác định các mô-đun cho các loại, hoạt động và liên kết có thể được chia sẻ trên các tài liệu WSDL.

Nó cũng cho phép bạn tận dụng mọi mô-đun Lược đồ XML hiện có mà bạn có thể đã thiết kế. Bạn có thể tách các lược đồ thành các mô-đun chi tiết thể hiện các kiểu phức tạp riêng lẻ. Điều này thiết lập một kho lưu trữ tập trung các lược đồ có thể được tập hợp thành các định nghĩa lược đồ chính tùy chỉnh. Bằng cách cho phép bạn nhập các mô-đun Lược đồ XML vào cấu trúc kiểu của định nghĩa WSDL, giờ đây bạn có thể có

Các tài liệu WSDL sử dụng cùng các mô-đun lược đồ đó.

VÍ DỤ NGHIÊN CỨU TRƯỜNG HỢP

TLS xem xét việc nhập cấu trúc liên kết để nó có thể được sử dụng lại và thậm chí có thể được xác định một cách linh hoạt. Tuy nhiên, sau đó người ta quyết định để lại các ràng buộc được xây dựng như một phần của tài liệu WSDL. Ví dụ 8.8 cho thấy cách sử dụng câu lệnh nhập để thực hiện kiểm tra này.

```
<import namespace="http://.../common/wsdl/"
           location="http://.../common/wsdl/binds.wsdl"/>
```

Ví dụ 8.8

Một phần tử nhập được sử dụng để kéo cấu trúc liên kết nằm trong một tệp riêng biệt.

Sử dụng không gian tên một cách cẩn thận

Định nghĩa WSDL bao gồm một tập hợp các phần tử có nguồn gốc khác nhau. Vì thế, mỗi định nghĩa thường sẽ liên quan đến một số không gian tên khác nhau. Sau đây là danh sách các không gian tên phổ biến được sử dụng để thể hiện các phần tử dựa trên thông số kỹ thuật:

```
http://schemas.xmlsoap.org/wsdl/
http://schemas.xmlsoap.org/wsdl/soap/
http://www.w3.org/2001/XMLSchema/
http://schemas.xmlsoap.org/wsdl/http/
http://schemas.xmlsoap.org/wsdl/mime/
http://schemas.xmlsoap.org/soap/envelope/
```

Khi tập hợp một WSDL từ các mô-đun, các không gian tên bổ sung sẽ phát huy tác dụng, đặc biệt khi nhập các định nghĩa Lược đồ XML. Hơn nữa, khi xác định các phần tử của riêng bạn, bạn có thể thiết lập nhiều không gian tên hơn để thể hiện các phần dành riêng cho ứng dụng của Tài liệu WSDL. Không có gì lạ khi các tài liệu WSDL lớn hơn chứa tới mười không gian tên khác nhau và các bộ định tính đi cùng với chúng. Vì vậy, nó rất cao khuyên bạn nên tổ chức việc sử dụng các không gian tên một cách cẩn thận trong và trên các tài liệu WSDL.

Quy ước chung là yêu cầu sử dụng thuộc tính targetNamespace để gán toàn bộ không gian tên cho WSDL. Nếu lược đồ XML được nhúng bên trong định nghĩa WSDL thì nó cũng có thể được gán một giá trị targetNamespace (có thể là giá trị cùng một giá trị được sử dụng bởi WSDL targetNamespace).

VÍ DỤ NGHIÊN CỨU TRƯỞNG HỢP

Một số không gian tên phổ biến được xác định trước đó không được dịch vụ TLS Nhân viên yêu cầu và do đó bị loại bỏ khỏi danh sách các thuộc tính định nghĩa .

Như được hiển thị trong Ví dụ 8.9, không gian đích được thêm vào, cùng với hai không gian tên được liên kết với hai lược đồ đã nhập.

```
<định nghĩa name="Nhân viên"
    targetNamespace="http://www.xmltc.com/tls/employee/wsdl/"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:hành động=
        "http://www.xmltc.com/tls/employee/schema/accounting/"
    xmlns:hr="http://www.xmltc.com/tls/employee/schema/hr/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
```

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:tns="http://www.xmltc.com/tls/employee/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    ...
</wsdl:definitions>
```

Ví dụ 8.9

Các khai báo vùng tên trong phần tử định nghĩa của tệp TLS Staff.wsdl.

Sử dụng Tài liệu SOAP và các giá trị thuộc tính theo nghĩa đen

Hai thuộc tính cụ thể thiết lập định dạng tải trọng thông báo SOAP và hệ thống kiểu dữ liệu được sử dụng để thể hiện dữ liệu tải trọng. Đây là thuộc tính style được sử dụng bởi phần tử Soap:bind và thuộc tính use được gán cho phần tử Soap:body . Cả hai phần tử này đều nằm trong cấu trúc liên kết WSDL .

Cách thiết lập các thuộc tính này rất quan trọng vì nó liên quan đến cách cấu trúc và trình bày nội dung thông báo SOAP.

Thuộc tính style có thể được gán giá trị là "document" hoặc "rpc". Cái trước hỗ trợ việc nhúng toàn bộ tài liệu XML trong phần thân SOAP, trong khi cái sau được thiết kế nhiều hơn để phản ánh giao tiếp RPC truyền thống và do đó hỗ trợ dữ liệu loại tham số.

Thuộc tính use có thể được đặt thành giá trị "nghĩa đen" hoặc "được mã hóa". SOAP ban đầu cung cấp hệ thống kiểu riêng của nó được sử dụng để thể hiện nội dung nội dung. Sau này, hỗ trợ cho XML Các kiểu dữ liệu lược đồ đã được kết hợp. Giá trị thuộc tính này cho biết loại hệ thống nào bạn muốn tin nhắn của mình sử dụng. Cài đặt "theo nghĩa đen" cho biết các kiểu dữ liệu Lược đồ XML sẽ được áp dụng.

Khi xem xét hai thuộc tính này, có thể có bốn kết hợp sau:

và được hỗ trợ bởi SOAP:

- kiểu:RPC + sử dụng:được mã hóa
- kiểu:RPC + sử dụng:nghĩa đen
- kiểu:tài liệu + sử dụng:được mã hóa
- kiểu:tài liệu + cách sử dụng:chữ

Sự kết hợp style:document + use:literal được SOA ưa thích hơn vì nó hỗ trợ khái niệm về mô hình nhắn tin kiểu tài liệu vốn là chìa khóa để hiện thực hóa các tính năng của nhiều đặc tả WS-*.

VÍ DỤ NGHIÊN CỨU TRƯỜNG HỢP

Khi xây dựng phần cụ thể của định nghĩa giao diện dịch vụ Nhân viên, kiến trúc sư TLS quyết định sử dụng kết hợp style:document + use:literal , như trong Ví dụ 8.10.

```
<bind name="EmployeeBinding"
    type="tns:EmployeeInterface"> <soap:bind
        style="document" Transport="http://
            schemas.xmlsoap.org/soap/http"/>
        <tên hoạt động="GetWeeklyHoursLimit"> <xà phòng:hoạt
            động
            SoapAction="http://www.xmltc.com/soapaction"/> <input> <soap:body
                use="literal"/> </input> <output>
                <soap:body
                    use="literal"/
                    > < /output> </ hoạt động> <tên hoạt

            động="UpdateHistory"> <soap: hoạt động

                SoapAction="http://www.xmltc.com/soapaction"/> <input> <soap:body use="literal"/
                > </input>
                <output> <soap:body use="literal"/> < /đầu
                    xà> </hoạt
                    động> </ràng
                    buộc>
```

Ví dụ 8.10

Cấu trúc ràng buộc của tài liệu TLS Staff.wsdl.

Trang này có ý đẻ trống



Chương 9

API dịch vụ và thiết kế hợp đồng với Dịch vụ REST và dịch vụ vi mô

9.1 Những cân nhắc khi thiết kế mô hình dịch

vụ 9.2 Nguyên tắc thiết kế dịch vụ REST

GHI CHÚ

Các phần của chương này đề cập đến cú pháp HTTP và các công nghệ liên quan đến REST được đề cập trong SOA với sách giáo khoa về loạt bài REST: Nguyên tắc, Mẫu & Ràng buộc.

Hợp đồng dịch vụ REST thường được thiết kế xoay quanh các chức năng chính của các phương thức HTTP, tạo tài liệu và tiêu chuẩn cho các giao thức REST. Hợp đồng khác biệt rõ ràng với hợp đồng dịch vụ Web dựa trên hoạt động. Bất kể sự khác biệt về ký hiệu, cách tiếp cận bao quát đầu tiên dựa trên hợp đồng để thiết kế.

Hợp đồng dịch vụ REST là điều tối quan trọng khi xây dựng dịch vụ cho kho dịch vụ được tiêu chuẩn hóa.

Với các dịch vụ REST nói riêng, có thể đạt được những lợi ích sau:

- Hợp đồng dịch vụ REST có thể được thiết kế để nhóm các khả năng một cách hợp lý liên quan đến bối cảnh chức năng được thiết lập trong quá trình phân tích hướng dịch vụ.
- Các quy ước có thể được áp dụng để chuẩn hóa chính thức tên tài nguyên và thông tin đầu vào để miêu tả dữ liệu.
- Các phương pháp phức tạp có thể được định nghĩa và tiêu chuẩn hóa để gói gọn một tập hợp các tương tác được xác định trước giữa dịch vụ và người tiêu dùng dịch vụ.
- Người sử dụng dịch vụ phải tuân theo cách thể hiện của dịch vụ hợp đồng chứ không phải ngược lại.
- Việc thiết kế các nguồn lực tập trung vào hoạt động kinh doanh và các phương pháp phức tạp có thể được hỗ trợ bởi các nhà phân tích kinh doanh có thể giúp thiết lập một biểu thức chính xác và hành vi của logic kinh doanh.

Chương này cung cấp hướng dẫn thiết kế hợp đồng dịch vụ cho các ứng viên dịch vụ được mô hình hóa nhờ giai đoạn phân tích hướng dịch vụ được đề cập trong Chương 7.

Lưu ý rằng thiết kế vật lý của API hợp đồng dịch vụ REST có thể tiết lộ các yêu cầu chức năng phù hợp hơn với các phương tiện triển khai thay thế. Ví dụ, nhu cầu thiết kế một API hoặc chức năng giao dịch phong phú hơn có thể đảm bảo việc xem xét việc sử dụng các dịch vụ Web dựa trên SOAP, như được giải thích trong Chương 8.

MẪU SOA

Theo mẫu Giao thức kép [339], các dịch vụ trong cùng một kho dịch vụ có thể dựa trên các phương tiện triển khai và giao thức truyền thông khác nhau. Ví dụ: các dịch vụ REST có thể nằm cùng với Web dựa trên SOAP dịch vụ.

Các mẫu Hợp đồng đồng thời [332] và Mặt tiền dịch vụ [360] có thể được áp dụng thêm để cho phép cùng một phần logic dịch vụ hiển thị các hợp đồng dịch vụ thay thế để hỗ trợ hai giao thức truyền thông tiêu chuẩn.

9.1 Những cản nhắc về thiết kế mô hình dịch vụ

Hợp đồng dịch vụ REST dựa trên bối cảnh chức năng được thiết lập trong quá trình phân tích hướng dịch vụ. Tùy thuộc vào bản chất của chức năng trong bối cảnh nhất định, mỗi dịch vụ sẽ được phân loại trong một mô hình dịch vụ.

Sau đây là một tập hợp các cản nhắc thiết kế hợp đồng dịch vụ cụ thể cho từng mô hình dịch vụ.

Thiết kế dịch vụ thực thể

Mỗi dịch vụ thực thể thiết lập một ranh giới chức năng liên kết với một hoặc nhiều thực thể kinh doanh liên quan (chẳng hạn như hóa đơn, yêu cầu bồi thường, khách hàng, v.v.). Các loại khả năng dịch vụ được cung cấp bởi một dịch vụ thực thể diễn hình tập trung vào các chức năng xử lý dữ liệu cơ bản được liên kết với thực thể (hoặc các thực thể).

Các hợp đồng dịch vụ thực thể REST thường bị chi phối bởi các khả năng dịch vụ bao gồm các phương thức GET, PUT hoặc DELETE vốn bình thường và đáng tin cậy. Các dịch vụ thực thể có thể cần hỗ trợ cập nhật trạng thái của chúng một cách nhất quán với những thay đổi đối với các dịch vụ thực thể khác. Các dịch vụ thực thể cũng thường bao gồm khả năng truy vấn để tìm kiếm các thực thể hoặc bộ phận của thực thể phù hợp với tiêu chí nhất định và do đó trả về các siêu liên kết đến các thực thể liên quan. và các đơn vị có liên quan.

Nếu các phương pháp phức tạp được cho phép như một phần của tiêu chuẩn thiết kế kho dịch vụ, thì các dịch vụ thực thể có thể được hưởng lợi từ việc bổ sung các khả năng dựa trên phương thức HTTP tiêu chuẩn với các tương tác được xác định trước được biểu thị bằng các phương thức phức tạp.

222 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

Hình 9.1 cung cấp một ví dụ về dịch vụ thực thể với hai phương thức HTTP tiêu chuẩn và hai phương thức phức tạp.

Các phương pháp phức tạp sẽ được trình bày ở cuối chương này trong phần Thiết kế Phương pháp Phức hợp .

Hình 9.1

Một dịch vụ thực thể dựa trên thực thể kinh doanh Hóa đơn xác định phạm vi chức năng giới hạn khả năng của dịch vụ trong việc thực hiện xử lý liên quan đến hóa đơn. Dịch vụ Hóa đơn bắt khai tri này sẽ được các dịch vụ khác trong cùng kho dịch vụ sử dụng lại và tổng hợp để hỗ trợ các quy trình kinh doanh tự động khác nhau cần xử lý dữ liệu liên quan đến hóa đơn. Hợp đồng dịch vụ hóa đơn cụ thể này hiển thị hai khả năng dịch vụ dựa trên các phương pháp nguyên thủy và hai khả năng dịch vụ dựa trên các phương pháp phức tạp.



MẪU SOA

Mẫu Liên kết thực thể [342] thường được áp dụng cho các dịch vụ thực thể dựa trên REST. Như được giải thích ở phần sau của chương này, các dịch vụ REST có thể xử lý dữ liệu được gửi lại bởi các lược đồ, chẳng hạn như các lược đồ được cung cấp bởi JSON và XML Schema specification. Với các dịch vụ thực thể nói riêng, điều này có thể nhấn mạnh rất nhiều về việc áp dụng nhất quán các mẫu Canonical Schema [326] và Schema Centralization [356].

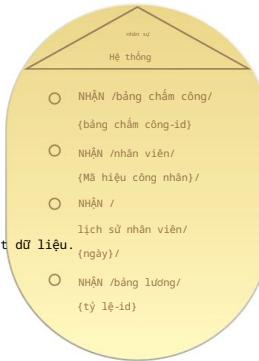
Thiết kế dịch vụ tiện ích

Giống như các dịch vụ thực thể, các dịch vụ tiện ích được mong đợi là có tính bắt khai tri và có thể tái sử dụng. Tuy nhiên, không giống như các dịch vụ thực thể, chúng thường không có phạm vi chức năng được xác định trước. Vì vậy, chúng tôi cần xác nhận rằng, trước khi hoàn tất hợp đồng dịch vụ, phương thức này và các kết hợp tài nguyên mà chúng tôi đã chọn trong giai đoạn phân tích hướng dịch vụ là những gì chúng tôi muốn cam kết đối với một thiết kế dịch vụ tiện ích nhất định.

Trong khi các dịch vụ tiện ích riêng lẻ nhóm các khả năng dịch vụ liên quan thì các dịch vụ rạn giời chức năng có thể thay đổi đáng kể. Ví dụ được minh họa trong Hình 9.2 là một dịch vụ tiện ích hoạt động như một trinh bao bọc cho một hệ thống cũ.

Hình 9.2

Hợp đồng dịch vụ tiện ích này bao gồm một hệ thống nhân sự kế thừa (và được đặt tên tương ứng). Các khả năng dịch vụ mà nó cung cấp cung cấp các chức năng truy cập dữ liệu chung, chỉ đọc đối với dữ liệu được lưu trữ trong kho lưu trữ kế thừa cơ bản. Ví dụ: dịch vụ thực thể Nhân viên (được tạo bởi dịch vụ nhiệm vụ Xác minh Bảng chấm công) có thể gọi ra khả năng dịch vụ liên quan đến dữ liệu của nhân viên để truy xuất dữ liệu. Loại dịch vụ tiện ích này có thể cung cấp quyền truy cập vào một trong một số nguồn dữ liệu có sẵn về nhân viên và liên quan đến nhân sự.



MẪU SOA

Các dịch vụ tiện ích có nhiều khả năng đảm bảo hỗ trợ cho các giao thức truyền thông thay thế, điều này làm cho việc áp dụng các mẫu Giao thức kép [339], Hợp đồng đồng thời [332] và Mặt tiền dịch vụ [360] có nhiều khả năng hơn so với thực thể dịch vụ. Một mẫu khác thường được áp dụng trong giai đoạn thiết kế hợp đồng dịch vụ tiện ích là Legacy Wrapper [347].

Thiết kế vi dịch vụ

Việc cản nhắc thiết kế chủ yếu áp dụng cho các hợp đồng vi dịch vụ là tính linh hoạt mà chúng tôi có trong cách tiếp cận thiết kế hợp đồng. Do thực tế là các dịch vụ vi mô thường dựa trên bối cảnh chức năng bắt khả tri có chủ ý, nên chúng sẽ thường có người tiêu dùng dịch vụ hạn chế. Đôi khi một microservice có thể chỉ có một người sử dụng dịch vụ duy nhất. Bởi vì chúng tôi già định rằng microservice sẽ không bao giờ cần tạo điều kiện thuận lợi cho bất kỳ người tiêu dùng dịch vụ nào khác trong tương lai (vì nó không được coi là có thể tái sử dụng bên ngoài quy trình kinh doanh), nên việc áp dụng một số định hướng dịch vụ nguyên tắc trở thành tùy chọn.

Đáng chú ý nhất, điều này bao gồm nguyên tắc Hợp đồng dịch vụ tiêu chuẩn hóa (291). Ở một mức độ nhất định, các API dịch vụ vi mô có thể không chuẩn để các khả năng riêng lẻ của chúng có thể được tối ưu hóa để hỗ trợ các yêu cầu về độ tin cậy và hiệu suất thời gian chạy.

Tính linh hoạt này tiếp tục được áp dụng cho việc áp dụng các nguyên tắc Trùu tượng dịch vụ (294) và Khớp nối lồng lèo dịch vụ (293).

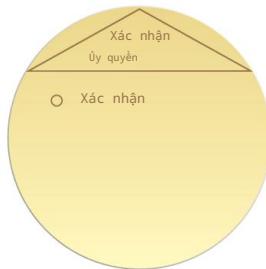
Các ngoại lệ đối với quyền tự do thiết kế này chủ yếu liên quan đến cách vi dịch vụ tương tác như một phần của thành phần dịch vụ lớn hơn. Chi phí để đạt được các yêu cầu về hiệu suất riêng lẻ của một vi dịch vụ cần phải được cân nhắc với các yêu cầu của giải pháp định hướng dịch vụ tổng thể mà nó là một phần trong đó.

Ví dụ, nguyên tắc Hợp đồng dịch vụ tiêu chuẩn hóa (291) có thể cần được áp dụng ở một mức độ nào đó để đảm bảo rằng hợp đồng vi dịch vụ được thiết kế để hỗ trợ lược đồ tiêu chuẩn đại diện cho một tài liệu kinh doanh chung. Việc cho phép vi dịch vụ giới thiệu một lược đồ không chuẩn có thể mang lại lợi ích cho hiệu quả xử lý của vi dịch vụ, nhưng dẫn đến các yêu cầu chuyển đổi dữ liệu để dữ liệu đó được chuyển đổi vào lược đồ tiêu chuẩn được sử dụng bởi các thành viên dịch vụ còn lại có thể không hợp lý.

Hình 9.3 thể hiện hợp đồng dịch vụ cho microservice được mô hình hóa trong Chương 6.

Hình 9.3

Một hợp đồng vi dịch vụ có phạm vi chức năng đơn mục đích, không xác định rõ ràng. Dịch vụ này cung cấp ba khả năng cụ thể và hỗ trợ quy trình kinh doanh chính của nó.



MẪU SOA

Ngoài các Giao thức kép [339], Hợp đồng đồng thời [332], Mặt tiền dịch vụ [360] và Legacy Wrapper [347], các vi dịch vụ dựa trên REST thường sẽ yêu cầu ứng dụng mẫu Triển khai vi dịch vụ [349] và có thể cả ứng dụng mẫu Containerization [333].

Có thể còn được yêu cầu thêm rằng các thành phần mà vi dịch vụ có thể yêu cầu quyền truy cập phải được sao chép hoặc triển khai dự phòng trong môi trường triển khai vi dịch vụ. Những loại yêu cầu này có thể được giải quyết bằng các mẫu triển khai như Sao chép dữ liệu dịch vụ [358], Triển khai dự phòng [354], và thậm chí cả Quyền tự chủ về bố cục [331], nếu cần.

Thiết kế dịch vụ nhiệm vụ

Các dịch vụ tác vụ thường có ít khả năng phục vụ, đôi khi chỉ giới hạn ở một người độc thân. Điều này là do thực tế là mục đích sử dụng chính của hợp đồng dịch vụ nhiệm vụ là dành cho thực hiện logic quy trình kinh doanh (hoặc nhiệm vụ) tự động. Khả năng dịch vụ có thể dựa trên một động từ đơn giản, chẳng hạn như **Bắt đầu** hoặc **Xử lý**. Động từ đó cùng với tên của dịch vụ tác vụ (điều đó sẽ cho biết bản chất của tác vụ) thường là tất cả những gì cần thiết cho các tác vụ đồng bộ. Các khả năng dịch vụ bổ sung có thể được thêm vào để hỗ trợ giao tiếp không đồng bộ, chẳng hạn như truy cập thông tin trạng thái hoặc hủy luồng công việc đang hoạt động dù, như thể hiện trong Hình 9.4.

Các dịch vụ tác vụ dựa trên REST thường sẽ có khả năng dịch vụ được kích hoạt bởi yêu cầu POST. Tuy nhiên, phương pháp này vốn không đáng tin cậy. Một số kỹ thuật tồn tại để đạt được POST đáng tin cậy, bao gồm cả việc đưa vào các tiêu đề bổ sung và xử lý của các thông báo phản hồi hoặc bao gồm một mã định danh yêu cầu duy nhất do người tiêu dùng tạo ra trong mã định danh tài nguyên.

Để cung cấp đầu vào cho một dịch vụ nhiệm vụ được tham số hóa, hợp đồng dịch vụ nhiệm vụ sẽ bao gồm nhiều số nhận dạng khác nhau vào mẫu nhận dạng tài nguyên của khả năng. (đó có thể là tham số trong thông báo SOAP). Điều này giải phóng dịch vụ để hiển thị các tài nguyên bổ sung thay vì xác định loại phương tiện tùy chỉnh làm đầu vào cho quá trình xử lý của nó.

Nếu dịch vụ tác vụ tự động hóa một quy trình kinh doanh dài hạn, nó sẽ trả về phản hồi tạm thời cho người tiêu dùng trong khi vẫn có thể cần phải thực hiện các bước xử lý tiếp theo. Nếu dịch vụ tác vụ bao gồm các khả năng bổ sung để kiểm tra hoặc tương tác với trạng thái của quy trình công việc (hoặc phiên bản tổng hợp), thì nó thường sẽ bao gồm một siêu liên kết đến một hoặc nhiều tài nguyên liên quan đến trạng thái này trong thông báo phản hồi ban đầu.

Hình 9.4

Một dịch vụ nhiệm vụ mẫu, có thể nhận biết bằng động từ trong tên của nó. Hợp đồng chỉ cung cấp khả năng dịch vụ được người khởi tạo thành phần sử dụng để kích hoạt việc thực thi quy trình công việc Xác thực Bằng chấm công mà logic dịch vụ tác vụ gói gọn. Trong trường hợp này, khả năng dịch vụ nhận được mã định danh tài nguyên bằng chấm công sẽ được sử dụng làm cơ sở logic xác thực, cộng với mã định danh yêu cầu duy nhất do người tiêu dùng tạo hỗ trợ kích hoạt quy trình một cách đáng tin cậy. Hai khả năng dịch vụ bổ sung cho phép người tiêu dùng kiểm tra không đồng bộ tiến trình của nhiệm vụ xác thực bằng chấm công và hủy nhiệm vụ khi nó đang được thực hiện.



VÍ DỤ NGHIÊN CỨU TRƯỜNG HỢP

MUA tuân theo các kỹ thuật thiết kế hợp đồng dịch vụ REST đã được chứng minh cùng với các tiêu chuẩn thiết kế tùy chỉnh được thiết lập cụ thể cho doanh nghiệp MUA. Kiến trúc sử dụng các ứng viên dịch vụ chọn lọc được mô hình hóa trong Chương 7 làm cơ sở cho thiết kế hợp đồng dịch vụ của họ.

Trao Hợp đồng dịch vụ giải thưởng sinh viên (Nhiệm vụ)

Học sinh nộp đơn đăng ký trao giải thưởng sẽ thực hiện việc này thông qua trình duyệt Web. Do đó, một giao diện người dùng riêng biệt được thiết kế để cho phép người dùng nhập thông tin chi tiết về ứng dụng. Việc gửi biểu mẫu dựa trên trình duyệt này sẽ khởi tạo dịch vụ tác vụ.

Khi nhận được nội dung gửi, tập lệnh phía máy chủ sẽ sắp xếp dữ liệu biểu mẫu thành một Tài liệu XML dựa trên loại phương tiện sau:

application/vnd.edu.mua.student-award-conferral-application/xhtml+xml

Ví dụ 9.1 cung cấp một mẫu đơn đăng ký được hoàn thiện với dữ liệu mẫu được thu thập từ người dùng. Điều này thể hiện tập dữ liệu khởi động và thúc đẩy việc thực thi toàn bộ phiên bản của quy trình kinh doanh Giải thưởng Sinh viên Confer.

```
<?xml version="1.0" mã hóa="UTF-8"?>
<!DOCTYPE html CÔNG KHAI "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="vi" >
<đầu>
    <title>Đơn xin trao giải thưởng sinh viên</title>
</head>
<cơ thể>
    <p>Sinh viên:<br/>
        <a rel="sinh viên"
            href="http://student.mua.edu/student/555333">
            John Smith (Mã số sinh viên 555333)
        </a>
    </p>
    <p>Giải thưởng:<br/>
        <a rel="giải thưởng"
            href="http://award.mua.edu/award/BS/CompSci">
            Cử nhân Khoa học với chuyên ngành Khoa học Máy tính
        </a>
    </p>
</body>
```

9.1 Những cản nhắc về thiết kế mô hình dịch vụ

```

<p>Sự kiện:
<a rel="sự kiện"
  href="http://event.mua.edu/achievement">
  Thành tựu nổi bật
</a>
</p>
</body>
</html>

```

Ví dụ 9.1

Dữ liệu ứng dụng mẫu, được gửi tới máy chủ Web. Cấu trúc tài liệu này chứa cả thông tin con người có thể đọc được và thông tin có thể xử lý bằng máy.

Hình 9.5 hiển thị hợp đồng dịch vụ Giải thưởng Sinh viên Confer.

Loại phương tiện trước đây được thiết kế có chủ ý để bao gồm dữ liệu mà con người có thể đọc được và máy có thể đọc được ở dạng phù hợp cho việc lưu trữ lâu dài.

Tài liệu được gửi tới một khả năng dịch vụ tương ứng trực tiếp với khả năng Bắt đầu được xác định trong ứng cử viên dịch vụ Giải thưởng Sinh viên Confer.

Như được thể hiện trong Hình 9.5, trong quá trình thiết kế đối với hợp đồng dịch vụ này, người ta đã quyết định bổ sung thêm dịch vụ mới phó khả năng cung cấp các chức năng sau:

- **DELETE /task/{id}** - Khả năng này đã được thêm vào cho phép một phiên bản thực thi của Confer Sinh viên Giải thưởng quá trình kinh doanh được chấm dứt.
- **GET /task/{id}** - Khả năng này cho phép trạng thái của một phiên bản đang thực thi của Trao giải thưởng sinh viên cho quy trình kinh doanh được truy vấn.

Lưu ý rằng tính chất nhạy cảm của loại ứng dụng này có nghĩa là lệnh **GET /task/{id}** chỉ có thể được truy cập bởi nhân viên được ủy quyền và sinh viên. Chỉ học sinh mới có thể truy cập khả năng **DELETE /task/{id}** để hủy quá trình đăng ký.



Hình 9.5

Hợp đồng dịch vụ Giải thưởng Sinh viên Confer.

228 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

Hợp đồng dịch vụ sự kiện (Thực thể)

Dịch vụ thực thể Sự kiện được trang bị khả năng dịch vụ GET /event/{id}, được sử dụng để truy vấn thông tin sự kiện và tương ứng với ứng viên có khả năng Nhận thông tin chi tiết từ ứng viên dịch vụ Sự kiện (Hình 9.6).

Trong quá trình thiết kế hướng dịch vụ, kiến trúc sư đã quyết định bổ sung thêm các khả năng GET /event/{id}/calendar và GET /event/{id}/description cho phép truy xuất thông tin sự kiện cụ thể hơn. Những khả năng này không được bổ sung cụ thể để hỗ trợ quy trình kinh doanh Giải thưởng Sinh viên Confer, mà hơn thế nữa là để cung cấp một phạm vi rộng hơn của chức năng có thể tái sử dụng dự kiến.

Hợp đồng dịch vụ trao thưởng (Thực thể)

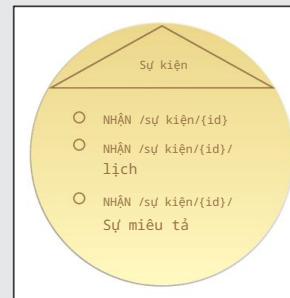
Ngoài việc triển khai ba khả năng dịch vụ từ ứng viên dịch vụ Giải thưởng ban đầu (Hình 9.6), một số kiến trúc sư SOA của MUA quyết định tạo ra một số có những thay đổi.

Trở lại Chương 7, các nhà phân tích MUA đã xác định rằng hành động sau đây cần được thực hiện được bao gồm trong logic dịch vụ nhiệm vụ của Giải thưởng Sinh viên Confer:

- Xác minh Bảng điểm Học sinh Đủ điều kiện để nhận Giải thưởng dựa trên Quy tắc Trao Giải thưởng

Tuy nhiên, với các quy tắc cụ thể cho từng loại giải thưởng, họ xác định rằng phải là dịch vụ của thực thể Giải thưởng áp dụng phần lớn các quy tắc này. Tuy nhiên, cần phải áp dụng một số kiểm tra chung để logic được phân chia giữa dịch vụ nhiệm vụ Confer Sinh viên và dịch vụ thực thể Giải thưởng.

Để tránh yêu cầu dịch vụ nhiệm vụ chuyển chi tiết bản ghi đầy đủ vào dịch vụ thực thể Giải thưởng để xác minh, người ta quyết định sử dụng phương pháp mã theo yêu cầu. Các Dịch vụ thực thể giải thưởng cung cấp logic nhưng logic được thực thi bởi dịch vụ nhiệm vụ. Quyết định xác định logic tập trung trong dịch vụ thực thể Giải thưởng được điều chỉnh dựa trên nhu cầu tạo ra đầu ra mà con người có thể đọc được (dành cho sinh viên), cùng với đầu ra có thể đọc được bằng máy (đối với dịch vụ Giải thưởng dành cho sinh viên Confer). Kết quả là, Dịch vụ Thực thể cung cấp khả năng dịch vụ GET /award/conferral-rules mới (Hình 9.7) hỗ trợ đầu ra của hai định dạng cho logic quy tắc: định dạng đầu tiên trong



Hình 9.6

Hợp đồng dịch vụ sự kiện.

dạng con người có thể đọc được và dạng thứ hai ở dạng có thể dễ dàng nhúng vào logic của dịch vụ tác vụ.

Các kiến trúc sư MUA chọn JavaScript cho mục đích này vì họ nhận thấy rằng thời gian chạy JavaScript sẵn có cho nhiều nền tảng công nghệ đã được sử dụng để phát triển các dịch vụ trong kho. Việc chọn JavaScript thay vì các công nghệ khác cũng khiến nó trở thành ngôn ngữ được lựa chọn cho cấp giao diện người dùng của kho dịch vụ.

Khả năng dịch vụ tương tự có thể trả về các quy tắc trao đổi trong JavaScript hoặc dưới dạng HTML mà con người có thể đọc được. Quyết định thực hiện chuyển đổi nào phụ thuộc vào tiêu đề Chấp nhận nào được người tiêu dùng dịch vụ cung cấp. Ví dụ:

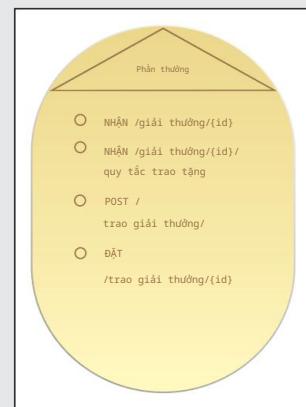
nhiệm vụ Trao giải thưởng Sinh viên

dịch vụ yêu cầu loại phương tiện ứng dụng/javascript , trong khi người tiêu dùng dịch vụ yêu cầu đầu ra mà con người có thể đọc được sẽ yêu cầu loại phương tiện văn bản/html .

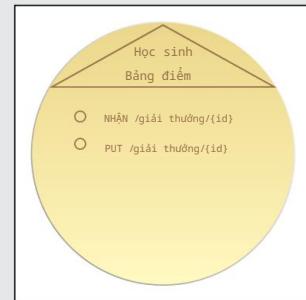
Hợp đồng dịch vụ bảng điểm sinh viên (Thực thể)

Dịch vụ Sinh viên ban đầu được dự định là một dịch vụ thực thể tập trung sẽ bao gồm tất cả các chức năng và quyền truy cập dữ liệu liên quan đến sinh viên. Tuy nhiên, các lần lặp lại quy trình lập mô hình dịch vụ REST xảy ra sau các ví dụ được đề cập trong Chương 7 đã dẫn đến một kế hoạch chi tiết kiểm kê dịch vụ cho thấy ứng viên dịch vụ Sinh viên có mức độ chi tiết cao hơn nhiều so với bất kỳ ứng viên nào khác. Điều này chủ yếu là do sự phức tạp của thực thể Sinh viên và các mối quan hệ của nó với các thực thể liên quan khác.

Sau khi xem xét ứng viên Dịch vụ Sinh viên, nó đã được quyết tâm tạo ra một tập hợp các dịch vụ thực thể liên quan đến sinh viên. Một trong những biến thể chuyên biệt hơn này đã trở thành ứng viên dịch vụ Bảng điểm Sinh viên (Hình 9.8).



Hình 9.7
Hợp đồng dịch vụ giải thưởng.



Hình 9.8
Ứng viên dịch vụ Bảng điểm Sinh viên đã được xác định sau ứng viên Dịch vụ Sinh viên. Dịch vụ này thay thế hiệu quả dịch vụ Sinh viên trong thành phần dịch vụ Giải thưởng Sinh viên Confer.

Bởi vì quy trình kinh doanh Trao Giải thưởng Sinh viên chỉ yêu cầu quyền truy cập vào thông tin bảng điểm của sinh viên nên nó chỉ cần soạn thảo dịch vụ Bảng điểm sinh viên chứ không phải dịch vụ sinh viên thực tế. Như được hiển thị trong Hình 9.9, dịch vụ Bảng điểm Sinh viên chứa các khả năng dịch vụ tương ứng với các ứng viên năng lực dịch vụ được cung cấp bởi ứng viên dịch vụ Bảng điểm Sinh viên.

Hợp đồng dịch vụ thông báo và tài liệu (Tiện ích)

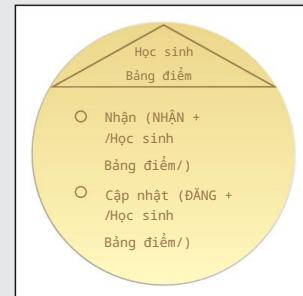
Dịch vụ Thông báo và Dịch vụ tài liệu xử lý dữ liệu con người có thể đọc được tương tự nhau. Các thông báo được gửi qua email hoặc bản in đều có thể được mã hóa thành định dạng tài liệu mà con người có thể đọc được, chẳng hạn như HTML hoặc PDF.

Dịch vụ Thông báo được giữ lại cho các thông báo qua email trong khi dịch vụ Tài liệu đã được phát triển thành dịch vụ tiện ích lấy máy in làm trung tâm và chuyển phát bưu điện. Dịch vụ nhiệm vụ Giải thưởng Sinh viên Confer có thể gửi tài liệu cho sinh viên ở định dạng ưu tiên bằng cách tra cứu phương thức gửi ưu tiên trong mẫu đơn đăng ký ban đầu.

Như được hiển thị trong Hình 9.10, các dịch vụ Thông báo và Tài liệu đều có thể được gọi bằng phương thức POST.

Hình 9.10

Hợp đồng dịch vụ Thông báo và Tài liệu.

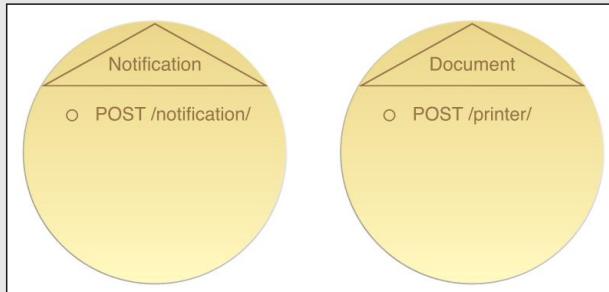


Hình 9.9

Hợp đồng dịch vụ Bảng điểm Sinh viên.

Hình 9.10

Hợp đồng dịch vụ Thông báo và Tài liệu.



Sinh viên mẫu (John Smith) từ mẫu đơn đăng ký được sử dụng làm thông tin đầu vào cho dịch vụ nhiệm vụ Giải thưởng Sinh viên Confer đã chỉ định tùy chọn liên hệ của mình bằng siêu liên kết tới mailto:s555333@student.mua.edu. Tiêu chuẩn kiểm kê dịch vụ để xử lý một địa chỉ như vậy là chuyển đổi URL thành <http://notification.mua.edu/sender?to=s555333@student.mua.edu> và sử dụng phương thức POST để phân phối. Thông báo của John Smith sẽ được gửi qua email tới địa chỉ này.

9.2 Nguyên tắc thiết kế dịch vụ REST

Sau đây là một loạt các hướng dẫn và cân nhắc chung khi thiết kế hợp đồng dịch vụ REST.

Những cân nhắc về thiết kế hợp đồng thống nhất

Khi tạo một hợp đồng thống nhất về kho dịch vụ, chúng tôi có trách nhiệm trang bị và giới hạn các tính năng của nó để nó được sắp xếp hợp lý nhằm đáp ứng hiệu quả các yêu cầu và hạn chế duy nhất đối với kho dịch vụ. Các đặc điểm mặc định của kiến trúc công nghệ lấy Web làm trung tâm có thể cung cấp cơ sở hiệu quả cho hợp đồng thống nhất của kho dịch vụ, mặc dù các hình thức tiêu chuẩn hóa và tùy chỉnh bổ sung có thể được yêu cầu đối với các kiến trúc kiểm kê dịch vụ không tầm thường.

Các phần sau đây khám phá các yếu tố chung của một hợp đồng thống nhất (phương pháp, loại phương tiện truyền thông và các trường hợp ngoại lệ nói riêng) có thể được tùy chỉnh để đáp ứng nhu cầu kiểm kê dịch vụ cá nhân.

Phương pháp thiết kế và tiêu chuẩn hóa

Khi chúng ta thảo luận về các phương pháp liên quan đến hợp đồng thống nhất, nó được coi là cách viết tắt của cơ chế liên lạc phản hồi yêu cầu cũng bao gồm các phương thức, tiêu đề, mã phản hồi và ngoại lệ. Các phương pháp được tập trung hóa như một phần của hợp đồng thống nhất để đảm bảo rằng luôn có một số cách nhỏ để di chuyển thông tin trong kho dịch vụ cụ thể và người tiêu dùng dịch vụ hiện tại sẽ làm việc chính xác với các dịch vụ mới hoặc được sửa đổi khi chúng được thêm vào kho. Mặc dù điều quan trọng là phải giảm thiểu số lượng các phương pháp trong hợp đồng thống nhất, nhưng các phương pháp có thể và nên được thêm vào khi yêu cầu tương tác về kho dịch vụ yêu cầu. Đây là một phần tự nhiên của việc phát triển kho dịch vụ để đáp ứng với sự thay đổi của doanh nghiệp.

GHI CHÚ

Các phương thức HTTP ít được biết đến hơn đã xuất hiện và biến mất trong quá khứ. Ví dụ: tại nhiều thời điểm, đặc tả HTTP đã bao gồm phương thức PATCH phù hợp với cơ chế truyền thông cập nhật một phần hoặc lưu trữ một phần. PATCH hiện được chỉ định riêng biệt với các phương thức HTTP trong tài liệu RFC 5789 của IETF. Các thông số kỹ thuật khác của IETF, chẳng hạn như RFC 4918 của WebDAV và RFC 3261 của Giao thức khởi tạo phiên, đã giới thiệu các phương pháp mới cũng như các tiêu đề và mã phản hồi mới (hoặc các cách diễn giải đặc biệt về chúng).

HTTP cung cấp nền tảng vững chắc bằng cách cung cấp tập hợp các phương thức cơ bản (chẳng hạn như GET, PUT, DELETE, POST) đã được chứng minh bằng cách sử dụng trên Web và được hỗ trợ rộng rãi bởi những người có sẵn thành phần phần mềm và thiết bị phần cứng. Nhưng có thể này sinh nhu cầu thể hiện các loại tương tác khác cho kho dịch vụ. Ví dụ: bạn có thể quyết định thêm một phương thức đặc biệt có thể được sử dụng để kích hoạt tài nguyên một cách đáng tin cậy để thực thi tác vụ nhiều nhất một lần, thay vì sử dụng phương thức HTTP POST kèm tin cậy hơn.

HTTP được thiết kế để mở rộng theo những cách này. Đặc tả HTTP hỗ trợ rõ ràng khái niệm về các phương thức mở rộng, các tiêu đề tùy chỉnh và khả năng mở rộng trong các lĩnh vực khác.

khu vực. Việc tận dụng tính năng này của HTTP có thể có hiệu quả, miễn là các tiện ích mở rộng mới được thêm cần thận và ở mức phù hợp với số lượng dịch vụ triển khai HTTP trong kho. Bằng cách này, tổng số tùy chọn để di chuyển dữ liệu

xung quanh (các dịch vụ và người tiêu dùng bắt buộc phải hiểu) vẫn có thể quản lý được.

GHI CHÚ

Ở cuối chương này, chúng ta khám phá một tập hợp các phương pháp mở rộng mẫu (được gọi là các phương pháp phức tạp). Mỗi phương thức sử dụng nhiều phương thức HTTP cơ bản hoặc một phương thức HTTP cơ bản nhiều lần để thực hiện các tương tác tin nhắn được tiêu chuẩn hóa, được xác định trước.

Các trường hợp phổ biến có thể đảm bảo việc tạo ra các phương pháp mới bao gồm:

- Các siêu liên kết có thể được sử dụng để tạo thuận lợi cho chuỗi các cặp yêu cầu-phản hồi. Khi họ bắt đầu đọc giống như động từ thay vì danh từ và có xu hướng gợi ý rằng chỉ có một sê hợp lệ trên mục tiêu của siêu liên kết, chúng ta có thể xem xét việc giới thiệu thay vào đó là một phương pháp mới. Ví dụ: siêu liên kết “khách hàng” cho hóa đơn tài nguyên gợi ý rằng các yêu cầu GET và PUT có giá trị như nhau đối với nguồn lực khách hàng. Nhưng siêu liên kết “bắt đầu giao dịch” hoặc siêu liên kết “đăng ký” chỉ gợi ý POST là hợp lệ và có thể cho biết cần có một phương thức mới thay thế.
- Dữ liệu có ngữ nghĩa phải hiểu có thể cần thiết trong tiêu đề thư. TRONG trong trường hợp này, dịch vụ bỏ qua siêu dữ liệu này có thể gây ra hoạt động thời gian chạy không chính xác. HTTP không bao gồm phương tiện để xác định các tiêu đề riêng lẻ hoặc thông tin trong các tiêu đề là “phải hiểu”. Một phương pháp mới có thể được sử dụng để thực thi yêu cầu này vì phương thức tùy chỉnh sẽ tự động bị từ chối bởi một dịch vụ không hiểu yêu cầu (trong khi quay lại mặc định Phương thức HTTP sẽ cho phép dịch vụ bỏ qua thông tin tiêu đề mới).

Điều quan trọng là phải thừa nhận rằng việc giới thiệu các phương pháp tùy chỉnh có thể có tác động tiêu cực khi khám phá sự đa dạng của nhà cung cấp trong môi trường triển khai. Nó có thể ngăn chặn các thành phần sẵn có (chẳng hạn như bộ đệm, bộ cân bằng tải, tường lửa và các khung phần mềm dựa trên HTTP khác nhau) để có đầy đủ chức năng trong kho dịch vụ. Chỉ nên thử loại bỏ HTTP và các phương thức mặc định của nó trong kho dịch vụ hoàn thiện khi đã hiểu đầy đủ các tác động lên kiến trúc và cơ sở hạ tầng công nghệ cơ bản.

Một số lựa chọn thay thế để tạo ra các phương pháp mới cũng có thể được khám phá. Ví dụ, dịch vụ các tương tác yêu cầu một số bước có thể sử dụng siêu liên kết để hướng dẫn người tiêu dùng thực hiện các yêu cầu mà họ cần thực hiện. Tiêu đề Liên kết HTTP (RFC 5988) có thể được xem xét để tách các siêu liên kết này khỏi nội dung tài liệu thực tế.

MẪU SOA

Làm việc và tùy chỉnh giao diện thống nhất liên quan đến ứng dụng tự nhiên của mẫu Hợp đồng tái sử dụng [355].

Thiết kế và chuẩn hóa tiêu đề HTTP

Trao đổi tin nhắn bằng siêu dữ liệu là điều phổ biến trong thiết kế giải pháp hướng dịch vụ. Do nhấn mạnh vào việc kết hợp một tập hợp các dịch vụ lại với nhau để tự động kết hợp chung một tác vụ nhất định trong thời gian chạy, nên thường cần có một thông báo để cung cấp một loạt các Thông tin tiêu đề liên quan đến cách xử lý thông báo bởi các đại lý dịch vụ nhật ký trung gian và các dịch vụ đọc theo đường dẫn thông báo của nó.

Tiêu đề HTTP tích hợp có thể được sử dụng theo một số cách:

- Để thêm các tham số liên quan đến phương thức yêu cầu thay thế cho việc sử dụng truy vấn chuỗi để biểu thị các tham số trong URL. Ví dụ: Chấp nhận tiêu đề có thể bổ sung phương thức GET bằng cách cung cấp dữ liệu kèm phán nội dung.
- Để thêm các tham số liên quan đến mã phản hồi. Ví dụ: tiêu đề Vị trí có thể được sử dụng với mã phản hồi 201 Created để cho biết mã định danh của một tài nguyên mới được tạo ra.
- Để truyền đạt thông tin chung về dịch vụ hoặc người tiêu dùng. Ví dụ, tiêu đề Nâng cấp có thể chỉ ra rằng người tiêu dùng dịch vụ hỗ trợ và thích một giao thức khác nhau, trong khi tiêu đề Người giới thiệu có thể cho biết tài nguyên nào người tiêu dùng đến từ khi theo dõi một loạt siêu liên kết.

234 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

Loại siêu dữ liệu chung này có thể được sử dụng cùng với bất kỳ phương thức HTTP nào.

Tiêu đề HTTP cũng có thể được sử dụng để thêm siêu dữ liệu phong phú. Vì mục đích này, các tiêu đề tùy chỉnh thường được yêu cầu, điều này dẫn đến nhu cầu xác định xem có hay không nội dung tin nhắn phải được người nhận hiểu hoặc có thể tùy ý bỏ qua nội dung đó. Sự kết hợp giữa ngữ nghĩa phải hiểu với các phương thức mới và ngữ nghĩa phải bỏ qua với các tiêu đề thư mới không phải là một tính năng vốn có của REST, nhưng nó là một tính năng của HTTP.

Khi giới thiệu các tiêu đề HTTP tùy chỉnh mà các dịch vụ có thể bỏ qua, HTTP thông thường phương pháp có thể được sử dụng một cách an toàn. Điều này cũng làm cho việc sử dụng tiêu đề tùy chỉnh tương thích ngược khi tạo phiên bản mới của loại thông báo hiện có.

Như đã nêu trước đây trong phần Phương pháp thiết kế và tiêu chuẩn hóa , HTTP mới các phương thức có thể được giới thiệu để thực thi nội dung phải hiểu bằng cách yêu cầu các dịch vụ được thiết kế để hỗ trợ phương thức tùy chỉnh hoặc từ chối hoàn toàn nỗ lực gọi phương thức. Để hỗ trợ hành vi này, tiêu đề Phải hiểu mới có thể được tạo ở cùng định dạng với tiêu đề Kết nối hiện có , tiêu đề này sẽ liệt kê tất cả tiêu đề mà người nhận tin nhắn cần hiểu được.

Nếu loại sửa đổi này được thực hiện đối với HTTP thì đó sẽ là trách nhiệm của SOA Văn phòng Chương trình Quản trị chịu trách nhiệm về việc kiểm kê dịch vụ để đảm bảo rằng các ngữ nghĩa này được triển khai nhất quán như một phần của tiêu chuẩn thiết kế trên toàn bộ kiểm kê. Nếu các tiêu đề HTTP tùy chỉnh, phải hiểu được thiết lập thành công trong một dịch vụ khoảng không quảng cáo, chúng ta có thể khám phá một loạt ứng dụng của siêu dữ liệu nhắn tin. Ví dụ, chúng tôi có thể xác định liệu có thể hoặc khả thi khi mô phỏng siêu dữ liệu nhắn tin, chẳng hạn như siêu dữ liệu thường được sử dụng trong khung nhắn tin SOAP dựa trên tiêu chuẩn WS-* hay không.

Trong khi các tiêu đề tùy chỉnh thực thi độ tin cậy hoặc định tuyến nội dung (theo tiêu chuẩn WS-ReliableMessaging và WS-Addressing) có thể được thêm vào để tạo lại các tương tác xác nhận và cân bằng tải thông minh, các dạng WS-* khác các chức năng phải tuân theo các giới hạn tích hợp của giao thức HTTP. Ví dụ nổi bật nhất là việc sử dụng WS-Security để kích hoạt các tính năng bảo mật ở cấp độ thông báo, chẳng hạn như mã hóa và chữ ký số. Bảo mật cấp độ tin nhắn bảo vệ tin nhắn bằng cách thực sự chuyển đổi nội dung để những người trung gian đọc theo đường dẫn tin nhắn không thể đọc hoặc thay đổi nội dung tin nhắn. Chỉ những người nhận tin nhắn được ủy quyền trước mới có thể truy cập nội dung.

Kiểu chuyển đổi thông báo này không được hỗ trợ trong HTTP/1.1. HTTP có một số tính năng cơ bản để chuyển đổi nội dung của thông báo thông qua

Tiêu đề mã hóa nội dung , nhưng điều này thường bị giới hạn ở việc nén tin nhắn phần thân và không bao gồm việc chuyển đổi các tiêu đề. Nếu tính năng này được sử dụng cho mục đích mã hóa thì ý nghĩa của tin nhắn vẫn có thể được sửa đổi hoặc kiểm tra trong quá trình truyền, mặc dù phần nội dung của tin nhắn có thể được bảo vệ. Bản chất thông báo cũng không thể có trong HTTP/1.1 vì không có dạng chuẩn nào cho thông báo HTTP ký và không có tiêu chuẩn ngành nào xác định những sửa đổi nào mà các nhật ký trung gian sẽ được phép thực hiện đối với một thông báo như vậy.

Thiết kế và chuẩn hóa mã phản hồi HTTP

HTTP ban đầu được thiết kế như một giao thức máy khách-máy chủ đồng bộ để trao đổi các trang HTML trên World Wide Web. Các đặc điểm này tương thích với các ràng buộc REST và làm cho nó cũng phù hợp như một giao thức được sử dụng để gọi các khả năng của dịch vụ REST.

Phát triển một dịch vụ sử dụng HTTP rất giống với việc xuất bản nội dung động trên máy chủ Web. Mỗi yêu cầu HTTP gọi ra một khả năng dịch vụ REST và lệnh gọi đó kết thúc bằng việc gửi thông báo phản hồi lại cho người sử dụng dịch vụ.

Một thông báo phản hồi nhất định có thể chứa bất kỳ một trong số nhiều loại mã HTTP, mỗi mã trong đó có một số được chỉ định. Một số dãy mã số nhất định được liên kết với các loại điều kiện cụ thể như sau:

- 100-199 là các mã thông tin được sử dụng làm cơ chế báo hiệu mức thấp, chẳng hạn như xác nhận yêu cầu thay đổi giao thức.
- 200-299 là mã thành công chung dùng để mô tả các loại thành công khác nhau điều kiện.
- 300-399 là mã chuyển hướng được sử dụng để yêu cầu người tiêu dùng thử lại yêu cầu một mã định danh tài nguyên khác hoặc thông qua một trung gian khác.
- 400-499 đại diện cho mã lỗi phía người tiêu dùng cho biết rằng người tiêu dùng có đưa ra một yêu cầu không hợp lệ vì lý do nào đó.
- 500-599 đại diện cho mã lỗi phía dịch vụ cho biết rằng người tiêu dùng yêu cầu có thể hợp lệ nhưng dịch vụ đó không thể xử lý nó trong lý do nội tại.

Các danh mục ngoại lệ phía người tiêu dùng và phía dịch vụ rất hữu ích cho việc “gắn đỗi lối” nhưng lại không thực sự giúp người tiêu dùng dịch vụ phục hồi sau thất bại. Đây là bởi vì, trong khi các mã và lý do do HTTP cung cấp được tiêu chuẩn hóa, thì dịch vụ

người tiêu dùng không được yêu cầu phải cung cấp khi nhận được mã phản hồi. Khi chuẩn hóa thiết kế dịch vụ cho kho dịch vụ, cần thiết lập một bộ quy ước gán ý nghĩa và cách xử lý cụ thể cho các mã phản hồi.

Bảng 9.1 cung cấp các mô tả chung về cách người tiêu dùng dịch vụ có thể được thiết kế để đáp ứng các mã phản hồi chung.

Phản ứng Mã số	Cụm từ lý do	Sự đối đãi
100	Tiếp tục	không xác định
101	Chuyển đổi giao thức	không xác định
1xx	Bất kỳ mã 1xx nào khác	Sự thất bại
200	Được rồi	Thành công
201	Tạo	Thành công
202	Đã được chấp nhận	Thành công
203	Thông tin không có thẩm quyền thành công	
204	Không có nội dung	Thành công
205	Đặt lại nội dung	Thành công
206	Nội dung một phần	Thành công
2xx	Bất kỳ mã 2xx nào khác	Thành công
300	nhiều lựa chọn	Sự thất bại
301	Đã di chuyển vĩnh viễn	không xác định (Hành vi phổ biến: Sửa đổi mã nhận dạng tài nguyên và thử lại.)

Phản ứng Mã số	Cụm từ lý do	Sự đổi dâí
302	Thành lập	không xác định
303	Xem Khác	(Hành vi phổ biến: Thay đổi yêu cầu thành GET và thử lại bằng cách sử dụng mã định danh tài nguyên được chỉ định.)
304	Không được sửa đổi	Thành công (Hành vi phổ biến: Tải lại phản hồi đã lưu trong bộ nhớ đệm cho người tiêu dùng.)
305	Sử dụng proxy	không xác định (Hành vi phổ biến: Kết nối với proxy nhận dạng và gửi lại tin nhắn gốc.)
307	Chuyển hướng tạm thời	không xác định (Hành vi phổ biến: Thử lại một lần với mã nhận dạng tài nguyên được chỉ định.)
3xx	Bất kỳ mã 3xx nào khác	sự thất bại
400	Yêu cầu không hợp lệ	sự thất bại
401	Không được phép	không xác định (Hành vi phổ biến: Thử lại với thông tin xác thực chính xác.)
402	yêu cầu thanh toán	sự thất bại
403	Cấm	sự thất bại
404	Không tìm thấy	Thành công nếu yêu cầu bị XÓA, nếu không sự thất bại
405	Phương pháp không được phép	sự thất bại
406	Không thể chấp nhận	sự thất bại

Phản Ứng Mã số	Cụm từ lý do	Sự đói dãi
407	Xác thực proxy Yêu cầu	không xác định (Hành vi phổ biến: Thủ lại với thông tin xác thực chính xác.)
408	Hết thời gian yêu cầu	Sự thất bại
409	Xung đột	Sự thất bại
410	Đi mất	Thành công nếu yêu cầu bị XÓA, khác Thất bại
411	Độ dài yêu cầu	Sự thất bại
412	Điều kiện tiên quyết không thành công	Sự thất bại
413	Thực thể yêu cầu quá lớn	Sự thất bại
414	URI yêu cầu quá dài	Sự thất bại
415	Loại phương tiện không được hỗ trợ	Sự thất bại
416	Phạm vi yêu cầu Không Có thể thỏa mãn	Sự thất bại
417	Kỳ vọng không thành công	Sự thất bại
4xx	Bất kỳ mã 4xx nào khác	Sự thất bại
500	Lỗi máy chủ nội bộ	Sự thất bại
501	Không được thực hiện	Sự thất bại
502	Công xấu	Sự thất bại
503	dịch vụ Không sẵn có	Lặp lại nếu tiêu đề Retry-After được chỉ định. Ngược lại, Thất bại.
504	Công Time-out	Lặp lại nếu yêu cầu là bình thường. Ngược lại, Thất bại.

Phản ứng Mã số	Cụm từ lý do	Sự đối đãi
505	Phiên bản HTTP không được hỗ trợ	Sự thất bại
5xx	Bất kỳ mã 5xx nào khác	Sự thất bại

Bảng 9.1

Mã phản hồi HTTP và hành vi tiêu dùng tương ứng điển hình.

Như được thấy rõ khi xem xét Bảng 9.1, các mã phản hồi HTTP vượt xa sự phân biệt đơn giản giữa thành công và thất bại. Chúng cung cấp dấu hiệu về cách người tiêu dùng có thể phản ứng và phục hồi sau các trường hợp ngoại lệ.

Chúng ta hãy xem xét kỹ hơn một số giá trị từ cột Điều trị trong Bảng 9.1:

- Lặp lại có nghĩa là người tiêu dùng được khuyến khích lặp lại yêu cầu, cân nhắc tính đến bất kỳ sự chậm trễ nào được chỉ định trong các phản hồi, chẳng hạn như 503 Dịch vụ không khả dụng. Cái này có thể có nghĩa là ngủ trước khi thử lại. Nếu người tiêu dùng chọn không lặp lại request, nó phải coi phương thức đó là thất bại.
- Thành công có nghĩa là người tiêu dùng nên coi việc truyền tải thông điệp là một hành động thành công và do đó không được lặp lại hành động đó. (Lưu ý rằng các mã thành công cụ thể có thể yêu cầu giải thích tinh tế hơn.)
- Không thành công có nghĩa là người tiêu dùng không được lặp lại yêu cầu không thay đổi, mặc dù nó có thể đưa ra một yêu cầu mới có tính đến phản hồi. Người tiêu dùng nên coi đây là phương pháp thất bại nếu không thể tạo yêu cầu mới. (Ghi chú mã lỗi cụ thể đó có thể yêu cầu diễn giải tinh tế hơn.)
- Không xác định có nghĩa là người tiêu dùng cần sửa đổi yêu cầu của mình theo cách thức chỉ ra. Yêu cầu không được lặp lại không thay đổi và yêu cầu mới tính đến phản hồi sẽ được tạo ra. Kết quả cuối cùng của sự tương tác sẽ phụ thuộc vào yêu cầu mới. Nếu người tiêu dùng không thể tạo ra một yêu cầu mới thì mã này phải được coi là không thành công.

Bởi vì HTTP là một giao thức chữ không phải là một tập hợp logic xử lý tin nhắn nên nó tùy thuộc vào service để quyết định mã trạng thái nào (thành công, thất bại hoặc cách khác) sẽ trả về. Như đã đề cập trước đây, vì hành vi của người tiêu dùng không phải lúc nào cũng được chuẩn hóa đầy đủ bởi REST cho các tương tác giữa máy với máy, nó cần phải rõ ràng và có ý nghĩa được chuẩn hóa như một phần của dự án SOA.

240 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

Ví dụ: các mã không xác định có xu hướng chỉ ra rằng người tiêu dùng dịch vụ phải xử lý một tình huống sử dụng logic tùy chỉnh của riêng họ. Chúng ta có thể chuẩn hóa các loại mã này theo hai cách:

- Tiêu chuẩn thiết kế có thể xác định những mã không xác định nào có thể và không thể được được phát hành bởi logic dịch vụ.
- Các tiêu chuẩn thiết kế có thể xác định logic của người tiêu dùng dịch vụ phải giải thích những điều đó như thế nào mã không xác định được cho phép.

Tùy chỉnh mã phản hồi

Đặc tả HTTP cho phép mở rộng mã phản hồi. Tính năng mở rộng này chủ yếu có sẵn để cho phép các phiên bản HTTP trong tương lai giới thiệu các mã mới. Nó cũng được sử dụng bởi một số thông số kỹ thuật khác (chẳng hạn như WebDAV) để xác định mã tùy chỉnh. Điều này thường được thực hiện với những con số không có khả năng xung đột với các mã HTTP mới.

có thể đạt được bằng cách đặt chúng ở gần cuối phạm vi cụ thể (ví dụ: 299
dường như không bao giờ được sử dụng bởi tiêu chuẩn HTTP chính).

Việc kiểm kê dịch vụ cụ thể có thể tuân theo phương pháp này bằng cách đưa ra các mã phản hồi tùy chỉnh như một phần của tiêu chuẩn thiết kế kiểm kê dịch vụ. Để hỗ trợ ràng buộc Hợp đồng thống nhất {311},
mã phản hồi tùy chỉnh chỉ nên được xác định ở mức thống nhất
cấp hợp đồng, không phải ở cấp hợp đồng dịch vụ REST.

Khi tạo mã phản hồi tùy chỉnh, điều quan trọng là chúng phải được đánh số dựa trên
phạm vi chúng rơi vào. Ví dụ: mã 2xx sẽ truyền đạt thành công, trong khi
Mã 4xx chỉ nên thể hiện các điều kiện lỗi.

Ngoài ra, cách tốt nhất là tiêu chuẩn hóa việc chèn nội dung mà con người có thể đọc được
vào thông báo phản hồi HTTP thông qua Cụm từ lý do. Ví dụ: mã 400 có
cụm từ lý do mặc định của "Yêu cầu không hợp lệ". Điều này là đủ để người tiêu dùng dịch vụ có thể xử lý
phản ứng là một thất bại chung, nhưng nó không cho con người biết bất cứ điều gì hữu ích về
vấn đề thực tế. Đặt cụm từ lý do thành "Thiếu yêu cầu của người tiêu dùng dịch vụ
trường địa chỉ Khách hàng" hoặc thậm chí có thể là "Xác thực nội dung yêu cầu không thành công đối với
lực đồ <http://example.com/customer>" hữu ích hơn, đặc biệt khi xem xét
nhật ký về các điều kiện ngoại lệ có thể không có tài liệu đầy đủ được đính kèm.

Người tiêu dùng có thể kết logic chung để xử lý mã phản hồi trong từng phạm vi này,
nhưng cũng có thể cần liên kết logic cụ thể với các mã cụ thể. Một số mã có thể bị giới hạn

dễ chúng chỉ được tạo nếu người tiêu dùng yêu cầu một tính năng đặc biệt của HTTP, điều đó có nghĩa là một số mã có thể không được triển khai bởi người tiêu dùng không yêu cầu các tính năng này.

Các ngoại lệ hợp đồng thống nhất thường được tiêu chuẩn hóa trong bối cảnh một loại tương tác mới cụ thể được yêu cầu giữa dịch vụ và người tiêu dùng. Chúng thường sẽ được giới thiệu cùng với một hoặc nhiều phương thức và/hoặc tiêu đề mới. Ngữ cảnh này sẽ hướng dẫn loại ngoại lệ được tạo. Ví dụ, có thể cần thiết

dể giới thiệu mã phản hồi mới để chỉ ra rằng yêu cầu không thể được thực hiện do tài nguyên bị khóa. (WebDAV cung cấp mã 423 Locked cho mục đích này.)

Khi giới thiệu và chuẩn hóa mã phản hồi tùy chỉnh cho hợp đồng thống nhất kiểm kê dịch vụ, chúng tôi cần đảm bảo rằng:

- Mỗi mã tùy chỉnh đều phù hợp và thực sự cần thiết
- Mã tùy chỉnh mang tính chung chung và có khả năng tái sử dụng cao bởi các dịch vụ
- Mức độ mà hành vi của người tiêu dùng dịch vụ được điều chỉnh và không quá hạn chế để quy tắc có thể áp dụng cho nhiều tình huống có thể xảy ra
- Giá trị mã được thiết lập để tránh xung đột tiềm ẩn với mã phản hồi từ các mã liên quan đặc tả giao thức bên ngoài
- Giá trị mã được đặt để tránh xung đột với mã tùy chỉnh từ các phát minh dịch vụ khác các câu chuyện (để hỗ trợ việc trao đổi thông điệp kiểm kê hàng tồn kho giữa các dịch vụ tiềm năng có thể được yêu cầu)

Phạm vi số của mã phản hồi có thể được coi là một dạng ké thừa ngoại lệ. Bất kỳ mã nào trong một phạm vi cụ thể dự kiến sẽ được xử lý bởi một bộ logic mặc định, giống như nếu phạm vi là loại cha mẹ cho từng ngoại lệ trong phạm vi đó.

Trong phần này, chúng ta đã khám phá ngắn gọn các mã phản hồi trong ngữ cảnh của HTTP.

Tuy nhiên, điều đáng chú ý là REST có thể được áp dụng với các giao thức khác (và các giao thức khác). mã phản hồi). Cuối cùng, giao thức cơ sở của kiến trúc kiểm kê dịch vụ sẽ xác định cách báo cáo các điều kiện bình thường và đặc biệt.

Ví dụ: bạn có thể cân nhắc việc chuẩn hóa kho dịch vụ dựa trên REST

về việc sử dụng các thông báo SOAP dẫn đến các ngoại lệ dựa trên SOAP thay vì các mã ngoại lệ HTTP. Điều này cho phép các phạm vi mã phản hồi được thay thế cho việc kế thừa các ngoại lệ.

242 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

Thiết kế các loại phương tiện truyền thông

Trong suốt thời gian tồn tại của kiến trúc kiểm kê dịch vụ, chúng ta có thể mong đợi sẽ có nhiều thay đổi hơn đối với tập hợp các loại phương tiện truyền thông của hợp đồng thống nhất hơn là các phương pháp của nó. Vì Ví dụ: một loại phương tiện mới sẽ được yêu cầu bắt cứ khi nào một dịch vụ hoặc người tiêu dùng cần truyền đạt thông tin máy có thể đọc được nhưng không phù hợp với yêu cầu về định dạng hoặc lược đồ của bất kỳ loại phương tiện hiện có nào.

Một số loại phương tiện phổ biến trên Web cần xem xét đối với hàng tồn kho dịch vụ và hợp đồng dịch vụ bao gồm:

- văn bản/thuần túy; charset=utf-8 cho các biểu diễn đơn giản, chẳng hạn như số nguyên và dữ liệu chuỗi. Dữ liệu nguyên thủy có thể được mã hóa dưới dạng chuỗi, theo Lược đồ XML tích hợp Loại dữ liệu
- application/xhtml+xml dành cho các danh sách, bảng, văn bản người ta có thể đọc được phức tạp hơn, liên kết hypermedia với các loại mối quan hệ rõ ràng và dữ liệu bổ sung dựa trên microformats.org và các thông số kỹ thuật khác
- application/json cho một giải pháp thay thế nhẹ nhàng cho XML có sự hỗ trợ rộng rãi bởi ngôn ngữ lập trình
- text/uri-list cho danh sách URI đơn giản
- Ứng dụng/atom+xml dành cho nguồn cấp thông tin sự kiện mà con người có thể đọc được hoặc các thông tin khác bộ sưu tập dữ liệu có liên quan đến thời gian (hoặc theo thứ tự thời gian)

Trước khi phát minh ra các loại phương tiện mới để sử dụng trong kho dịch vụ, bạn nên trước tiên hãy tiến hành tìm kiếm các loại phương tiện truyền thông đã được thiết lập trong ngành có thể phù hợp.

Cho dù chọn loại phương tiện hiện có hay tạo phương tiện tùy chỉnh, việc xem xét các phương pháp hay nhất sau đây sẽ rất hữu ích:

- Mỗi loại phương tiện cụ thể lý tưởng nhất là cụ thể cho một lược đồ. Cho kỵ thí-ple, application/xml hoặc application/json không phải là lược đồ cụ thể, trong khi ứng dụng/atom+xml được sử dụng làm định dạng cung cấp phải đủ cụ thể để hữu ích cho việc đảm phán nội dung và xác định cách xử lý tài liệu.
- Các loại phương tiện truyền thông nên trùu tượng ở chỗ chúng chỉ xác định càng nhiều thông tin càng tốt người nhận của họ cần trích xuất thông qua lược đồ của họ. Giữ các loại phương tiện trùu tượng cho phép chúng được tái sử dụng trong nhiều hợp đồng dịch vụ hơn.

- Các loại phương tiện truyền thông mới nên sử dụng lại các từ vựng và khái niệm lâu đời trong ngành các thông số cụ thể bất cứ khi nào thích hợp. Điều này làm giảm rủi ro mà các khái niệm chính có bị bỏ sót hoặc được xây dựng kém và cải thiện hơn nữa khả năng tương thích với các thiết bị khác ứng dụng của những từ vựng giống nhau.
- Loại phương tiện nên bao gồm siêu liên kết bắt cứ khi nào nó cần tham chiếu đến một loại phương tiện liên quan tài nguyên có đại diện nằm bên ngoài tài liệu trực tiếp. liên kết các loại quan hệ có thể được xác định bởi lược đồ của loại phương tiện hoặc, trong một số trường hợp, riêng biệt, như một phần của hồ sơ quan hệ liên kết.
- Các loại phương tiện tùy chỉnh phải được xác định bằng ngữ nghĩa phải bỏ qua hoặc các loại khác các điểm mở rộng cho phép thêm dữ liệu mới vào các phiên bản phương tiện trong tương lai loại không có dịch vụ cũ và người tiêu dùng từ chối phiên bản mới.
- Các loại phương tiện cần được xác định bằng các hướng dẫn xử lý tiêu chuẩn mô tả cách bộ xử lý mới xử lý các tài liệu cũ có thể thiếu một số thông tin. Thông thường các hướng dẫn xử lý này đảm bảo rằng các phiên bản của tài liệu có ngữ nghĩa tương thích. Bằng cách này, các dịch vụ mới và người tiêu dùng không phải từ chối các phiên bản cũ.

Tất cả các loại phương tiện được phát minh cho kho dịch vụ cụ thể hoặc được sử dụng lại từ nguồn khác phải được ghi lại trong hồ sơ hợp đồng thống nhất, cùng với định nghĩa của các phương pháp thống nhất.

HTTP sử dụng các mã định danh loại phương tiện Internet phù hợp với một cú pháp cụ thể. Các loại phương tiện tùy chỉnh thường được xác định bằng ký hiệu:

`application/vnd. Organisation.type+supertype`

trong đó ứng dụng là tiền tố x phô biến cho biết loại được sử dụng cho tiêu chuẩn và tiêu chuẩn của máy. Trường tố chức xác định không gian tên nhà cung cấp, tùy chọn có thể được đăng ký với IANA.

Phần loại là tên duy nhất cho loại phương tiện trong tổ chức, trong khi phần supertype chỉ ra rằng loại này là sự cải tiến của loại phương tiện khác. Ví dụ, `application/vnd.com.examplebooks.purchase-order+xml` có thể chỉ ra rằng:

- Loại dành cho máy tiêu dùng.
- Loại này do nhà cung cấp chỉ định và tổ chức đã xác định loại đó là "examplebooks.com."

244 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

- Loại dành cho đơn đặt hàng (và có thể được liên kết với Mua hàng chuẩn Lược đồ XML đặt hàng).
- Loại này có nguồn gốc từ XML, nghĩa là người nhận có thể rõ ràng xử lý nội dung bằng trình phân tích cú pháp XML.

Các loại dành cho việc sử dụng liên tổ chức tổng quát hơn có thể được xác định bằng không gian tên loại phương tiện của tổ chức chịu trách nhiệm cuối cùng trong việc xác định loại.

Ngoài ra, chúng có thể được xác định mà không cần thông tin nhận dạng nhà cung cấp trong đặt bằng cách đăng ký trực tiếp từng loại, theo quy trình được xác định trong RFC 4288 sự chỉ rõ.

MẪU SOA

Mẫu Đầm phán Nội dung [334] chính thức hóa khả năng nguyên gốc của các dịch vụ REST để xử lý thông tin loại phương tiện trong thời gian chạy.

Thiết kế lược đồ cho các loại phương tiện

Trong kho dịch vụ, hầu hết các loại phương tiện tùy chỉnh được tạo để thể hiện hoạt động kinh doanh dữ liệu và tài liệu sẽ được xác định bằng Lược đồ XML hoặc Lược đồ JSON. Về cơ bản, điều này có thể thiết lập một tập hợp các mô hình dữ liệu được tiêu chuẩn hóa được các dịch vụ REST sử dụng lại trong kho ở bất kỳ mức độ khả thi nào.

Để điều này thành công, đặc biệt với các tập hợp dịch vụ lớn hơn, các lược đồ cần được thiết kế linh hoạt. Điều này có nghĩa là các lược đồ thường nên thực thi mức độ chi tiết ràng buộc xác thực ở mức độ thô cho phép mỗi lược đồ có thể áp dụng được để sử dụng với phạm vi yêu cầu tương tác dữ liệu rộng hơn.

REST yêu cầu các loại phương tiện và lược đồ của chúng chỉ được xác định ở cấp hợp đồng thống nhất. Nếu khả năng dịch vụ yêu cầu cấu trúc dữ liệu duy nhất cho thông báo phản hồi thì nó vẫn phải sử dụng một trong các loại phương tiện chuẩn được cung cấp bởi đồng phục hợp đồng. Việc thiết kế các lược đồ linh hoạt và được đánh máy yêu có thể đáp ứng nhiều yêu cầu trao đổi thông báo dành riêng cho từng dịch vụ, nhưng có lẽ không phải cho mọi trường hợp.

Ví dụ 9.2 cung cấp một ví dụ về thiết kế lược đồ linh hoạt.

