KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Rozproszone technologie WebServices

Prezentacja jest współfinansowana przez Unię Europejską w ramach Europejskiego Funduszu Społecznego w projekcie pt.

„Innowacyjna dydaktyka bez ograniczeń - zintegrowany rozwój Politechniki Łódzkiej - zarządzanie Uczelnią, nowoczesna oferta edukacyjna i wzmacniania zdolności do zatrudniania osób niepełnosprawnych”

Prezentacja dystrybuowana jest bezpłatnie

Politechnika Łódzka

Politechnika Łódzka, ul. Żeromskiego 116, 90-924 Łódź, tel. (042) 631 28 83
www.kapitalludzki.p.lodz.pl

Rozproszone technologie Web Services

Bartłomiej Świercz

Katedra Mikroelektroniki i Technik Informatycznych

Łódź, 11 kwietnia 2010

Wstęp

Oprogramowanie napisane w różnych językach i uruchomione na różnych platformach może wykorzystać Web Services do wymiany danych za pomocą sieci komputerowej w sposób przypominający standardową komunikację pomiędzy procesami działającymi na jednym komputerze. W przypadku użycia otwartych standardów możliwa jest współpraca programów napisanych w wielu językach (np. Java, Python) i różnych platform systemowych (np. MS Windows i Linux).

Istnieją dwie organizacje (W3C i OASIS) odpowiedzialne za standaryzację i rozwój architektury systemów nazywanych Web Services.

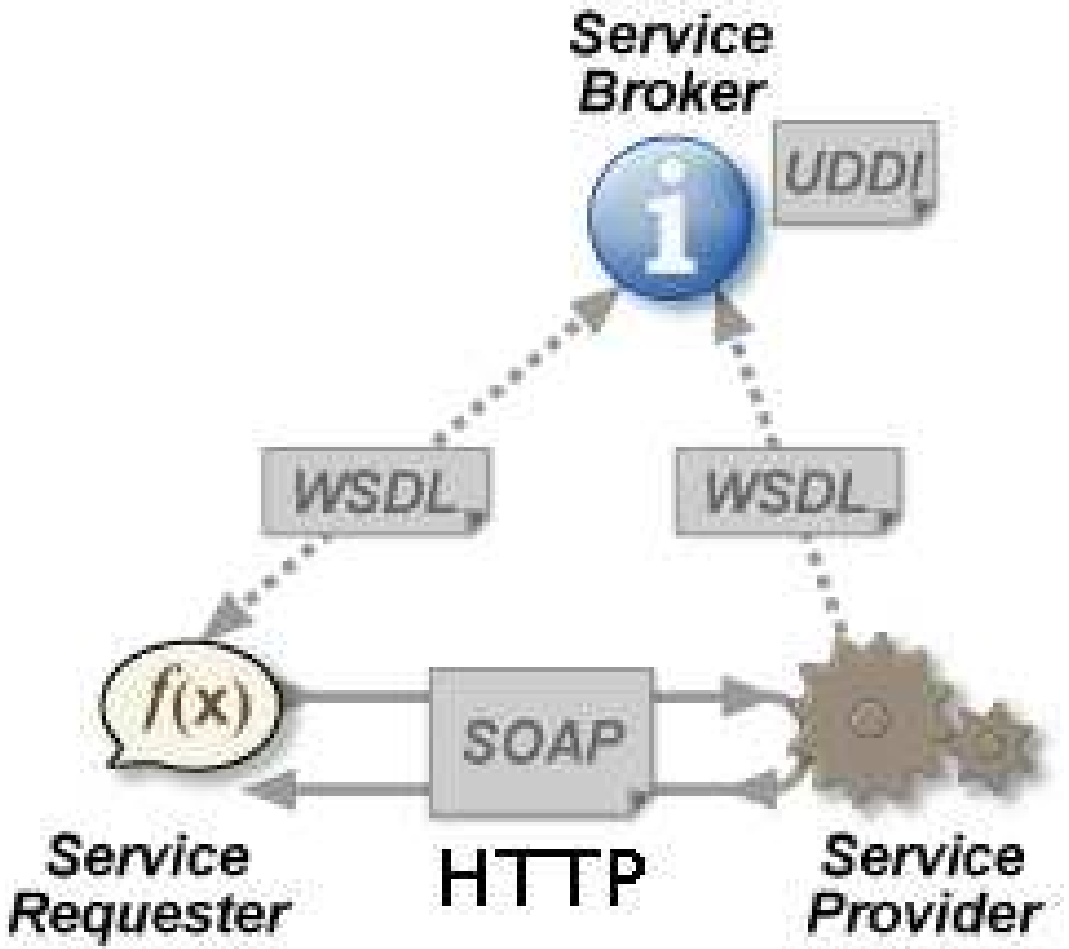
Powołana została również specjalna organizacja WS-I zajmująca się promocją technologi Web Services.

Web Services

Definicja W3C (`http://www.w3.org/TR/ws-arch/`):

