A4M36AOS – Architektury orientované na služby

4. Business Process Modeling

Jiří Vokřínek

Agent Technology Center
Department of Computer Science and Engineering
Faculty of Electrical Engineering, Czech Technical University in Prague

jiri.vokrinek@fel.cvut.cz http://agents.felk.cvut.cz

Business Process Execution Language (BPEL)

- Web Service composition language
- Used for web service orchestration
- BPEL was originally developed by BEA, IBM, and Microsoft. Version 1.1 also includes input from SAP and Siebel.
- The OASIS TC "Web Services Business Process Execution Language" now continues the standardization of BPEL

BPEL

- BPEL4WS 1.0 (7/2002)
  - Original proposal from BEA, IBM, Microsoft
  - Combined ideas from IBM's WSFL and Microsoft's XLANG
- BPEL4WS 1.1 (5/2003)
  - Revised proposal submitted to OASIS
  - With additional contributions from SAP and Siebel

BPEL

- WS-BPEL 2.0 (6/2007)
  - Formalization of 1.1 capabilities
  - OASIS formally adopted standard
- WS-BPEL 2.0 and beyond (10/2007)
  - Additional proposals on the table
  - Vendors beginning to ship products conforming to standards

BPEL

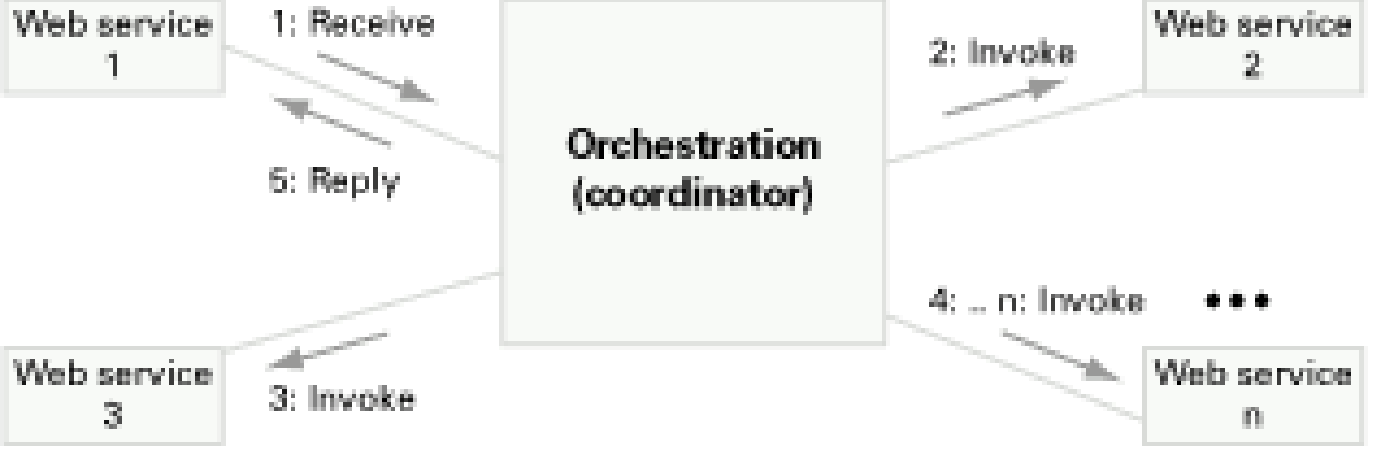

Is this your
"new age" IT
organization?
BPEL
business
process
modeling

BPEL


Is this your
"new age" IT
organizatio
BPEL
?
business
process
modeling

BPEL

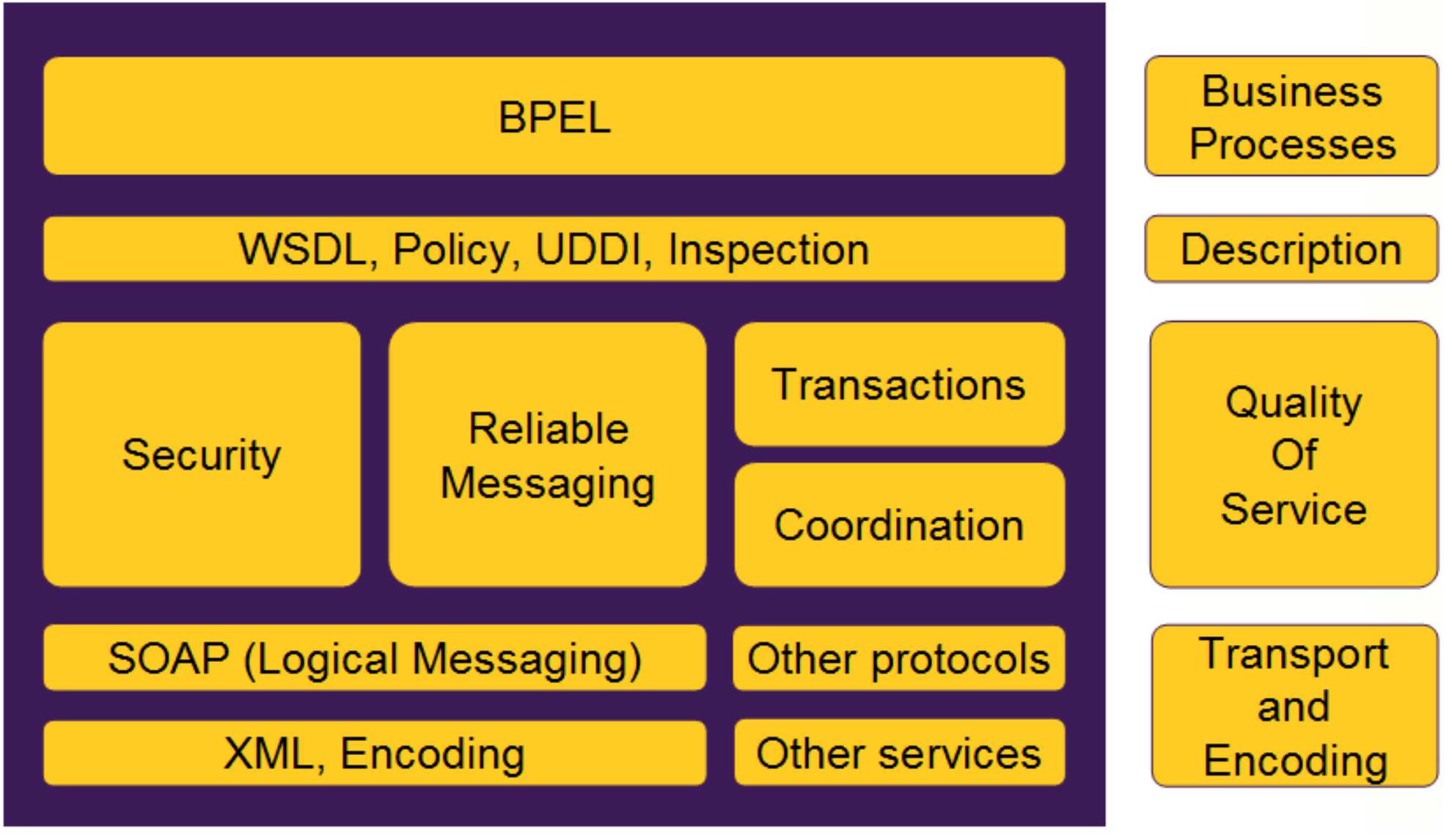
- Defines business processes that interact with external entities through Web services
- The definitions use XML and are not concerned with the graphical representation of processes
- Defines a set of Web service orchestration concepts



BPEL

- Supports the implicit creation and termination of process instances as the basic lifecycle mechanism
- Defines a long-running transaction model to support failure recovery
- Uses Web services as the model for process decomposition and assembly
- Builds on compatible Web services standards

BPEL


BPEL
WSDL, Policy, UDDI, Inspection
Security
Reliable Messaging
Transactions
Coordination
SOAP (Logical Messaging)
Other protocols
XML, Encoding
Other services
Business Processes
Description
Quality Of Service
Transport and Encoding

