

Camunda BPM for Developers

Getting started with the Camunda BPM Platform



2

Agenda

Day 1

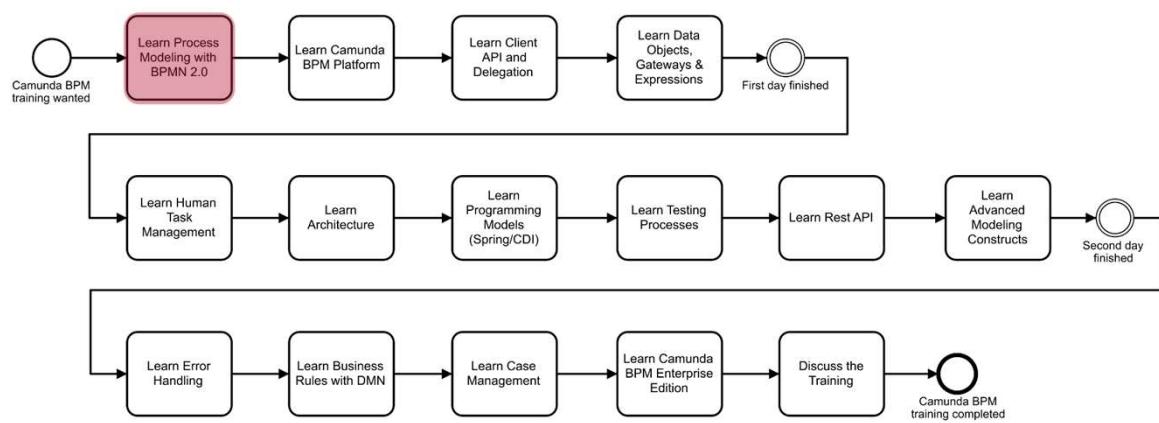
1. Process Modelling with BPMN 2.0
2. Camunda BPM Platform
3. Client-API and Delegation
4. Data Objects, Gateways & Expressions

Day 2

5. Human Task Management
6. Architecture
7. Programming Models (Spring/CDI)
8. Testing Processes
9. REST API
10. Advanced BPMN Constructs

Day 3

11. Error Handling
12. Business Rules with DMN
13. Case Management
14. Camunda BPM Enterprise Edition
15. Wrap up: Outlook, Feedback and Open Discussion



4

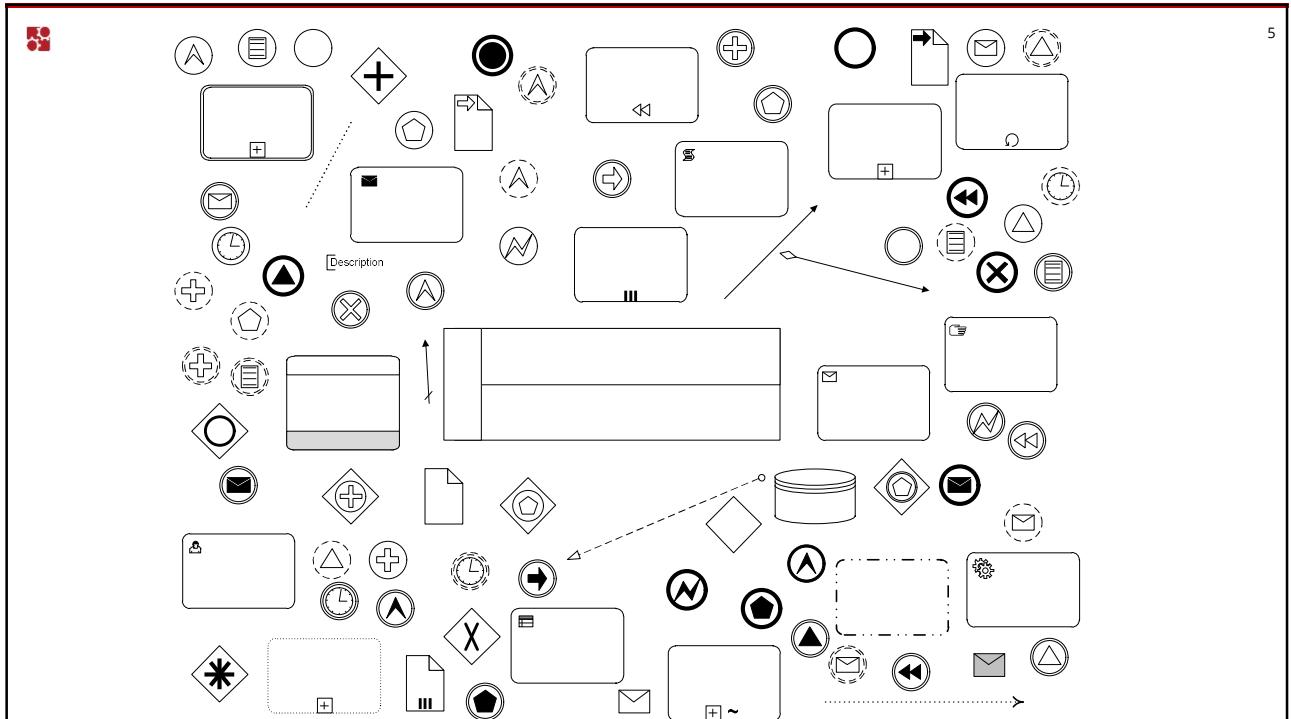
Business Process Model and Notation (BPMN)

- BPMN is a worldwide OMG standard
- Latest version is BPMN 2.0
- Lots of companies joined the standardization committee
- All major BPMS vendors go for BPMN

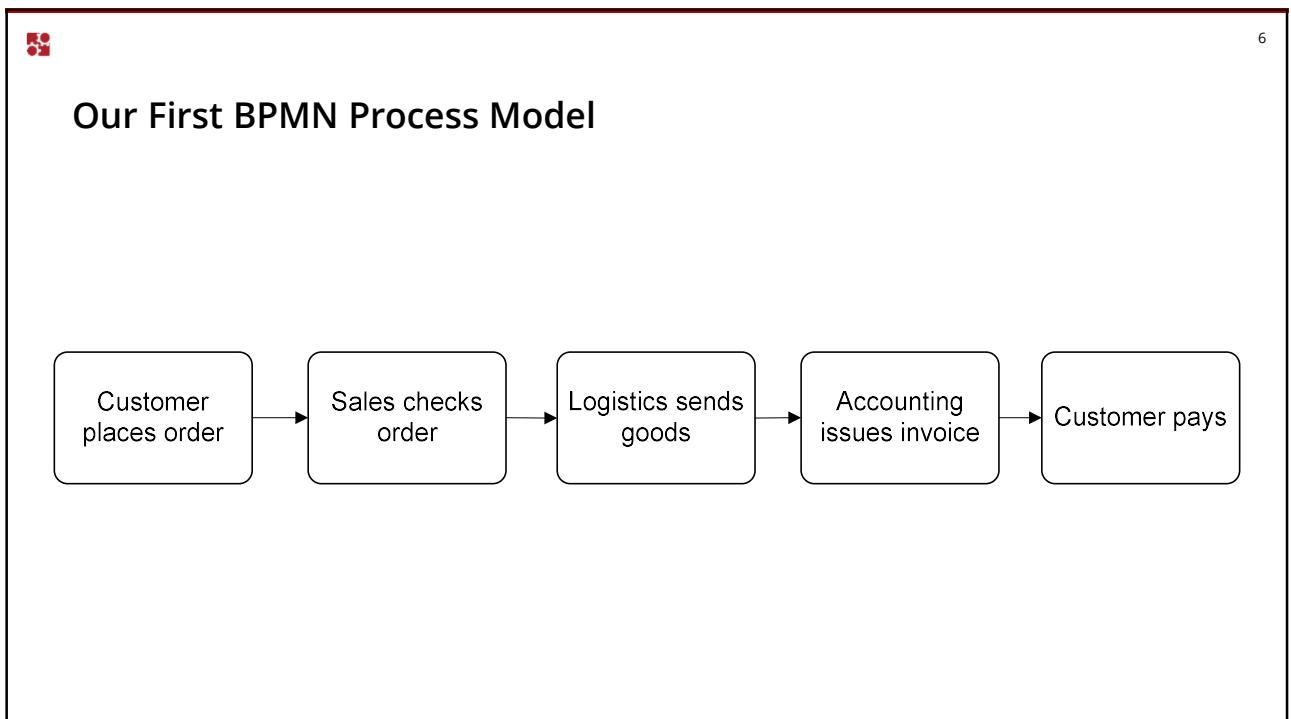


OBJECT MANAGEMENT GROUP





5

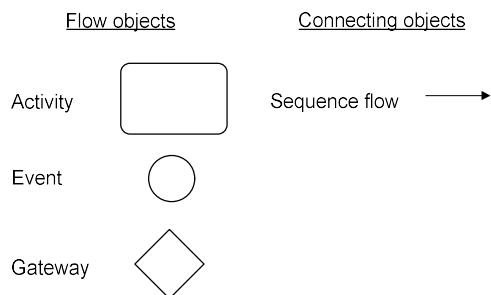


6



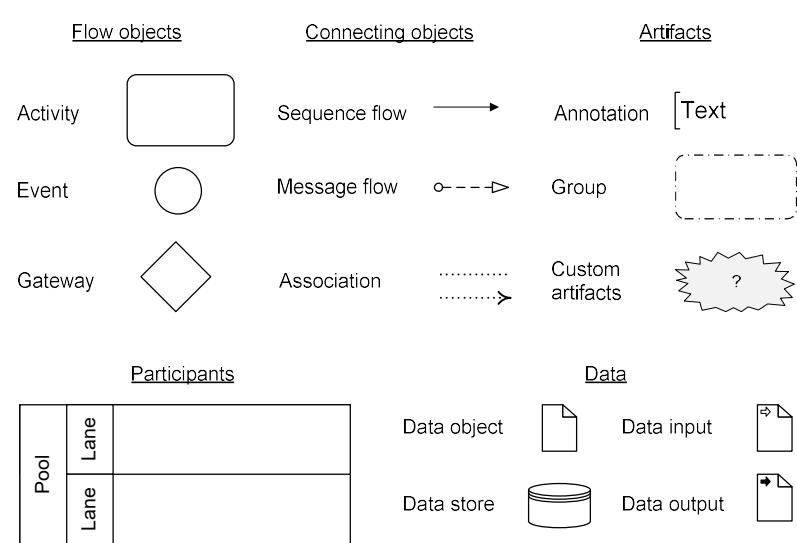
7

Basic Elements



8

Basic Elements - Documentation





9

The Scope of BPMN

BPMN at its best:

- Processes

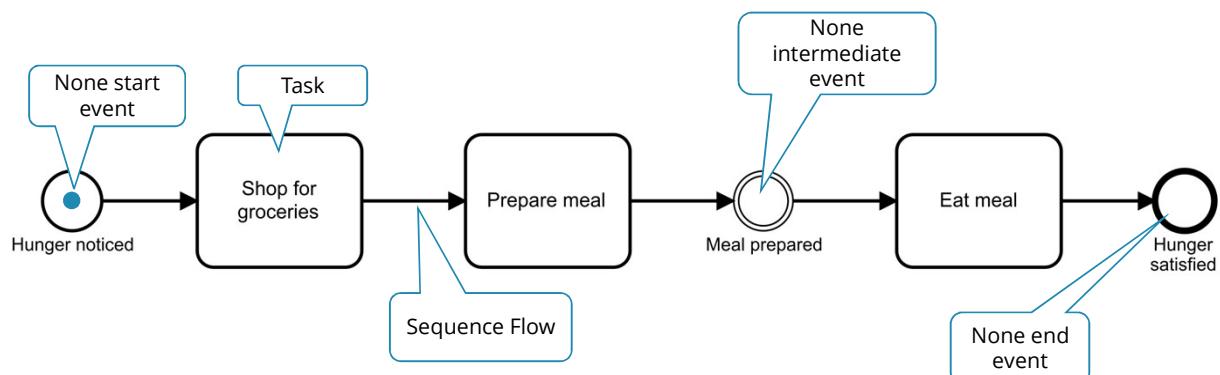
BPMN is not suitable for:

- Process landscapes
- Organizational landscapes
- Data
- Strategies
- Business rules
- IT landscapes



10

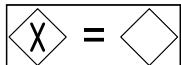
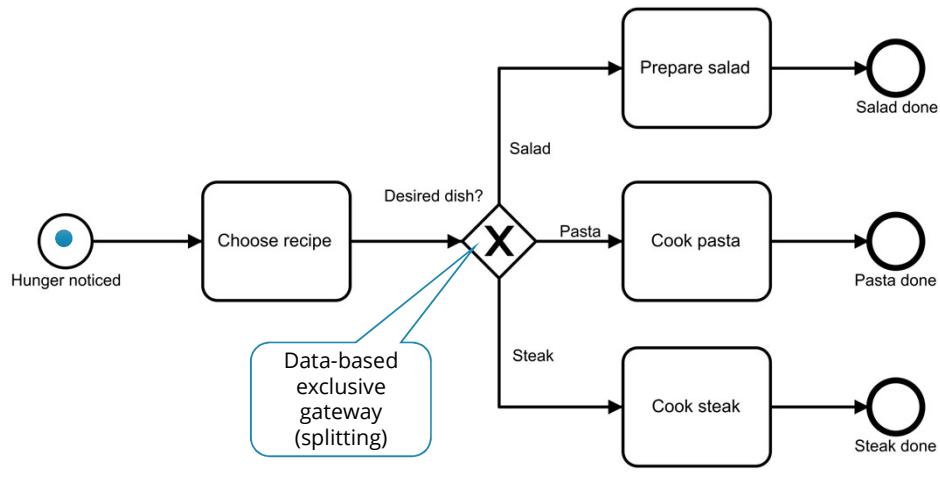
Tasks and events





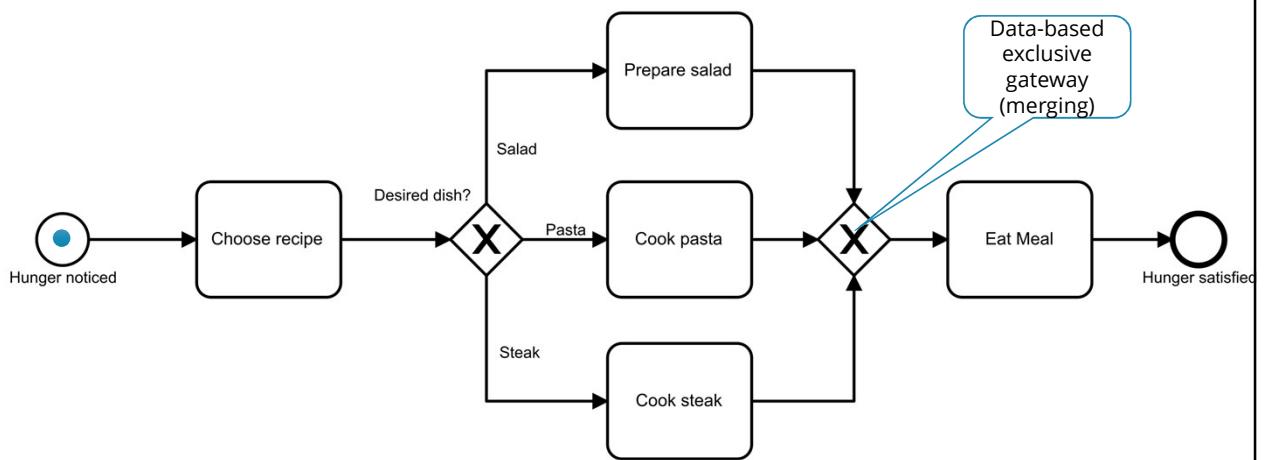
11

The XOR-gateway



12

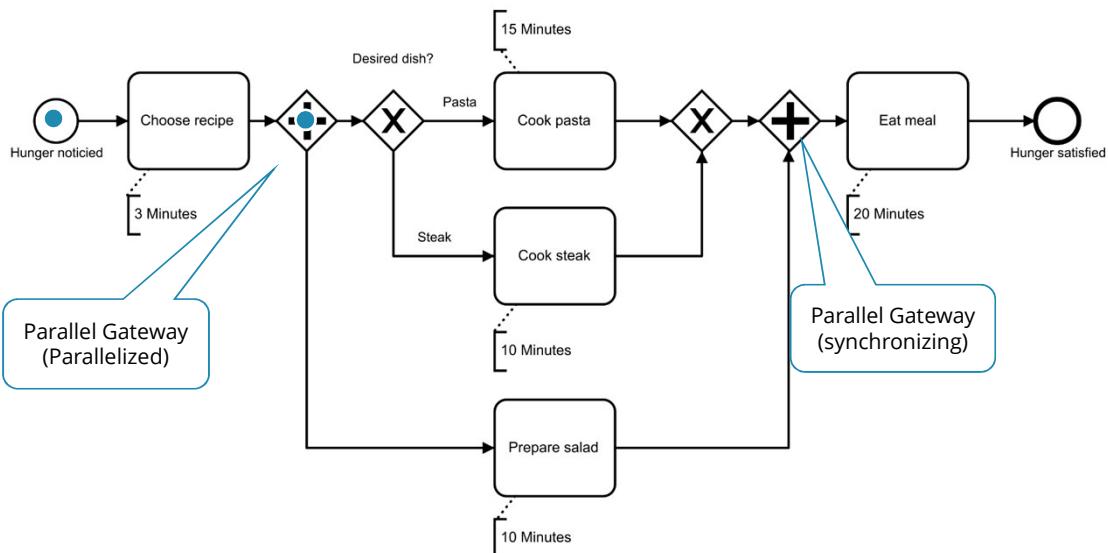
XOR-gateways can also merge





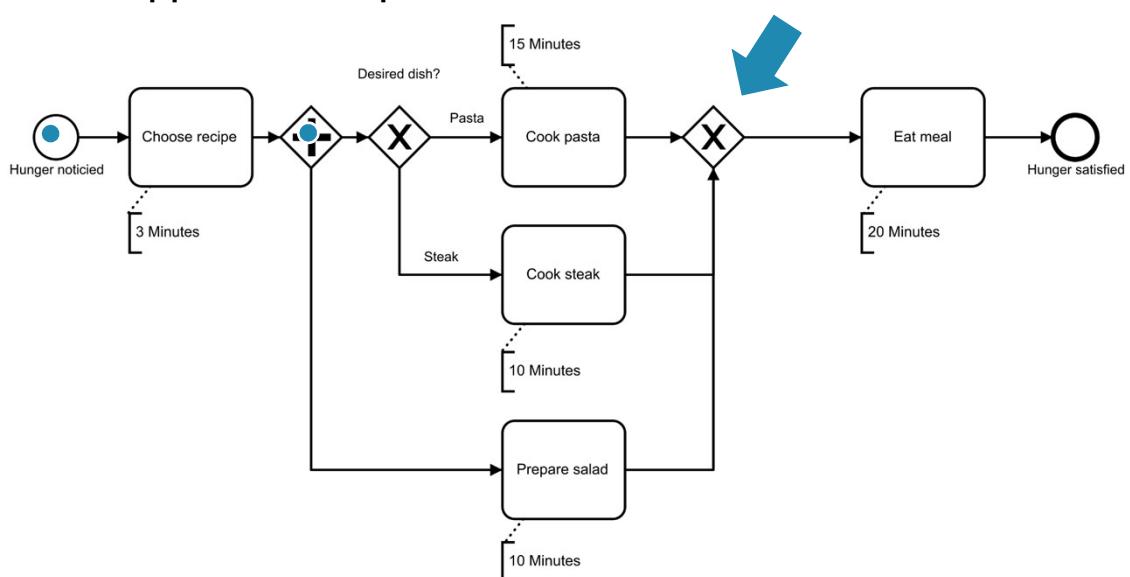
13

Preparing salad and main course at the same time



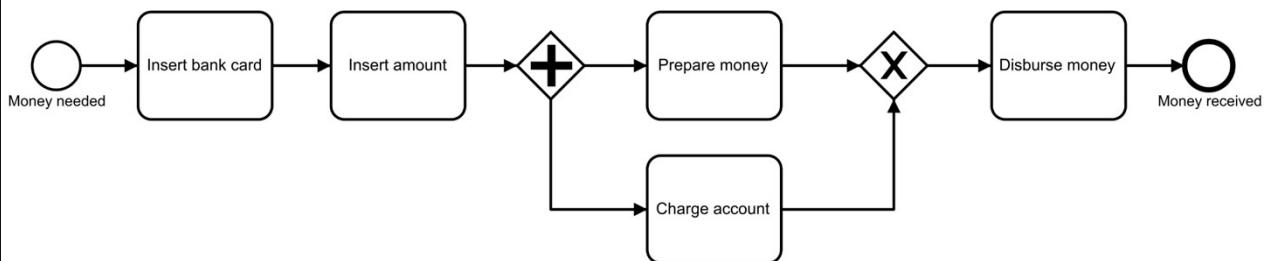
14

What happens in this process?

