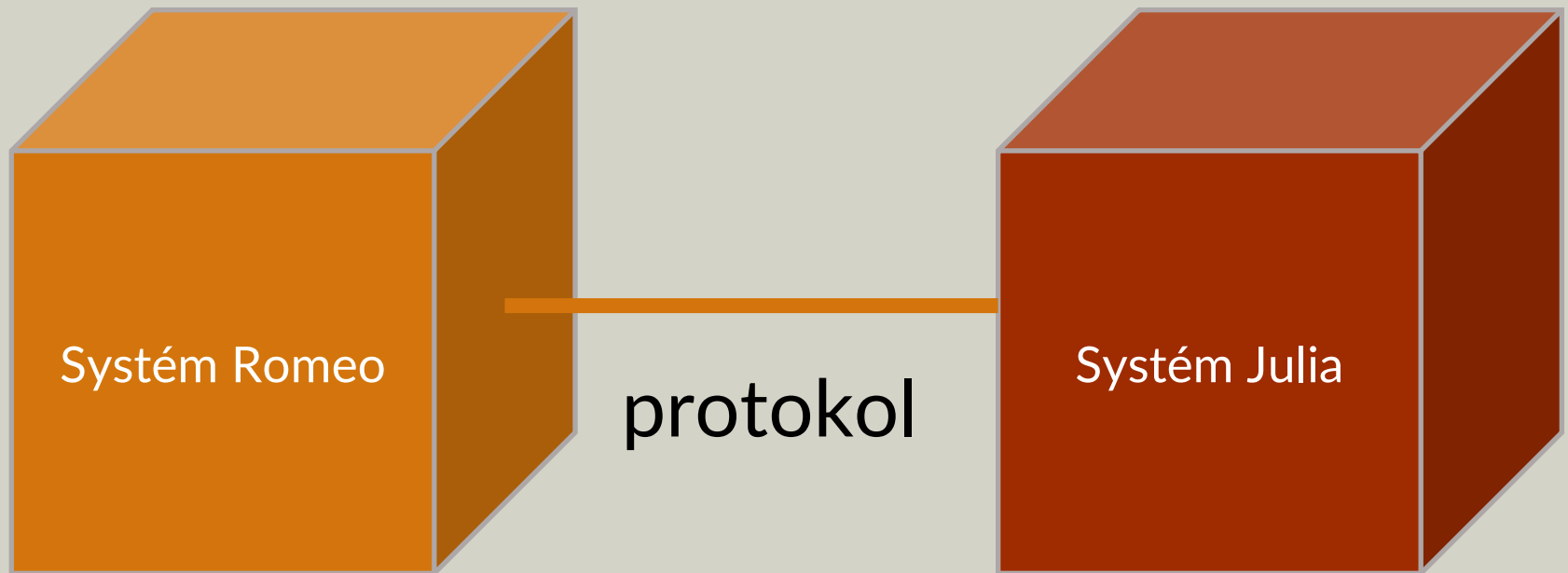


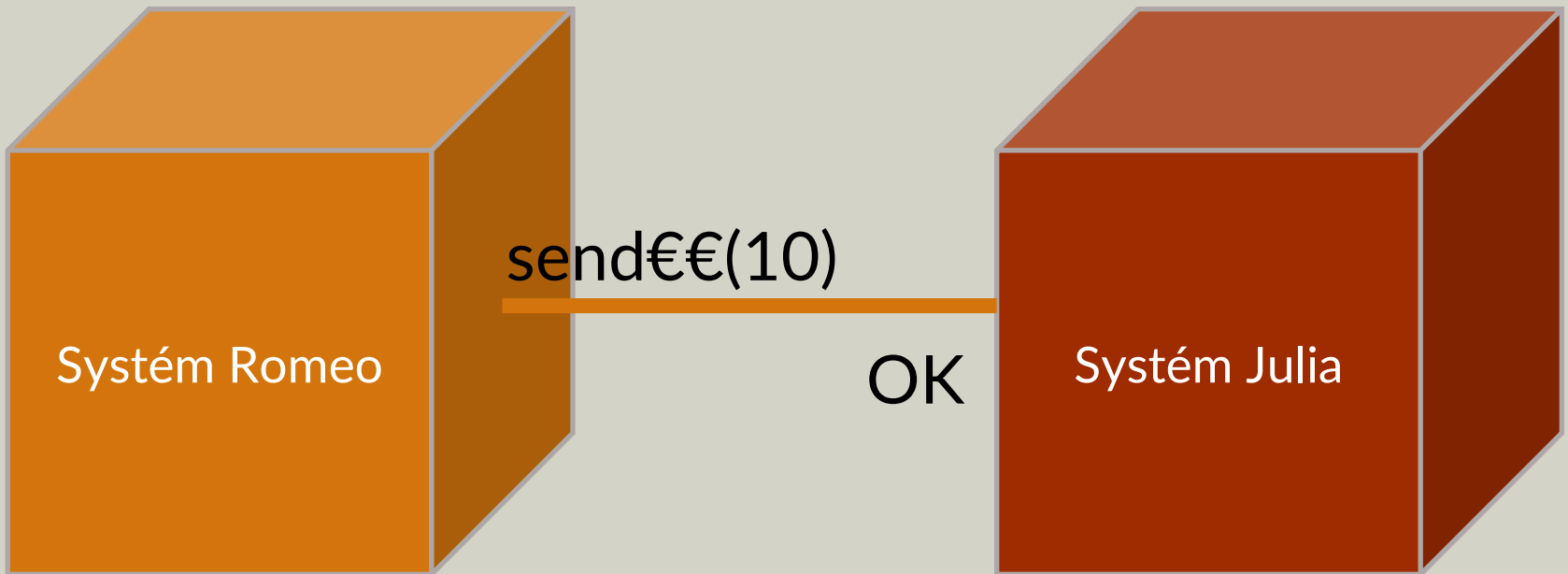
SOAPované distribuované systémy

Róbert Novotný
(UINF/KOPR)