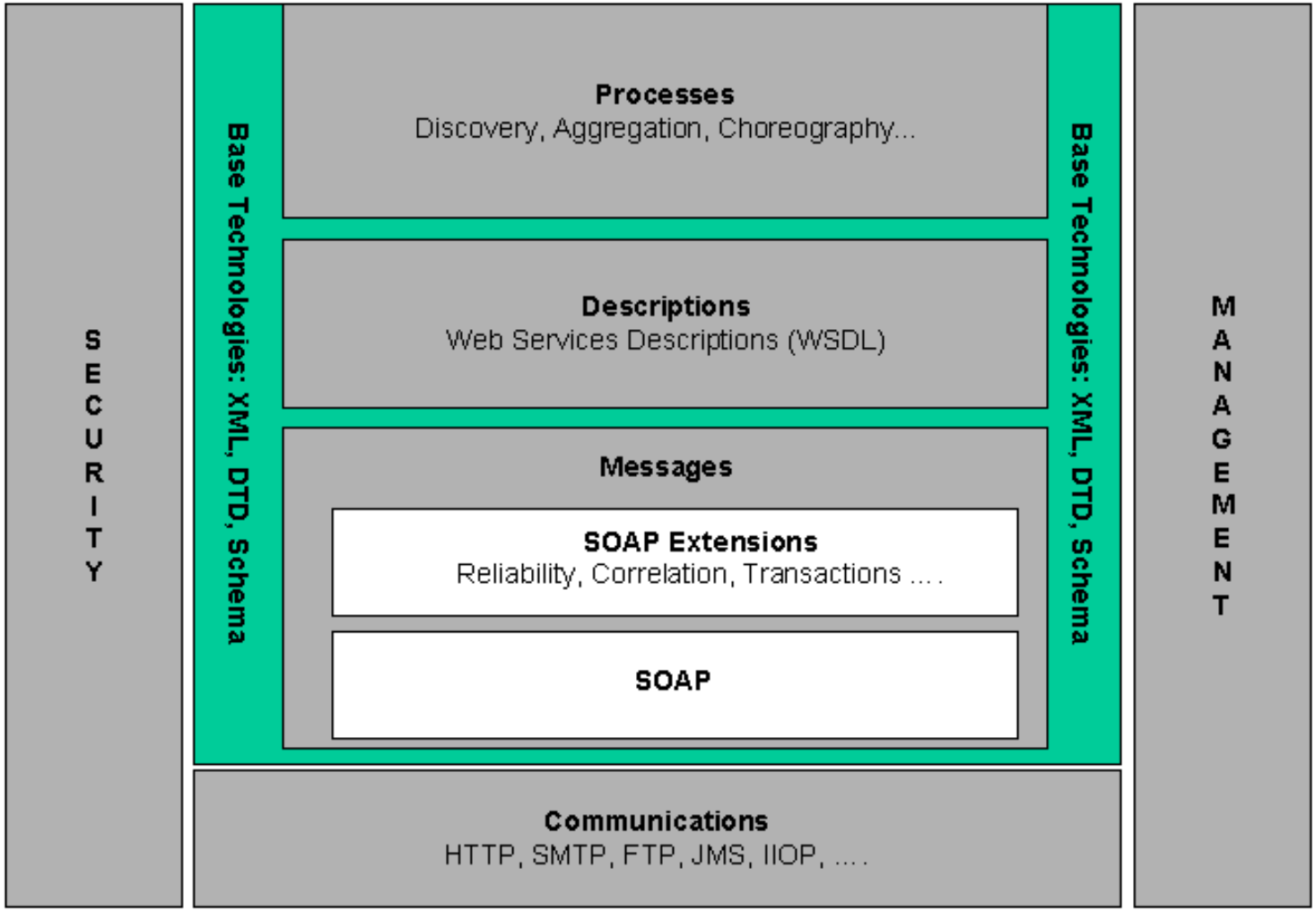
> *A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.*

Architektura Web Services



Rysunek: Autor: H. Voormann

Architektura stosowa (Web Services Protocol Stack)



Rysunek: Źródło: W3C

Języki Web Services

Języki znaczników używane w systemach Web Services:

- BEEP - Blocks Extensible Exchange Protocol
- BPEL - Business Process Execution Language
- E-Business XML
- SOAP - Simple Object Access Protocol
- UDDI - Universal Description, Discovery, and Integration
- WSDL - Web Services Description Language
- WSFL - Web Services Flow Language
- WSCL - Web Services Conversation Language
- XML-RPC - XML Remote Procedure Call

Standardy

Na obecny kształt systemów Web Services mają wpływ poniższe standardy:

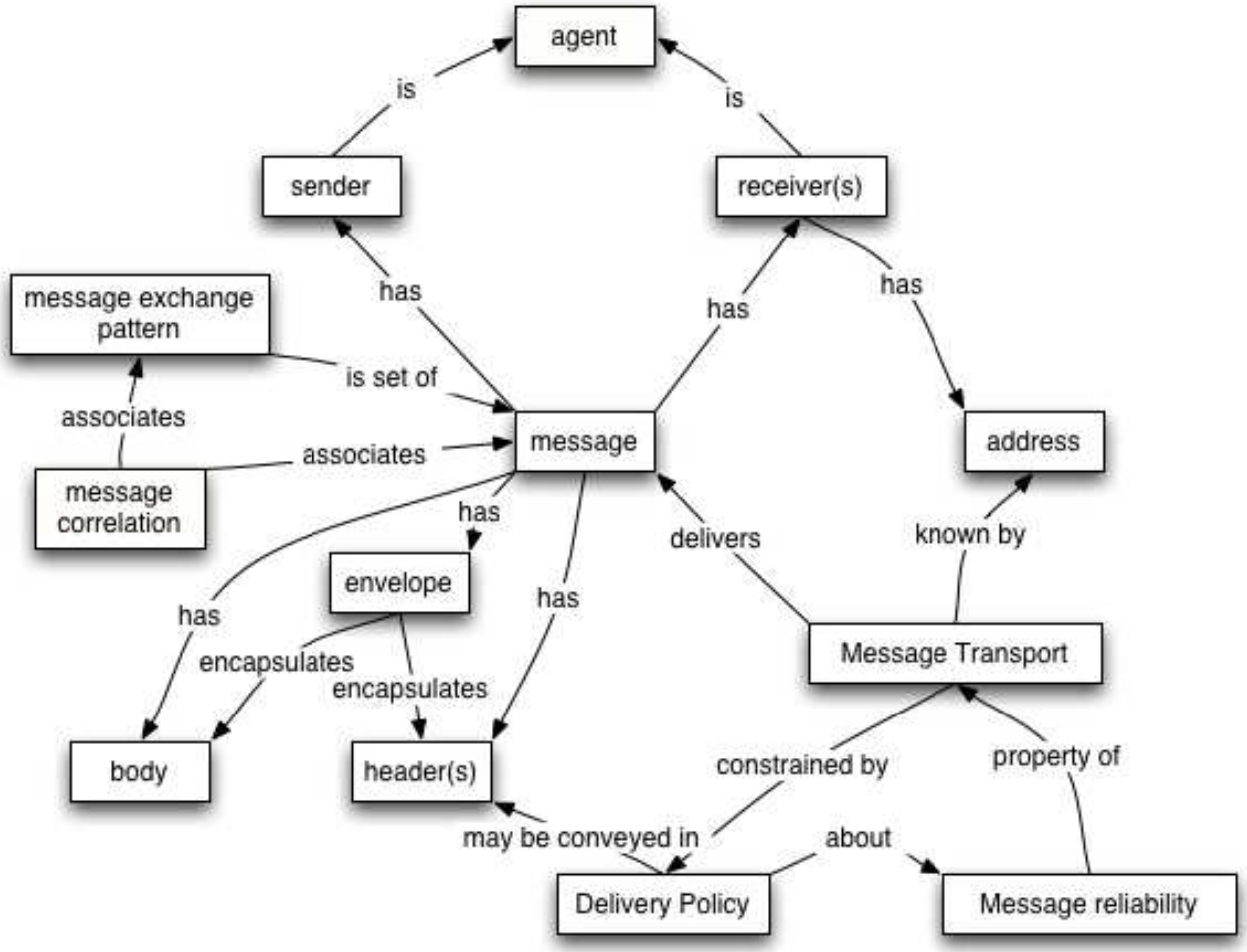
- SOAP 1.1
- WSDL 1.1
- UDDI 2.0
- XML 1.0 (Second Edition)
- XML Schema Part 1: Structures
- XML Schema Part 2: Datatypes