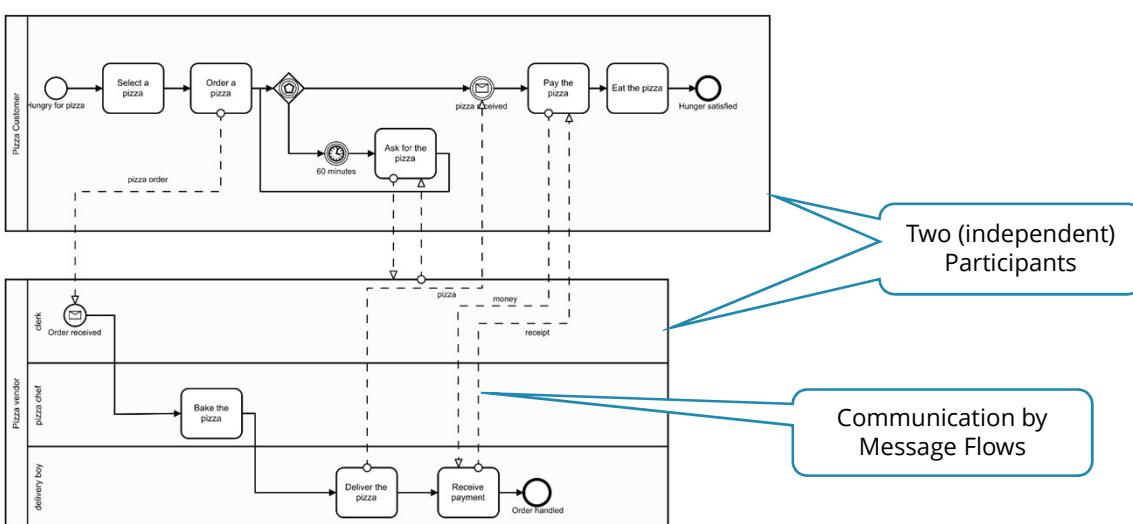




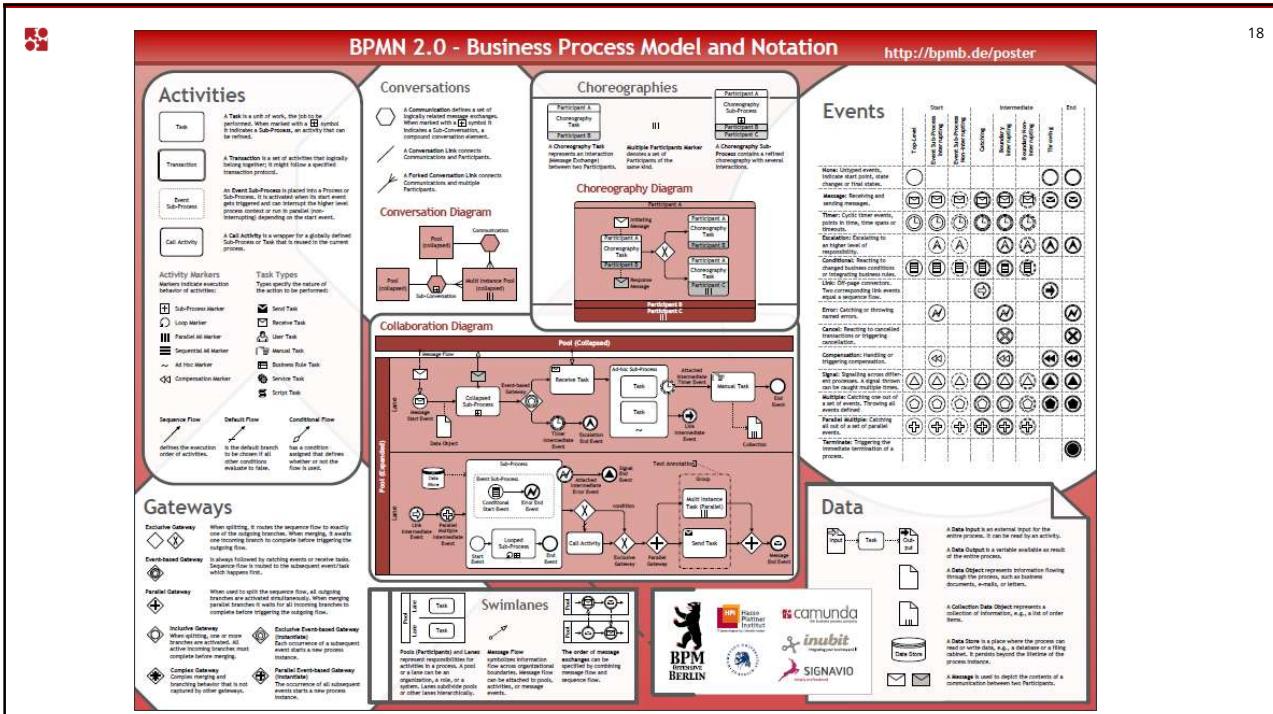
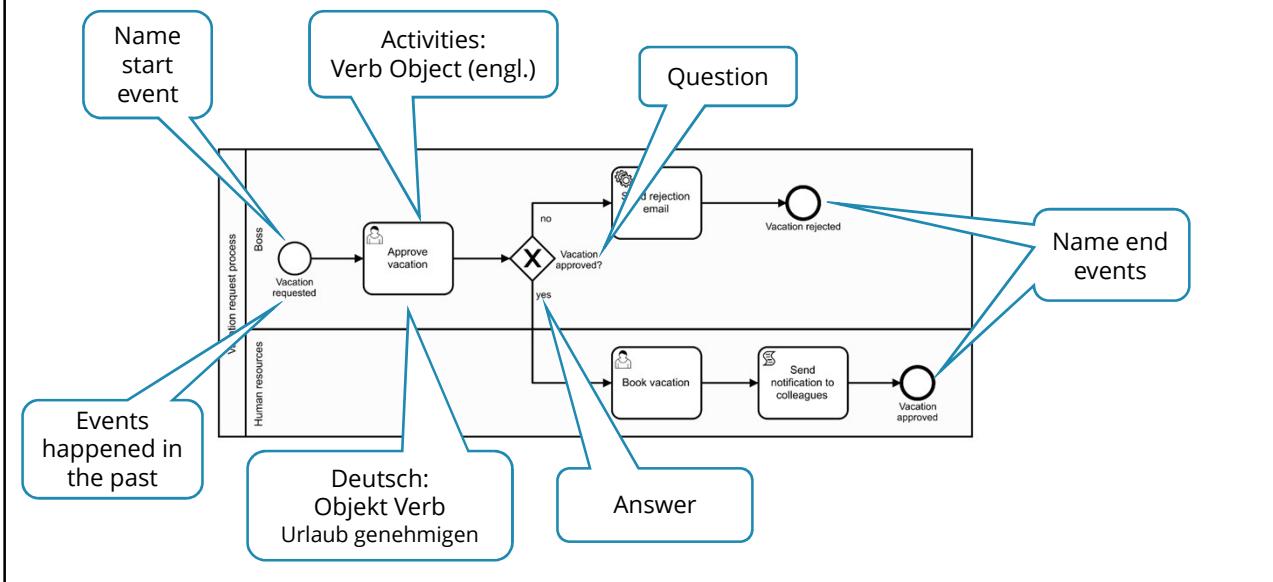
Example: ATM



Collaboration Diagram



Create Readable Process Models

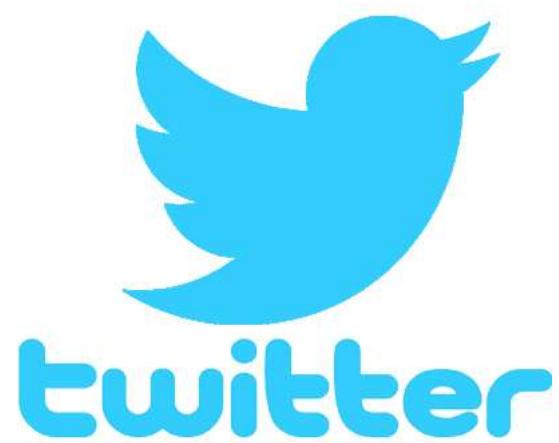


Scenario for Exercises



20

You Know Twitter?





21

Imagine the Anti Agile Inc.

Why do we need Twitter?

- We do not trust our employees.
- We want to control everything.
- We do not like Twitter,
- But we want to be part of it.



The Anti Agile Inc. Needs Twitter QA!



23

Group Exercise 0

Use a whiteboard or flipchart
to model a Twitter QA process



24

QA Process for Twitter

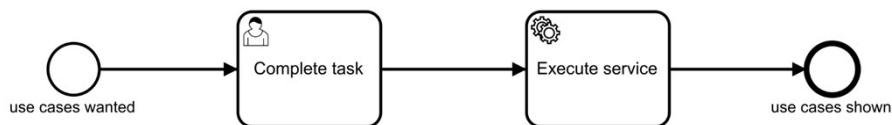
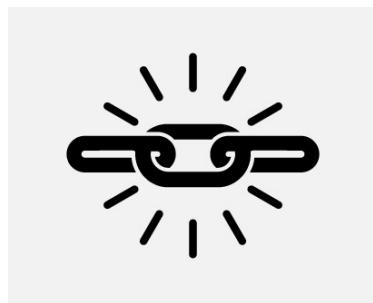


Process Automation Use Cases

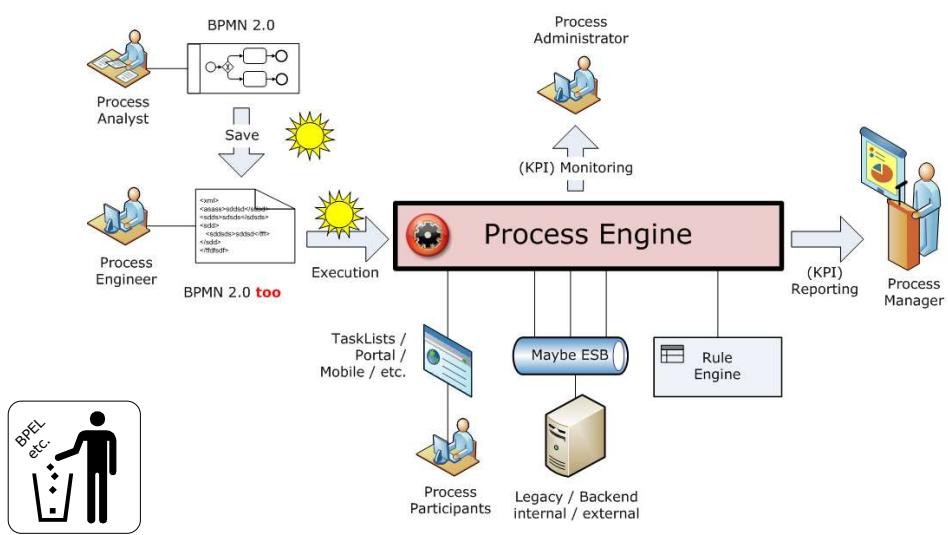
Human Task Management



System Integration



What's So Cool About BPMN 2.0





27

Why Using a Process Engine?

- Features
 - Task management
 - Timers & escalation
 - Error management and retry mechanisms
 - Versioning
 - The power of BPMN 2.0
 - Tools (modeler, cockpit, ...)
 - ...
- Transparency and Agility
 - Process is not hidden in the code
 - Process model can be used for monitoring and operations
- Quality

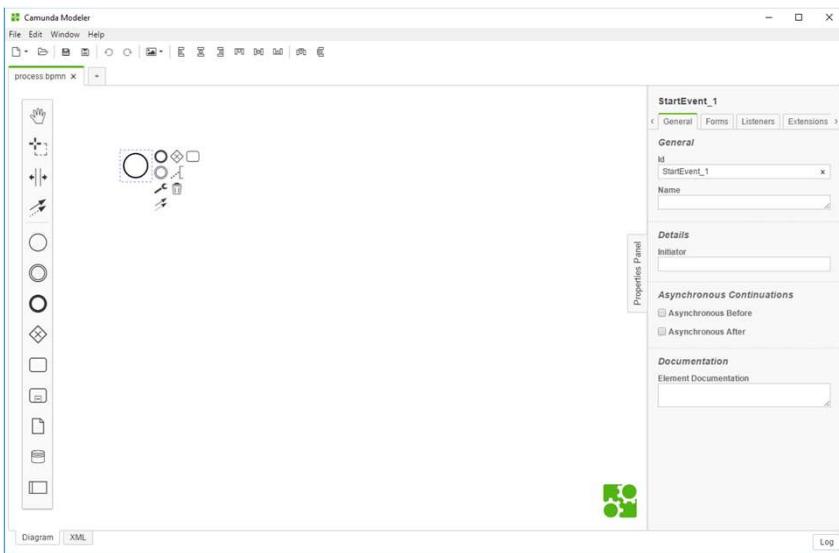


28

Demo



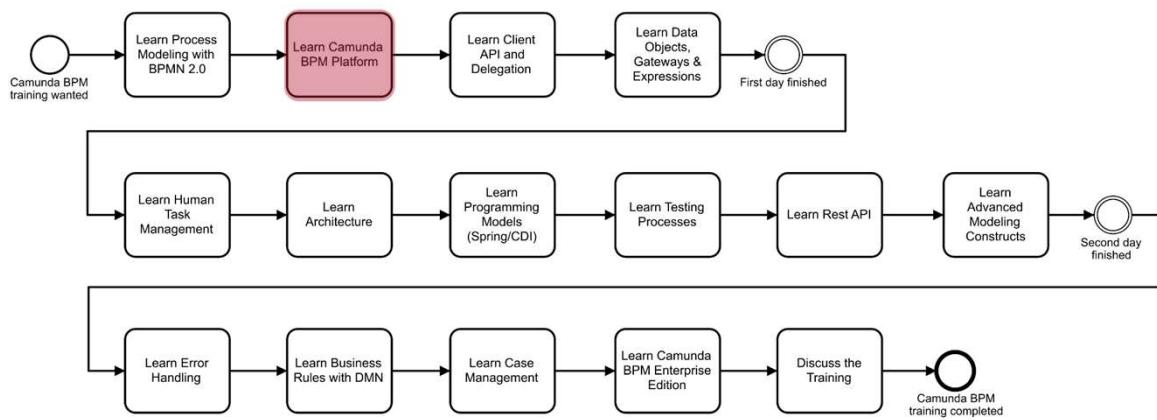
Start Modeling!



Exercise 1

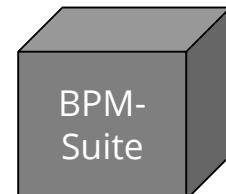
Follow the provided
instructions for exercise 1





32

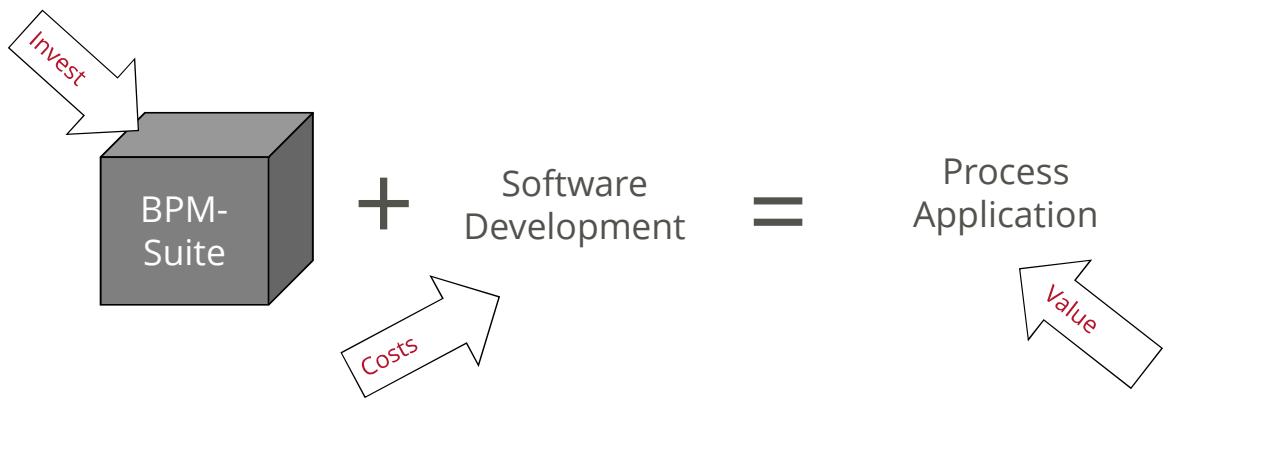
Shiny BPM Suites?





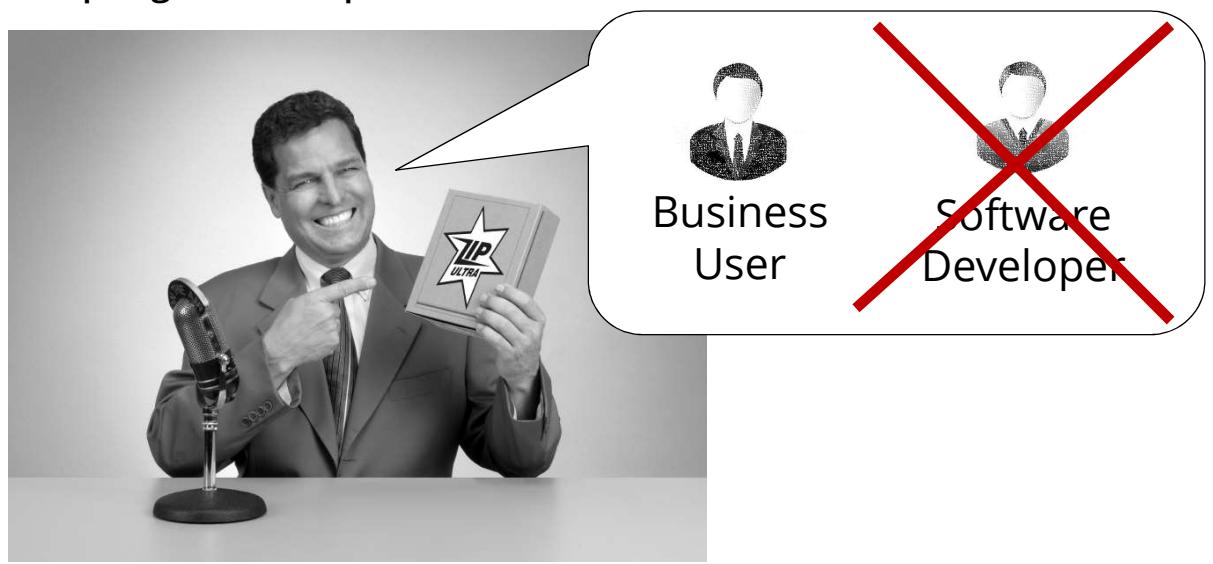
33

You Always Need Software Development



34

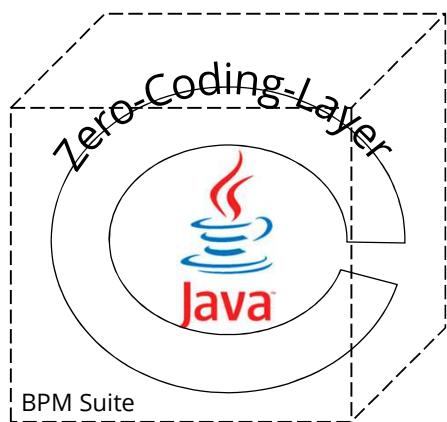
Tempting, but Deceptive





35

A Fundamentally Wrong Approach*



complicated
restrictive



Business User

restrictive
proprietary



Software
Developer

*for automating core business processes

36

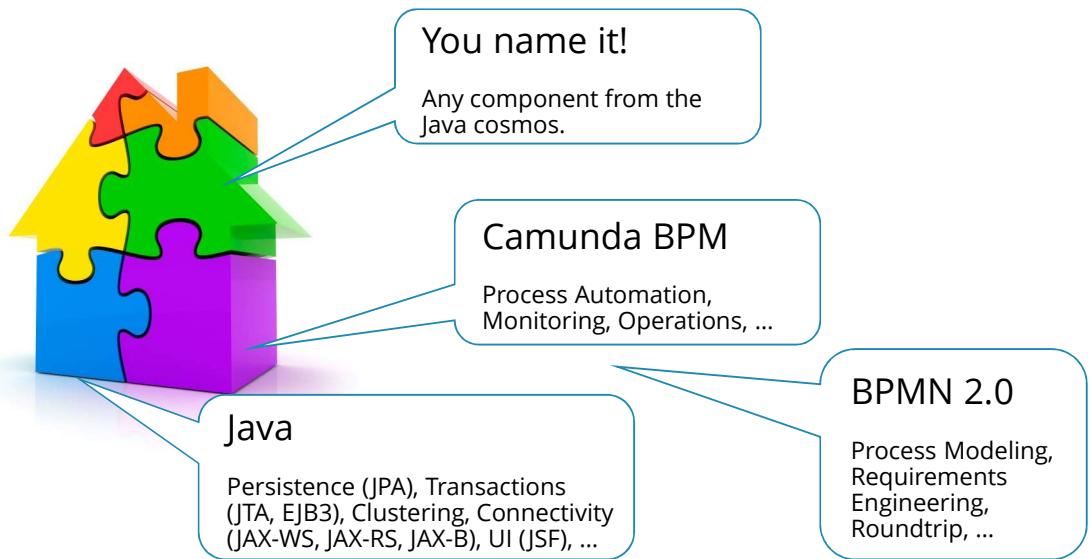
Our Approach:
BPM + Java



```
@Inject  
ProcessEngine engine;
```

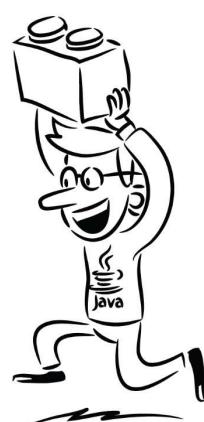


Best of Breed – Based on Standards



Lightweight and Open Source BPM Framework

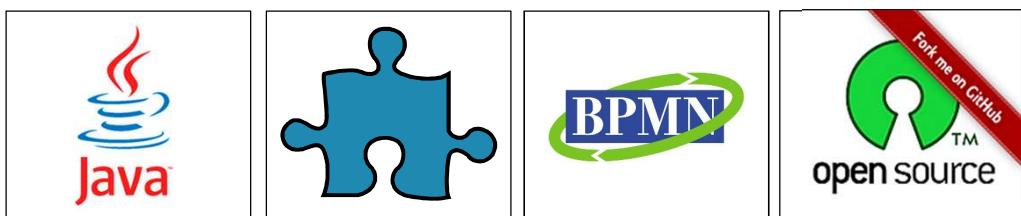
source code :
github.com/camunda



...Plus BPMN Based Business-IT-Alignment



Summary: Core Concepts of Camunda BPM





41

Decide Yourself!



BPM in a can

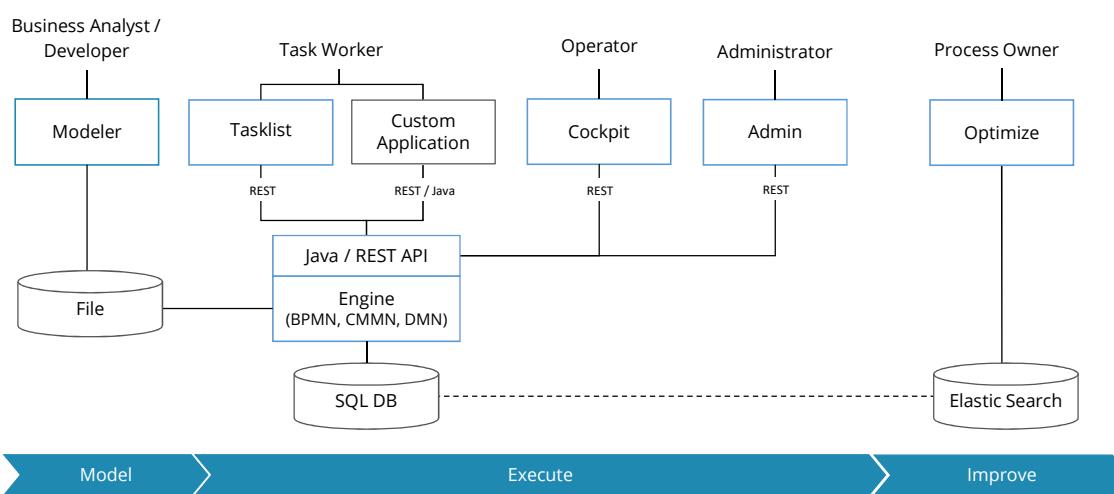


BPM cook-it-yourself



42

Camunda BPM Components





43

Exercise 2a

Follow the provided
instructions for exercise 2a



44

Demo



Camunda Tasklist

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with 'My Tasks' and 'My Group Tasks (2)'. Under 'My Group Tasks (2)', it lists 'Accounting', 'John's Tasks', 'Mary's Tasks', 'Peter's Tasks', and 'All Tasks'. The main area displays two tasks: 'Approve Invoice' and another 'Approve Invoice'. The first task is for 'Invoice Receipt' with details: Due 2 days ago, created 9 days ago; Invoice Amount: 10.99; Invoice Number: PSACE-5342. The second task is for 'Invoice Receipt' with details: Due 5 days ago, created 12 days ago; Invoice Amount: 30; Invoice Number: GPFE-23232323. Both tasks have a 'Form' tab selected, showing fields like Creditor (Papa Steve's all you can eat), Amount (10.99), Invoice Category (Travel Expenses), and Invoice Number (PSACE-5342). A checkbox 'Do you approve?' is present, along with 'Save' and 'Complete' buttons. The top navigation bar includes links for 'Create task', 'Start process', and 'Demo'.