```

Loại phương tiện = application/vnd.com.actioncon.po+xml
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://example.org/schema/po"
  xmlns="http://example.org/schema/po">
  <xsd:tên phần tử="LineItemList" type="LineItemListType"/>
  <xsd:complexType name="LineItemListType">
    <xsd:tên phần tử="LineItem" type="LineItemType"
      minOccurs="0"/>
  </xsd:complexType>
  <xsd:complexType name="LineItemType">
    <xsd:trình tự>
      <xsd:element name="productID" type="xsd:anyURI"/>
      <xsd:element name="productName" type="xsd:string"/>
      <xsd:tên phần tử="có sẵn" type="xsd:boolean"
        minOccurs="0"/>
    </xsd:trình tự>
  </xsd:complexType>
</xsd:lược đồ>

```

Ví dụ 9.2

Một trong những cách đơn giản nhất để làm cho loại phương tiện có thể tái sử dụng nhiều hơn là thiết kế lược đồ để hỗ trợ danh sách không có mục nào hoặc nhiều mục hơn. Điều này cho phép loại phương tiện cho phép một phiên bản của loại cơ bản nhưng cũng cho phép các truy vấn trả về 0 hoặc nhiều phiên bản. Việc tạo các thành phần riêng lẻ trong tài liệu là tùy chọn cũng có thể làm tăng khả năng sử dụng lại.

MẪU SOA

Mẫu Trình tự Xác thực [365] cung cấp một kỹ thuật để có tính gõ yếu các định nghĩa Lược đồ XML (cũng được khám phá trong Chương 6, 12,

và 13 trong cuốn sách có tiêu đề Thiết kế và tạo phiên bản hợp đồng dịch vụ web cho SOA). Mẫu Đàm phán nội dung [334] có thể được áp dụng để giải quyết tùy chọn có một dịch vụ REST duy nhất hỗ trợ hai lược đồ thay thế.

Về mặt kỹ thuật, các hợp đồng dịch vụ REST riêng lẻ có thể giới thiệu các lược đồ XML dành riêng cho hợp đồng, nhưng khi làm như vậy, chúng ta cần chấp nhận rằng Đồng nhất Ràng buộc hợp đồng {311} sẽ bị vi phạm.

Điều này có thể được đảm bảo khi khả năng dịch vụ cần tạo ra bản tin phản hồi chứa dữ liệu duy nhất (hoặc tổ hợp dữ liệu duy nhất) mà:

- Không tồn tại lược đồ chuẩn phù hợp
- Không có lược đồ chuẩn mới nào được tạo ra do thực tế là nó sẽ không có thể tái sử dụng bởi các dịch vụ khác

Hậu quả của việc không tuân thủ Hợp đồng thống nhất {311} là có khả năng làm tăng mức độ kết nối tiêu cực giữa người tiêu dùng dịch vụ và khả năng cung cấp dịch vụ dựa trên các loại phương tiện dành riêng cho dịch vụ. Các loại phương tiện dành riêng cho dịch vụ phải được xác định rõ ràng và cần nỗ lực để giảm thiểu số lượng logic được tiếp xúc trực tiếp và phụ thuộc vào các loại này.

Thiết kế phương pháp phức tạp

Hợp đồng thống nhất thiết lập một tập hợp các phương thức cơ bản được sử dụng để thực hiện các chức năng giao tiếp dữ liệu cơ bản. Như chúng tôi đã giải thích, mức độ trừu tượng chức năng cao này là điều gì làm cho hợp đồng thống nhất có thể tái sử dụng được ở mức độ mà chúng tôi có thể định vị nó là cơ chế trao đổi dữ liệu tổng thể, duy nhất cho toàn bộ kho dịch vụ. Bên cạnh đó tính đơn giản vốn có của nó, phần này của kiến trúc kiểm kê dịch vụ sẽ tự động mang lại kết quả trong việc tiêu chuẩn hóa cơ bản các thành phần hợp đồng dịch vụ và trao đổi tin nhắn.

Việc tiêu chuẩn hóa HTTP trên World Wide Web dẫn đến một đặc tả giao thức mô tả những điều mà các dịch vụ và người tiêu dùng "có thể", "nên" hoặc "phải" thực hiện. để phù hợp với giao thức. Mức độ tiêu chuẩn hóa đạt được chỉ được coi ý ở mức cao cần thiết để đảm bảo chức năng cơ bản của Web. Nó để lại một số quyết định về cách ứng phó với các điều kiện khác nhau theo logic trong từng dịch vụ và người tiêu dùng. Mức độ tiêu chuẩn hóa "nguyên thủy" này rất quan trọng đối với Web, nơi chúng ta có thể có nhiều người tiêu dùng dịch vụ nước ngoài tương tác với nhau.

với các dịch vụ của bên thứ ba tại bất kỳ thời điểm nào.

Tuy nhiên, kho dịch vụ thường đại diện cho một môi trường riêng tư và được kiểm soát trong một doanh nghiệp CNTT. Điều này cho chúng ta cơ hội tùy chỉnh tiêu chuẩn hóa này ngoài việc sử dụng các phương pháp thông thường và nguyên thủy. Hình thức tùy chỉnh này có thể được biện minh khi chúng ta có yêu cầu tăng mức độ có thể đoán trước và chất lượng dịch vụ vượt xa những gì World Wide Web có thể cung cấp.

Ví dụ: giả sử chúng tôi muốn giới thiệu một tiêu chuẩn thiết kế theo đó tất cả các tài liệu liên quan đến kế toán (hóa đơn, đơn đặt hàng, giấy báo cáo, v.v.) phải được được truy xuất bằng logic, khi gặp lỗi truy xuất, sẽ tự động thử lại truy xuất nhiều lần. Logic sẽ yêu cầu thêm rằng các nỗ lực truy xuất tiếp theo không làm thay đổi trạng thái của tài nguyên đại diện cho các tài liệu kinh doanh (bất kể nỗ lực truy xuất nhất định có thành công hay không).

Với loại tiêu chuẩn thiết kế này, về cơ bản chúng tôi đang giới thiệu một bộ quy tắc và các yêu cầu về cách thức truy xuất một loại tài liệu cụ thể cần được thực hiện. Đây là những quy tắc và yêu cầu không thể được thể hiện hoặc thực thi thông qua các phương thức cơ bản, nguyên thủy do HTTP cung cấp. Thay vào đó, chúng ta có thể áp dụng chúng ngoài mức độ tiêu chuẩn hóa được thực thi bởi HTTP bằng cách tập hợp chúng (cùng với các loại chức năng thời gian chạy khác) thành các tương tác tổng hợp. Đây là cơ sở của phương pháp phức tạp.

Một phương thức phức tạp bao gồm một tập hợp các tương tác được xác định trước giữa dịch vụ và người sử dụng dịch vụ. Những tương tác này có thể bao gồm việc gọi các phương thức HTTP tiêu chuẩn. Để phân biệt rõ hơn các phương thức cơ sở này với các phương thức phức tạp đóng gói chúng, chúng ta sẽ coi các phương thức HTTP cơ sở là các phương thức nguyên thủy (chỉ một thuật ngữ được sử dụng khi thảo luận về thiết kế phương pháp phức tạp).

Các phương thức phức tạp được coi là "phức tạp" vì chúng:

- Có thể liên quan đến việc kết hợp nhiều phương thức nguyên thủy
- Có thể liên quan đến việc kết hợp một phương thức nguyên thủy nhiều lần
- Có thể giới thiệu chức năng bổ sung ngoài việc gọi phương thức
- Có thể yêu cầu các tiêu đề hoặc thuộc tính tùy chọn được hỗ trợ hoặc đưa vào tin nhắn

Như đã nêu trước đây, các phương pháp phức tạp thường được tùy chỉnh và tiêu chuẩn hóa trong một kho dịch vụ nhất định. Để một phương pháp phức tạp được chuẩn hóa, nó cần phải được ghi lại như một phần của đặc tả kiến trúc kiểm kê dịch vụ. Chúng ta có thể xác định một số phương pháp phức tạp phổ biến như một phần của hợp đồng thống nhất mà sau đó sẽ có sẵn để tất cả các dịch vụ trong kho dịch vụ triển khai.

Các phương thức phức tạp có tên riêng biệt. Các ví dụ về phương pháp phức tạp mà chúng tôi đang đề cập đến được gọi là:

- Tìm nạp - Một loạt các yêu cầu GET có thể phục hồi từ nhiều trường hợp ngoại lệ khác nhau
- Lưu trữ - Một loạt các yêu cầu PUT hoặc DELETE có thể phục hồi từ nhiều ngoại lệ
- Delta - Một loạt các yêu cầu GET giúp người tiêu dùng luôn đồng bộ với việc thay đổi trạng thái tài nguyên
- Không đồng bộ - Yêu cầu được sửa đổi ban đầu và các tương tác tiếp theo hỗ trợ xử lý tin nhắn yêu cầu không đồng bộ

Các dịch vụ hỗ trợ một phương thức phức tạp truyền đạt điều này bằng cách hiển thị tên phương thức như một phần của khả năng dịch vụ riêng biệt (Hình 9.11), bên cạnh các phương thức nguyên thủy mà phương thức phức tạp được xây dựng dựa trên đó. Khi các nhóm dự án tạo chương trình tiêu dùng cho một số dịch vụ nhất định, họ có thể xác định logic phía người tiêu dùng cần thiết cho một dịch vụ nhất định. phương pháp phức tạp bằng cách xác định những phương pháp phức tạp nào mà dịch vụ hỗ trợ, như được chỉ ra trong hợp đồng dịch vụ đã công bố.

Hình 9.11

Hợp đồng dịch vụ Hóa đơn hiển thị hai khả năng dịch vụ dựa trên các phương pháp nguyên thủy và hai khả năng dịch vụ dựa trên các phương pháp phức tạp. Ban đầu chúng ta có thể giả định rằng hai phương pháp phức tạp kết hợp việc sử dụng hai phương pháp nguyên thủy và tiến hành xác nhận điều này bằng cách nghiên cứu đặc tả thiết kế ghi lại các phương pháp phức tạp.



GHI CHÚ

Khi áp dụng nguyên tắc Trừu tượng dịch vụ (294) cho thiết kế thành phần dịch vụ REST, chúng tôi có thể loại trừ hoàn toàn việc mô tả một số phương thức nguyên thủy khỏi hợp đồng dịch vụ. Đây có thể là kết quả của các tiêu chuẩn thiết kế chỉ cho phép sử dụng một phương pháp phức tạp trong một số trường hợp nhất định. Quay trở lại ví dụ trước về việc sử dụng một phương pháp phức tạp để truy xuất các tài liệu liên quan đến kế toán, chúng ta có thể có một tiêu chuẩn thiết kế cấm các tài liệu này được truy xuất thông qua phương thức GET thông thường (vì phương thức GET không thực thi các yêu cầu bổ sung). yêu cầu về độ tin cậy).

Điều quan trọng cần lưu ý là không cần thiết phải sử dụng các phương pháp phức tạp. Bên ngoài các môi trường được kiểm soát trong đó các phương pháp phức tạp có thể được xác định, tiêu chuẩn hóa và áp dụng một cách an toàn để hỗ trợ mục tiêu Tăng cường khả năng tương tác nội tại, việc sử dụng là không phổ biến và thường không được khuyến khích. Khi xây dựng kiến trúc kiểm kê dịch vụ, chúng ta có thể chọn tiêu chuẩn hóa một số tương tác nhất định thông qua việc sử dụng các phương thức phức tạp hoặc chúng ta có thể chọn giới hạn tương tác dịch vụ REST chỉ ở mức sử dụng các phương thức nguyên thủy. Quyết định này sẽ dựa chủ yếu vào bản chất riêng biệt của các yêu cầu kinh doanh được giải quyết và tự động hóa bởi các dịch vụ trong kho dịch vụ.

Bất chấp tên gọi của chúng, các phương pháp phức tạp nhằm mục đích tăng thêm sự đơn giản cho kiến trúc kho dịch vụ. Ví dụ: hãy tưởng tượng chúng ta quyết định không sử dụng phương pháp xác định trước

và sau đó nhận ra rằng có những quy tắc hoặc chính sách chung mà chúng tôi áp dụng cho nhiều dịch vụ và người tiêu dùng của chúng. Trong trường hợp này, chúng ta sẽ xây dựng điểm chung logic tương tác dư thừa trên mỗi cặp dịch vụ tiêu dùng riêng lẻ. Bởi vì logic chưa được chuẩn hóa nên việc triển khai dư thừa của nó có thể sẽ tồn tại theo cách khác. Khi cần thay đổi các quy tắc hoặc chính sách chung, chúng ta sẽ cần xem lại từng triển khai dự phòng cho phù hợp. Gánh nặng bảo trì này và thực tế là việc triển khai sẽ tiếp tục không đồng bộ khiến kiến trúc này trở nên phức tạp một cách không cần thiết. Đây chính xác là vấn đề mà việc sử dụng các phương pháp phức tạp nhằm tránh.

Các phần sắp tới giới thiệu một tập hợp các phương thức phức tạp mẫu được tổ chức thành hai phần:

- Các phương pháp phức tạp phi trạng thái
- Các phương thức phức tạp có trạng thái

Lưu ý rằng những phương pháp này hoàn toàn không phải là tiêu chuẩn ngành. Tên của chúng, loại tương tác thông báo cũng như lời gọi phương thức nguyên thủy mà chúng bao gồm đã được tùy chỉnh để giải quyết các loại chức năng phổ biến.

GHI CHÚ

Ví dụ nghiên cứu điển hình ở cuối chương này sẽ khám phá sâu hơn về chủ đề này. Trong ví dụ này, để đáp ứng các yêu cầu kinh doanh cụ thể, hai phương thức phức tạp mới (một phương thức không trạng thái, phương thức còn lại có trạng thái) được xác định.

Phương pháp phức tạp không quốc tịch

Bộ sưu tập các phương thức phức tạp đầu tiên này đóng gói các tương tác thông báo tuân thủ ràng buộc Không trạng thái {308}.

Phương pháp tìm nạp

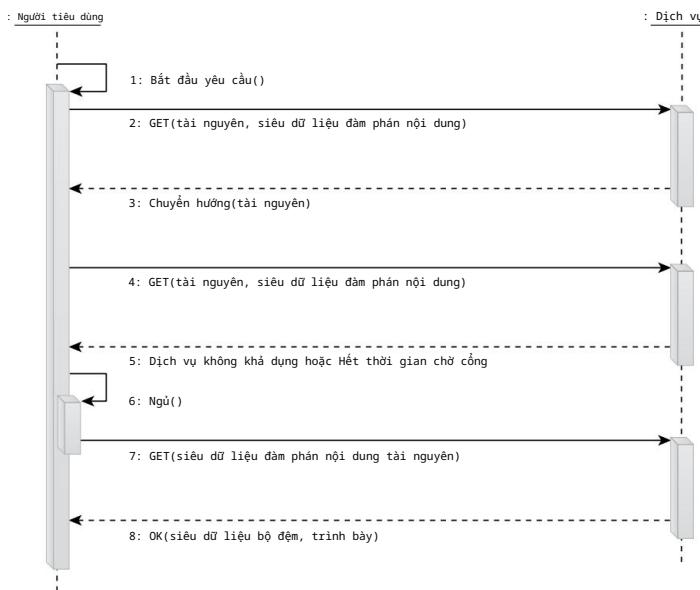
Thay vì chỉ dựa vào một lệnh gọi duy nhất của phương thức HTTP GET (và các tiêu đề và hành vi liên quan của nó) để truy xuất nội dung, chúng ta có thể xây dựng một dữ liệu phức tạp hơn phương pháp truy xuất với các tính năng như

- Tự động thử lại khi hết thời gian hoặc kết nối không thành công
- Hỗ trợ cẩn thiết cho việc đàm phán nội dung thời gian chạy để đảm bảo người tiêu dùng dịch vụ nhận dữ liệu ở dạng nó hiếu

250 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

- Hỗ trợ chuyển hướng cần thiết để đảm bảo rằng những thay đổi trong hợp đồng dịch vụ có thể được thực hiện được phục vụ một cách duyên dáng bởi người tiêu dùng dịch vụ
- Dịch vụ cần hỗ trợ chỉ thị kiểm soát bộ nhớ đệm để đảm bảo độ trễ tối thiểu, mức sử dụng băng thông tối thiểu và khả năng xử lý tối thiểu đối với các yêu cầu dữ trữ

Chúng tôi sẽ gọi loại phương thức phức tạp chỉ đọc nâng cao này là **Tìm nạp**. Hình 9.12 cho thấy một ví dụ về tương tác thông báo được xác định trước của phương pháp Tìm nạp được thiết kế để thực hiện đảm phán nội dung và thử lại tự động.



Hình 9.12

Một ví dụ về phương thức phức tạp Tìm nạp bao gồm các lệnh gọi phương thức GET liên tiếp.

Phương pháp lưu trữ

Khi sử dụng các phương thức PUT hoặc DELETE tiêu chuẩn để thêm tài nguyên mới, đặt trạng thái của tài nguyên hiện có hoặc xóa tài nguyên cũ, người sử dụng dịch vụ có thể phải chịu thời gian chờ yêu cầu hoặc phản hồi ngoại lệ. Mặc dù đặc tả HTTP giải thích ý nghĩa của từng ngoại lệ nhưng nó không áp đặt các hạn chế về cách xử lý chúng. Với mục đích này, chúng ta có thể tạo một phương thức Store tùy chỉnh để chuẩn hóa các hành vi cần thiết.

Phương thức Store có thể có một số tính năng tương tự như Tìm nạp, chẳng hạn như yêu cầu tự động thử lại các yêu cầu, hỗ trợ đàm phán nội dung và hỗ trợ chuyển hướng

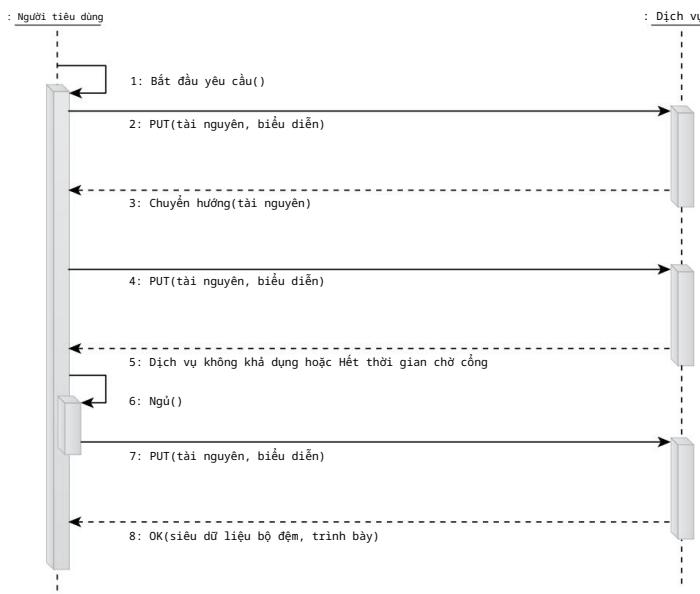
ngoại lệ. Sử dụng PUT và DELETE, nó cũng có thể đánh bại các kết nối băng thông thấp bằng cách luôn gửi trạng thái gần đây nhất mà người tiêu dùng yêu cầu, thay vì cần hoàn thành các yêu cầu trước đó trước.

Giống như cách mà các phương thức HTTP nguyên thủy riêng lẻ có thể bình thường, Store phương pháp có thể được thiết kế để hành xử bình thường. Bằng cách xây dựng dựa trên các phương thức idempotent nguyên thủy, mọi thông báo yêu cầu thành công, lặp lại sẽ không còn tác dụng nữa sau khi thông báo yêu cầu đầu tiên được thực hiện thành công.

Ví dụ: khi đặt trạng thái hóa đơn từ "Chưa thanh toán" thành "Đã thanh toán":

- Yêu cầu "chuyển đổi" sẽ không bình thường vì việc lặp lại yêu cầu chuyển đổi trạng thái trở lại "Chưa thanh toán".
- Yêu cầu "PUT" không có giá trị khi đặt hóa đơn thành "Đã thanh toán" vì nó có hiệu quả tương tự, bất kể yêu cầu được lặp lại bao nhiêu lần.

Điều quan trọng là phải hiểu rằng Cửa hàng và các yêu cầu PUT và DELETE cơ bản của nó là các yêu cầu đối với logic dịch vụ, không phải là một hành động được thực hiện trên cơ sở dữ liệu cơ bản của dịch vụ. Như được hiển thị trong Hình 9.13, các loại yêu cầu này được nêu theo cách bình thường để cho phép thử lại các yêu cầu một cách hiệu quả mà không cần số thứ tự để thêm hỗ trợ nhắn tin đáng tin cậy.



Hình 9.13

Một ví dụ về sự tương tác được thực hiện bởi phương thức phức tạp Store.

GHI CHÚ

Các khả năng dịch vụ kết hợp loại phương pháp này là một ví dụ về ứng dụng mẫu Khả năng bình thường [345].

Phương pháp Delta

Người tiêu dùng dịch vụ thường cần phải đồng bộ hóa với trạng thái của tài nguyên đang thay đổi. Phương pháp Delta là một cơ chế đồng bộ hóa tạo điều kiện cho việc đồng bộ hóa không trạng thái về trạng thái của một tài nguyên đang thay đổi giữa dịch vụ sở hữu trạng thái này và người tiêu dùng cần duy trì sự liên kết với trạng thái.

Phương pháp Delta tuân theo logic xử lý dựa trên ba chức năng cơ bản sau:

1. Dịch vụ lưu giữ lịch sử các thay đổi đối với tài nguyên.
2. Người tiêu dùng nhận được một URL để cập đến vị trí trong lịch sử đại diện lần cuối cùng người tiêu dùng truy vấn trạng thái của tài nguyên.
3. Lần tiếp theo người tiêu dùng truy vấn trạng thái tài nguyên, dịch vụ (sử dụng URL do người tiêu dùng cung cấp) trả về danh sách các thay đổi đã xảy ra kể từ lần trước người tiêu dùng đã truy vấn trạng thái tài nguyên.

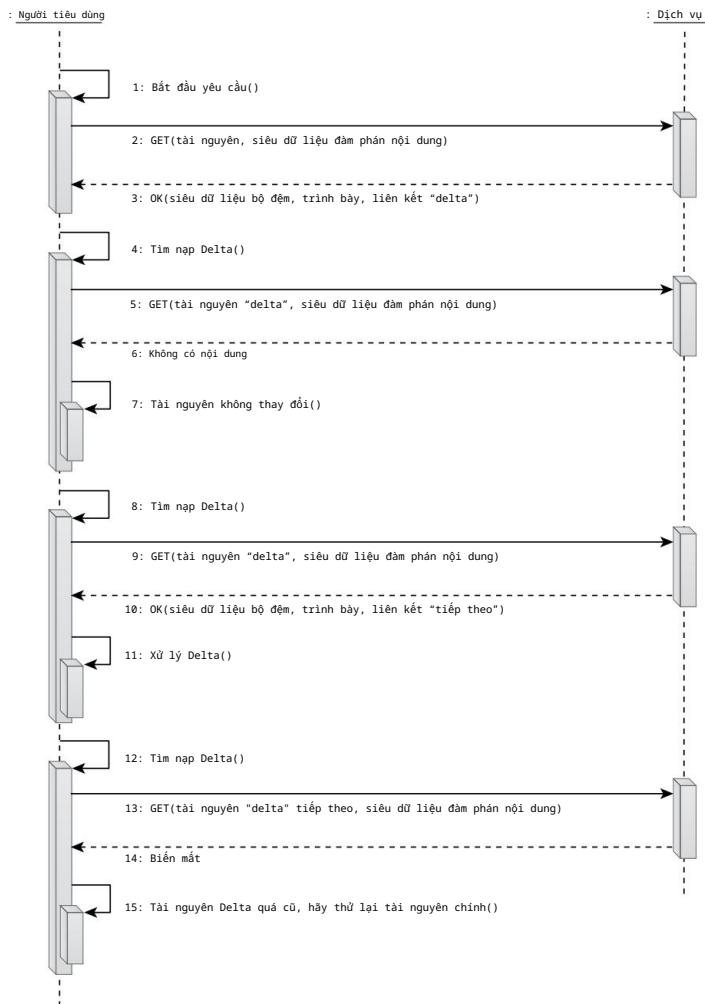
Hình 9.14 minh họa điều này bằng cách sử dụng một loạt lệnh gọi GET.

Dịch vụ này cung cấp tài nguyên "chính" đáp ứng các yêu cầu GET bằng cách trả về trạng thái hiện tại của tài nguyên. Bên cạnh tài nguyên chính, nó cung cấp một bộ sưu tập "delta" mỗi tài nguyên trả về danh sách các thay đổi từ một điểm được chỉ định trong bộ đệm lịch sử.

Người tiêu dùng của phương pháp Delta kích hoạt định kỳ hoặc khi được yêu cầu bởi logic người tiêu dùng cốt lõi. Nếu nó có mã định danh tài nguyên delta, nó sẽ gửi yêu cầu đến vị trí đó. Nếu nó không có mã định danh tài nguyên delta, nó sẽ truy xuất tài nguyên chính để trở thành được đồng bộ hóa. Trong phản hồi tương ứng, người tiêu dùng nhận được một liên kết đến delta cho điểm hiện tại trong bộ đệm lịch sử. Liên kết này sẽ được tìm thấy trong tiêu đề Liên kết (RFC 5988) với loại quan hệ Delta.

9.2 Nguyên tắc thiết kế dịch vụ REST

253



Hình 9.14

Một ví dụ về tương tác thông điệp được bao hàm bởi phương pháp phức Delta.

Tài nguyên delta được yêu cầu có thể ở bất kỳ trạng thái nào sau đây:

1. Nó có thể biểu thị một tập hợp một hoặc nhiều thay đổi đã xảy ra đối với phần chính tài nguyên kể từ thời điểm trong lịch sử mà mã định danh tài nguyên delta đề cập đến. TRONG trường hợp này, tất cả các thay đổi trong lịch sử từ điểm được chỉ định sẽ được trả về cùng với với một liên kết đến vùng đồng bằng mới cho điểm hiện tại trong bộ đệm lịch sử. liên kết này sẽ được tìm thấy trong tiêu đề Liên kết có kiểu quan hệ Tiếp theo.
2. Nó có thể không có một tập hợp các thay đổi vì không có thay đổi nào xảy ra kể từ điểm được chỉ định của nó trong bộ đệm lịch sử, trong trường hợp đó nó có thể trả về 204 Không có Nội dung mã phản hồi để cho biết rằng người sử dụng dịch vụ đã được cập nhật và có thể tiếp tục sử dụng tài nguyên delta cho lần truy xuất tiếp theo.
3. Những thay đổi có thể đã xảy ra nhưng đồng bằng đã hết hạn vì danh nghĩa điểm ghi trong lịch sử bây giờ đã quá cũ đến nỗi dịch vụ đã chọn không bảo tồn những thay đổi. Trong trường hợp này, tài nguyên có thể trả về mã 410 Gone để cho biết rằng người tiêu dùng đã mất đồng bộ hóa và phải truy xuất lại tài nguyên chính.

Tài nguyên Delta sử dụng chiến lược bộ nhớ đệm giống như tài nguyên chính.

Dịch vụ kiểm soát số lượng vùng đồng bằng lịch sử được chuẩn bị tích lũy, dựa trên họ mong đợi (trung bình) người tiêu dùng sẽ mất bao nhiêu thời gian để cập nhật thông tin. Trong một số trường hợp nhất định khi duy trì dấu vết kiểm tra đầy đủ cho các mục đích khác, số lượng delta có thể là vô thời hạn. Dung lượng không giới hạn cần thiết để lưu giữ hồ sơ này là không đổi và có thể dự đoán được bắt kể số lượng người tiêu dùng, tùy theo từng người tiêu dùng dịch vụ.

để theo dõi vị trí của nó trong bộ đệm lịch sử.

Phương pháp không đồng bộ

Phương pháp phức tạp này cung cấp các tương tác được xác định trước để trao đổi thành công và hủy bỏ các tin nhắn không đồng bộ. Điều này rất hữu ích khi một yêu cầu nhất định cần nhiều thời gian để thực thi hơn thời gian chờ yêu cầu HTTP tiêu chuẩn cho phép.

Thông thường, nếu một yêu cầu mất quá nhiều thời gian, logic xử lý tin nhắn của người tiêu dùng sẽ mất thời gian out hoặc người trung gian sẽ trả lại mã phản hồi 504 Gateway Timeout cho người dùng dịch vụ. Phương thức Async cung cấp cơ chế dự phòng để xử lý các yêu cầu và trả về các phản hồi không yêu cầu người sử dụng dịch vụ duy trì kết nối HTTP mở trong tổng thời gian tương tác yêu cầu.

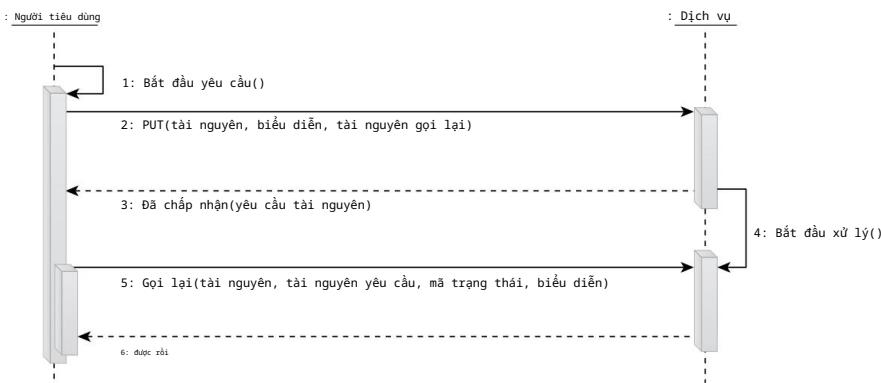
Như được hiển thị trong Hình 9.15, người sử dụng dịch vụ đưa ra một yêu cầu nhưng chỉ định xõi một mã định danh tài nguyên gọi lại. Nếu dịch vụ chọn sử dụng mã định danh này, nó sẽ phản hồi

9.2 Nguyên tắc thiết kế dịch vụ REST

255

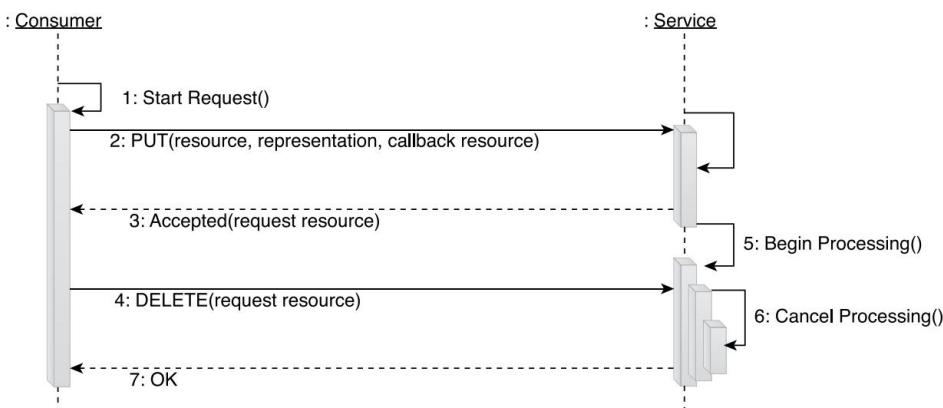
với mã phản hồi 202 được chấp nhận và có thể tùy ý trả về mã nhận dạng tài nguyên trong tiêu đề Vị trí để giúp nó theo dõi vị trí của yêu cầu không đồng bộ trong hàng đợi xử lý của nó.

Khi yêu cầu đã được xử lý đầy đủ, kết quả của nó sẽ được cung cấp bởi dịch vụ sau đó đưa ra yêu cầu đến địa chỉ gọi lại của người tiêu dùng dịch vụ. Nếu người sử dụng dịch vụ đưa ra yêu cầu XOA (như trong Hình 9.16) trong khi yêu cầu Async vẫn ở trong hàng đợi xử lý (và trước khi phản hồi được trả về), thì một yêu cầu được xác định trước riêng biệt sẽ được thực hiện. tương tác được thực hiện để hủy yêu cầu không đồng bộ. Trong trường hợp này, không có phản hồi được trả về và dịch vụ sẽ hủy việc xử lý yêu cầu.



Hình 9.15

Tương tác yêu cầu không đồng bộ được bao gồm bởi phương thức phức tạp Async.



Hình 9.16

Tương tác hủy không đồng bộ được bao gồm bởi phương thức phức tạp Async.

256 Chương 9: API dịch vụ và thiết kế hợp đồng với dịch vụ REST và microservice

Nếu người tiêu dùng không thể nghe yêu cầu gọi lại, nó có thể sử dụng yêu cầu không đồng bộ nhận dạng để thăm dò dịch vụ định kỳ. Sau khi yêu cầu được xử lý thành công, có thể truy xuất kết quả của nó bằng phương pháp Tìm nạp được mô tả trước đó trước khi xóa trạng thái yêu cầu không đồng bộ. Các dịch vụ thực hiện tương tác được bao gồm bởi phương pháp này phải có phương tiện loại bỏ các yêu cầu không đồng bộ cũ nếu người tiêu dùng dịch vụ không sẵn sàng nhận phản hồi hoặc nói cách khác là “quên” xóa tài nguyên yêu cầu.

Các phương thức phức tạp có trạng thái

Hai phương pháp phức tạp sau đây sử dụng REST làm cơ sở cho thiết kế dịch vụ nhưng kết hợp các tương tác có chủ ý vi phạm ràng buộc Không trạng thái {308}. Mặc dù các kịch bản được thể hiện bằng các phương pháp này tương đối phổ biến trong các thiết kế ứng dụng doanh nghiệp truyền thống, nhưng kiểu giao tiếp này không được coi là có nguồn gốc từ Thế giới.

Web rỗng. Việc sử dụng các phương pháp phức tạp có trạng thái có thể được đảm bảo khi chúng tôi chấp nhận việc giảm khả năng mở rộng đi kèm với quyết định thiết kế này.

Phương pháp chuyển đổi

Phương thức Trans về cơ bản cung cấp các tương tác cần thiết để thực hiện cam kết hai giai đoạn giữa một người tiêu dùng dịch vụ và một hoặc nhiều dịch vụ. Những thay đổi được thực hiện trong giao dịch được đảm bảo sẽ lan truyền thành công trên tất cả các dịch vụ tham gia hoặc tất cả các dịch vụ được khôi phục về trạng thái ban đầu.

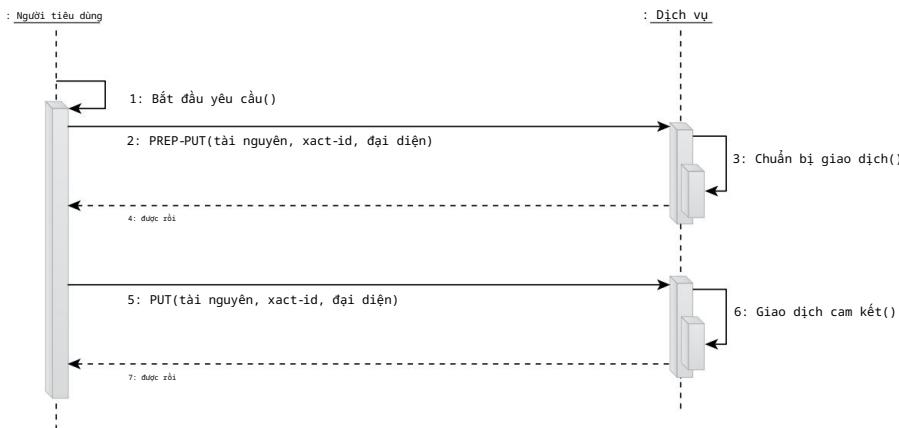
Loại phương pháp phức tạp này yêu cầu chức năng “chuẩn bị” cho mỗi người tham gia trước khi một cam kết hoặc khôi phục cuối cùng được thực hiện. Chức năng thuộc loại này không được HTTP hỗ trợ nguyên bản. Vì vậy, chúng ta cần giới thiệu một phương thức PREP-PUT tùy chỉnh (một biến thể của phương thức PUT), như trong Hình 9.17.

Trong ví dụ này, phương thức PREP-PUT tương đương với PUT, nhưng nó không cam kết hành động PUT. Một tên phương thức khác được sử dụng để đảm bảo rằng nếu dịch vụ không hiểu cách tham gia vào phương thức phức hợp Trans, thì dịch vụ đó sẽ từ chối phương thức PREP-PUT và cho phép người tiêu dùng hủy bỏ giao dịch.

Việc thực hiện logic đằng sau một phương thức phức hợp Trans diễn hình thường sẽ yêu cầu sự tham gia của bộ điều khiển giao dịch để đảm bảo rằng các chức năng cam kết và khôi phục được thực hiện thực sự và đáng tin cậy với tính nguyên tử.

9.2 Nguyên tắc thiết kế dịch vụ REST

257



Hình 9.17

Một ví dụ về phương thức phức tạp Trans, sử dụng phương thức nguyên thủy tùy chỉnh có tên PREP-PUT.

Phương thức PubSub

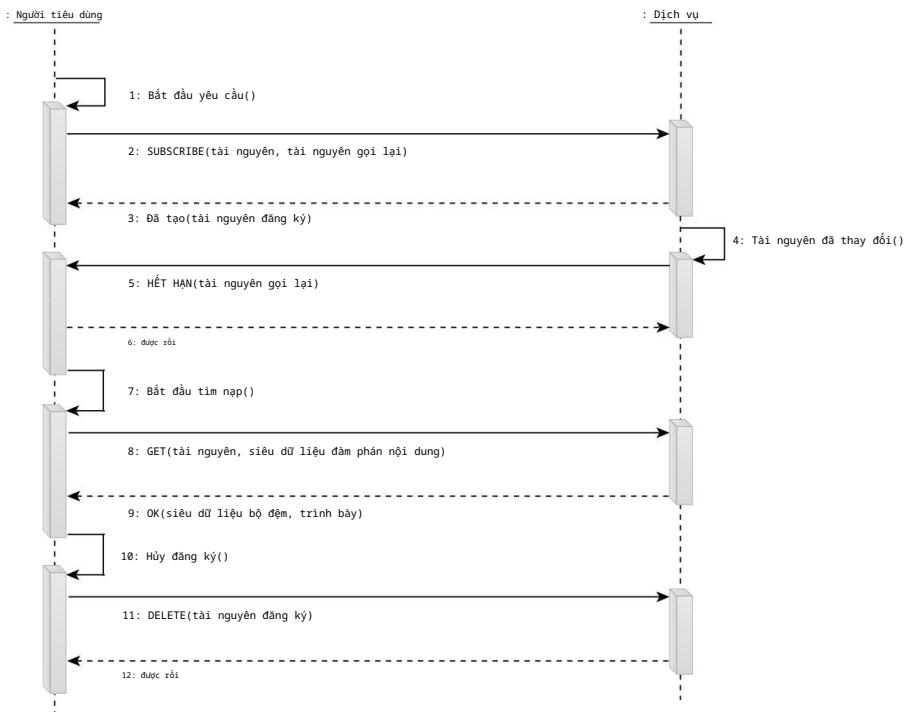
Nhiều tùy chọn xuất bản-đăng ký có sẵn sau khi quyết định có tình vi phạm ràng buộc Không quốc tịch {308}. Các loại cơ chế này được thiết kế để hỗ trợ các tương tác thời gian thực trong đó người tiêu dùng dịch vụ phải hành động ngay lập tức khi xảy ra một số sự kiện được xác định trước tại một tài nguyên nhất định.

Có nhiều cách khác nhau để thiết kế phương pháp phức tạp này. Hình 9.18 minh họa một cách tiếp cận coi tin nhắn xuất bản-đăng ký là "sự vô hiệu hóa bộ đệm"

cơ chế.

Hình thức tương tác xuất bản-đăng ký này được coi là "nhẹ" vì nó không yêu cầu dịch vụ gửi những thay đổi thực tế đến người đăng ký. Thay vào đó, nó thông báo cho họ biết rằng tài nguyên đã thay đổi bằng cách đưa mã định danh tài nguyên ra ngoài và sau đó sử dụng lại một phương thức Tìm nạp hiện có, có thể lưu vào bộ nhớ đệm khi người sử dụng dịch vụ lấy các biểu diễn mới của tài nguyên đã thay đổi.

Lượng trạng thái cần thiết để quản lý các đăng ký này được ràng buộc vào một bản ghi có kích thước cố định cho mỗi người sử dụng dịch vụ. Nếu có nhiều thông báo không hợp lệ xếp hàng cho một sự kiện đã đăng ký cụ thể, chúng có thể được gộp lại với nhau thành một thông báo duy nhất. Bất kể cho dù người tiêu dùng có nhận được một hoặc nhiều thông báo vô hiệu hay không thì nó vẫn sẽ chỉ cần gọi một phương thức Tìm nạp để tự cập nhật trạng thái tài nguyên của nó mỗi khi nó nhìn thấy một hoặc nhiều thông báo vô hiệu mới.



Hình 9.18

Một ví dụ về phương pháp phức tạp PubSub dựa trên việc vô hiệu hóa bộ đệm. Khi dịch vụ xác định rằng có điều gì đó đã thay đổi trên một hoặc nhiều tài nguyên, nó sẽ đưa ra thông báo hết hạn bộ đệm cho người đăng ký. Sau đó, mỗi người đăng ký có thể sử dụng phương pháp Tìm nạp phức tạp (hoặc phương pháp tương đương) để cập nhật cho người đăng ký về các thay đổi.

Phương thức PubSub có thể được điều chỉnh thêm để phân phối tài đăng ký và lưu trữ trạng thái phiền đến các vị trí khác nhau trên mạng. Kỹ thuật này có thể đặc biệt hiệu quả trong các môi trường dựa trên đám mây cung cấp nhiều tài nguyên lưu trữ.

MẪU SOA

Mẫu Nhắn tin theo hướng sự kiện [343] có thể được áp dụng để hỗ trợ phương pháp phức tạp này. Nó cung cấp một giải pháp thay thế cho việc thăm dò tài nguyên lặp đi lặp lại, điều này có thể tác động tiêu cực đến hiệu suất nếu tần suất bỏ phiếu tăng lên để phát hiện các thay đổi với độ trễ tối thiểu.

VÍ DỤ NGHIÊN CỨU TRƯỜNG HỢP

Nhóm MUA chịu trách nhiệm thiết kế dịch vụ gấp phải một số yêu cầu về truy cập và cập nhật trạng thái tài nguyên. Ví dụ:

- Một người tiêu dùng dịch vụ cần đọc nguyên tử trạng thái của tài nguyên, thực hiện xử lý và lưu trữ trạng thái cập nhật trở lại tài nguyên.
- Một người sử dụng dịch vụ khác cần hỗ trợ các hành động đồng thời của người dùng sửa đổi cùng một tài nguyên. Những hành động này cập nhật các thuộc tính tài nguyên nhất định trong khi những người khác cần phải giữ nguyên.

Việc cho phép người tiêu dùng dịch vụ riêng lẻ chứa logic tùy chỉnh khác nhau để thực hiện các loại chức năng này sẽ vô tình dẫn đến sự cố và ngoại lệ thời gian chạy khi bất kỳ người tiêu dùng dịch vụ nào cố gắng cập nhật cùng một tài nguyên cùng một lúc.

Các kiến trúc sư MUA kết luận rằng cách đơn giản nhất để tránh điều này là giới thiệu một phương pháp phức tạp mới nhằm đảm bảo rằng tài nguyên bị khóa trong khi được cập nhật bởi một người tiêu dùng nhất định. Sử dụng các quy tắc khóa lạc quan, một cách tiếp cận thường được sử dụng với các bản cập nhật cơ sở dữ liệu, họ có thể tạo ra một phương thức phức tạp không trạng thái và tận dụng các tính năng tiêu chuẩn hiện có của giao thức HTTP. Họ đặt tên cho phương thức này là “OptLock” và viết mô tả chính thức được tạo thành một phần của hồ sơ hợp đồng thống nhất.

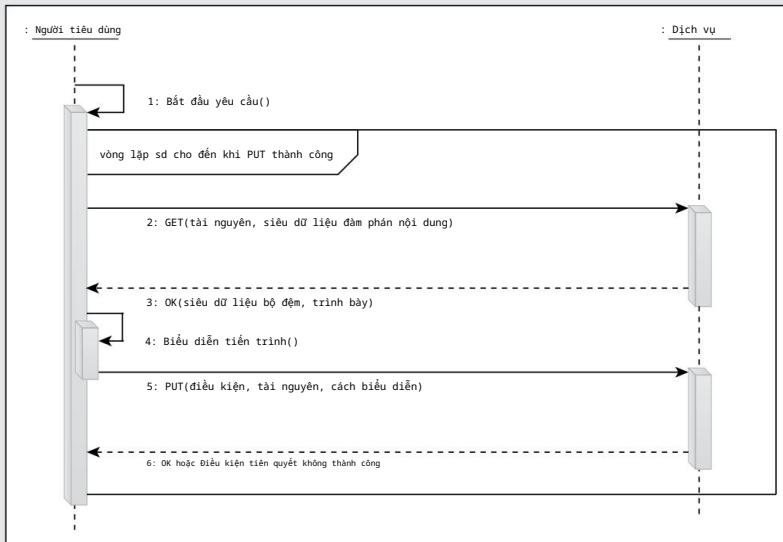
Phương pháp phức tạp OptLock

Nếu hai người sử dụng dịch vụ riêng biệt cố gắng cập nhật trạng thái của tài nguyên cùng một lúc, hành động của họ rõ ràng sẽ xung đột với nhau vì kết quả phụ thuộc vào thứ tự các yêu cầu của họ đến được dịch vụ. Phương pháp OptLock (Hình 9.19) giải quyết vấn đề này bằng cách cung cấp phương tiện để người tiêu dùng dịch vụ có thể xác định xem trạng thái của tài nguyên có thay đổi kể từ lần cuối người tiêu dùng đọc nó trước khi thử cập nhật hay không.

Cụ thể, trước tiên người tiêu dùng sẽ truy xuất trạng thái hiện tại được liên kết với tài nguyên nhận dạng bằng phương pháp Tìm nạp. Cùng với dữ liệu, người tiêu dùng cũng nhận được “ETag.” ETag là một khái niệm từ HTTP giúp xác định duy nhất phiên bản của tài nguyên một cách không rõ ràng. Bất cứ khi nào tài nguyên thay đổi trạng thái, ETag của nó được đảm bảo hely khác đi. Khi người tiêu dùng dịch vụ khởi tạo một Cửa hàng, nó sẽ thực hiện điều đó một cách có điều kiện bằng cách yêu cầu dịch vụ chỉ thực hiện tương tác với Cửa hàng nếu ETag của tài nguyên vẫn

khớp với cái mà nó có khi tìm nạp. Điều này được thực hiện với tiêu đề If-Match .

Dịch vụ có thể sử dụng giá trị ETag trong điều kiện để phát hiện xem trạng thái tài nguyên có bị thay đổi trong thời gian chờ đợi hay không.



Hình 9.19

