9. SOA and Business Processes

Distributed Systems
Prof. Dr. Alexander Schill
http://www.rn.inf.tu-dresden.de
Outline

- **Services**
  - Service Definition
  - Service-oriented Architectures

- **Business Processes and Workflows**
  - Business Process Modeling with Petri-Nets
  - BPMN (Business Process Model and Notation)
  - BPEL (Business Process Execution Language)
  - Other approaches
Components provide:
- Separation of code and configuration
- Simplified programming interfaces for the access of system services
- Specific tools for component based software development
- Component model and platform: Enterprise JavaBeans, OSGi and Microsoft .NET-Components

But:
- Mainly use of homogeneous component model and platform
- Usually bound to one programming language
- Close coupling of components within intranets of companies
  - Limited flexibility for changing business processes
  - Issues with firewalls
  - Issues with integrating business of different companies in a global way
  - Issues with heterogeneous enterprise infrastructures
A service is a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description.

[OASIS reference model]

- Enabling of access to capabilities
- Service description defining constraints and policies
- Functionality provided by formally specified interface
  - Encapsulated functionality
  - Implementation transparency
- Loose coupling

What is a service?

.NET  
service user  

OSGi  

EJB  

service implementation

service description
Service Oriented Architectures

- Focus on business processes
- Main functionality of applications provided by services
- Services have higher granularity than objects and components
- Encapsulate part of business process functionality
- Loose coupling
  - Services are offered, searched for and used
  - Services may be arbitrarily distributed
  - Services are dynamically composed to business processes
  - Interoperability over platform and business borders
Example business process

- Business Process: is a “collection of activities performed by human users or software applications that together constitute the different steps to be completed to achieve a particular business objective”. [Alonso et. al]

- Workflow: “The terms workflow, workflow process, or sometimes simply process refer to a formal, executable description of a business process.” [Alonso et. al]
- Modeling and decomposition of business process into sub-processes
- Mapping to service infrastructure required to be executable
- Creation of new services to provide application logic required for business process
Business Process Management

Business Process Layer: Order Handling

- Purchasing
- Customer Management
- Order Processing
- Payment and Invoicing
- Shipping

Business Process Modeling

- Purchase verification
- Customer management (parallel)
- Risk analysis
- Calculate discount
- Payment
- Invoicing
- Shipping

Application Logic

- Web forms
- CRM
- Order Processing
- Payment Processing
- Shipping and Invoicing
- Warehouse Management

Business process perspective

Application logic perspective
Business process modeling

- **Formal notion based on Petri-Nets**
  - Introduced in 1962 by mathematician Carl Adam Petri
  - Formal mathematical model are directed graphs
- **Basic concepts:**
  - **Places** represent states of workflow execution
    - E.g. results, documents or data buffers
  - **Transitions** model activities of a workflow
    - Representation of task or basic services to process a workflow
    - Conditions can define input predicates for transitions
  - **Arcs (Links)** connect places and transitions
    - Representation of route or flow of tasks
  - **Markers** model the flow at a certain point in time
    - Places can have markers, capacity of place defines max. number of markers of one place

![Diagram of Petri-Nets](image)
Properties:
- limited \((\text{Marker Quantity}, \text{for instance, as here} \leq 2)\)
- non-alive (terminating)
- secure \((\text{Marker Quantity} \text{ of each place / state is 0 or 1})\)
- non-conservative \((\text{Marker Quantity} \text{ varies})\)
Languages for Business process modeling

- **Business Process Model and Notation (BPMN)**
  - Based on flow diagrams
  - Mapping from BPMN to BPEL possible
  - Widespread adoption in industry

- **Petri-net based modelling languages**
  - Difficult to understand in practice

- **Activity-diagrams in UML**
  - (Unified Modeling Language)
  - Limitations for the modeling of complex processes
Business Process Modeling Notation (BPMN)

- Standard of the Object Management Group
- Modeling of activities which are performed at certain points in time

Basic Elements
- Flow elements
  - Event, Activity, Gateway
- Connection Objects
  - Sequence Flow, Message Flow, Association
- Swimlanes
  - Pools and Lanes
- Artefacts
  - Data objects, Group, Annotation

- Further elements for expressing timeouts, error handling and transactions
• **Step 1: Model Business Process**
  - Graphical representation

![BPMN Diagram](image-url)

- **XML representation**

```xml
<process id="financialReport" name="Monthly financial report reminder process">
  <startEvent id="theStart" />
  <sequenceFlow id='flow1' sourceRef='theStart' targetRef='writeReportTask' />
  <userTask id="writeReportTask" name="Write monthly financial report" >
    <documentation>
      Write monthly financial report for publication to shareholders.
    </documentation>
    <potentialOwner>
      <resourceAssignmentExpression>
        <formalExpression>accountancy</formalExpression>
      </resourceAssignmentExpression>
    </potentialOwner>
  </userTask>
  ......
</process>
```