Process Overview

The screenshot shows the Camunda Tasklist interface with the 'Approve Invoice' task selected. The 'Diagram' tab is active, displaying a BPMN process diagram. The process starts with an 'INVOC RECEIPT' event, which triggers a 'RECEIVE' gateway. From this gateway, one path leads to a 'RECEIVED' event, and another path leads to a 'REJECT INVOICE' gateway. This second gateway has two paths: one leading to a 'REJECTED' event and another leading to a 'RECEIVED' event. The process then continues through a 'RECEIVED' event, a 'RECEIVE' gateway, and a 'RECEIVED' event to a 'FINISH' event. The diagram also shows a 'Finance Accounting System' database at the bottom. The top navigation bar includes links for 'Create task', 'Start process', and 'Demo'.

Camunda Cockpit

Welcome to the Camunda... X Camunda Cockpit | Insurance Application X +

localhost:8080/camunda/app/cockpit/default/#/process-definition/insurance-application:2:4ad07ec9-5b09-11e6-...

Suchen

Camunda Cockpit Processes Decisions Deployments Reports Batches Demo Demo

Dashboard » Processes » Insurance Application : History

Information Filter

Definition Version: 2 -

Definition ID: insurance-application:...

Definition Key: insurance-application

Definition Name: Insurance Application

Tenant ID: null

Deployment ID: 4acc1117-5b09-11e6-...

Instances Running:

- current version: 0
- all versions: 1

Related

- Reports
- Migration

Process Instances Job Log

Powered by camunda BPM / v7.5.2-ee

The screenshot shows a BPMN process titled "Insurance Application". The process starts with an "Initial Activity" leading to a decision gateway. From the first outgoing path of the gateway, an "Application received" event leads to a "Risk Result" gateway. This gateway has three paths: "green (no risk)" leading to "Accept application", "yellow (medium risk)" leading to "Decide about application" (with a timer "2 days"), and "red (At least one severe risk)" leading to "Escalate handled". The "Accept application" path leads to "Handle policy in Backend" and "Send policy". The "Decide about application" path leads to "Escalate handled". The "Escalate handled" path leads to "Report Escalation in Backend" and "Send rejection". A "Policy issued" event is shown at the end. A heatmap overlay is applied to the process, color-coding nodes based on risk levels: green for low risk, yellow for medium risk, and red for high risk. A blue callout box in the top right corner points to the heatmap feature with the text "History view and Heatmap Enterprise Feature".

Cockpit Features

- Dashboard
- Monitor BPMN Processes
- Monitor DMN Decisions
- Browse deployments and resources in the process engine repository
- Graphical Process Instance Modification
- Auditing of Cockpit operations by REST API
- Batch view
- Reports (EE only)



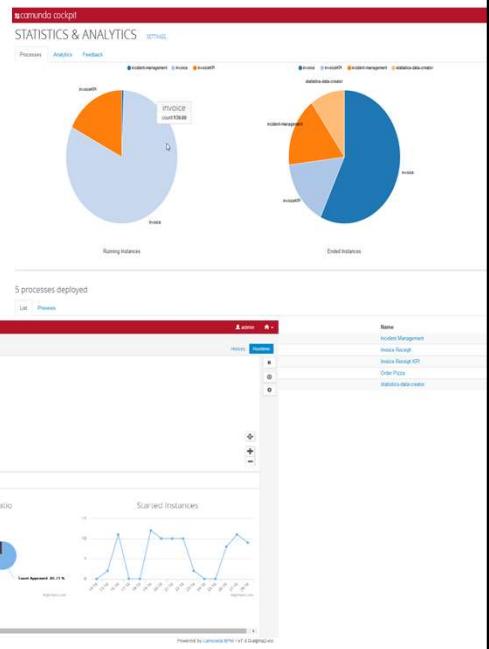
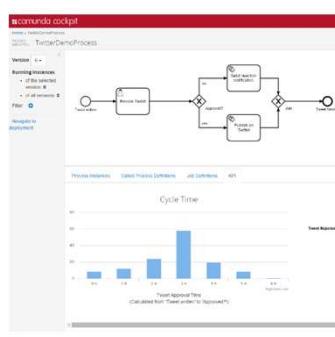
49

Cockpit Plugins

2 process definitions deployed

List	Previews				
State	Running Instances	Name	Tenant ID	History view	Report
1	1	Insurance Application		History view	all
1	5	Invoice Receipt		History view	all

Cockpit Plugin
Yeeeeah!



50

Camunda Admin

Welcome to the Camunda... X Camunda Admin | Dashbo... X

localhost:8080/camunda/app/admin/default/#/

Camunda Admin Users Groups Tenants Authorizations System

Demo Demo

Users Create New User List of Users My Profile	Groups Create New Group List of Groups	Tenants Create New Tenant List of Tenants
Authorizations Manage Authorizations	System General Execution Metrics License Key	

Powered by camunda BPM / v7.6.1-ea

Authorizations

The screenshot shows the Camunda Admin interface for managing authorizations. On the left, a sidebar lists various authorization types: Application, Authorization, Batch, Decision Definition, Deployment, Filter, Group, Group Membership, Process Definition (which is selected), Process Instance, Task, Tenant, Tenant Membership, and User. The main content area is titled "Process Definition Authorizations". It contains a table with the following data:

Type	User / Group	Permissions	Resource Id	Action
ALLOW	# accounting	READ, READ_HISTORY	invoice	Edit Delete
ALLOW	# camunda-admin	ALL	*	Edit Delete
ALLOW	# clerk	READ, UPDATE_INSTANCE, READ_HISTORY	insurance-application	Edit Delete
ALLOW	# lisa	READ, READ_INSTANCE, UPDATE_INSTANCE, READ_HISTORY	insurance-application	Edit Delete
ALLOW	# management	READ, READ_HISTORY	invoice	Edit Delete
ALLOW	# sales	READ, READ_HISTORY	invoice	Edit Delete

At the bottom right of the table, there is a button labeled "Create new authorization +". The footer of the page indicates it is "Powered by camunda BPM / v7.5.2-ee".

Camunda Modeler

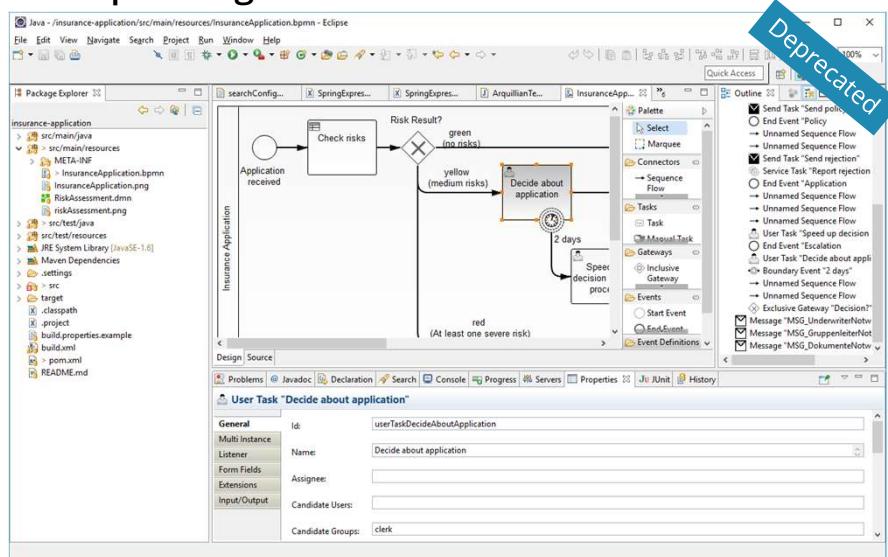
The screenshot shows the Camunda Modeler interface with a BPMN process diagram titled "InsuranceApplication.bpmn". The diagram includes the following elements:

- Start Event:** Initiates the process.
- Check risks:** An activity node with an outgoing arrow labeled "green (no risks)" leading to a decision diamond.
- Risk Result? Decision Diamond:** A decision diamond with three outgoing paths:
 - "green (no risks)": Leads to the "Accept application" path.
 - "yellow (medium risks)": Leads to a parallel gateway with two outgoing paths:
 - One leads to a task node "Decide about application" with a timer "2 days".
 - The other leads to a task node "Speed up decision making process".
 - "red (At least one severe risk)": Leads to a task node "Report rejection in insurance backend".
- Accept application:** An activity node connected to the "Accept application" path of the decision diamond.
- Issue policy in insurance backend:** An activity node connected to the "Accept application" path.
- Decision? Decision Diamond:** A decision diamond with two outgoing paths:
 - "Accept application": Leads to the "Issue policy in insurance backend" node.
 - "Reject application": Leads to a task node "Report rejection in insurance backend".
- Report rejection in insurance backend:** An activity node connected to both "Reject application" paths.

On the right side of the interface, there is a properties panel for a user task named "userTaskDecideAboutApplication". The properties include:

- General:**
 - Id:** userTaskDecideAboutApplication
 - Name:** Decide about application
 - Assignee:** clerk
 - Candidate Users:**
 - Candidate Groups:** clerk
 - Due Date:** \${dateTime()}.plusDays(1).toDate()
 - Follow Up Date:**
 - Priority:**
- Asynchronous Continuations:**
 - Asynchronous Before
 - Asynchronous After
- Documentation:**

Camunda Eclipse Plugin



bpmn.io – a Developer Kit and Web Modeler for BPMN 2.0

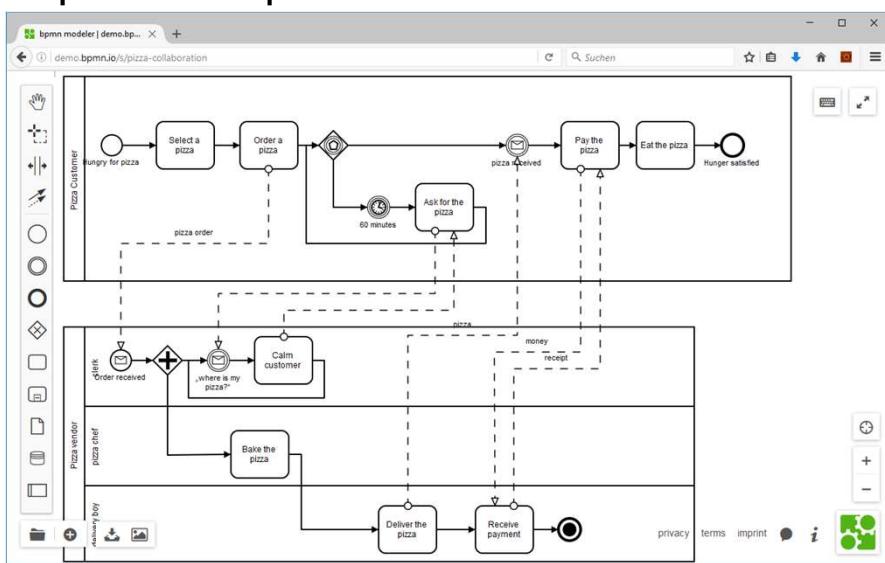
The screenshot shows the bpmn.io interface, which includes a browser-based modeler and a toolkit for embedding BPMN diagrams. At the top, there's a header with the URL "https://bpmn.io/toolkit/bpmn-js/" and a search bar. Below the header, the text "BPMN everywhere, for everyone" is displayed, followed by the subtext "Create, embed and extend BPMN diagrams in your Browser. Use it standalone or integrate it into your application." Three main sections are shown: "Model", "Embed and Annotate", and "Extend". The "Model" section shows a simple BPMN diagram with a start event, a task, and an end event. The "Embed and Annotate" section shows a more complex diagram with a gateway, tasks, and annotations like "Customers like our performance". The "Extend" section shows how to integrate an in-browser process engine. A large blue diamond-shaped watermark with the word "Deprecated" is overlaid across the entire screenshot.

<https://github.com/bpmn-io>

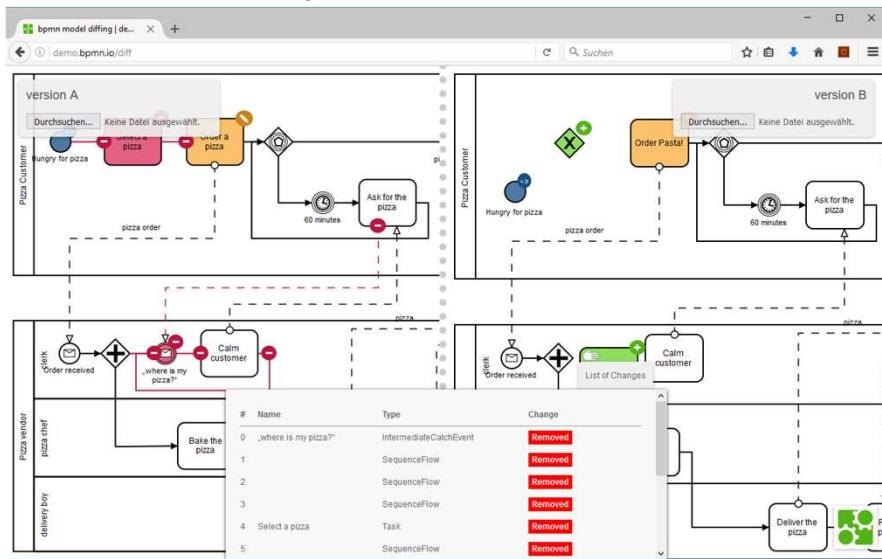
The screenshot shows the GitHub organization page for bpmn.io. It displays three repositories:

- bpmn.io**: A BPMN 2.0 rendering toolkit and web modeler. Updated 15 minutes ago.
- cmmn-js-properties-panel**: A properties panel for cmmn-js. Updated an hour ago.
- bpmn-js**: A BPMN 2.0 rendering toolkit and web modeler. Updated an hour ago.

<https://demo.bpmn.io>



BPMN Diff With bpmn.io



<https://docs.camunda.org>

The Camunda BPM Manual

Welcome to the Camunda BPM Manual! Camunda BPM is a light-weight, open-source platform for Business Process Management. The Manual introduces key concepts in Camunda BPM, provides installation procedures as well as a comprehensive reference section.

Getting Started

The Guide GET STARTED WITH BPMN 2.0 is the quickest and easiest way to get up and running with Camunda. Alternatives include:

BPMM 2.0	CMMN 1.1	DMN 1.1
Spring Framework	Java EE 6	Apache Maven

Once you completed a Getting Started Guide, you may find the following topics useful:

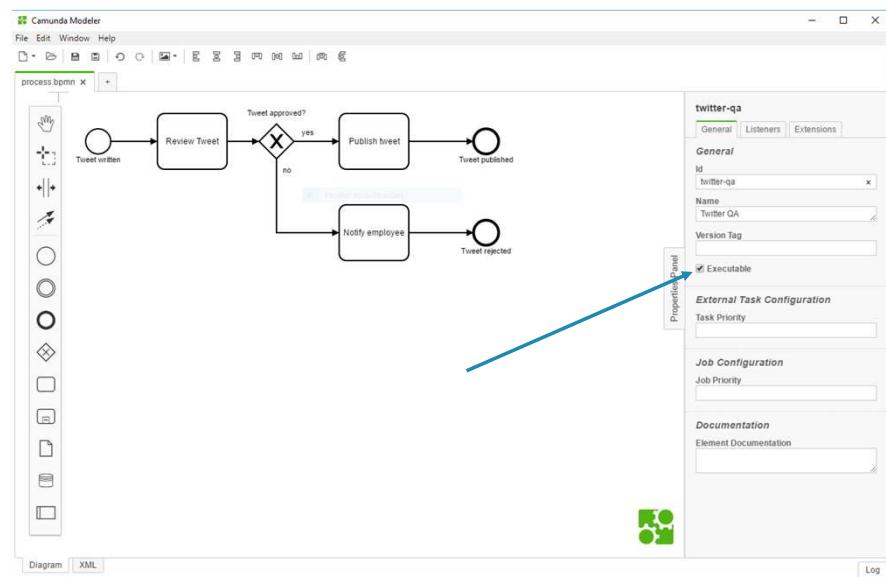
Introduction	Guides	References
Overview	Installation	REST API
Download	User Guide	Javadoc
Architecture	Modeler	BPMN 2.0
Supported Environments	Web Applications	CMMN 1.1
	Update & Migration	DMN 1.1
		HTML Forms

[OPTIONS](#)

camunda.org and docs.camunda.org are part of camunda BPM | Built by camunda and contributors — Privacy Statement — camunda Services GmbH © 2015