Một ví dụ về phương pháp phức tạp OptLock.

Phương thức phức tạp OptLock không giới thiệu bất kỳ tính năng mới nào cho HTTP, nhưng thay vào đó đưa ra các yêu cầu mới để xử lý các yêu cầu GET và PUT.

Cụ thể, yêu cầu GET phải trả về giá trị ETag và yêu cầu PUT phải xử lý tiêu đề If-Match . Ngoài ra, nếu tài nguyên đã thay đổi, dịch vụ phải đảm bảo thêm không thực hiện yêu cầu PUT.

Có một số kỹ thuật để tính toán ETAGs. Một số tính toán giá trị băm từ thông tin trạng thái liên quan đến tài nguyên, một số chỉ giữ lại “kết quả cuối cùng”.

modified” dấu thời gian cho từng tài nguyên và những tài nguyên khác theo dõi phiên bản của tài nguyên rõ ràng.

Phương pháp OptLock có thể không mở rộng hiệu quả để có khả năng truy cập đồng thời cao vào một tài nguyên cụ thể. Nếu yêu cầu cập nhật của người tiêu dùng bị từ chối do Xung đột HTTP 409 mã phản hồi, phương pháp OptLock quy định cách người tiêu dùng có thể khôi phục bằng cách

tìm nạp phiên bản mới hơn của tài nguyên mà họ phải tính toán lại thay đổi và thử lại phương thức Store. Tuy nhiên, điều này có thể thất bại lần nữa do xung đột yêu cầu cập nhật. Người tiêu dùng dịch vụ tương tác với tài nguyên theo cách này dựa vào tài nguyên cụ thể đó có tỷ lệ truy cập ghi tương đối thấp.

Phương thức phức hợp OptLock có sẵn như một phần của hợp đồng thống nhất và được một số dịch vụ triển khai. Tuy nhiên, xuất hiện các tình huống trong đó nhiều người tiêu dùng cố gắng sửa đổi tài nguyên cùng lúc, gây ra các ngoại lệ thường xuyên.

và cập nhật không thành công. Những tình huống này xảy ra trong thời gian sử dụng cao điểm và do khối lượng sử dụng đồng thời dự kiến sẽ tăng thêm, người ta xác định rằng cần phải thiết lập các phương tiện hiệu quả để tuân tự hóa các bản cập nhật cho tài nguyên.

Thay vào đó, người ta đề xuất thay đổi phương pháp phức tạp OptLock để thực hiện khóa bí quan, theo mô tả phương pháp phức tạp PesLock sau đây.

Phương pháp phức hợp PesLock

Khóa bí quan mang lại sự linh hoạt và chắc chắn hơn so với khóa lạc quan.

Từ góc độ REST, điều này phải trả giá bằng việc giới thiệu các tương tác có trạng thái và hạn chế quyền truy cập đồng thời trong khi khóa bí quan được giữ.

Như được hiển thị trong Hình 9.20, các tiện ích mở rộng WebDAV cho HTTP cung cấp các nguyên tắc khóa có thể được sử dụng trong kiến trúc thành phần có ý vi phạm ràng buộc Không trạng thái {308}. Một người tiêu dùng có thể khóa người khác truy cập vào tài nguyên, vì vậy phải cẩn thận để có chính sách kiểm soát truy cập phù hợp.

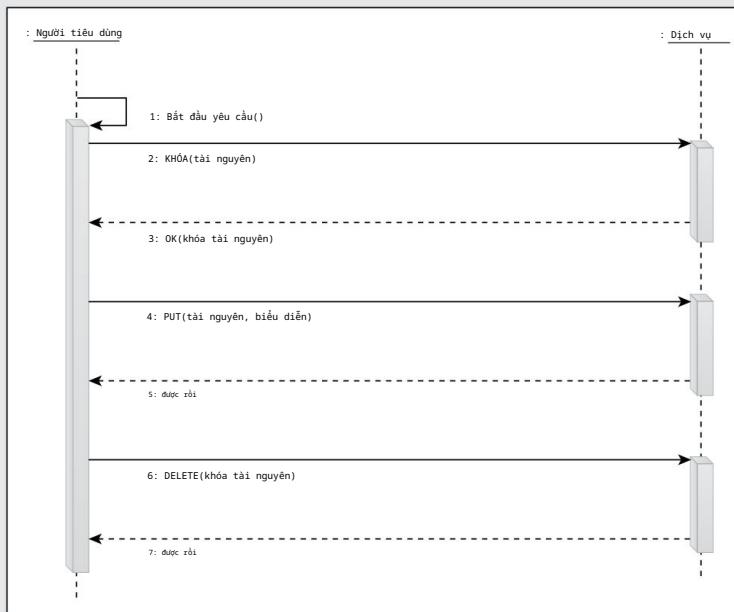
Người tiêu dùng cũng có thể bị lỗi khi khóa được giữ, điều đó có nghĩa là khóa phải có khả năng hết thời gian độc lập với người tiêu dùng đăng ký chúng.

Bằng cách này, người sử dụng dịch vụ sẽ có thể khóa tài nguyên trong thời gian nó cần đọc trạng thái, sửa đổi nó và viết lại. Mặc dù dịch vụ khác người tiêu dùng vẫn sẽ gặp phải các trường hợp ngoại lệ trong khi cố gắng cập nhật tài nguyên cùng lúc với người tiêu dùng đã khóa tài nguyên đó, điều đó được coi là thích hợp hơn không thể đoán trước được việc quản lý tài nguyên như một phần của mô hình khóa lạc quan.

Giải pháp này không được tất cả các kiến trúc sư MUA chấp nhận vì việc duy trì khóa tài nguyên đòi hỏi phải vi phạm ràng buộc Không trạng thái {308}. Hơn nữa, nó có thể dẫn đến nguy cơ khởi động ở khóa cũ, ảnh hưởng đến hiệu suất và khả năng mở rộng. Đặc biệt, trừ khi các biện pháp thích hợp được thực hiện để đảm bảo rằng chỉ những người được phép

người tiêu dùng có thể khóa một tài nguyên, điều này khiến tài nguyên có thể bị tấn công từ chối dịch vụ bởi những người tiêu dùng độc hại có thể khóa tất cả những người tiêu dùng khác.

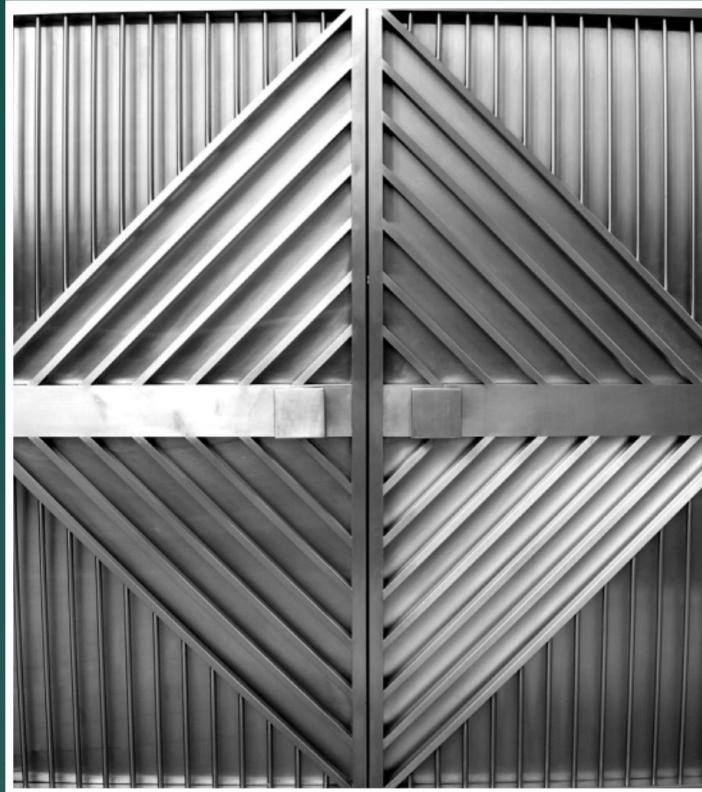
Sau khi thảo luận thêm, một thỏa hiệp đã đạt được. Phương pháp OptLock sẽ được thử trước tiên. Là một phương pháp dự phòng, nếu người tiêu dùng thử ba lần và không thành công, nó sẽ thử phương thức PesLock có trạng thái để đảm bảo có thể hoàn thành hành động.



Hình 9.20

Một ví dụ về phương pháp phức tạp PesLock.

Chương 10



API dịch vụ và lập phiên bản hợp đồng với dịch vụ web và dịch vụ REST

10.1 Khái niệm cơ bản về

lập phiên bản 10.2 Lập phiên bản và khă

năng tương thích 10.3 Cân nhắc về tính tương thích của dịch vụ REST

10.4 Mã nhận dạng phiên bản

10.5 Chiến lược tạo phiên bản

10.6 Các cân nhắc về tạo phiên bản dịch vụ REST

GHI CHÚ

Chương này cung cấp một số ví dụ về mã giúp minh họa các kịch bản và cách tiếp cận phiên bản khác nhau. Lưu ý rằng các ví dụ mã này không liên quan đến bất kỳ ví dụ mã nào được cung cấp trong Ví dụ nghiên cứu trường hợp từ các chương trước.

Sau khi hợp đồng dịch vụ được triển khai, chúng tôi buộc phải dành thời gian để xác định điều gì với bắt đầu được hình thành-hợp đồng, chúng ta cần tìm hiểu:

- Liệu những thay đổi này có tác động tiêu cực đến hiện tại (và có thể là trong tương lai) hay không dịch vụ người tiêu dùng
- Những thay đổi sẽ và sẽ không ảnh hưởng đến người tiêu dùng nên được thực hiện như thế nào và truyền đạt

Những vấn đề này dẫn đến sự cần thiết phải lập phiên bản. Bất cứ khi nào bạn đưa khái niệm lập phiên bản vào một dự án SOA, một số câu hỏi có thể sẽ được đặt ra, ví dụ:

- Chính xác thì điều gì cấu thành một phiên bản mới của hợp đồng dịch vụ? Có gì khác biệt giữa phiên bản chính và phiên bản phụ?
- Các phần của số phiên bản cho biết điều gì?
- Liệu phiên bản mới của hợp đồng có còn hiệu quả với những người tiêu dùng hiện tại đã được thiết kế cho phiên bản hợp đồng cũ?
- Phiên bản hiện tại của hợp đồng có phù hợp với người tiêu dùng mới có thể có yêu cầu trao đổi dữ liệu khác nhau?
- Cách tốt nhất để thêm các thay đổi vào hợp đồng hiện có trong khi giảm thiểu tác động tới người tiêu dùng?
- Chúng ta có cần lưu trữ các hợp đồng cũ và mới cùng một lúc không? nếu được thì trong bao lâu?

Chúng tôi sẽ giải quyết những câu hỏi này và cung cấp một tập hợp các tùy chọn để giải quyết các vấn đề phổ biến về phiên bản. Các phần sắp tới bắt đầu bằng cách đề cập đến một số khái niệm cơ bản, thuật ngữ và chiến lược cụ thể để lập phiên bản hợp đồng dịch vụ.

10.1 Khái niệm cơ bản về lập phiên bản

bản Vì vậy, khi chúng tôi nói rằng chúng tôi đang tạo một phiên bản mới của hợp đồng dịch vụ, chính xác thì điều gì là chúng tôi đề cập đến? Các phần sau đây giải thích một số thuật ngữ và khái niệm cơ bản cũng như phân biệt rõ hơn giữa hợp đồng dịch vụ Web và hợp đồng dịch vụ REST.

Phiên bản dịch vụ web

Như chúng tôi đã thiết lập nhiều lần trong cuốn sách này, một hợp đồng dịch vụ Web có thể bao gồm của một số tài liệu và định nghĩa riêng lẻ được liên kết và tập hợp lại với nhau để tạo thành một giao diện kỹ thuật hoàn chỉnh.

Ví dụ: một hợp đồng dịch vụ Web nhất định có thể bao gồm:

- Một (đôi khi nhiều hơn) định nghĩa WSDL
- Một (thường là nhiều) định nghĩa Lược đồ XML
- Một số định nghĩa về Chính sách WS (đôi khi không có)

Hơn nữa, mỗi tài liệu định nghĩa này có thể được chia sẻ bởi các dịch vụ Web khác. hợp đồng. Ví dụ,

- Định nghĩa Lược đồ XML tập trung sẽ thường được sử dụng bởi nhiều WSDL các định nghĩa.
- Định nghĩa Chính sách WS tập trung thường sẽ được áp dụng cho nhiều WSDL các định nghĩa.
- Một mô tả WSDL trừu tượng có thể được nhập bởi nhiều WSDL cụ thể mô tả hoặc ngược lại.

Trong tất cả các phần khác nhau của hợp đồng dịch vụ Web, phần thiết lập giao diện kỹ thuật cơ bản thú vị là mô tả trừu tượng về định nghĩa WSDL. Điều này thể hiện cốt lõi của hợp đồng dịch vụ Web và sau đó được mở rộng và chi tiết hơn nữa thông qua các định nghĩa lược đồ, các định nghĩa chính sách và một hoặc nhiều WSDL cụ thể.
mô tả.

Do đó, khi chúng ta cần tạo một phiên bản mới của hợp đồng dịch vụ Web, chúng ta có thể giả định rằng đã có sự thay đổi trong mô tả WSDL trừu tượng hoặc một trong các tài liệu hợp đồng liên quan đến mô tả WSDL trừu tượng. Nội dung hợp đồng dịch vụ Web thường có thể thay đổi là nội dung lược đồ XML cung cấp các kiểu định nghĩa thông báo của mô tả trừu tượng. Cuối cùng, một hợp đồng khác liên quan đến

266 Chương 10: API dịch vụ và tạo phiên bản hợp đồng với dịch vụ web và dịch vụ REST

công nghệ vẫn có thể áp đặt các yêu cầu về phiên bản nhưng ít có khả năng thực hiện điều đó một cách đơn giản vì đây là một phần ít phổ biến hơn trong các hợp đồng dịch vụ Web là WS-Policy.

Phiên bản các dịch vụ REST Nếu chúng

ta làm theo mô hình REST sử dụng một hợp đồng thống nhất để thể hiện khả năng dịch vụ thì việc chia sẻ các tài liệu định nghĩa giữa các hợp đồng dịch vụ thậm chí còn rõ ràng hơn. Ví dụ,

- Tất cả các phương thức HTTP được sử dụng trong hợp đồng đều là tiêu chuẩn trên toàn bộ kiến trúc.
- Các định nghĩa về Lược đồ XML là tiêu chuẩn vì chúng được gói gọn trong các phương tiện truyền thông chung các loại.
- Cú pháp nhận dạng cho các điểm cuối dịch vụ hạng nhẹ (được gọi là tài nguyên) là chuẩn trên toàn bộ kiến trúc.

Những thay đổi đối với các khía cạnh hợp đồng thống nhất làm nền tảng cho từng hợp đồng dịch vụ có thể ảnh hưởng đến bất kỳ dịch vụ REST nào trong kho dịch vụ.

Các ràng buộc hạt mìn và thô

Bất kể các lược đồ XML được sử dụng với các dịch vụ Web hay các dịch vụ REST, các thay đổi về phiên bản thường gắn liền với việc tăng hoặc giảm số lượng hoặc mức độ chi tiết của các ràng buộc được thể hiện trong định nghĩa lược đồ. Vì vậy, chúng ta hãy tóm tắt ngắn gọn ý nghĩa của thuật ngữ mức độ chi tiết của ràng buộc liên quan đến định nghĩa kiểu.

Lưu ý các phần in đậm và in nghiêng trong Ví dụ 10.1:

```

<xsd:element name="LineItem" type="LineItemType"/> <xsd:complexType
name="LineItemType"> <xsd:sequence> <xsd :element
name="productID"
type="xsd:string"/> < xsd:element name="productName"
type="xsd:string"/> <xsd:any minOccurs="0" maxOccurs="unbounded"
namespace="##any" processContents="lax"/> </xsd:sequence
> <xsd:anyAttribution namespace="##any"/>

</ xsd:complexType>

```

Ví dụ 10.1

Một cấu trúc complexType chứa các ràng buộc chi tiết và chi tiết thô.

Như được chỉ ra bởi văn bản in đậm, có các phần tử có tên và kiểu dữ liệu cụ thể đại diện cho các phần của định nghĩa thông điệp với mức độ chi tiết ràng buộc tốt.

Tất cả các phiên bản thông báo (tài liệu XML thực tế sẽ được tạo dựa trên cấu trúc này) phải tuân theo các ràng buộc này để được coi là hợp lệ (đó là lý do tại sao chúng được coi là các ràng buộc “tối thiểu” tuyệt đối).

Văn bản in nghiêng hiển thị các ký tự đại diện của phần tử và thuộc tính cũng được chứa trong loại pharc tạp này. Chúng đại diện cho các phần của định nghĩa thông điệp với một cách hiểu cực kỳ thô thiển. mức độ chi tiết ràng buộc trong các thông báo đó hoàn toàn không cần phải tuân theo các phần này của định nghĩa thông báo.

Việc sử dụng các thuật ngữ “hạt mìn” và “hạt thô” mang tính chủ quan cao. Cái gì có thể là một ràng buộc chi tiết trong một hợp đồng nhưng có thể không có trong một hợp đồng khác. Vấn đề là phải hiểu cách áp dụng các thuật ngữ này khi so sánh các phần của định nghĩa thông điệp hoặc khi so sánh các định nghĩa thông điệp khác nhau với nhau.

10.2 Phiên bản và khả năng tương thích

Mỗi quan tâm số một khi phát triển và triển khai phiên bản mới của hợp đồng dịch vụ là tác động của nó đối với các bộ phận khác của doanh nghiệp đã hình thành hoặc sẽ phụ thuộc vào nó. Biện pháp tác động này liên quan trực tiếp đến mức độ tương thích của phiên bản hợp đồng mới với phiên bản cũ và môi trường xung quanh nó nói chung.

Phần này thiết lập các loại tương thích cơ bản liên quan đến nội dung và thiết kế của các phiên bản hợp đồng mới, đồng thời cũng gắn với các mục tiêu và hạn chế của các chiến lược tạo phiên bản khác nhau được giới thiệu ở cuối chương này.

Khả năng tương thích ngược

Phiên bản mới của hợp đồng dịch vụ tiếp tục hỗ trợ các chương trình dành cho người tiêu dùng được thiết kế để hoạt động với phiên bản cũ được coi là tương thích ngược. Từ góc độ thiết kế, điều này có nghĩa là hợp đồng mới không thay đổi theo cách có thể tác động đến các chương trình tiêu dùng hiện tại đang sử dụng hợp đồng.

Khả năng tương thích ngược trong dịch vụ web

Ví dụ 10.2 cung cấp một ví dụ đơn giản về thay đổi tương thích ngược dựa trên việc bổ sung một thao tác mới vào định nghĩa WSDL hiện có:

268 Chương 10: API dịch vụ và tạo phiên bản hợp đồng với dịch vụ web và dịch vụ REST

```

<định nghĩa name="Đơn đặt hàng" targetNamespace=
  "http://actioncon.com/contract/po" xmlns="http://
  schemas.xmlsoap.org/wsdl/" xmlns:tns="http://
  actioncon.com/contract/po" xmlns:po= "http://actioncon.com/
  schema/po">
  ...
<portType name="ptPurchaseOrder">
  <Operation name="opSubmitOrder"> <input
    message="tns:msgSubmitOrderRequest"/> <output
    message="tns:msgSubmitOrderResponse"/> </Operation> <Operation
      name="opCheckOrderStatus">
        <input message="tns:msgCheckOrderRequest"/> <output
          message="tns:msgCheckOrderResponse"/>
        </Operation>
      <Operation name="opChangeOrder"> <input
        message="tns:msgChangeOrderRequest"/> <output
        message="tns:msgChangeOrderResponse"/>
      </Operation>
      <Operation name="opCancelOrder"> <input
        message="tns:msgCancelOrderRequest"/> <output
        message="tns:msgCancelOrderResponse"/>
      </Operation>
      <Operation name="opGetOrder"> <input
        message="tns:msgGetOrderRequest"/> <output
        message="tns:msgGetOrderResponse"/>
      </Operation>
    </portType> </
  định nghĩa>

```

Ví dụ 10.2

Việc bổ sung một thao tác mới thể hiện một thay đổi tương thích ngược phô biến.

Bằng cách thêm một hoạt động hoàn toàn mới, chúng tôi đang tạo một phiên bản mới của hợp đồng nhưng thay đổi này tương thích ngược và sẽ không ảnh hưởng đến bất kỳ người tiêu dùng hiện tại nào. Việc triển khai dịch vụ mới sẽ tiếp tục hoạt động với người tiêu dùng dịch vụ cũ vì tất cả các hoạt động mà người tiêu dùng dịch vụ hiện tại có thể gọi vẫn hiện diện và tiếp tục đáp ứng các yêu cầu của phiên bản hợp đồng dịch vụ trước đó.

Khả năng tương thích ngược trong dịch vụ REST

Thay đổi tương thích ngược đối với hợp đồng dịch vụ tuân thủ REST có thể liên quan đến việc thêm một số tài nguyên mới hoặc thêm khả năng mới vào tài nguyên hiện có. Trong mỗi trường hợp này, người sử dụng dịch vụ hiện tại sẽ chỉ gọi các phương thức cũ trên các tài nguyên cũ và chúng vẫn tiếp tục hoạt động như trước đây.

Như được minh họa trong Ví dụ 10.3, việc hỗ trợ một phương pháp mới mà người tiêu dùng dịch vụ hiện tại không sử dụng sẽ dẫn đến thay đổi tương thích ngược. Tuy nhiên, trong một dịch vụ kiểm kê với nhiều dịch vụ REST, chúng tôi có thể thực hiện các bước để đảm bảo rằng dịch vụ mới người tiêu dùng sẽ tiếp tục làm việc với các phiên bản dịch vụ cũ.

Dịch vụ: po.actioncon.com

Khả năng:

ĐĂNG /đơn hàng

Trong =application/vnd.com.actioncon.po+xml

NHẬN /đơn hàng/{order-id}/trạng thái

Out = văn bản/thuần túy

PUT /đơn hàng/{order-id}

Trong =application/vnd.com/actioncon.po+xml

XÓA /orders/{order-id}

NHẬN /đơn hàng/{order-id}

Out = application/vnd.com.actioncon.po+xml

Ví dụ 10.3

Việc bổ sung tài nguyên mới hoặc phương thức được hỗ trợ mới trên tài nguyên là một thay đổi tương thích ngược đối với dịch vụ REST.

Như được hiển thị trong Ví dụ 10.4, điều quan trọng là người tiêu dùng dịch vụ phải có cách hợp lý để tiến hành tương tác nếu dịch vụ báo cáo rằng phương pháp mới không được triển khai.

Các phương pháp pháp lý cho việc kiểm kê dịch vụ của actioncon.com:

- * LẤY
 - * ĐẶT
 - * XÓA BỎ
 - * BƯỚU KIỆN
 - * ĐĂNG KÝ (người tiêu dùng phải quay lại dịch vụ GET if định kỳ báo cáo "không được thực hiện")
-

Ví dụ 10.4

Các phương pháp mới được thêm vào hợp đồng thống nhất của kho dịch vụ cần cung cấp một cách để người tiêu dùng dịch vụ "quay lại" phương pháp đã sử dụng trước đó nếu chúng thực sự tương thích ngược.

Những thay đổi đối với lược đồ và loại phương tiện tiếp cận khả năng tương thích ngược theo một cách khác, trong đó chúng mô tả cách thông tin có thể được mã hóa để truyền tải và sẽ thường được sử dụng trong cả thông báo yêu cầu và phản hồi. Trọng tâm của khả năng tương thích ngược là liệu người nhận tin nhắn mới có thể hiểu được thông tin được gửi từ nguồn cũ hay không. Nói cách khác, bộ xử lý mới phải tiếp tục hiểu thông tin do trình tạo thông báo cũ tạo ra.

270 Chương 10: API dịch vụ và lập phiên bản hợp đồng với dịch vụ web và dịch vụ REST

Một ví dụ về thay đổi được thực hiện đổi với lược đồ cho định nghĩa thông báo tương thích ngược là việc bổ sung một phần tử tùy chọn (như được hiển thị trong mã đánh dấu in đậm trong Ví dụ 10.5).

```
Loại phương tiện = application/vnd.com.actioncon.po+xml
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://actioncon.com/schema/po"
    xmlns="http://actioncon.com/schema/po">
    <xsd:tên phần tử="LineItem" type="LineItemType"/>
    <xsd:complexType name="LineItemType">
        <xsd:trình tự>
            <xsd:element name="productID" type="xsd:string"/>
            <xsd:element name="productName" type="xsd:string"/>
            <xsd:tên phần tử="có sẵn" type="xsd:boolean"
                minOccurs="0"/>
        </xsd:trình tự>
    </xsd:complexType>
</xsd:lược đồ>
```

Ví dụ 10.5

Trong định nghĩa Lược đồ XML, việc bổ sung phần tử tùy chọn cũng được coi là tương thích ngược.

Ở đây chúng tôi đang sử dụng một phiên bản đơn giản hóa của định nghĩa Lược đồ XML cho dịch vụ Đơn đặt hàng. Phần tử có sẵn tùy chọn được thêm vào loại phức tạp LineItemType . Điều này không có tác động đến các tổ chức tạo hiện có vì họ không bắt buộc phải cung cấp phần tử này trong các thông điệp của mình. Bộ xử lý mới phải được thiết kế để đối phó với tình trạng không có thông tin mới nếu chúng muốn duy trì khả năng tương thích ngược.

Việc thay đổi bất kỳ thành phần hiện có nào trong ví dụ trước từ bắt buộc thành tùy chọn (bằng cách thêm cài đặt minOccurs="0") cũng sẽ được coi là thay đổi tương thích ngược. Khi chúng tôi có quyền kiểm soát cách chúng tôi chọn thiết kế phiên bản tiếp theo của hợp đồng dịch vụ Web, khả năng tương thích ngược thường có thể đạt được.

Tuy nhiên, những thay đổi bắt buộc (chẳng hạn như những thay đổi do luật pháp hoặc quy định áp đặt) thường có thể buộc chúng tôi phải phá vỡ khả năng tương thích ngược.

GHI CHÚ

Cả hai chiến lược tạo phiên bản Linh hoạt và Lỏng lẻo được giải thích ở cuối chương này đều hỗ trợ khả năng tương thích ngược.

Khả năng tương thích chuyển tiếp

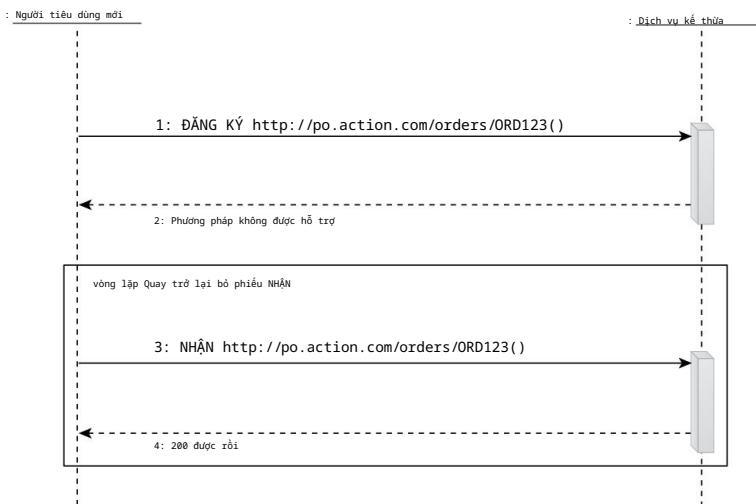
Khi một hợp đồng dịch vụ được thiết kế theo cách có thể hỗ trợ một loạt các chương trình tiêu dùng trong tương lai thì nó được coi là có mức độ tương thích về phía trước.

Điều này có nghĩa là hợp đồng về cơ bản có thể điều chỉnh các chương trình tiêu dùng sẽ phát triển như thế nào theo thời gian.

Việc hỗ trợ khả năng tương thích chuyển tiếp cho các hoạt động dịch vụ Web hoặc các phương thức hợp đồng thống nhất yêu cầu phải có các loại ngoại lệ trong hợp đồng để cho phép người tiêu dùng dịch vụ khôi phục nếu họ cố gắng gọi một hoạt động hoặc phương thức mới và không được hỗ trợ.

Ví dụ: phản hồi "phương pháp không được triển khai" cho phép người tiêu dùng dịch vụ phát hiện rằng nó đang xử lý một dịch vụ không tương thích, từ đó cho phép nó xử lý việc này ngoại lệ một cách duyên dáng.

Mã ngoại lệ chuyển hướng giúp các dịch vụ REST triển khai hợp đồng thống nhất thay đổi số nhận dạng tài nguyên trong hợp đồng khi được yêu cầu. Đây là một cách khác mà hợp đồng dịch vụ có thể cho phép người tiêu dùng dịch vụ cũ tiếp tục sử dụng dịch vụ sau khi thay đổi hợp đồng đã diễn ra (Ví dụ 10.6).



Ví dụ 10.6

Dịch vụ REST đảm bảo khả năng tương thích về phía trước bằng cách đưa ra một ngoại lệ bắt cứ khi nào nó không hiểu hợp đồng có thể sử dụng lại hoặc phương thức hợp đồng thống nhất.

Khả năng tương thích chuyển tiếp của các lược đồ trong các dịch vụ REST yêu cầu phải có các điểm mở rộng để có thể thêm thông tin mới để các bộ xử lý cũ bỏ qua một cách an toàn.

272 Chương 10: API dịch vụ và tạo phiên bản hợp đồng với dịch vụ web và dịch vụ REST

Ví dụ:

- Mọi xác nhận mà bộ xử lý thực hiện không được từ chối tài liệu được định dạng theo lược đồ mới.
- Tất cả thông tin hiện có mà bộ xử lý có thể cần phải có trong phiên bản tương lai của lược đồ.
- Mọi thông tin mới được thêm vào lược đồ phải an toàn đối với các bộ xử lý kế thừa bỏ qua (nếu bộ xử lý phải hiểu thông tin mới thì thay đổi đó không thể tương thích về phía trước).
- Bộ xử lý phải bỏ qua mọi thông tin mà nó không hiểu.

Một cách phổ biến để đảm bảo việc xác thực không từ chối các phiên bản trong tương lai của lược đồ là sử dụng các ký tự đại diện trong phiên bản cũ hơn. Chúng cung cấp các điểm mở rộng nơi thông tin mới có thể được thêm vào trong các phiên bản lược đồ trong tương lai, như trong Ví dụ 10.7.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://actioncon.com/schema/po"
  xmlns="http://actioncon.com/schema/po">
  <xsd:teil phần tử="LineItem" type="LineItemType"/>
  <xsd:complexType name="LineItemType">
    <xsd:trình tự>
      <xsd:element name="productID" type="xsd:string"/>
      <xsd:element name="productName" type="xsd:string"/>
      <xsd:any namespace="##any" processContents="lax" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:trình tự>
    <xsd:anyAttribution namespace="##any" />
  </xsd:complexType>
</xsd:lược đồ>
```

Ví dụ 10.7

Để hỗ trợ khả năng tương thích về phía trước trong định nghĩa thông báo thường yêu cầu sử dụng các ký tự đại diện Lược đồ XML.

Trong ví dụ này, các phần tử xsd:any và xsd:anyAttribution được thêm vào để cho phép một loạt các yếu tố và dữ liệu chưa biết được chấp nhận bởi hợp đồng dịch vụ. Nói cách khác, lược đồ đang được thiết kế trước để phù hợp với những thay đổi không lường trước được trong tương lai.

Điều quan trọng là phải hiểu rằng việc đưa các điểm mở rộng vào hợp đồng dịch vụ để có khả năng tương thích về phía trước không có nghĩa là loại bỏ nhu cầu xem xét các vấn đề về tính tương thích khi thực hiện thay đổi hợp đồng. Thông tin mới chỉ có thể được thêm vào lược đồ

theo cách tương thích về phía trước nếu nó thực sự an toàn để bộ xử lý bỏ qua. Các hoạt động mới chỉ có thể được thực hiện tương thích về phía trước nếu người tiêu dùng dịch vụ có một hoạt động hiện có để quay trở lại khi phát hiện thấy hoạt động mà họ có gắng gọi ban đầu không được hỗ trợ.

Dịch vụ có hợp đồng tương thích về phía trước thường sẽ không thể xử lý tất cả nội dung tin nhắn. Hợp đồng của nó được thiết kế đơn giản để chấp nhận phạm vi dữ liệu rộng hơn chưa được biết tại thời điểm thiết kế.

GHI CHÚ

Khả năng tương thích chuyển tiếp tạo thành cơ sở của chiến lược lập phiên bản lồng léo được giải thích ngắn gọn.

Thay đổi tương thích

Khi chúng tôi thực hiện một thay đổi đối với hợp đồng dịch vụ mà không ảnh hưởng tiêu cực đến người tiêu dùng hiện tại thì bản thân thay đổi đó được coi là thay đổi tương thích.

GHI CHÚ

Trong cuốn sách này, thuật ngữ "thay đổi tương thích" mặc định ám chỉ khả năng tương thích ngược.

Khi được sử dụng để tham chiếu đến khả năng tương thích về phía trước, nó còn được coi là một thay đổi tương thích về phía trước.

Một ví dụ đơn giản về thay đổi tương thích là khi chúng tôi đặt thuộc tính `minOccurs` của một phần tử từ "1" thành "0", biến phần tử bắt buộc thành phần tử tùy chọn một cách hiệu quả, như thể hiện trong ví dụ 10.8.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://actioncon.com/schema/po"
  xmlns="http://actioncon.com/schema/po">
<xsd:teil phần tử="LineItem" type="LineItemType"/>
<xsd:complexType name="LineItemType">
  <xsd:trình tự>
    <xsd:element name="productID" type="xsd:string"/>
    <xsd:element name="productName" type="xsd:string"
      minOccurs="0"/>
    <xsd:teil phần tử="có sẵn" type="xsd:boolean"
      minOccurs="0"/>
```

274 Chương 10: API dịch vụ và tạo phiên bản hợp đồng với dịch vụ web và dịch vụ REST

```
</xsd:trình tự>
</xsd:complexType>
</xsd:lực đòn>
```

Ví dụ 10.8

Giá trị mặc định của thuộc tính `minOccurs` là "1". Do đó, vì thuộc tính này trước đây không có trong phần khai báo phần tử `ProductName` nên nó được coi là phần tử bắt buộc. Việc thêm cài đặt `minOccurs="0"` sẽ biến nó thành một phần tử tùy chọn, dẫn đến thay đổi tương thích. (Lưu ý rằng việc thực hiện thay đổi này đối với điều ra thông báo từ dịch vụ sẽ là một thay đổi không tương thích.)

Loại thay đổi này sẽ không ảnh hưởng đến các chương trình tiêu dùng hiện tại được sử dụng để gửi giá trị phần tử tới dịch vụ Web, cũng như không ảnh hưởng đến các chương trình tiêu dùng trong tương lai có thể được thiết kế để gửi phần tử đó một cách tùy ý.

Một ví dụ khác về thay đổi tương thích đã được cung cấp trước đó trong Ví dụ 10.3, khi trước tiên chúng tôi đã thêm phần khai báo phần tử có sẵn tùy chọn. Mặc dù chúng tôi đã mở rộng loại này với một thành phần hoàn toàn mới, nhưng vì nó là tùy chọn nên nó được coi là tương thích thay đổi.

Dưới đây là danh sách các thay đổi tương thích phổ biến:

- Thêm định nghĩa hoạt động WSDL mới và các định nghĩa thông báo liên quan
- Thêm một phương thức tiêu chuẩn mới vào tài nguyên REST hiện có
- Thêm một bộ tài nguyên REST mới
- Thay đổi số nhận dạng cho một tập hợp tài nguyên REST (bao gồm phân tách và hợp nhất các dịch vụ) bằng cách sử dụng mã phản hồi chuyển hướng để tạo điều kiện di chuyển REST người tiêu dùng dịch vụ cho các nhận dạng mới
- Thêm định nghĩa loại công WSDL mới và các định nghĩa hoạt động liên quan
- Thêm các định nghĩa dịch vụ và ràng buộc WSDL mới
- Mở rộng phương thức hợp đồng thống nhất hiện có theo cách có thể an toàn bị bỏ qua bởi các dịch vụ REST có thể dựa trên logic dịch vụ cũ (ví dụ: thêm "If-None-Match" làm tính năng của hoạt động HTTP GET để nếu dịch vụ bỏ qua nó, người tiêu dùng vẫn sẽ nhận được đại diện hiện tại và chính xác cho nguồn)
- Thêm phương thức hợp đồng thống nhất mới khi có phản hồi ngoại lệ cho dịch vụ không hiểu phương pháp sử dụng (và người tiêu dùng có thể phục hồi từ ngoại lệ này)

- Thêm một phần tử hoặc thuộc tính Lược đồ XML tùy chọn mới vào một thông báo sự định nghĩa
- Giảm mức độ chi tiết ràng buộc của một phần tử hoặc thuộc tính Lược đồ XML của một loại định nghĩa thông điệp được sử dụng cho thông điệp đầu vào
- Thêm ký tự đại diện Lược đồ XML mới vào loại định nghĩa thông báo
- Thêm xác nhận WS-Policy tùy chọn mới
- Thêm giải pháp thay thế Chính sách WS mới

Những thay đổi không tương thích

Nếu sau khi thay đổi, một hợp đồng không còn tương thích với người tiêu dùng thì hợp đồng đó được coi là đã nhận được một thay đổi không tương thích. Đây là những loại thay đổi có thể phá vỡ hợp đồng hiện có và do đó đặt ra nhiều thách thức nhất khi lập phiên bản.

GHI CHÚ

Theo mặc định, thuật ngữ "thay đổi không tương thích" cũng biểu thị khả năng tương thích ngược. Những thay đổi không tương thích ảnh hưởng đến khả năng tương thích về phía trước sẽ được coi là "những thay đổi không tương thích về phía trước".

Quay lại ví dụ của chúng tôi, nếu chúng tôi đặt thuộc tính minOccurs của một phần tử từ "0" thành bất kỳ số nào trên 0, thì chúng tôi sẽ đưa ra một thay đổi không tương thích cho các thông báo đầu vào, như trong Ví dụ 10.9:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://
actioncon.com/schema/po" xmlns="http://actioncon.com/schema /po">
<xsd:element name="LineItem" type="LineItemType"/>
<xsd:complexType name="LineItemType"> <xsd:sequence> <xsd:element
name="productID" type="xsd:string" /> <xsd:element
name="productName"
type="xsd:string"

minOccurs="3"/>
<xsd:tên phần tử="có sẵn" type="xsd:boolean"
minOccurs="3"/> </
xsd:sequence> </
xsd:complexType> </xsd:schema>
```

Ví dụ 10.9 Việc

tăng giá trị thuộc tính minOccurs của bất kỳ khai báo phần tử đã thiết lập nào sẽ tự động là một thay đổi không tương thích.