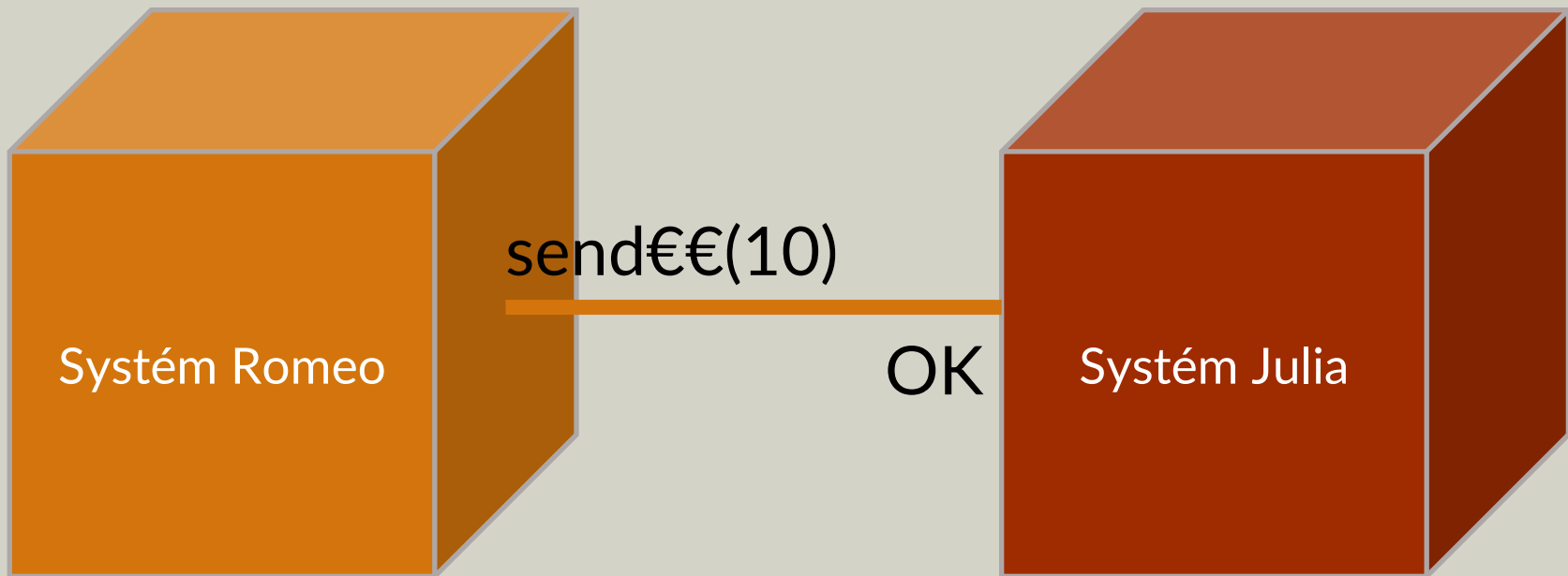
Stroj vs stroj



Remote Procedure Call



Remote Procedure Call



voláme procedúry, akoby boli v jednom programe

21. storočie v RPC

- REST
- gRPC (Google)
- Apache Avro
- WebSockets
- RSocket (Netflix)
- SOAP

Folklór

SOAP == webservisy

SOAP

- správa je vždy XML
- synchrónny protokol
 - obvykle cez HTTP
 - sloveso je HTTP POST
 - odíde požiadavka a vždy sa čaká na odpoveď
- jediná adresa pre klientov

SOAP Interop

- SOAP spolupracuje medzi jazykmi
- Java švitorí s .NET: špecifikácia WS-IT
- každý bežný jazyk má podprou

klient a server môžu byť v iných jazykoch!


```
POST http://www.websvcicex.net/stockquote.asmx
SOAPAction: "http://www.webserviceX.NET/GetQuote"
Host: www.websvcicex.net
Content-Type: text/xml;charset=UTF-8
```

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetQuote xmlns="http://www.webserviceX.NET/">
      <symbol>AAPL</symbol>
    </GetQuote>
  </soap:Body>
</soap:Envelope>
```

Request

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

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <GetQuoteResult xmlns="http://www.webserviceX.NET/">
      <Price>34.5</Price>
    </GetQuoteResult>
  </soap:Body>
</soap:Envelope>
```

Response

Prečo nie REST?

- RESTu v enterprise systémoch niet
- SOAP má stále
 - vymakaný tooling
 - vyladené možnosti, ktoré sa RESTu len plazia
 - kryptovanie správ, čo REST nemá

Prečo nie REST?

- REST = konvencie + JSON + OpenAPI
- SOAP = priemysel + XML + WSDL

Ako postaviť endpoint?

- vezmem HTTP server
- prijmem XML v požiadavke + naparsujem
- spočítam výsledok
- zlepím XML, pošlem do odpovede
- **PROFIT!**

Instantný SOAP service v Jave

- použijem štandard JAX-WS
- dostupné ako knižnica **Eclipse Metro**
- jedna trieda
- jeden spúšťač!



Vytvorme "Hello World"
endpoint v JAX-WS

Použime **SoapUI** ako klienta.

JAX-WS: operácie

@WebMethod / @WebParam

anotácia pre metódy/operácie

JAX-WS: operácie

@WebService

anotácia pre triedu s metódami

JAX-WS: dáta

JAXB anotácie

anotácie pre prispôsobenie parametrov a návratových hodnôt

@WebService

```
public class ChocolateService {  
    private List<Chocolate> chocolates  
        = new CopyOnWriteArrayList<Chocolate>( );  
  
    public ChocolateService() {  
        chocolates.add(new Chocolate( "Fasa 70%", 70 ));  
    }  
  
    public List<Chocolate> list() {  
        return chocolates;  
    }  
  
    public void add(Chocolate chocolate) {  
        chocolates.add(chocolate);  
    }  
}
```

všetky public metódy sa
zverejnia

```
Endpoint.publish("http://localhost:10000/ws/chocolates",  
                 new ChocolateService()  
                 );
```

zverejní SOAP
službu cez HTTP

navštívme URL browserom



DEMO

[JAX-WS]

Ako sa pripojiť?