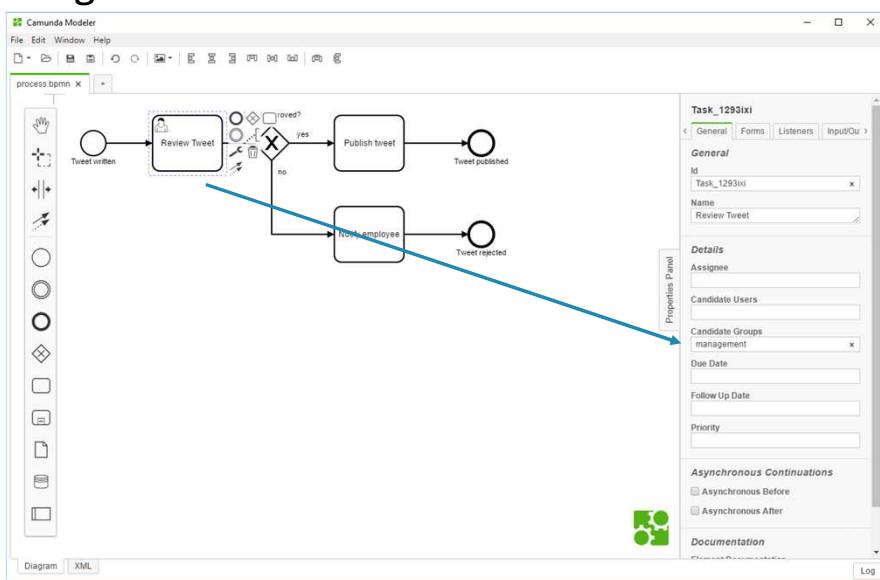
Prepare for First Launch

Make Process Executable

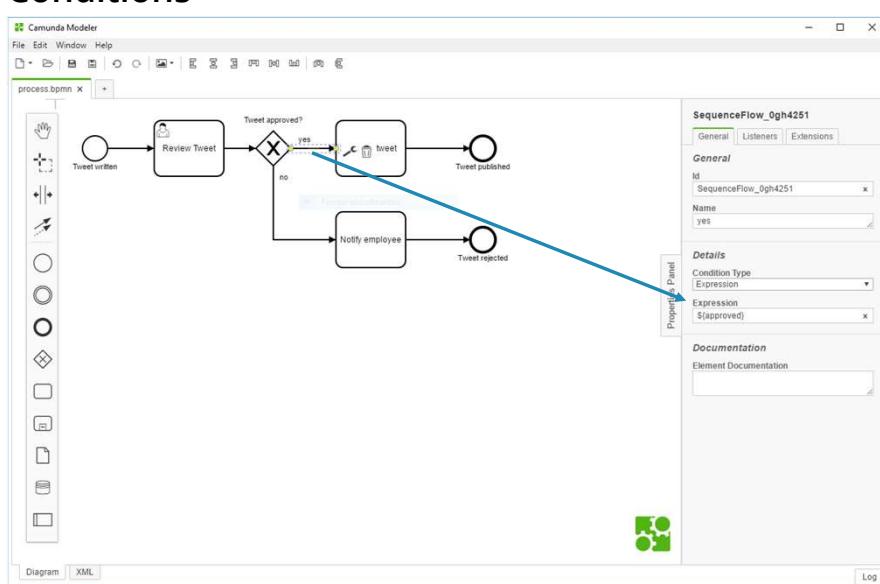


60

Assign User Task

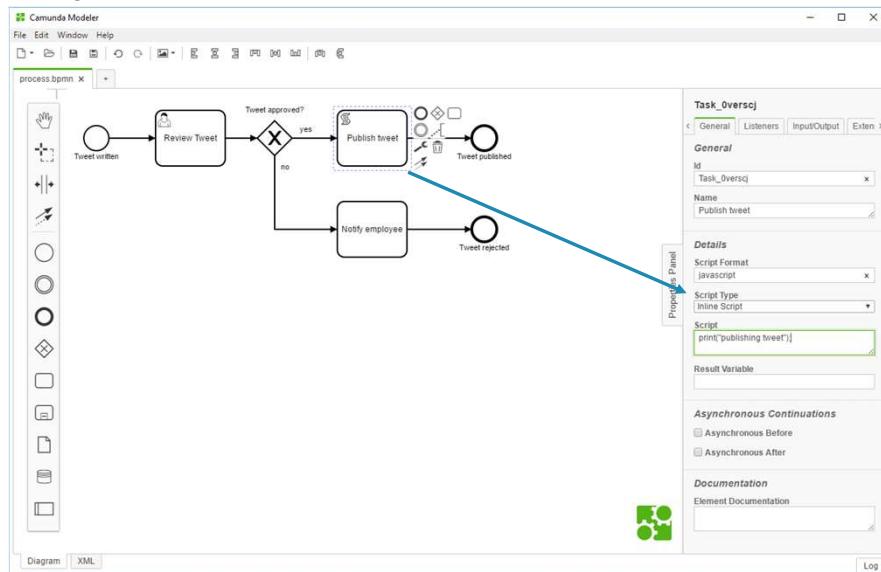


Conditions

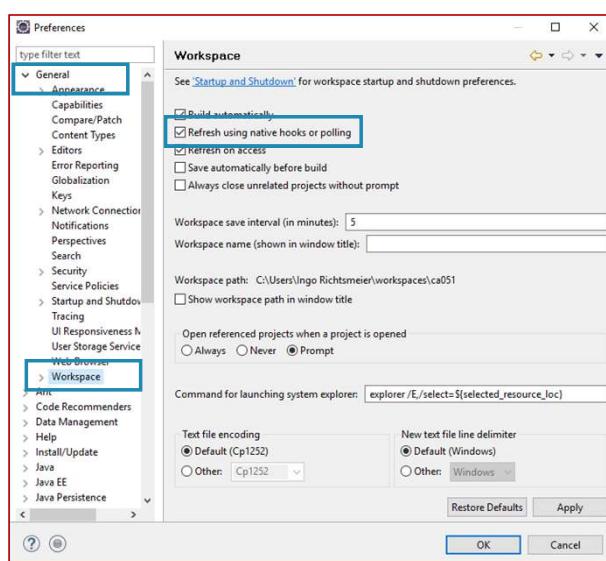




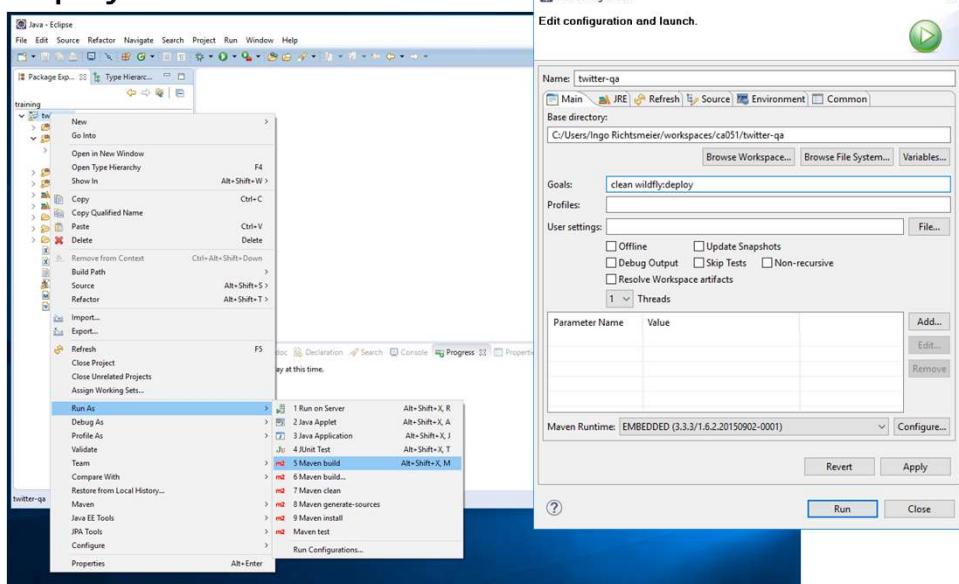
Scripts



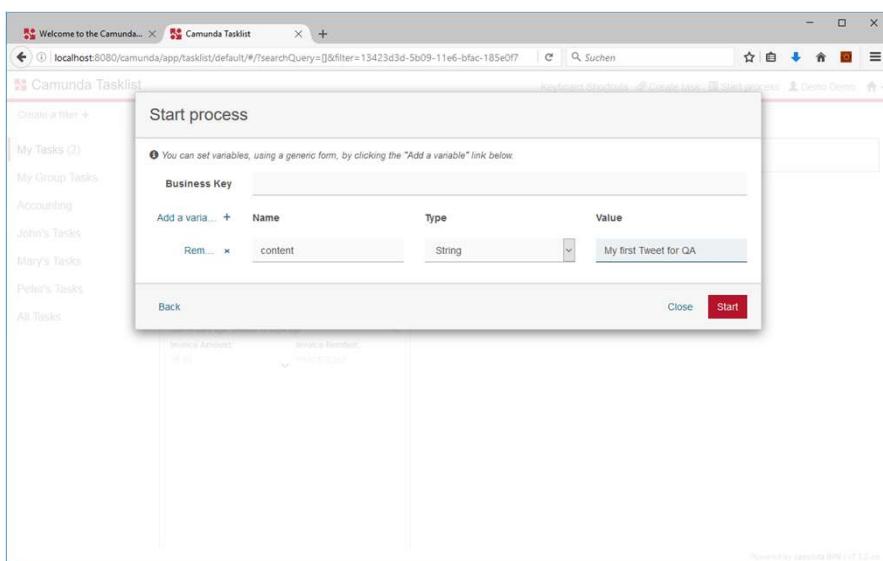
Setup Eclipse for Automatic Synchronization



Deploy With Maven



Use Generic Task Forms



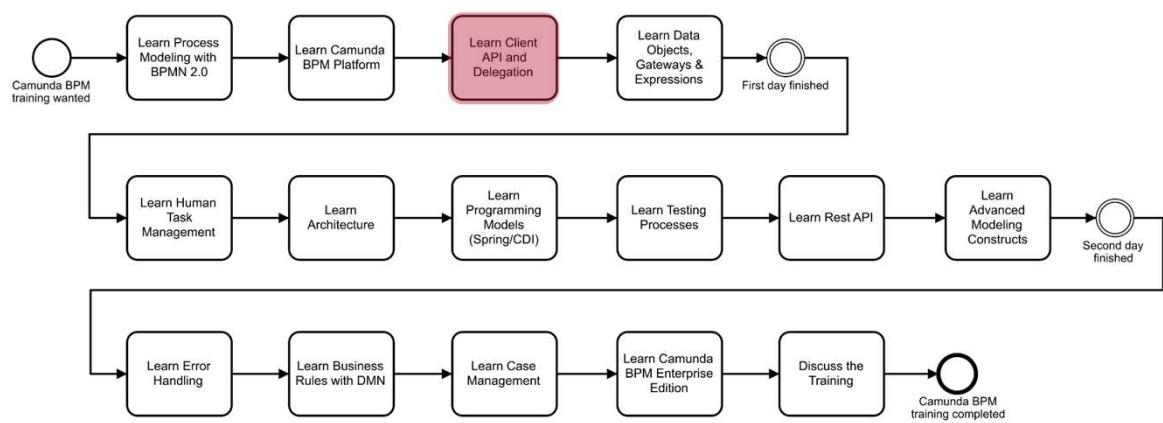
Complete Generic Task Forms

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with navigation links like 'My Tasks', 'My Group Tasks (2)', 'Accounting', 'John's Tasks', 'Mary's Tasks', 'Peter's Tasks', and 'All Tasks'. The main area displays two tasks under 'My Group Tasks (2)'. The first task is 'Approve Invoice' for John, with details: Due 3 days ago, created 10 days ago; Invoice Amount: 10.00; Invoice Number: PSAGE-5342. The second task is another 'Approve Invoice' for John, with details: Due 6 days ago, created 13 days ago; Invoice Amount: 10; Invoice Number: GPFE-23232323. To the right, a modal window titled 'Review Tweet' is open, showing a 'Twitter QA' section with buttons for 'Set follow-up date' and 'Set due date'. Below this is a 'Form' tab, which is selected, showing a table for 'Business Key'. The table has two rows: one for 'content' (String type, value 'My first Tweet for QA') and one for 'approved' (Boolean type, checked). A red 'Complete' button is at the bottom right of the form.

Exercise 2b

Follow the provided instructions for exercise 2b





BPM + Java

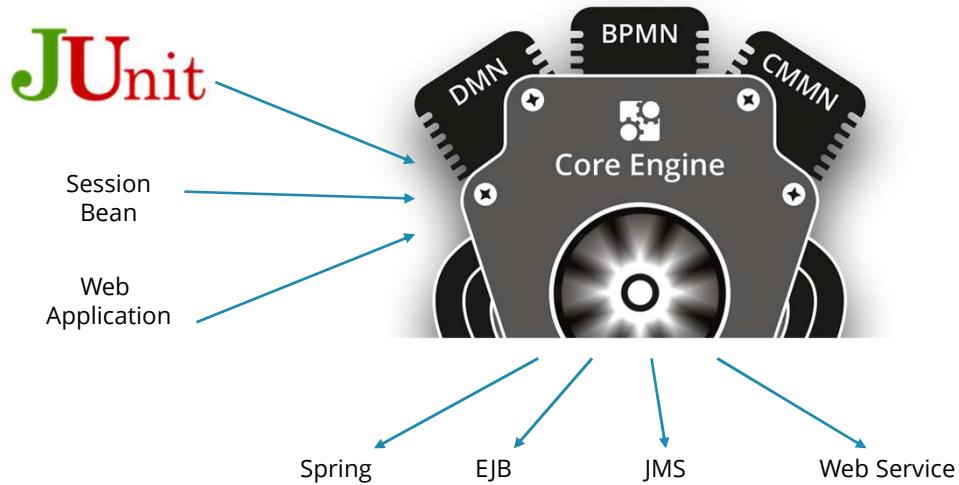
70





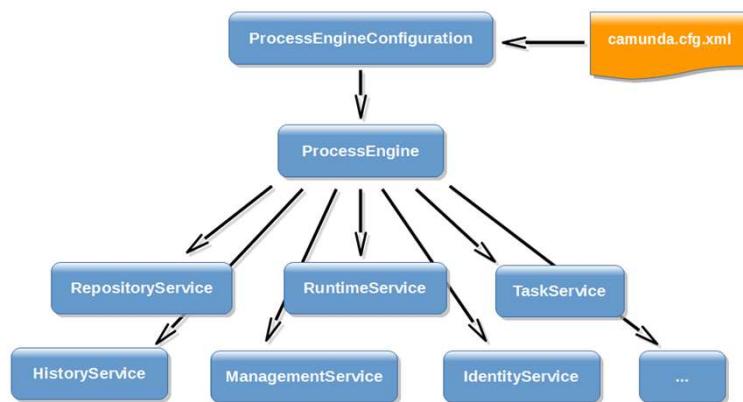
71

We Need a Way INTO and OUT of the Engine



72

Services, the Way IN





Overview Engine Services

Service	Description
Repository	Deployment, Process definitions
Runtime	Start of process instances, Variables, Event correlation
Task	Task lists, Task life cycle
Identity	Users, Groups
Form	Form key, Form property, Form
History	Logs, Statistics, KPIs
Management	Jobs, Statistics, Maintenance
Filter	Tasklist filters
ExternalTask	External task lifecycle
Case	Start of cases, Case execution lifecycle
Decision	Evaluate decisions
Authorization	Permissions



Java Client API

```
ProcessEngine processEngine = ProcessEngineConfiguration
    .createStandaloneProcessEngineConfiguration()
    .setJdbcUrl("jdbc:h2:./camunda-h2-dbs/process-engine;DB_CLOSE_DELAY=1000")
    .setDatabaseSchemaUpdate("true")
    .setJobExecutorActivate(true)
    .buildProcessEngine();

processEngine.getRepositoryService()
    .createDeployment()
    .addClasspathResource("vacation-request.bpmn")
    .deploy();

ProcessInstance processInstance = processEngine.getRuntimeService()
    .startProcessInstanceByKey("VacationRequestProcess");
```



Process Versioning

Process Definitions

ID_	KEY_	VERSION_	...
invoice:1:12ce6c9a-5b09-11e6-bfac-185e0f710ea8	invoice	1	
invoice:2:131a1ba0-5b09-11e6-bfac-185e0f710ea8	invoice	2	

Process Instances

ID_	PROC_DEF_ID_	ACT_ID_	...
1378dcae-5b09-11e6-bfac-185e0f710ea8	invoice:1:12ce6c9a-5b09-11e6-bfac-185e0f710ea8	reviewInvoice	
14b81440-5b09-11e6-bfac-185e0f710ea8	invoice:2:131a1ba0-5b09-11e6-bfac-185e0f710ea8	approveInvoice	



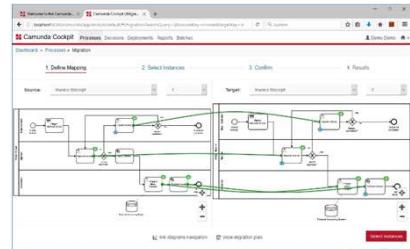
Start New Process Instances

Default Behavior:

1. New version upon each changed BPMN XML deployment
2. New process instances start with the latest version
3. Running process instances continue using the version they started with

```
ProcessInstance processInstance = processEngine.getRuntimeService()
.startProcessInstanceByKey("VacationRequestProcess");
```

If needed, process instance migration is possible





77

Key vs. ID vs. Name

```
ProcessInstance processInstance = processEngine.getRuntimeService()  
    .startProcessInstanceByKey("VacationRequestProcess");
```

- ProcessDefinitionKey: BPMN Process ID
- ProcessDefinitionName: BPMN Process Name
- ProcessDefinitionId: Generated Unique ID

```
<bpmn2:process id="VacationRequestProcess"  
    name="Vacation request"  
    isExecutable="true">
```

Participant_041airq

General Listeners Extensions

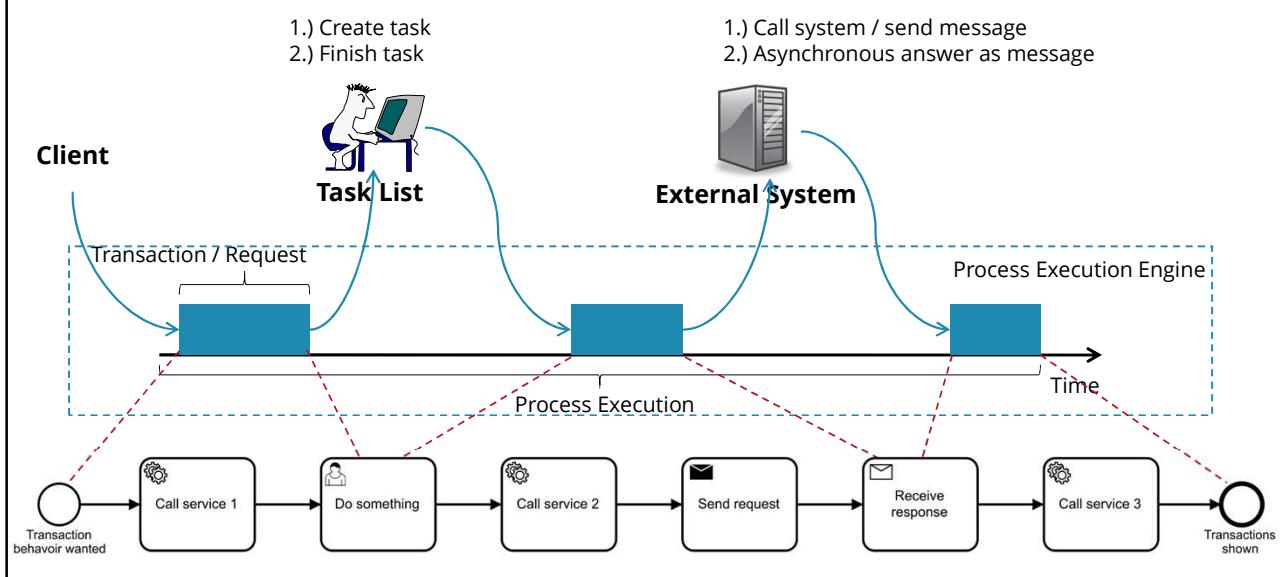
General

Id	Participant_041airq
Name	Vacation request process
Version Tag	
Process Id	VacationRequestProcess
Process Name	Vacation request
<input checked="" type="checkbox"/> Executable	

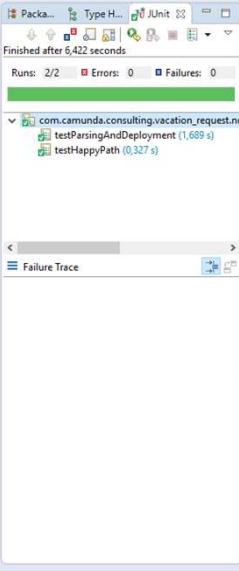


78

Process Engine and Transactions



Unit Testing



```

import static org.camunda.bpm.engine.test.assertions.ProcessEngineTests.*;
public class VacationRequestProcessTest {

    @Rule
    public ProcessEngineRule rule = new ProcessEngineRule();

    @Test
    @Deployment(resources = "vacation-request.bpmn")
    public void testHappyPath() {
        ProcessInstance processInstance = runtimeService()
            .startProcessInstanceByKey(PROCESS_DEFINITION_KEY);
        assertThat(processInstance).isWaitingAt("approveVacationUserTask");
        complete(task(), withVariables("approved", true));
        assertThat(processInstance).isWaitingAt("bookVacationUserTask");
        complete(task());
        assertThat(processInstance).isEnded();
    }
}

```

Shortcuts from camunda-bpm-assert

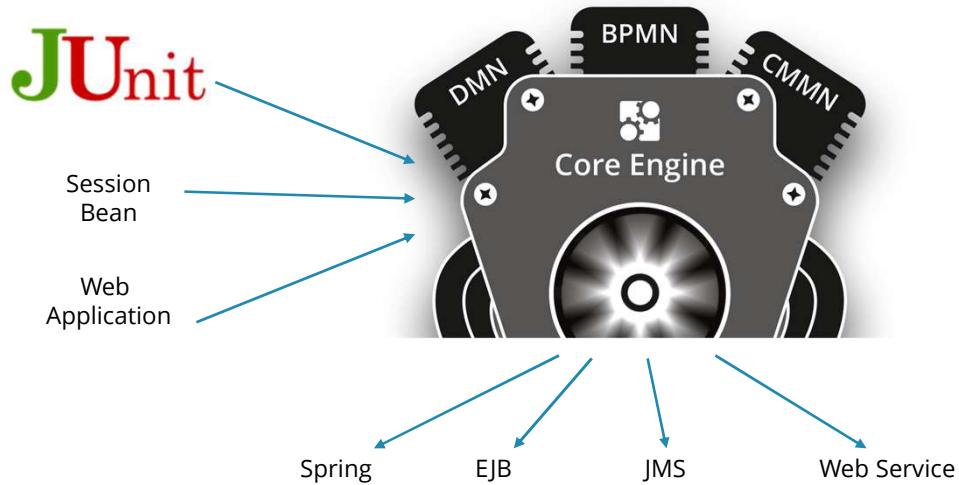
Exercise 3a

Follow the provided instructions for exercise 3a

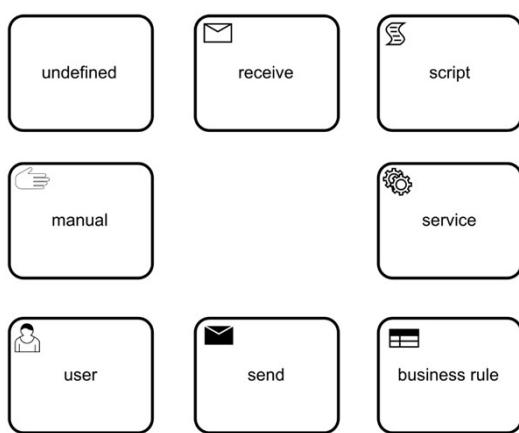




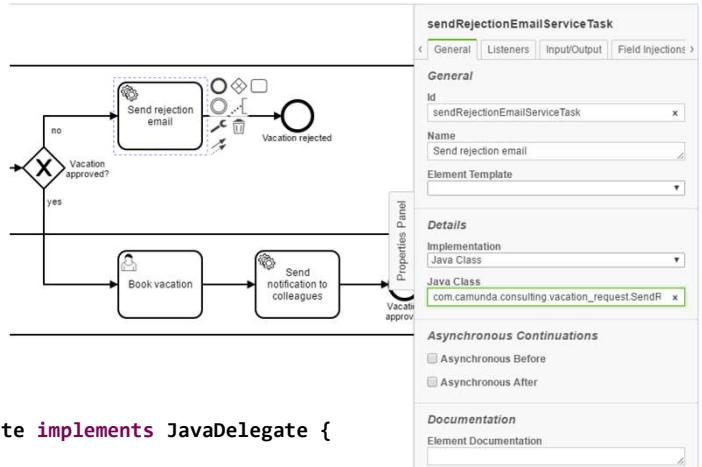
The Way OUT



BPMN 2.0 Task Types



Service Task With Java Class



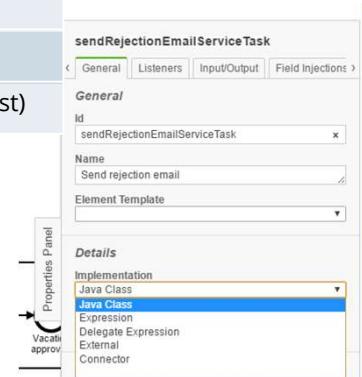
```
public class SendRejectionNotificationDelegate implements JavaDelegate {

    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Send the rejection to the employee...
        String reason = (String) execution.getVariable("rejectionReason");
    }
}
```

Other Options for Service Tasks

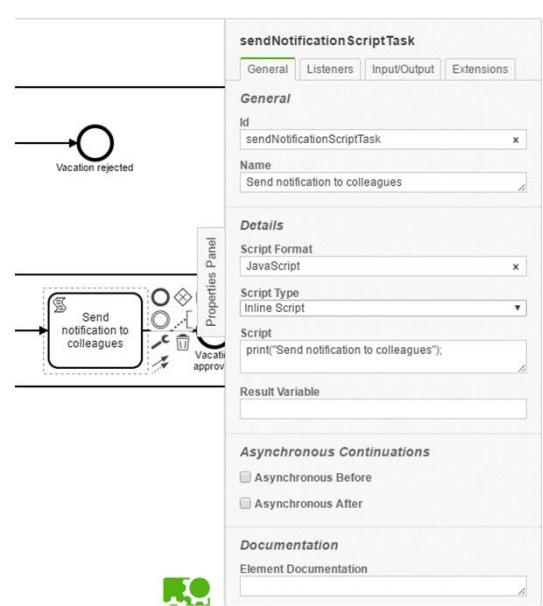
Implementation	Description
Expression	Resolve the implementation from an expression
Delegate Expression	Resolve the implementation from an expression, that implements JavaDelegate
External	Implement the service outside of the engine
Connector	Configure the call to a web service (soap or rest)

More details come later



Script Task

- Every JSR-223 compliant scripting engine can be used (we tested JS, Groovy, Jython, JRuby)
- Script sources can be externalized
- Templates can be used as script (for data transformation)
- Scripts get compiled and cached



Script Task Examples

```
<bpmn:scriptTask id="sendNotificationScriptTask" name="Send notification to colleagues"
scriptFormat="JavaScript">
  <bpmn:script>![CDATA[var message = [requester<br/>" will be on holiday from ";
message = message + from + " until " + until;
execution.setVariable("message", message);]]</bpmn:script>
</bpmn:scriptTask>
```

Special key word

Process variables

```
<bpmn:scriptTask id="checkUniqueRSPIDScriptTask" name="Check unique RSP ID"
scriptFormat="JavaScript" camunda:resultVariable="resultCheckUniqueRspID"
camunda:resource="scripts/check_unique_rsp_id.js">
</bpmn:scriptTask>
```

Load script from class path

Store script result



Business Rule Task

- Behaves like a Service Task
- Link to a DMN Decision Table
- More Details later



Execution Listeners

sendRejectionEmailServiceTask

General	Listeners	Input/Output	Field Injections
---------	-----------	--------------	------------------

Listeners

Execution Listener

Execution Listener

Event Type: start
Listener Type: Java Class
Java Class: com.camunda.consulting.vacation_request.listener.DataTransformationListener