276 Chương 10: API dịch vụ và tạo phiên bản hợp đồng với dịch vụ web và dịch vụ REST

Những gì trước đây là một phần tử tùy chọn bây giờ là bắt buộc. Điều này chắc chắn sẽ ảnh hưởng đến những người tiêu dùng hiện tại không được thiết kế để tuân thủ ràng buộc mới này, bởi vì việc thêm một yếu tố bắt buộc mới sẽ tạo ra một ràng buộc bắt buộc đối với hợp đồng.

Những thay đổi không tương thích phổ biến bao gồm:

- Đổi tên định nghĩa hoạt động WSDL hiện có
- Loại bỏ định nghĩa hoạt động WSDL hiện có
- Thay đổi MEP của định nghĩa hoạt động WSDL hiện có
- Thêm thông báo lỗi vào định nghĩa hoạt động WSDL hiện có
- Thêm một phần tử hoặc khai báo thuộc tính Lược đồ XML bắt buộc mới vào một định nghĩa tin nhắn
- Tăng mức độ chi tiết ràng buộc của một phần tử hoặc thuộc tính Lược đồ XML khai báo định nghĩa thông điệp
- Đổi tên một thành phần hoặc thuộc tính Lược đồ XML tùy chọn hoặc bắt buộc trong một thông báo sự định nghĩa
- Xóa phần tử hoặc thuộc tính ký tự đại diện Lược đồ XML tùy chọn hoặc bắt buộc từ một định nghĩa thông điệp
- Thêm xác nhận hoặc biểu thức WS-Policy bắt buộc mới
- Thêm biểu thức WS-Policy mới có thể bỏ qua (hầu hết thời gian)

Những thay đổi không tương thích có xu hướng gây ra hầu hết các thách thức với việc lập phiên bản hợp đồng dịch vụ.

10.3 Cân nhắc về tính tương thích của dịch vụ REST

Các dịch vụ REST trong kho dịch vụ nhất định thường chia sẻ hợp đồng thống nhất cho mọi tài nguyên, bao gồm các phương thức và loại phương tiện thống nhất. Các loại phương tiện truyền thông tương tự được sử dụng trong cả yêu cầu và phản hồi, đồng thời các khía cạnh hợp đồng thống nhất mới được sử dụng lại thường xuyên hơn nhiều so với những gì chúng được thêm vào. Việc nhấn mạnh vào việc tái sử dụng hợp đồng dịch vụ trong kho dịch vụ tuân thủ REST dẫn đến nhu cầu nêu bật một số cân nhắc đặc biệt, vì những thay đổi đối với hợp đồng thống nhất sẽ tự động tác động đến một loạt người tiêu dùng dịch vụ vì:

- Các phương thức hợp đồng thống nhất được chia sẻ bởi tất cả các dịch vụ.
- Các loại phương tiện hợp đồng thống nhất được chia sẻ bởi cả dịch vụ và dịch vụ người tiêu dùng.

Kết quả là, cả khả năng tương thích ngược và khả năng tương thích chuyển tiếp đều được cân nhắc gần như quan trọng như nhau.

MẪU SOA

Các hợp đồng dịch vụ sử dụng mẫu Schema Centralization [356] mà không nhất thiết phải tuân thủ REST thường sẽ cần phải áp đặt một quan điểm cứng nhắc tương tự về khả năng tương thích xuôi và tương thích ngược.

Các phương pháp hợp đồng thống nhất hệ thống hóa các loại tương tác có thể xảy ra giữa dịch vụ và người tiêu dùng. Ví dụ: GET codifi es “tìm nạp một số dữ liệu” trong khi PUT codifi es “lưu trữ một số dữ liệu.”

Bởi vì các loại tương tác xảy ra giữa các dịch vụ REST trong cùng một tồn kho dịch vụ có xu hướng tương đối hạn chế và ổn định, các phương pháp thường sẽ thay đổi ở mức thấp so với các loại phương tiện hoặc tài nguyên. Các vấn đề tương thích thường liên quan đến một tập hợp các phương pháp được phép chỉ được thay đổi sau khi xem xét cẩn thận từng trường hợp cụ thể.

Một ví dụ về thay đổi tương thích đối với HTTP là việc bổ sung các tiêu đề If-None-Match cho các yêu cầu GET. Nếu người tiêu dùng dịch vụ biết phiên bản cuối cùng (hoặc etag) của tài nguyên mà nó đã tìm nạp, thì nó có thể đặt yêu cầu GET có điều kiện. Tiêu đề If -None-Match cho phép người tiêu dùng tuyên bố rằng yêu cầu GET không nên được thực thi nếu phiên bản của tài nguyên vẫn giống như phiên bản tìm nạp cuối cùng của người tiêu dùng. Thay vào đó, nó sẽ trả về phản hồi GET bình thường, mặc dù nó sẽ thực hiện như vậy ở chế độ không tối ưu.

Một ví dụ về thay đổi không tương thích đối với HTTP là việc bổ sung tiêu đề Máy chủ được sử dụng để hỗ trợ nhiều máy chủ Web. HTTP/1.0 không yêu cầu đưa tên dịch vụ vào thông báo yêu cầu, nhưng HTTP/1.1 lại yêu cầu điều này. Nếu tiêu đề Máy chủ đặc biệt bị thiếu, các dịch vụ HTTP/1.1 phải từ chối yêu cầu vì yêu cầu đó không tốt.

hình thành. Tuy nhiên, các dịch vụ HTTP/1.1 cũng được yêu cầu phải tương thích ngược, vì vậy nếu một yêu cầu HTTP/1.0 đến dịch vụ REST thì yêu cầu đó vẫn sẽ được xử lý theo quy tắc HTTP/1.0.

Các loại phương tiện hợp đồng thống nhất mã hóa thêm các loại thông tin có thể được trao đổi giữa các dịch vụ REST và người tiêu dùng. Như đã nêu trước đây, các loại phương tiện truyền thông có xu hướng thay đổi với tốc độ nhanh hơn các phương thức HTTP trong hợp đồng thống nhất; tuy nhiên, loại phương tiện vẫn thay đổi chậm hơn tài nguyên. Thay đổi tương thích là nhiều hơn một

278 Chương 10: API dịch vụ và lập phiên bản hợp đồng với dịch vụ web và dịch vụ REST

quan tâm trực tiếp đến các loại phương tiện truyền thông và chúng ta có thể rút ra một số quy tắc chung hơn về cách để đổi phó với họ.

Ví dụ: nếu bộ tạo thông báo cho bộ xử lý thông báo biết rằng nó phù hợp với một loại phương tiện cụ thể, bộ xử lý thường không cần biết phiên bản nào của lược đồ đã được sử dụng và bộ xử lý cũng không cần phải được xây dựng chồng lại cùng một phiên bản của lược đồ. Bộ xử lý hy vọng rằng tất cả các phiên bản của lược đồ cho một loại phương tiện cụ thể sẽ tương thích thuận và tương thích ngược với loại mà nó được phát triển để hỗ trợ. Tương tự như vậy, máy phát điện mong đợi rằng khi nó tạo ra một thông báo phù hợp với một phiên bản lược đồ cụ thể thì tất cả bộ xử lý tin nhắn sẽ hiểu nó.

Khi thực hiện các thay đổi không tương thích đối với lược đồ, thông thường cần phải có mã định danh loại phương tiện mới để đảm bảo rằng:

- Bộ xử lý có thể quyết định cách phân tích tài liệu dựa trên loại phương tiện nhận dạng
- Dịch vụ và người tiêu dùng có thể thương lượng về một loại phương tiện truyền thông cụ thể sẽ được được bộ xử lý hiểu khi thông báo được tạo ra

Đảm phán nội dung là giải pháp dự phòng cuối cùng để đảm bảo tính tương thích trong kho dịch vụ tuân thủ REST. Đối với tương tác tìm nạp, điều này thường liên quan đến việc người tiêu dùng chỉ ra cho dịch vụ loại phương tiện nào nó có thể hỗ trợ và dịch vụ trả về nhiều nhất loại thích hợp mà nó hỗ trợ. Cơ chế này cho phép thực hiện các thay đổi không tương thích đối với các loại phương tiện theo yêu cầu.

GHI CHÚ

Một cách để hiểu rõ hơn các vấn đề về phiên bản liên quan đến loại phương tiện là xem cách chúng được sử dụng trong HTML. Một ví dụ về thay đổi tương thích với HTML mà không dẫn đến nhu cầu về loại phương tiện mới là việc bổ sung phần tử abbr vào phiên bản 4.0 của ngôn ngữ HTML. Phần tử này cho phép các bộ xử lý tài liệu HTML mới hỗ trợ di chuột qua để mở rộng các chữ viết tắt trên trang web và hỗ trợ khả năng truy cập trang tốt hơn. Các bộ xử lý cũ bỏ qua phần mở rộng một cách an toàn nhưng sẽ tiếp tục hiển thị chính xác chữ viết tắt.

Một ví dụ về thay đổi không tương thích với HTML và yêu cầu loại phương tiện mới là việc chuyển đổi HTML 4.0 sang XML (dẫn đến phiên bản 1.0 của XHTML). Loại phương tiện cho phiên bản SGML truyền thống vẫn là văn bản/html, trong khi phiên bản XML trở thành application/xhtml+xml. Điều này cho phép đảm phán nội dung xảy ra giữa hai bên

loại và để bộ xử lý chọn chiến lược xác thực và phân tích cú pháp chính xác dựa trên loại được dịch vụ chỉ định.

Một số thay đổi không tương thích cũng đã được thực hiện đối với HTML mà không thay đổi loại phương tiện. HTML 4.0 không còn dùng các phần tử APPLET, BASEFONT, CENTER, DIR, FONT, ISINDEX, MENU, S, STRIKE và U để thay thế bằng các phần tử mới hơn. Những phần tử này phải tiếp tục được hiểu nhưng việc sử dụng chúng trong các tài liệu HTML đang bị loại bỏ dần. HTML 4.0 đã khai Listing, PLAINTEXT và XMP trả nên lỗi thời. Những phần tử này không nên được sử dụng trong tài liệu HTML 4.0 và không cần phải hiểu nữa.

Việc loại bỏ các phần tử trong một khoảng thời gian dài và cuối cùng xác định chúng là lỗi thời khi chúng không còn được sử dụng bởi các dịch vụ hiện tại hoặc người tiêu dùng là một kỹ thuật có thể được sử dụng cho các loại phương tiện REST để cập nhật dần dần lược đồ mà không cần phải thay đổi phương tiện kiểu.

10.4 Số nhận dạng phiên bản

Một trong những mẫu thiết kế cơ bản nhất liên quan đến thiết kế hợp đồng dịch vụ Web là mẫu Nhận dạng Phiên bản. Về cơ bản, nó ủng hộ rằng số phiên bản phải được thể hiện rõ ràng, không chỉ ở cấp độ hợp đồng mà còn ở cả các phiên bản của các lược đồ làm nền tảng cho các định nghĩa thông điệp.

Bước đầu tiên để thiết lập một chiến lược tạo phiên bản hiệu quả là quyết định một phương tiện chung để các phiên bản được xác định và thể hiện trong các hợp đồng dịch vụ Web.

Các phiên bản hầu như luôn được thông báo bằng số phiên bản. Định dạng phổ biến nhất là số thập phân, sau là dấu chấm và sau đó là số thập phân khác, như được hiển thị ở đây:

phiên bản="2.0"

Đôi khi, bạn sẽ thấy các cặp dấu chấm cộng với số thập phân bổ sung dẫn đến kết quả chi tiết hơn. số phiên bản như thế này:

phiên bản="2.0.1.1"

Ý nghĩa điển hình gắn liền với những con số này là thước đo hoặc ý nghĩa của sự thay đổi. Việc tăng số thập phân đầu tiên thường biểu thị sự thay đổi (hoặc nâng cấp) phiên bản chính trong phần mềm, trong khi số thập phân sau giai đoạn đầu tiên thường biểu thị các mức độ thay đổi phiên bản nhỏ khác nhau.

280 Chương 10: API dịch vụ và lập phiên bản hợp đồng với dịch vụ web và dịch vụ REST

Từ góc độ tương thích, chúng ta có thể liên kết ý nghĩa bổ sung với những con số này. Cụ thể, quy ước sau đây đã xuất hiện trong ngành:

- Phiên bản phụ được kỳ vọng sẽ tương thích ngược với các phiên bản phụ khác được liên kết với phiên bản chính. Ví dụ: phiên bản 5.2 của chương trình phải hoàn toàn tương thích ngược với các phiên bản 5.0 và 5.1.
- Một phiên bản chính thường được cho là sẽ phá vỡ khả năng tương thích ngược với các chương trình thuộc phiên bản chính khác. Điều này có nghĩa là phiên bản chương trình 5.0 là dự kiến sẽ không tương thích ngược với phiên bản 4.0.

GHI CHÚ

Số phiên bản "bản vá" thứ ba đôi khi cũng được sử dụng để thể hiện những thay đổi vừa tương thích về phía trước vừa tương thích ngược. Thông thường, các phiên bản này chỉ nhằm mục đích làm rõ lược đồ hoặc để khắc phục các sự cố với lược đồ được phát hiện sau khi triển khai. Ví dụ: phiên bản 5.2.1 dự kiến sẽ hoàn toàn tương thích với phiên bản 5.2.0 nhưng có thể được thêm vào nhằm mục đích làm rõ.

Quy ước biểu thị tính tương thích thông qua số phiên bản chính và phụ này được gọi là **đảm bảo tính tương thích**. Một cách tiếp cận khác, được gọi là "**số lượng làm việc**," sử dụng số phiên bản để truyền đạt nỗ lực đã thực hiện thay đổi. Một **Việc tăng** phiên bản nhỏ cho thấy một nỗ lực khiêm tốn và việc tăng phiên bản chính có thể dự đoán được sẽ đòi hỏi rất nhiều công việc.

Hai quy ước này có thể được kết hợp và thường là như vậy. Kết quả thường là số phiên bản tiếp tục thể hiện khả năng tương thích như đã giải thích trước đó, nhưng đôi khi chúng tăng lên vài chữ số, tùy thuộc vào mức độ nỗ lực đã bỏ ra. **mỗi** phiên bản.

Có nhiều tùy chọn cú pháp khác nhau để thể hiện số phiên bản. Ví dụ, bạn có thể nhận thấy rằng câu lệnh khai báo bắt đầu một tài liệu XML có thể chứa một số biểu thị phiên bản đặc tả XML đang được sử dụng:

```
<?xml phiên bản="1.0"?>
```

Thuộc tính phiên bản tương tự có thể được sử dụng với phần tử `xsd:schema` gốc , như sau:

```
<xsd:schema version="2.0" ...>
```

Bạn có thể tạo thêm một biến thể tùy chỉnh của thuộc tính này bằng cách gán nó cho bất kỳ thành phần nào mà bạn xác định (trong trường hợp đó bạn không bắt buộc phải đặt tên cho thuộc tính là "phiên bản").

```
<Phiên bản LineItem="2.0">
```

Một cách tiếp cận tùy chỉnh thay thế là nhúng số phiên bản chính vào vùng tên hoặc mã định danh loại phương tiện, như được hiển thị ở đây:

```
<LineItem xmlns="http://actioncon.com/schema/po/v2">
```

hoặc

```
application/vnd.com.actioncon.po.v2+xml
```

Lưu ý rằng việc sử dụng các giá trị ngày tháng trong các khung gian tên khi lập phiên bản các lược đồ XML đã trở thành một quy ước chung, như sau:

```
<LineItem xmlns="http://actioncon.com/schema/po/2010/09">
```

Trong trường hợp này, ngày thay đổi đóng vai trò là mã nhận dạng phiên bản chính. Để giữ biểu thức của các phiên bản định nghĩa Lược đồ XML phù hợp với các phiên bản định nghĩa WSDL, chúng tôi sử dụng số phiên bản thay vì giá trị ngày trong các ví dụ sắp tới. Tuy nhiên, khi làm việc trong môi trường có các định nghĩa Lược đồ XML được tách riêng được sở hữu như một phần của kiến trúc dữ liệu độc lập, không có gì lạ khi các mã định danh phiên bản lược đồ khác với các mã định danh được sử dụng bởi các định nghĩa WSDL.

Bất kể bạn chọn tùy chọn nào, điều quan trọng là phải xem xét mẫu Phiên bản chuẩn quy định rằng việc thể hiện thông tin phiên bản phải được chuẩn hóa trên tất cả các hợp đồng dịch vụ trong ranh giới của kho dịch vụ. Trong môi trường lớn hơn, điều này thường đòi hỏi một cơ quan trung ương có thể đảm bảo

tuyên tính, tính nhất quán và chất lượng mô tả của thông tin phiên bản. Những loại này các quy ước chuyển sang cách thể hiện thông tin châm dứt dịch vụ, như được khám phá sâu hơn trong Chương 23 về Thiết kế và lập phiên bản hợp đồng dịch vụ web cho SOA.

MẪU SOA

Tất nhiên, bạn cũng có thể được yêu cầu làm việc với các lược đồ bên thứ ba và các định nghĩa WSDL có thể đã triển khai các quy ước lập phiên bản của riêng họ. Trong trường hợp này, mức độ mà mẫu Phiên bản Canonical [327] có thể được áp dụng sẽ bị hạn chế.

282 Chương 10: API dịch vụ và lập phiên bản hợp đồng với dịch vụ web và dịch vụ REST

10.5 Chiến lược tạo phiên bản

Không có cách tiếp cận phiên bản nào phù hợp với tất cả mọi người. Bởi vì việc lập phiên bản thể hiện một giai đoạn liên quan đến quản trị trong vòng đời tổng thể của một dịch vụ nên đó là một thông lệ.

tuân theo các quy ước, ưu tiên và yêu cầu riêng biệt với bất kỳ doanh nghiệp.

Mặc dù thực tế không có kỹ thuật tạo phiên bản nào cho WSDL, Lược đồ XML, và nội dung WS-Policy bao gồm các hợp đồng dịch vụ Web, một số nội dung phổ biến và các phương pháp tiếp cận phiên bản được ủng hộ đã xuất hiện, mỗi phương pháp đều có những lợi ích và sự đánh đổi.

Trong phần này, chúng tôi chỉ ra ba chiến lược phổ biến sau:

- Nghiêm ngặt - Mọi thay đổi tương thích hoặc không tương thích đều dẫn đến phiên bản mới của hợp đồng dịch vụ. Cách tiếp cận này không hỗ trợ lùi hoặc tiến khả năng tương thích.
- Linh hoạt - Mọi thay đổi không tương thích đều dẫn đến phiên bản mới của hợp đồng dịch vụ và hợp đồng được thiết kế để hỗ trợ khả năng tương thích ngược nhưng không hỗ trợ chuyển tiếp khả năng tương thích.
- Lỏng lẻo - Mọi thay đổi không tương thích đều dẫn đến phiên bản mới của hợp đồng dịch vụ và hợp đồng được thiết kế để hỗ trợ khả năng tương thích ngược và chuyển tiếp khả năng tương thích.

Những chiến lược này được giải thích riêng lẻ trong các phần sắp tới.

Chiến lược chặt chẽ (Thay đổi mới, Hợp đồng mới)

Cách tiếp cận đơn giản nhất đối với việc lập phiên bản hợp đồng dịch vụ Web là yêu cầu phát hành một phiên bản mới của hợp đồng bất cứ khi nào có bất kỳ loại thay đổi nào được thực hiện đối với bất kỳ phần nào của hợp đồng.

Điều này thường được triển khai bằng cách thay đổi giá trị vùng tên đích của định nghĩa WSDL (và có thể cả định nghĩa Lược đồ XML) mỗi khi thực hiện một thay đổi tương thích hoặc không tương thích đối với nội dung WSDL, Lược đồ XML hoặc Chính sách WS liên quan đến hợp đồng. Không gian tên được sử dụng để nhận dạng phiên bản thay vì thuộc tính phiên bản vì việc thay đổi giá trị không gian tên sẽ tự động buộc thay đổi trong tất cả các chương trình người tiêu dùng cần truy cập vào phiên bản mới của lược đồ xác định loại thông báo.

Cách tiếp cận “siêu nghiêm ngặt” này không thực sự thực tế nhưng lại an toàn nhất và đôi khi được bảo đảm khi có những tác động pháp lý đối với việc sửa đổi hợp đồng dịch vụ Web, chẳng hạn như như khi các hợp đồng được công bố để trao đổi dữ liệu giữa các tổ chức nhất định. Bởi vì cả những thay đổi tương thích và không tương thích đều sẽ dẫn đến một phiên bản hợp đồng mới, điều này cách tiếp cận này không hỗ trợ khả năng tương thích ngược cũng như không tương thích về phía trước.

Ưu và nhược điểm

Lợi ích của chiến lược này là bạn có toàn quyền kiểm soát sự phát triển của hợp đồng dịch vụ và vì khả năng tương thích ngược và xuôi bị bỏ qua một cách có chủ ý nên bạn không cần phải lo lắng về tác động của bất kỳ thay đổi cụ thể nào.

(vì tất cả các thay đổi đều phá vỡ hợp đồng một cách hiệu quả).

Mặt khác, bằng cách buộc một không gian tên mới vào hợp đồng với mỗi thay đổi, bạn đang đảm bảo rằng tất cả người dùng dịch vụ hiện tại sẽ không còn tương thích với bất kỳ phiên bản mới nào của hợp đồng. Người tiêu dùng sẽ chỉ có thể tiếp tục liên lạc với dịch vụ Web trong khi hợp đồng cũ vẫn có sẵn cùng với phiên bản mới hoặc cho đến khi bản thân người tiêu dùng được cập nhật để tuân thủ hợp đồng mới.

Do đó, cách tiếp cận này sẽ làm tăng gánh nặng quản trị các dịch vụ riêng lẻ và sẽ yêu cầu các chiến lược chuyển đổi thận trọng. Việc có hai hoặc nhiều phiên bản của cùng một dịch vụ cùng tồn tại cùng một lúc có thể trở thành một yêu cầu chung mà cơ sở hạ tầng kiểm kê dịch vụ hỗ trợ cần phải được chuẩn bị.

Chiến lược linh hoạt (Khả năng tương thích ngược)

Một cách tiếp cận phổ biến được sử dụng để cân bằng những cân nhắc thực tế với nỗ lực giảm thiểu tác động của những thay đổi đối với hợp đồng dịch vụ Web là cho phép những thay đổi tương thích xảy ra mà không cần buộc phải có phiên bản hợp đồng mới, đồng thời không có gắng hỗ trợ chuyển tiếp. khả năng tương thích cả.

Điều này có nghĩa là bất kỳ thay đổi tương thích ngược nào đều được coi là an toàn vì nó sẽ kéo dài hoặc tăng cường hợp đồng đã thiết lập mà không ảnh hưởng đến bất kỳ dịch vụ nào. người tiêu dùng hiện tại. Một ví dụ phổ biến về điều này là thêm một thao tác mới vào định nghĩa WSDL hoặc thêm một khai báo phần tử tùy chọn vào định nghĩa lược đồ của thông báo.

Giống như chiến lược Nghiêm, bất kỳ thay đổi nào vi phạm hợp đồng hiện tại sẽ dẫn đến phiên bản hợp đồng mới, thường được triển khai bằng cách thay đổi giá trị vùng tên đích của định nghĩa WSDL và có thể cả định nghĩa Lược đồ XML.

284 Chương 10: API dịch vụ và tạo phiên bản hợp đồng với dịch vụ web và dịch vụ REST

Ưu và nhược điểm

Ưu điểm chính của phương pháp này là nó có thể được sử dụng để đáp ứng nhiều thay đổi khác nhau trong khi vẫn duy trì nhất quán khả năng tương thích ngược của hợp đồng. Tuy nhiên, khi những thay đổi tương thích được thực hiện, những thay đổi này sẽ trở thành vĩnh viễn và không thể được đảo ngược mà không đưa ra một thay đổi không tương thích. Do đó, cần có một quy trình quản trị trong đó mỗi thay đổi được đề xuất đều được đánh giá để các hợp đồng không trở nên quá cồng kềnh hoặc phức tạp. Đây là điều cần cân nhắc đặc biệt quan trọng đối với các dịch vụ bắt khả tri được tái sử dụng nhiều.

Chiến lược lồng lèo (Khả năng tương thích ngược và tiến)

Giống như hai cách tiếp cận trước, chiến lược này đòi hỏi những thay đổi không tương thích dẫn đến một phiên bản hợp đồng dịch vụ mới. Sự khác biệt ở đây là ở cách hợp đồng dịch vụ được thiết kế ban đầu.

Thay vì đáp ứng các yêu cầu trao đổi dữ liệu đã biết, các tính năng đặc biệt từ các ngôn ngữ WSDL, Lược đồ XML và Chính sách WS được sử dụng để làm cho các phần của hợp đồng có thể mở rộng về bản chất để chúng vẫn có thể hỗ trợ nhiều loại trong tương lai, yêu cầu trao đổi dữ liệu chưa biết. Ví dụ:

- Giá trị thuộc tính AnyType được cung cấp bởi ngôn ngữ WSDL 2.0 cho phép một thông báo để bao gồm bất kỳ tài liệu XML hợp lệ nào.
- Các ký tự đại diện Lược đồ XML có thể được sử dụng để cho phép một loạt dữ liệu chưa biết được truyền trong định nghĩa thông điệp.
- Các xác nhận chính sách có thể bị bỏ qua có thể được xác định để truyền đạt các đặc điểm dịch vụ mà người tiêu dùng trong tương lai có thể tùy ý thửa nhặt.

Những tính năng này và các tính năng khác liên quan đến khả năng tương thích chuyển tiếp sẽ được thảo luận trong Thiết kế hợp đồng dịch vụ web và lập phiên bản cho SOA.

Ưu và nhược điểm

Thực tế là các ký tự đại diện cho phép nội dung không xác định được chuyển qua các hợp đồng dịch vụ Web mang lại cơ hội liên tục để mở rộng hơn nữa phạm vi nội dung dữ liệu và phần tử thông báo được chấp nhận. Một khác, việc sử dụng ký tự đại diện sẽ tự nhiên dẫn đến các hợp đồng dịch vụ mơ hồ và quá thô sơ, đặt gánh nặng xác thực lên logic dịch vụ cơ bản.

Tóm tắt chiến lược được

cung cấp trong Bảng 10.1 là bản tóm tắt tổng quát về cách so sánh ba chiến lược dựa trên ba đặc điểm cơ bản.

		Chiến lược	
		Nghiêm ngặt	Linh hoạt
Sự nghiêm khắc	Cao	Trung bình	Thấp
Tác động quản trị	Cao	Trung bình	Cao
Độ phức tạp	Thấp	Trung bình	Cao

Bảng 10.1

So sánh chung về ba chiến lược phiên bản.

Ba đặc điểm được sử dụng trong bảng này để làm cơ sở cho sự so sánh này như sau:

- Tính nghiêm ngặt - Tính cứng nhắc của các lựa chọn phiên bản hợp đồng. Cách tiếp cận nghiêm ngặt rõ ràng là cứng nhắc nhất trong các quy tắc phiên bản của nó, trong khi chiến lược Loose cung cấp phạm vi tùy chọn phiên bản rộng nhất do nó phụ thuộc vào các ký tự đại diện.
- Tác động quản trị - Gánh nặng quản trị do một chiến lược đặt ra.
Cả hai cách tiếp cận Nghiêm ngặt và Lỏng lẻo đều làm tăng tác động quản trị nhưng đối với các mục đích khác nhau lý do. Chiến lược nghiêm ngặt yêu cầu phát hành thêm nhiều phiên bản hợp đồng mới, tác động đến người tiêu dùng và cơ sở hạ tầng xung quanh, trong khi Loose Cách tiếp cận này đưa ra khái niệm về các bộ thông báo chưa biết cần được cung cấp riêng biệt thông qua lập trình tùy chỉnh.
- Độ phức tạp - Độ phức tạp tổng thể của quá trình tạo phiên bản. Do việc sử dụng ký tự đại diện và dữ liệu tin nhắn không xác định, chiến lược Loose có tiềm năng phức tạp cao nhất, trong khi các quy tắc đơn giản tạo thành nền tảng của Strict cách tiếp cận làm cho nó trở thành lựa chọn đơn giản nhất.

Trong suốt quá trình so sánh này, chiến lược Linh hoạt cung cấp một cách tiếp cận thẻ hiện mức độ nghiêm ngặt, nỗ lực quản trị và mức độ phức tạp tổng thể ở mức trung bình nhất quán.

286 Chương 10: API dịch vụ và lập phiên bản hợp đồng với dịch vụ web và dịch vụ REST

10.6 Những cân nhắc về phiên bản dịch vụ REST

Các dịch vụ REST chia sẻ cùng một hợp đồng thống nhất sẽ duy trì các thông số kỹ thuật được phiên bản riêng biệt cho các mục sau:

- Số phiên bản hoặc thông số kỹ thuật của cú pháp nhận dạng tài nguyên (theo "Yêu cầu nhận xét 6986 - Mã định danh tài nguyên thống nhất (URI): Cú pháp chung" sự chỉ rõ)
- Đặc tả việc thu thập các phương pháp pháp lý, mã trạng thái và các chi tiết giao thức tương tác khác (theo "Yêu cầu Nhận xét 2616 - Truyền Siêu văn bản Giao thức - đặc tả HTTP/1.1")
- Thông số kỹ thuật riêng cho các loại phương tiện truyền thông hợp pháp (ví dụ: HTML 4.01 và "Yêu cầu nhận xét 4287 - Đặc tả Định dạng phân phối nguyên tử")
- Các quy định riêng cho các hợp đồng dịch vụ sử dụng mã định danh nguồn lực pháp lý cú pháp, phương thức và loại phương tiện

Mỗi phần của hợp đồng thống nhất được quy định và phiên bản độc lập với những phần khác. Việc thay đổi bất kỳ một thông số kỹ thuật nào thường không yêu cầu một thông số kỹ thuật khác phải được cập nhật hoặc phiên bản. Tương tự như vậy, việc thay đổi bất kỳ khía cạnh hợp đồng thống nhất nào các thông số kỹ thuật không yêu cầu thay đổi hợp đồng dịch vụ riêng lẻ hoặc thay đổi đối với số phiên bản của họ.

Điểm cuối cùng này mâu thuẫn với một số chiến lược tạo phiên bản thông thường. Người ta có thể mong đợi rằng nếu lược đồ được sử dụng trong hợp đồng dịch vụ thay đổi thì hợp đồng dịch vụ sẽ cần phải được sửa đổi. Tuy nhiên, với các dịch vụ REST có xu hướng duy trì cả khả năng tương thích xuôi và khả năng tương thích ngược. Nếu người tiêu dùng dịch vụ REST gửi một tin nhắn phù hợp với lược đồ mới hơn thì dịch vụ đó có thể xử lý nó như nếu nó phù hợp với lược đồ cũ hơn. Nếu khả năng tương thích giữa các lược đồ này được duy trì thì dịch vụ sẽ hoạt động bình thường. Tương tự như vậy, nếu dịch vụ trả về một thông báo tới người tiêu dùng tuân theo lược đồ cũ, người tiêu dùng dịch vụ mới hơn vẫn có thể xử lý tin nhắn một cách chính xác.

Hợp đồng dịch vụ REST chỉ cần xem xét trực tiếp việc tạo phiên bản của hợp đồng thống nhất khi các loại phương tiện được sử dụng không còn được dùng nữa hoặc khi lược đồ tiến triển như vậy cho đến nay các yếu tố và thuộc tính mà dịch vụ phụ thuộc vào đang dần trở nên lỗi thời. Khi điều này xảy ra, hợp đồng dịch vụ cần được cập nhật và cùng với đó, logic dịch vụ cơ bản xử lý các loại phương tiện.



Phần III

Phụ lục

Phụ lục A: Tài liệu tham khảo về Nguyên tắc Định hướng Dịch vụ

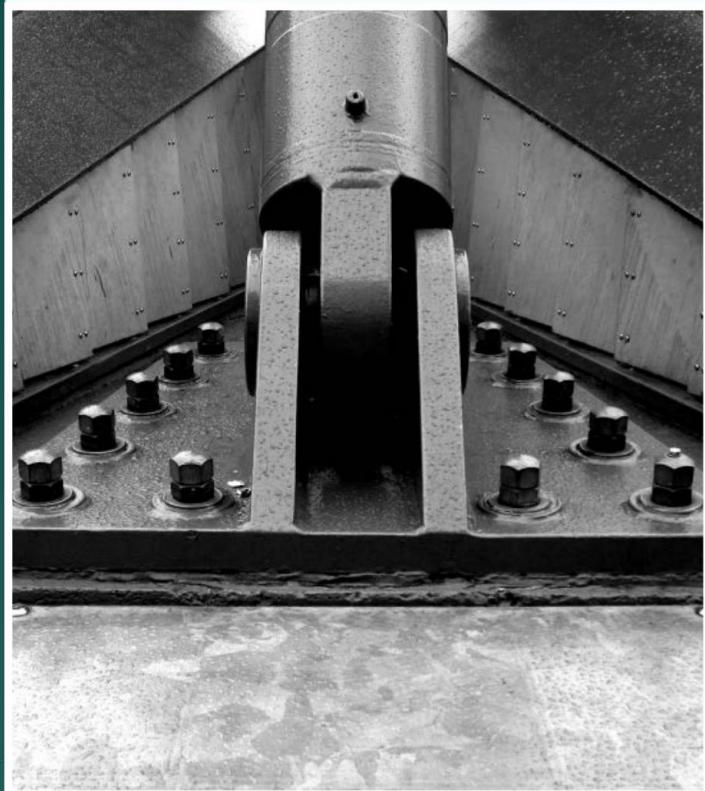
Phụ lục B: Tham khảo ràng buộc REST

Phụ lục C: Tham khảo các mẫu thiết kế SOA

Phụ lục D: Tuyên bố SOA được chú thích

Trang này có ý đẻ trống

Phụ lục A



Nguyên tắc định hướng dịch vụ

Thẩm quyền giải quyết

Phụ lục này sẽ cung cấp các khái niệm và định nghĩa, mô tả cách thức và mục tiêu của công nghệ dịch vụ. Tất cả các khái niệm và định nghĩa đều được bổ sung và cập nhật liên tục.

Mỗi bảng hồ sơ chứa các phần sau:

- Định nghĩa ngắn gọn - Một định nghĩa ngắn gọn, đơn giản nhằm thiết lập nền tảng mục đích tinh thần của nguyên tắc.
- Định nghĩa dài - Một mô tả dài hơn về nguyên tắc cung cấp nhiều chi tiết hơn như đến những gì nó dự định đạt được.
- Mục tiêu - Danh sách các mục tiêu thiết kế cụ thể được mong đợi từ việc áp dụng nguyên tắc. Về cơ bản, danh sách này cung cấp kết quả cuối cùng của nguyên tắc hiện thực hóa.
- Đặc điểm thiết kế - Danh sách các đặc điểm thiết kế cụ thể có thể thực hiện được thông qua việc áp dụng nguyên tắc. Điều này cung cấp một số hiểu biết sâu sắc về cách thức các nguyên tắc kết thúc việc định hình dịch vụ.
- Yêu cầu thực hiện - Danh sách các điều kiện tiên quyết chung để áp dụng hiệu quả nguyên tắc thiết kế. Chúng có thể bao gồm từ công nghệ đến tổ chức yêu cầu.

Lưu ý rằng các bảng tóm tắt này chỉ cung cấp phiên bản tóm tắt của các nguyên tắc. Bao quát đầy đủ tóm tắt của các nguyên tắc thiết kế định hướng dịch vụ, bao gồm các nghiên cứu điển hình, được cung cấp trong cuốn sách Nguyên tắc thiết kế dịch vụ SOA.

Để biết thêm thông tin về chủ đề này và các tựa sách khác trong Chuỗi Công nghệ Dịch vụ Prentice Hall của Thomas Erl liên quan đến định , hãy truy cập www.servicetechbooks.com . Nội dung tóm tắt các chủ đề hướng dịch vụ cũng có thể được tìm thấy trực tuyến tại www.serviceorientation.com .

Hợp đồng dịch vụ được tiêu chuẩn hóa	
Định nghĩa ngắn	"Dịch vụ chia sẻ hợp đồng tiêu chuẩn hóa."
Độ phân giải dài	"Các dịch vụ trong cùng một kho dịch vụ đều tuân thủ các tiêu chuẩn thiết kế hợp đồng giống nhau."
Bàn thang	<ul style="list-style-type: none"> Để kích hoạt các dịch vụ có mức độ tương tác tự nhiên có ý nghĩa trong phạm vi kho dịch vụ. Cái này giảm nhu cầu chuyển đổi dữ liệu vì tính nhất quán mô hình dữ liệu được sử dụng để trao đổi thông tin. Để cho phép mục đích và khả năng của dịch vụ được nâng cao hơn được hiểu một cách dễ dàng và trực quan. Sự nhất quán với điều đó Chức năng dịch vụ được thể hiện thông qua hợp đồng dịch vụ tăng khả năng diễn giải và khả năng dự đoán tổng thể của các điểm cuối dịch vụ trong toàn bộ kho dịch vụ. <p>Lưu ý rằng những mục tiêu này còn được hỗ trợ thêm bởi các nguyên tắc định hướng dịch vụ khác.</p>
Thiết kế Đặc trưng	<ul style="list-style-type: none"> Hợp đồng dịch vụ (bao gồm giao diện kỹ thuật hoặc một hoặc nhiều tài liệu mô tả dịch vụ) được cung cấp cùng với dịch vụ. Hợp đồng dịch vụ được chuẩn hóa thông qua việc áp dụng tiêu chuẩn thiết kế.
Thực hiện Yêu cầu	<p>Thực tế là các hợp đồng cần phải được tiêu chuẩn hóa có thể đưa ra các yêu cầu thực hiện đáng kể đối với các tổ chức chưa có lịch sử sử dụng tiêu chuẩn.</p> <p>Ví dụ:</p> <ul style="list-style-type: none"> Các tiêu chuẩn và quy ước thiết kế cần phải được đưa vào một cách lý tưởng đặt trước khi cung cấp bất kỳ dịch vụ nào để đảm bảo tiêu chuẩn hóa có phạm vi thích hợp. (Đối với những tổ chức đã sản xuất các dịch vụ Web đặc biệt, trang bị theo phong cách cổ điển chiến lược có thể cần phải được sử dụng.) Cần đưa ra các quy trình chính thức để đảm bảo rằng các dịch vụ được mô hình hóa và thiết kế nhất quán, kết hợp được chấp nhận nguyên tắc thiết kế, quy ước và tiêu chuẩn.

• Bởi vì việc đạt được các hợp đồng dịch vụ được tiêu chuẩn hóa nói chung đòi hỏi cách tiếp cận “ưu tiên hợp đồng” đối với thiết kế hướng dịch vụ, việc áp dụng đầy đủ nguyên tắc này thường sẽ yêu cầu sử dụng các công cụ phát triển có khả năng nhập hợp đồng dịch vụ tùy chỉnh mà không áp đặt các thay đổi.

• Cần có bộ kỹ năng phù hợp để thực hiện quá trình mô hình hóa và thiết kế bằng các công cụ đã chọn. Khi làm việc với các dịch vụ Web, nhu cầu về mức độ thành thạo cao với lược đồ XML và các ngôn ngữ WSDL trên thực tế là không thể tránh khỏi.

Chuyên môn về Chính sách WS cũng có thể được yêu cầu.

Những yêu cầu này và các yêu cầu khác có thể góp phần tạo nên nỗ lực chuyển đổi đáng chú ý vượt xa việc áp dụng công nghệ.

Bảng A.1

Hồ sơ về nguyên tắc Hợp đồng dịch vụ được tiêu chuẩn hóa

Dịch vụ khớp nối lỏng lẻo	
Định nghĩa ngắn	"Các dịch vụ được kết nối lỏng lẻo."
Độ phân giải dài	"Hợp đồng dịch vụ áp đặt yêu cầu kết nối người tiêu dùng thấp và bản thân chúng bị tách rời khỏi môi trường xung quanh."
Bàn thắng	Bằng cách liên tục thúc đẩy việc giảm sự kết nối trong và giữa các dịch vụ, chúng tôi đang nỗ lực hướng tới một trạng thái nơi các hợp đồng dịch vụ tăng tính độc lập trong việc triển khai và các dịch vụ của chúng ngày càng độc lập với nhau. Điều này thúc đẩy một môi trường trong đó các dịch vụ và người tiêu dùng có thể được phát triển thích ứng theo thời gian với tác động tối thiểu lên nhau.
Thiết kế Đặc trưng	<ul style="list-style-type: none"> Sự tồn tại của một hợp đồng dịch vụ được tách rời một cách lý tưởng từ công nghệ và chi tiết thực hiện. Bối cảnh dịch vụ chức năng không phụ thuộc vào bên ngoài Hợp lý. Yêu cầu ghép nối người tiêu dùng tối thiểu.
Thực hiện Yêu cầu	<ul style="list-style-type: none"> Các dịch vụ liên kết lỏng lẻo thường được yêu cầu thực hiện xử lý thời gian chạy nhiều hơn nếu chúng chật chẽ hơn được ghép nối. Kết quả là, việc trao đổi dữ liệu nối chung có thể tiêu tốn nhiều tài nguyên thời gian chạy hơn, đặc biệt là trong quá trình truy cập đồng thời và các tình huống sử dụng cao. Đạt được sự cân bằng phù hợp của khớp nối, đồng thời hỗ trợ áp dụng các nguyên tắc định hướng dịch vụ khác ảnh hưởng đến hợp đồng thiết kế, đòi hỏi phải tăng cường khả năng thiết kế hợp đồng dịch vụ.

Bảng A.2

Hồ sơ về nguyên tắc Khớp nối lỏng lẻo dịch vụ

Trữ tượng hóa dịch vụ	
Định nghĩa ngắn	"Thông tin dịch vụ không thiết yếu được tóm tắt."
Độ phân giải dài	"Hợp đồng dịch vụ chỉ chứa đựng những thông tin, thông tin cần thiết về dịch vụ được giới hạn ở những gì được công bố trong hợp đồng dịch vụ."
Bàn thắng	Nhiều nguyên tắc khác nhau mạnh mẽ cần thiết phải xuất bản nhiều hơn thông tin trong hợp đồng dịch vụ. Vai trò chính của nguyên tắc này là giữ số lượng và chi tiết của nội dung hợp đồng ngắn gọn và cân bằng, đồng thời ngăn chặn việc truy cập không cần thiết vào các chi tiết dịch vụ bổ sung.
Thiết kế Đặc trưng	<ul style="list-style-type: none"> • Các dịch vụ luôn trữ tượng hóa thông tin cụ thể về công nghệ, logic và chức năng khỏi thế giới bên ngoài (thế giới bên ngoài ranh giới dịch vụ). • Dịch vụ có hợp đồng xác định ngắn gọn sự tương tác yêu cầu và ràng buộc cũng như meta dịch vụ bắt buộc khác chi tiết. • Ngoài những gì được ghi trong hợp đồng dịch vụ, thông tin về dịch vụ sẽ được kiểm soát hoặc ẩn hoàn toàn trong một môi trường cụ thể.
Thực hiện Yêu cầu	Điều kiện tiên quyết chính để đạt được mức độ trữ tượng thích hợp cho từng dịch vụ là mức độ kỹ năng thiết kế hợp đồng dịch vụ được áp dụng.

Bảng A.3

Hồ sơ về nguyên tắc Trữ tượng hóa dịch vụ

Khả năng sử dụng lại dịch vụ	
Định nghĩa ngắn	"Dịch vụ có thể tái sử dụng được."
Độ phân giải dài	"Các dịch vụ chứa đựng và thể hiện logic bất khả tri và có thể được định vị như các tài nguyên doanh nghiệp có thể tái sử dụng được."
Bàn thắng	<p>Các mục tiêu đăng sau Khả năng sử dụng lại dịch vụ được gắn trực tiếp với một số mục tiêu chiến lược nhất của điện toán hướng dịch vụ:</p> <ul style="list-style-type: none"> • Để cho phép logic dịch vụ được tận dụng nhiều lần thời gian để đạt được lợi nhuận ngày càng cao trên đầu tư ban đầu cho việc cung cấp dịch vụ. • Để tăng tính linh hoạt trong kinh doanh ở cấp độ tổ chức bằng cách cho phép đáp ứng nhanh chóng các yêu cầu tự động hóa kinh doanh trong tương lai thông qua việc kết hợp dịch vụ trên quy mô rộng. • Để cho phép thực hiện các mô hình dịch vụ bất khả tri. • Để có thể tạo ra hàng tồn kho dịch vụ với tỷ lệ cao tỷ lệ phần trăm của các dịch vụ bất khả tri.
Đặc điểm thiết kế	<ul style="list-style-type: none"> • Logic được gói gọn trong dịch vụ được liên kết với một bối cảnh đủ bát khả tri đối với bất kỳ kịch bản sử dụng nào để có thể được coi là có thể tái sử dụng. • Logic được dịch vụ gói gọn đủ chung chung, cho phép nó tạo điều kiện thuận lợi cho nhiều tình huống sử dụng của những người tiêu dùng dịch vụ khác nhau. • Hợp đồng dịch vụ đủ linh hoạt để xử lý nhiều loại công việc thông báo đầu vào và đầu ra. • Các dịch vụ được thiết kế để tạo điều kiện truy cập đồng thời bởi nhiều chương trình tiêu dùng.

Thực hiện Yêu cầu	<p>Từ góc độ triển khai, Khả năng sử dụng lại dịch vụ có thể là nguyên tắc đòi hỏi khắt khe nhất mà chúng tôi đã đề cập xa. Dưới đây là các yêu cầu chung để tạo các dịch vụ có thể tái sử dụng và hỗ trợ sự tồn tại lâu dài của chúng:</p> <ul style="list-style-type: none"> • Mô hình lưu trữ thời gian chạy có thể mở rộng có khả năng sử dụng dịch vụ đồng thời ở mức cao đến cực cao. Khi một kho dịch vụ tương đối hoàn thiện, các dịch vụ có thể tái sử dụng sẽ tự tìm thấy với số lượng sáng tác ngày càng nhiều. • Một hệ thống kiểm soát phiên bản vững chắc để phát triển hợp đồng một cách hợp lý đại diện cho các dịch vụ có thể tái sử dụng. • Các nhà phân tích và thiết kế dịch vụ có trình độ chuyên môn cao chuyên môn về chủ đề có thể đảm bảo rằng dịch vụ xanh giới và hợp đồng thể hiện chính xác dịch vụ bối cảnh chức năng có thể tái sử dụng. • Mức độ phát triển dịch vụ và thương mại cao chuyên môn phát triển phần mềm để cấu trúc các logic cơ bản thành chung và có khả năng phân tách được các thành phần và thủ tục. <p>Những yêu cầu này và các yêu cầu khác nhấn mạnh vào việc bồi trí nhân sự phù hợp cho nhóm cung cấp dịch vụ, cũng như tầm quan trọng của mô hình lưu trữ và cơ sở hạ tầng hỗ trợ mạnh mẽ và có thể mở rộng.</p>
------------------------------	--

Bảng A.4

Hồ sơ về nguyên tắc Tái sử dụng dịch vụ

Tự chủ dịch vụ	
Định nghĩa ngắn	"Dịch vụ là tự chủ."
Độ phân giải dài	"Các dịch vụ thực hiện mức độ kiểm soát cao đối với thời gian chạy cơ bản của chúng môi trường thực thi."
Bàn thắng	<ul style="list-style-type: none"> Để tăng độ tin cậy, hiệu suất thời gian chạy của dịch vụ và khả năng dự đoán, đặc biệt là khi được tái sử dụng và soạn thảo. Để tăng mức độ kiểm soát của một dịch vụ trong thời gian chạy của nó môi trường. <p>Bằng cách theo đuổi môi trường thời gian chạy và thiết kế tự động, chúng tôi về cơ bản là nhằm mục đích tăng cường khả năng kiểm soát sau triển khai đối với dịch vụ và khả năng kiểm soát của dịch vụ đối với việc thực thi của chính nó môi trường.</p>
Thiết kế Đặc trưng	<ul style="list-style-type: none"> Dịch vụ có hợp đồng thể hiện ranh giới chức năng được xác định rõ ràng và không trùng lặp với các dịch vụ khác. Các dịch vụ được triển khai trong một môi trường mà chúng thực hiện ít nhất nhiều quyền kiểm soát (và tốt nhất là ở mức độ độc quyền). Các phiên bản dịch vụ được lưu trữ trên một môi trường phù hợp ngày có tính đồng thời cao cho mục đích mở rộng.
Thực hiện Yêu cầu	<ul style="list-style-type: none"> Mức độ kiểm soát cao đối với cách logic dịch vụ được thiết kế và đã phát triển. Tùy thuộc vào mức độ tự chủ được yêu cầu, điều này cũng có thể liên quan đến việc kiểm soát các mô hình dữ liệu hỗ trợ. Một môi trường triển khai có thể phân phối được, để cho phép dịch vụ được di chuyển, cách ly hoặc soạn thảo theo yêu cầu. Cơ sở hạ tầng có khả năng hỗ trợ mong muốn các cấp độ tự chủ.

Bảng A.5

Hồ sơ về nguyên tắc Tự chủ Dịch vụ

Dịch vụ không quốc tịch	
Định nghĩa ngắn	"Dịch vụ giảm thiểu trạng thái."
Độ phân giải dài	"Dịch vụ giảm thiểu việc tiêu thụ tài nguyên bằng cách trì hoãn việc quản lý thông tin của tiểu bang khi cần thiết."
Bàn thắng	<ul style="list-style-type: none"> • Để tăng khả năng mở rộng dịch vụ. • Để hỗ trợ thiết kế logic dịch vụ bắt khả tri và cải thiện tiềm năng tái sử dụng dịch vụ.
Thiết kế Đặc trưng	<p>Điều làm cho nguyên tắc này trở thành một nguyên tắc độc đáo là thực tế là nó đang thúc đẩy một điều kiện của dịch vụ có tính chất tạm thời. Tùy thuộc vào mô hình dịch vụ và phương pháp trì hoãn trạng thái được sử dụng, có thể có các loại đặc điểm thiết kế khác nhau. được thực hiện. Một số ví dụ bao gồm:</p> <ul style="list-style-type: none"> • Tính logic bắt khả tri của quy trình kinh doanh cao để dịch vụ được không được thiết kế để lưu giữ thông tin trạng thái cho bất kỳ phụ huynh cụ thể nào quá trình kinh doanh. • Hợp đồng dịch vụ ít ràng buộc hơn để cho phép nhận và truyền tải nhiều loại dữ liệu trạng thái hơn trong thời gian chạy. • Tăng số lượng quy trình lập trình diễn giải có khả năng phân tích một loạt thông tin trạng thái được cung cấp bởi tin nhắn và phản hồi một loạt hành động tương ứng yêu cầu.
Thực hiện Yêu cầu	<p>Mặc dù việc trì hoãn trạng thái có thể làm giảm mức tiêu thụ tổng thể bộ nhớ và tài nguyên hệ thống, nhưng các dịch vụ được thiết kế có cân nhắc đến việc không có trạng thái cũng có thể đưa ra một số yêu cầu về hiệu năng liên quan đến việc truy xuất và diễn giải dữ liệu trạng thái trì hoãn trong thời gian chạy.</p> <p>Dưới đây là danh sách kiểm tra ngắn về các yêu cầu chung có thể được sử dụng để đánh giá sự hỗ trợ của các thiết kế dịch vụ không trạng thái theo công nghệ của nhà cung cấp và vị trí triển khai mục tiêu:</p> <ul style="list-style-type: none"> • Môi trường thời gian chạy phải cho phép dịch vụ chuyển từ trạng thái không hoạt động sang trạng thái xử lý hoạt động ở mức cao cách làm hiệu quả.