Standardy C.D.

Na obecny kształt systemów Web Services mają wpływ poniższe standardy:

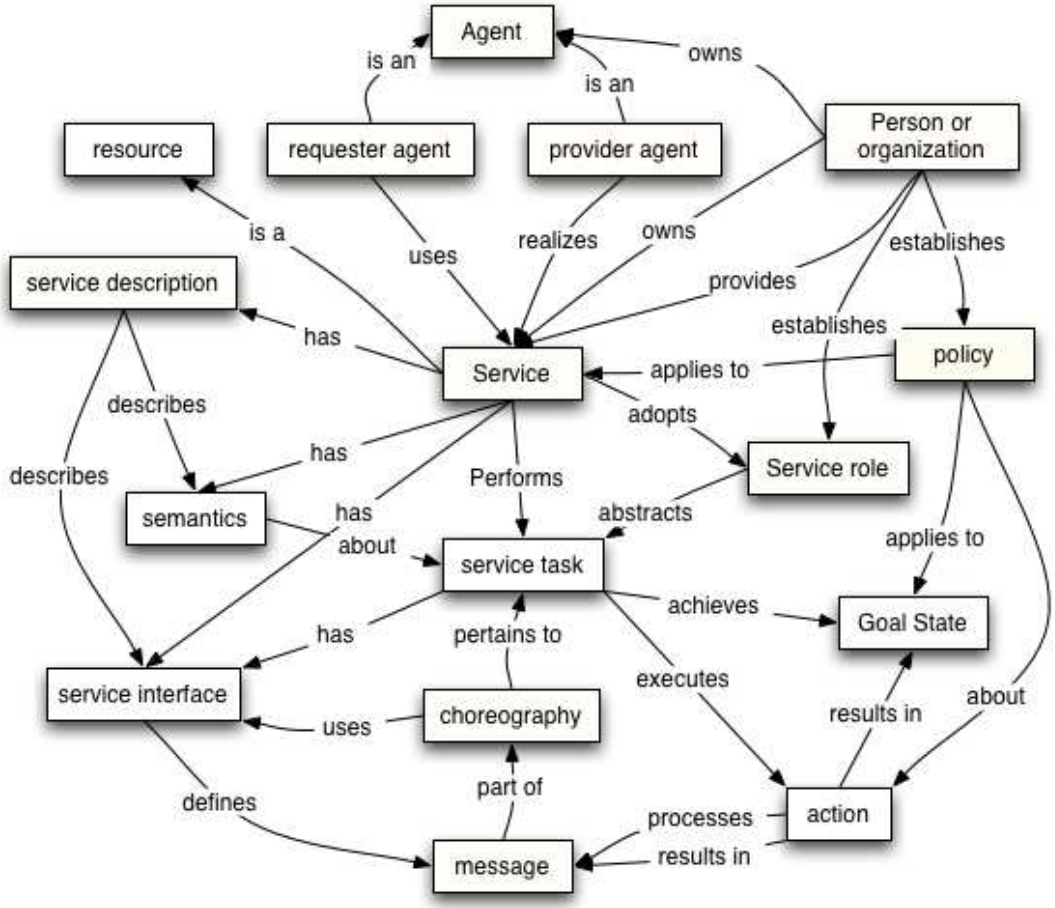
- RFC2246: The Transport Layer Security Protocol version 1.0
- RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile
- RFC2616: HyperText Transfer Protocol 1.1
- RFC2818: HTTP over TLS
- RFC2965: HTTP State Management Mechanism
- The Secure Sockets Layer Protocol version 3.0