- priamo posielať správy cez HTTP
 - odkiaľ poznáme tvar správ?
 - odkiaľ zoznam poskytovaných metód?
 -
- potrebujeme dokumentáciu!

WSDL


- metadátový popis webovej služby
 - operácie?
 - dátové typy?
 - spôsob prekladu do XML a späť?
- XML formát pre čítanie strojom
- z neho možno generovať **klientov!**

JAX-WS autogeneruje WSDL

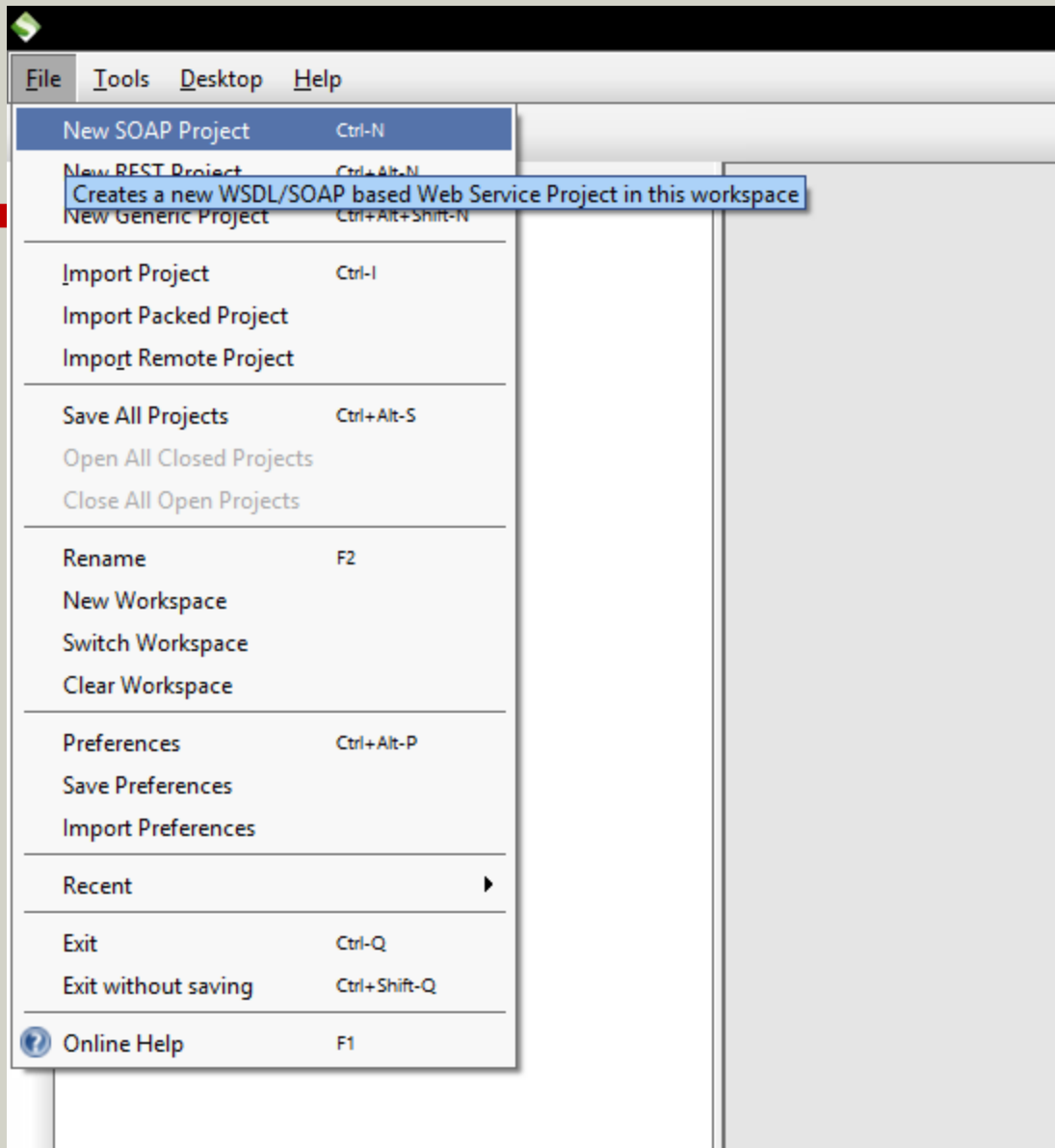
- JAX-WS server sprístupní WSDL na:

<http://localhost:10000/ws/chocolates?wsdl>

- automaticky generované
- určené pre stroje, nie pre ľudí!
- ale neskôr budeme vytvárať ručne



Vytvorte klienta pre službu v
JAX-WS pomocou SoapUI





New SOAP Project



New SOAP Project

Creates a WSDL/SOAP based Project in this workspace



Project Name:

Initial WSDL:

Create Requests: Create sample requests for all operations?

Create TestSuite: Creates a TestSuite for the imported WSDL

Relative Paths: Stores all file paths in project relatively to project file (requires save)



spustenie dopytu

The screenshot shows an IDE interface. On the left, a 'Projects' tree view displays a project named 'chocolates' with a sub-project 'ChocolateServicePortBinding'. Under this sub-project, there are two operations: 'add' and 'list', each with a 'Request 1' associated with it. An orange arrow points from the 'Request 1' under 'add' to the main viewer window. The main viewer window is titled 'Raw XML' and shows a SOAP request and response. The request is a SOAP envelope with a header and a body containing a 'jax:add' operation with optional parameters for 'percentage' and 'title'. The response is a SOAP envelope with a header and a body containing an 'ns2:addResponse' operation. The status bar at the bottom indicates 'response time: 19ms (181 bytes)' and '1:1'.

operácie a
ukážkové
dopyty

dopyt +
odpoved' zo
servera

Context menu options:

- Show Interface Viewer Enter
- Add JMS endpoint
- Generate Code
- Check WSI Compliance Ctrl+Alt-W
- Launch TcpMon
- Generate TestSuite
- Generate MockService
- Generate Documentation
- Update Definition F5**
- Export Definition
- Clone Interface F9
- Remove Delete
- Online Help F1

Tooltip for Update Definition: Reloads the definition for this interface and its operations

po zmene
API treba
aktualizovat'
klienta



Vygenerujte JAX-WS klienta


Java 11 a novšia?