- Trình phân tích cú pháp XML cấp doanh nghiệp hoặc hiệu suất cao và bộ tăng tốc kho (và bộ xử lý SOAP) nên được cung cấp để cho phép các dịch vụ được triển khai dưới dạng dịch vụ Web đạt hiệu quả cao hơn - phân tích một cách khoa học tải trọng tin nhắn lớn hơn với hiệu suất thấp hơn hạn chế.

- Việc sử dụng các tệp định kèm có thể cần được hỗ trợ bởi Web dịch vụ cho phép tin nhắn bao gồm nội dung của dữ liệu tải trọng không trải qua quá trình xác nhận hoặc dịch ở cấp độ giao diện sang các định dạng cụb bộ.

Bản chất của việc hỗ trợ triển khai theo yêu cầu của dịch vụ không trạng thái có độ tuổi trung bình trong một môi trường sẽ phụ thuộc vào phương pháp trì hoãn trạng thái được sử dụng trong kiến trúc hướng dịch vụ.

Bảng A.6

Hồ sơ về nguyên tắc Dịch vụ không quốc tịch

Khả năng khám phá dịch vụ	
Định nghĩa ngắn	"Dịch vụ có thể được khám phá."
Độ phân giải dài	"Các dịch vụ được bổ sung siêu dữ liệu giao tiếp nhờ đó chúng có thể được phát hiện và giải thích một cách hiệu quả."
Bàn thắng	<ul style="list-style-type: none"> • Dịch vụ được định vị là tài nguyên có khả năng khám phá cao trong doanh nghiệp. • Mục đích và khả năng của từng dịch vụ được thể hiện rõ ràng được thể hiện để con người và các chương trình phần mềm có thể giải thích chúng. <p>Để đạt được những mục tiêu này đòi hỏi tầm nhìn xa và sự hiểu biết vững chắc về bản chất của dịch vụ. Tùy thuộc vào loại mô hình dịch vụ được thiết kế, việc thực hiện nguyên tắc này có thể đòi hỏi cả chuyên môn kinh doanh và kỹ thuật.</p>
Thiết kế Đặc trưng	<ul style="list-style-type: none"> • Hợp đồng dịch vụ được trang bị siêu dữ liệu phù hợp sẽ được tham chiếu chính xác khi truy vấn khám phá được đưa ra. • Hợp đồng dịch vụ được trang bị thêm với meta bổ sung thông tin truyền đạt rõ ràng mục đích của họ và khả năng cho con người. • Nếu số đăng ký dịch vụ tồn tại, các bản ghi đăng ký sẽ được đi kèm với sự chú ý tương tự đến thông tin meta như vừa mô tả. • Nếu số đăng ký dịch vụ không tồn tại, tài liệu hồ sơ dịch vụ được soạn thảo để bổ sung hợp đồng dịch vụ và hình thành cơ sở cho các hồ sơ đăng ký tương lai.

Thực hiện Yêu cầu

• Sự tồn tại của các tiêu chuẩn thiết kế chi phối meta thông tin được sử dụng để làm cho các hợp đồng dịch vụ có thể được khám phá và có thể giải thích được, cũng như các hướng dẫn về cách thức và thời điểm cung cấp dịch vụ hợp đồng cần được bổ sung thêm các chú thích.

• Sự tồn tại của các tiêu chuẩn thiết kế nhằm thiết lập tính nhất quán phương tiện ghi lại thông tin meta dịch vụ bên ngoài hợp đồng. Thông tin này được thu thập dưới dạng bổ sung tài liệu chuẩn bị đăng ký dịch vụ hoặc được đặt trong chính cơ quan đăng ký.

Bạn có thể nhận thấy sự vắng mặt của số đăng ký dịch vụ trong danh sách các yêu cầu triển khai. Như đã thành lập trước đó, Mục tiêu của nguyên tắc này là triển khai các đặc điểm thiết kế trong dịch vụ chứ không phải trong kiến trúc.

Bảng A.7

Hồ sơ về nguyên tắc Khả năng khám phá dịch vụ

Khả năng kết hợp dịch vụ	
Định nghĩa ngắn	"Dịch vụ có thể kết hợp được."
Độ phân giải dài	"Dịch vụ là những thành phần tham gia có hiệu quả, bắt kè quy mô và độ phức tạp của thành phần."
Bàn thắng	<p>Khi thảo luận về các mục tiêu của Khả năng kết hợp dịch vụ, hầu hết áp dụng các mục tiêu về Khả năng sử dụng lại dịch vụ. Điều này là do sự kết hợp dịch vụ thường trở thành một dạng tái sử dụng dịch vụ. Trên thực tế, bạn có thể nhớ lại rằng một trong những mục tiêu mà chúng tôi đã liệt kê cho Nguyên tắc tái sử dụng dịch vụ là cho phép kết hợp dịch vụ trên quy mô rộng.</p> <p>Tuy nhiên, trên hết chỉ đơn giản là đạt được việc tái sử dụng, dịch vụ Thành phần cung cấp phương tiện để qua đó chúng ta có thể đạt được điều thường được phân loại là mục tiêu cuối cùng của điện toán hướng dịch vụ. Bằng cách thành lập một doanh nghiệp bao gồm logic giải pháp được thể hiện bằng danh sách các dịch vụ có khả năng tái sử dụng cao, chúng tôi cung cấp các phương tiện để đáp ứng phần lớn các yêu cầu tự động hóa kinh doanh trong tương lai thông qua việc kết hợp dịch vụ.</p>
Thiết kế Đặc trưng	Lý tưởng nhất là mọi khả năng dịch vụ (đặc biệt là những khả năng cung cấp logic có thể tái sử dụng) đều được coi là thành viên tổng hợp tiềm năng. Về cơ bản, điều này có nghĩa là các đặc điểm thiết kế đã được thiết lập theo nguyên tắc Tái sử dụng dịch vụ đều có liên quan như nhau đối với việc xây dựng các thành viên hợp thành hiệu quả.
Đối với thành phần Khả năng của thành viên	<p>Ngoài ra, còn có hai đặc điểm nữa được nhấn mạnh bởi nguyên tắc này:</p> <ul style="list-style-type: none"> Dịch vụ cần có khả năng thực thi hiệu quả cao môi trường. Hơn cả khả năng quản lý đồng thời, hiệu quả mà các thành viên trong nhóm thực hiện xử lý riêng lẻ nên được điều chỉnh cao. Hợp đồng dịch vụ cần phải linh hoạt để có thể tạo điều kiện thuận lợi cho đưa ra các loại yêu cầu trao đổi dữ liệu khác nhau cho các yêu cầu tương tự chức năng. Điều này thường liên quan đến khả năng hợp đồng trao đổi cùng loại dữ liệu ở các cấp độ khác nhau. <p>Cách thức mà những phẩm chất này vượt xa khả năng sử dụng lại đơn thuần chủ yếu liên quan đến khả năng tối ưu hóa trách nhiệm xử lý thời gian chạy của dịch vụ để hỗ trợ nhiều thành phần đồng thời.</p>

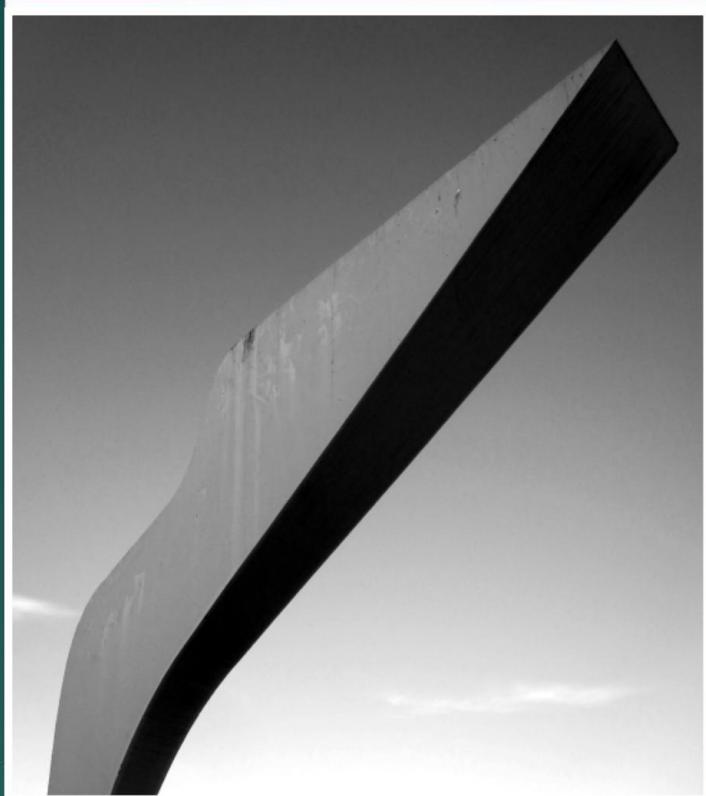
Thiết kế Đặc điểm cho Thành phần Bộ điều khiển Khả năng	<p>Các thành viên sáng tác thường cũng sẽ cần đóng vai trò là người điều khiển hoặc người điều khiển phụ trong các cấu hình sáng tác khác nhau.</p> <p>Tuy nhiên, các dịch vụ được thiết kế dưới dạng bộ điều khiển được chỉ định nhìn chung được giảm bớt do nhiều yêu cầu về hiệu suất cao đặt ra cho các thành viên tổng hợp.</p> <p>Do đó, các loại dịch vụ này có tập hợp các đặc điểm thiết kế riêng:</p> <ul style="list-style-type: none"> • Logic được gói gọn trong bộ điều khiển được chỉ định sẽ gần như luôn bị giới hạn trong một nhiệm vụ kinh doanh duy nhất. Thông thường, nhiệm vụ mô hình dịch vụ được sử dụng, dẫn đến những đặc điểm chung mô hình đó đang được áp dụng cho loại dịch vụ này. • Mặc dù các bộ điều khiển được chỉ định có thể tái sử dụng được nhưng việc tái sử dụng dịch vụ thường không phải là sự cân nhắc thiết kế chính. Vì vậy, đặc điểm thiết kế được thúc đẩy bởi Khả năng sử dụng lại dịch vụ là được xem xét và áp dụng khi thích hợp, nhưng với ít sự nghiêm ngặt thông thường được áp dụng cho các dịch vụ bắt khả tri. • Tính không quốc tịch không phải lúc nào cũng được nhấn mạnh nghiêm ngặt đối với những người kiểm soát được chỉ định cũng như với các thành viên hợp thành. Tùy về các lựa chọn trì hoãn của tiểu bang có sẵn ở khu vực xung quanh kiến trúc, bộ điều khiển được chỉ định đôi khi có thể cần phải được thiết kế để duy trì trạng thái đầy đủ trong khi các thành viên cấu thành cơ bản thực hiện các phản ứng tương ứng của chúng nhiệm vụ tổng thể. <p>Tất nhiên, bất kỳ khả năng nào đóng vai trò là người điều khiển đều có thể trở thành thành viên của một thành phần lớn hơn, mang lại thành phần trước đó cũng đã tính đến các đặc điểm thiết kế của thành phần thành phần.</p>
--	--

Bảng A.8

Hồ sơ về nguyên tắc Khả năng kết hợp dịch vụ

Trang này có ý đẻ trống

Phụ lục B



Tham khảo ràng buộc REST

Phụ lục này cung cấp các bảng hồ sơ cho các ràng buộc REST được tham chiếu qua được bổ sung thêm số trang của bảng hồ sơ tương ứng trong phụ lục này.

Mỗi bảng hồ sơ chứa các phần sau:

- Định nghĩa ngắn gọn - Một định nghĩa ngắn gọn, đơn giản nhằm thiết lập nền tảng mục đích tinh thần của sự ràng buộc.
- Định nghĩa dài - Mô tả dài hơn về ràng buộc để cung cấp nhiều chi tiết hơn về những gì nó dự định đạt được.
- Ứng dụng - Danh sách các bước và yêu cầu chung để áp dụng ràng buộc.
- Tác động - Danh sách các tác động tích cực và tiêu cực có thể xảy ra do việc áp dụng sự hạn chế.
- Mối quan hệ với REST - Giải thích ngắn gọn về cách ràng buộc có thể liên quan đến các ràng buộc khác các ràng buộc và kiến trúc REST tổng thể.
- Mục tiêu REST liên quan - Danh sách các mục tiêu thiết kế REST có liên quan và phù hợp với việc áp dụng hạn chế này.
- Các nguyên tắc định hướng dịch vụ liên quan - Danh sách các nguyên tắc định hướng dịch vụ liên quan đến hạn chế.
- Các mẫu SOA liên quan - Một danh sách các mẫu thiết kế SOA liên quan đến ràng buộc.

Lưu ý rằng các bảng hồ sơ này chỉ cung cấp phiên bản tóm tắt của các ràng buộc.

Nội dung đầy đủ về các ràng buộc REST, bao gồm cả các nghiên cứu điển hình, được cung cấp trong Sách SOA với REST: Nguyên tắc, mô hình & ràng buộc để xây dựng giải pháp doanh nghiệp với REST .

Để biết thêm thông tin về chủ đề này và các tựa sách khác trong Chuỗi Công nghệ Dịch vụ Prentice Hall từ các chủ đề liên quan , hãy truy cập www.servicetechbooks.com . Nội dung tóm tắt của REST-đến Thomas Erl, bạn cũng có thể tìm thấy trực tuyến tại www.whatisrest.com .

Máy khách-Máy chủ	
Định nghĩa ngắn	"Logic giải pháp được tách thành logic người tiêu dùng và dịch vụ có chung một hợp đồng kỹ thuật."
Độ phân giải dài	"Logic tự động hóa kinh doanh được tổ chức thành một giải pháp bao gồm đơn vị logic của người tiêu dùng và dịch vụ. Người tiêu dùng dịch vụ chủ động yêu cầu các khả năng của dịch vụ bằng cách gửi các tin nhắn tuân thủ hợp đồng dịch vụ kỹ thuật đã công bố. Các dịch vụ chờ đợi một cách thụ động để xử lý các tin nhắn yêu cầu và phản hồi việc nhận được chúng theo đúng hợp đồng kỹ thuật."
Ứng dụng	<ul style="list-style-type: none"> Giải pháp logic phải trải qua một quá trình tuân theo để tách biệt các mối quan tâm. Điều này phân vùng logic thành các đơn vị giải quyết các mối quan tâm đã xác định. Các đơn vị logic này được sáng tác để tạo thành giải pháp trong thời gian chạy. Kiến thức cần thiết của người tiêu dùng về một dịch vụ và kiến thức cần thiết về dịch vụ của người tiêu dùng bị giới hạn ở nội dung của hợp đồng kỹ thuật chung.
Tác động	<ul style="list-style-type: none"> Logic dịch vụ có thể mở rộng hơn và có thể tái sử dụng nhiều hơn vì nó được giải phóng khỏi việc phải triển khai logic dành riêng cho người tiêu dùng. Logic dịch vụ và người tiêu dùng được đơn giản hóa do tương ứng ăn thông tin. Việc triển khai dịch vụ và người tiêu dùng có thể được phát triển độc lập theo những cách không yêu cầu thay đổi đối với hợp đồng chia sẻ. Tương tác giữa các dịch vụ và người tiêu dùng phá vỡ hợp đồng kỹ thuật chung bị cấm, có khả năng dẫn đến mất cơ hội tối ưu hóa giải pháp ngành kiến trúc.
Mối quan hệ với NGHĨ NGÓI	Đây là một ràng buộc cơ bản xác định sự tách biệt giữa dịch vụ, người tiêu dùng và hợp đồng kỹ thuật mà họ chia sẻ. Tất cả các ràng buộc khác đều tham chiếu đến các tạo phẩm này và do đó được xây dựng dựa trên ràng buộc này.
Mục tiêu REST liên quan Khả năng sửa đổi, Khả năng mở rộng	
Có liên quan Định hướng dịch vụ Nguyên tắc	Khớp nối lồng lèo dịch vụ (293), Trưởng tượng hóa dịch vụ (294)
SOA liên quan mẫu	Thành phần năng lực [328], Không chuẩn hóa hợp đồng [335], Hợp đồng tách rời [337], Phân rã chức năng [344]

Không quốc tịch	
Định nghĩa ngắn	"Các dịch vụ vẫn không có trạng thái giữa các lần trao đổi tin nhắn yêu cầu/phản hồi với người tiêu dùng dịch vụ."
Độ phân giải dài	"Việc giao tiếp giữa dịch vụ và người tiêu dùng được quy định để người tiêu dùng cung cấp tất cả dữ liệu cần thiết để dịch vụ hiểu được từng yêu cầu của người tiêu dùng. Giữa các yêu cầu, dịch vụ được không được phép giữ lại bất kỳ dữ liệu trạng thái nào cụ thể về tương tác của nó với phiên bản người tiêu dùng, cho phép nó tồn tại trong điều kiện không trạng thái. Thay vì, trạng thái phiên được hoãn lại cho người tiêu dùng vào cuối mỗi yêu cầu."
Ứng dụng	<ul style="list-style-type: none"> • Logic tiêu dùng phải được thiết kế để bảo toàn dữ liệu trạng thái giữa các yêu cầu và đưa ra các thông báo yêu cầu có chứa dữ liệu trạng thái. • Thông báo yêu cầu phải chứa tất cả dữ liệu trạng thái cần thiết-tiền để dịch vụ xử lý yêu cầu và dịch vụ phải có thẻ "quên" dữ liệu trạng thái khi đưa ra phản hồi mà không ảnh hưởng đến sự tương tác tổng thể. • Bởi vì dịch vụ chỉ liên quan đến việc tự động hóa một giải pháp khi người tiêu dùng chủ động đưa ra yêu cầu đối với nó, giữa các lần yêu cầu dịch vụ ở trạng thái "nghỉ ngơi" và do đó không sử dụng Tài nguyên CPU, bộ nhớ hoặc mạng thay thế cho người tiêu dùng. • Dịch vụ không thể được yêu cầu lưu trữ dữ liệu cụ thể vào phiên bản thời gian chạy của người sử dụng dịch vụ. Tuy nhiên, dịch vụ này vẫn được phép lưu trữ dữ liệu liên quan đến chức năng riêng của mình bối cảnh.

Tác động	<ul style="list-style-type: none"> Yêu cầu người tiêu dùng chịu trách nhiệm bảo quản dữ liệu trạng thái giúp dịch vụ không phải lưu trữ và sao chép dữ liệu dễ bay hơi chỉ liên quan đến người tiêu dùng cá nhân chương trình. Trì hoãn trạng thái phiên cho người tiêu dùng giữa các yêu cầu giải phóng tài nguyên bộ nhớ dịch vụ, cho phép dịch vụ mở rộng quy mô với số lượng yêu cầu đồng thời, thay vì tổng số số lượng người tiêu dùng đồng thời. Dịch vụ có thể hiểu được tin nhắn mà không cần đã kiểm tra các tin nhắn trước đó. Điều này có thể đơn giản hóa dịch vụ thiết kế logic và giảm bớt độ phức tạp của việc gỡ lỗi. Yêu cầu truyền liên tục dữ liệu trạng thái có khả năng dư thừa có thể làm tăng lưu lượng và xử lý mạng trên không. Độ tin cậy của dữ liệu trạng thái có thể vừa tích cực vừa tiêu cực bị ảnh hưởng: Các lỗi phiên bản dịch vụ có thể được xử lý một cách dễ dàng vì dịch vụ không giữ lại trạng thái, nhưng lỗi của người tiêu dùng dịch vụ có thể dẫn đến mất dữ liệu trạng thái.
Mối quan hệ với NGHÌ NGƠI	Mặc dù hạn chế này được xây dựng dựa trên Máy khách-Máy chủ {307}, nhưng nó giúp bật Bộ đệm ẩn {310} và Hệ thống phân lớp {313}.
Các mục tiêu REST liên quan	Khả năng sửa đổi, Khả năng mở rộng, Hiệu suất (tiêu cực), Khả năng hiển thị, Độ tin cậy
Dịch vụ liên quan- Định hướng Nguyên tắc	Dịch vụ không quốc tịch (298)
SOA liên quan mẫu	Tin nhắn trạng thái [362]

Bộ nhớ đệm	
Định nghĩa ngắn	"Người tiêu dùng dịch vụ có thể lưu vào bộ nhớ đệm và sử dụng lại dữ liệu tin nhắn phản hồi."
Độ phân giải dài	"Dữ liệu được cung cấp bởi tin nhắn phản hồi trước có thể được tạm thời được người tiêu dùng dịch vụ lưu trữ và sử dụng lại cho các tin nhắn yêu cầu sau này."
Ứng dụng	<ul style="list-style-type: none"> Các dịch vụ phải được thiết kế để tạo ra khả năng kiểm soát bộ nhớ đệm chính xác siêu dữ liệu và trả lại nó trong tin nhắn phản hồi. Thông báo phản hồi được đánh dấu là có thể lưu vào bộ nhớ đệm hoặc không thể lưu vào bộ nhớ đệm bằng siêu dữ liệu tin nhắn rõ ràng hoặc như một phần của định nghĩa hợp đồng. Kho lưu trữ bộ đệm trung gian hoặc phía người tiêu dùng tùy chọn cho phép người tiêu dùng sử dụng lại dữ liệu phản hồi có thể lưu trong bộ nhớ đệm cho lần sau yêu cầu tin nhắn. Các thông báo yêu cầu phải có khả năng so sánh được để xác định liệu hay không thi chúng tương đương nhau. Hợp đồng phải bao gồm các tuyên bố rõ ràng về khả năng lưu vào bộ nhớ đệm của các phản hồi hoặc phải cho phép đưa siêu dữ liệu kiểm soát bộ đệm vào phản hồi.
Tác động	<ul style="list-style-type: none"> Hiệu quả thời gian chạy được cải thiện bằng cách loại bỏ nhu cầu về các thông điệp phản hồi trùng lặp sẽ được truyền đi và xử lý. Bộ đệm cung cấp một cơ chế mạnh mẽ và đơn giản để thực hiện "sao chép lười biếng" dữ liệu trạng thái dịch vụ cho người tiêu dùng. Một số dạng dữ liệu được lưu trong bộ nhớ đệm có thể trở nên cũ và lỗi thời nếu không được kiểm tra và cập nhật thường xuyên.
Mối quan hệ với NGHÌ NGƠI	Một số kỹ thuật đã được thiết lập để đẩy dữ liệu đến người tiêu dùng không được phép bởi ứng dụng Client-Server (307) và Stateless (308). Ràng buộc bộ đệm cung cấp một cơ chế được các ràng buộc khác cho phép và một cơ chế dẫn đến một kiến trúc đơn giản và mạnh mẽ để sử dụng lại và tối ưu hóa việc phân phối dữ liệu.
Các mục tiêu REST liên quan Hiệu suất, Khả năng mở rộng, Độ tin cậy (tiêu cực)	
Có liên quan	không có
Định hướng dịch vụ	
Nguyên tắc	
SOA liên quan mẫu	Tin nhắn trạng thái [362]

Hợp đồng thống nhất	
Định nghĩa ngắn	"Người tiêu dùng dịch vụ và dịch vụ có chung một điểm chung, bao quát, chung hợp đồng kỹ thuật."
Độ phân giải dài	"Người tiêu dùng tiếp cận khả năng dịch vụ thông qua các phương pháp, loại phương tiện và một cú pháp nhận dạng tài nguyên chung được chuẩn hóa cho nhiều người tiêu dùng và dịch vụ. Khả năng dịch vụ cung cấp quyền truy cập vào các tài nguyên có thể cung cấp thêm liên kết đến các tài nguyên khác."
Ứng dụng	<ul style="list-style-type: none"> • Một hợp đồng thống nhất với các phương pháp, phương tiện truyền thông chung và có thể tái sử dụng các loại và cú pháp nhận dạng tài nguyên được thiết lập cho một bộ sưu tập người tiêu dùng và dịch vụ. • Logic xử lý tin nhắn của người tiêu dùng được thiết kế chặt chẽ kèm theo hợp đồng thống nhất. • Logic xử lý tin nhắn của người tiêu dùng được thiết kế để tách rời hoặc kết hợp lồng lênh với các khả năng và dịch vụ cụ thể tài nguyên. • Các tài nguyên có thể cung cấp thêm liên kết tới các tài nguyên khác người tiêu dùng dịch vụ có thể "khám phá" và tùy ý truy cập, động trong thời gian chạy.
Tác động	<ul style="list-style-type: none"> • Việc áp dụng hạn chế này dẫn đến việc tiêu chuẩn hóa cơ bản các đặc điểm giao diện kỹ thuật trên tất cả các dịch vụ trong phạm vi áp dụng. Mức độ tiêu chuẩn hóa này có thể thúc đẩy khả năng tương tác trên tất cả các dịch vụ bị ảnh hưởng. • Tiêu chuẩn hóa từ Hợp đồng thống nhất có thể bao gồm lược đồ chuẩn được liên kết với các loại phương tiện truyền thông. Chung việc sử dụng các lược đồ như vậy có thể cải thiện hơn nữa mức độ hiểu biết nội tại khả năng tương tác. • Bằng cách hạn chế việc kết hợp với hợp đồng thống nhất và tận dụng ràng buộc năng động, người tiêu dùng và dịch vụ có thể đạt được mức giảm mức độ yêu cầu ghép nối tổng thể. • Có thể khó xác định và hoàn toàn dựa vào các công cụ tích hợp sẵn ngữ nghĩa hợp đồng thống nhất cho các tương tác giữa máy với máy cần được nhiều dịch vụ tái sử dụng và người tiêu dùng. • Tin nhắn yêu cầu và phản hồi dựa trên các phương pháp thống nhất và các loại phương tiện truyền thông có thể chứa nhiều thông tin hơn mức quy định cần thiết cho một tương tác cụ thể. Việc chuyển giao dữ liệu dữ liệu có thể tăng chi phí hiệu suất.

Mối quan hệ với NGHĨ NGƠI	Ràng buộc Hợp đồng thống nhất được xây dựng dựa trên Máy khách-Máy chủ {307} để hỗ trợ việc tái sử dụng và kết hợp các ứng dụng tiêu dùng và dịch vụ.
Các mục tiêu REST liên quan Tính đơn giản, Khả năng sửa đổi, Hiệu suất (tiêu cực), Khả năng hiển thị	
Có liên quan Định hướng dịch vụ Nguyên tắc	Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lồng léo (293), Trừu tượng hóa dịch vụ (294)
SOA liên quan mẫu	Hợp đồng tách rời [337]

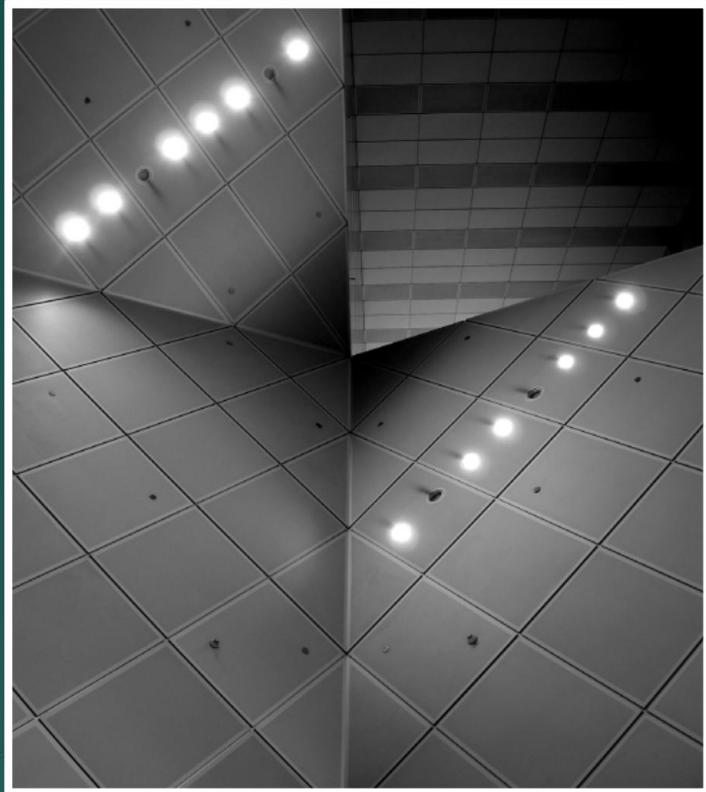
Hệ thống phân lớp	
Định nghĩa ngắn	"Một giải pháp có thể bao gồm nhiều lớp kiến trúc."
Độ phân giải dài	<p>"Một giải pháp được xác định dưới dạng các lớp kiến trúc, trong đó không có ai lớp có thể nhìn qua lớp tiếp theo. Các lớp có thể bao gồm người tiêu dùng và dịch vụ có hợp đồng được xuất bản hoặc các thành phần phần mềm trung gian hướng sự kiện (trung gian) thiết lập các lớp xử lý giữa người tiêu dùng và dịch vụ. Trong cả hai trường hợp, logic trong lớp giải pháp nhất định không thể có kiến thức vượt ra ngoài các lớp ngay lập tức bên trên hoặc bên dưới nó (trong hệ thống phân cấp giải pháp)."</p>
Ứng dụng	<ul style="list-style-type: none"> Người tiêu dùng được thiết kế để yêu cầu các dịch vụ mà không cần biết về những dịch vụ khác mà các dịch vụ đó cũng có thể gọi. Các bên trung gian được thêm vào để thực hiện quá trình xử lý thông báo trong thời gian chạy mà không biết những thông báo đó có thể tiếp tục như thế nào được xử lý ngoài lớp xử lý tiếp theo. Kiến trúc giải pháp được thiết kế để cho phép phần mềm trung gian mới các lớp được thêm vào hoặc các lớp phần mềm trung gian cũ sẽ bị xóa mà không thay đổi hợp đồng kỹ thuật giữa dịch vụ và người tiêu dùng. Thông báo yêu cầu và phản hồi không được tiết lộ lớp nào tin nhắn đến từ người nhận của họ.
Tác động	<ul style="list-style-type: none"> Ở cấp độ người tiêu dùng/dịch vụ, hạn chế này đảm bảo mức độ che giấu thông tin, điều này làm giảm một cách tự nhiên sự kết nối giữa người tiêu dùng và dịch vụ. Ở cấp độ thành phần phần mềm trung gian, ràng buộc này ủng hộ việc sử dụng các tác nhân xuyên suốt có khả năng thực hiện chung, chức năng tập trung vào tiện ích trên các tin nhắn được trao đổi bởi người tiêu dùng và dịch vụ. Những loại lớp kiến trúc này có thể cung cấp một phương tiện linh hoạt phát triển kiến trúc giải pháp và/hoặc cấu trúc cơ sở hạ tầng cơ bản của nó đồng thời giảm thiểu tác động lên logic giải pháp chính nó. Sự phân tách và phân bổ các bộ phận chuyển động ngày càng tăng thực hiện xử lý logic giải pháp có thể tác động tiêu cực đến chi phí hiệu suất tổng thể (đặc biệt khi phần mềm trung gian các thành phần đang được sử dụng lại bởi nhiều giải pháp). Bằng cách giới hạn kiến thức về toàn bộ kiến trúc giải pháp ở mức nhà thiết kế tiêu dùng, cơ hội tối ưu hóa thời gian chạy hiệu suất của một giải pháp có thể bị mất.

Mối quan hệ với NGHĨ NGƠI	Các thành phần phần mềm trung gian thường được giới thiệu khi áp dụng ràng buộc này có thể trực tiếp hỗ trợ hoặc kích hoạt Hợp đồng thống nhất {311}, Bộ nhớ đệm {310} và Không trạng thái {308}.
Các mục tiêu REST liên quan	Khả năng sửa đổi, Khả năng mở rộng, Hiệu suất (tiêu cực), Tính đơn giản, Hiển thị
Có liên quan Định hướng dịch vụ Nguyên tắc	Khớp nối lòng lẻo dịch vụ (293), Trừu tượng hóa dịch vụ (294)
SOA liên quan mẫu	Thành phần năng lực [328], Đại lý dịch vụ [357]

Mã theo yêu cầu	
Định nghĩa ngắn	"Người sử dụng dịch vụ hỗ trợ việc thực thi logic dịch vụ bị trì hoãn."
Độ phân giải dài	"Các kiến trúc tiêu dùng dịch vụ bao gồm một môi trường thực thi cho logic được cung cấp bởi một dịch vụ. Logic trì hoãn này có thể được sử dụng để mở rộng chức năng của người tiêu dùng hoặc để tạm thời chuyên môn hóa nó cho một mục đích cụ thể."
Ứng dụng	<ul style="list-style-type: none"> Người sử dụng dịch vụ được thiết kế để xử lý logic được tải xuống chúng theo các dịch vụ trong thời gian chạy. Các dịch vụ đưa ra quyết định rõ ràng về việc liệu họ có tự thực thi logic hoặc trì hoãn việc thực thi logic đó người tiêu dùng của họ.
Tác động	<ul style="list-style-type: none"> Các tính năng có thể được thêm vào người tiêu dùng mà không cần cần phải nâng cấp chúng một cách chính thức. Các dịch vụ có thể tránh bị tắc nghẽn trong quá trình thực thi bằng cách trì hoãn logic cho người tiêu dùng hơn là thực thi logic chúng tôi. Môi trường thực thi cần thiết để người tiêu dùng xử lý logic dịch vụ có thể gây ra các lỗ hổng bảo mật.
Mối quan hệ với NGHĨ NGỜI	không có
Các mục tiêu REST liên quan Khả năng sửa đổi, Khả năng mở rộng, Hiệu suất, Khả năng hiển thị (tiêu cực), Đơn giản (tiêu cực)	
Có liên quan Định hướng dịch vụ Nguyên tắc	không có
SOA liên quan mẫu	không có

Trang này có ý đẻ trống

Phụ lục C



Tham khảo các mẫu thiết kế SOA

Phụ lục này nêu các cách hổ trợ cho các mẫu được tham chiếu trong suốt phần này. Mọi tham chiếu mẫu được thêm vào phần này số bảng hổ sơ tương ứng của nó trong phụ lục này.

Mẫu thiết kế là gì?

Cách đơn giản nhất để mô tả một mẫu là nó cung cấp một giải pháp đã được chứng minh cho một vấn đề chung được ghi lại riêng lẻ ở định dạng nhất quán và thường là một phần của bộ sưu tập lớn hơn.

Khái niệm về hình mẫu đã là một phần cơ bản của cuộc sống hàng ngày. Không thừa nhận điều đó mỗi lần, chúng tôi sử dụng các giải pháp đã được chứng minh một cách tự nhiên để giải quyết các vấn đề chung mỗi lần. Ngày. Các mẫu trong thế giới CNTT xoay quanh việc thiết kế các hệ thống tự động được gọi là các mẫu thiết kế.

Các mẫu thiết kế rất hữu ích vì chúng:

- Trình bày các giải pháp đã được thử nghiệm tại hiện trường cho các vấn đề thiết kế phổ biến
- Sắp xếp thông tin thiết kế thành một định dạng được tiêu chuẩn hóa và dễ "tham khảo"
- Nói chung có thể lặp lại được bởi hầu hết các chuyên gia CNTT liên quan đến thiết kế
- Có thể được sử dụng để đảm bảo tính nhất quán trong cách thiết kế và xây dựng hệ thống
- Có thể trở thành cơ sở cho các tiêu chuẩn thiết kế
- Thường linh hoạt và tùy chọn (và ghi lại một cách công khai tác động của việc áp dụng chúng) và thậm chí đề xuất các phương pháp thay thế
- Có thể được sử dụng làm công cụ hỗ trợ giáo dục bằng cách ghi lại các khía cạnh cụ thể của thiết kế hệ thống (bất kể chúng có được áp dụng hay không)
- Đôi khi có thể được áp dụng trước và sau khi triển khai hệ thống
- Có thể được hỗ trợ thông qua việc áp dụng các mẫu thiết kế khác là một phần của cùng một bộ sưu tập
- Làm phong phú vốn từ vựng của một lĩnh vực CNTT nhất định vì mỗi mẫu có một tên có ý nghĩa

Hơn nữa, do các giải pháp được cung cấp bởi các mẫu thiết kế đã được chứng minh nên việc áp dụng nhất quán của chúng có xu hướng cải thiện chất lượng của các thiết kế hệ thống một cách tự nhiên.

Hãy cung cấp một ví dụ đơn giản (không liên quan đến SOA) về mẫu thiết kế nhằm giải quyết một vấn đề thiết kế giao diện người dùng:

Vấn đề: Làm cách nào để hạn chế người dùng nhập giá trị của trường biểu mẫu vào một tập hợp các giá trị được xác định trước?

Giải pháp: Sử dụng danh sách thả xuống chứa các giá trị được xác định trước làm trường đầu vào.

Điều mà ví dụ này cũng nhấn mạnh là thực tế là giải pháp được cung cấp bởi một mẫu nhất định có thể không nhất thiết phải là giải pháp phù hợp duy nhất cho vấn đề đó. Trong thực tế, có thể có nhiều mẫu cung cấp giải pháp thay thế cho cùng một vấn đề.

Mỗi giải pháp sẽ có những yêu cầu và hệ quả riêng và người thực hành sẽ quyết định mô hình nào là phù hợp nhất.

Trong ví dụ trước, một giải pháp khác cho vấn đề đã nêu là sử dụng một hộp danh sách thay vì danh sách thả xuống. Sự thay thế này sẽ tạo thành cơ sở cho việc mô tả mẫu thiết kế riêng biệt. Người thiết kế giao diện người dùng có thể nghiên cứu và so sánh cả hai mẫu để tìm hiểu về lợi ích và sự cân bằng của từng mẫu. Một danh sách thả xuống, dành cho chặng hạm, chiếm ít không gian hơn hộp danh sách nhưng yêu cầu người dùng luôn thực hiện một hành động riêng biệt để truy cập vào danh sách. Vì hộp danh sách có thể hiển thị nhiều dòng trường cùng lúc nên người dùng có thể dễ dàng định vị giá trị mong muốn hơn.

GHI CHÚ

Mặc dù các mẫu thiết kế cung cấp các giải pháp thiết kế đã được chứng minh, nhưng việc chỉ sử dụng chúng không thể đảm bảo rằng các vấn đề thiết kế luôn được giải quyết theo yêu cầu. Nhiều yếu tố ảnh hưởng đến thành công cuối cùng của việc sử dụng một mẫu thiết kế, bao gồm các ràng buộc do môi trường triển khai áp đặt, năng lực của người thực hành, các yêu cầu kinh doanh khác nhau, v.v. Tất cả những điều này thể hiện các khía cạnh ảnh hưởng đến mức độ mà một mô hình có thể được áp dụng thành công.

Ngôn ngữ mẫu thiết kế là gì?

Ngôn ngữ mẫu là một tập hợp các mẫu có liên quan, hoạt động như các khái niệm xây dựng, trong đó chúng có thể được thực hiện theo một hoặc nhiều chuỗi mẫu được xác định trước hoặc được gợi ý trong đó mỗi mẫu tiếp theo được xây dựng dựa trên mẫu trước. Khái niệm về ngôn ngữ mẫu bắt nguồn từ kiến trúc xây dựng cũng như thuật ngữ trình tự mẫu được sử dụng cùng với thứ tự mà các mẫu có thể được thực hiện.

Một số ngôn ngữ mẫu có kết thúc mở, cho phép các mẫu được kết hợp thành nhiều chuỗi mẫu khác nhau, trong khi các ngôn ngữ khác có cấu trúc chặt chẽ hơn theo đó các nhóm mẫu được trình bày theo thứ tự ứng dụng gợi ý. Thứ tự này thường dựa trên mức độ chi tiết của các mẫu, trong đó các mẫu có chi tiết thô hơn được áp dụng trước các mẫu có chi tiết mịn hơn, sau đó xây dựng trên hoặc mở rộng nền tảng được thiết lập bởi các mẫu có chi tiết thô. Trong các loại ngôn ngữ mẫu này, cách thức tổ chức các mẫu thành các chuỗi mẫu bị giới hạn ở cách chúng được áp dụng trong các nhóm.

Ngôn ngữ mẫu có cấu trúc rất hữu ích vì chúng:

- Có thể tổ chức các nhóm mẫu thiết kế đã được thử nghiệm tại hiện trường thành các mẫu được đề xuất, đã được thử nghiệm tại hiện trường trình tự ứng dụng
- Đảm bảo tính nhất quán trong cách đạt được các mục tiêu thiết kế cụ thể (bởi vì bằng cách thực hiện các tập hợp các mô hình phụ thuộc lẫn nhau theo một thứ tự đã được chứng minh, chất lượng của kết quả sẽ được cải thiện có thể được đảm bảo dễ dàng hơn)
- Là những công cụ học tập hiệu quả có thể cung cấp cái nhìn sâu sắc về cách thức và lý do một vấn đề cụ thể phương pháp hoặc kỹ thuật cần được áp dụng cũng như tác dụng của việc áp dụng nó.
- Cung cấp mức độ sâu hơn liên quan đến ứng dụng mẫu (vì chúng ghi lại các mẫu riêng lẻ cộng với các hiệu ứng tích lũy của ứng dụng của chúng)
- Linh hoạt ở chỗ trình tự áp dụng mẫu cuối cùng tùy thuộc vào người thực hành (và cũng vì việc áp dụng bất kỳ mẫu nào trong ngôn ngữ tổng thể) có thể là tùy chọn)

Cuốn sách Các mẫu thiết kế SOA cung cấp một ngôn ngữ mẫu chính, có kết thúc mở cho SOA. Mức độ liên quan của các mẫu khác nhau có thể khác nhau, nhưng nhìn chung chúng có chung một mục tiêu chung và có thể khám phá chuỗi mẫu vô tận.

Hồ sơ mẫu

Mỗi bảng hồ sơ chứa các phần sau:

- Yêu cầu - Yêu cầu là một tuyên bố ngắn gọn, có một câu trình bày yêu cầu cơ bản được giải quyết theo mẫu dưới dạng một câu hỏi.
Mọi mô tả mẫu đều bắt đầu bằng câu lệnh này.
- Biểu tượng - Mỗi mô tả mẫu được kèm theo một hình ảnh biểu tượng hoạt động như một nhận dạng trực quan. Các biểu tượng được hiển thị cùng với các câu lệnh yêu cầu trong mỗi hồ sơ mẫu.
- Vấn đề - Vấn đề gây ra vấn đề và ảnh hưởng của vấn đề đó. Nó là đây vấn đề mà mẫu được kỳ vọng sẽ cung cấp giải pháp.
- Giải pháp - Điều này thể hiện giải pháp thiết kế được mẫu đề xuất để giải quyết vấn đề và đáp ứng yêu cầu.
- Ứng dụng - Phần này được dành riêng để mô tả cách áp dụng mẫu. Nó có thể bao gồm các hướng dẫn, chi tiết triển khai và đôi khi thậm chí là một quy trình được đề xuất.
- Tác động - Phần này nêu bật những hậu quả, chi phí và yêu cầu chung liên quan đến việc áp dụng một mẫu và cũng có thể cung cấp các lựa chọn thay thế có thể được xem xét.
- Nguyên tắc - Tham khảo các nguyên tắc định hướng dịch vụ liên quan.
- Kiến trúc - Tham khảo các kiểu kiến trúc SOA liên quan.

Lưu ý rằng các bảng hồ sơ này chỉ cung cấp phiên bản tóm tắt của các mẫu.

Nội dung đầy đủ về các mẫu thiết kế SOA, bao gồm cả các nghiên cứu điển hình, được cung cấp trong sách Các mẫu thiết kế SOA .

Để biết thêm thông tin về cuốn sách này và các tựa sách khác trong Chuỗi Công nghệ Dịch vụ Prentice Hall của Thomas Erl, hãy truy cập www.servicetechbooks.com . Các phiên bản tóm tắt của tất cả các hồ sơ mẫu SOA có thể được tìm thấy trực tuyến tại www.soapaterns.org .

Khả năng bắt khả tri

Bởi Thomas Erl



Làm cách nào để logic dịch vụ đa mục đích có thể được sử dụng và tổng hợp một cách hiệu quả?

Vấn đề

Khả năng dịch vụ bắt nguồn từ những mối quan tâm cụ thể có thể không hữu ích cho nhiều người tiêu dùng dịch vụ, do đó làm giảm tiềm năng tái sử dụng của dịch vụ bắt khả tri.

Giải pháp

Logic dịch vụ bắt khả tri được phân chia thành một tập hợp các khả năng được xác định rõ ràng nhằm giải quyết các mối quan tâm chung không cụ thể cho bất kỳ vấn đề nào. Thông qua phân tích tiếp theo, bối cảnh bắt khả tri của khả năng được cải tiến hơn nữa.

Ứng dụng

Khả năng dịch vụ được xác định và cải tiến lặp đi lặp lại thông qua các quy trình phân tích và mô hình hóa đã được chứng minh.

Tác động

Việc xác định từng khả năng dịch vụ đòi hỏi nỗ lực thiết kế và phân tích trước nhiều hơn.

Nguyên tắc

Hợp đồng dịch vụ được tiêu chuẩn hóa (291), Khả năng tái sử dụng dịch vụ (295), Khả năng kết hợp dịch vụ (302)

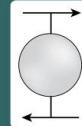
Ngành kiến trúc

Dịch vụ

Bối cảnh bất khả tri

Bởi Thomas Erl

Làm thế nào logic dịch vụ đa mục đích có thể được định vị như một nguồn lực doanh nghiệp hiệu quả?



Vấn đề

Logic đa mục đích được nhóm lại cùng với logic mục đích duy nhất dẫn đến các chương trình có ít hoặc không có tiềm năng tái sử dụng, gây lãng phí và dư thừa trong doanh nghiệp.

Giải pháp

Tách biệt logic không cụ thể cho một mục đích thành các dịch vụ riêng biệt với bối cảnh bất khả tri riêng biệt.

Ứng dụng

Bối cảnh dịch vụ bất khả tri được xác định bằng cách thực hiện các quy trình phân tích hướng dịch vụ và mô hình hóa dịch vụ.

Tác động

Mẫu này định vị logic giải pháp có thể tái sử dụng ở cấp độ doanh nghiệp, có khả năng làm tăng độ phức tạp của thiết kế và vấn đề quản trị doanh nghiệp.

Nguyên tắc

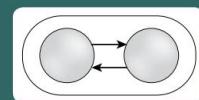
Khả năng tái sử dụng dịch vụ (295)

Ngành kiến trúc

Dịch vụ

Giao dịch dịch vụ nguyên tử

Bởi Thomas Erl



Làm cách nào để một giao dịch có khả năng khôi phục có thể được phổ biến trên các dịch vụ dựa trên tin nhắn?

Vấn đề	Khi các hoạt động thời gian chạy trải rộng trên nhiều dịch vụ không thành công, nhiệm vụ công việc chính chưa hoàn thành và các hành động được thực hiện cũng như các thay đổi được thực hiện cho đến thời điểm đó có thể ảnh hưởng đến tính toàn vẹn của giải pháp và kiến trúc cơ bản.
Giải pháp	Các hoạt động dịch vụ thời gian chạy có thể được gói gọn trong một giao dịch với tính năng khôi phục sẽ đặt lại tất cả các hành động và thay đổi nếu nhiệm vụ công việc chính không thể hoàn thành thành công.
Ứng dụng	Hệ thống quản lý giao dịch được tạo thành một phần của kiến trúc khoảng không quảng cáo và sau đó được sử dụng bởi các thành phần dịch vụ yêu cầu tính năng khôi phục.
Tác động	Các hoạt động dịch vụ được giao dịch có thể tiêu tốn nhiều bộ nhớ hơn do yêu cầu mỗi dịch vụ phải duy trì trạng thái ban đầu cho đến khi được thông báo khôi phục hoặc thực hiện các thay đổi.
Nguyên tắc	Dịch vụ không quốc tịch (298)
Ngành kiến trúc	Hàng tồn kho, Thành phần

Biểu thức chính tắc

Bởi Thomas Erl



Làm thế nào để hợp đồng dịch vụ có thể được hiểu và giải thích một cách nhất quán?

Vấn đề

Hợp đồng dịch vụ có thể thể hiện những khả năng tương tự theo những cách khác nhau, dẫn đến sự mâu thuẫn và có nguy cơ hiểu sai.

Giải pháp

Hợp đồng dịch vụ được chuẩn hóa bằng cách sử dụng quy ước đặt tên.

Ứng dụng

Quy ước đặt tên được áp dụng cho hợp đồng dịch vụ như một phần của quy trình phân tích và thiết kế chính thức.

Tác động

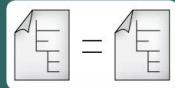
Việc sử dụng các quy ước đặt tên toàn cầu đưa ra các tiêu chuẩn toàn doanh nghiệp cần được sử dụng và thực thi một cách nhất quán.

Nguyên tắc

Hợp đồng dịch vụ được tiêu chuẩn hóa (291), Khả năng khám phá dịch vụ (300)

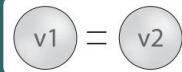
Ngành kiến trúc

Doanh nghiệp, Hàng tồn kho, Dịch vụ

Lược đồ chuẩn Bởi Thomas Erl		
Làm thế nào các dịch vụ có thể được thiết kế để tránh chuyển đổi mô hình dữ liệu?		
Vấn đề Các dịch vụ có mô hình khác nhau cho dữ liệu tương tự đặt ra các yêu cầu chuyển đổi làm tăng nỗ lực phát triển, độ phức tạp của thiết kế và chi phí hiệu suất thời gian chạy.		
Giải pháp Các mô hình dữ liệu cho các bộ thông tin chung được chuẩn hóa cho các hợp đồng dịch vụ trong phạm vi ranh giới hàng tồn kho.		
Ứng dụng Các tiêu chuẩn thiết kế được áp dụng cho các lược đồ được hợp đồng dịch vụ sử dụng như một phần của quy trình thiết kế chính thức.		
Tác động Việc duy trì tiêu chuẩn hóa các lược đồ hợp đồng có thể tạo ra nỗ lực quản trị đáng kể và những thách thức về văn hóa.		
Nguyên tắc Hợp đồng dịch vụ tiêu chuẩn (291)		
Ngành kiến trúc Hàng tồn kho, Dịch vụ		

Phiên bản chuẩn

Bởi Thomas Erl



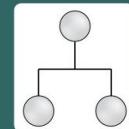
Làm cách nào để các hợp đồng dịch vụ trong cùng một kho dịch vụ có thể được phiên bản với tác động tối thiểu?

Vấn đề	Các hợp đồng dịch vụ trong cùng một kho dịch vụ được phiên bản khác nhau sẽ gây ra nhiều vấn đề về khả năng tương tác và quản trị.
Giải pháp	Các quy tắc lập phiên bản hợp đồng dịch vụ và cách thể hiện thông tin phiên bản được chuẩn hóa trong ranh giới kiểm kê dịch vụ.
Ứng dụng	Cần có các tiêu chuẩn quản trị và thiết kế để đảm bảo việc lập phiên bản nhất quán của các hợp đồng dịch vụ trong phạm vi ranh giới hàng tồn kho.
Tác động	Việc tạo và thực thi các tiêu chuẩn phiên bản bắt buộc đưa ra các nhu cầu quản trị mới.
Nguyên tắc	Hợp đồng dịch vụ tiêu chuẩn (291)
Ngành kiến trúc	Dịch vụ, Hàng tồn kho

Thành phần năng lực

Bởi Thomas Erl

Khả năng dịch vụ có thể giải quyết vấn đề đòi hỏi logic bên ngoài ranh giới dịch vụ như thế nào?

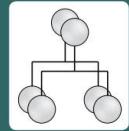


Vấn đề	Một khả năng có thể không thể đáp ứng được các yêu cầu xử lý của nó nếu không bổ sung logic nằm ngoài chức năng dịch vụ của nó. bối cảnh, do đó làm tổn hại đến tính toàn vẹn của bối cảnh dịch vụ và có nguy cơ mất chuẩn hóa dịch vụ.
Giải pháp	Khi yêu cầu quyền truy cập vào logic nằm ngoài phạm vi của dịch vụ ranh giới, logic khả năng trong dịch vụ được thiết kế để soạn một hoặc nhiều khả năng trong các dịch vụ khác.
Ứng dụng	Chức năng được gói gọn trong một khả năng bao gồm logic có thể gọi các khả năng khác từ các dịch vụ khác.
Tác động	Việc thực hiện logic thành phần yêu cầu lời gọi từ bên ngoài, điều này thêm chi phí hiệu suất và giảm quyền tự chủ dịch vụ.
Nguyên tắc	Tất cả
Ngành kiến trúc	Hàng tồn kho, Thành phần, Dịch vụ

Tái cấu trúc khả năng

Bởi Thomas Erl

Làm thế nào khả năng tương tự có thể được sử dụng để giúp giải quyết nhiều vấn đề?



Vấn đề Việc sử dụng logic dịch vụ bắt khả tri để chỉ giải quyết một vấn đề duy nhất là lãng phí và không tận dụng được tiềm năng tài sử dụng của logic.

Giải pháp Khả năng dịch vụ bắt khả tri có thể được thiết kế để được gọi nhiều lần nhằm hỗ trợ nhiều thành phần giải quyết nhiều vấn đề.

Ứng dụng Việc kết hợp lại hiệu quả đòi hỏi sự phối hợp, thành công và áp dụng lặp đi lặp lại một số mẫu bỗ sung.