Distributed Systems – Lecture 9: Services and Platforms
BPMN Example: Activity

- Step 2: Deploy Process to BPMN-Engine &
- Step 3: Start Process
  - Via API of engine:
    ```java
    Deployment deployment = repositoryService.createDeployment()
        .deploy();
    ProcessInstance processInstance = runtimeService.startProcessInstanceByKey("financialReport");
    ```
  - Via Process Engine Interface:

![Activiti Explorer](image)
- **Business Process Execution Language**: XML-based Notation for Business Process Description and Composition of Web Services
- **Two kinds of business processes definable**
  - **Abstract Process**
    - Specification of the message exchanges between multiple participants
    - No definition of internal process details
    - Interfaces defined through set of receive and reply ports
    - Definition of a Choreography ("bottom-up")
  - **Executable Process**
    - Orchestration of specific activities and particular services which must be executed
    - Executable via a specific execution engine
    - Definition of an Orchestration ("top-down")
Choreography vs. Orchestration

**Choreography**

1. Invoke
2. Invoke
3. Reply
4. Invoke
5. Invoke

**Orchestration**

1. Invoke
2. Invoke
3. Invoke
4. ... n: Invoke
5. Reply

Orchestration (Web Service)
BPEL: Structure

- **BPEL Process** describes the business process, subdivided into **activities**, connected by **partner links**
- **Activities** map to **web services** with WSDL interfaces via **invoke interface**, augmented by **receive** and **reply** interfaces for response messages
- Parallel execution controlled via **flow elements**
- Selective execution via **switch elements** with different **case** alternatives
BPEL: Example Process

**Purchasing**
- **portType:** purchaseOrderPT
  - **Operation:** sendPO
- **reply**
- **receive**

**BPEL-Process**
- **invoke**
- **receive**

**Shipping**
- **portType:** shippingPT
  - **Operation:** requestShipping
- **invoke**
- **receive**
- **invoke**
- **receive**

**Invoicing**
- **portType:** computePricePT
  - **Operation:** initiatePriceCalc
- **invoke**
- **receive**
- **receive**

**Partner Link**
- Partner Link
- Partner Link
BPEL: Example of a Process Step

- BPEL File for Purchase Order Process

```xml
<process name="purchaseOrderProcess"...>
  <partnerLinks>
    ...
    <partnerLink name="invoicing" partnerLinkType="lns:invoicingLT"
      myRole="invoiceRequester" partnerRole="invoiceService"/>
  </partnerLinks>
  ...
  <sequence>
    ...
    <invoke partnerLink="invoicing"...
      operation="initiatePriceCalc">...<invoke>
    <receive partnerLink="invoicing"...
      operation="sendInvoice">...<receive>
  </sequence>
</process>

- Source: http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpel-v2.0-CS01.html

WSDL File for Invoicing WS

...<plnk:partnerLinkType name="invoicingLT">...
...
The entire set will be enveloped in a BPEL sequence.

Exclusive Gateway can be mapped to BPEL switch.

Parallel Gateway maps to BPEL flow.

A task maps to a BPEL invoke.

Gateway alternative maps to BPEL case in switch.
Other Approaches for Service Coupling

- **Enterprise Service Bus (ESB)**
  - based on message broker concepts
  - no standardization

- **Java Business Integration (JBI)**
  - specified as JSR (Java Specification Request)
  - integration standard for Java-based world
Enterprise Service Bus: Example

Order Processing
JEE, .NET

Legacy Applications
Customer Administration
Middleware

Service Flow Specification
(e.g. based on BPMN)

ESB – Message Broker
Common Runtime Environment

SOAP
Warehouse Management

JMS
B2B Integration
Payment Service

SOAP
Shipping
Enterprise Service Bus: Summary

- Advantages:
  - Loosely coupled, scalable and reliable architecture
  - Support of routing, mediation and complex message patterns

- Disadvantages:
  - No standardisation of features, protocols and messages
    ⇒ Products of different vendors are not compatible
  - Message broker concept adds additional abstraction layer

- Implementations:
  - Open: Open ESB, Apache Service Mix, Apache Tuscany
  - Commercial products by IBM, Oracle, Microsoft and others
JBI (Java Business Integration) Concept:

- **Service Engines**: extensible business logic (e.g., EJB-Wrapper)
- **Binding-Components**: Proxy for service users and remote services
  \[\Rightarrow\] access independent from transport protocol
- **Normalized Message Router** (Standardized ESB): exchange of (normalized) messages
  \[\Rightarrow\] interoperability between components
- **JBI runtime environment**
Java Business Integration: Summary

- **Advantages:**
  - Open standard for integrating applications
  - Flexibility through plug-in architecture, refinement of ESB for Java

- **Disadvantages:**
  - Limited to Java, official specification withdrawn
  - Lacking support by major players like IBM and Oracle

- **Implementations:**
  - Open ESB, Apache Service Mix
Summary

- Service-oriented architectures focus on the creation of business processes
  - Services encapsulate application logic
  - Business processes are implemented with service compositions

- Business processes can involve many processing steps which overlap departments/companies and therefore involve various different applications/services located on different servers

- Language support available for modelling business processes
- Mapping from business oriented description (e.g. BPMN) to executable process model (e.g. BPEL)

- Various alternative approaches available such as ESB and JBI