- použite knižnicu **JAX-WS Metro 3** alebo novšiu
- použite **Maven plugin**

<https://bit.ly/3foWz6S>

Ako vytvoriť klienta v main()

- stvoríme továreň na **webservisy**
- továreň vyprodukuje **port**
 - webservis s operáciami
- na porte voláme **metódy** zodpovedajúce operáciám



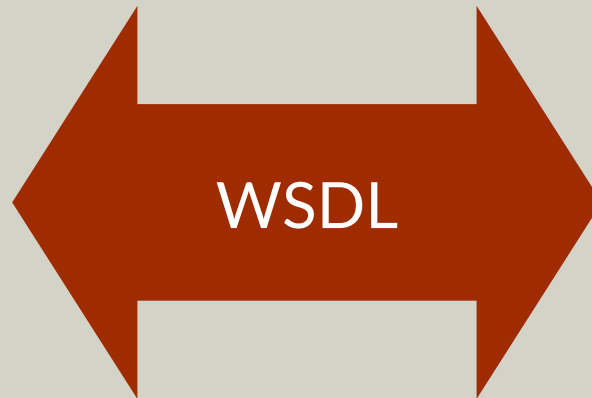
```
ChocolateServiceService service = new ChocolateServiceService();  
    IChocolateService port = getChocolateServicePort();  
    List<Chocolate> chocolates = port.list();
```



port je nerozoznateľný od servisnej implementácie na serveri

JAX-WS

automaticky
vygenerovaný
popisovač
WSDL



vygenerovaný
z WSDL cez

- SoapUI
- Maven
- wsimport

SOAP Server

SOAP Client

Sumár

- vieme vytvoriť JAX-WS server i klienta
- Java<->Java
- čo však s interoperabilitou
 - medzi jazykmi a platformami?
- čo však so stabilitou API?