Komunikacja zorientowana na wiadomości



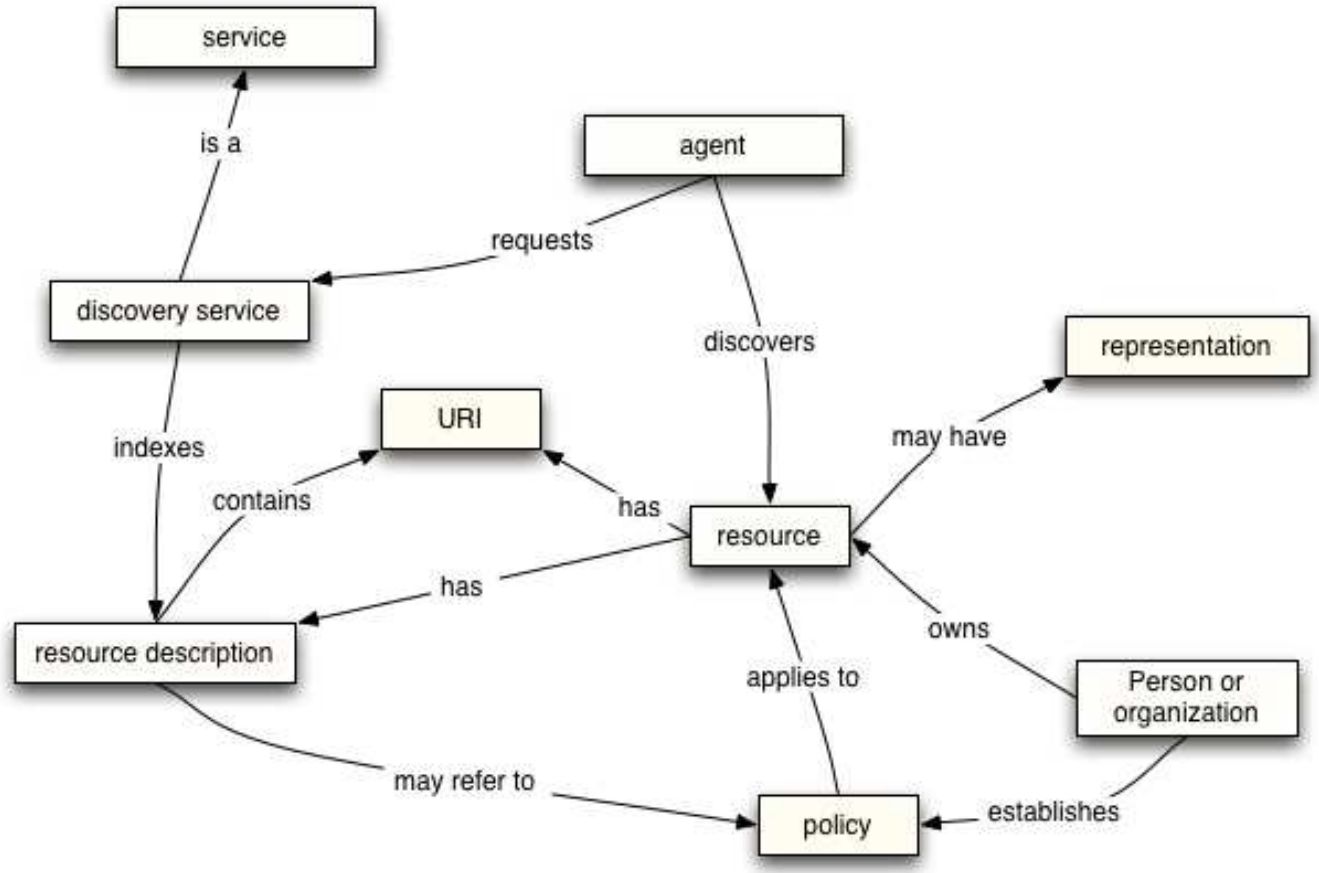
Rysunek: Źródło: W3C

Model aplikacji zorientowanej na serwisy



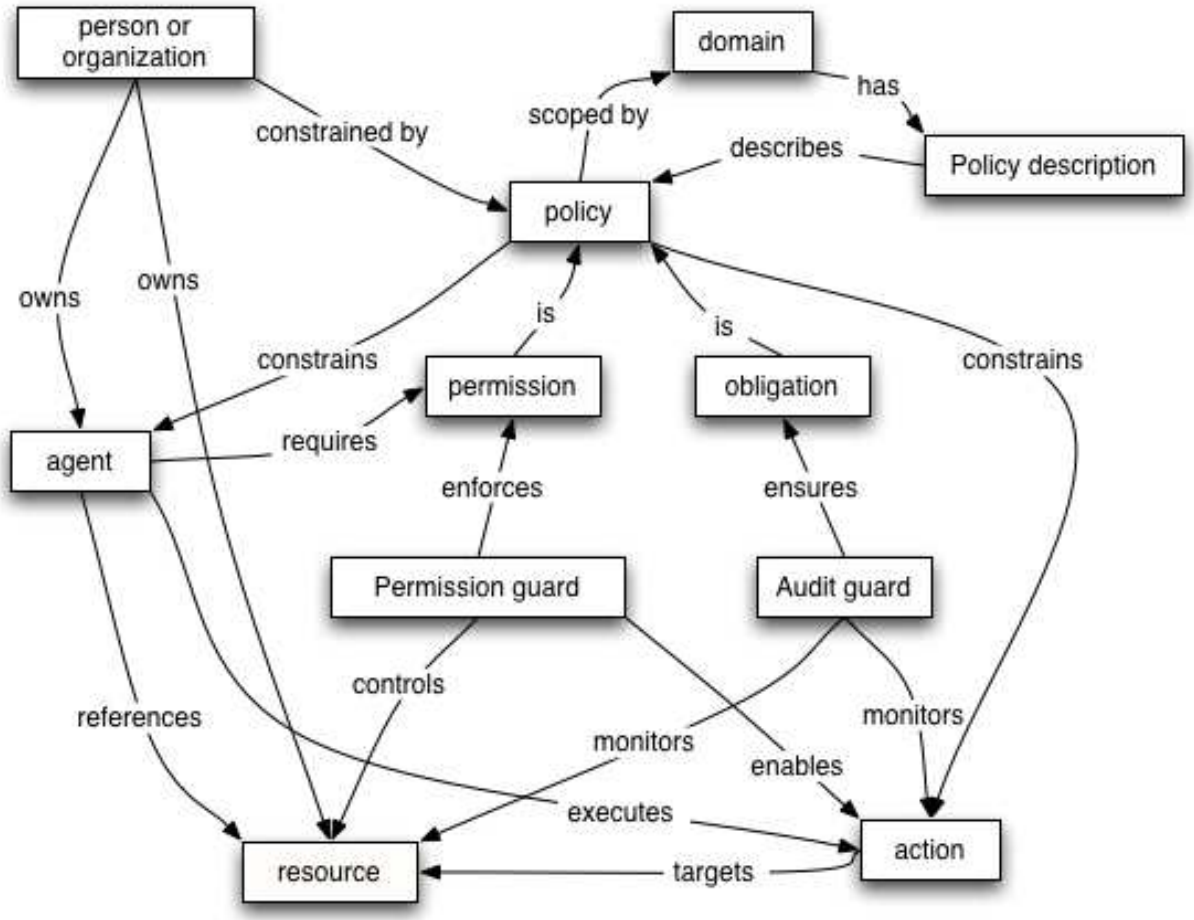
Rysunek: Źródło: W3C

Model aplikacji udostępniającej zasoby



Rysunek: Źródło: W3C

Model polityki bezpieczeństwa



Rysunek: Źródło: W3C

Zalety Web Services

- Web Services pozwalają na współdziałanie programów napisanych w różnych językach i działających na różnych platformach.
- Web Services używają otwartych standardów i protokołów bazujących na tekscie czytelnym dla człowieka.
- Poprzez wykorzystanie protokołu HTTP jako kanału transportowego Web Services pozbywają się problemu obecności firewall w sieci.
- Web Services pozwalają na łączenie zasobów wielu instytucji i firm.

Wady Web Services

- Ciągle brakuje standardów dla bardziej zaawansowanych technik (np. transakcje) realizowanych w oparciu o Web Services.
- Aplikacje oparte o Web Services mają słabą wydajność w porównaniu do aplikacji zbudowanych w oparciu o inne technologie rozproszone (DCOM, CORBA, RMI).
- Poprzez wykorzystanie protokołu HTTP i obejściu większości zapór sieciowych pojawiają się problemy z właściwym zabezpieczeniem aplikacji.

Powody dla których warto użyć Web Services

Podstawowym powodem dla którego warto użyć technologi Web Services jest naturalny podział projektu na moduły, co ułatwia późniejszy rozwój aplikacji.

Wstęp do XML-RPC

Internet dostarcza wiele możliwości programistą piszącym aplikacje komunikujące się pomiędzy różnymi komputerami i systemami. Zazwyczaj projektowanie rozproszonego oprogramowania jest procesem złożonym, szczególnie jeżeli oprócz aplikacji projektuje się również protokół komunikacji.

Wstęp do XML-RPC

Internet dostarcza wiele możliwości programistą piszącym aplikacje komunikujące się pomiędzy różnymi komputerami i systemami. Zazwyczaj projektowanie rozproszonego oprogramowania jest procesem złożonym, szczególnie jeżeli oprócz aplikacji projektuje się również protokół komunikacji.

