KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

POLITECHNIKA BIAŁOSTOCKA

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Rozproszone systemy Internetowe

Web Services – narzędzia i przykłady

„Podniesienie potencjału uczelni wyższych jako czynnik rozwoju gospodarki opartej na wiedzy”

Nr projektu: UDA-POKL.04.01.01-00-143/09-00

Apache Axis2

http://ws.apache.org/axis2/

Obsługiwane specyfikacje:

- SOAP 1.1, 1.2
- Message Transmission Optimization Mechanism
- WSDL 1.1, 2.0
- WS-Addressing
- WS-Policy

Przykład nr 1: słownik

- Usługa implementuje operację
  tlumacz: String ⟶ String
  w postaci zwykłego obiektu
- Szkielet klienta wygenerujemy za pomocą narzędzia wsdl2java (Axis Data Binding)
- Model RPC a zatem:

```
<messageReceiver
  mep="http://www.w3.org/2004/08/wsdl/in-out"
  class =
"org.apache.axis2.rpc.receivers.RPCMessageReceiver"
/>
```

Bezpośrednie przetwarzanie komunikatów SOAP oparte na AXIOM

- Usługa:
  - W zależności od wzorca komunikacyjnego pobiera i zwraca obiekty typu OMElement
  - W opisie usługi zmieniamy rodzaj klasy obsługującej wywołania np: *org.apache.axis2.receivers.RawXMLINOutMessageReceiver*
- Komunikaty wejściowe są zgodne z opisem WSDL
- Komunikaty wyjściowe konstruowane przez metodę powinny być zgodne z opisem WSDL (pamiętajmy o przestrzeniach nazewniczych)

Wykorzystanie AXIOM: usługa

```
public OMElement tlumacz(OMElement element)  {
        element.build();
        eleament.detach();
        OMElement node = element.getFirstElement();
        String key = node.getText();
       ...
        OMFactory fac =
                  OMAbstractFactory.getOMFactory();
        OMNamespace omNs =
                  fac.createOMNamespace(...);
        OMElement response =
                  fac.createOMElement(...,omNs);
        response.addChild(...);
        return response;
    }
```

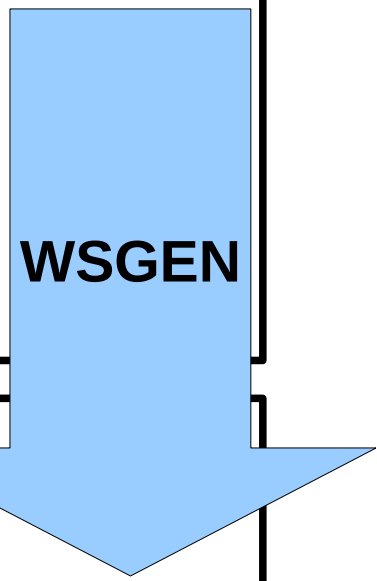
Wykorzystanie AXIOM: klient

```
EndpointReference EPR =
            new EndpointReference(...)
OMElement data = … tworzymy komunikat
Options options = new Options();
options.setTo(EPR);
options.setTransportInProtocol(
              Constants.TRANSPORT_HTTP);
ServiceClient sender = new
ServiceClient();
sender.setOptions(options);
OMElement result =
sender.sendReceive(data);
```

JAX-WS

Java API for XML Web Services - zestaw interfejsów programistycznych wspomagających realizację WS w Javie

```
@WebService
public class Slownik {
@WebMethod  public String tlumacz
((@WebParam String word) { ... } }
```




```
 @WebServiceRef(wsdlLocation="...")
 static SlownikService service;
 Slownik port = service.getSlownikPort();
 port.tlumacz(„pies”);
```

Dodatkowe zadania do przebadania/przemyślenia

- Inne sposoby transportu komunikatów np. TCP zamiast HTTP
- Asynchroniczne wywołanie usługi
- Jeszcze jedna implementacja - Apache CXF
- Style wywołań w WS: **RPC (literal)**, **Document (literal)**