Field Injection

Fields

```
public class DataTransformationListener implements ExecutionListener {

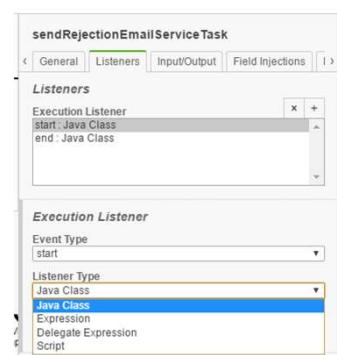
    @Override
    public void notify(DelegateExecution execution) throws Exception {
        String upperVar = (String) execution.getVariable("someText");
        upperVar = upperVar.toUpperCase();
        execution.setVariable("someText", upperVar);
    }
}
```



Events for Execution Listeners

Level	Event
On process level	Start, end
On activity level	Start, end
On sequence flow	take

Listener Type
Java Class
Expression
Delegate Expression
Script



Java Delegates for Execution Listeners

```
public class DataTransformationDelegate implements JavaDelegate {  
  
    @Override  
    public void execute(DelegateExecution execution) throws Exception {  
        String upperVar = (String) execution.getVariable("someText");  
        upperVar = upperVar.toUpperCase();  
        execution.setVariable("someText", upperVar);  
    }  
    or  
}  
  
public class DataTransformationListener implements ExecutionListener {  
  
    @Override  
    public void notify(DelegateExecution execution) throws Exception {  
        String upperVar = (String) execution.getVariable("someText");  
        upperVar = upperVar.toUpperCase();  
        execution.setVariable("someText", upperVar);  
    }  
}
```

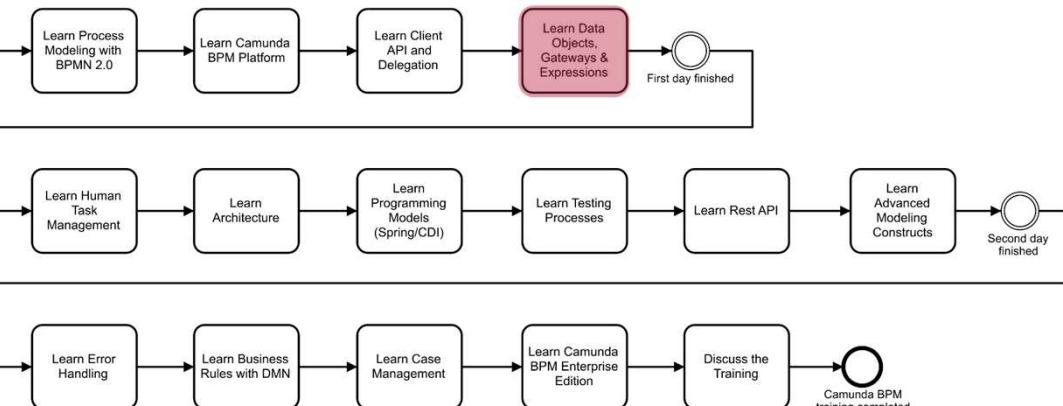


Exercise 3b

Follow the provided instructions for exercise 3b



Camunda BPM training wanted





Accessing Process Variables

Client Code:

```
Map<String, Object> variables = new HashMap<String, Object>();
variables.put("employee", "Jim");
runtimeService().startProcessInstanceByKey("VacationRequestProcess", variables);
```

Delegate Code:

```
public class SendRejectionNotificationDelegate implements JavaDelegate {

    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Send the rejection to the employee...
        String reason = (String) execution.getVariable("rejectionReason");
        String employee = (String) execution.getVariable("employee");

        // create a message to the employee
        execution.setVariable("message", "Some message to " + employee + " about " + reason);
    }
}
```

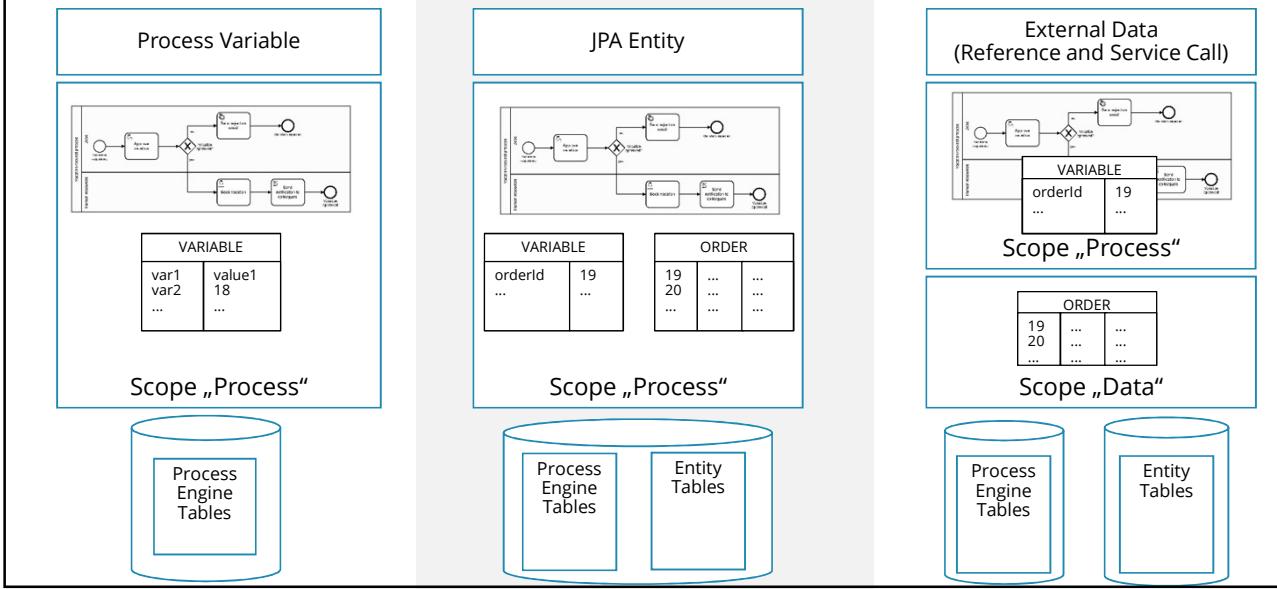


Process Variables

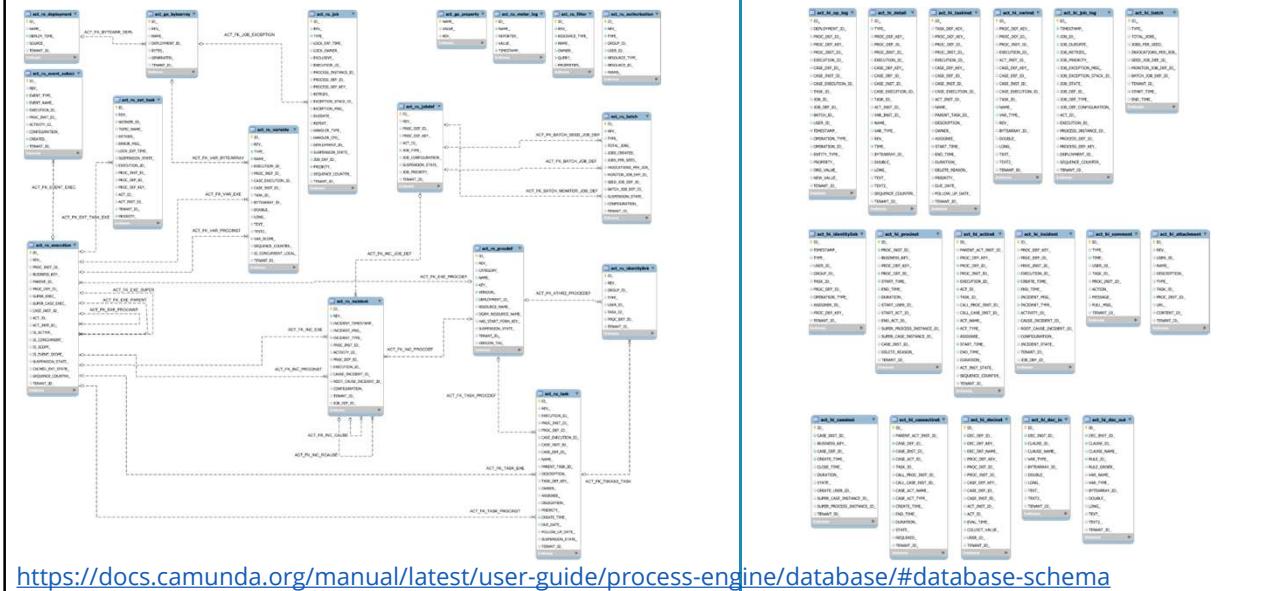
- Are not modeled in BPMN 2.0
- Behave like a `java.util.Map`
- Are persisted with the process instance
- Recommended: Primitive types, Date & String
- Objects can be serialized as XML or JSON
- Default fallback: Serializable Java Objects (not recommended!)
- Serialization can be exchanged:
<https://docs.camunda.org/manual/latest/user-guide/data-formats/data-formats-in-processes/#extending-serialization>



Storage



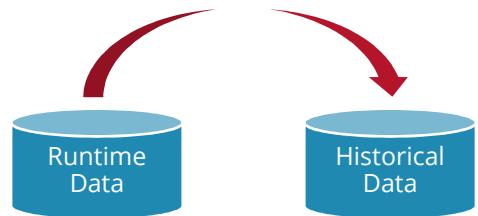
Database Schema





History Configuration

finished process instances are removed⁹⁷ from runtime tables



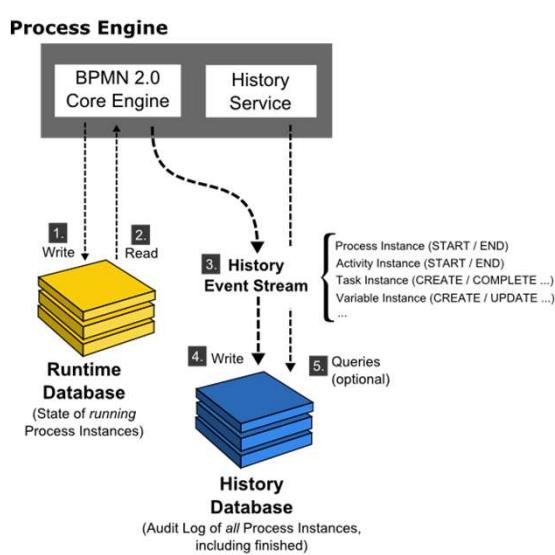
Configuration options:

- None: no history at all
- Activity: process instances and trail of executed BPMN activities
- Audit: + last value of every process variable
 - + submitted form properties (works only with forms)
- Full: + variable updates
- Custom: e.g. own filter for events of particular elements or variables



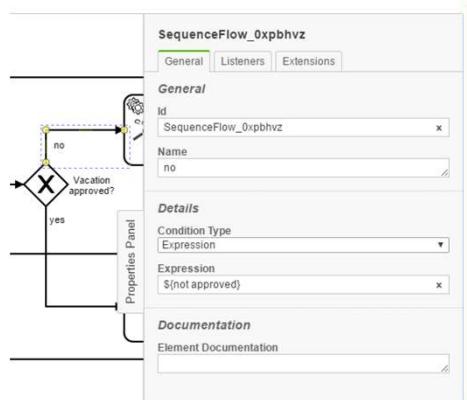
98

Asynchronous History



Expressions

- Work on data objects
- May work on Spring/CDI beans
- Used in:
 - Java service tasks
 - Event listeners
 - Conditional sequence flows
- Can use attributes or methods



`${myVar}
 ${myBean.myProperty}`

`${printer.print()}
 ${myBean.addNewOrder('orderName')}
 ${myBean.doSomething(myVar, execution)}`



Unified Expression Language (UEL)

EL Expression	Result	100
<code> \${1 > (4/2)}</code>	false	
<code> \${4.0 >= 3}</code>	true	
<code> \${100.0 == 100}</code>	true	
<code> \${(10*10) ne 100}</code>	false	
<code> \${'a' < 'b'}</code>	true	
<code> \${'hip' gt 'hit'}</code>	false	
<code> \${4 > 3}</code>	true	
<code> \${1.2E4 + 1.4}</code>	12001.4	
<code> \${3 div 4}</code>	0.75	
<code> \${10 mod 4}</code>	2	
<code> \${empty param.Add}</code>	False if the request parameter named Add is null or an empty string.	
<code> \${pageContext.request.contextPath}</code>	The context path.	
<code> \${sessionScope.cart.numberOfItems}</code>	The value of the numberOfItems property of the session-scoped attribute named cart.	
<code> \${param['mycom.productId']}</code>	The value of the request parameter named mycom.productId.	
<code> \${header["host"]}</code>	The host.	
<code> \${departments[deptName]}</code>	The value of the entry named deptName in the departments map.	
<code> \${requestScope['javax.servlet.forward.servlet_path']}</code>	The value of the request-scoped attribute named javax.servlet.forward.servlet_path.	
<code> #{customer.IName}</code>	Gets the value of the property IName from the customer bean during an initial request. Sets the value of IName during a postback	

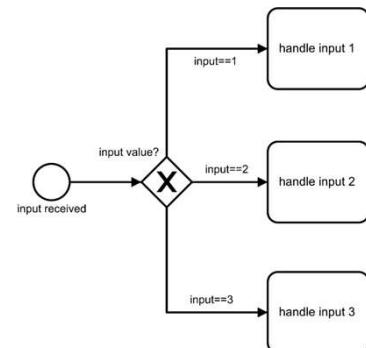
from

<http://java.sun.com/javaee/5/docs/tutorial/doc/bnahq.html#bnain>



101

Exclusive Gateway (Decision)



```
<bpmn:exclusiveGateway id="exclusiveGateway" name="input value?">
</bpmn:exclusiveGateway>
<bpmn:sequenceFlow id="fLow3" name="input==2" sourceRef="exclusiveGateway" targetRef="handleInput2Task">
  <bpmn:conditionExpression xsi:type="bpmn:tFormalExpression">${input==2}</bpmn:conditionExpression>
</bpmn:sequenceFlow>

<bpmn:sequenceFlow id="fLow2" name="input==1" sourceRef="exclusiveGateway" targetRef="handleInput1Task">
  <bpmn:conditionExpression xsi:type="bpmn:tFormalExpression">${input==1}</bpmn:conditionExpression>
</bpmn:sequenceFlow>

<bpmn:sequenceFlow id="fLow4" name="input==3" sourceRef="exclusiveGateway" targetRef="handleInput3Task">
  <bpmn:conditionExpression xsi:type="bpmn:tFormalExpression">${input==3}</bpmn:conditionExpression>
</bpmn:sequenceFlow>
```



102

Service Task With UEL Expression



```
<bpmn:serviceTask id="javaService"
  name="My Java Service Task"
  camunda:expression="${printer.printMessage()}" />

<bpmn:serviceTask id="javaService"
  name="My Java Service Task"
  camunda:expression="${printer.printMessage(execution, myVar)}" />
```



Camunda Spin Project

Motivation

- Handle serialized formats such as JSON or XML while interacting with external systems.
- Process such data like parsing, manipulating or mapping from/to Java objects.

Features

- Provide data format functionality
- Plugged into the engine
- Wrapper for processing data formats like XML and JSON
- Integrated with the engine's data handling functionality
- Designed to be extensible

<https://docs.camunda.org/manual/latest/user-guide/data-formats/>

<https://docs.camunda.org/manual/latest/reference/spin/>



Spin Example

```
<?xml version="1.0" encoding="UTF-8"?>
<customer xmlns="http://camunda.org/example" name="Jonny">
  <address>
    <street>12 High Street</street>
    <postcode>1234</postcode>
  </address>
</customer>
```

Imagine a response from a web service call

Access in an expression:

```
 ${XML(customer).xpath("/customer/address/postcode").element().textContent() == "1234"}
```

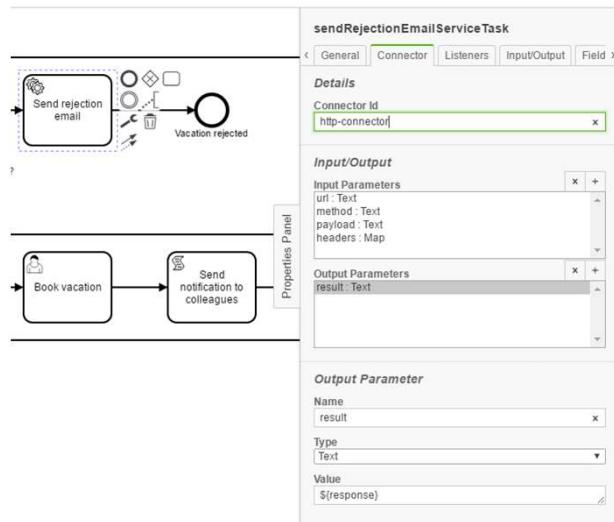
Maybe a condition on a sequence flow



Connectors: Configure Service Task Without Java Code

From the Java developer view:

- No class needed
- All configuration in the BPMN2.0 File
- Scripting for dynamic service calls
- May lead to a lot of different resources
- May be harder to manage than a JavaDelegate
- Difficult to mock during JUnit tests



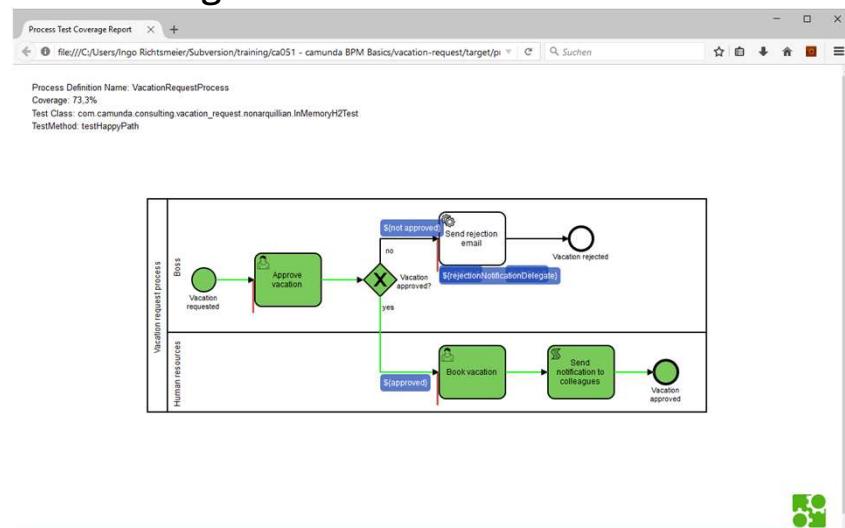
Connectors: Configure Service Task in BPMN2.0 File

From the Script developer view:

- Use expressions or scripts to calculate the parameters of the service call
- Inspect the result with Camunda Spin
- Scripts run in the JVM of the process engine
- Easy access to process variables from the JVM
- Complete control of the service call possible

```
<bpmn:serviceTask id="ServiceTask_1dwrg4e" name="Check mandatory field values" camunda:asyncBefore="true">
  <bpmn:extensionElements>
    <camunda:connector>
      <camunda:inputOutput>
        <camunda:inputParameter name="url">http://localhost:3000/checkmandatory-fields</camunda:inputParameter>
        <camunda:inputParameter name="method">POST</camunda:inputParameter>
        <camunda:inputParameter name="headers">
          <camunda:map>
            <camunda:entry key="Content-Type">application/json</camunda:entry>
          </camunda:map>
        </camunda:inputParameter>
        <camunda:inputParameter name="payLoad">
          <camunda:script scriptFormat="javascript" resource="scripts/generate-service-payload.js" />
        </camunda:inputParameter>
        <camunda:outputParameter name="resultCheckMandatoryFields">
          <camunda:script
            scriptFormat="javaScript">S(response);</camunda:script>
        </camunda:outputParameter>
      </camunda:inputOutput>
      <camunda:connectorId>http-connector</camunda:connectorId>
    </camunda:connector>
  </bpmn:extensionElements>
</bpmn:serviceTask>
```

Process Test Coverage



<https://github.com/camunda/camunda-bpm-process-test-coverage>

Exercise 4



Follow the provided instructions for exercise 4



Improve JUnit with Simple Mocks

1. Make Delegate a Bean with @Named

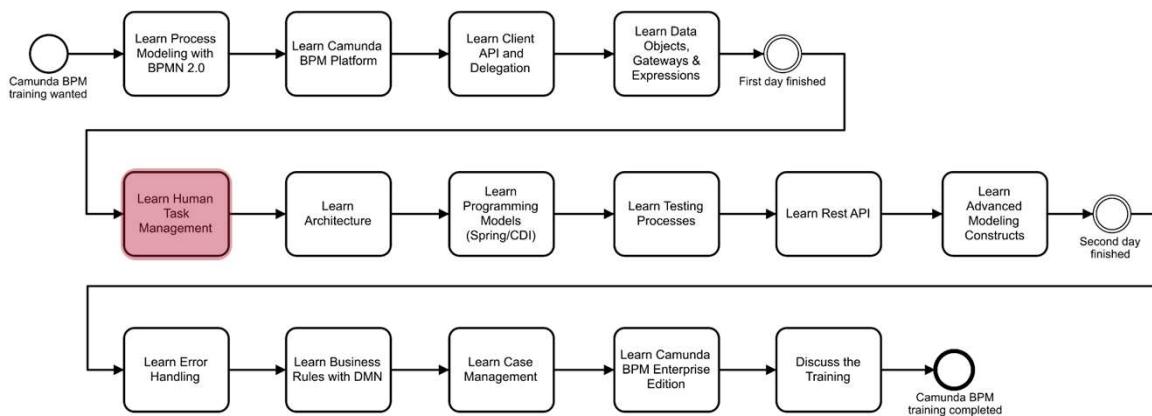
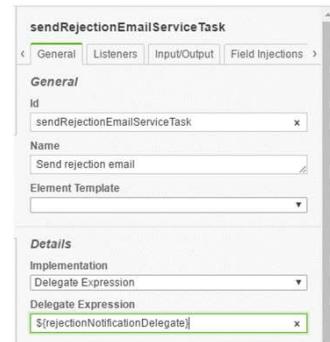
```
@Named("rejectionNotificationDelegate")
public class SendRejectionNotificationDelegate implements JavaDelegate {

    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Send the rejection to the employee...
```

2. Use Indirection with DelegateExpression in bpmn process

3. Use Mocks.register("bean", new MockDelegate()) in the test method

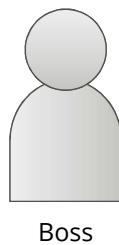
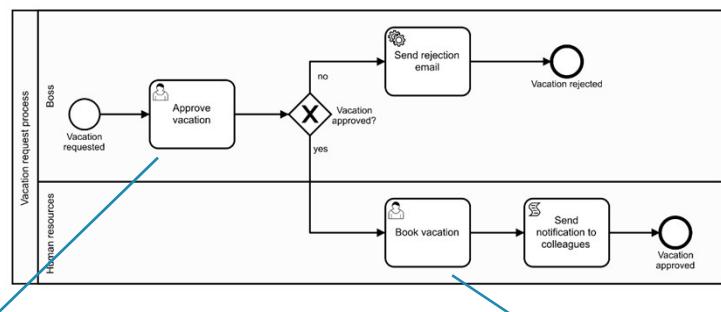
```
@Test
@Deployment(resources = "vacation-request.bpmn")
public void testRejectionVacation() {
    Mocks.register("rejectionNotificationDelegate",
        new LoggerDelegate());
    ...
}
```



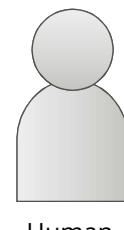


111

Task Management



Boss

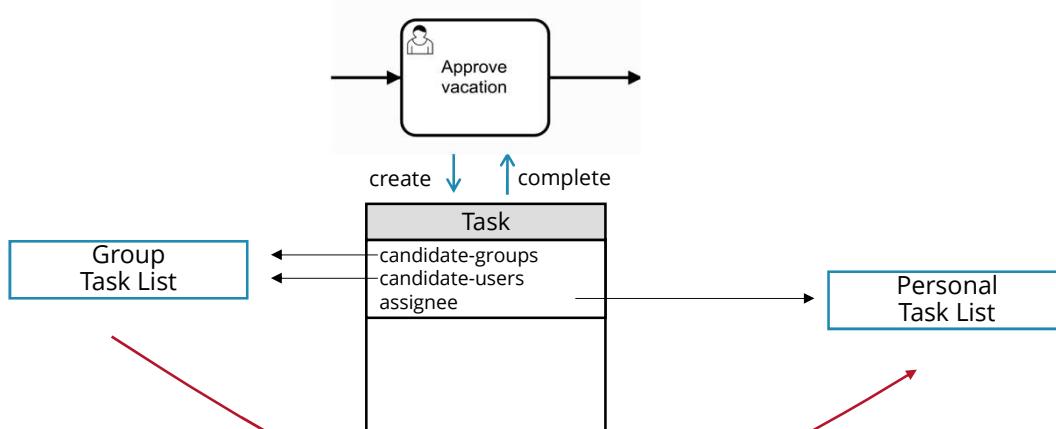


Human Resources



112

Task Management



Tasklist in Camunda BPM

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with 'My Tasks' and 'My Group Tasks (10)'. Step 1 is highlighted with a blue circle containing the number '1'. Step 2 is highlighted with a blue circle containing the number '2'. Step 3 is highlighted with a blue circle containing the number '3'. The main area shows a task titled 'Approve vacation' with details like 'Vacation request' and 'Created a minute ago'. Below it is another task 'Approve Invoice' with details like 'Invoice Receipt' and 'Due 6 days ago, created 13 days ago'. At the bottom right, there are 'Save' and 'Complete' buttons.