Obecnie jednak podstawowym pytaniem podczas projektowania aplikacji rozproszonej jest pytanie, który sposób komunikacji (jaką bibliotekę) należy wybrać.

Wstęp do XML-RPC

Dwa najbardziej popularne protokoły (framework) to:

- CORBA
- DCOM

Obydwie technologie są bardzo silne i pozwalają na zbudowanie dowolnej architektury połączeń. Są jednak skomplikowane.

XML-RPC

XML-RPC – XML-Remote Procedure Call

XML-RPC jest technologią zdalnego wykonywania procedur. Jest to prosta, przenośna i niezależna od języka programowania technologia komunikacji rozproszonych systemów wykorzystująca protokół HTTP jako kanał transportowy język XML do opisu danych.

`http://www.xml-rpc.org`

Protokół HTTP

Hypertext Transfer Protocol jest protokołem warstwy aplikacji przeznaczonym dla rozproszonych i zorientowanych na media systemów informacyjnych. Jest to protokół bardzo ogólny, bezstanowy, który może być wykorzystany do dystrybucji informacji hipertekstowej oraz jako **serwer rozproszonych zorientowanych obiektowo systemów (poprzez rozszerzenie metody "request", kodów błędów i nagłówków).**

`http://www.w3.org/Protocols/rfc2616/rfc2616.html`

Protokół HTTP - przykład

Przykład zapytania do serwera HTTP

```
GET / HTTP/1.1
Host: localhost:8000
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US;
rv:1.7.10) Gecko/20050911 Firefox/1.0.6 (Debian
package 1.0.6-5)
Accept: text/xml, application/xml,
application/xhtml+xml, text/html;q=0.9,
text/plain;q=0.8,image/png, */*;q=0.5
Accept-Language: pl,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
```

Protokół HTTP - przykład

Przykład odpowiedzi serwera HTTP

```
HTTP/1.1 302 Found
Date: Fri, 28 Oct 2005 05:39:29 GMT
Server: Apache/2.0.54 (Debian GNU/Linux)
PHP/4.3.10-15
Location: http://localhost:8000/apache2-default/
Content-Length: 323
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1
```

XML-RPC: omówienie

Komunikat XML-RPC jest realizowany jako żądanie HTTP-POST. Treścią wiadomości jest struktura XML. Procedura wykonywana jest na serwerze i wynik jest zwracany jako poprawnie sformatowany dokument XML.
Parametrem procedury mogą być liczby, napisy, struktury i tablice.