BPEL and WSDL

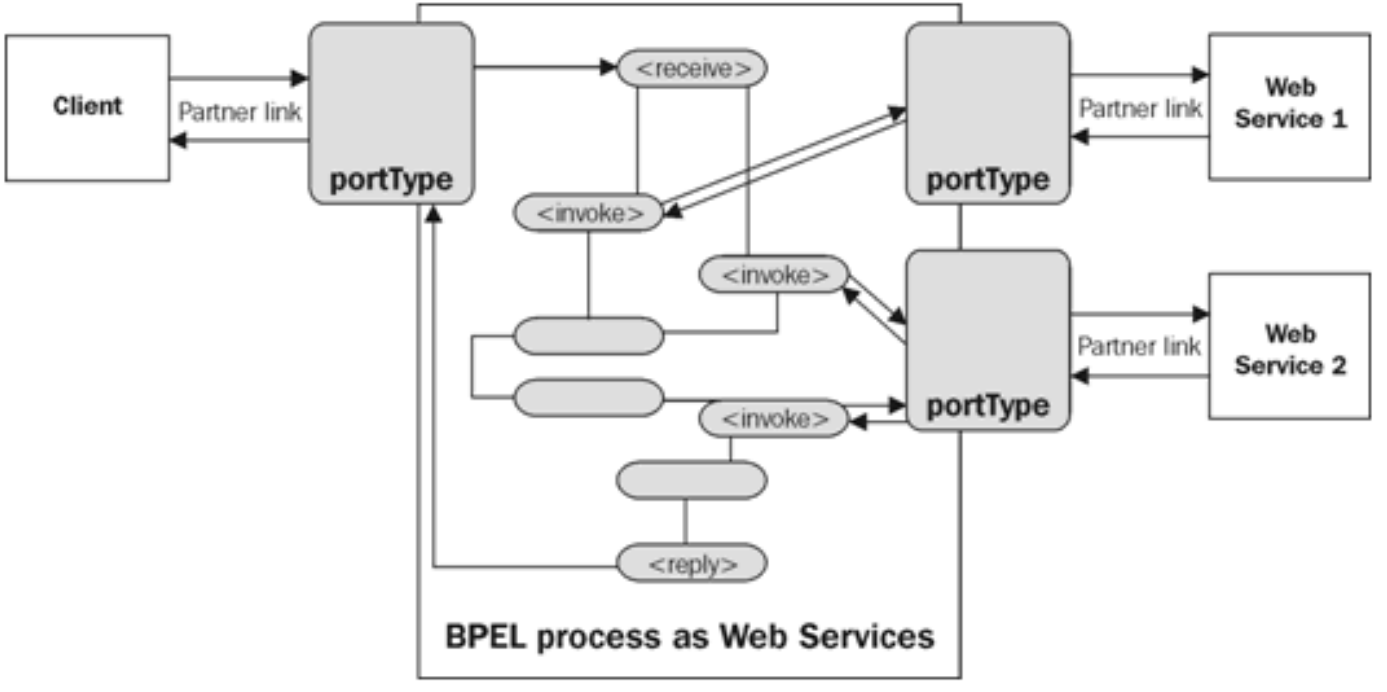
- BPEL processes exposed as WSDL services
- Message exchanges map to WSDL operations
- WSDL can be derived from partner definitions and the role played by the process in interactions with partners
- Interfaces exposed by the BPEL process
- Interfaces consumed by the BPEL process

BPEL and WSDL

- BPEL uses Web Services (BPEL is orchestrating these services)
- The BPEL process itself is a Web Service (it has interfaces) – when defining a BPEL process, it also is described by a WSDL (its interface)
- WSDL Port Types are named sets of abstract operations
- WSDL extensions are used to identify which port types are used to link services

BPEL as Process

- Most BPEL applications are executable processes
- Describes the interfaces to external data sources
- Describes the control flow for orchestrating these data sources



BPEL Partners

- BPEL supports different relationships with partners
  - Partners may invoke the BPEL process
  - BPEL process may invoke partners
  - Partners and the BPEL process play both roles
- BPEL processes will have at least one client (the partner activating the process)

BPEL as Language

- Business process modeling language that is executable
- Language for specifying business process behavior based on Web Services
- Serialized in XML and aims to enable programming in the large (generally refers to the high-level state transition interactions of a process)
- No standardized graphical notation for BPEL – XML is used as the standardized syntax