garantované riešenie: ručné WSDL



Od WSDL ku kódu

Metadáta webovej služby: WSDL

operácie

metódy

dátové typy

vstupy a výstupy operácií

štýly

prevod dát na správy

bindings

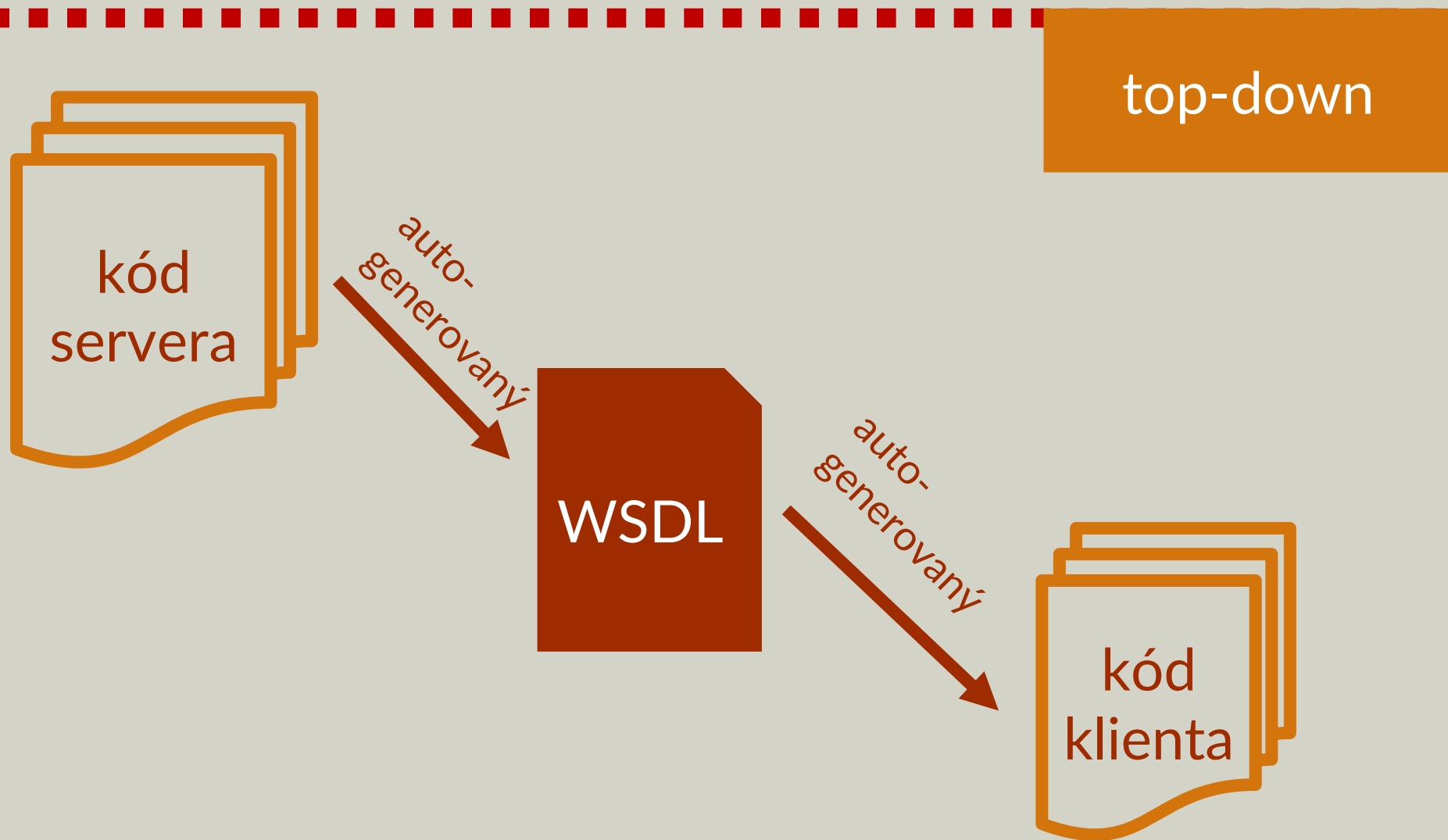
sieťové protokoly

Metadáta webovej služby: WSDL

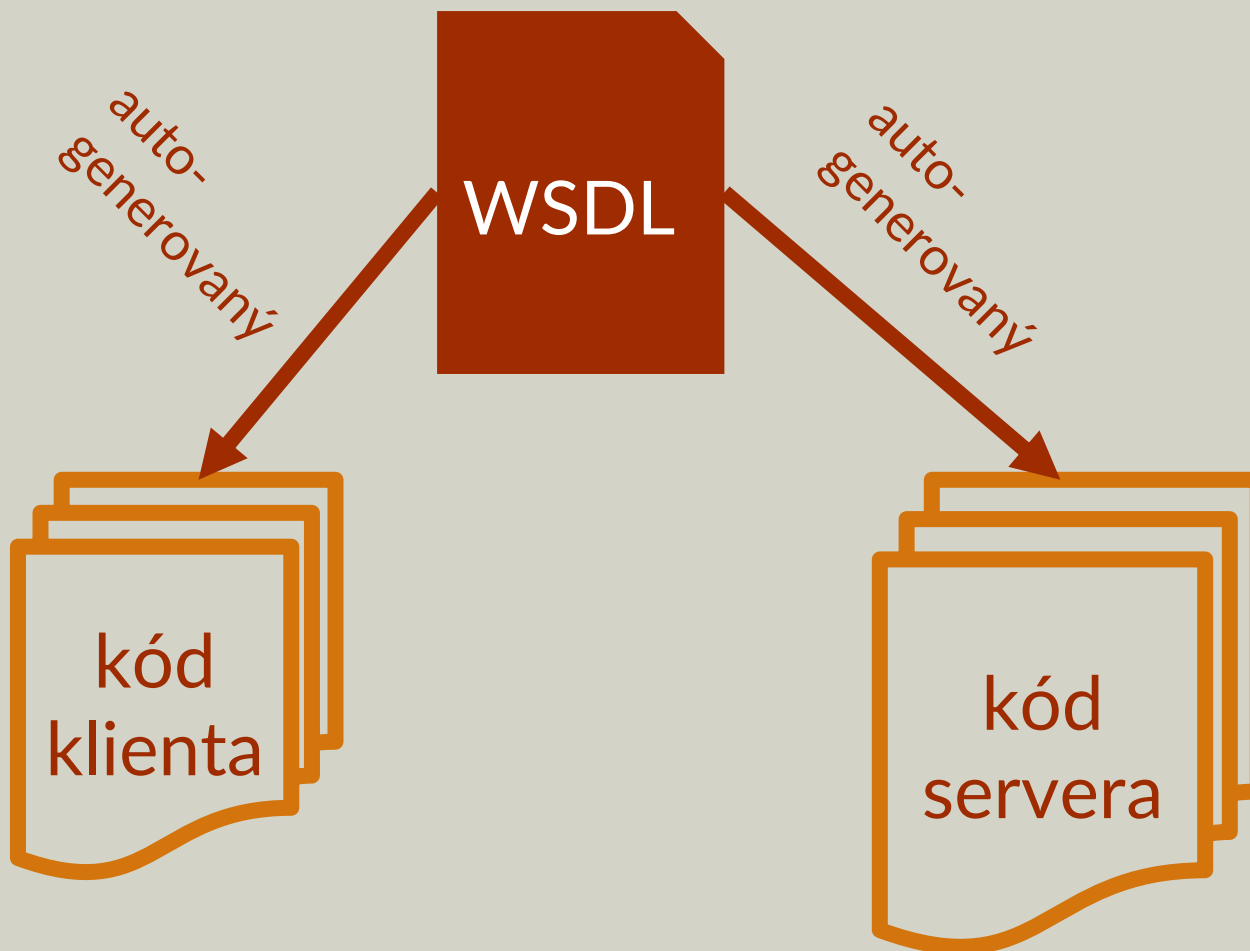
- XML formát pre čítanie strojom
- strojová a typovo exaktná dokumentácia k službe
- prameň pre generovanie klientov!

Web Services Description Language

Dizajn služby: contract-last



Dizajn služby: top-down



Contract-last! Bottom-up

- uvažujeme v objektoch konkrétneho programovacieho jazyka
- expresný bootstrap!
 - ale: ohrozený interop!
 - ale: nestabilné API => nestabilné WSDL => stále opravovaní klienti

Contract-first! Top Down!

- pri tabuli dohodneme
 - operácie
 - štruktúru dát:
 - XML pre vstupné správy
 - XML pre výstupné správy

WSDL je platformovo nezávislé

XML schéma: jazyk pre formát správ

- aké sú povolené XML správy?
- aké elementy/atribúty majú?
- v akej hierarchii?
- aké dátové typy?

XSD: definícia tvaru správ

XML schéma: jazyk pre formát správ

- pomocou schémy vieme validovať správy!
- pomocou schémy vieme mapovať schému na triedu!
 - automatický prevod XML na objekty a späť