XML-RPC: omówienie

Przykład zapytania XML-RPC

```
POST / HTTP/1.0
Host: localhost:9000
User-Agent: xmlrpclib.py/1.0.1
Content-Type: text/xml
Content-Length: 187
<?xml version='1.0'?>
<methodCall>
    <methodName>pow</methodName>
    <params>
        <param>
            <value><int>2</int></value>
        </param>
        <param>
            <value><int>8</int></value>
        </param>
    </params>

</methodCall>
```

XML-RPC: omówienie

Przykład odpowiedzi XML-RPC

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.3 Python/2.3.5
Date: Wed, 26 Oct 2005 11:44:59 GMT
Content-type: text/xml
Content-length: 123
<?xml version='1.0'?>
<methodResponse>
    <params>
        <param>
            <value><int>256</int></value>
        </param>
    </params>
</methodResponse>
```

Typy skalarne

Znacznik:	Opis:
<i4> lub <int>	cztero-bajtowa liczba całkowita ze znakiem
<boolean>	wartość logiczna: 0 (fałsz) lub 1 (prawda)
<string>	napis
<double>	liczba zmiennoprzecinkowa podwójnej precyzji
<dateTime.iso8601>	format daty i czasu: 19980717T14:08:55
<base64>	wartość binarna zakodowana algorytmem Base64

Typy strukturalne

```
<struct>
    <member>
        <name>lowerBound</name>
        <value><i4>18</i4></value>
    </member>
    <member>
        <name>upperBound</name>
        <value><i4>139</i4></value>
    </member>
</struct>
```

Typy tablicowe

```
<array>
    <data>
        <value><i4>12</i4></value>
        <value><string>Egypt</string></value>
        <value><boolean>0</boolean></value>
        <value><i4>-31</i4></value>
    </data>
</array>
```

Czym jest SOAP?

- Akronim SOAP oznacza Simple Object Access Protocol.
- SOAP jest protokołem komunikacyjnym służącym do wymiany wiadomości pomiędzy aplikacjami.
- SOAP przeznaczony jest do wymiany danych w sieci internet.
- SOAP jest niezależny od platformy sprzętowej oraz języka programowania.
- Wiadomości SOAP są opisane za pomocą XML.
- SOAP jest standardem W3C: `http://www.w3.org/TR/soap/`

Dlaczego SOAP?

Komunikacja

Systemy wykorzystujące komunikację rozproszoną w postaci Remote Procedure Call jak CORBA lub DCOM napotykają na problemy bezpieczeństwa i dostępności serwisów w sieci internet. Przeszkodą są licznie wykorzystywane w sieci Internet systemy firewall i proxy.

Powszechność

SOAP jest wykorzystywany przez wiele firm i aplikacji komercyjnych. Jest podstawowym elementem architektury Microsoft .NET dla aplikacji internetowych.

Założenia projektowe SOAP

Podstawowym założeniem projektu SOAP była prostota i rozszerzalność projektowanego protokołu. Oznacza to, że wiele elementów występujących w innych protokołach RPC nie pojawia się w SOAP np:

- Rozproszony odśmiecacz pamięci (Distributed garbage collection).
- Przekazywanie obiektów przez referencję (Objects-by-reference).

Podział SOAP

Protokół SOAP logicznie dzieli się na trzy części:

- Kopertę (SOAP envelope) - jest to definicja opisująca co znajduje się w wiadomości, dla kogo jest przeznaczona i czy wiadomość jest obowiązkowa lub opcjonalna.
- Zasady kodowania (SOAP encoding rules) - możliwość definiowania typów wykorzystywanych przez aplikację.
- Reprezentacja RPC (SOAP RPC representation) - określa reguły opisu zdalnych procedur i ich odpowiedzi.

Wiadomość SOAP

Wiadomość jest podstawową komunikacją jednokierunkową pomiędzy nadawcą, a odbiorcą. Wiadomość SOAP może wykorzystać różne protokoły jako medium transportowe (np. HTTP).

Aplikacja SOAP, która odebrała wiadomość SOAP musi ją przetworzyć zgodnie z zasadami:

- Rozpoznać wszystkie części wiadomości.
- Zdecydować, czy wszystkie części wiadomości są wspierane przez aplikację.
- Jeżeli aplikacja nie jest adresatem wiadomości to musi ją przekazać do adresata.

Przykładowa wiadomość SOAP

Wiadomość SOAP osadzona w HTTP - żądanie

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"


<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <m:GetLastTradePrice xmlns:m="Some-URI">
            <symbol>DIS</symbol>
        </m:GetLastTradePrice>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Przykładowa wiadomość SOAP

Wiadomość SOAP osadzona w HTTP - odpowiedź

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn


<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    <SOAP-ENV:Body>
        <m:GetLastTradePriceResponse xmlns:m="Some-URI">
            <Price>34.5</Price>
        </m:GetLastTradePriceResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Budowa wiadomości SOAP

Wiadomość SOAP jest dokumentem XML zawierającym elementy:

- Wymagane elementy koperty, które identyfikują dokument XML jako wiadomość SOAP.
- Opcjonalny nagłówek.
- Wymagany element Body, który zawiera informacje żądającego lub odpowiadającego na wiadomość.
- Opcjonalny element Fault opisujący błędy mogące pojawić się podczas przetwarzania wiadomości.

Składnia wiadomości SOAP

- Wiadomość SOAP musi być zapisana przy użyciu XML.
- Wiadomość SOAP musi zawierać przestrzeń nazw SOAP Envelope.
- Wiadomość SOAP musi zawierać przestrzeń nazw SOAP Encoding.
- Wiadomość SOAP nie może zawierać definicji DTD.
- Wiadomość SOAP nie może zawierać instrukcji przetwarzania XML.

Szkielet wiadomości SOAP

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
    <soap:Header>
        :
    </soap:Header>
    <soap:Body>
        :
        <soap:Fault>
            :
        </soap:Fault>
    </soap:Body>
</soap:Envelope>
```

Koperta wiadomości SOAP

Koperta wiadomości SOAP jest jednocześnie głównym (root) elementem dokumentu XML. Definiuje dokument XML jako wiadomość SOAP.