Tasks From Different Sources

The diagram illustrates the integration of different task sources within the Camunda platform:

- Processes (BPMN 2.0):** A BPMN process diagram for 'Vacation request process' is shown on the left. It starts with a 'Start' event, followed by a 'Vacation requested' task, an 'Approve vacation' task, a decision diamond 'Vacation approved?', and two outgoing paths: 'no' leading to 'Send rejection email' and 'yes' leading to 'Block vacation' and 'Send notification to colleagues'. Both paths converge back to a final 'Vacation rejected' event.
- AdHoc / API:** A separate box contains a diagram of an 'Invoice Clarification' task, which includes 'Clarify Invoice', 'Assign Reviewer', and 'Review Invoice' steps.
- Cases (CMMN 1.1):** A separate box contains a diagram of an 'Invoice Clarification' case, which includes 'Clarify Invoice', 'Assign Reviewer', and 'Review Invoice' steps.
- Camunda Tasklist Integration:** Arrows from both the AdHoc/API and CMMN boxes point to the Camunda Tasklist interface in the center, indicating that these external tasks can be managed and integrated within the same platform.



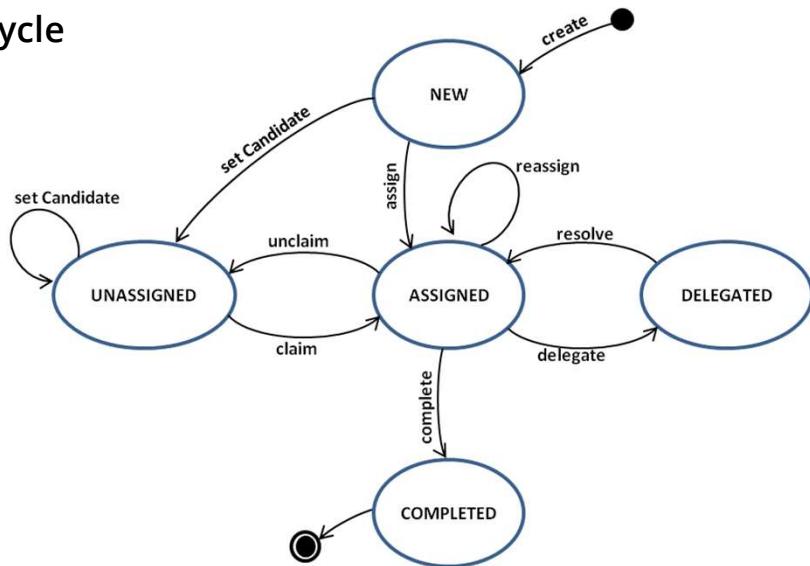
Groups & Personal Task List (Via Filters)

The screenshot shows the Camunda Tasklist interface with the 'Edit filter' dialog open. The dialog has several sections:

- General**: Contains a 'Criteria' section with two entries:
 - 'Unassigned' under 'Key' and '✓' under 'Value'.
 - 'Candidate Groups *' under 'Key' and '\$(currentUserGroups())' under 'Value'.
- Permissions**: A checkbox for 'Include assigned tasks' is checked, with a note explaining it includes tasks assigned to users in candidate queries.
- Variables**: An empty section.
- Buttons**: 'Delete filter', 'Close', and a red 'Save' button.

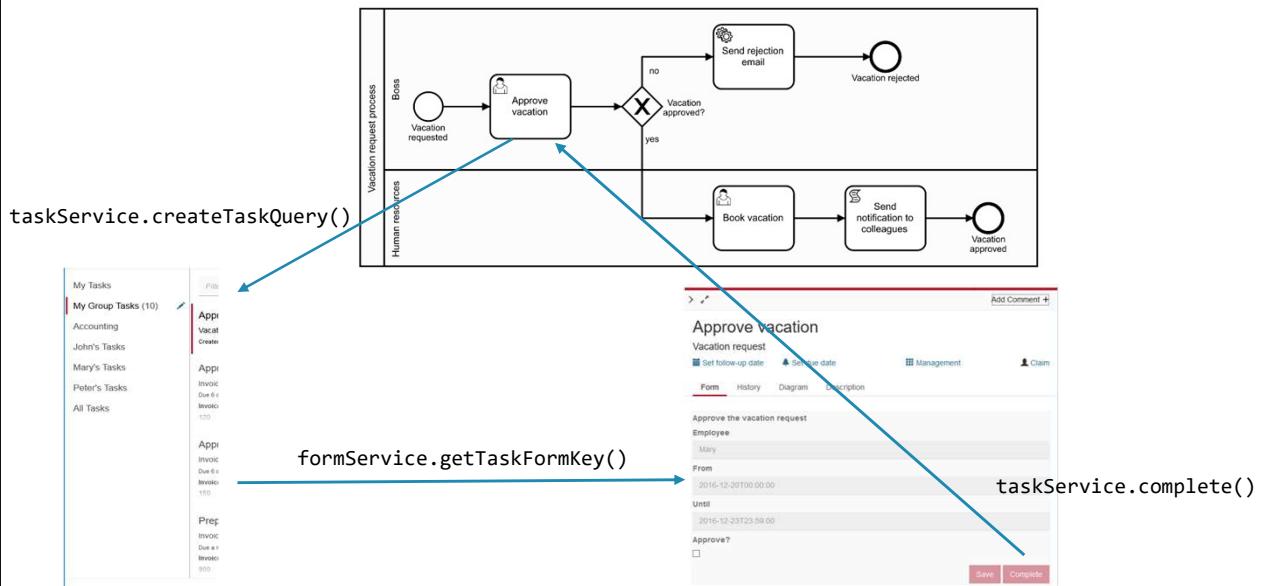


Task Lifecycle





Task Management API



Personal and Group Task Lists

```
TaskService taskService = processEngine.getTaskService();

List<Task> personalTaskList = taskService.createTaskQuery()
    .taskAssignee("demo").list();

List<Task> groupTaskList = taskService.createTaskQuery()
    .taskCandidateUser("demo").list();

groupTaskList.addAll(taskService.createTaskQuery()
    .taskCandidateGroup("marketing").list());
```



Task Query & Completion

```
TaskService taskService = processEngine.getTaskService();

List<Task> personalTaskList = taskService.createTaskQuery()
    .taskAssignee("demo").list();

Task task = personalTaskList.get(0);

Map<String, Object> variables = new HashMap<String, Object>();
variables.put("approved", false);

taskService.complete(task.getId(), variables);
```

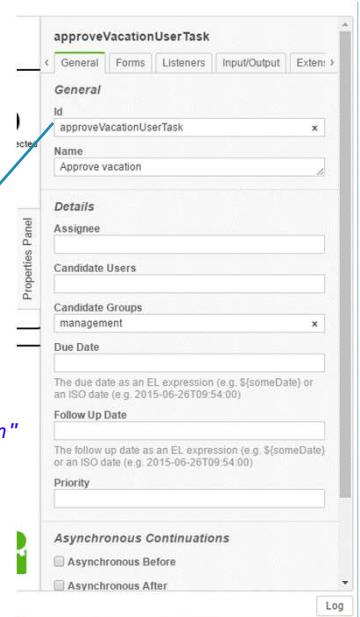


Key vs. ID vs. Name

```
taskService.complete(task.getId(), variables);
```

- `task.getId()`:
- `task.getTaskDefinitionKey()`:
- `task.getName()`:

Task Instance ID
BPMN User Task ID
BPMN User Task Name



```
<bpmn:userTask id="approveVacationUserTask" name="Approve vacation"
    camunda:candidateGroups="management">
</bpmn:userTask>
```



121

Task Listener

Typical Use Cases:

- Send notification e-mail
- Task modifications, e.g. priority, due date, display name
- Vacation substitute
- Create task in third-party application

Available events:

create
assignment
complete
delete

```
public class TaskCreationListener implements TaskListener {  
  
    @Override  
    public void notify(DelegateTask delegateTask) {  
        // send a mail to manager  
    }  
}
```



122

Exercise 5

Follow the provided instructions for exercise 5





Task Forms

The screenshot illustrates a vacation approval process. On the left, a 'Start process' form is shown with fields for 'Your Name', 'Start date', 'Last day of vacation', and a 'Back' button. A blue arrow points from this form to a BPMN diagram. The diagram shows a 'Vacation requested' event leading to a 'Boss' user task labeled 'Approve vacation'. This leads to a decision diamond 'Vacation approved?'. If 'no', it goes to a 'Send rejection email' task. If 'yes', it returns to the 'Boss' task. A red arrow points from the decision diamond back to the 'Boss' task. The background of the diagram is divided into 'Human resources' and 'Boss' regions.

Start process

Here you can request your vacation
Your Name

Start date

Last day of vacation

Back

Vacation requested

Boss

Approve vacation

Vacation approved?

Send rejection email

Human resources

Add Comment +

Approve vacation

Vacation request

Form History Diagram Description Management Claim

Approve the vacation request

Employee: Mary

From: 2016-12-20T00:00:00

Until: 2016-12-23T23:59:00

Approve?

Save Complete

approveVacationUserTask

General Forms Listeners Input/Output Extension

Form Type: Form Key: Form Key: embedded.app.forms/embedded/task-form.html



Task Forms in Camunda Tasklist

Generic form

Second User Task

Generic Form Process

Form History Diagram Description

You can set variables, using a generic form, by clicking the "Add a variable" link below.

Business Key

Add a variable: Name: Type: Value:

Remove: Variable name: Variable type: Value:

Load Variables

Complete

Embedded form

Approve vacation

Vacation request

Form History Diagram Description Management Claim

Approve the vacation request

Employee: Mary

From: 2016-12-20T00:00:00

Until: 2016-12-23T23:59:00

Approve?

Save Complete

Generated form

Start process

Firstname:

Lastname:

Date of Birth:

Back Close Start

External form

Review Result

username: Leftheris-Demertzis@camunda

CaseId: 10000000000000000000000000000000

Process: Review

Comments:

Close



125

Generic Task Forms

- `formKey=""`
- Completely generic, shows all process variables after load
- Add variables manually on demand
- Edit existing variables

Second User Task
Generic Form Process

Set follow-up date Set due date Add groups Demo Demo x

Form History Diagram Description

You can set variables, using a generic form, by clicking the "Add a variable" link below.

Business Key	Name	Type	Value
Add a variable +	Variable name	Variable type	Value
Remove x			

Load Variables ⓘ Complete



126

Generated Task Forms (From Form Fields)

- Form Data Metadata provided by BPMN 2.0 XML
- Rendered in task list

Employee
Jim

Start date
09/11/2016

Last day of vacation
11/11/2016

Approved by
Peter Meter

bookVacationUserTask

Forms

Form Type: Form Data

Form Fields:

- employee
- from
- until
- approvedBy

Form Field

ID: approvedBy

Type: string

Label: Approved by

Default Value:

Validation

Add Constraint +

Name: readonly

Properties



Embedded Task Forms

- formKey="embedded:app:approve-vacation.html"
- HTML-Form provided by Process Application (HTML-File)
- Mix HTML and Angular-JS JavaScript
- Rendered in task list

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with 'My Group Tasks (10)' and categories like Accounting, John's Tasks, Mary's Tasks, Peter's Tasks, and All Tasks. In the main area, a task titled 'Approve vacation' is selected. The task details show it's a 'Vacation request' created a minute ago. Below this, there are other tasks: 'Approve Invoice' (Invoice Receipt due 6 days ago), 'Approve Invoice' (Invoice Receipt due 6 days ago, created 13 days ago), and 'Prepare Bank Transfer' (Invoice Receipt due a month ago, created a month ago). On the right, a modal window titled 'Approve vacation' is open, containing fields for 'Employee' (Mary), 'From' (2016-12-29T00:00:00), 'Until' (2016-12-23T23:59:00), and 'Approve?' (checkbox). At the bottom of the modal are 'Save' and 'Complete' buttons.



External Task Form

- formKey="app:reviewTweet.jsf"
- Params taskId and callbackURL needed

The screenshot shows a web browser window with the title 'Aufgabe: Review Tweet'. The URL in the address bar is 'localhost:8080/twitter/reviewTweet.jsf?taskId=d0fb4562-a59f-11e6-8382-185e0f710ea&callbackUrl=http://...'. The page content is a form titled 'Review Tweet'. It contains the following fields:

- Author: ingo
- E-Mail: igo.nichtsmeier@camunda.com
- Content: Tweet for Training Slides
- Approve? (checkbox)
- Comments: (text area)
- Submit button



Embedded Task Forms Form SDK

- **Form handling:** Attach to a form existing in the DOM or load a form from a URL.
- **Variable handling:** Load and submit variables used in the form.
- **Script handling:** Execute custom JavaScript in Forms
- **AngularJS Integration:** The Forms SDK optionally integrates with AngularJS to take advantage of AngularJS form validation and other AngularJS goodies.

cam-variable-name: Bind input to the model

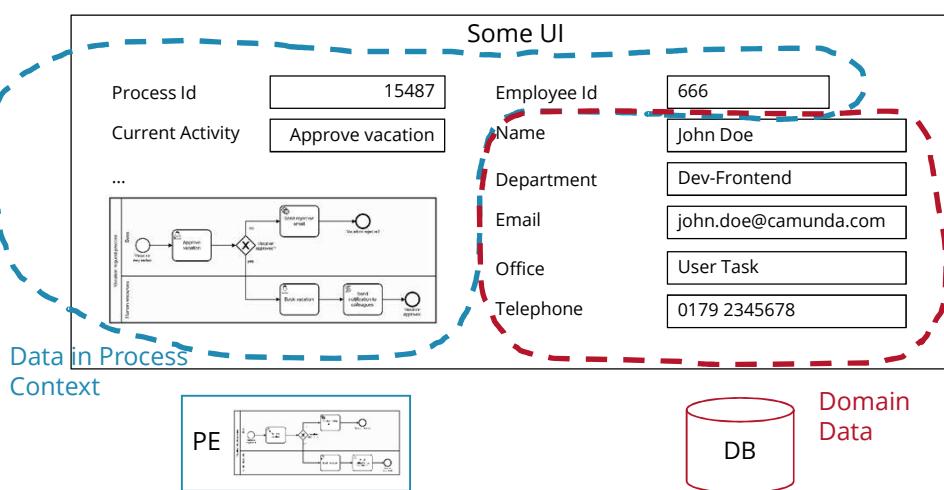
cam-variable-type: Type of process variable

Form templates are generated from the archetypes

<https://docs.camunda.org/manual/latest/reference/embedded-forms/>



Typical: Process Data and External Data Are Mixed

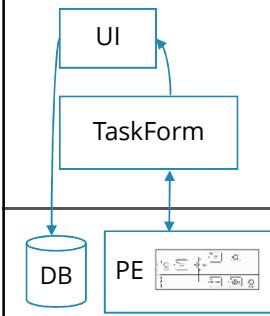




Alternatives to Call Data Service

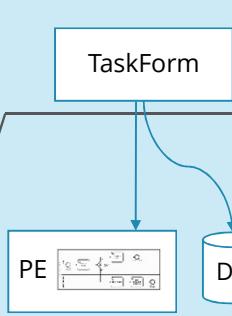
Alternative 1: Separate UI

- Use existing UI
- Started manually or via link in the Task Form
- Completely independent



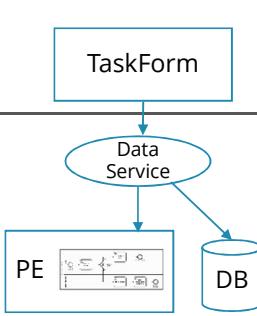
Alternative 2: UI Integration

- The UI calls the Data Service and the Process Service
- No TX



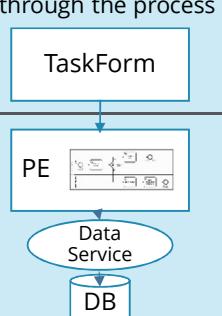
Alternative 3: Composed Service

- The UI calls a combined service
- This service calls the Data Service and the Process Service
- TX possible



Alternative 4: Process Service

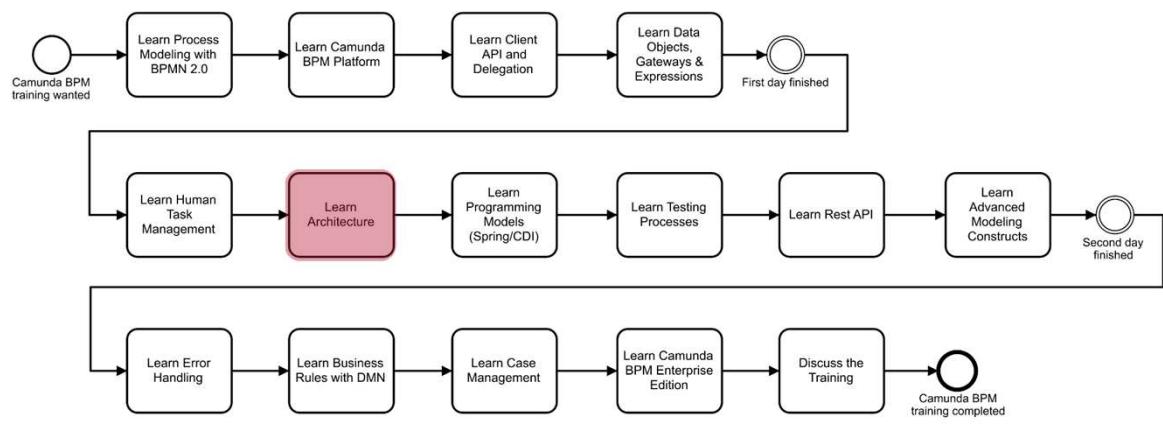
- The UI calls the Process Service
- The Process calls the Data Service
- Full control, but data has to be transported through the process



Exercise 6

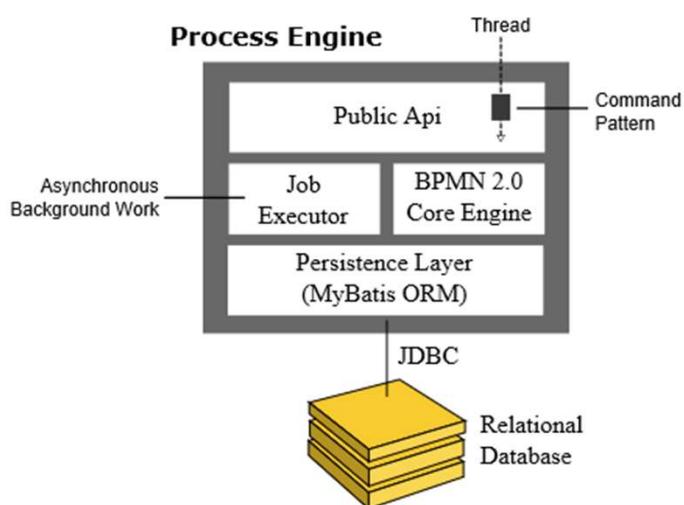
Follow the provided
instructions for exercise 6





134

Process Engine Architecture





Starting a Process Instance via REST

1. HTTP POST to /engine-rest/process-definition/key/my-process/start
2. ProcessEngine.getRuntimeService()
3. RuntimeServiceImpl.startProcessInstanceByKey(...)
4. commandExecutor.execute(new StartProcessInstanceCmd(...))
5. CommandInterceptors (e.g. TransactionInterceptor)
6. StartProcessInstanceCmd.execute(commandContext)
7. Internal API composed of so-called Managers:
 1. DeploymentCache.findDeployedLatestProcessDefinitionByKey(...)
 2. processDefinition.createProcessInstance(...)
 3. processInstance.start(variables)
8. AtomicOperations (e.g. start process, execute activity, take transition)
9. ActivityBehaviours implement actual BPMN execution semantics
10. When no token can move further, CommandInterceptors store state changes in DB and commit transaction
11. HTTP response is sent



Start an Engine in Pure Java

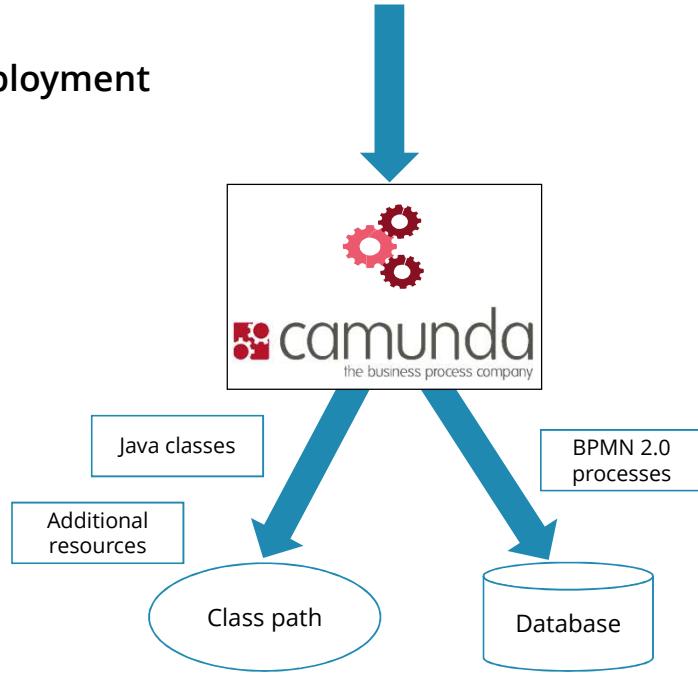
```
ProcessEngine processEngine =
ProcessEngineConfiguration.createStandaloneInMemProcessEngineConfiguration()
.setDatabaseSchemaUpdate(ProcessEngineConfiguration.DB_SCHEMA_UPDATE_FALSE)
.setJdbcUrl("jdbc:h2:mem:my-own-db;DB_CLOSE_DELAY=1000")
.setJobExecutorActivate("true")
.buildProcessEngine();

processEngine.getRepositoryService().createDeployment()
.addClasspathResource("jax.bpmn").deploy();
```



137

Process Deployment



138

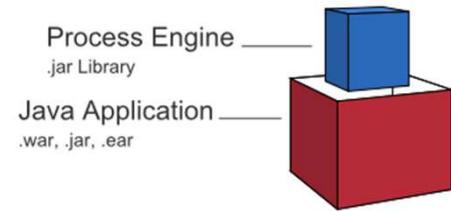
Deploying Programmatically With Java

```
repositoryService.createDeployment()  
    .addClasspathResource("process.bpmn")  
    .enableDuplicateFiltering(true)  
    .deploy();
```



139

Embedded Process Engine



Various environments possible:



Basic question: Who controls the lifecycle of the engine (startup / shutdown)?