Tác động Thành phần dịch vụ lặp đi lặp lại đòi hỏi sự quản lý và tiêu chuẩn hóa hiện có và liên tục.

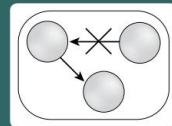
Nguyên tắc Tắt cá

Ngành kiến trúc Hàng tồn kho, Thành phần, Dịch vụ

Giao dịch dịch vụ bồi thường

Bởi Clemens Utschig-Utschig, Berthold Maier, Bernd Trops, Hajo Normann,

Diễn viên: Torsten Winterberg Brian Loesgen Mark Little



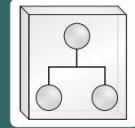
Làm cách nào để có thể cung cấp các ngoại lệ trong thời gian chạy tổng hợp một cách nhất quán mà không yêu cầu dịch vụ khóa tài nguyên?

Vấn đề	Trong khi các ngoại lệ thời gian chạy không được kiểm soát có thể gây nguy hiểm cho thành phần dịch vụ, gói thành phần đó trong một giao dịch nguyên tử có thể ràng buộc quá nhiều nguồn lực, do đó ảnh hưởng tiêu cực hiệu suất và khả năng mở rộng.
Giải pháp	Các thủ tục bù trừ được giới thiệu, cho phép thời gian chạy các trường hợp ngoại lệ sẽ được giải quyết với cơ hội giảm việc khóa tài nguyên và tiêu thụ bộ nhớ.
Ứng dụng	Logic bù được xác định trước và triển khai như một phần của logic bộ điều khiển thành phần gốc hoặc thông qua dịch vụ "hoàn tác" riêng lẻ khả năng.
Tác động	Không giống như các giao dịch nguyên tử bị chi phối bởi các quy tắc cụ thể, việc sử dụng logic bù có kết thúc mở và có thể khác nhau về hiệu quả thực tế của nó.
Nguyên tắc	Khớp nối lỏng lẻo (293)
Ngành kiến trúc	Hàng tồn kho, Thành phần

Tự chủ về thành phần

Bởi Thomas Erl

Làm cách nào để triển khai các sáng tác để giảm thiểu việc mất quyền tự chủ?



Vấn đề

Các dịch vụ của bộ điều khiển thành phần đương nhiên sẽ mất quyền tự chủ khi ủy quyền các tác vụ xử lý cho các dịch vụ tổng hợp, một số trong đó có thể được chia sẻ trên nhiều tác phẩm.

Giải pháp

Tất cả những người tham gia sáng tác có thể được cách ly để tối đa hóa quyền tự chủ của toàn bộ sáng tác.

Ứng dụng

Các dịch vụ thành viên bắt đầu của một tổ hợp được triển khai dự phòng trong một mô trường biệt lập cùng với dịch vụ tác vụ.

Tác động

Tăng quyền tự chủ ở cấp độ thành phần dẫn đến tăng chi phí cơ sở hạ tầng và trách nhiệm của chính phủ.

Nguyên tắc

Tự chủ dịch vụ (297), Khả năng tái sử dụng dịch vụ (295), Dịch vụ
Khả năng kết hợp (302)

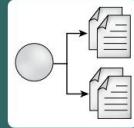
Ngành kiến trúc

Thành phần

Hợp đồng đồng thời

Bởi Thomas Erl

Làm thế nào một dịch vụ có thể tạo điều kiện thuận lợi cho các yêu cầu kết nối nhiều người tiêu dùng và các mối quan tâm trữ lượng cùng một lúc?



Vấn đề

Hợp đồng dịch vụ có thể không phù hợp hoặc áp dụng cho tất cả người tiêu dùng dịch vụ tiềm năng.

Giải pháp

Có thể tạo nhiều hợp đồng cho một dịch vụ, mỗi hợp đồng được nhắm mục tiêu ở một loại người tiêu dùng cụ thể.

Ứng dụng

Mẫu này được áp dụng lý tưởng cùng với Service Façade [360] để hỗ trợ các hợp đồng mới theo yêu cầu.

Tác động

Mỗi hợp đồng mới có thể thêm điểm cuối dịch vụ mới vào kho một cách hiệu quả, từ đó tăng cường nỗ lực quản trị tương ứng.

Nguyên tắc

Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lồng leó (293), Khả năng tái sử dụng dịch vụ (295)

Ngành kiến trúc

Dịch vụ

Container hóa

Bởi Roger Stoffers



Làm thế nào một môi trường có thể được cung cấp sự hỗ trợ tối đa cho các dịch vụ có yêu cầu về khả năng mở rộng và phục hồi hiệu suất cao?

Vấn đề

Các dịch vụ được triển khai trên máy chủ kim loại trần hoặc máy chủ áo có thể gây ra dấu ấn đáng kể. Áo hóa cải thiện tính di động nhưng giới thiệu một lớp xử lý trung gian có thể làm tăng thêm dấu chân. Việc triển khai giải pháp nguyên khối có thể dẫn đến giảm hiệu suất và tính khả dụng trên diện rộng khi bất kỳ thành phần dịch vụ hoặc giải pháp nào bị ngừng hoạt động hoặc ngoại lệ trong thời gian chạy.

Giải pháp

Các dịch vụ được triển khai độc lập hoặc cùng với các dịch vụ tổng hợp dịch vụ, như các đơn vị tự trị được đóng gói thành các dịch vụ độc lập hình ảnh container có thể quản lý và tự trị, mỗi hình ảnh bao gồm các phần phụ thuộc hệ thống cơ bản của dịch vụ. Dụng cụ là được cung cấp để quản lý việc xây dựng, triển khai và vận hành hệ thống hộp đựng.

Ứng dụng

Hệ thống quản lý container hoặc công cụ container được sử dụng để triển khai và vận hành container.

Tác động

Việc sử dụng công nghệ container hóa có thể đặt ra các yêu cầu bổ sung về cơ sở hạ tầng cũng như các yêu cầu liên quan. tăng chi phí quản lý của kiến trúc dịch vụ.

Nguyên tắc

Tự chủ dịch vụ (297), Khớp nối lỏng lẻo dịch vụ (293)

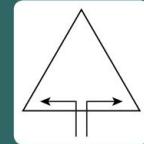
Ngành kiến trúc

Thành phần, Dịch vụ

Đàm phán nội dung

Bởi Raj Balasubramanian, David Booth, Thomas Erl

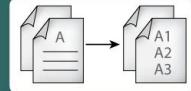
Làm thế nào một khả năng dịch vụ có thể đáp ứng người tiêu dùng dịch vụ với các yêu cầu về định dạng hoặc biểu diễn dữ liệu khác nhau?



Vấn đề	Những người sử dụng dịch vụ khác nhau có thể có những yêu cầu khác nhau về cách định dạng hoặc thể hiện dữ liệu được cung cấp bởi một khả năng dịch vụ nhất định.
Giải pháp	Cho phép khả năng dịch vụ hỗ trợ các định dạng và cách trình bày thay thế bằng cách cung cấp phương tiện để người tiêu dùng và dịch vụ có thể "thương lượng" các đặc điểm dữ liệu trong thời gian chạy.
Ứng dụng	Mẫu này được áp dụng phổ biến nhất thông qua các loại phương tiện HTTP có thể xác định định dạng và/hoặc cách trình bày dữ liệu tin nhắn. Loại phương tiện của dữ liệu được tách khỏi chính dữ liệu đó, cho phép dịch vụ hỗ trợ nhiều loại phương tiện truyền thông. Người tiêu dùng cung cấp siêu dữ liệu trong mỗi thông báo yêu cầu để xác định các loại phương tiện được ưu tiên và hỗ trợ. Dịch vụ có gắng đáp ứng các tùy chọn nhưng cũng có thể trả về dữ liệu theo cách khác các loại phương tiện được hỗ trợ khi đưa ra thông báo phản hồi.
Tác động	Cần ít khả năng dịch vụ hơn để đáp ứng sự thay đổi trong yêu cầu của người tiêu dùng dịch vụ. Các dịch vụ có thể hỗ trợ đồng thời các phiên bản dịch vụ cũ và mới dành cho người tiêu dùng sử dụng cùng các khả năng dịch vụ. Sự phức tạp của việc triển khai bộ đệm tăng lên và yêu cầu siêu dữ liệu bộ nhớ đệm đó cho biết dữ liệu đầu vào siêu dữ liệu nào cho mỗi yêu cầu có thể ảnh hưởng đến cách trình bày nào sẽ được trả về. Yêu cầu siêu dữ liệu không đủ trừu tượng có thể khiến người tiêu dùng gặp phải sự kết hợp triển khai dịch vụ.
Nguyên tắc	Hợp đồng dịch vụ được tiêu chuẩn hóa, (291) Khớp nối dịch vụ lồng léo (293)
Ngành kiến trúc	Thành phần, Dịch vụ

Hợp đồng không chuẩn hóa

Bởi Thomas Erl



Làm thế nào một hợp đồng dịch vụ có thể tạo điều kiện thuận lợi cho các chương trình tiêu dùng với các yêu cầu trao đổi dữ liệu khác nhau?

Vấn đề

Các dịch vụ có hợp đồng được chuẩn hóa nghiêm ngặt có thể áp đặt các yêu cầu về hiệu suất và chức năng không cần thiết đối với một số chương trình tiêu dùng.

Giải pháp

Hợp đồng dịch vụ có thể bao gồm mức độ không chuẩn hóa được đo lường, cho phép nhiều khả năng thể hiện dư thừa các chức năng cốt lõi theo nhiều cách khác nhau cho các loại chương trình tiêu dùng khác nhau.

Ứng dụng

Hợp đồng dịch vụ được gia hạn cần thận với các khả năng bổ sung cung cấp các biến thể chức năng của khả năng chính.

Tác động

Việc sử dụng quá mức mẫu này trên cùng một hợp đồng có thể làm tăng đáng kể kích thước của nó, gây khó khăn cho việc diễn giải và khó sử dụng. quản lý.

Nguyên tắc

Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293)

Ngành kiến trúc

Dịch vụ

Lớp tiện ích tên miền chéo

Bởi Thomas Erl

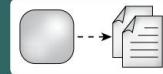
Làm cách nào để tránh logic tiện ích dư thừa trong kho dịch vụ miền?



Vấn đề	Mặc dù việc kiểm kê dịch vụ tên miền có thể cần thiết để quản trị doanh nghiệp độc lập nhưng chúng có thể tạo ra sự dư thừa không cần thiết.
Giải pháp	Một lớp dịch vụ tiện ích chung có thể được thiết lập, trải rộng trên hai hoặc nhiều hàng tồn kho dịch vụ tên miền hơn.
Ứng dụng	Một tập hợp chung các dịch vụ tiện ích cần được xác định và tiêu chuẩn hóa với sự phối hợp của chủ sở hữu kho dịch vụ.
Tác động	Cần phải tăng cường nỗ lực để điều phối và quản lý lớp dịch vụ tiện ích có trong kho chéo.
Nguyên tắc	Khả năng sử dụng lại dịch vụ (295), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Doanh nghiệp, Hàng tồn kho

Hợp đồng tách rời

Bởi Thomas Erl



Làm thế nào một dịch vụ có thể thể hiện khả năng của mình một cách độc lập với việc triển khai nó?

Vấn đề

Để dịch vụ được định vị là nguồn lực doanh nghiệp hiệu quả, nó phải được trang bị một kỹ thuật tồn tại độc lập với việc triển khai nhưng vẫn phù hợp với các dịch vụ khác.

Giải pháp

Hợp đồng dịch vụ được tách rời về mặt vật lý khỏi việc thực hiện nó.

Ứng dụng

Giao diện kỹ thuật của dịch vụ được tách biệt về mặt vật lý và tuân theo nguyên tắc thiết kế hướng dịch vụ có liên quan.

Tác động

Chức năng dịch vụ được giới hạn ở bộ tính năng của phương tiện hợp đồng được tách rời.

Nguyên tắc

Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lồng léo (293)

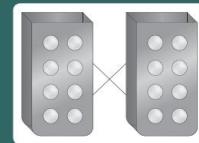
Ngành kiến trúc

Dịch vụ

Khoảng không quảng cáo tên miền

Bởi Thomas Erl

Làm thế nào các dịch vụ có thể được cung cấp để tối đa hóa việc tái cơ cấu khi không thể tiêu chuẩn hóa toàn doanh nghiệp?



Vấn đề	Việc thiết lập một kho dịch vụ doanh nghiệp duy nhất có thể khó quản lý đối với một số doanh nghiệp và những nỗ lực thực hiện điều đó có thể gây nguy hiểm cho sự thành công của việc áp dụng SOA nói chung.
Giải pháp	Các dịch vụ có thể được nhóm thành các dịch vụ có thể quản lý theo miền cụ thể hàng tồn kho, mỗi loại có thể được chuẩn hóa độc lập, được quản lý và sở hữu.
Ứng dụng	Ranh giới miền hàng tồn kho cần phải được thiết lập cẩn thận.
Tác động	Sự chênh lệch về tiêu chuẩn giữa các kho dịch vụ miền đặt ra các yêu cầu chuyển đổi và làm giảm tiềm năng lợi ích tổng thể của việc áp dụng SOA.
Nguyên tắc	Hợp đồng dịch vụ được tiêu chuẩn hóa (291), Trừu tượng hóa dịch vụ (294), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Doanh nghiệp, Hàng tồn kho

Giao thức kép



Bởi Thomas Erl

Làm cách nào kho dịch vụ có thể khắc phục các hạn chế của giao thức chuẩn trong khi vẫn được chuẩn hóa?

Vấn đề

Giao thức Canonical yêu cầu tất cả các dịch vụ phải tuân thủ việc sử dụng cùng một công nghệ truyền thông; tuy nhiên, một giao thức duy nhất có thể không thể đáp ứng được tất cả các yêu cầu dịch vụ, do đó đưa ra những hạn chế.

Giải pháp

Kiến trúc kiểm kê dịch vụ được thiết kế để hỗ trợ các dịch vụ dựa trên giao thức chính và phụ.

Ứng dụng

Các cấp độ dịch vụ chính và phụ được tạo và đại diện chung cho lớp điểm cuối dịch vụ. Tất cả các dịch vụ đều phải được cân nhắc trong thiết kế theo định hướng dịch vụ tiêu chuẩn và tuân theo các hướng dẫn cụ thể để giảm thiểu tác động của việc không tuân theo Giao thức Canonical.

Tác động

Mẫu này có thể dẫn đến cấu trúc khoảng không quảng cáo phức tạp, tăng nồng lực và chi phí quản trị, và (khi áp dụng) sự phụ thuộc không lành mạnh vào Giao thức kết nối. Bởi vì lớp điểm cuối là bán liên kết nên số lượng tiềm năng người tiêu dùng và cơ hội tái sử dụng bị giảm.

Nguyên tắc

Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lồng léo (293), Trừu tượng hóa dịch vụ (294), Tự chủ dịch vụ (297), Dịch vụ Khả năng kết hợp (302)

Ngành kiến trúc

Hàng tồn kho, Dịch vụ

Hàng tồn kho doanh nghiệp

Bởi Thomas Erl



Làm cách nào để phân phối các dịch vụ nhằm tối đa hóa khả năng kết hợp lại?

Vấn đề	Việc cung cấp dịch vụ một cách độc lập thông qua các nhóm dự án khác nhau trong toàn doanh nghiệp sẽ tạo ra rủi ro thường trực về việc tạo ra các triển khai kiến trúc và dịch vụ không nhất quán, ảnh hưởng đến cơ hội tái cấu trúc.
Giải pháp	Các dịch vụ cho nhiều giải pháp có thể được thiết kế để phân phối trong kiến trúc kiểm kê được tiêu chuẩn hóa trên toàn doanh nghiệp, trong đó chúng có thể được tự do và tái tạo lại nhiều lần.
Ứng dụng	Kho dịch vụ doanh nghiệp được lập mô hình trước một cách lý tưởng, và các tiêu chuẩn toàn doanh nghiệp được áp dụng cho các dịch vụ do các nhóm dự án khác nhau cung cấp.
Tác động	Cần phải phân tích trước một cách đáng kể để xác định kế hoạch chi tiết kiểm kê của doanh nghiệp và nhiều tác động về mặt tổ chức xuất phát từ các yêu cầu quản trị tiếp theo.
Nguyên tắc	Hợp đồng dịch vụ được tiêu chuẩn hóa (291), Trữu tượng hóa dịch vụ (294), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Doanh nghiệp, Hàng tồn kho

Trùu tượng hóa thực thể

Bởi Thomas Erl



Làm thế nào logic kinh doanh bất khả tri có thể được tách ra, tái sử dụng và quản lý độc lập?

Vấn đề

Việc kết hợp cá logic kinh doanh theo quy trình và quy trình cụ thể vào cùng một dịch vụ cuối cùng sẽ dẫn đến việc tạo ra logic kinh doanh bất khả tri dư thừa trên nhiều dịch vụ.

Giải pháp

Một lớp dịch vụ kinh doanh bất khả tri có thể được thiết lập, dành riêng đến các dịch vụ dựa trên bối cảnh chức năng của chúng trên các thực thể kinh doanh hiện có.

Ứng dụng

Bối cảnh dịch vụ thực thể có nguồn gốc từ các mô hình thực thể kinh doanh và sau đó thiết lập một lớp logic được mô hình hóa trong giai đoạn phân tích.

Tác động

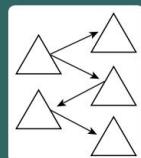
Bản chất cốt lõi, lấy kinh doanh làm trung tâm của các dịch vụ được giới thiệu bởi mô hình này đòi hỏi sự chú ý nhiều hơn về mô hình hóa và thiết kế, đồng thời các yêu cầu quản trị của chúng có thể tạo ra những thay đổi mạnh mẽ về mặt tổ chức.

Nguyên tắc

Khớp nối lồng lèo dịch vụ (293), Trùu tượng hóa dịch vụ (294), Dịch vụ Khả năng sử dụng lại (295), Khả năng kết hợp dịch vụ (302)

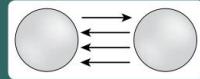
Ngành kiến trúc

Hàng tồn kho, Thành phần, Dịch vụ

Liên kết thực thể	
Bởi Raj Balasubramanian, David Booth, Thomas Erl	
Làm thế nào các dịch vụ có thể thể hiện mối quan hệ vốn có giữa các thực thể kinh doanh để hỗ trợ thành phần liên kết lồng lèo?	
Vấn đề	Các thực thể kinh doanh có các mối quan hệ tự nhiên, tuy nhiên các dịch vụ của thực thể lại thường được thiết kế tự động mà không có dấu hiệu nào về các mối quan hệ này. Người sử dụng dịch vụ đóng vai trò là người kiểm soát thành phần thường phải có logic liên kết thực thể được mã hóa cứng để làm việc với các mối quan hệ của thực thể. Điều này giới hạn bô điều khiển thành phần đối với bất kỳ liên kết bổ sung nào có thể trở nên phù hợp và tăng thêm gánh nặng quản trị để đảm bảo rằng logic liên kết thực thể được mã hóa cứng được giữ đồng bộ với doanh nghiệp.
Giải pháp	Dịch vụ thông báo cho người tiêu dùng về sự tồn tại của các thực thể liên quan như một phần trong tương tác của người tiêu dùng với dịch vụ.
Ứng dụng	Các liên kết được bao gồm trong các tin nhắn phản hồi có liên quan từ dịch vụ. Người tiêu dùng dịch vụ có thể di chuyển hướng từ thực thể này sang thực thể khác bằng cách di theo các liên kết này và tích lũy thêm kiến thức kinh doanh dọc đường. Điều này cho phép người tiêu dùng dịch vụ có ít logic liên kết thực thể ban đầu có thể soạn thảo chính xác các dịch vụ thực thể dựa trên mối quan hệ của chúng.
Tác động	Các mã định danh tài nguyên đại diện cho các thực thể kinh doanh cần duy trì tương đối ổn định trong suốt vòng đời của các thực thể kinh doanh mà chúng xác định. Khi một thông tin nhận dạng được biết đến, nó có thể được nhắc lại trong tương lai bởi những người tiêu dùng dịch vụ tương tự. Có thể khó xác định các liên kết nếu số nhận dạng của các thực thể kinh doanh cụ thể đối với các dịch vụ sở hữu chúng. Việc áp dụng Điểm cuối nhẹ có thể giúp đạt được cú pháp thống nhất cho các số nhận dạng được liên kết. Các liên kết không có giá trị nếu người tiêu dùng dịch vụ không thể truy cập thông tin về thực thể được liên kết. Vì thế, càng việc áp dụng Hợp đồng tái sử dụng [355] có thể đảm bảo rằng người tiêu dùng dịch vụ có thể tương tác với các thực thể được liên kết.
Nguyên tắc	Khả năng sử dụng lại dịch vụ (295), Trữu tượng hóa dịch vụ (294), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Hàng tồn kho, Dịch vụ

Nhắn tin theo hướng sự kiện

Bởi Mark Little, Thomas Rischbeck, Arnaud Simon



Làm cách nào người tiêu dùng dịch vụ có thể được thông báo tự động về các sự kiện dịch vụ trong thời gian chạy?

Vấn đề

Các sự kiện xảy ra trong ranh giới chức năng được gói gọn bởi một dịch vụ có thể liên quan đến người tiêu dùng dịch vụ, nhưng không sử dụng sự tương tác dựa trên cuộc bỏ phiếu không hiệu quả, người tiêu dùng có không có cách nào để tìm hiểu về những sự kiện này.

Giải pháp

Người tiêu dùng tự thiết lập mình như một người đăng ký dịch vụ. Đến lượt mình, dịch vụ sẽ tự động đưa ra thông báo về các sự kiện liên quan với điều này và bất kỳ người đăng ký nào của nó.

Ứng dụng

Khung nhắn tin được triển khai có khả năng hỗ trợ MEP xuất bản và đăng ký cũng như xử lý và theo dõi sự kiện phức tạp liên quan.

Tác động

Việc trao đổi tin nhắn theo hướng sự kiện không thể dễ dàng được kết hợp như một phần của Giao dịch dịch vụ nguyên tử [324] và nhà xuất bản/người đăng ký vấn đề săn có có thể phát sinh.

Nguyên tắc

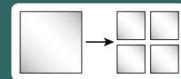
Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293), Tự chủ dịch vụ (297)

Ngành kiến trúc

Hàng tồn kho, Thành phần

Suy giảm chức năng

Bởi Thomas Erl



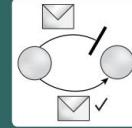
Làm cách nào để giải quyết một vấn đề kinh doanh lớn mà không cần phải xây dựng một nhóm giải pháp logic độc lập?

Vấn đề	Để giải quyết một vấn đề kinh doanh lớn, phức tạp, một lượng tương ứng logic giải pháp cần được tạo ra, dẫn đến một hệ thống khép kín ứng dụng với các hạn chế về quản trị truyền thống và khả năng sử dụng lại.
Giải pháp	Vấn đề kinh doanh lớn có thể được chia thành một tập hợp các vấn đề nhỏ hơn, có liên quan, cho phép giải pháp logic cần thiết cũng được phân hủy thành một tập tương ứng nhỏ hơn, có liên quan đơn vị logic giải pháp.
Ứng dụng	Tùy thuộc vào bản chất của vấn đề lớn, một hướng dịch vụ Quá trình phân tích có thể được tạo ra để giải mã nó thành các vấn đề nhỏ hơn một cách rõ ràng.
Tác động	Việc sở hữu nhiều chương trình nhỏ hơn có thể làm tăng độ phức tạp trong thiết kế và thách thức quản trị.
Nguyên tắc	không có
Ngành kiến trúc	Dịch vụ

Khả năng bình thường

Bởi Cesare Pautasso, Herbjørn Wilhelmsen

Làm cách nào một khả năng dịch vụ có thể chấp nhận nhiều bản sao của cùng một thông báo một cách an toàn để xử lý lỗi giao tiếp?



Vấn đề

Lỗi phần cứng mạng và máy chủ có thể dẫn đến mất tin nhắn, dẫn đến trường hợp người tiêu dùng dịch vụ không nhận được phản hồi đối với yêu cầu của mình. Nỗ lực phát hành lại thông báo yêu cầu có thể dẫn đến hành vi không thể đoán trước hoặc không mong muốn khi khả năng dịch vụ vô tình nhận được nhiều bản sao của cùng một thông báo yêu cầu.

Giải pháp

Thiết kế các khả năng dịch vụ với logic bình thường cho phép chúng chấp nhận việc trao đổi tin nhắn lặp đi lặp lại một cách an toàn.

Ứng dụng

Idempotency đảm bảo rằng việc yêu cầu lặp lại khả năng dịch vụ là an toàn và sẽ không có tác động tiêu cực.

Khả năng bình thường thường bị giới hạn ở việc truy xuất và truy vấn dữ liệu chỉ đọc. Đối với các khả năng yêu cầu thay đổi trạng thái dịch vụ, logic của chúng thường dựa trên "đặt", "đặt" hoặc "xóa" các hành động có hậu điều kiện không phụ thuộc vào trạng thái ban đầu của dịch vụ.

Thiết kế của một khả năng bình thường có thể bao gồm việc sử dụng một mã định danh duy nhất cho mỗi yêu cầu sao cho các yêu cầu lặp lại (có cùng giá trị định danh) đã được xử lý sẽ bị khả năng dịch vụ loại bỏ hoặc bỏ qua, thay vì bị loại bỏ.

được xử lý lại.

Tác động

Việc sử dụng một mã định danh duy nhất để xác định khả năng bình thường yêu cầu trạng thái phiên phải được dịch vụ ghi lại một cách đáng tin cậy và được bảo toàn trong các lỗi phần cứng máy chủ. Điều này có thể gây tốn hại đến khả năng mở rộng của dịch vụ và có thể phức tạp hơn nếu việc triển khai dịch vụ dự phòng đang hoạt động tại các địa điểm khác nhau gặp phải lỗi mạng.

Không phải tất cả các khả năng dịch vụ đều có thể bình thường. Các khả năng có khả năng không an toàn bao gồm những khả năng cần thực hiện "tăng dần", các hàm chuyển đổi "đảo ngược" hoặc "tăng cường", trong đó điều kiện sau khi thực hiện phụ thuộc vào trạng thái ban đầu của dịch vụ.

Nguyên tắc

Hợp đồng dịch vụ được tiêu chuẩn hóa (291), Dịch vụ không quốc tịch (298), Khả năng kết hợp dịch vụ (302)

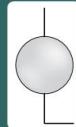
Ngành kiến trúc

Hàng tồn kho, Thành phần, Dịch vụ

Điểm cuối khoảng không quảng cáo

Bởi Thomas Erl

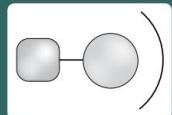
Làm cách nào để bảo vệ kho dịch vụ khỏi sự truy cập từ bên ngoài trong khi vẫn cung cấp khả năng dịch vụ cho người tiêu dùng bên ngoài?



Vấn đề	Một nhóm dịch vụ được phân phối cho một khoảng không quảng cáo cụ thể có thể cung cấp các khả năng hữu ích cho các dịch vụ bên ngoài khoảng không quảng cáo đó. Tuy nhiên, vì lý do an ninh và quản trị, có thể không mong muốn đưa tất cả các dịch vụ hoặc tất cả khả năng của dịch vụ ra bên ngoài người tiêu dùng.
Giải pháp	Tóm tắt các khả năng liên quan vào một dịch vụ điểm cuối hoạt động như một điểm nhập hàng tồn kho chính thức dành riêng cho một nhóm người tiêu dùng bên ngoài cụ thể.
Ứng dụng	Dịch vụ điểm cuối có thể hiển thị một hợp đồng có các khả năng tương tự như các dịch vụ cơ bản của nó, nhưng được tăng cường các chính sách hoặc các đặc điểm khác để đáp ứng các yêu cầu tương tác bên ngoài của người tiêu dùng.
Tác động	Dịch vụ điểm cuối có thể tăng cường quyền tự do quản trị của các dịch vụ cơ bản nhưng cũng có thể tăng cường nỗ lực quản trị bằng cách đưa hợp đồng và logic dịch vụ dự phòng vào kho.
Nguyên tắc	Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293), Trữu tượng hóa dịch vụ (294)
Ngành kiến trúc	Hàng tồn kho

Trình bao bọc kế thừa

Bởi Thomas Erl, Satadru Roy



Làm cách nào để ngăn chặn các dịch vụ bao bọc có hợp đồng phi tiêu chuẩn lan truyền sự kết hợp gián tiếp giữa người tiêu dùng và người thực hiện?

Vấn đề

Các dịch vụ bao bọc cần thiết để đóng gói logic kế thừa thường bị buộc phải đưa ra hợp đồng dịch vụ phi tiêu chuẩn với các yêu cầu ghép nối công nghệ cao, dẫn đến sự gia tăng nhanh chóng kết nối thực hiện xuyên suốt tất cả người tiêu dùng dịch vụ các chương trình.

Giải pháp

Dịch vụ bao bọc không theo tiêu chuẩn có thể được thay thế bằng hoặc được bao bọc thêm bằng một hợp đồng dịch vụ được tiêu chuẩn hóa để trích xuất, đóng gói và có thể loại bỏ các chi tiết kỹ thuật cũ khỏi hợp đồng.

Ứng dụng

Hợp đồng dịch vụ tùy chỉnh và logic dịch vụ bắt buộc cần được phát triển để thể hiện giao diện kế thừa độc quyền.

Tác động

Việc giới thiệu một dịch vụ bổ sung sẽ bổ sung thêm một lớp xử lý và chi phí hoạt động liên quan.

Nguyên tắc

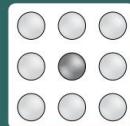
Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293), Trừu tượng hóa dịch vụ (294)

Ngành kiến trúc

Dịch vụ

Tập trung logic

Bởi Thomas Erl



Làm thế nào có thể tránh được việc lạm dụng logic dịch vụ dự phòng?

Vấn đề	Nếu các dịch vụ bắt khả tri không được tái sử dụng một cách nhất quán, thì các dịch vụ dư thừa sẽ chức năng có thể được cung cấp trong các dịch vụ khác, dẫn đến các vấn đề liên quan đến việc không chuẩn hóa hàng tồn kho và quyền sở hữu và quản trị dịch vụ.
Giải pháp	Quyền truy cập vào chức năng có thể tái sử dụng được giới hạn ở các dịch vụ bắt khả tri chính thức.
Ứng dụng	Các dịch vụ bắt khả tri cần phải được thiết kế và quản lý hợp lý, và việc sử dụng chúng phải được thực thi thông qua các tiêu chuẩn doanh nghiệp.
Tác động	Các vấn đề về tổ chức gợi nhớ đến các dự án tái sử dụng trong quá khứ có thể gây trở ngại cho việc áp dụng mô hình này.
Nguyên tắc	Khả năng sử dụng lại dịch vụ (295), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Hàng tồn kho, Thành phần, Dịch vụ

Triển khai vi dịch vụ

Bởi Paulo Merson



Làm cách nào để một dịch vụ có thể được triển khai độc lập nhằm tránh những hạn chế do việc triển khai nguyên khói gây ra?

Vấn đề

Các dịch vụ và các thành phần khác của giải pháp phần mềm được đóng gói cùng nhau trong một gói triển khai nguyên khói. Việc triển khai phiên bản mới của dịch vụ là một phần của giải pháp có thể yêu cầu triển khai lại toàn bộ giải pháp. Ngoài ra, có ít sự linh hoạt hơn để cấu hình khả năng mở rộng theo dịch vụ cụ thể, tính khả dụng, tính bền bỉ, giám sát và logic bảo mật.

Giải pháp

Mỗi dịch vụ được coi như một sản phẩm độc lập và được triển khai như một gói riêng biệt góp phần mang lại quyền tự chủ cho dịch vụ.

Ứng dụng

Các dịch vụ được đóng gói và triển khai trong một môi trường có tính tự trị cao, có thể sử dụng công nghệ đóng gói trong container. Việc đóng gói và triển khai các dịch vụ thường được tự động hóa cao. Các dịch vụ thường được thiết kế để sử dụng với HTTP/REST và hỗ trợ giao tiếp giữa các dịch vụ không đồng bộ.

Tác động

Dịch vụ có thể được phát triển và phát triển độc lập hơn. Việc triển khai dịch vụ có thể được điều chỉnh và các phiên bản mới có thể được phát hành với thời gian ngừng hoạt động tối thiểu. Có thể cần phải tăng dung lượng bộ nhớ và có thể tăng chi phí hiệu suất do nhu cầu liên lạc dựa trên mạng ngày càng tăng.

Nguyên tắc

Tự chủ dịch vụ (297), Khớp nối lỏng lẻo dịch vụ (293)

Ngành kiến trúc

Thành phần, Dịch vụ

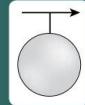
GHI CHÚ

“Microservice” là một thuật ngữ trong ngành có thể được sử dụng cho các dịch vụ tuân theo mô hình dịch vụ vi mô và định hướng dịch vụ đã được áp dụng (và do đó là một phần của môi trường SOA), cũng như cho các dịch vụ không phải là một phần của môi trường SOA. Là một phần của danh mục mẫu SOA, mẫu Triển khai vi dịch vụ chỉ được thiết kế cho các dịch vụ là một phần của môi trường SOA và phổ biến nhất là mẫu Trùu tượng nhiệm vụ vi mô [350] đã được áp dụng.

Trùu tượng hóa nhiệm vụ vi mô	
Bởi Thomas Erl	
Làm thế nào logic phi bất khả tri với các yêu cầu xử lý chuyên biệt có thể được tách biệt và quản lý một cách độc lập?	
Vấn đề	Việc nhóm logic bất khả tri với các yêu cầu triển khai và xử lý chuyên biệt cùng với logic phi bất khả tri không có những yêu cầu như vậy có thể làm tổn hại đến tính chất của logic bất khả tri. khả năng đáp ứng một cách nhất quán các yêu cầu của mình.
Giải pháp	Các đơn vị logic bất khả tri riêng lẻ có yêu cầu xử lý và triển khai chuyên biệt được tách riêng bằng mô hình vi dịch vụ và được trùu tượng hóa thành lớp vi dịch vụ, trong đó có quyền tự do kiến trúc để điều chỉnh môi trường nhằm hỗ trợ các yêu cầu triển khai và xử lý dịch vụ chuyên biệt.
Ứng dụng	Khi logic quy trình kinh doanh bất khả tri đã được tách khỏi logic bất khả tri, nó sẽ được xem xét để xác định các đơn vị logic với yêu cầu xử lý và triển khai chuyên biệt phù hợp với lớp microservice.
Tác động	Việc trùu tượng hóa logic tác vụ vi mô thành một lớp dịch vụ riêng biệt có thể đưa ra chi phí phân tích, thiết kế và quản trị. Mẫu Triển khai microservice [349] thường được áp dụng cho logic tác vụ vi mô để hiện thực hóa môi trường triển khai dịch vụ cần thiết. Điều này có thể đưa ra các giao thức truyền thông khác nhau và yêu cầu hơn nữa công nghệ triển khai chuyên biệt có thể áp đặt cơ sở hạ tầng mới, yêu cầu quản lý và điều hành.
Nguyên tắc	Trùu tượng hóa dịch vụ (294), Tự chủ dịch vụ (297), Dịch vụ Khả năng kết hợp (302), Khớp nối lỏng dịch vụ (293)
Ngành kiến trúc	Thành phần, Hàng tồn kho, Dịch vụ

Bối cảnh phi thuyết bất khả tri

Bởi Thomas Erl



Làm thế nào logic dịch vụ đơn mục đích có thể được định vị như một nguồn lực doanh nghiệp hiệu quả?

Vấn đề

Logic phi bất khả tri không hướng tới dịch vụ có thể hạn chế tính hiệu quả của các thành phần dịch vụ sử dụng các dịch vụ bất khả tri.

Giải pháp

Logic giải pháp bất khả tri phù hợp cho việc đóng gói dịch vụ có thể được đặt trong các dịch vụ cốt trung với tư cách là thành viên chính thức của kho dịch vụ.

Ứng dụng

Bối cảnh dịch vụ chức năng đơn mục đích được xác định.

Tác động

Mặc dù chúng không được kỳ vọng sẽ cung cấp tiềm năng tái sử dụng, nhưng các dịch vụ bất khả tri vẫn phải tuân theo tính chất chẽ của định hướng dịch vụ.

Nguyên tắc

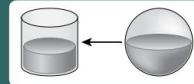
Hợp đồng dịch vụ được tiêu chuẩn hóa (291), Khả năng kết hợp dịch vụ (302)

Ngành kiến trúc

Dịch vụ

Trì hoãn một phần của tiêu bang

Bởi Thomas Erl



Làm cách nào các dịch vụ có thể được thiết kế để tối ưu hóa mức tiêu thụ tài nguyên trong khi vẫn duy trì trạng thái?

Vấn đề	Khả năng dịch vụ có thể được yêu cầu để lưu trữ và quản lý lượng lớn dữ liệu trạng thái, dẫn đến tăng mức tiêu thụ bộ nhớ và giảm khả năng mở rộng.
Giải pháp	Ngay cả khi các dịch vụ được yêu cầu duy trì trạng thái, một tập hợp con của chúng dữ liệu trạng thái có thể được tạm thời trì hoãn.
Ứng dụng	Có nhiều lựa chọn trì hoãn quản lý nhà nước khác nhau, tùy thuộc vào kiến trúc xung quanh.
Tác động	Trì hoãn quản lý trạng thái một phần có thể làm tăng thêm độ phức tạp của thiết kế và ràng buộc dịch vụ với kiến trúc.
Nguyên tắc	Dịch vụ không quốc tịch (298)
Ngành kiến trúc	Hàng tồn kho, Dịch vụ

Trữu tượng hóa quy trình

Bởi Thomas Erl

Làm thế nào logic quy trình bắt khả tri có thể được tách biệt và quản lý một cách độc lập?



Vấn đề

Việc nhóm logic lấy nhiệm vụ làm trung tâm cùng với logic bắt khả tri về nhiệm vụ sẽ cản trở việc quản lý logic dành riêng cho nhiệm vụ và việc sử dụng lại logic bắt khả tri.

Giải pháp

Lớp dịch vụ quy trình công việc gốc chuyên dụng được thiết lập để hỗ trợ tính độc lập trong quản trị và định vị các dịch vụ nhiệm vụ là tài nguyên doanh nghiệp tiềm năng.

Ứng dụng

Logic quy trình nghiệp vụ thường được lọc ra sau khi các dịch vụ tiện ích và thực thể đã được xác định, cho phép xác định nhiệm vụ các dịch vụ bao gồm lớp này.

Tác động

Ngoài việc cân nhắc về mô hình hóa và thiết kế liên quan đến việc tạo ra các dịch vụ tác vụ, việc trữu tượng hóa quy trình kinh doanh gốc logic thiết lập sự phụ thuộc vốn có vào việc thực hiện logic đó thông qua việc kết hợp các dịch vụ khác.

Nguyên tắc

Khớp nối lồng lèo dịch vụ (293), Trữu tượng hóa dịch vụ (294), Dịch vụ Khả năng kết hợp (302)

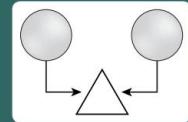
Ngành kiến trúc

Hàng tồn kho, Thành phần, Dịch vụ

Triển khai dự phòng	
Bởi Thomas Erl	
Làm thế nào để tăng độ tin cậy và tính sẵn sàng của một dịch vụ?	
Vấn đề	Một dịch vụ đang được tích cực sử dụng lại sẽ có một điểm lỗi tiềm ẩn có thể gây nguy hiểm cho độ tin cậy của tất cả các thành phần mà nó tham gia nếu xảy ra tình trạng lỗi không mong muốn.
Giải pháp	Các dịch vụ tái sử dụng có thể được triển khai thông qua việc triển khai dự phòng hoặc hỗ trợ chuyển đổi dự phòng.
Ứng dụng	Việc triển khai dịch vụ tương tự được triển khai dự phòng hoặc được hỗ trợ bởi cơ sở hạ tầng có tính năng dự phòng.
Tác động	Cần có nỗ lực quản trị bổ sung để giữ cho tất cả các hoạt động triển khai dự phòng được đồng bộ.
Nguyên tắc	Tự chủ dịch vụ (297)
Ngành kiến trúc	Dịch vụ

Hợp đồng tái sử dụng

Bởi Raj Balasubramanian, Benjamin Carlyle, Thomas Erl, Cesare Pautasso



Làm thế nào người tiêu dùng dịch vụ có thể soạn thảo các dịch vụ mà không cần phải tuân theo các hợp đồng dành riêng cho dịch vụ?

Vấn đề

Để truy cập khả năng dịch vụ của một dịch vụ bằng hợp đồng dành riêng cho dịch vụ, người tiêu dùng dịch vụ phải được thiết kế để tự kết hợp với hợp đồng dịch vụ. Khi hợp đồng dịch vụ thay đổi, người sử dụng dịch vụ có thể không còn chức năng. Để truy cập phiên bản mới của hợp đồng dịch vụ hoặc để truy cập các phiên bản khác hợp đồng dịch vụ để tạo ra các dịch vụ khác, người tiêu dùng dịch vụ phải trải qua các chu kỳ phát triển bổ sung, do đó phát sinh thời gian, công sức, chi phí.

Giải pháp

Giới hạn sự liên kết chặt chẽ với một hợp đồng kỹ thuật chung, có thể tái sử dụng được chia sẻ bởi nhiều dịch vụ. Hợp đồng kỹ thuật chỉ cung cấp chung, cấp cao các chức năng ít có khả năng bị ảnh hưởng khi logic dịch vụ thay đổi.

Ứng dụng Một hợp đồng dịch vụ có thể tái sử dụng có thể cung cấp các phương pháp trao đổi dữ liệu trữ lượng và bắt khắt trại, không có phương pháp nào liên quan đến một chức năng kinh doanh cụ thể. phương pháp trong hợp đồng có thể sử dụng lại thường tập trung vào các loại dữ liệu hơn là bối cảnh kinh doanh của dữ liệu.

Tập hợp các phương thức của hợp đồng có thể tái sử dụng được bổ sung bởi các mã định danh tài nguyên dành riêng cho dịch vụ và các loại phương tiện để áp dụng bối cảnh được thiết lập bằng các phương pháp có thể tái sử dụng cho các khả năng dịch vụ riêng lẻ.

HTTP cung cấp một hợp đồng có thể tái sử dụng thông qua các phương thức chung, chẳng hạn như GET, PUT, và DELETE, cho phép các chương trình tiêu dùng truy cập các tài nguyên dựa trên Web bằng cách cung cấp thêm số nhận dạng tài nguyên. Sự kết hợp giữa mã định danh tài nguyên, phương thức HTTP và loại phương tiện có thể bao gồm khả năng dành riêng cho dịch vụ.

Một hợp đồng có thể tái sử dụng cũng có thể được tạo bằng cách sử dụng định nghĩa WSDL tập trung, miễn là các phép toán được xác định đủ chung chung.

Tác động

Việc chia sẻ cùng một hợp đồng giữa các dịch vụ sẽ làm tăng tầm quan trọng của việc đạt được hợp đồng đúng đắn, cả ở giai đoạn đầu và trong suốt thời hạn của hợp đồng.

Hợp đồng có thể sử dụng lại có thể vẫn cần thay đổi nếu các dịch vụ mới có yêu cầu chức năng cấp cao mới được đưa vào kho dịch vụ.

Hợp đồng có thể tái sử dụng có thể thiếu dữ liệu dữ liệu để cho phép khám phá một dịch vụ một cách hiệu quả. Siêu dữ liệu dành riêng cho dịch vụ có thể cần được duy trì tách biệt với định nghĩa hợp đồng có thể sử dụng lại để đảm bảo rằng người tiêu dùng dịch vụ có thể chọn khả năng dịch vụ chính xác để tương tác.

Nguyên tắc

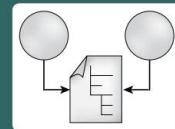
Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối lòng lèo dịch vụ (293), Dịch vụ Trữu tượng hóa (294), Khả năng khám phá dịch vụ (300), Khả năng kết hợp dịch vụ (302)

Kho kiến trúc, Thành phần, Dịch vụ

Tập trung lược đồ

Bởi Thomas Erl

Hợp đồng dịch vụ có thể được thiết kế như thế nào để tránh việc trình bày dữ liệu dư thừa?



Vấn đề	Các hợp đồng dịch vụ khác nhau thường cần thể hiện khả năng xử lý các tài liệu hoặc tập dữ liệu kinh doanh tương tự, dẫn đến nội dung lược đồ dư thừa khó quản lý.
Giải pháp	Chọn các lược đồ tồn tại dưới dạng các phần riêng biệt về mặt vật lý của hợp đồng dịch vụ được chia sẻ trên nhiều hợp đồng.
Ứng dụng	Cần có nỗ lực phân tích trước để thiết lập lớp lược đồ độc lập và hỗ trợ lớp dịch vụ.
Tác động	Việc quản lý các lược đồ dùng chung ngày càng trở nên quan trọng vì nhiều dịch vụ có thể hình thành sự phụ thuộc vào cùng một định nghĩa lược đồ.
Nguyên tắc	Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293)
Ngành kiến trúc	Hàng tồn kho, Dịch vụ

Đại lý dịch vụ

Bởi Thomas Erl



Làm thế nào logic hướng sự kiện có thể được tách biệt và quản lý độc lập?

Vấn đề

Các thành phần dịch vụ có thể trở nên lớn và kém hiệu quả, đặc biệt là khi được yêu cầu gọi rõ ràng, do đó giảm kích thước và cung thẳng hiệu suất của các thành phần dịch vụ.

Giải pháp

Logic hướng sự kiện có thể được chuyển sang các chương trình hướng sự kiện không yêu cầu gọi rõ ràng, do đó giảm kích thước và cung thẳng hiệu suất của các thành phần dịch vụ.

Ứng dụng

Các đại lý dịch vụ có thể được thiết kế để tự động đáp ứng các điều kiện được xác định trước mà không cần yêu cầu thông qua hợp đồng được công bố.

Tác động

Độ phức tạp của logic tổng hợp tăng lên khi nó được phân phối trên các dịch vụ, các tác nhân hướng sự kiện và sự phụ thuộc vào dịch vụ
các đại lý có thể gắn kết hơn nữa kiến trúc hàng tồn kho với công nghệ độc quyền của nhà cung cấp.

Nguyên tắc

Khớp nối lồng lèo dịch vụ (293), Khả năng tái sử dụng dịch vụ (295)

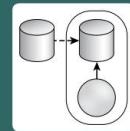
Ngành kiến trúc

Hàng tồn kho, Thành phần

Sao chép dữ liệu dịch vụ

Bởi Thomas Erl

Làm cách nào để duy trì quyền tự chủ của dịch vụ khi các dịch vụ yêu cầu quyền truy cập vào các nguồn dữ liệu được chia sẻ?



Vấn đề	Logic dịch vụ có thể được triển khai độc lập để tăng tính tự chủ của dịch vụ, nhưng các dịch vụ vẫn tiếp tục mất quyền tự chủ khi yêu cầu truy cập vào các nguồn dữ liệu được chia sẻ.
Giải pháp	Các dịch vụ có thể có cơ sở dữ liệu chuyên dụng riêng với khả năng sao chép sang các nguồn dữ liệu được chia sẻ.
Ứng dụng	Cần cung cấp cơ sở dữ liệu bổ sung cho dịch vụ và cần bật một hoặc nhiều kênh sao chép giữa cơ sở dữ liệu đó và các nguồn dữ liệu được chia sẻ.
Tác động	Mô hình này dẫn đến chi phí và nhu cầu cơ sở hạ tầng bổ sung, và việc quản lý quá nhiều kênh sao chép có thể khó khăn.
Nguyên tắc	Tự chủ dịch vụ (297)
Ngành kiến trúc	Hàng tồn kho, Dịch vụ

Đóng gói dịch vụ

Bởi Thomas Erl



Làm thế nào logic giải pháp có thể được cung cấp như một nguồn tài nguyên của doanh nghiệp?

Vấn đề

Logic giải pháp được thiết kế cho một môi trường ứng dụng duy nhất thường bị hạn chế về khả năng tương tác với hoặc được các bộ phận khác của doanh nghiệp tận dụng.

Giải pháp

Logic của giải pháp có thể được gói gọn trong một dịch vụ để nó được định vị như một nguồn tài nguyên doanh nghiệp có khả năng hoạt động vượt ra ngoài ranh giới mà nó được phân phối ban đầu.

Ứng dụng

Logic giải pháp phù hợp cho việc đóng gói dịch vụ cần phải được xác định.

Tác động

Logic của giải pháp đóng gói dịch vụ phải được xem xét bổ sung về thiết kế và quản trị.

Nguyên tắc

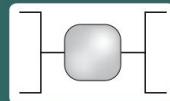
không có

Ngành kiến trúc

Dịch vụ

Mặt tiền dịch vụ

Bởi Thomas Erl



Làm cách nào một dịch vụ có thể điều chỉnh các thay đổi đối với hợp đồng hoặc cách triển khai của nó trong khi vẫn cho phép logic dịch vụ cốt lõi phát triển độc lập?

Vấn đề	Việc kết hợp logic dịch vụ cốt lõi với các hợp đồng và tài nguyên triển khai có thể cản trở sự phát triển của nó và tác động tiêu cực đến người tiêu dùng dịch vụ.
Giải pháp	Thành phần mặt tiền dịch vụ được sử dụng để trừu tượng hóa một phần kiến trúc dịch vụ có khả năng ghép âm.
Ứng dụng	Một thành phần mặt tiền riêng biệt được tích hợp vào thiết kế dịch vụ.
Tác động	Việc bổ sung thành phần mặt tiền giới thiệu nỗ lực thiết kế và chi phí hiệu suất.
Nguyên tắc	Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293)
Ngành kiến trúc	Dịch vụ

Chuẩn hóa dịch vụ

Bởi Thomas Erl

Làm cách nào để kho dịch vụ có thể tránh được logic dịch vụ dư thừa?



Vấn đề

Khi cung cấp dịch vụ như một phần của kho dịch vụ, có rủi ro thường trực là các dịch vụ sẽ được tạo ra với ranh giới chức năng chéo, gây khó khăn cho việc phổ biến rộng rãi.
tái sử dụng.

Giải pháp

Danh mục dịch vụ cần được thiết kế chú trọng vào việc cẩn chỉnh ranh giới dịch vụ.

Ứng dụng

Ranh giới dịch vụ chức năng được mô hình hóa như một phần của quy trình phân tích chính thức và tồn tại trong suốt quá trình thiết kế và quản lý hàng tồn kho.

Tác động

Việc đảm bảo rằng các ranh giới dịch vụ luôn được cẩn chỉnh phù hợp sẽ đưa ra những phân tích ban đầu bổ sung và nỗ lực quản trị liên tục.

Nguyên tắc

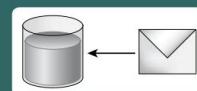
Tự chủ dịch vụ (297)

Ngành kiến trúc

Hàng tồn kho, Dịch vụ

Tin nhắn trạng thái

Bởi Anish Karmarkar

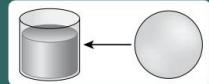


Làm thế nào một dịch vụ có thể duy trì trạng thái không trạng thái trong khi tham gia vào các tương tác có trạng thái?

Vấn đề	Khi các dịch vụ được yêu cầu duy trì thông tin trạng thái trong bộ nhớ giữa các lần trao đổi tin nhắn với người tiêu dùng, khả năng mở rộng có thể được bao gồm, và chúng có thể trở thành một hiệu suất gánh nặng lên cơ sở hạ tầng xung quanh.
Giải pháp	Thay vì giữ lại dữ liệu trạng thái trong bộ nhớ, việc lưu trữ nó được tạm thời được ủy quyền cho tin nhắn.
Ứng dụng	Tùy thuộc vào cách áp dụng mẫu này, cả dịch vụ và người tiêu dùng có thể cần được thiết kế để xử lý dữ liệu trạng thái dựa trên thông báo.
Tác động	Mẫu này có thể không phù hợp với mọi dạng dữ liệu trạng thái và nếu tin nhắn bị mất, mọi thông tin trạng thái mà chúng mang theo có thể bị cũng bị mất.
Nguyên tắc	Hợp đồng dịch vụ được tiêu chuẩn hóa (201), Dịch vụ không quốc tịch (298), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Thành phần, Dịch vụ

Kho lưu trữ nhà nước

Bởi Thomas Erl



Làm cách nào dữ liệu trạng thái dịch vụ có thể được duy trì trong thời gian dài mà không tiêu tốn tài nguyên thời gian chạy dịch vụ?

Vấn đề

Một lượng lớn dữ liệu trạng thái được lưu vào bộ nhớ đệm để hỗ trợ hoạt động trong một thành phần dịch vụ đang chạy có thể tiêu tốn quá nhiều bộ nhớ, đặc biệt là đối với các hoạt động kéo dài, do đó làm giảm khả năng mở rộng.

Giải pháp

Dữ liệu trạng thái có thể được ghi tạm thời vào và sau đó được lấy ra từ kho lưu trữ trạng thái chuyên dụng.

Ứng dụng

Kho lưu trữ dùng chung hoặc dành riêng được cung cấp như một phần của kho lưu trữ hoặc kiến trúc dịch vụ.