Struktura koperty

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  : :
  : :   Miejsce na wiadomość.
</soap:Envelope>
```

Koperta zawsze musi być skojarzona z przestrzenią nazw `http://www.w3.org/2001/12/soap-envelope`.

Atrybut encodingStyle

Atrybut encodingStyle jest używany do zdefiniowania typów używanych w wiadomości. Wiadomość SOAP nie ma domyślnego kodowania.

Element Header

Opcjonalny element Header zawiera informacje specyficzne dla aplikacji (autoryzacja, transakcje itd.). Jeżeli element Header jest obecny w wiadomości to musi być pierwszym potomkiem elementu Envelope.

Element Body

Element Body jest wymagany w każdej wiadomości SOAP. Elementy potomne elementu Body mogą być umieszczone w innej przestrzeni nazw. Specyfikacja SOAP definiuje tylko jeden element potomny umieszczony w domyślnej przestrzeni nazw. Jest to element Fault.

Element Body

Przykład zapytania

```
<?xml version="1.0"?>

<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding"
    >

    <soap:Body>
        <m:GetPrice
        xmlns:m="http://www.w3schools.com/prices">
            <m:Item>Apples</m:Item>
        </m:GetPrice>
    </soap:Body>

</soap:Envelope>
```

Element Body

Przykład odpowiedzi

```
<?xml version="1.0"?>

<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding"
    >

    <soap:Body>
        <m:GetPriceResponse
        xmlns:m="http://www.w3schools.com/prices">
            <m:Price>1.90</m:Price>
        </m:GetPriceResponse>
    </soap:Body>

</soap:Envelope>
```

Element Fault

Informacja o błędzie dotyczącym wiadomości SOAP przekazywana jest wewnątrz elementu Fault. Element Fault musi być potomkiem elementu Body i może wystąpić tylko raz w dokumencie.

Element Fault

Informacja o błędzie dotyczącym wiadomości SOAP przekazywana jest wewnątrz elementu Fault. Element Fault musi być potomkiem elementu Body i może wystąpić tylko raz w dokumencie.

Przykład:

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Body>
        <SOAP-ENV:Fault>
            <faultcode>
                SOAP-ENV:MustUnderstand
            </faultcode>
            <faultstring>
                SOAP Must Understand Error
            </faultstring>
        </SOAP-ENV:Fault>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Elementy potomne elementu Fault

Element Fault może mieć zagnieżdżone elementy:

Element:	**Opis:**
<faultcode>	Kod identyfikujący błąd.
<faultstring>	Czytelny opis kodu.
<faultactor>	Informacja o tym co wywołało błąd.
<detail>	Dodatkowe informacje o błędzie.

Kody błędów

Kod:	**Opis:**
VersionMismatch	Nieprawidłowa przestrzeń nazw dla elementu Envelope.
MustUnderstand	Odbiorca wiadomości nie zrozumiał elementu, który miał ustawiony atrybut mustUnderstand.
Client	Wiadomość została nieprawidłowo zbudowana lub zawiera niepoprawne dane.
Server	Wystąpił błąd podczas pracy serwera.

Wstęp do WSDL

- WSDL jest skrótem od Web Services Description Language.
- WSDL jest dokumentem XML.
- WSDL służy do opisywania usług udostępnianych przez Web Services.
- WSDL jest używany do lokalizacji usługi (Web Services).
- WSDL W3C draft: `http://www.w3.org/TR/wsdl`

Wstęp do WSDL

- WSDL jest skrótem od Web Services Description Language.
- WSDL jest dokumentem XML.
- WSDL służy do opisywania usług udostępnianych przez Web Services.
- WSDL jest używany do lokalizacji usługi (Web Services).
- WSDL W3C draft: `http://www.w3.org/TR/wsdl`

WSDL jest powiązany z technologiami: SOAP 1.1, HTTP GET/POST i MIME.

Przykład dokumentu XML

http://www.w3.org/TR/wsdl#_wsdl Część 1:

```
<?xml version="1.0"?>
<definitions name="StockQuote"

targetNamespace="http://example.com/stockquote.wsdl"
          xmlns:tns="http://example.com/stockquote.wsdl"
          xmlns:xsd1="http://example.com/stockquote.xsd"
          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
          xmlns="http://schemas.xmlsoap.org/wsdl/">
```

Przykład dokumentu XML

http://www.w3.org/TR/wsdl#_wsdl Część 2:

```
<types>
<schema targetNamespace="http://example.com/stockquote.xsd"
          xmlns="http://www.w3.org/2000/10/XMLSchema">
    <element name="TradePriceRequest">
      <complexType>
        <all>
          <element name="tickerSymbol" type="string"/>
        </all>
      </complexType>
    </element>
    <element name="TradePrice">
      <complexType>
        <all>
          <element name="price" type="float"/>
        </all>
      </complexType>
    </element>
</schema>
</types>
```

Przykład dokumentu XML

http://www.w3.org/TR/wsdl#_wsdl Część 3:

```
<message name="GetLastTradePriceInput">
    <part name="body" element="xsd1:TradePriceRequest"/>
</message>

<message name="GetLastTradePriceOutput">
    <part name="body" element="xsd1:TradePrice"/>
</message>

<portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
        <input message="tns:GetLastTradePriceInput"/>
        <output message="tns:GetLastTradePriceOutput"/>
    </operation>
</portType>
```

Przykład dokumentu XML

http://www.w3.org/TR/wsdl#_wsdl Część 4:

```
<binding name="StockQuoteSoapBinding"
type="tns:StockQuotePortType">
    <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetLastTradePrice">
        <soap:operation
        soapAction="http://example.com/GetLastTradePrice"/>
        <input>
            <soap:body use="literal"/>
        </input>
        <output>
            <soap:body use="literal"/>
        </output>
    </operation>
</binding>
```

Przykład dokumentu XML

http://www.w3.org/TR/wsdl#_wsdl Część 5:

```
    <service name="StockQuoteService">
        <documentation>My first service</documentation>
        <port name="StockQuotePort"
        binding="tns:StockQuoteBinding">
            <soap:address
            location="http://example.com/stockquote"/>
        </port>
    </service>

</definitions>
```

Przykład dokumentu XML

Dokument WSDL jest zbiorem definicji!

Podstawowe elementy WSDL

Element	Znaczenie:
<portType>	Operacje dostarczane przez Web Services
<message>	Wiadomości używane przez Web Services
<types>	Definicja typów używanych przez Web Services
<binding>	Opis protokołu używanego przez Web Services

Podstawowe elementy WSDL

Element	Znaczenie:
<portType>	Operacje dostarczane przez Web Services
<message>	Wiadomości używane przez Web Services
<types>	Definicja typów używanych przez Web Services
<binding>	Opis protokołu używanego przez Web Services

Dokument WSDL może zawierać inne elementy rozszerzające jego funkcjonalność oraz elementy pozwalające na łączenie wielu dokumentów WSDL w jeden opisujący rożne Web Services.

Struktura dokumentu WSDL

```
<definitions>

    <types>
        . . . . . .
    </types>

    <message>
        . . . . . .
    </message>

    <portType>
        . . . . . .
    </portType>

    <binding>
        . . . . . .
    </binding>

</definitions>
```

Element WSDL portType

Jest to najważniejszy element dokumentu WSDL. Definiuje usługi Web Services i powiązane z nimi wiadomości SOAP. Element portType można porównać do klasy w tradycyjnym scentralizowanym programowaniu.

Element WSDL portType

Jest to najważniejszy element dokumentu WSDL. Definiuje usługi Web Services i powiązane z nimi wiadomości SOAP. Element portType można porównać do klasy w tradycyjnym scentralizowanym programowaniu.

Rodzaj operacji:	**Opis:**
One-way	Operacja pobiera dane ale nic nie zwraca (procedura)
Request-response	Operacja pobiera dane i zwraca wynik (funkcja)
Solicit-response	Operacja wysyła rządanie i czeka na odpowiedź
Notification	Operacja wysyła rządanie i nie czeka na odpowiedź

Element WSDL message

Element message definiuje dane na których operują funkcje dostarczane przez Web Services. Element ten można porównać do deklaracji funkcji w tradycyjnych językach programowania.

Element WSDL types

Element types definiuje typy danych, które są używane przez Web Services. Dokumenty WSDL używają XML Schema do definicji typów danych.

Element WSDL binding

Element binding definiuje format wiadomości (najczęściej SOAP) i protokół transportowy (najczęściej HTTP) dla każdego portu (usługi).

Przykład dokumentu XML – tym razem ze zrozumieniem

http://www.w3.org/TR/wsdl#_wsdl Część 1:

```
<?xml version="1.0"?>
<definitions name="StockQuote"

targetNamespace="http://example.com/stockquote.wsdl"
          xmlns:tns="http://example.com/stockquote.wsdl"
          xmlns:xsd1="http://example.com/stockquote.xsd"
          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
          xmlns="http://schemas.xmlsoap.org/wsdl/">
```

Przykład dokumentu XML – tym razem ze zrozumieniem

http://www.w3.org/TR/wsdl#_wsdl Część 2:

```
<types>
<schema targetNamespace="http://example.com/stockquote.xsd"
          xmlns="http://www.w3.org/2000/10/XMLSchema">
    <element name="TradePriceRequest">
      <complexType>
        <all>
          <element name="tickerSymbol" type="string"/>
        </all>
      </complexType>
    </element>
    <element name="TradePrice">
      <complexType>
        <all>
          <element name="price" type="float"/>
        </all>
      </complexType>
    </element>
</schema>
</types>
```

Przykład dokumentu XML – tym razem ze zrozumieniem

http://www.w3.org/TR/wsdl#_wsdl Część 3:

```
<message name="GetLastTradePriceInput">
    <part name="body" element="xsd1:TradePriceRequest"/>
</message>

<message name="GetLastTradePriceOutput">
    <part name="body" element="xsd1:TradePrice"/>
</message>

<portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
        <input message="tns:GetLastTradePriceInput"/>
        <output message="tns:GetLastTradePriceOutput"/>
    </operation>
</portType>
```

Przykład dokumentu XML – tym razem ze zrozumieniem

http://www.w3.org/TR/wsdl#_wsdl Część 4:

```
<binding name="StockQuoteSoapBinding"
type="tns:StockQuotePortType">
    <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetLastTradePrice">
        <soap:operation
        soapAction="http://example.com/GetLastTradePrice"/>
        <input>
            <soap:body use="literal"/>
        </input>
        <output>
            <soap:body use="literal"/>
        </output>
    </operation>
</binding>
```

Przykład dokumentu XML – tym razem ze zrozumieniem

http://www.w3.org/TR/wsdl#_wsdl Część 5:

```
    <service name="StockQuoteService">
        <documentation>My first service</documentation>
        <port name="StockQuotePort"
        binding="tns:StockQuoteBinding">
            <soap:address
            location="http://example.com/stockquote"/>
        </port>
    </service>

</definitions>
```

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Rozproszone technologie WebServices

Prezentacja jest współfinansowana przez Unię Europejską w ramach Europejskiego Funduszu Społecznego w projekcie pt.

„Innowacyjna dydaktyka bez ograniczeń - zintegrowany rozwój Politechniki Łódzkiej - zarządzanie Uczelnią, nowoczesna oferta edukacyjna i wzmacniania zdolności do zatrudniania osób niepełnosprawnych”

Prezentacja dystrybuowana jest bezpłatnie

Politechnika Łódzka

Politechnika Łódzka, ul. Żeromskiego 116, 90-924 Łódź, tel. (042) 631 28 83
www.kapitalludzki.p.lodz.pl