XSD: definícia tvaru správ

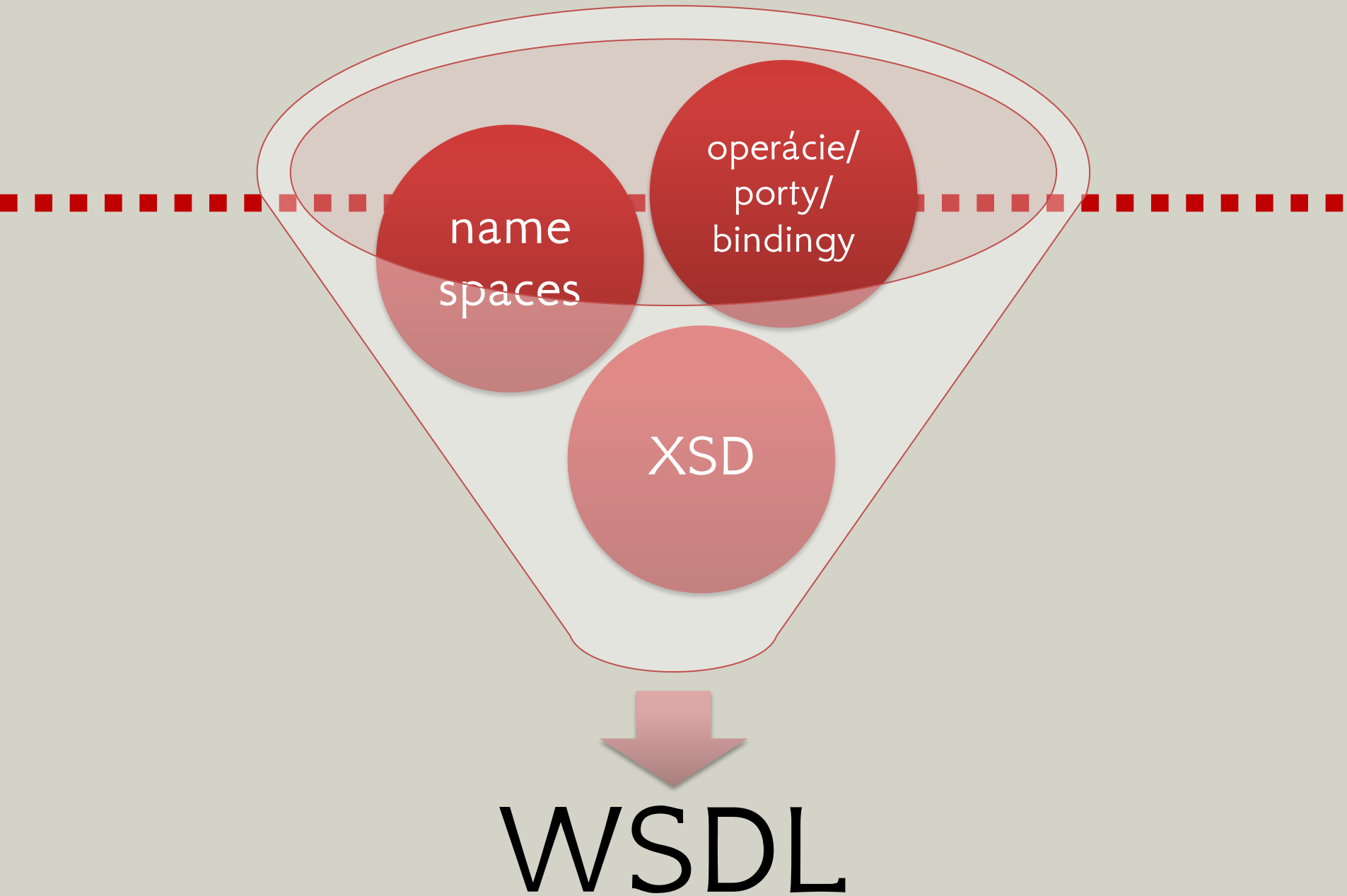
část XSD schémy

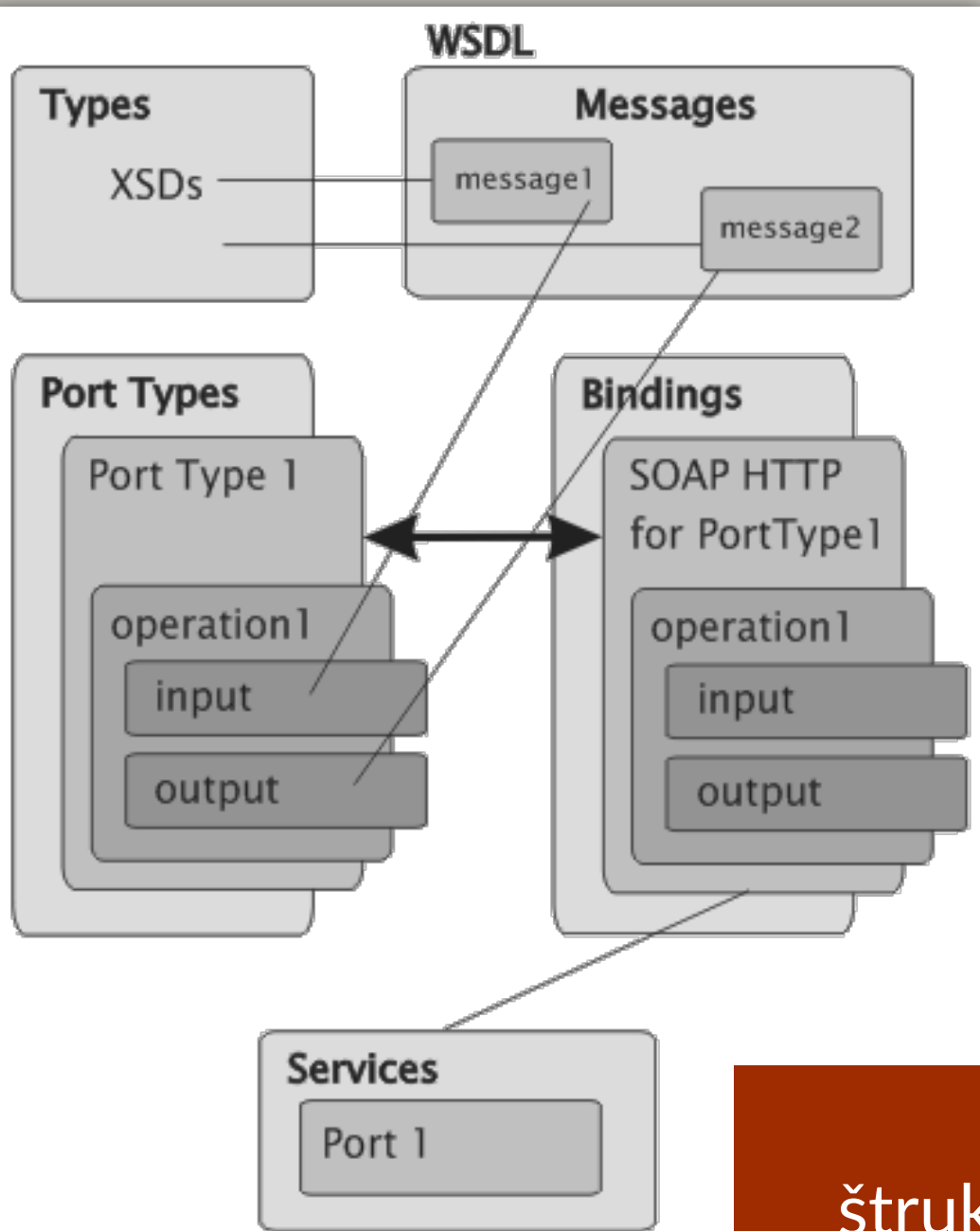
```
<xs:complexType name="chocolate">
  <xs:sequence>
    <xs:element name="id"
      type="xs:long"
      minOccurs="0" />
    <xs:element name="title" type="xs:string" />
  </xs:sequence>
</xs:complexType>
```

```
public class Chocolate {
  private long id;
  private String title;
}
```