Tác động

Việc bổ sung chức năng ghi và đọc cần thiết sẽ làm tăng độ phức tạp trong thiết kế dịch vụ và có thể ảnh hưởng tiêu cực đến hiệu suất.

Nguyên tắc

Dịch vụ không quốc tịch (298)

Ngành kiến trúc

Hàng tồn kho, Dịch vụ

Trừu tượng tiện ích

Bởi Thomas Erl

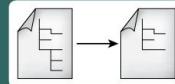


Làm thế nào logic thông thường tập trung vào phi kinh doanh có thể được tách ra, tái sử dụng, và được quản lý độc lập?

Vấn đề	Khi logic xử lý không tập trung vào nghiệp vụ được đóng gói cùng với logic dành riêng cho nghiệp vụ, điều đó sẽ dẫn đến dư thừa thực hiện các chức năng tiện ích chung trên các dịch vụ khác nhau.
Giải pháp	Một lớp dịch vụ dành riêng cho xử lý tiện ích được thiết lập, cung cấp các dịch vụ tiện ích có thể tái sử dụng để sử dụng cho các dịch vụ khác trong kho.
Ứng dụng	Mô hình dịch vụ tiện ích được tích hợp vào các quy trình phân tích và thiết kế để hỗ trợ việc trừu tượng hóa logic tiện ích và các bước tiếp theo được dùng để xác định bởi cảnh dịch vụ cân bằng.
Tác động	Khi logic tiện ích được phân phối trên nhiều dịch vụ, nó có thể làm tăng nhu cầu về kích thước, độ phức tạp và hiệu suất của sáng tác.
Nguyên tắc	Khớp nối lồng lèo dịch vụ (293), Trừu tượng hóa dịch vụ (294), Dịch vụ Khả năng sử dụng lại (295), Khả năng kết hợp dịch vụ (302)
Ngành kiến trúc	Hàng tồn kho, Thành phần, Dịch vụ

Trữu tượng xác thực

Bởi Thomas Erl



Hợp đồng dịch vụ có thể được thiết kế như thế nào để dễ dàng thích ứng hơn với những thay đổi logic xác thực?

Vấn đề

Hợp đồng dịch vụ chứa các ràng buộc xác thực chi tiết sẽ dễ dàng bị vô hiệu hơn khi các quy tắc đãng sau những ràng buộc đó thay đổi.

Giải pháp

Logic và quy tắc xác thực chi tiết có thể được trữu tượng hóa khỏi hợp đồng dịch vụ, do đó làm giảm mức độ chi tiết của ràng buộc và tăng thời hạn tiềm năng của hợp đồng.

Ứng dụng

Logic và quy tắc xác thực trữu tượng cần được chuyển sang logic dịch vụ cơ bản, một dịch vụ khác, tác nhân dịch vụ hoặc ở nơi khác.

Tác động

Mẫu này có thể phân cấp phần nào logic xác thực và cũng có thể làm phức tạp việc tiêu chuẩn hóa lược đồ.

Nguyên tắc

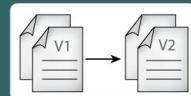
Hợp đồng dịch vụ tiêu chuẩn hóa (291), Khớp nối dịch vụ lỏng lẻo (293), Trữu tượng hóa dịch vụ (294)

Ngành kiến trúc

Dịch vụ

Nhận dạng phiên bản

Bởi David Orchard, Chris Riley



Làm thế nào người tiêu dùng có thể biết được thông tin về phiên bản hợp đồng dịch vụ?

Vấn đề	Khi hợp đồng dịch vụ đã công bố bị thay đổi mà không biêt người tiêu dùng sẽ bỏ lỡ cơ hội tận dụng sự thay đổi hoặc có thể bị tác động tiêu cực bởi sự thay đổi.
Giải pháp	Thông tin phiên bản liên quan đến những thay đổi tương thích và không tương thích có thể được thể hiện như một phần của hợp đồng dịch vụ, cho cả mục đích liên lạc và thực thi.
Ứng dụng	Với hợp đồng dịch vụ Web, số phiên bản có thể được kết hợp thành các giá trị không gian tên và dưới dạng chú thích.
Tác động	Mẫu này có thể yêu cầu thông tin phiên bản được thể hiện bằng từ vựng độc quyền mà các nhà thiết kế người tiêu dùng cần phải hiểu trước.
Nguyên tắc	Hợp đồng dịch vụ tiêu chuẩn (291)
Ngành kiến trúc	Dịch vụ

Phụ lục D



Tuyên bố SOA được chú thích

Tuyên ngôn SOA

Tuyên ngôn SOA đã được khám phá

Tuyên ngôn SOA là một tuyên bố chính thức nhằm xác định rõ ràng về khái niệm SOA. Được biên soạn bởi một nhóm làm việc bao gồm các nhà lãnh đạo tư tưởng trong ngành, Tuyên ngôn SOA đề cập đến các giá trị cốt lõi và các ưu tiên của định hướng dịch vụ. Bằng cách nghiên cứu Tuyên ngôn SOA, chúng ta có thể thu được những quan điểm và hiểu biết sâu sắc có giá trị về mô hình thiết kế hướng dịch vụ.

Phụ lục này trước tiên trình bày Tuyên bố SOA và sau đó chia nhỏ nó để giải thích chi tiết về ý nghĩa và ý nghĩa của các tuyên bố riêng lẻ. Ngoài việc thúc đẩy sự hiểu biết sâu sắc hơn về định hướng dịch vụ, việc khám phá các giá trị và ưu tiên này có thể giúp xác định tính tương thích của chúng với các giá trị, ưu tiên, và mục tiêu.

Tuyên ngôn SOA

Sau đây là nguyên văn Tuyên ngôn SOA, được xuất bản lần đầu tại www.softwaremanifesto.org.

Định hướng dịch vụ là một mô hình định hình những gì bạn làm. Kiến trúc hướng dịch vụ (SOA) là một loại kiến trúc có được từ việc áp dụng định hướng dịch vụ.

Chúng tôi đã và đang áp dụng định hướng dịch vụ để giúp các tổ chức luôn mang lại giá trị kinh doanh bền vững, với tính linh hoạt và hiệu quả chỉ phí tăng lên, phù hợp với sự thay đổi của hoạt động kinh doanh. nhu cầu.

Qua công việc của mình, chúng tôi đã ưu tiên:

- Giá trị kinh doanh hơn chiến lược kỹ thuật
- Mục tiêu chiến lược so với lợi ích cụ thể của dự án
- Khả năng tương tác nội tại qua tích hợp tùy chỉnh
- Các dịch vụ được chia sẻ qua việc triển khai có mục đích cụ thể
- Tính linh hoạt trong việc tối ưu hóa
- Cải tiến hóa qua việc theo đuổi sự hoàn hảo ban đầu

Nghĩa là, trong khi chúng ta coi trọng những món đồ ở bên phải thì chúng ta lại coi trọng những món đồ ở bên trái hơn.

Nguyên tắc hướng dẫn

Chúng tôi tuân theo những nguyên tắc sau:

- Tôn trọng cơ cấu xã hội và quyền lực của tổ chức.
- Nhận thức được rằng SOA cuối cùng đòi hỏi phải thay đổi ở nhiều cấp độ.
- Phạm vi áp dụng SOA có thể khác nhau. Giữ những nỗ lực có thể quản lý được và có ý nghĩa ranh giới.
- Chỉ riêng sản phẩm và tiêu chuẩn sẽ không cung cấp cho bạn SOA cũng như không áp dụng định hướng dịch vụ mô hình cho bạn.
- SOA có thể được hiện thực hóa thông qua nhiều công nghệ và tiêu chuẩn khác nhau.
- Thiết lập một bộ tiêu chuẩn và chính sách thống nhất của doanh nghiệp dựa trên các tiêu chuẩn ngành, thực tế và cộng đồng.
- Theo đuổi sự đồng nhất ở bên ngoài đồng thời cho phép sự đa dạng ở bên trong.
- Xác định các dịch vụ thông qua cộng tác với các bên liên quan về kinh doanh và công nghệ.
- Tối đa hóa việc sử dụng dịch vụ bằng cách xem xét phạm vi sử dụng hiện tại và tương lai.
- Xác minh rằng các dịch vụ có đáp ứng các yêu cầu và mục tiêu kinh doanh hay không.
- Phát triển các dịch vụ và tổ chức của chúng để đáp ứng nhu cầu sử dụng thực tế.
- Tách biệt các khía cạnh khác nhau của một hệ thống có tốc độ thay đổi khác nhau.
- Giảm các phần phụ thuộc tiềm ẩn và xuất bản tất cả các phần phụ thuộc bên ngoài để tăng cường độ mạnh mẽ và giảm thiểu tác động của sự thay đổi.
- Ở mọi mức độ trao đổi, hãy tổ chức từng dịch vụ xung quanh một đơn vị gắn kết và có thể quản lý được về chức năng.

Tuyên ngôn SOA đã được khám phá

Sau thông báo về Tuyên ngôn SOA, một phiên bản có chủ thích đã được soạn thảo cụ thể cho cuốn sách SOA thế hệ tiếp theo: Giới thiệu ngắn gọn về Công nghệ dịch vụ & Định hướng dịch vụ . Nó đã được xuất bản trước tại www.soa-manifesto.com để tạo điều kiện thuận lợi cho việc thảo luận về các tuyên bố của bản tuyên ngôn trong ngành. Được cung cấp trong phần này là nội dung Tuyên bố SOA được chú thích ban đầu với một số sửa đổi nhỏ.

Lời mở đầu

Định hướng dịch vụ là một mô hình định hình những gì bạn làm. Kiến trúc hướng dịch vụ (SOA) là một loại kiến trúc có được từ việc áp dụng định hướng dịch vụ.

Ngay từ đầu, người ta đã hiểu rằng đây là một tuyên ngôn về hai chủ đề riêng biệt nhưng có liên quan chặt chẽ với nhau: mô hình kiến trúc hướng dịch vụ và định hướng dịch vụ, mô hình mà qua đó kiến trúc được xác định. Định dạng này

Tuyên ngôn được mô phỏng theo Tuyên ngôn Agile, trong đó giới hạn nội dung ở những tuyên bố ngắn gọn thể hiện tham vọng, giá trị và nguyên tắc chỉ đạo để hiện thực hóa những tham vọng và giá trị đó. Tuyên ngôn đó không phải là một bản mô tả, một mô hình tham khảo hay thậm chí là một sách trắng và không có tùy chọn cung cấp định nghĩa thực tế, chúng tôi quyết định thêm thông tin này vào lời mở đầu để làm rõ cách thức và lý do các thuật ngữ này được tham chiếu trong các phần khác của tài liệu tuyên ngôn.

Chúng tôi đã và đang áp dụng định hướng dịch vụ.

Mô hình định hướng dịch vụ được xem tốt nhất là một phương pháp hoặc một cách tiếp cận để hiện thực hóa một trạng thái mục tiêu cụ thể được xác định rõ hơn bằng một tập hợp các mục tiêu và lợi ích chiến lược. Khi chúng tôi áp dụng định hướng dịch vụ, chúng tôi định hình các chương trình phần mềm và công nghệ kiến trúc hỗ trợ hiện thực hóa trạng thái mục tiêu này. Đây chính là điều khiến kiến trúc công nghệ được coi là hướng tới dịch vụ.

.để giúp các tổ chức mang lại giá trị kinh doanh bền vững một cách nhất quán với khả năng linh hoạt ngày càng tăng và hiệu quả chi phí.

Phản mở đầu tiếp theo này nêu bật một số lợi ích chiến lược nổi bật nhất và thường được mong đợi nhất của điện toán hướng dịch vụ. Hiểu được những lợi ích này sẽ giúp làm sáng tỏ trạng thái mục tiêu nói trên mà chúng tôi dự định đạt được nhờ áp dụng định hướng dịch vụ.

Sự linh hoạt ở cấp độ doanh nghiệp có thể so sánh với khả năng đáp ứng của một tổ chức. Nhiều hơn một tổ chức có thể ứng phó với những thay đổi trong kinh doanh một cách dễ dàng và hiệu quả thì càng hiệu quả - hiệu quả và thành công thì nó sẽ thích ứng với những tác động của sự thay đổi (và tận dụng hơn nữa bất kỳ lợi ích nào mà sự thay đổi có thể mang lại).

Định hướng dịch vụ định vị dịch vụ là tài sản CNTT dự kiến sẽ cung cấp giá trị lặp lại theo thời gian vượt xa khoản đầu tư ban đầu cần thiết để cung cấp chúng. Hiệu quả chi phí liên quan chủ yếu đến lợi tức đầu tư dự kiến. Bằng nhiều cách,

sự gia tăng hiệu quả chi phí đi đôi với sự gia tăng tính linh hoạt; nếu có
có nhiều cơ hội hơn để sử dụng lại các dịch vụ hiện có và nhìn chung sẽ có ít chi phí hơn
cần thiết để xây dựng các giải pháp mới.

Giá trị doanh nghiệp “bền vững” để cập đến các mục tiêu dài hạn của việc định hướng dịch vụ nhằm
thiết lập các chương trình phần mềm dưới dạng dịch vụ có tính linh hoạt vốn có để liên tục được kết hợp thành
các cấu hình giải pháp mới và được phát triển để đáp ứng các yêu cầu kinh doanh luôn thay đổi.

phù hợp với nhu cầu kinh doanh đang thay đổi.

Sáu từ cuối cùng của lời mở đầu là chia khóa để hiểu triết lý cơ bản của điện toán hướng dịch vụ. Nhu cầu
thích ứng với sự thay đổi kinh doanh liên tục là nền tảng cho định hướng dịch vụ và được coi là mục tiêu chiến
lược tổng thể cơ bản.

Ưu tiên

Thông qua công việc của mình, chúng tôi đã ưu tiên:

Các tuyên bố sắp tới thiết lập một tập hợp các giá trị cốt lõi, mỗi giá trị được thể hiện dưới dạng
ưu tiên hơn một cái gì đó cũng được coi là có giá trị. Mục đích của hệ thống giá trị này là giải quyết các lựa
chọn khó khăn cần được thực hiện thường xuyên để các mục tiêu chiến lược và lợi ích của điện toán hướng dịch vụ
được hiện thực hóa một cách nhất quán.

Giá trị kinh doanh hơn chiến lược kỹ thuật

Như đã nêu trước đây, nhu cầu thích ứng với sự thay đổi trong kinh doanh là một vấn đề bao quát
mục tiêu chiến lược. Vì vậy, chất lượng nền tảng của kiến trúc hướng dịch vụ và
của bất kỳ chương trình, giải pháp và hệ sinh thái phần mềm nào có được từ việc áp dụng
định hướng dịch vụ là họ hướng tới kinh doanh. Vấn đề không phải là công nghệ quyết định hướng đi của doanh
nghiệp; đó là về tầm nhìn kinh doanh quyết định việc sử dụng
thuộc về Công nghệ.

Ưu tiên này có thể có tác động lan tỏa sâu sắc trong các khu vực của doanh nghiệp CNTT. Nó giới thiệu những
thay đổi đối với hầu hết các phần của vòng đời phân phối CNTT, từ cách chúng tôi lập kế hoạch cho
và tài trợ cho các giải pháp tự động hóa về cách chúng tôi xây dựng và quản lý chúng. Tất cả các giá trị và
nguyên tắc khác trong tuyên ngôn, bằng cách này hay cách khác, đều hỗ trợ việc hiện thực hóa giá trị này.

Mục tiêu chiến lược so với lợi ích cụ thể của dự án

Trong lịch sử, nhiều dự án CNTT chỉ tập trung vào việc xây dựng các ứng dụng được thiết kế cụ thể - nhằm tự động hóa các yêu cầu về quy trình kinh doanh hiện hành vào thời điểm đó. Điều này đáp ứng được các nhu cầu (chiến thuật) tức thời, nhưng vì ngày càng có nhiều ứng dụng có mục đích đơn lẻ này được chuyển giao, nó đã tạo ra một doanh nghiệp CNTT chứa đầy các hòn đảo logic và dữ liệu được tham chiếu thành ứng dụng "silo". Khi các yêu cầu kinh doanh mới xuất hiện, các silo mới đã được tạo hoặc các kênh tích hợp giữa các silo được thiết lập. Khi có nhiều thay đổi trong kinh doanh xuất hiện, các kênh tích hợp phải được tăng cường, thậm chí nhiều silo cũng phải được mở rộng hơn. Được tạo ra và chằng bao lâu sau, bối cảnh doanh nghiệp CNTT trở nên phức tạp và ngày càng nặng nề, tốn kém và chậm phát triển.

Theo nhiều cách, định hướng dịch vụ đã xuất hiện để giải quyết những vấn đề này. Đây là một mô hình cung cấp giải pháp thay thế cho việc phát triển ứng dụng tích hợp, dựa trên từng dự án cụ thể và theo từng dự án bằng cách kiên quyết ưu tiên đạt được các mục tiêu chiến lược, dài hạn.

mục tiêu kinh doanh. Trạng thái mục tiêu được ủng hộ bởi định hướng dịch vụ không có các kho ứng dụng truyền thống. Và ngay cả khi các tài nguyên kế thừa và kho ứng dụng tồn tại trong các môi trường áp dụng định hướng dịch vụ, thì trạng thái mục tiêu là trạng thái mà chúng được hài hòa ở mức độ khả thi nhất.

Khả năng tương tác nội tại qua tích hợp tùy chỉnh

Để các chương trình phần mềm chia sẻ dữ liệu, chúng cần có khả năng tương tác. Nếu các chương trình phần mềm không được thiết kế để tương thích thì chúng sẽ không thể tương tác được. Để kích hoạt khả năng tương tác giữa các chương trình phần mềm không tương thích đòi hỏi chúng phải được tích hợp. Do đó, tích hợp là nỗ lực cần thiết để đạt được khả năng tương tác giữa các chương trình phần mềm khác nhau.

Mặc dù thường cần thiết nhưng việc tích hợp tùy chỉnh có thể tốn kém và mất thời gian và có thể dẫn đến những kiến trúc mông manh, khó phát triển. Một trong những mục tiêu của định hướng dịch vụ là giảm thiểu nhu cầu tích hợp tùy chỉnh bằng cách định hình các chương trình phần mềm (trong một miền nhất định) sao cho chúng tương thích nguyên gốc. Đây là một phẩm chất được gọi là khả năng tương tác nội tại. Các nguyên tắc thiết kế bao gồm mô hình định hướng dịch vụ đều hướng tới việc thiết lập khả năng tương tác nội tại ở nhiều cấp độ.

Khả năng tương tác nội tại, như một đặc điểm của các chương trình phần mềm nằm trong một miền nhất định, là chìa khóa để hiện thực hóa các lợi ích chiến lược, chẳng hạn như tăng hiệu quả chi phí và sự nhanh nhẹn.

Các dịch vụ được chia sẻ qua việc triển khai mục đích c cụ thể

Khi được áp dụng ở mức độ có ý nghĩa, các nguyên tắc định hướng dịch vụ sẽ định hình một phần mềm chương trình thành một đơn vị logic hướng dịch vụ có thể được gọi một cách hợp pháp là một dịch vụ.

Các dịch vụ được trang bị các đặc điểm cụ thể (chẳng hạn như các đặc điểm cho phép khả năng tương tác nội tại) hỗ trợ trực tiếp trạng thái mục tiêu được mô tả trước đó. Một trong những đặc điểm này, được bồi dưỡng cụ thể bằng việc áp dụng Khả năng sử dụng lại dịch vụ (295)

nguyên tắc, là sự đóng gói logic đa mục đích có thể được chia sẻ và tái sử dụng trong hỗ trợ tự động hóa các quy trình kinh doanh khác nhau.

Dịch vụ chia sẻ tự thiết lập như một tài sản CNTT có thể cung cấp giá trị kinh doanh lặp lại đồng thời giảm chi phí và nỗ lực cung cấp các giải pháp tự động hóa mới.

Mặc dù có giá trị trong các ứng dụng truyền thống, đơn mục đích giải quyết các yêu cầu kinh doanh mang tính chiến thuật, việc sử dụng các dịch vụ dùng chung mang lại giá trị lớn hơn trong việc hiện thực hóa mục tiêu chiến lược của điện toán hướng dịch vụ (một lần nữa bao gồm sự tăng hiệu quả chi phí và tính linh hoạt).

Tinh linh hoạt trong tối ưu hóa

Đây có lẽ là tuyên bố rộng nhất trong số các tuyên bố về ưu tiên giá trị và được xem tốt nhất như một triết lý hướng dẫn về cách ưu tiên tốt hơn các cân nhắc khác nhau khi cung cấp và phát triển các dịch vụ riêng lẻ cũng như kho dịch vụ.

Tối ưu hóa chủ yếu đề cập đến việc hoàn thành các lợi ích chiến thuật bằng cách điều chỉnh một thiết kế ứng dụng nhất định hoặc đầy nhanh quá trình phân phối ứng dụng đó để đáp ứng nhu cầu trước mắt. Không có gì không mong muốn về điều này, ngoại trừ việc nó có thể dẫn đến các môi trường dựa silo nói trên khi không được ưu tiên hợp lý trong việc thúc đẩy tính linh hoạt.

Ví dụ, đặc tính linh hoạt vượt xa khả năng các dịch vụ chia sẻ dữ liệu một cách hiệu quả (và về bản chất). Để thực sự đáp ứng các yêu cầu kinh doanh luôn thay đổi, các dịch vụ cũng phải linh hoạt trong cách chúng có thể được kết hợp và tổng hợp thành các giải pháp tổng hợp. Không giống như các ứng dụng phân tán truyền thống thường tương đối tĩnh mặc dù thực tế là chúng đã được thành phần hóa, các thành phần dịch vụ cần được thiết kế với mức độ linh hoạt vốn có cho phép tăng cường liên tục. Điều này có nghĩa là khi một quy trình kinh doanh hiện tại thay đổi hoặc khi một quy trình kinh doanh mới được giới thiệu, chúng tôi cần có khả năng thêm, xóa và mở rộng.

các dịch vụ trong kiến trúc thành phần với nỗ lực (tích hợp) tối thiểu. Đây là lý do tại sao Khả năng kết hợp dịch vụ (302) là một trong những nguyên tắc thiết kế hướng dịch vụ quan trọng.

Cải tiến tiến hóa trong việc theo đuổi sự hoàn hảo ban đầu

Có một điểm dễ nhầm lẫn khi nói đến thuật ngữ “sự nhanh nhẹn” trong môi quan hệ sang định hướng dịch vụ. Một số phương pháp thiết kế ứng hộ việc cung cấp nhanh chóng các chương trình phần mềm để đạt được lợi ích ngay lập tức. Điều này có thể được coi là “sự linh hoạt về mặt chiến thuật” vì trọng tâm là về lợi ích chiến thuật, ngắn hạn. Định hướng dịch vụ ứng hộ việc đạt được sự linh hoạt trong cấp độ tổ chức hoặc doanh nghiệp với mục đích trao quyền cho tổ chức, nói chung là có khả năng đáp ứng với sự thay đổi. Hình thức linh hoạt của tổ chức này cũng có thể được gọi là “sự linh hoạt về mặt chiến lược” vì sự nhấn mạnh vào tính lâu dài trong đó, với mọi chương trình phần mềm chúng tôi cung cấp, chúng tôi muốn hướng tới trạng thái mục tiêu thúc đẩy sự linh hoạt có giá trị chiến lược lâu dài.

Để một doanh nghiệp CNTT có thể kích hoạt tính linh hoạt của tổ chức, nó phải phát triển song song với việc kinh doanh. Nhìn chung, chúng tôi không thể dự đoán doanh nghiệp sẽ cần phát triển như thế nào theo thời gian và do đó ban đầu chúng tôi không thể xây dựng các dịch vụ hoàn hảo. Đồng thời, thường có rất nhiều kiến thức đã có sẵn trong kiến thức kinh doanh hiện có của tổ chức mà có thể được thu thập trong các giai đoạn phân tích và lập mô hình của các dự án SOA.

Thông tin này, cùng với các nguyên tắc định hướng dịch vụ và các phương pháp đã được chứng minh, có thể giúp chúng tôi xác định và xác định một tập hợp các dịch vụ nắm bắt cách thức hoạt động kinh doanh. tồn tại và hoạt động ngày nay đồng thời đủ linh hoạt để thích ứng với những thay đổi của doanh nghiệp theo thời gian.

Nghĩa là, trong khi chúng ta coi trọng những món đồ ở bên phải thì chúng ta lại coi trọng những món đồ ở bên trái hơn.

Bằng cách nghiên cứu cách các giá trị này được ưu tiên, chúng tôi hiểu rõ hơn về những gì phân biệt hướng dịch vụ từ các phương pháp và mô hình thiết kế khác. Ngoài việc thiết lập các tiêu chí cơ bản mà chúng ta có thể sử dụng để xác định xem định hướng dịch vụ tương thích như thế nào đối với một tổ chức nhất định, nó còn có thể giúp xác định mức độ mà định hướng dịch vụ có thể hoặc nên được áp dụng.

Việc đánh giá cao các giá trị cốt lõi cũng có thể giúp chúng ta hiểu được việc thực hiện thành công các dự án SOA trong một số môi trường nhất định có thể gặp khó khăn như thế nào. Ví dụ, một số ưu tiên này có thể xung đột trực tiếp với những niềm tin và sở thích đã được thiết lập. Trong trường hợp như vậy, lợi ích của việc định hướng dịch vụ cần được cân nhắc nỗ lực và tác động mà việc áp dụng chúng có thể có (không chỉ về công nghệ mà còn về tổ chức và văn hóa CNTT).

Các nguyên tắc hướng dẫn sắp tới được cung cấp để giúp giải quyết nhiều loại thách thức này.

Nguyên tắc hướng dẫn

Chúng tôi tuân theo những nguyên tắc sau:

Cho đến nay, tuyên ngôn đã thiết lập được tầm nhìn tổng thể cũng như bộ giá trị cốt lõi gắn liền với tầm nhìn đó. Phần còn lại của tuyên bố bao gồm một bộ nguyên tắc được cung cấp dưới dạng hướng dẫn để tuân thủ các giá trị và hiện thực hóa tầm nhìn.

Điều quan trọng cần ghi nhớ là đây là những nguyên tắc chỉ đạo được soạn thảo một cách cụ thể để hỗ trợ cho bản tuyên ngôn này. Không nên nhầm lẫn chúng với các nguyên tắc thiết kế bao gồm định hướng dịch vụ.

Tôn trọng cơ cấu xã hội và quyền lực của tổ chức.

Một trong những cạm bẫy phổ biến nhất của SOA là tiếp cận việc áp dụng như một sáng kiến lấy công nghệ làm trung tâm. Làm như vậy hầu như luôn dẫn đến thất bại vì đơn giản là chúng ta chưa chuẩn bị cho những tác động không thể tránh khỏi của tổ chức.

Việc áp dụng định hướng dịch vụ là chuyển đổi cách chúng ta tự động hóa hoạt động kinh doanh. Tuy nhiên, bất kể chúng ta có kế hoạch gì để thực hiện sự chuyển đổi này nỗ lực xảy ra, chúng ta phải luôn bắt đầu bằng sự hiểu biết và đánh giá cao tổ chức, cơ cấu, mục tiêu và văn hóa của nó.

Việc áp dụng định hướng dịch vụ là một trải nghiệm của con người. Nó đòi hỏi sự hỗ trợ từ những người có thẩm quyền và yêu cầu văn hóa CNTT áp dụng tư duy chiến lược, lấy cộng đồng làm trung tâm. Chúng ta phải thừa nhận đầy đủ và lập kế hoạch cho mức độ thay đổi tổ chức này để nhận được các cam kết dài hạn cần thiết nhằm đạt được trạng thái mục tiêu của định hướng dịch vụ.

Những loại cản nhắc này không chỉ giúp chúng tôi xác định cách tiến hành tốt nhất với sáng kiến SOA mà còn hỗ trợ chúng tôi trong việc xác định phạm vi và điều kiện phù hợp nhất. cách tiếp cận để áp dụng.

Nhận thức được rằng SOA cuối cùng đòi hỏi phải thay đổi ở nhiều cấp độ.

Có câu nói rằng: "Thành công là chuẩn bị sẵn sàng cho cơ hội". Có lẽ Bài học số một rút ra từ các dự án SOA đã được thực hiện trước đây là chúng ta phải hiểu đầy đủ, sau đó lập kế hoạch và chuẩn bị cho khối lượng và phạm vi thay đổi mang lại do việc áp dụng định hướng dịch vụ. Dưới đây là một số ví dụ.

Định hướng dịch vụ thay đổi cách chúng ta xây dựng các giải pháp tự động hóa bằng cách định vị các chương trình phần mềm là tài sản CNTT có giá trị kinh doanh lâu dài và có thể lặp lại. Phụ thuộc vào mức độ mà cơ sở hạ tầng dựa trên đám mây có thể được tận dụng, một bước tiến đáng kể đầu tư có thể được yêu cầu để tạo ra một môi trường bao gồm các tài sản đó. Hơn nữa, cần có cam kết liên tục để duy trì và tận dụng giá trị của chúng.

Vì vậy, ngay từ đầu, cần phải có những thay đổi về cách chúng ta tài trợ, do lường và duy trì hệ thống trong doanh nghiệp CNTT.

Ngoài ra, do định hướng dịch vụ giới thiệu các dịch vụ được định vị là nguồn lực của doanh nghiệp, sẽ có những thay đổi trong cách chúng ta sở hữu các bộ phận khác nhau của hệ thống cũng như điều chỉnh thiết kế và cách sử dụng chúng, chưa kể đến những thay đổi về cơ sở hạ tầng cần thiết để đảm bảo khả năng mở rộng liên tục và độ tin cậy. Các hệ thống quản trị SOA hoàn thiện và các công nghệ dịch vụ có thể giải quyết những mối lo ngại này.

Phạm vi áp dụng SOA có thể khác nhau. Giữ những nỗ lực có thể quản lý được và trong ranh giới có ý nghĩa.

Một quan niệm sai lầm phổ biến là để hiện thực hóa các mục tiêu chiến lược của điện toán hướng dịch vụ, định hướng dịch vụ phải được áp dụng trên cơ sở toàn doanh nghiệp. Cái này có nghĩa là thiết lập và thực thi các tiêu chuẩn ngành và thiết kế trong toàn doanh nghiệp CNTT để tạo ra một kho lưu trữ toàn doanh nghiệp về các dịch vụ có khả năng tương tác nội tại.

Mặc dù lý tưởng này không có gì sai nhưng nó không phải là mục tiêu thực tế đối với nhiều tổ chức, đặc biệt là những tổ chức có doanh nghiệp CNTT lớn hơn.

Phạm vi phù hợp nhất cho bất kỳ nỗ lực áp dụng SOA cụ thể nào cần phải được xác định dựa trên kết quả của việc lập kế hoạch và phân tích kết hợp với các cân nhắc mang tính thực dụng, chẳng hạn như vì những tác động nói trên đến cơ cấu tổ chức, lĩnh vực quyền lực và những thay đổi về văn hóa được mang lại. Việc tính đến trụ cột Phạm vi cân bằng trong các giai đoạn lập kế hoạch sẽ hỗ trợ xác định phạm vi áp dụng ban đầu phù hợp dựa trên sự trưởng thành và sẵn sàng của tổ chức.

Những yếu tố này tiếp tục giúp xác định phạm vi áp dụng được coi là có thể quản lý được.

Nhưng để bất kỳ nỗ lực áp dụng nào nhằm tạo ra một môi trường giúp doanh nghiệp CNTT phát triển hướng tới trạng thái mục tiêu chiến lược mong muốn thì phạm vi cũng phải có ý nghĩa. Ở nơi khác nói cách khác, nó phải được chuyển giao chéo một cách có ý nghĩa để các bộ sưu tập dịch vụ có thể được phân phối trong mối quan hệ với nhau trong một ranh giới được xác định trước. Nói cách khác, chúng tôi muốn tạo ra "các lục địa dịch vụ" chứ không phải "các hòn đảo dịch vụ" đáng sợ.

Khái niệm xây dựng kho lưu trữ dịch vụ được sở hữu và quản lý độc lập trong các miền của cùng một doanh nghiệp CNTT dựa trên Kho lưu trữ miền [338]

mẫu thiết kế ban đầu được xuất bản như một phần của danh mục mẫu thiết kế SOA tại www.soapaterns.org. Cách tiếp cận này làm giảm nhiều rủi ro thường được quy cho các dự án SOA “vụ nổ lớn” và hơn nữa còn giảm nhẹ tác động của cả hai. Những thay đổi về tổ chức và công nghệ (vì tác động được giới hạn ở phạm vi được phân đoạn và quản lý). Đây cũng là một cách tiếp cận cho phép áp dụng theo từng giai đoạn trong đó mỗi lần có thể thiết lập một kho dịch vụ tên miền.

Chỉ riêng các sản phẩm và tiêu chuẩn sẽ không cung cấp cho bạn SOA cũng như không áp dụng mô hình định hướng dịch vụ cho bạn.

Nguyên tắc chỉ đạo này giải quyết hai quan niệm sai lầm riêng biệt nhưng có liên quan rất nhiều. Đầu tiên là bạn có thể mua được SOA bằng các sản phẩm công nghệ hiện đại, và thứ hai là là giả định rằng việc áp dụng các tiêu chuẩn ngành (như XML, WSDL, SCA, v.v.) đương nhiên sẽ dẫn đến kiến trúc công nghệ hướng dịch vụ.

Cộng đồng nhà cung cấp và tiêu chuẩn ngành đã được ghi nhận là có công trong việc xây dựng sự đổi mới công nghệ dịch vụ hiện đại dựa trên các khuôn khổ và nền tảng không độc quyền.

Mọi thứ từ ảo hóa dịch vụ đến điện toán đám mây và điện toán lưới đã giúp nâng cao tiềm năng xây dựng các giải pháp hướng dịch vụ tinh vi và phức tạp. Tuy nhiên, không có công nghệ nào trong số này là dành riêng cho SOA. Bạn có thể giống như

dễ dàng xây dựng các hệ thống dựa trên silo trên đám mây như bạn có thể làm trên các máy chủ riêng của mình.

Không có cái gọi là “SOA trong một hộp” bởi vì để đạt được mục tiêu hướng dịch vụ kiến trúc công nghệ, định hướng dịch vụ cần được áp dụng thành công; cái này, trong quay lại, yêu cầu mọi thứ chúng tôi thiết kế và xây dựng phải được điều khiển theo hướng duy nhất, tâm nhín và yêu cầu của doanh nghiệp.

SOA có thể được hiện thực hóa thông qua nhiều công nghệ và tiêu chuẩn khác nhau.

Định hướng dịch vụ là một mô hình trung lập về công nghệ và trung lập với nhà cung cấp. Kiến trúc hướng dịch vụ là mô hình kiến trúc trung lập về công nghệ và trung lập với nhà cung cấp.

Điện toán hướng dịch vụ có thể được xem như một dạng đặc biệt của điện toán phân tán. Do đó, các giải pháp hướng dịch vụ có thể được xây dựng bằng cách sử dụng bất kỳ công nghệ và tiêu chuẩn ngành nào phù hợp cho điện toán phân tán.

Trong khi một số công nghệ (đặc biệt là những công nghệ dựa trên tiêu chuẩn ngành) có thể làm tăng tiềm năng áp dụng một số nguyên tắc thiết kế hướng dịch vụ, thì đó thực sự là tiềm năng đáp ứng các yêu cầu kinh doanh mà cuối cùng xác định sự lựa chọn phù hợp nhất về công nghệ và tiêu chuẩn ngành. Các mẫu thiết kế SOA, chẳng hạn như Dual

Giao thức [339] và Hợp đồng đồng thời [332], hỗ trợ việc sử dụng và tiêu chuẩn hóa công nghệ dịch vụ thay thế trong cùng một kho dịch vụ.

Thiết lập một bộ tiêu chuẩn và chính sách thống nhất của doanh nghiệp dựa trên ngành, thực tế và Tiêu chuẩn cộng đồng.

Các tiêu chuẩn ngành thể hiện các đặc tả công nghệ không độc quyền giúp thiết lập, trong số những thứ khác, các đặc điểm cơ bản nhất quán (như vận chuyển, giao diện, định dạng thông báo, v.v.) của kiến trúc công nghệ. Tuy nhiên, việc sử dụng công nghiệp riêng các tiêu chuẩn không đảm bảo rằng các dịch vụ về bản chất sẽ có khả năng tương tác.

Để hai chương trình phần mềm hoàn toàn tương thích, các quy ước bổ sung (chẳng hạn như dữ liệu các mô hình và chính sách) cần được tuân thủ. Đây là lý do tại sao các doanh nghiệp CNTT phải thiết lập và thực thi các tiêu chuẩn thiết kế. Việc không tiêu chuẩn hóa và quản lý hợp lý việc tiêu chuẩn hóa các dịch vụ trong một lĩnh vực nhất định sẽ bắt đầu phá hủy cơ cấu của khả năng tương tác mà dựa vào đó việc hiện thực hóa nhiều lợi ích chiến lược.

Nguyên tắc hướng dẫn này ủng hộ việc sử dụng các tiêu chuẩn thiết kế và nguyên tắc thiết kế của doanh nghiệp, chẳng hạn như Hợp đồng dịch vụ được tiêu chuẩn hóa (291) và Khớp nối lỏng lẻo của dịch vụ. (293). Nó cũng nhắc nhở chúng ta rằng, bất cứ khi nào có thể và khả thi, các tiêu chuẩn thiết kế tùy chỉnh phải dựa trên và kết hợp các tiêu chuẩn và nguyên tắc thiết kế hướng dịch vụ đã được ngành và cộng đồng nói chung sử dụng.

Theo đuổi sự đồng nhất ở bên ngoài đồng thời cho phép sự đa dạng ở bên trong.

Liên kết có thể được định nghĩa là sự hợp nhất của một tập hợp các thực thể khác nhau. Mặc dù cho phép mỗi thực thể được quản lý độc lập từ bên trong, nhưng tất cả đều đồng ý tuân thủ một mặt trận chung, thống nhất.

Một phần cơ bản của kiến trúc hướng dịch vụ là giới thiệu lớp điểm cuối liên kết giúp tóm tắt các chi tiết triển khai dịch vụ trong khi xuất bản một tập hợp các điểm cuối đại diện cho các dịch vụ riêng lẻ trong một miền nhất định theo cách thống nhất. Việc hoàn thành điều này thường liên quan đến việc đạt được sự thống nhất dựa trên sự kết hợp giữa các tiêu chuẩn công nghiệp và thiết kế. Tính nhất quán của sự thống nhất này giữa các dịch vụ là chìa khóa để hiện thực hóa khả năng tương tác nội tại, vì nó thể hiện mục đích chính và trách nhiệm của nguyên tắc thiết kế Hợp đồng dịch vụ được tiêu chuẩn hóa (291).

Lớp điểm cuối liên kết còn giúp tăng cơ hội khám phá các tùy chọn đa dạng của nhà cung cấp. Ví dụ: một dịch vụ có thể cần được xây dựng hoàn toàn dựa trên

nền tảng khác với nền tảng khác. Miễn là các dịch vụ này duy trì các điểm cuối tương thích, việc quản lý việc triển khai tương ứng của chúng có thể vẫn độc lập.

Điều này không chỉ nhấn mạnh rằng các dịch vụ có thể được xây dựng bằng cách sử dụng các phương tiện triển khai khác nhau (như EJB, .NET, SOAP, REST, v.v.), mà còn nhấn mạnh rằng các nền tảng và công nghệ trung gian khác nhau có thể được sử dụng cùng nhau, nếu cần.

Lưu ý rằng loại đa dạng này đi kèm với một mức giá. Nguyên tắc này không ủng hộ việc đa dạng hóa - nó chỉ khuyến nghị rằng chúng ta nên cho phép đa dạng hóa khi phù hợp, để các công nghệ và nền tảng "tốt nhất" có thể được tận dụng để tối đa hóa việc đáp ứng các yêu cầu kinh doanh.

Xác định các dịch vụ thông qua hợp tác với các bên liên quan về kinh doanh và công nghệ.

Để giải pháp công nghệ hướng tới doanh nghiệp, công nghệ phải đồng bộ với doanh nghiệp. Vì vậy, một mục tiêu khác của tính toán hướng dịch vụ là sắp xếp công nghệ và kinh doanh thông qua việc áp dụng định hướng dịch vụ. Giai đoạn mà sự liên kết này được hoàn thành ban đầu là trong quá trình phân tích và mô hình hóa thường đi trước việc phát triển và cung cấp dịch vụ thực tế.

Yếu tố quan trọng để thực hiện phân tích theo định hướng dịch vụ là phải có cả chuyên gia kinh doanh và công nghệ cùng làm việc để xác định và xác định các dịch vụ ứng viên. Ví dụ: các chuyên gia kinh doanh có thể giúp xác định chính xác bối cảnh chức năng liên quan đến các dịch vụ lấy doanh nghiệp làm trung tâm, trong khi các chuyên gia công nghệ có thể cung cấp đầu vào thực tế để đảm bảo rằng mức độ chi tiết và định nghĩa của các dịch vụ khái niệm vẫn mang tính thực tế so với môi trường triển khai cuối cùng của chúng.

Tối đa hóa việc sử dụng dịch vụ bằng cách xem xét phạm vi sử dụng hiện tại và tương lai.

Phạm vi của một dự án SOA nhất định có thể là toàn doanh nghiệp hoặc có thể bị giới hạn ở một miền của doanh nghiệp. Bất kể phạm vi nào, một ranh giới được xác định trước sẽ được thiết lập để bao gồm một danh sách các dịch vụ cần được mô hình hóa về mặt khái niệm trước khi chúng có thể được phát triển. Bằng cách lập mô hình nhiều dịch vụ có liên quan với nhau, về cơ bản chúng tôi thiết lập kế hoạch chi tiết về các dịch vụ mà cuối cùng chúng ta sẽ xây dựng. Bài tập này rất quan trọng khi cố gắng xác định và xác định các dịch vụ có thể được chia sẻ bởi các giải pháp khác nhau.

Có nhiều phương pháp và cách tiếp cận khác nhau có thể được sử dụng để thực hiện các giai đoạn phân tích hướng dịch vụ. Tuy nhiên, một chủ đề chung giữa tất cả chúng là ranh giới chức năng của các dịch vụ được chuẩn hóa để tránh sự dư thừa. Thậm chí sau đó,

các dịch vụ được chuẩn hóa không nhất thiết tạo ra các dịch vụ có khả năng tái sử dụng cao. Các yếu tố khác phát huy tác dụng, chẳng hạn như mức độ chi tiết của dịch vụ, quyền tự chủ, quản lý trạng thái, khả năng mở rộng, khả năng kết hợp và mức độ logic dịch vụ đủ chung để có thể được tái sử dụng một cách hiệu quả.

Những loại côn nhắc này, được hướng dẫn bởi chuyên môn về kinh doanh và công nghệ, mang lại cơ hội xác định các dịch vụ đáp ứng các yêu cầu sử dụng hiện tại đồng thời có tính linh hoạt để thích ứng với sự thay đổi trong tương lai.

Xác minh rằng các dịch vụ đáp ứng các yêu cầu và mục tiêu kinh doanh.

Như với bất cứ điều gì, dịch vụ có thể bị lạm dụng. Khi phát triển và quản lý danh mục đầu tư dịch vụ, việc sử dụng và hiệu quả của chúng trong việc đáp ứng các yêu cầu kinh doanh cần phải được xác minh và đo lường. Các công cụ hiện đại cung cấp nhiều phương tiện giám sát việc sử dụng dịch vụ, nhưng có những yếu tố vô hình cũng cần được xem xét để đảm bảo rằng các dịch vụ không chỉ được sử dụng vì chúng sẵn có mà còn để xác minh rằng chúng thực sự đáp ứng nhu cầu kinh doanh và đáp ứng mong đợi.

Điều này đặc biệt đúng với các dịch vụ dùng chung có nhiều sự phụ thuộc. Các dịch vụ chia sẻ không chỉ yêu cầu cơ sở hạ tầng dày dặn để đảm bảo khả năng mở rộng và độ tin cậy cho tất cả các giải pháp tái sử dụng chúng, mà chúng còn cần được thiết kế và được mở rộng hết sức cẩn thận để đảm bảo bối cảnh chức năng của chúng không bao giờ bị sai lệch.

Phát triển các dịch vụ và tổ chức của họ để đáp ứng nhu cầu sử dụng thực tế.

Nguyên tắc chỉ đạo này liên quan trực tiếp đến "Sự cải tiến mang tính tiên hóa trong việc theo đuổi tuyên bố giá trị hoàn hảo ban đầu", cũng như mục tiêu tổng thể là duy trì sự liên kết giữa kinh doanh và công nghệ.

Chúng ta không bao giờ có thể mong đợi phải dựa vào phỏng đoán khi xác định mức độ chi tiết của dịch vụ, phạm vi chức năng mà dịch vụ cần thực hiện hoặc cách thức dịch vụ sẽ cần. Để được tổ chức thành các tác phẩm. Dựa trên bất kỳ mức độ phân tích nào mà chúng tôi có thể thực hiện ban đầu, một dịch vụ nhất định sẽ được chỉ định một bối cảnh chức năng xác định và sẽ chứa một hoặc nhiều khả năng chức năng có khả năng liên quan đến nó trong một hoặc nhiều thành phần dịch vụ.

Khi các yêu cầu và hoàn cảnh kinh doanh trong thế giới thực thay đổi, dịch vụ có thể cần được tăng cường, mở rộng, tái cấu trúc hoặc thậm chí có thể được thay thế. Định hướng dịch vụ nguyên tắc thiết kế xây dựng tính linh hoạt tự nhiên vào kiến trúc dịch vụ sao cho, với tư cách là phần mềm,

các chương trình, dịch vụ có tính linh hoạt và thích ứng với sự thay đổi và bị thay đổi trong đáp ứng với việc sử dụng trong thế giới thực.

Tách biệt các khía cạnh khác nhau của một hệ thống thay đổi với tốc độ khác nhau.

Điều làm cho các hệ thống nguyên khôi và dựa trên silo trở nên linh hoạt là sự thay đổi có thể có tác động kép đến việc sử dụng hiện tại của chúng. Đây là lý do tại sao việc tạo các ứng dụng dựa trên silo mới thường dễ dàng hơn thay vì tăng cường hoặc mở rộng các ứng dụng hiện có.

Cơ sở lý luận đằng sau lý thuyết phân tách mối quan tâm là một vấn đề lớn hơn có thể được giải quyết hiệu quả hơn khi được phân tách thành một tập hợp các vấn đề hoặc mối quan tâm nhỏ hơn.

Khi áp dụng định hướng dịch vụ để phân tách các mối quan tâm, chúng tôi xây dựng các đơn vị logic giải pháp tương ứng để giải quyết các mối quan tâm riêng lẻ, từ đó cho phép chúng tôi tổng hợp các đơn vị để giải quyết vấn đề lớn hơn ngoài việc cho chúng tôi cơ hội tổng hợp chúng thành cấu hình khác nhau để giải quyết các vấn đề khác.

Bên cạnh việc thúc đẩy khả năng sử dụng lại dịch vụ, phương pháp này còn giới thiệu nhiều lớp sự trừu tượng hóa giúp bảo vệ các hệ thống bao gồm dịch vụ khỏi tác động của sự thay đổi. Hình thức trừu tượng này có thể tồn tại ở nhiều cấp độ khác nhau. Ví dụ: nếu cần thay thế các tài nguyên cũ được đóng gói bởi một dịch vụ thì tác động của thay đổi đó có thể được giảm thiểu như miễn là dịch vụ có thể giữ lại điểm cuối và hoạt động chức năng ban đầu của nó.

Một ví dụ khác là sự tách biệt logic bắt khai tri khỏi logic phi bắt khai tri. Loại logic trước đây có khả năng tái sử dụng cao nếu nó đa mục đích và ít có khả năng thay đổi. Mặt khác, logic bắt khai tri thường đại diện cho các phần có mục đích duy nhất của logic quy trình kinh doanh gốc, thường dễ biến động hơn. Tách những thứ tương ứng

các loại logic thành các lớp dịch vụ khác nhau còn giới thiệu thêm tính trữ lượng cho phép sử dụng lại dịch vụ trong khi bảo vệ các dịch vụ và bất kỳ giải pháp nào sử dụng chúng, từ tác động của sự thay đổi.

Giảm các phần phụ thuộc tiềm ẩn và xuất bản tất cả các phần phụ thuộc bên ngoài để tăng cường độ mạnh mẽ và giảm tác động của thay đổi.

Nguyên tắc hướng dẫn này thể hiện mục đích của nguyên tắc thiết kế Khớp nối lồng dịch vụ (293). Cách kiến trúc dịch vụ được cấu trúc nội bộ và cách các dịch vụ liên quan đến các chương trình sử dụng chúng (có thể bao gồm các dịch vụ khác) đều phụ thuộc vào sự phụ thuộc được hình thành trên các bộ phận chuyển động riêng lẻ là một phần của kiến trúc dịch vụ.

Các lớp trừu tượng giúp giảm bớt sự thay đổi mang tính tiền hóa bằng cách khoanh vùng các tác động của sự thay đổi đối với các vùng được kiểm soát. Ví dụ: trong kiến trúc dịch vụ, mặt tiền dịch vụ có thể được sử dụng cho các phần trừu tượng của quá trình triển khai nhằm giảm thiểu phạm vi tiếp cận của các phần phụ thuộc triển khai.

Mặt khác, các hợp đồng dịch vụ kỹ thuật được công bố cần phải bộc lộ những sự phụ thuộc mà người tiêu dùng dịch vụ phải hình thành để tương tác với dịch vụ. Theo nguyên tắc Trừu tượng dịch vụ (294), việc giảm các phụ thuộc nội bộ có thể

ánh hưởng đến các hợp đồng kỹ thuật này khi thay đổi xảy ra sẽ giảm thiểu sự gia tăng tác động của những thay đổi đó đối với người tiêu dùng dịch vụ phụ thuộc.

Ở mọi mức độ trừu tượng, hãy tổ chức từng dịch vụ xung quanh một đơn vị gắn kết và có thể quản lý được. chức năng.

Mỗi dịch vụ yêu cầu một bối cảnh chức năng được xác định rõ ràng để xác định logic nào thuộc và không thuộc về ranh giới chức năng của dịch vụ. Xác định phạm vi và mức độ chi tiết của các ranh giới dịch vụ chức năng này là một trong những trách nhiệm quan trọng nhất trong vòng đời cung cấp dịch vụ.

Các dịch vụ có mức độ chi tiết chức năng thấp có thể quá kém linh hoạt để có hiệu quả, đặc biệt nếu chúng được kỳ vọng có thể tái sử dụng được. Mặt khác, các dịch vụ quá chi tiết có thể đánh thuế cơ sở hạ tầng trong đó các thành phần dịch vụ sẽ cần bao gồm số lượng thành viên hợp thành ngày càng tăng.

Việc xác định sự cân bằng phù hợp giữa phạm vi chức năng và mức độ chi tiết đòi hỏi sự kết hợp giữa chuyên môn kinh doanh và công nghệ, đồng thời đòi hỏi sự hiểu biết về các dịch vụ trong một ranh giới nhất định có liên quan với nhau như thế nào.

Nhiều nguyên tắc hướng dẫn được mô tả trong tuyên ngôn này giúp đưa ra quyết định này trong việc hỗ trợ định vị từng dịch vụ như một tài sản CNTT có khả năng thúc đẩy doanh nghiệp CNTT hướng tới trạng thái mục tiêu, nhờ đó các lợi ích chiến lược của điện toán hướng dịch vụ được hiện thực hóa. .

Tuy nhiên, cuối cùng thì chính việc đạt được giá trị kinh doanh trong thế giới thực mới là điều quyết định, từ quan niệm đến việc cung cấp cho việc sử dụng lặp đi lặp lại, con đường phát triển của bất kỳ đơn vị chức năng hướng dịch vụ nào.

Giới thiệu về tác giả

Thomas Erl

Thomas Erl là tác giả CNTT có sách bán chạy nhất, người sáng lập Arcitura Education và là biên tập viên của tạp chí Chuỗi Công nghệ Dịch vụ Prentice Hall của Thomas Erl. Với hơn 300.000 bản in trên toàn thế giới, sách của ông đã trở thành sách bán chạy nhất trên toàn thế giới và được chính thức xác nhận bởi các thành viên cấp cao của các tổ chức CNTT lớn như IBM, Micro-soft, Oracle, Intel, Accenture, IEEE, HL7, MITER, SAP, CISCO, HP và nhiều tổ chức khác. Với tư cách là Giám đốc điều hành của Arcitura Education Inc., Thomas đã lãnh đạo việc phát triển chương trình giảng dạy cho Chứng chỉ Khoa học Dữ liệu Lớn (BDSCP), Đám mây được quốc tế công nhận Chứng nhận Chuyên gia được Chứng nhận (CCP) và Chứng nhận Chuyên nghiệp của SOA (SOACP) program, đã thiết lập một loạt các chứng chỉ ngành chính thức, trung lập với nhà cung cấp, được hàng nghìn chuyên gia CNTT trên khắp thế giới nhận được. Thomas đã đi lưu diễn hơn 20 quốc gia với tư cách là diễn giả và người hướng dẫn. Hơn 100 bài báo và bài phỏng vấn của Thomas đã được đăng trên nhiều ấn phẩm, trong đó có The Wall Tạp chí đường phố và Tạp chí CIO.