BPEL as Language

- BPEL processes can be executed and thus are programs
- BPEL is a specialized and dedicated programming language
- BPEL combines two tasks
  - Creates a new Web Service which is described by a WSDL interface
  - Implements the Web Service by orchestrating a number of partners

BPEL as Language

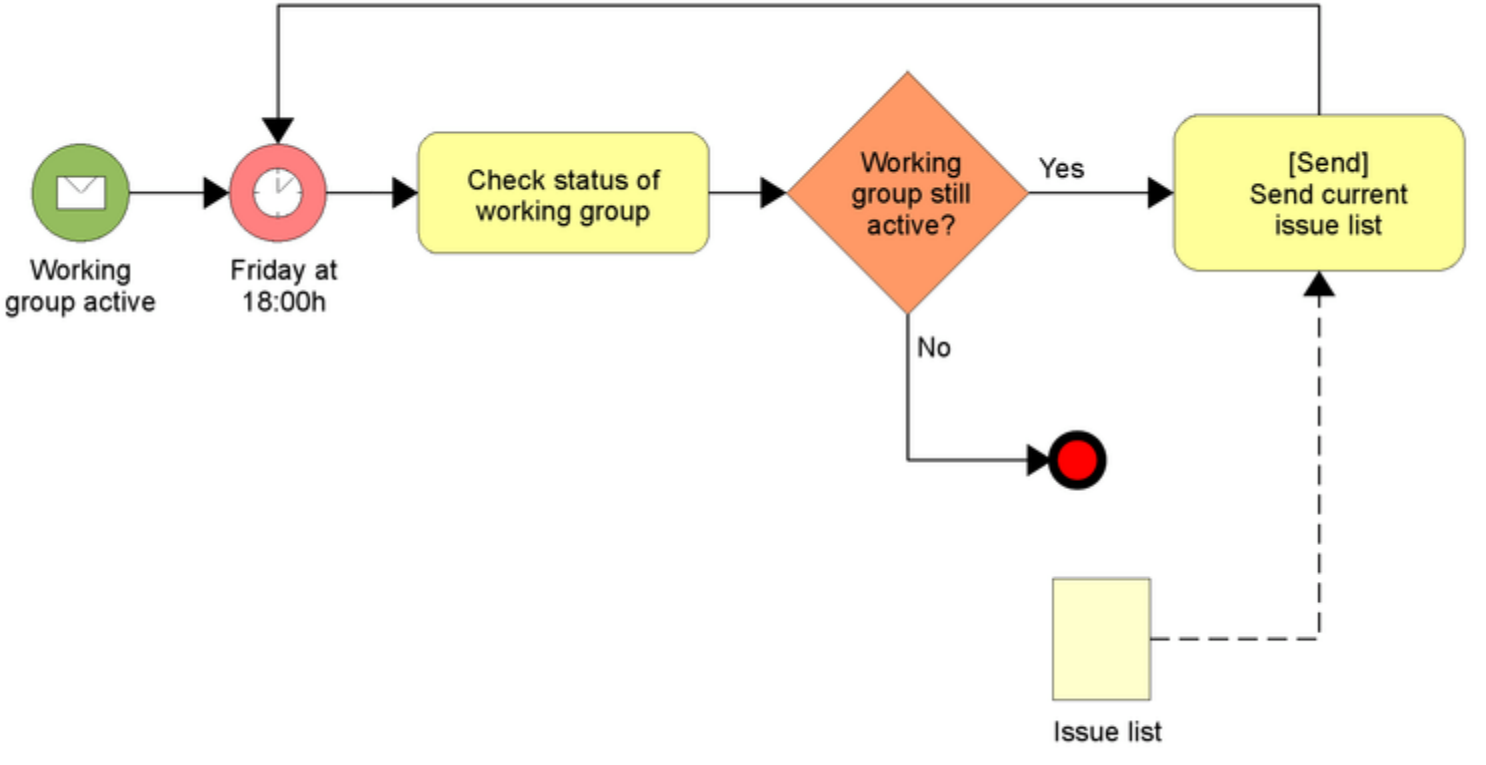
```
<?xml version = "1.0" encoding = "UTF-8" ?>
<process name="BPELDynamicPL"...>
  -->
  <partnerLinks.../>
  <variables>
    <variable name="Receive_File_Get_InputVariable" messageType="ns1:Get_msg"/>
    <variable name="Invoke_FTPServer1_Put_InputVariable" messageType="ns2:Put_msg"/>
    <variable name="jndiLocation" type="xsd:string"/>
  </variables>
  <sequence name="main">
    <receive .../>
    <assign name="Assign_Payload".../>
    <assign name="Assign_FtpServer1_JNDI">
      <copy>
        <from expression="'eis/Ftp/FtpAdapter1'"/>
        <to variable="jndiLocation"/>
      </copy>
    </assign>
    <invoke name="Invoke_FTPServer1"
            inputVariable="Invoke_FTPServer1_Put_InputVariable"
            partnerLink="FTPOut" portType="ns2:Put_ptt" operation="Put">
      <bpelx:inputProperty name="jca.jndi" variable="jndiLocation"/>
    </invoke>

    <assign name="Assign_FtpServer2_JNDI">
      <copy>
        <from expression="'eis/Ftp/FtpAdapter2'"/>
        <to variable="jndiLocation"/>
      </copy>
    </assign>
    <invoke name="Invoke_FTPServer2"
            inputVariable="Invoke_FTPServer1_Put_InputVariable"
            partnerLink="FTPOut" portType="ns2:Put_ptt" operation="Put">
      <bpelx:inputProperty name="jca.jndi" variable="jndiLocation"/>
    </invoke>

  </sequence>
</process>
```