Vytvorte WSDL pre službu a
vygenerujte implementácie.





štruktúra WSDL

Ruční tvorba WSDL



<https://bit.ly/3sGihWU>

GUI pre tvorbu WSDL

Eclipse IDE



Eclipse Java Web
Developer Tools

JAX-WS Tools

Klikacie editory pre

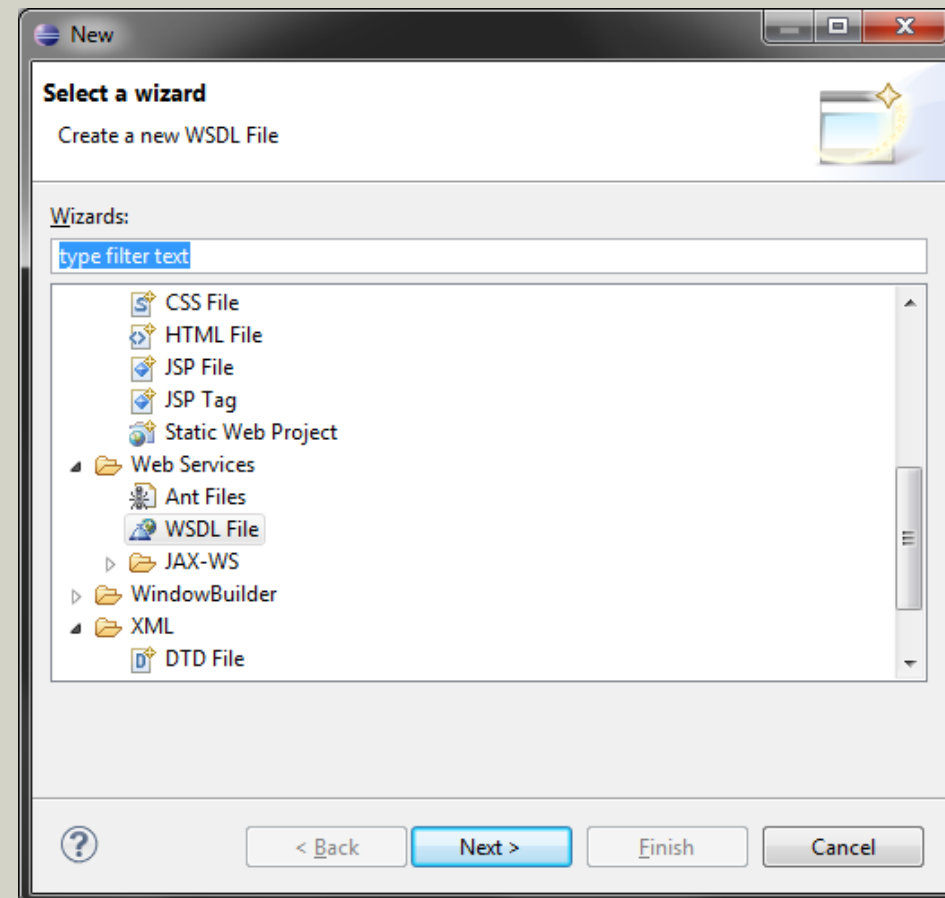
- WSDL
- XML Schéma XSDL

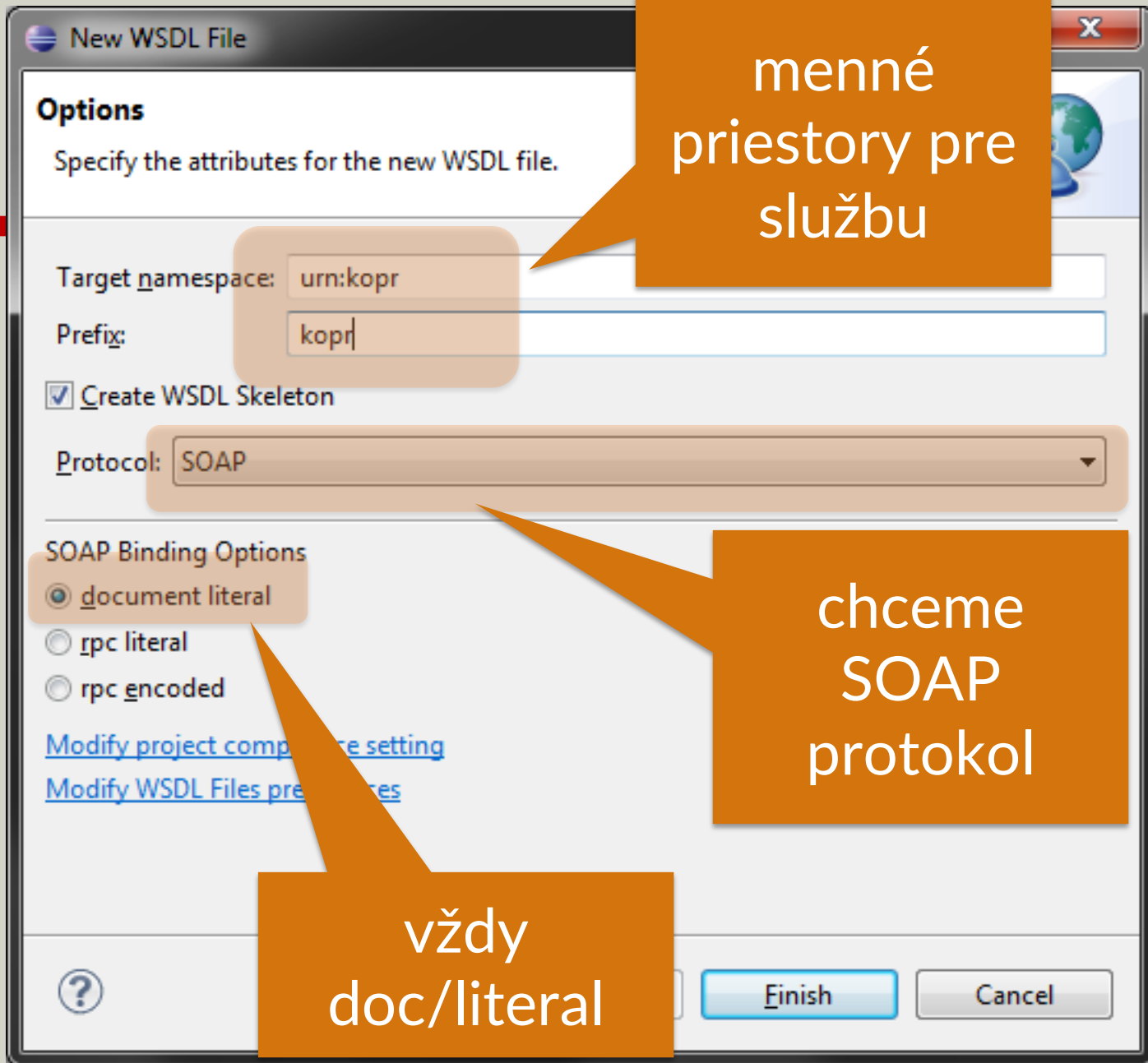
File

| New

| Web-Services

| WSDL File





menné priestory pre službu

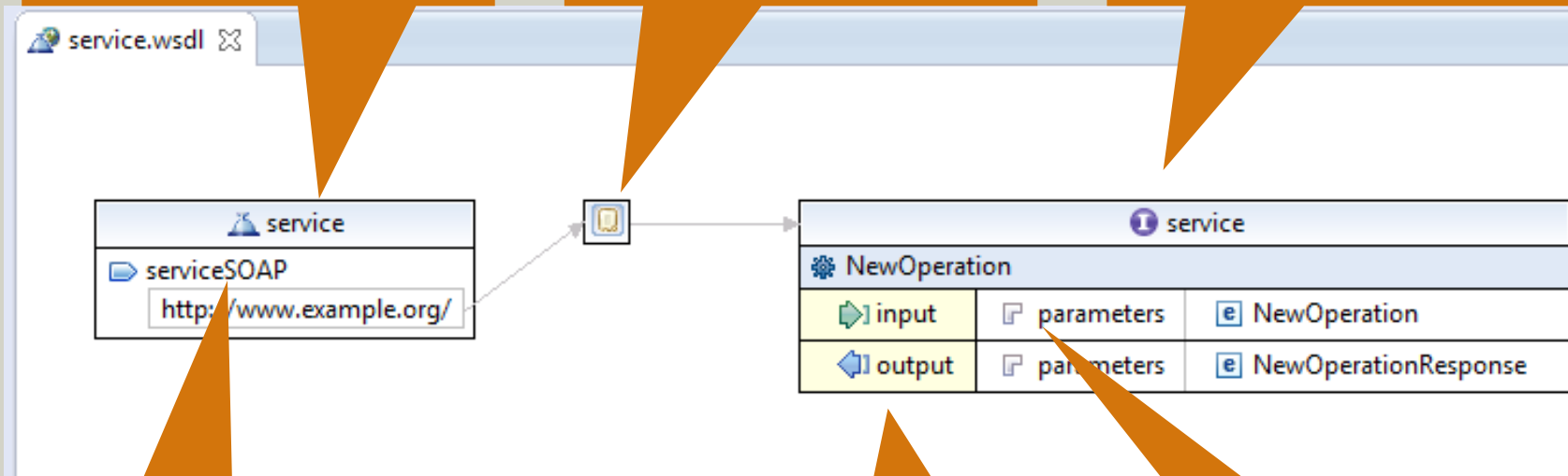
chceme SOAP protokol

vždy doc/literal

service: sada portov

binding: definícia nasadenia

definícia portu



port ~ Java interface

operácia: metóda

parametre + návratové hodnoty

Tvorba WSDL: screencast



<http://ics.upjs.sk/~novotnyr/home/skola/konkurentne-programovanie/2013/wSDL.htm>

WSDL -> Server

- z WSDL vygenerujeme kostru pre JAX-WS server
- proces generovania rovnaký ako pri klientovi
 - Maven Plugin
- doimplementujeme interfejs s telami metód

odkaz na interfejs
s metadátami

```
@WebService(endpointInterface = "org.example.ChocolateServicePortType" )  
public class DefaultChocolateService  
    implements ChocolateServicePortType  
{  
    ...  
}
```

implementujeme
metódy

JAX-WS Server: Tutoriál

<https://bit.ly/3sGR2vd>

WSDL -> klient

- z WSDL vygenerujeme JAX-WS klienta
- videli sme už minule

Maven Plugin pre Metro