Mục lục

- M**
- đại lý. Xem tinh linh hoạt của các tác nhân dịch vụ (tổ chức), 50-52 danh mục quy trình kinh doanh bắt khả tri, 115 được xác định, 114 mẫu thiết kế Khả năng bắt khả tri, 133, 322 giai đoạn khả năng bắt khả tri (các lớp dịch vụ), 119 Mẫu thiết kế bối cảnh bắt khả tri, 133, 323 bối cảnh bắt khả tri giai đoạn (các lớp dịch vụ), logic bắt khả tri 117-118, 23 Tuyên bố SOA được chú thích, các dịch vụ ứng dụng 367-382. Xem các ứng dụng dịch vụ tiện ích, dưới dạng thành phần dịch vụ, kiến trúc 38-43
- các mẫu thiết kế và, 70 kiến trúc dịch vụ, 70-76 kiến trúc thành phần dịch vụ, 70, 77-83 kiến trúc kiểm kê dịch vụ, 70, 83-85 kiến trúc doanh nghiệp hướng dịch vụ, 70, 85-86 Phương thức phức tạp không đồng bộ, 247, 254-256 Mẫu thiết kế giao dịch dịch vụ nguyên tử, 198, 324
- att ribute giá trị cho các thông báo SOAP, 216 hệ thống tự động hóa, nhận dạng, 99
- B**
- khả năng tương thích ngược, 267-270 chiến lược tạo phiên bản linh hoạt, chiến lược tạo phiên bản lồng lèo 283-284, phạm vi cân bằng 284 (trụ cột định hướng dịch vụ), 55-58, 97
- BDSCP (Chứng chỉ khoa học dữ liệu lớn chuyên nghiệp), 11 lợi ích của định hướng dịch vụ, 43
- Tên miền kinh doanh và công nghệ tăng Căn chỉnh, 48-49
- Liên đoàn tăng lên, 46 Tăng khả năng tương tác nội tại, 44-45 Tăng tinh linh hoạt của tổ chức, 50-52 ROI tăng, 48-50 Tăng cường các lựa chọn đa dạng hóa nhà cung cấp, 47-48 Giảm gánh nặng CNTT, 52-53
- Chứng chỉ chuyên nghiệp về khoa học dữ liệu lớn (BDSCP), 11 bản thiết kế. Xem sách kế hoạch kiểm kê dịch vụ
- được ánh xạ tới các chủ đề từ án bản đầu tiên, tổ chức 4-6, chiến lược phân phối dự án bott om-up 6-8, cộng đồng doanh nghiệp 91-92, mối quan hệ với cộng đồng CNTT, 86-90 định hướng kinh doanh (đặc điểm SOA), 61-63 mô hình kinh doanh, sự liên kết công nghệ với, 48-49
- phân rã quy trình nghiệp vụ, 115-124, 142, 164 hành động lọc, 144, 165 xác định logic bắt khả tri, 149, 169 xác định tài nguyên, 170-171 yêu cầu nghiệp vụ trong phân tích hướng dịch vụ, 99
- C**
- Ràng buộc bộ đệm, 186 hồ sơ, 310
- Mẫu thiết kế biểu thức Canonical ein, 209, 325 Mẫu thiết kế lược đồ Canonical 194, 222, 326 Mẫu thiết kế phiên bản Canonical ein, 281, 327 Mẫu thiết kế thành phần khả năng, 83, 134, 328 chi tiết của khả năng, 210 Mô hình thiết kế tái cấu trúc năng lực, nghiên cứu trường hợp 83, 134, 329
- Hiệp hội Đại học Trung Tây (MUA) phân tích yêu cầu xử lý, 177-178 áp dụng định hướng dịch vụ, 174 liên kết ứng viên năng lực dịch vụ với tài nguyên, 173-174 nền tảng, 15 phân rã quy trình nghiệp vụ, 164 phương pháp phức tạp, 259-262 xác định ứng viên dịch vụ thực thể, 167-169 xác định ứng viên dịch vụ vi mô, 181 xác định ứng viên dịch vụ tiện ích, 179-180

- cân nhắc thiết kế cho các hợp đồng dịch vụ REST, 226-230
- hành động lọc, 165 xác định logic bắt khai tri, 169-170 xác định tài nguyên, 171-172 xác định các ứng cử viên thành phần dịch vụ, 175-176
- Mô hình hóa dịch vụ REST, 162-163 sửa đổi các ứng cử viên thành phần dịch vụ, 182
- Transit Line Systems, Inc. (TLS)**
- phân tích yêu cầu xử lý, 152 nền tảng, 14-15
 - phân rã quy trình
 - kinh doanh, 142-144 xác định ứng viên dịch vụ thực thể, 146-149 xác định ứng viên dịch vụ vi mô, 155 xác định ứng viên dịch vụ tiện ích, 154 cân nhắc thiết kế cho dịch vụ Web, lọc 198-208 hành động, 145 xác định logic bắt khai tri, 149-150 xác định các ứng cử viên thành phần dịch vụ, 151 tài liệu WSDL mô-đun, 214 không gian tên, 215-216 sửa đổi các ứng cử viên thành phần dịch vụ, 156 giá trị SOAP attribute, 217 Khả năng mở rộng dịch vụ Web, 213 mức độ chi tiết của dịch vụ Web, 212 Mô hình hóa dịch vụ web, 141 CCP (Cloud Certified Professional), 10 ràng buộc Client-Server, hồ sơ, 307 Cloud Certified Professional (CCP), 10 điện toán đám mây, tài nguyên thông tin, 60 Điện toán đám mây: Khái niệm, Công nghệ & Kiến trúc, 60 Các mẫu thiết kế điện toán đám mây, 60 mức độ chi tiết thô, 211 phiên bản và, ràng buộc mã theo yêu cầu 266-267, hồ sơ, khả năng tương thích 315. Xem thêm phiên bản

Các cân nhắc về dịch vụ REST, lập phiên bản 276-279 và, khả năng tương thích ngược 267, các thay đổi tương thích 267-270, khả năng tương thích chuyên tiếp 273-275, các thay đổi không tương thích 271-273, 275-276 đám bảo khả năng tương thích, 280

những thay đổi tương thích, 273-275

Mẫu thiết kế giao dịch dịch vụ đền bù, 198, 330

nghiên cứu diễn hình về các phương pháp phức tạp, 259-262 thiết kế, 246-249 phương pháp có trạng thái, 256-258 phương pháp không có trạng thái, 249-256 thành phần. Xem kiến trúc thành phần dịch vụ. Xem thành phần dịch vụ ngành kiến trúc

Mẫu thiết kế tự chủ thành phần, 224, 331 lấy thành phần làm trung tâm, 68-69, 124 bộ điều khiển thành phần, 78

Mẫu thiết kế Hợp đồng đồng thời, 193, 195, 212, 221, 223, 332 độ chi tiết ràng buộc, phiên bản 210 và 266-267

ràng buộc (REST). Xem thêm các hạn chế về thiết kế Bộ đệm, 186, 310 Máy khách-Máy chủ, 307 Mã theo yêu cầu, 315 Hệ thống phân lớp, định dạng bảng hồ sơ 187.313-314, 306 Không quốc tịch, 186, 249, 256-257, 308-309 Hợp đồng thống nhất, 183, 187, 311-312 mô hình hợp đồng thống nhất và 186-187

Mẫu thiết kế container hóa, 333 Mẫu thiết kế đàm phán nội dung, 244-245, 334 SOA đương đại. Xem mẫu thiết kế phi chuẩn hóa hợp đồng SOA, hợp đồng 212, 335. Xem hợp đồng dịch vụ

Mẫu thiết kế lớp tiện ích tên miền chéo, 195, 336

D

độ chi tiết của dữ liệu, 210 phân rã các quy trình kinh doanh, 142, 164 giai đoạn phân rã (các lớp dịch vụ), 115-124

Mẫu thiết kế hợp đồng tách rời, 193, 337 chiến lược phân phối cho các dự án SOA, 91-92

Phương pháp phức hợp Delta, 247, 252-254 phụ thuộc, lập phiên bản và 264 giai đoạn triển khai (dự án SOA), 105

- cân nhắc thiết kế cho hợp đồng dịch vụ REST theo mô hình
 dịch vụ, nghiên cứu trường hợp 221-225, hướng dẫn 226-230, 231-258 cho hợp đồng thống nhất, 231
- Thiết kế phương pháp phức hợp HTT P, 246-249
- Thiết kế tiêu đề HTT P, 233-235
- Thiết kế phương pháp HTT P, 231-233
- Tùy chỉnh mã phản hồi HTT P, 240-241
- Thiết kế mã phản hồi HTT P, 235-236, 239-240 loại phương tiện, thiết kế lược đồ 242-244, 244-245
- phương thức phức tạp có trạng thái, 256-258 phương thức phức tạp không trạng thái, 249-256 cho hợp đồng dịch vụ Web
- theo mô hình dịch vụ, nghiên cứu trường hợp 193-198, hướng dẫn 198-208, 208-216
- các ràng buộc thiết kế, các quy ước cho hồ sơ, 8-9 mô hình thiết kế, 24-25 ngôn ngữ mẫu thiết kế. Xem các ưu điểm của mẫu thiết kế ngôn ngữ mẫu,
 318-319 Khả năng bắt khẩ tri, 322 Bối cảnh bắt khẩ tri, 323 kiến trúc và, 70 Giao dịch dịch vụ
 nguyên tử, 198, 324 Biểu thức Canonical, 209, 325 Lược đồ Canonical, 194, 222, 326 Phiên bản Canonical, 281, 327 Thành phần khà năng, 83, 134, 328 Tái hợp khà năng, 83, 134, 329 Giao dịch dịch vụ bù, 198, 330 Tự chủ thành phần, 224, 331 Hợp đồng đồng thời, 193, 195, 212, 221, 223, 332 Containerization , 333 Đàm phán nội dung, 244, 245, 334 Không chuẩn hóa
 hợp đồng, 212, 335 Lớp tiện ích tên miền chéo, 195, 336 Hợp đồng tách rời, 193, 337 được xác định, 318-319 Kiểm kê tên miền, 57, 83, 97, 187, 195, 338 Giao thức kép, 155, 193, 195, 212, 221, 223, 339
- Hàng tồn kho doanh nghiệp, 57, 83, 187, 340
 Trữ tương thực thể, 133, 341
- Liên kết thực thể, 222, 342
- Nhắn tin theo hướng sự kiện, 258, 343
- Phản hủy chức năng, 133, 344
- Năng Lực Bình Thường, 252, 345
- Điểm cuối hàng tồn kho, 86, 346
- Gói kế thừa, 195, 223, 347
- Tập trung logic, 135, 166, 348
- Triển khai vi dịch vụ, 349
- Trữ tương hóa nhiệm vụ vi mô, 134, 350
- Bối cảnh phi thuyết bát khẩ tri, 133, 351
- Trì hoãn một phần của tiểu bang, 198, 352
- Trữ tương hóa quy trình, 134, 353 hò sơ, quy ước cho, định dạng bảng hò sơ 8-9, 321
- Thực hiện dự phòng, 224, 354
- Hợp đồng tái sử dụng, 233, 355
- Tập trung lược đồ, 194, 222, 277, 356
- Đại Lý Dịch Vụ, 76, 357
- Sao chép dữ liệu dịch vụ, 224, 358
- Đóng gói dịch vụ, 67, 133, 359
- Mặt tiền dịch vụ, 193, 195, 221, 223, 360
- Chuẩn hóa dịch vụ, 135, 166, 361
- Tin nhắn Nhà nước, 198, 362
- Kho lưu trữ nhà nước, 198, 363
- cách sử dụng trong sách, 3-4
- Trữ tương tiện ích, 133, 364
- Trữ tương xác thực, 214, 245, 365
- Nhận dạng phiên bản, 279, 366 nguyên tắc thiết kế, 60-61 danh sách, 26, 29 hồ sơ, quy ước, định dạng bảng hồ sơ 8-9, 290
- Trữ tương hóa dịch vụ, 27, 73, 80, 150, 223, 248 khả năng tương tác, 45 hồ sơ, 294
- Dịch vụ tự chủ, 27, 73, 150, 174, 194 khả năng tương tác, 45 hồ sơ, 297
- Khả năng tương tác dịch vụ, 29, 68, 103, 127, 213 khả năng tương tác, hồ sơ 45, 302-303
- Khả năng khám phá dịch vụ, 28, 106 khả năng tương tác, 45 hồ sơ, 300-301

- Khớp nối lồng, 26, 150, 223
 khả năng tương tác,
 45 hò sơ, 293
 Khả năng tái sử dụng dịch vụ, 27, 194, 195, 213
 khả năng tương tác,
 45 hò sơ, 295-296
 Dịch vụ Không quốc tịch, 27, 73, 198
 khả năng tương tác,
 45 hò sơ, 298-299
 Hợp đồng dịch vụ được tiêu chuẩn hóa, 26, 103, 223-224
 khả năng tương tác,
 45 hò sơ, 291-292
 ưu tiên thiết kế, 69 kỹ
 luật (trụ cột định hướng dịch vụ), 55 giai đoạn
 khám phá (dự án SOA), 106 giá trị thuộc
 tính tài liệu cho các thông báo SOAP, 216
 Mẫu thiết kế kho miền, 57, 83, 97, 187,
 195, 338
 kho dịch vụ tên miền, 25
 Mẫu thiết kế Giao thức kép, 155, 193, 195, 212, 221, 223,
 339
- Giáo dục điện tử (trụ cột định hướng dịch vụ),
 55 lấy doanh nghiệp làm trung tâm (đặc điểm SOA), 66-67
 Mẫu thiết kế kho hàng doanh nghiệp 57, 83, 187, 340
- nguồn lực doanh nghiệp, 66
 thực thể, nguồn lực so với, 189
 Mẫu thiết kế trừu tượng thực thể, 133, 341 Giai
 đoạn trừu tượng hóa thực thể (lớp dịch vụ), 121
 Mẫu thiết kế Liên kết thực thể, 222, 342 ứng
 viên dịch vụ thực thể liên
 kết với tài nguyên, 172 định nghĩa,
 146, 166 dịch vụ
 thực thể được
 xác định, 113
 cân nhắc thiết kế cho
 hợp đồng dịch vụ REST, 221-222 cho dịch
 vụ Web, 193-194 lỗi, 9, 11
- Mẫu thiết kế nhẫn tin hướng sự kiện, 258, 343 khả năng mở
 rộng của dịch vụ Web, 212-213
- Liên đoàn F , Mục tiêu/lợi ích gia tăng của Liên đoàn, 46
 Tìm nạp phương pháp phức tạp, các hình 247,
 249-250, chú giải ký hiệu, 9
 mức độ chi tiết nhỏ, 211 phiên
 bản và, chiến lược tạo
 phiên bản linh hoạt 266-267, khả năng tương
 thích chuyển tiếp 283-285, chiến lược
 phiên bản lồng léo 271-273, phân
 rã chức năng 284 , 116
 Mẫu thiết kế phân rã chức năng, 133, 344 Giai đoạn
 phân hủy
 chức năng (lớp dịch vụ), 115
- G**
- mục tiêu định hướng dịch vụ, 43
 Tên miền kinh doanh và công nghệ tăng
 Căn chỉnh, 48-49
 Liên đoàn tăng lên, 46
 Tăng khả năng tương tác nội tại, 44-45
 Tăng tính linh hoạt của tổ chức, 50-52
 ROI tăng, 48-50
 Tăng cường các lựa chọn đa dạng hóa nhà cung cấp, 47-48
 Giảm gánh nặng CNTT, mức độ
 chi tiết ràng
 buộc về độ chi tiết 52-53, lập phiên bản và 266-267
 Mô hình hóa dịch vụ REST, 188
 dịch vụ Web, 210-212
- H**
- HTML, các thay đổi tương thích, 278-279
 Tiêu đề HTT P, thiết kế và tiêu chuẩn hóa, 233-235
- Các loại phương tiện HTT P
 thiết kế, thiết kế
 lực độ 242-244, 244-245
 Phương pháp HTT P
 thiết kế phương pháp phức tạp, 246-249
 nghiên cứu trường hợp các phương pháp phức
 tạp, 259-262 thiết kế và tiêu chuẩn hóa,
 231-233 phương pháp phức tạp có trạng thái,
 256-258 phương pháp phức tạp không trạng thái, 249-256
- Mã phản hồi HTT P
 tùy biến, thiết kế và tiêu
 chuẩn hóa 240-241, 235-236, 239-240

I Mẫu thiết kế Khả năng Idempotent, 252, 345 thay đổi không tương thích, 275-276 Khả năng tương tác nội tại tăng lên, tích hợp 44-45 trong định hướng dịch vụ, khả năng tương tác 40-42, kiến trúc kho lưu trữ 37-38, 44-45. Xem kiến trúc kho dịch vụ

Mẫu thiết kế điểm cuối hàng tồn kho, 86, 346 Cộng đồng CNTT, mối quan hệ với cộng đồng doanh nghiệp, 86-90

L

Ràng buộc hệ thống lớp, 187
hồ sơ, 313-314

Mẫu thiết kế Legacy Wrapper, 195, 223, 347 giá trị thuộc tính theo nghĩa đen cho các thông báo SOAP, 216 tập trung logic, 134
Mô hình thiết kế tập trung logic, chiến lược phiên bản lồng leo 135, 166, 348, 284-285

Giai đoạn bảo trì M (dự án SOA), thiết kế 105
loại phương tiện, thiết kế lược đồ 242-244, 244-245 loại phương tiện hợp đồng thống nhất, khả năng tương thích, 277-279

thông điệp (SOAP), giá trị thuộc tính, 216
phương pháp cho các dự án SOA, 91-92 phương pháp (HTTP) thiết kế

kế phương pháp phức tạp, 246-249 nghiên cứu trường hợp các phương pháp phức tạp, 259-262 thiết kế và tiêu chuẩn hóa, 231-233 phương pháp phức tạp có trạng thái, 256 -258 phương pháp phức tạp không trạng thái, 249-256 ứng viên microservice, định nghĩa, 154, 180 giai đoạn ứng viên microservice (lớp dịch vụ), 123
Mô hình thiết kế triển khai microservice, 349 microservices

được xác định, 113 cân nhắc thiết kế cho các hợp đồng dịch vụ REST, 223-224 cho các dịch vụ Web, 196 thành phần khả năng dịch vụ và, 130-131

Mô hình thiết kế trừu tượng hóa nhiệm vụ vi mô, 134, 350
giai đoạn trừu tượng hóa nhiệm vụ vi mô (các lớp dịch vụ), nghiên cứu điển hình 123 của Hiệp hội Đại học Trung Tây.

Xem các nghiên cứu trường hợp, tài liệu WSDL
mô-đun của Hiệp hội Đại học Trung Tây (MUA), 214 giai đoạn giám sát (dự án SOA), 105-106

Không gian tên N cho tài liệu WSDL, 215 tiêu chuẩn đặt tên cho dịch vụ Web, 208-209
SOA thẻ hệ tiếp theo: Giới thiệu ngắn gọn về

Công nghệ dịch vụ & Định hướng dịch vụ, 3 loại quy trình kinh doanh
bắt khái tri, 115 được xác định, 114

Mô hình thiết kế bối cảnh phi bắt khái tri, 133, 351 giai đoạn ngữ cảnh phi bắt khái tri (lớp dịch vụ), 122 logic phi bắt khái tri, 23 nhận dạng, 149, 169 trạng web dịch vụ thông báo, 11

O
các ngôn ngữ mô hình mở, 320 dịch vụ tác vụ được phối hợp, 114 tính linh hoạt của tổ chức, Tăng cường tính tổ chức
Mục tiêu/lợi ích linh hoạt, 50-52
vai trò tổ chức, các giai đoạn dự án SOA và 107-109

P

Mẫu thiết kế tri hoãn trạng thái một phần, 198, 352 ngôn ngữ mẫu, 320 mẫu. Xem các mô hình thiết kế trụ cột và định hướng dịch vụ, 54 phạm vi cân bằng,

55-58, 97 kỷ luật, 55 giờ
dục, 55 làm việc nhóm, 54
Chuỗi Công nghệ
Dịch vụ Prentice Hall từ Thomas Erl, 2, 4, 6, 290, 306, 321
phương pháp nguyên thủy, 247
nguyên tắc. Xem nguyên tắc thiết kế
Mẫu thiết kế trừu tượng hóa quy trình, 134, 353 Giai đoạn trừu tượng hóa quy trình (lớp dịch vụ), 123-124 hồ sơ, quy ước cho, 8-9

- các bảng hồ sơ. Xem *mẫu thiết kế*; *nguyên tắc thiết kế*; *nguyên tắc thiết kế*; *Ràng buộc REST*
- dự án. Xem các dự án SOA Phuong thức phức tạp PubSub, 257-258
- R**
- sự tái cấu trúc. Xem thành phần dịch vụ Giảm gánh nặng CNTT, 52-53 *Dự phòng*
- Mẫu thiết kế triển khai*, 224, 354
- tài nguyên
- liên kết các ứng viên có năng lực dịch vụ với, 172 *thực thể* so với, 189 *nhận dạng*, 170-171 để biết thông tin, 9 *sửa đổi* định nghĩa, 182-183 *mã phản hồi* (HTT P)
 - tùy biến, thiết kế và tiêu chuẩn hóa 240-241, 235-236, 239-240
- NGHÌ NGHÌ
- hạn chế
- Bộ đệm, 186, 310
 - Máy khách-Máy chủ, 307
 - Mã theo yêu cầu, 315
 - Hệ thống phân lớp, định dạng bảng hồ sơ 187, 313-314, 306
 - Không quốc tịch, 186, 249, 256-257, 308-309
 - Hợp đồng thống nhất, mô hình hợp đồng thống nhất 183, 187, 311-312 và mô hình kiểm kê dịch vụ 186-187, hợp đồng thống nhất mô hình hóa và, 183-186 mô hình hóa dịch vụ, 160-161 phân tích các yêu cầu xử lý, 176-177 áp dụng định hướng dịch vụ, 174, 181 liên kết các ứng viên có năng lực dịch vụ với tài nguyên, 172
 - phân rã quy trình kinh doanh, 164 ứng viên dịch vụ thực thể xác định, 166 ứng cử viên dịch vụ vì mô xác định, 180 ứng cử viên dịch vụ tiện ích xác định, 178 hành động lọc, 165 mức độ chi tiết, 188 xác định logic phi bắt khả thi, 169 tài nguyên xác định, 170-171 xác định ứng viên thành phần dịch vụ, 175 quy trình cho, 165
- nguồn lực so với các thực thể, 189 sửa đổi các nhóm ứng viên năng lực dịch vụ, 182-183
- sửa đổi thành phần ứng viên phục vụ, 181
- Dịch vụ REST, 21 khái niệm tương thích ngược, cân nhắc khả năng tương thích 268-270, khả năng tương thích chuyên tiếp 276-279, chuẩn hóa dịch vụ 271-273, lập phiên bản 135, trang web 266-286 để biết thông tin, 10
- Hợp đồng dịch vụ REST lợi ích của, 220 cân nhắc thiết kế theo mô hình dịch vụ, nghiên cứu trường hợp 221-225, 226-230 hướng dẫn cho, 231-236, 239-258 lợi tức đầu tư, Mục tiêu/lợi ích ROI tăng lên, 48, 50
- khả năng tái sử dụng của giải pháp logic, 35
- Mẫu thiết kế* hợp đồng tái sử dụng 233, 355
- ROI (lợi tức đầu tư), 48, 50
- S**
- Mẫu thiết kế* tập trung lược đồ, 194, 222, 277, 356
- lực đồ, thiết kế cho các loại phương tiện, 244-245
- phân tách mối quan tâm, 24-25
- Nguyên tắc thiết kế trữ lượng dịch vụ, 27, 73, 80, 150, 223, 248 khả năng tương tác, 45 hồ sơ, 294
- Mẫu thiết kế Service Agent*, 76, 357 service Agent, 76-77 service architecture, 70-76
- Nguyên tắc thiết kế Tự chủ dịch vụ, 27, 73, 150, 174, 194 khả năng tương tác, 45 hồ sơ, 297 ranh giới dịch vụ, 134 ứng viên dịch vụ, 115 khả năng dịch vụ, 76 ứng viên năng lực dịch vụ, 115 phân tích yêu cầu xử lý, 152, 176-177 liên kết với tài nguyên, 172 thành phần và sự tái chế, 127-133

- đã xác định,
 115 nhóm sửa đổi, 157, 182-183
 Nguyên tắc thiết kế Khả năng kết hợp dịch vụ, khả năng
 tương tác 29,
 68, 103, 127, 213, hồ
 sơ 45, thành phần
 dịch vụ 302-303
 các ứng dụng như, 38-43
 được xác định, 24,
 26, 77 ứng viên năng lực dịch vụ, 127-133 định
 hướng dịch vụ và, 124-127 ký hiệu, 24
 kiến trúc
 thành phần dịch vụ, 70, 77-83 ứng viên thành phần dịch
 vụ xác định, 151, 175 sửa đổi, 156, 181
 người tiêu dùng dịch vụ,
 23
- hợp đồng dịch vụ, 21, 74-75
 NGHĨ NGỜI
 lợi ích của, 220
 cân nhắc về thiết kế, 221-230 hướng
 dẫn thiết kế, 231-236, 239-258
 Dịch vụ web
 lợi ích của, 192
 cân nhắc về thiết kế, 193-208 hướng
 dẫn thiết kế, 208-216
 Mẫu thiết kế sao chép dữ liệu dịch vụ, 224, 358 triển khai và
 bảo trì dịch vụ, 105 phát triển dịch vụ, 103
- Nguyên tắc thiết kế Khả năng khám phá dịch vụ, 28, 106 Khả
 năng tương tác, 45 hồ
 sơ, khám phá dịch
 vụ 300-301, 106
 Mẫu thiết kế đóng gói dịch vụ, 67, 133, 359 Giai đoạn đóng gói
 dịch vụ (các lớp dịch vụ), 116-117
- Mẫu thiết kế mặt tiền dịch vụ, 193, 195, 221, 223, 360 chi
 tiết dịch
 vụ, 210 kho dịch vụ
- được xác định,
 25-26 ranh giới dịch vụ, 134
 ký hiệu, 25
 cân nhắc thiết kế hợp đồng thống nhất, 231
 Thiết kế phương pháp phức hợp HTT P, 246-249
 Thiết kế tiêu đề HTT P, 233-235
- Thiết kế phương pháp HTT P, 231-233
 Tùy chỉnh mã phản hồi HTT P, 240-241
 Thiết kế mã phản hồi HTT P, 235-236, 239-240 loại
 phương tiện, thiết kế
 lược đồ 242-244, 244-245
 phương thức phức tạp có trạng thái,
 256-258 phương thức phức tạp không trạng thái, 249-256
 phân tích kiểm kê dịch vụ, kiến trúc kiểm
 kê dịch vụ 96-97, kế hoạch chi tiết kiểm kê dịch
 vụ 70, 83-85, mô hình kiểm kê dịch vụ 84, 96-97
 (REST), mô hình hóa hợp đồng thống nhất và, giai đoạn
 phân rã lớp dịch vụ 183-186, được xác
 định 115-124 ,
 114 thiết kế logic dịch vụ, 103
- Nguyên tắc thiết kế Khớp nối lồng léo của dịch vụ, khả
 năng tương
 tác 26, 150, 223, 45
 hồ sơ, 293 mô
 hình hóa dịch vụ được
 xác định, 100
 bước quy trình nguyên thủy, 112
 Mô hình hóa dịch vụ REST, 160-161 phân
 tích các yêu cầu xử lý, 176-177 áp dụng định hướng
 dịch vụ, 174, 181 liên kết các ứng viên có
 năng lực dịch vụ với
 tài nguyên, 172
 phân rã quy trình kinh doanh, 164 ứng viên
 dịch vụ thực thể xác định, 166 ứng cử viên
 dịch vụ vì mô xác định, 180 ứng cử viên dịch
 vụ tiện ích xác định, 178 hành động lọc, 165
 mức độ chi tiết, 188 xác
 định logic phi bắt
 khả thi, 169 tài nguyên xác định, 170-171
 xác định ứng viên thành phần dịch
 vụ, 175 quy trình cho, 165 nguồn lực so với thực thể, 189
 sửa đổi nhóm ứng
 viên năng lực dịch vụ, 182-183
- sửa đổi thành phần ứng viên phục vụ, 181
 Dịch vụ web, 140
 phân tích yêu cầu xử lý, 152 ứng dụng định
 hướng dịch vụ, 150, 155

- phân xâ quy trình kinh doanh, 142 ứng cử viên dịch vụ thực thể xác định, 146 ứng cử viên dịch vụ vi mô xác định, 154 ứng cử viên dịch vụ tiện ích xác định, 153 hành động lọc, 144 xác định logic bất khả tri, 149 xác định ứng viên thành phần dịch vụ, 151 sửa đổi nhóm ứng viên khả năng dịch vụ, 157 sửa đổi các ứng cử viên thành phần dịch vụ, 156 mô hình dịch vụ được xác định, 113 cân nhắc thiết kế cho hợp đồng dịch vụ REST, 221-225 cho hợp đồng dịch vụ Web, danh sách 193-198, 113 chuẩn hóa dịch vụ, 134
- Mẫu thiết kế chuẩn hóa dịch vụ 135, 166, 361
- định hướng dịch vụ
- các ứng dụng trong, 38-43
 - áp dụng trong mô hình hóa dịch vụ, 150, 155, 174, 181
 - được xác định, 26 đặc điểm thiết kế của, 34-35
 - logic dành riêng cho ứng dụng, giảm thiểu, 36 khả năng tương tác, 37-38 logic giải pháp tổng thể, giảm bớt, 36 -37 logic giải pháp có thể tái sử dụng, 35 là mô hình thiết kế, 24-25
 - thành phần, 26 mục tiêu và lợi ích của, 43
 - Tên miền kinh doanh và công nghệ tăng Căn chỉnh, 48-49
 - Liên đoàn tăng lên, 46
 - Tăng khả năng tương tác nội tại, 44-45
 - Tăng tính linh hoạt của tổ chức, 50-52
 - ROI tăng, 48-50
 - Tăng cường các lựa chọn đa dạng hóa nhà cung cấp, 47-48
 - Giảm gánh nặng CNTT, 52-53
 - tích hợp và, 40-42 trụ cột, 54 phạm vi cân bằng, 55-58, 97 kỹ luật, 55 giáo dục, 55 làm việc nhóm, 54
- các vấn đề được giải quyết bằng, 29 kiến trúc phức tạp, 33 hiệu quả, thiêu, 32 doanh nghiệp phình to, 32-33 thách thức tích hợp, 34 kiến trú ứng dụng dựa trên silo, 29-31 lãng phí, 31-32 kết quả, 86-90 thành phần dịch vụ và, 124-127 nguyên tắc thiết kế phân tích hướng dịch vụ, 97-100. Xem SOA
- Kiến trúc hướng dịch vụ: Các khái niệm, Công nghệ và Thiết kế, 2-3 thiết kế hướng dịch vụ, 101-102 kiến trúc doanh nghiệp hướng dịch vụ, 70, 85-86 logic giải pháp hướng dịch vụ, 26 tài liệu hồ sơ dịch vụ, 76
- Dịch vụ Nguyên tắc thiết kế có thể tái sử dụng, 27, 194-195, 213 Khả năng tương tác, 45 hồ sơ, 295-296
- dịch vụ
- nhu cầu hợp các khả năng, 22-23 được xác định, 21, 26 được giải thích, 20-21
 - Dịch vụ REST, 21 ký hiệu cho, 21-22
 - Dịch vụ web, 21 hợp đồng dịch vụ, 21
- Nguyên tắc thiết kế không trạng thái của dịch vụ, khả năng tương tác 27, 73, 198, 45 hồ sơ, thử nghiệm dịch vụ 298-299, giám sát và sử dụng dịch vụ 103-104, lập phiên bản dịch vụ 105-106, kiến trúc ứng dụng dựa trên silo 106-107, 29-31
- Các đặc điểm của SOA (kiến trúc hướng dịch vụ), 61-69 lấy doanh nghiệp làm trung tâm, 61-63 lấy thành phần làm trung tâm, 68-69 lấy doanh nghiệp làm trung tâm, 66-67 lấy nhà cung cấp làm trung lập, 63-65 ưu tiên thiết kế, 69

- các loại, kiến trúc dịch vụ 70-71, kiến trúc 71-76, kiến trúc kiểm kê dịch vụ 77-83, kiến trúc doanh nghiệp hướng dịch vụ 83-85, 85-86
- Lập kế hoạch áp dụng SOA, 95
- SOACP (SOA được chứng nhận chuyên nghiệp), 10
- Mẫu thiết kế SOA, 3, 89, 320-321
- Quản trị SOA: Quản lý các dịch vụ chia sẻ Tại chỗ & trên đám mây, 3, 107
- Tuyên ngôn SOA
- phiên bản có chủ thích, ưu tiên
- thiết kế 367-382, 69
- mẫu SOA. Xem các mẫu thiết kế của các dịch vụ Web dựa trên SOAP. Xem các dịch vụ Web Thông báo SOAP, giá trị thuộc tính att, 216
- Nguyên tắc thiết kế dịch vụ SOA, 3, 290
- Các chiến lược và phương pháp phân phối dự án SOA, 91-92 giải đoạn, 94-95 vai trò tổ chức và, 107-109 triển khai và bảo trì dịch vụ, 105 phát triển dịch vụ, 103 khám phá dịch vụ, 106 phân tích kiểm kê dịch vụ, 96-97 thiết kế logic dịch vụ, 103 dịch vụ -phân tích theo định hướng, thiết kế theo định hướng dịch vụ 97-100, thử nghiệm dịch vụ 101-102, giám sát và sử dụng dịch vụ 103-104, lập phiên bản dịch vụ 105-106, 106-107
- Lập kế hoạch áp dụng SOA, 95
- SOA với REST: Nguyên tắc, Mô hình & Các ràng buộc đối với việc xây dựng giải pháp doanh nghiệp với REST, 3, 220, 306
- logic giải pháp logic dành riêng cho ứng dụng, giảm bớt, 36 logic tổng thể, giảm bớt, 36-37 khả năng sử dụng lại, 35
- Nguyên tắc thiết kế hợp đồng dịch vụ được tiêu chuẩn hóa, khả năng tương tác 26, 103, 223-224, 45 hồ sơ, 291-292
- phương pháp phức tạp có trạng thái, 256-258 phương pháp phức tạp không trạng thái, 249-256
- Ràng buộc không quốc tịch, hồ sơ 186, 249, 256-257, 308-309
- Mẫu thiết kế tin nhắn trạng thái, 198, 362
- Mẫu thiết kế Kho lưu trữ Nhà nước, 198, 363
- Lưu trữ phương thức phức tạp, 247, 250-251
- chiến lược tạo phiên bản nghiêm ngặt, 282-285 ưu điểm của ngôn ngữ mẫu có cấu trúc, 320 được xác định, 320 ký hiệu, chú giải 21-22, 9
- thành phần dịch vụ, 24 kho
- dịch vụ, 25
- T**
- dịch vụ nhiệm vụ đã xác định, 113 cân nhắc thiết kế cho các hợp đồng dịch vụ REST, 225 cho các dịch vụ Web, 196-198 giải đoạn dịch vụ nhiệm vụ (các lớp dịch vụ), 123-124 làm việc nhóm (trụ cột định hướng dịch vụ), 54 kién trúc công nghệ. Xem giải đoạn thử nghiệm kiến trúc (dự án SOA), chiến lược phân phối dự án từ trên xuống 103-104, phương pháp phức hợp 91-92
- Trans, nghiên cứu điển hình của 256 Transit Line Systems, Inc. Xem các nghiên cứu điển hình, Transit Line Systems, Inc. (TLS), 14
- bạn
- Ràng buộc hợp đồng thống nhất, 183, 187, 245 hồ sơ, 311-312 các loại phương tiện hợp đồng thống nhất, khả năng tương thích, 277-279
- mẫu hợp đồng thống nhất
- Các ràng buộc REST và, 186-187
- Mô hình kiểm kê dịch vụ REST và, 183-186 hợp đồng thống nhất, cân nhắc thiết kế, 231
- Thiết kế phương pháp phức hợp HTTP, 246-249
- Thiết kế tiêu đề HTTP, 233-235
- Thiết kế phương pháp HTTP, 231-233
- Tùy chỉnh mã phản hồi HTTP, 240-241
- Thiết kế mã phản hồi HTTP, 235-236, 239-240 loại phương tiện, thiết kế lược đồ 242-244, 244-245 phương thức
- phức tạp có trạng thái, 256-258 phương thức phức tạp không trạng thái, 249-256

cập nhật, 9

Mẫu thiết kế trừu tượng tiện ích, 133, 364 giai đoạn trừu tượng hóa tiện ích (lớp dịch vụ), 120 ứng viên dịch vụ tiện ích, xác định, 153, 178 dịch vụ tiện ích được xác định, 113 cân nhắc thiết kế cho hợp đồng dịch vụ REST, 222-223 cho dịch vụ Web, 194, 195

V.

Xác thực Mô hình thiết kế trừu tượng, 214, 245, 365 đa dạng hóa nhà cung cấp, Tăng nhà cung cấp
Đa dạng hóa các lựa chọn Mục tiêu/lợi ích, 47-48 trung lập với nhà cung cấp (đặc điểm SOA), mẫu thiết kế nhận dạng phiên bản 63-65, mã nhận dạng phiên bản 279, 366, phiên bản 279-281.

Xem thêm khả năng tương thích và khả

năng tương thích ngược

267, các thay đổi tương thích
267-270, khả năng tương thích
chuyển tiếp 273-275, các thay đổi không tương thích 271-273, độ chi tiết ràng buộc 275-276 và, các phụ thuộc 266-267 và, 264 dịch vụ REST, 266, 286 chiến lược, 282 so sánh, 285 chiến lược linh hoạt, 283-284 chiến lược lỏng leo, 284 chiến lược nghiêm ngặt, 282-283 số nhận dạng phiên bản, 279-281 Dịch vụ web, 265-266 giai đoạn tạo phiên bản (dự án SOA), 106-107

W

Thiết kế và tạo phiên bản hợp đồng dịch vụ web hoặc

SOA, 192, 245

Hợp đồng dịch vụ web

lợi ích của, 192

nghiên cứu trường hợp

cân nhắc thiết kế,

198-208 hướng dẫn,

208-216 theo mô hình dịch vụ, 193-198

Dịch vụ web

khả năng tương thích ngược, khả năng mở rộng 267-268, khả năng tương thích chuyển tiếp 212-213, 271 độ chi tiết, 210-212 tiêu chuẩn đặt tên, mô hình hóa dịch vụ 208-209, 140 phân tích yêu cầu xử lý, 152 áp dụng định hướng dịch vụ, 150, 155 phân rã quy trình nghiệp vụ, 142 định nghĩa ứng viên dịch vụ thực thể, 146 ứng viên xác định dịch vụ vi mô, 154 ứng viên xác định dịch vụ tiện ích, 153 hành động lọc, 144 xác định logic bắt khả tri, 149 ứng viên xác định thành phần dịch vụ, 151 sửa đổi nhóm ứng viên khả năng dịch vụ, 157 sửa đổi ứng viên thành phần dịch vụ, 156 dịch vụ chuẩn hóa, 135 phiên bản, 265-266 trang web

www.arcitura.com/notation, 9
www.bigdatapatt erns.org, 3
www.bigdatascienceschool.com, 11
www.cloudpatt erns.org, 3, 60
www.cloudschool.com, 10
www.serviceorientation.com, 10, 290
www.servicetechbooks.com, 6, 9, 11, 290, 306, 321
www.servicetechspecs.com, 10
www.soa-manifesto.com, 8
www.soapatt erns.org, 3, 8, 321
www.soaschool.com, 10
www.whatiscloud.com, 60
www.whatisrest.com, 10, 306 tài liệu WSDL

dưới dạng mô-đun,

214 không gian tên, 215

Prentice Hall Service Technology Series from Thomas Erl

THE WORLD'S TOP-SELLING SERVICE TECHNOLOGY TITLES

ABOUT THE SERIES

Chuỗi Công nghệ Dịch vụ Prentice Hall của Thomas Erl nhằm mục đích cung cấp cho ngành CNTT một mức độ hướng dẫn và hướng dẫn khách quan, thực tế và toàn diện nhất quán trong các lĩnh vực ứng dụng và đổi mới công nghệ dịch vụ và khoa học CNTT. Mỗi tựa sách trong bộ sách này được tác giả liên hệ với các tựa sách khác nhằm thiết lập một thư viện kiến thức bổ sung. Mặc dù bộ sách bao gồm nhiều chủ đề liên quan đến công nghệ dịch vụ, nhưng mỗi tiêu đề đều được soạn thảo tuân thủ các quy ước về ngôn ngữ, từ vựng và hình ảnh minh họa chung để cho phép người đọc liên tục khám phá nhiều chủ đề khác nhau.

nghiên cứu và giáo dục.



servicetechbooks.com/community

ABOUT THE SERIES EDITOR

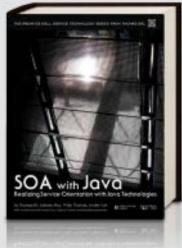
Thomas Erl là tác giả CNTT có sách bán chạy nhất, biên tập viên loạt sách về Công nghệ dịch vụ Prentice Hall của Thomas Erl, đồng thời là biên tập viên của Tập chí Công nghệ dịch vụ. Với tư cách là Giám đốc điều hành của Arcitura Education Inc., Thomas đã lãnh đạo việc phát triển chương trình giảng dạy cho các chương trình chứng nhận Chuyên gia được chứng nhận Khoa học dữ liệu lớn (BDSCP), Chuyên gia được chứng nhận nền tảng đám mây (CCP) và Chứng nhận chuyên nghiệp được chứng nhận SOA (SOACP). Các chứng nhận ngành chính thức, trung lập với nhà cung cấp. Thomas đã đi lưu diễn hơn 20 quốc gia với tư cách là diễn giả và người hướng dẫn. Hơn 100 bài báo và bài phỏng vấn của Thomas đã được xuất bản trên nhiều ấn phẩm, bao gồm Wall Street Journal và CIO Magazine.



informIT.com
THE TRUSTED TECHNOLOGY LEARNING SOURCE

 PRENTICE
HALL

 ServiceTech
PRESS



SOA với Java: Hiện thực hóa
Định hướng dịch vụ với công
ng nghệ Java
của T. Erl, S. Roy, P. Thomas,
A. Tost

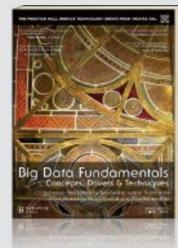
Mã số: 9780133859034
Bìa cứng, 592 trang



SOA với REST: Thiết kế, lập trình
& ràng buộc để xây dựng giải pháp
doanh nghiệp với REST
của R. Balasubramanian, B.
Carlyle, T. Erl, C. Pautasso
ISBN: 0137012519
Bìa cứng, 577 trang



Các mẫu thiết kế
điện toán đám mây
của T. Erl, R.
Cope, A. Naserpour
Mã số: 9780133858563
Bìa cứng, 528 trang



Nguyên tắc cơ bản về dữ liệu
lớn: Khái niệm, Trình
diều khiển & Kỹ thuật
của P. Buhler, T. Erl, W. Khattak
Mã số: 9780134291079
Bìa mềm, 218 trang



Kiến trúc hướng dịch
vụ: Phân tích & Thiết kế cho Dịch vụ và
Dịch vụ vi mô
(Phiên bản thứ hai)
bởi T. Erl
ISBN: 0133858588
Bìa mềm, ~ 300 trang



Web service contract
Design & Versioning
for SOA
Thiết kế & Phiên bản cho SOA
của T. Erl, A. Karmarkar,
P. Walmsley, H. Haas,
U. Yalcinalp, C. Liu,
D. Orchard, A. Tost, J. Pasley
ISBN: 013613517X
Bìa cứng, 826 trang



Quản trị SOA:
Quản lý dịch vụ chia sẻ
Tại chỗ & trên đám mây
của S. Bennett, T. Erl, C. Gee,
R. Laird, A. Manes,
R. Schneider, L. Shuster,
A. Tost, C. Venable
ISBN: 0138156751
Bìa cứng, 675 trang



SOA với .NET & Windows Azure
Restoring Service Orientation with the Microsoft Platform
Thiết kế & phát triển SOA với Microsoft .NET & Windows Azure
của D. Chou, J. deVadoss, T.
Erl, N. Gandhi, H.
Kommalapati, B. Loesgen, C.
Schittke, H. Wilhelmsen, M.
Williams
ISBN: 0131582313
Bìa cứng, 893 trang



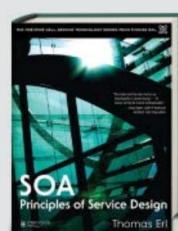
SOA thế hệ tiếp theo: Giới
thiệu ngắn gọn về công nghệ
dịch vụ &
Định hướng dịch vụ
của T. Erl, C. Gee, J.
Kress, B. Maier, H. Normann, P.
Raj, L. Shuster, B.
Trops, C. Utschig-Utschig, P.
Wik, T. Winterberg
Mã số: 9780133859041
Bìa mềm, 208 trang



Điện toán đám mây:
Khái niệm, Công nghệ
& Ngành kiến trúc
của T. Erl, Z. Mahmood,
R. Puttini
Mã số: 9780133875200
Bìa cứng, 528 trang



Kiến trúc hướng dịch vụ: Nguyên tắc SOA của Hướng dẫn hiện
trường về tích hợp thiết kế dịch vụ
Dịch vụ Web và XML
bởi T. Erl
ISBN: 0131428985
Bìa mềm, 534 trang



SOA
Principles of Service Design
Thomas Erl
Bìa cứng, 573 trang



SOA Mô hình Thiết kế SOA
bởi T. Erl
ISBN: 0136135161
Bìa cứng, 865 trang

PEARSON VUE

Các sách giáo khoa trong bộ sách này là bộ phận chính thức của
Chương trình đào tạo và cấp chứng chỉ của Arcitura. Tất cả các kỹ thi
tường ứng với các khóa học liên quan có sẵn
tại các trung tâm khảo Pearson VUE và thông qua
Giám sát trực tuyến Pearson VUE. Tham
www.pearsonvue.com/arciituta.

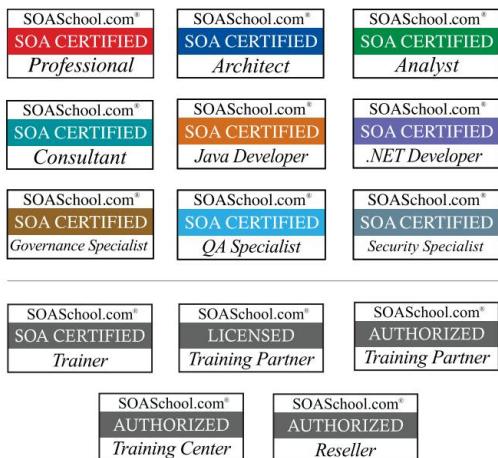
SOA & Cloud Computing Training & Certification

SOA Certified Professional (SOACP)

Content from this book and other series titles has been incorporated into the SOA Certified Professional (SOACP) program, an industry-recognized, vendor-neutral SOA certification curriculum developed by author Thomas Erl in cooperation with industry experts and academic communities and provided by SOASchool.com and training partners.

The SOA Certified Professional curriculum is comprised of a collection of 23 courses and labs that can be taken with or without formal testing and certification. Training can be delivered anywhere in the world by Certified Trainers. A comprehensive self-study program is available for remote, self-paced study, and exams can be taken world-wide via testing centers.

Dozens of public workshops are scheduled every quarter around the world by regional training partners.



All courses are reviewed and revised on a regular basis to stay in alignment with industry developments.

For more information, visit: www.soaschool.com



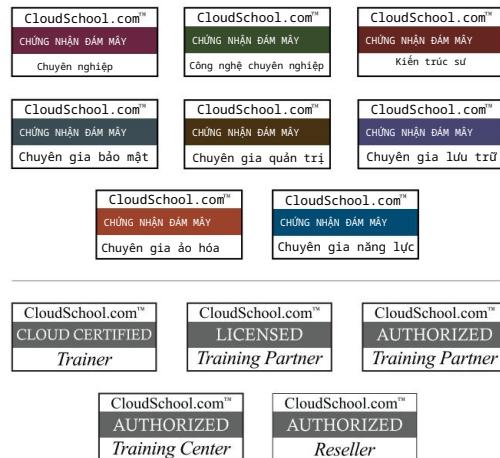
www.soaworkshops.com • www.soaseftstudy.com

Arcitura™
the IT education company

Cloud Certified Professional (CCP)

Chương trình Certified Cloud Professional (CCP), do CloudSchool.com cung cấp, thiết lập một loạt chứng chỉ ngành trung lập với nhà cung cấp dành riêng cho các lĩnh vực chuyên môn trong lĩnh vực điện toán đám mây. Cũng do tác giả Thomas Erl thành lập, chương trình này cho phép các chuyên gia CNTT học hỏi và được công nhận trong các lĩnh vực chủ đề chung và chuyên ngành trong lĩnh vực điện toán đám mây.

Chương trình giảng dạy Cloud Certified Professional bao gồm 21 khóa học và phòng thí nghiệm, mỗi khóa học đều có một bài kiểm tra tương ứng. Các hội thảo đào tạo tự nhiên và công cộng có thể được cung cấp trên toàn thế giới bởi các Giảng viên được chứng nhận. Bộ dụng cụ tự học cũng được cung cấp để bạn có thể tự học từ xa, theo nhịp độ riêng nhằm hỗ trợ các buổi hội thảo do người hướng dẫn hướng dẫn.



All courses are reviewed and revised on a regular basis to stay in alignment with industry developments.

For more information, visit: www.cloudschool.com



www.cloudworkshops.com • www.cloudselfstudy.com

PEARSON VUE

Tất cả các bài thi Arcitura đều có tại Pearson VUE testing trung tâm và thông qua Pearson VUE Online Proctoring

ArcituraTM

Big Data Science School

Vendor-Neutral Big Data Training & Certification
15 Course Modules • 15 Exams • 7 Certifications

The Big Data Science Certified Professional (BDSCP) program from the Arcitura Big Data Science School is dedicated to excellence in the fields of Big Data science, analysis, analytics, business intelligence, technology architecture, design and development, as well as governance. A collection of courses establishes a set of vendor-neutral industry certifications with different areas of specialization. Founded by best-selling author Thomas Erl, this curriculum enables IT professionals to develop real-world Big Data science proficiency. Because of the vendor-neutral focus of the course materials, the skills acquired by attaining certifications are applicable to any vendor or open-source platform.

For more information, visit: www.bigdatascienceschool.com



Big Data Science School
CERTIFIED BIG DATA
Professional

Big Data Science School
CERTIFIED BIG DATA
Science Professional

Big Data Science School
CERTIFIED BIG DATA
Scientist

Big Data Science School
CERTIFIED BIG DATA
Consultant

Big Data Science School
CERTIFIED BIG DATA
Engineer

Big Data Science School
CERTIFIED BIG DATA
Architect

Big Data Science School
CERTIFIED BIG DATA
Governance Specialist

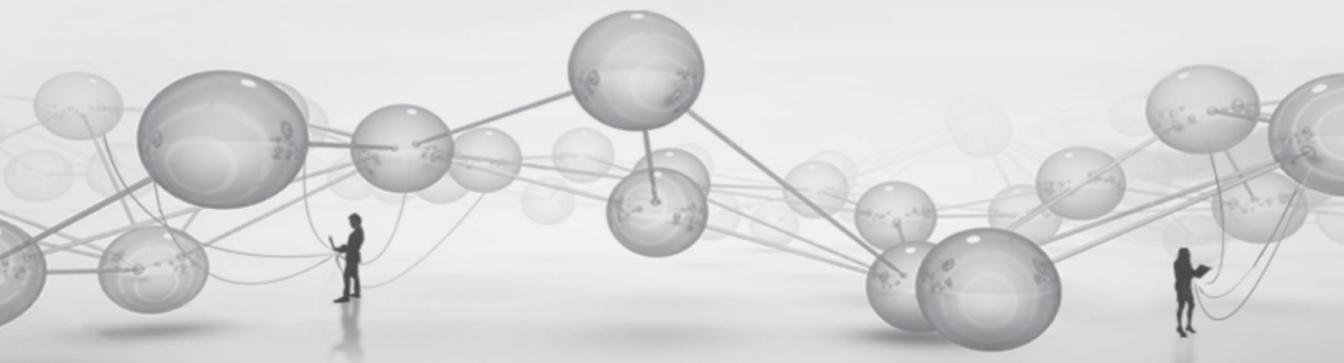
Big Data Science School
CERTIFIED BIG DATA
Trainer

Big Data Science School
LICENSED
Training Partner

Big Data Science School
AUTHORIZED
Training Partner

Big Data Science School
AUTHORIZED
Training Center

Big Data Science School
AUTHORIZED
Reseller



ArcituraTM
the IT education company

PEARSON VUE

All Arcitura exams are available at Pearson VUE testing centers and via Pearson VUE Online Proctoring