Business Process Modeling Notation (BPMN)

- Graphical representation for specifying business processes in a business process (only) modeling



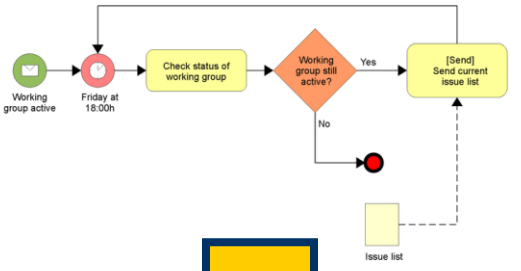
BPMN

- Based on a flowcharting technique very similar to activity diagrams from UML
- Intuitive notation to business users yet able to represent complex process semantics
- Provides a mapping between the graphics of the notation to the underlying constructs of execution languages (BPEL)

BPMN

BPEL

Application



?

```
<?xml version = "1.0
<process name="BPELDynami
  -->
  <partnerLinks.../>
  <variables>
    <variable name="Receive_File_Get_InputVariable" messageType="ns1:Get_msg"/>
    <variable name="Invoke_FTPServer1_Put_InputVariable" messageType="ns2:Put_msg"/>
    <variable name="jndiLocation" type="xsd:string"/>
  </variables>
  <sequence name="main">
    <receive .../>
    <assign name="Assign_Payload".../>
    <assign name="Assign_FtpServer1_JNDI">
      <copy>
        <from expression="'eis/Ftp/FtpAdapter1'"/>
        <to variable="jndiLocation"/>
      </copy>
    </assign>
    <invoke name="Invoke_FTPServer1"
            inputVariable="Invoke_FTPServer1_Put_InputVariable"
            partnerLink="FTPOut" portType="ns2:Put_ptt" operation="Put">
      <bpelx:inputProperty name="jca.jndi" variable="jndiLocation"/>
    </invoke>

    <assign name="Assign_FtpServer2_JNDI">
      <copy>
        <from expression="'eis/Ftp/FtpAdapter2'"/>
        <to variable="jndiLocation"/>
      </copy>
    </assign>
    <invoke name="Invoke_FTP
            inputVariable="I                    _InputVariable"
            partnerLink="FTP                    ut_ptt" operation="Put">
      <bpelx:inputProperty n                    able="jndiLocation"/>
    </invoke>

  </sequence>
</process>
```

?

BPMN

- Examples

http://diveintobpm.org

BPMN

- Process example

from A4B33SI tutorials by Michal Čáp