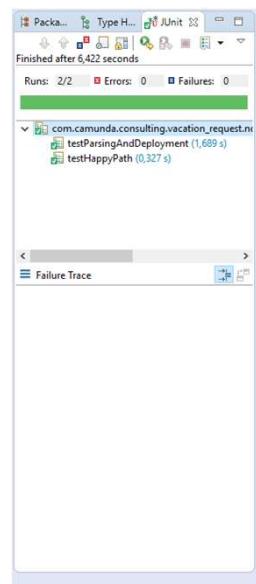
140

Example: Unit Test

```
import static org.camunda.bpm.engine.test.assertions.ProcessEngineTests.*;
public class VacationRequestProcessTest {

    @Rule
    public ProcessEngineRule rule = new ProcessEngineRule();

    @Test
    @Deployment(resources = "vacation-request.bpmn")
    public void testHappyPath() {
        ProcessInstance processInstance = runtimeService()
            .startProcessInstanceByKey(PROCESS_DEFINITION_KEY);
        assertThat(processInstance).isWaitingAt("approveVacationUserTask");
        complete(task(), withVariables("approved", true));
        assertThat(processInstance).isWaitingAt("bookVacationUserTask");
        complete(task());
        assertThat(processInstance).isEnded();
    }
}
```





Example: Spring

```
ClassPathXmlApplicationContext applicationContext =
    new ClassPathXmlApplicationContext("/spring-context.xml");
ProcessEngine processEngine = (ProcessEngine) applicationContext.getBean("processEngine");

processEngine.getRepositoryService().createDeployment()
    .addClasspathResource("vacation-request.bpmn").deploy();

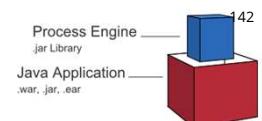
processEngine.getRuntimeService().startProcessInstanceByKey("vacation-request-process");

Task task = processEngine.getTaskService().createTaskQuery().taskAssignee("john").singleResult();

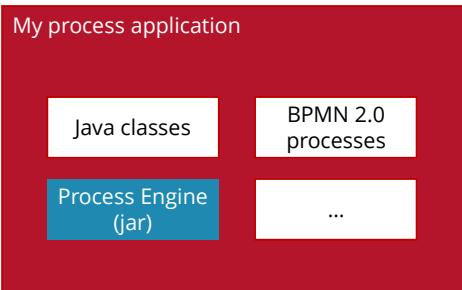
//Simulate task completion
processEngine.getTaskService().complete(task.getId());
```

Sample Spring Configuration:

<https://docs.camunda.org/get-started/spring/embedded-process-engine/>

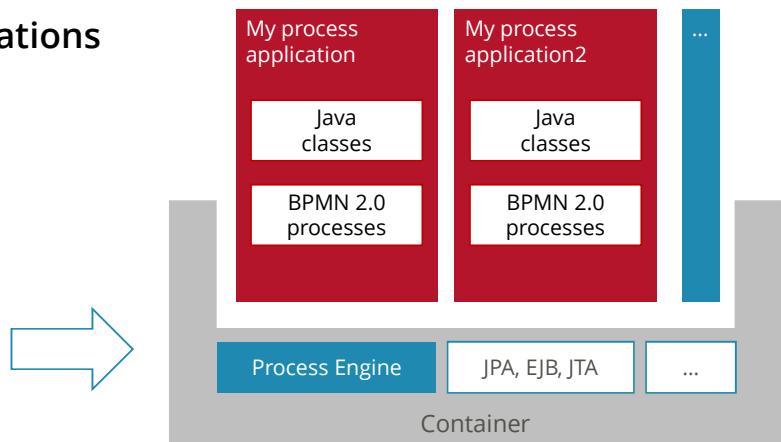
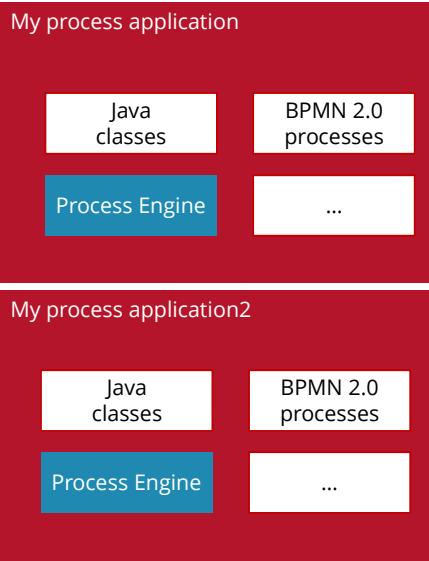


One Process Application



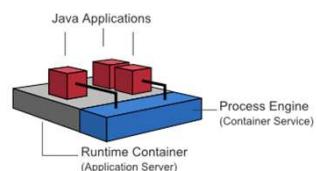


Multiple Process Applications



Supported on:

Tomcat, JBoss AS/EAP, Wildfly AS, WebSphere AS and WebLogic



Camunda Process Application: processes.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<process-application
    xmlns="http://www.camunda.org/schema/1.0/ProcessApplication"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

    <process-archive>
        <process-engine>default</process-engine>
        <properties>
            <property name="isDeleteUponUndeploy">false</property>
            <property name="isScanForProcessDefinitions">true</property>
        </properties>
    </process-archive>

</process-application>
```

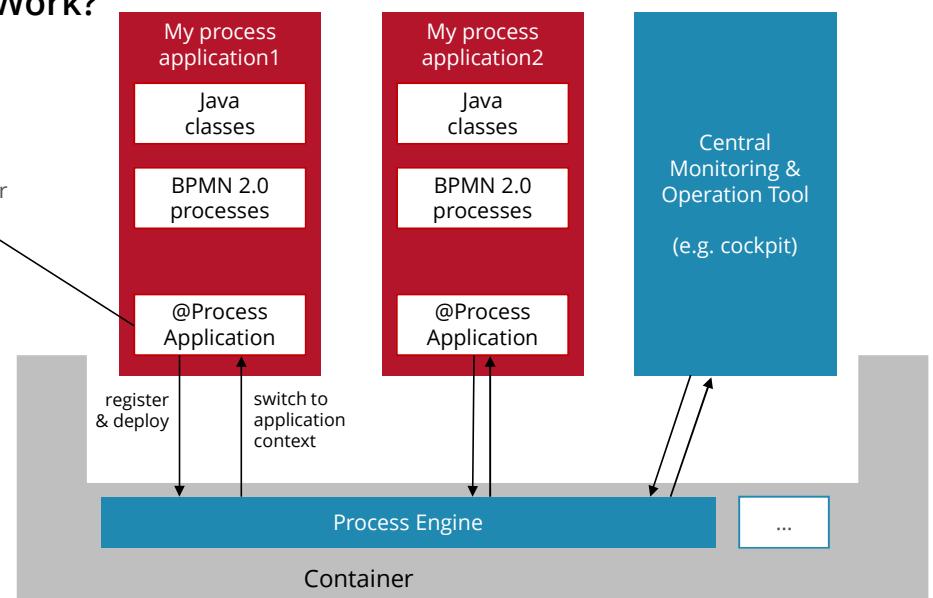


145

How Does This Work?

Might be

- ServletContextListener
- @Startup-EJB
- Spring-Bean
- ...

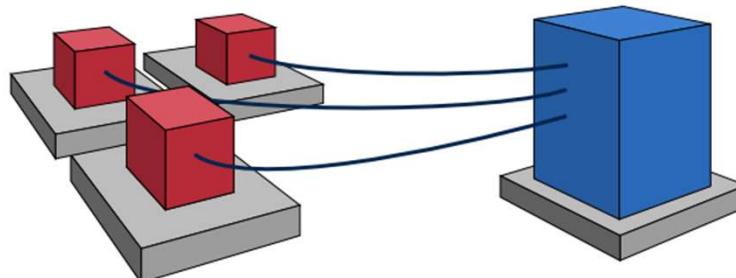


146

Standalone / Remote Process Engine Server

Remote Applications
Communication via Rest Webservices

Standalone
Process Engine Server

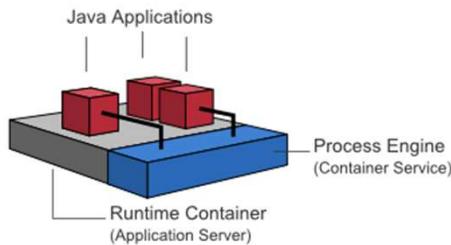




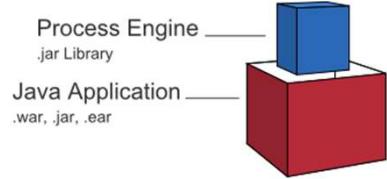
147

Process Engine Modes

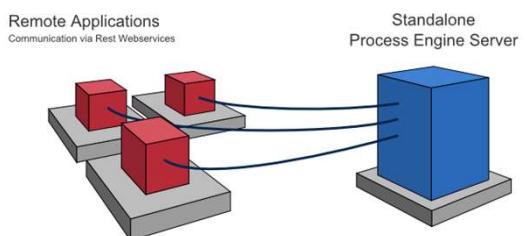
Shared, Container Managed Process Engine



Embedded Process Engine

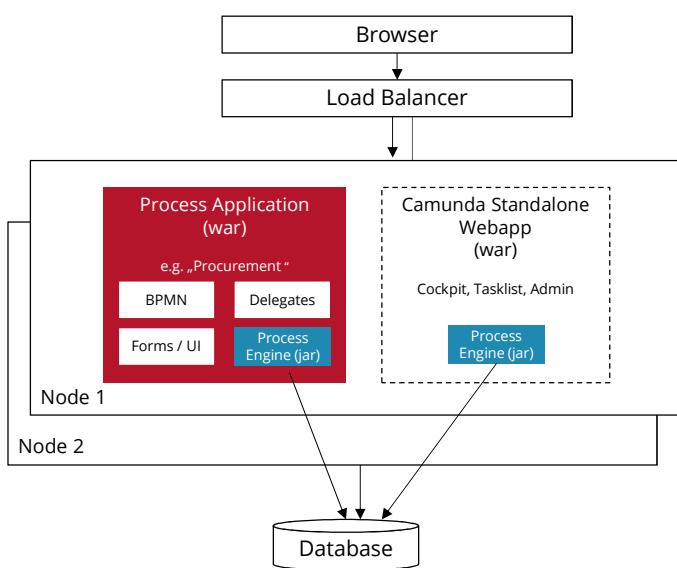


Remote Process Engine



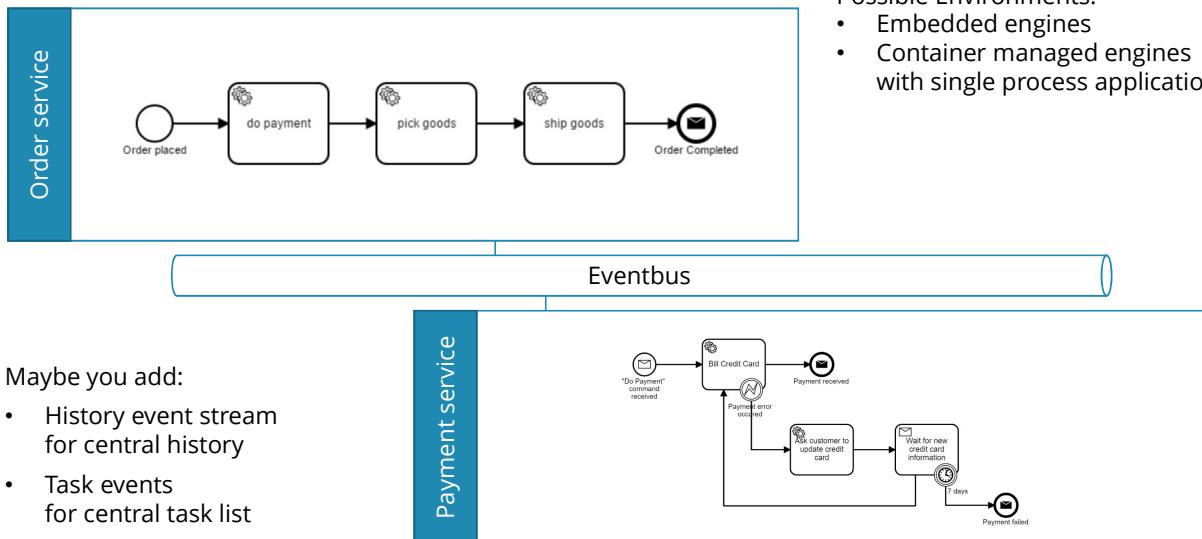
148

Example Architecture With Embedded Process Engines

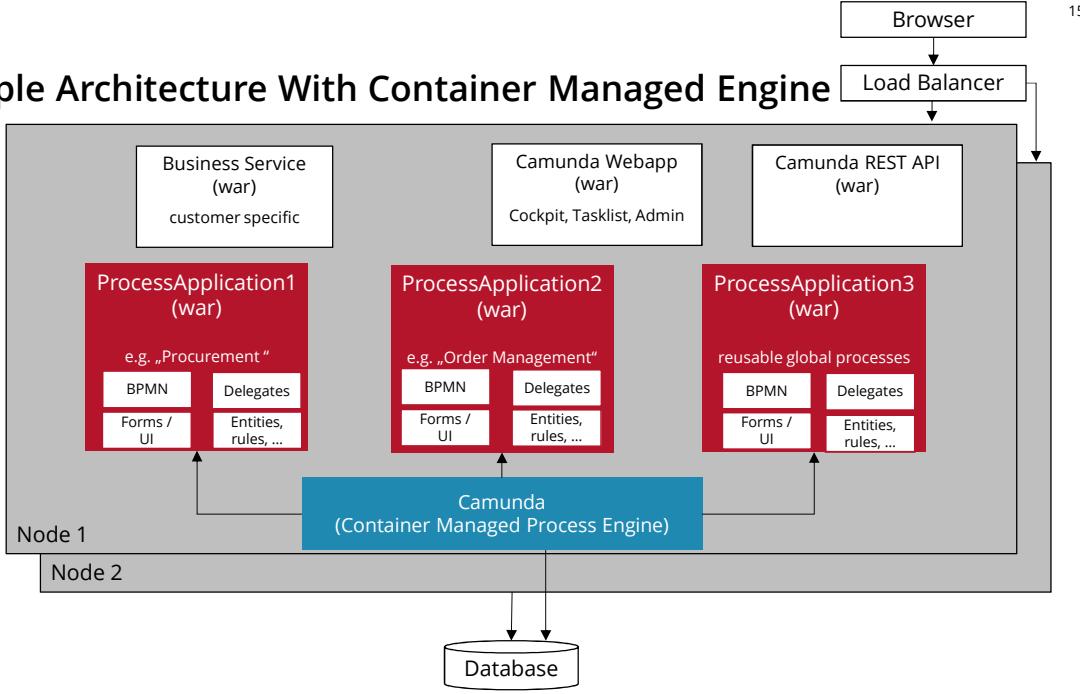




Orchestration With Microservices

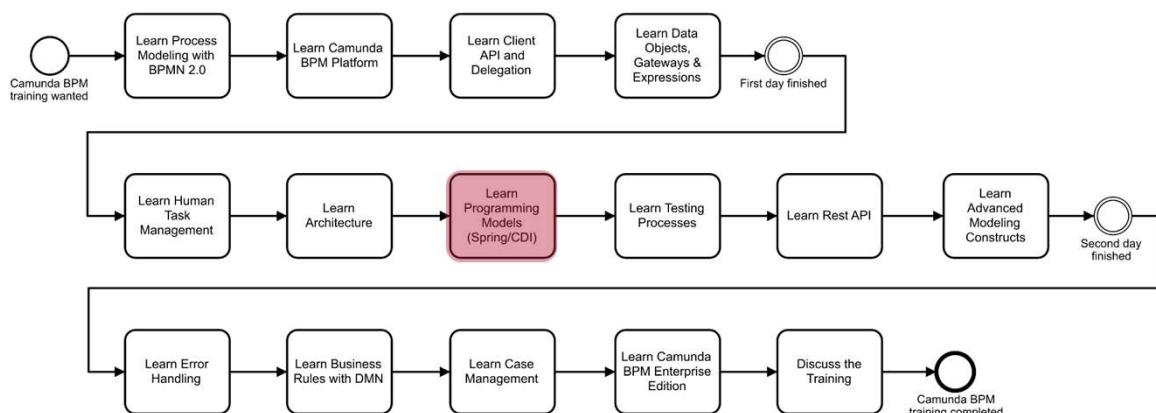
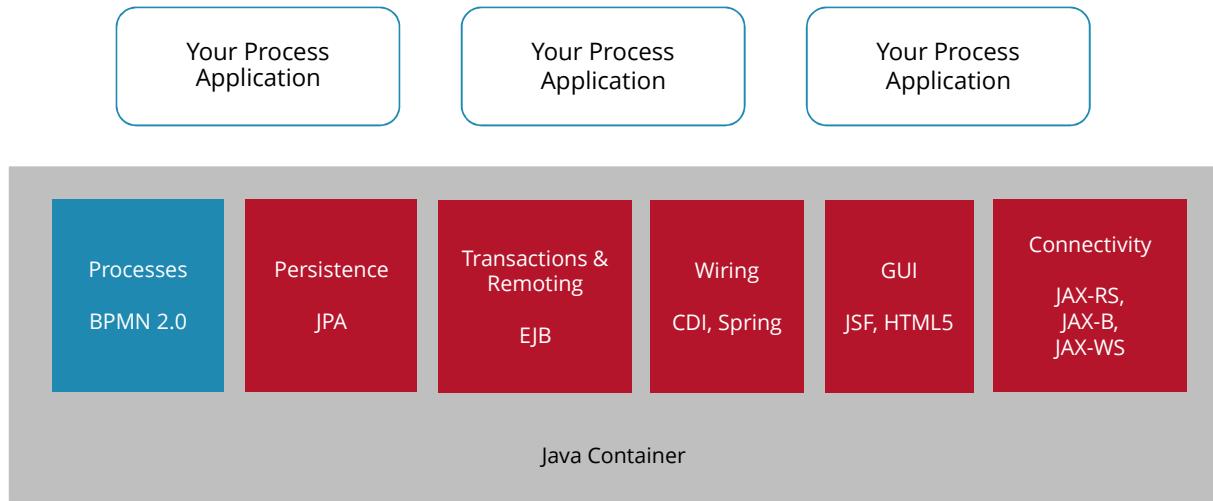


Example Architecture With Container Managed Engine





Process as „First Class Citizen“ in Development Stack





Camunda BPM Provides

Spring Integration

- Control lifecycle of process engine via Spring
- Inject engine and services via Spring
- Call named beans via Expression Language (works with container managed engine too)
- Limited TX integration in container managed engine on Tomcat (without JTA)

CDI Integration

- Inject engine and services via CDI
- Call named beans via Expression Language (works with container managed engine too)
- Contextual Process Implementation and CDI Events (see next slide)



Camunda BPM CDI Integration

basic

Access Camunda BPM

```
@Inject RuntimeService  
@Inject TaskService ...
```

Access your beans

```
<serviceTask id="taskId" name="service"  
    camunda:expression="${service.method(param)}" > ← CDI bean  
</serviceTask>
```

advanced

Contextual Process Implementation

```
@CompleteTask, @StartProcess  
@Inject BusinessProcess  
@BusinessProcessScoped  
@Inject Map<String, Object> processVariables  
...
```



Obtaining Engine Services Through Dependency Injection

Inject Process Engine Services
into Spring / CDI / EJB Beans

```
@Inject  
private RuntimeService runtimeService;  
  
@POST  
@Consumes(MediaType.APPLICATION_JSON)  
public void bookingRequest(Booking booking) {  
    runtimeService.startProcessInstanceByMessage("bookingRequest");  
}
```



Calling Your Names

The @Named qualifier allows to give a bean an Expression Language name:

```
@Named  
public class SimpleGreeter implements Greeter,  
Serializable { ... }
```

The EI-Name of this bean is "simpleGreeter". This allows us to invoke the bean from an Camunda BPM process:

```
<serviceTask id="taskId" name="service"  
camunda:expression="${simpleGreeter.greet('you')}">  
</serviceTask>
```

The resolved object is managed by the container (Spring/CDI) and can profit from other services (injection / lifecycle / ...).

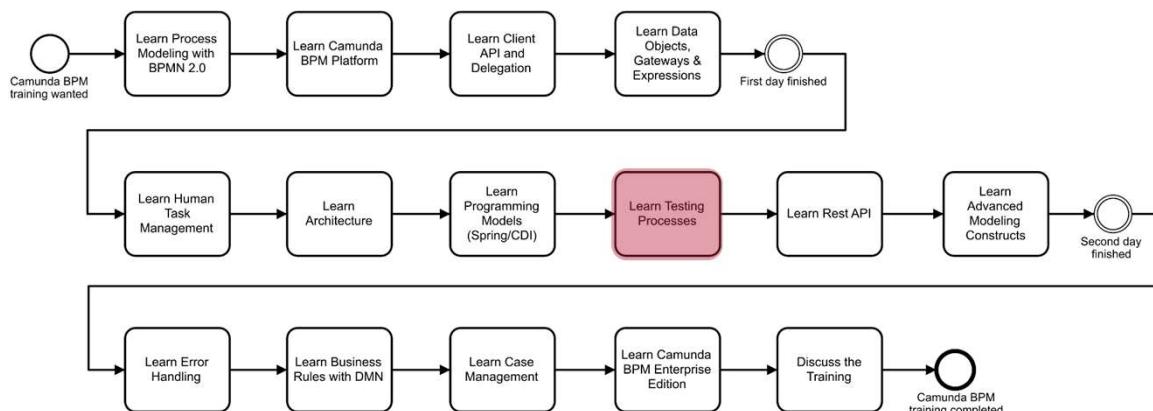


Delegate Expression

Details	
Implementation	Delegate Expression
Delegate Expression	<input type="text" value="\${bookingService}"/> x

```
@Named("bookingService")
public class BookingService implements JavaDelegate {
    private final static Logger LOG = Logger.getLogger(BookingService.class.getName());

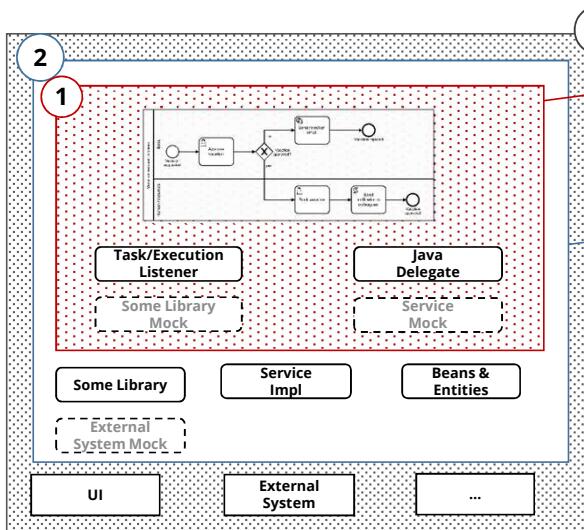
    public void execute(DelegateExecution execution) throws Exception {
        try {
            // perform booking
        } catch (Exception e) {
            LOG.log(Level.WARNING, "Exception while performing booking", e);
            throw new BpmnError("booking error");
        }
    }
}
```





159

Testing Scopes



Goal: Process Model with Data, EL & Adapter Logic

- In Memory
- Single Thread
- No Container
- No JobExecutor

Goal: Close to Real-Life environment

- Container
- Arquillian & co

Goal: It REALLY works

- Integration Tests
- Human Driven



160

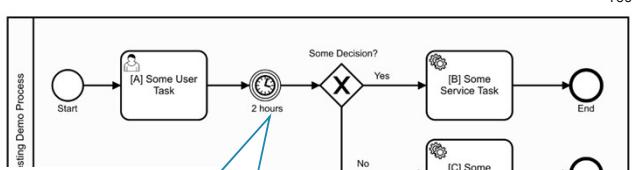
Example

```
@Named
public class CustomerServiceAdapter implements JavaDelegate {
    @Inject
    CustomerService customerService;

    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Input Mapping
        Customer customer = new Customer();
        customer.setName((String) execution.getVariable("customerName"));

        // Service call
        String customerId = customerService.createCustomer(customer);

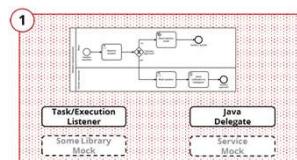
        // Output Mapping
        execution.setVariable("customerId", customerId);
    }
}
```



execute via API
execute(job());

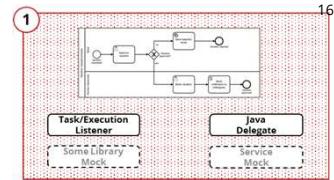
use mocks

This is what we want to mock





Testing on Scope 1



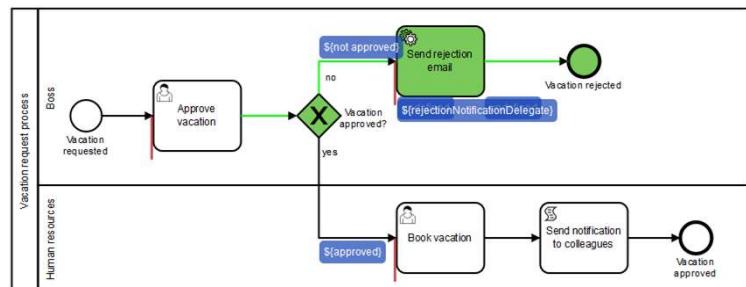
We recommend

- JUnit
- Camunda Rule (included in camunda-engine)
- Camunda-bpmn-assert (<https://github.com/camunda/camunda-bpmn-assert>)
- Mock-Provider of your choice, e.g. PowerMock + Mockito (or e.g. EasyMock or Jmockit)
- Process Instance Modification to start before the unit to test

Benefits

- In Memory Process Engine: Fast & always clean
- Single Thread / No Job Executor: Much easier to understand, no timing issues
- No Container: Less environment, faster turnaround times
- Runs quickly on every developer machine! By "right-click -> Run as Junit"!

Real Unit Test



```

@Test
@Deployment(resources = "vacation-request.bpmn")
public void testVacationNotApproved() {
    ProcessInstance processInstance = runtimeService()
        .createProcessInstanceByKey("VacationRequestProcess")
        .setVariables(withVariables("employee", "John",
            "from", new Date(),
            "to", new Date(),
            "approved", false))
        .startAfterActivity("approveVacationUserTask")
        .execute();

    assertThat(processInstance).isEnded().hasPassed("vacationRejectedEndEvent");
}

```

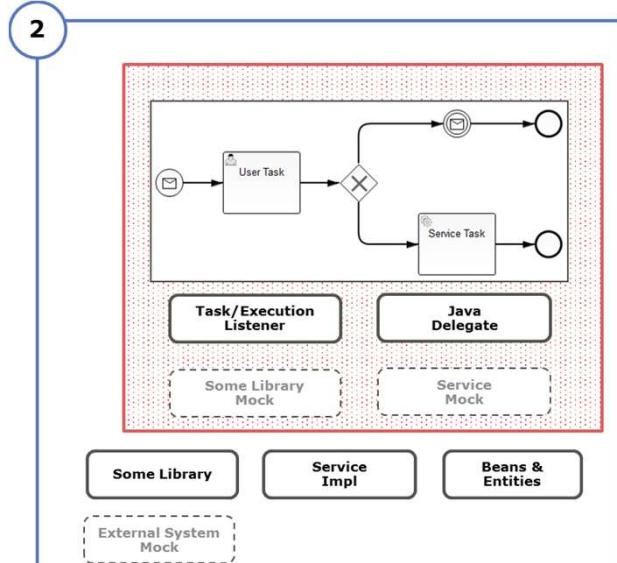
Process Instance Modification

162

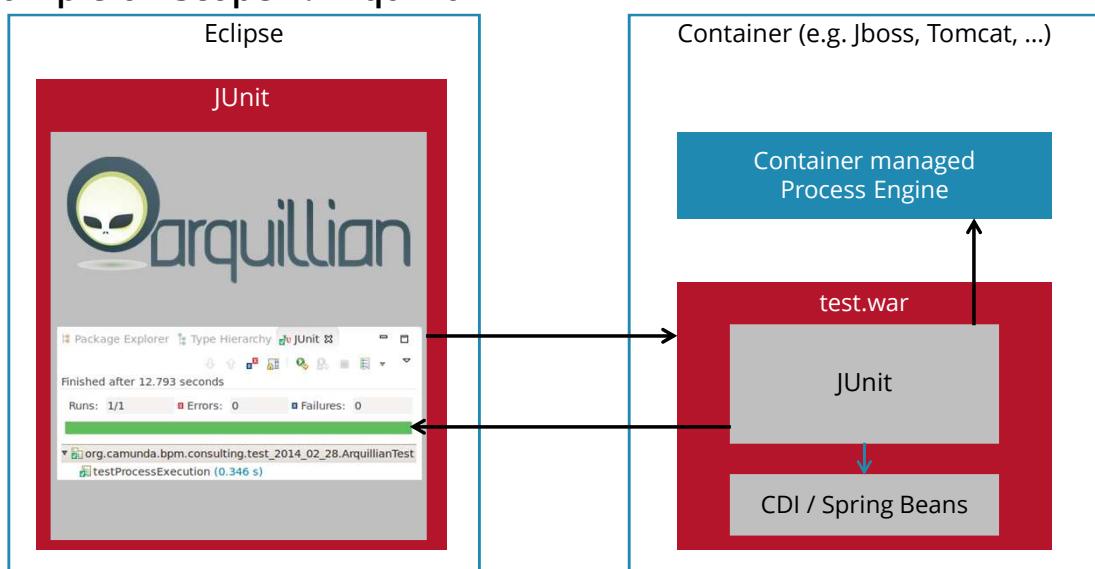


Testing on Scope 2

- Infrastructure needed (e.g. CDI, Transactions, JPA, ...)
- Possible Alternatives
 - Start own test environment
 - Minimalistic (Needle, ...)
 - Real (Weld, Hibernate, OpenEJB, ...)
 - Often Spring-Driven
 - Run tests as integration tests
 - Arquillian
- Forces:
Effort for setup & maintenance, effort to run, possibility to run on developer machine



Example on Scope 2: Arquillian



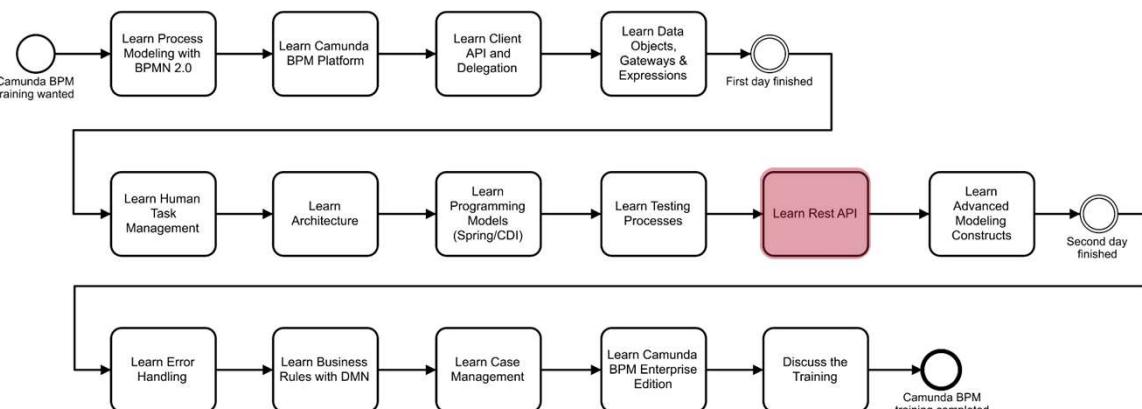


Exercise 7

Follow the provided instructions for exercise 7



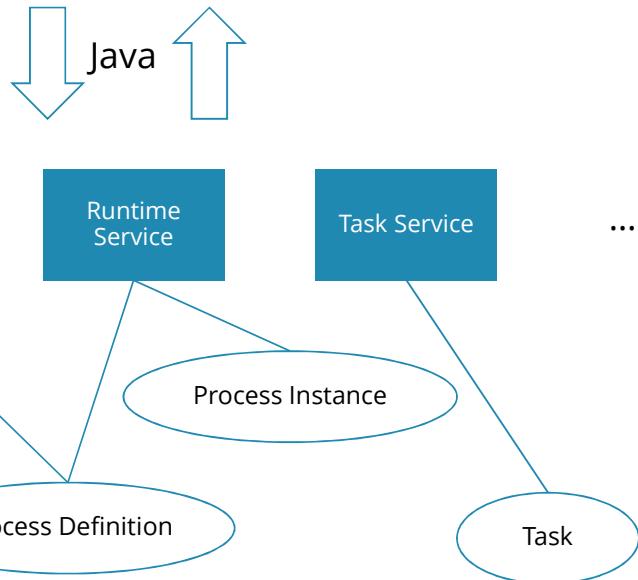
Camunda BPM training wanted





167

Java API



168

REST API



- Same power as Java API



Example Usage Scenarios

your custom
tasklist



your mobile
app



REST
API

Tasklist



Cockpit

process engine



Camunda BPM REST API

Some REST calls:

```
GET http://localhost:8080/engine-rest/process-definition
POST http://localhost:8080/engine-rest/process-definition/key/VacationRequestProcess/start

{"variables":
  {"employee" : {"value" : "Mary", "type": "String"},
   "from" : {"value" : "2016-12-20T00:00:00", "type": "Date"},
   "until" : {"value" : "2016-12-28T23:59:00", "type": "Date"}
 }
}
GET http://localhost:8080/engine-rest/task?processDefinitionKey=VacationRequestProcess
```

<https://docs.camunda.org/manual/latest/reference/rest/>

Documentation

The screenshot shows a browser window displaying the Camunda Documentation website. The URL is https://docs.camunda.org/manual/latest/reference/rest/history/process-instance/get-process-instance-query-count/. The page title is "Get Process Instances Count". On the left, there is a navigation sidebar with a tree structure under "Manual: latest (7.6)". The main content area contains sections for "Method" (with a link to GET /history/process-instance/count), "Parameters", and "Query Parameters". A callout bubble points to the "Method" section with the text "Most important for beginners". On the right side of the page, there is a sidebar with links to "ON THIS PAGE: Method, Parameters, Result, Response Codes, Example". Below this, there is a promotional message: "Get up to 24/7 support with the Camunda Enterprise Edition". At the bottom of the page, there is a footer with the text "camunda.org and docs.camunda.org are part of camunda BPM | Built by camunda and contributors — Privacy Statement — camunda Services GmbH © 2015".

REST API: Embed and Extend!

- REST API as a library
- Own JAX-RS implementation
- Extension/limitation of resources possible

```
@ApplicationPath("/")
public class MyApplication extends Application {
    @Override
    public Set<Class<?>> getClasses() {
        Set<Class<?>> classes = new HashSet<Class<?>>();
        // add your own classes
        // add camunda engine rest classes that you need
        classes.add(ProcessEngineRestServiceImpl.class);
        classes.add(ProcessDefinitionRestServiceImpl.class);
    }
}
```

My process application

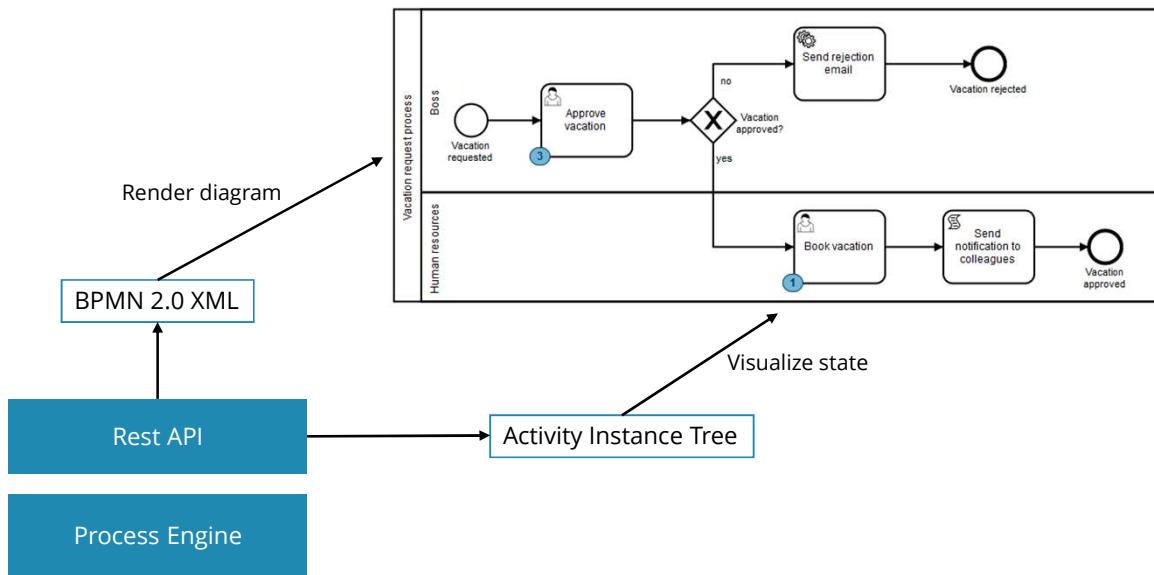
My JAX-RS web application

My favorite JAX-RS implementation

REST API (jar)



REST API for Process Visualization



Two REST APIs in the Distribution

engine-rest.war

Basic HTTP Authentication

check web-inf.xml

```
<!-- Http Basic Authentication Filter -->
<!-- <filter>
    <filter-name>camunda-auth</filter-name>
    ...
</filter-mapping> -->
```

Remove the comment before go to production!

embedded in camunda-webapp.war

Cookie based authentication:

POST

<http://localhost:8080/camunda/api/admin/auth/user/default/login/cockpit>

```
-H "Content-Type: application/x-www-form-urlencoded"
-H "Accept: application/json"
-d 'username=demo&password=demo'
```

POST <http://localhost:8080/camunda/api/admin/auth/user/default/login/cockpit>

Headers (2):

- Authorization
- Content-Type: application/x-www-form-urlencoded

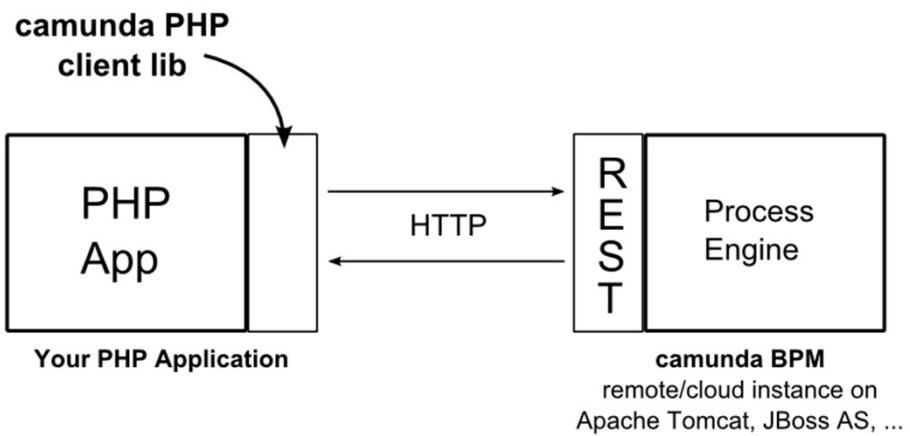
Body:

- form-data
- x-www-form-urlencoded
- raw
- binary

key	value
username	demo
password	demo



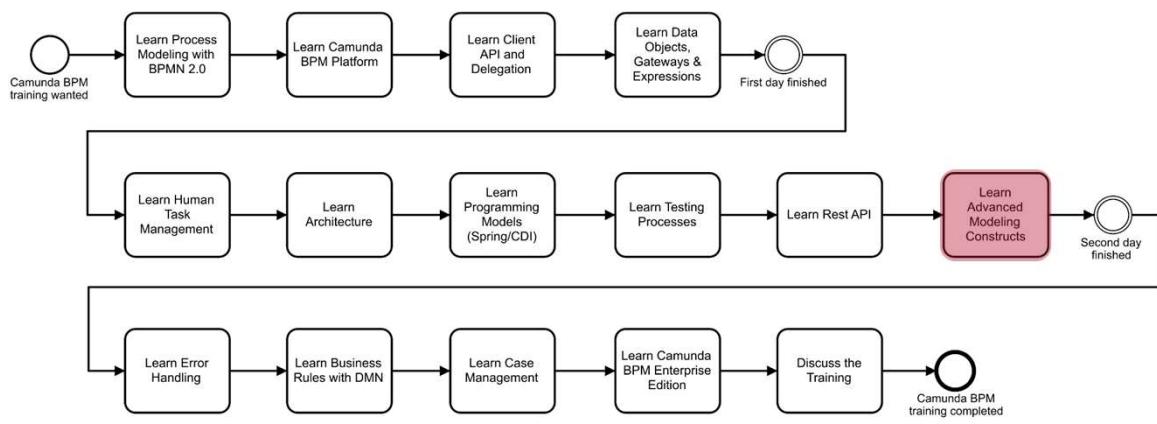
REST API Usage in Camunda PHP SDK



Exercise 8

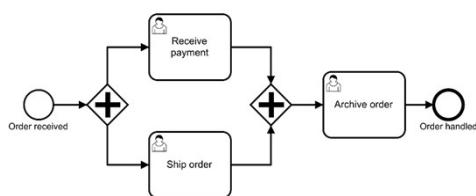
Follow the provided instructions for exercise 8





178

Parallel Gateway



```
<startEvent id="theStart">
<sequenceFlow id="flow1" sourceRef="theStart" targetRef="fork" />

<parallelGateway id="fork" />
<sequenceFlow sourceRef="fork" targetRef="receivePayment" />
<sequenceFlow sourceRef="fork" targetRef="shipOrder" />

<userTask id="receivePayment" name="Receive Payment" />
<sequenceFlow sourceRef="receivePayment" targetRef="join" />

<userTask id="shipOrder" name="Ship Order" />
<sequenceFlow sourceRef="shipOrder" targetRef="join" />

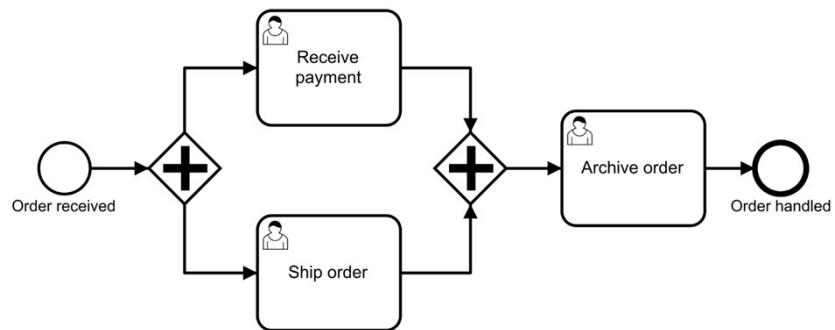
<parallelGateway id="join" />
<sequenceFlow sourceRef="join" targetRef="archiveOrder" />

<userTask id="archiveOrder" name="Archive Order" />
<sequenceFlow sourceRef="archiveOrder" targetRef="theEnd" />

<endEvent id="theEnd" />
```



How Many Transactions?



How Many Transactions?

Hidden solution



181

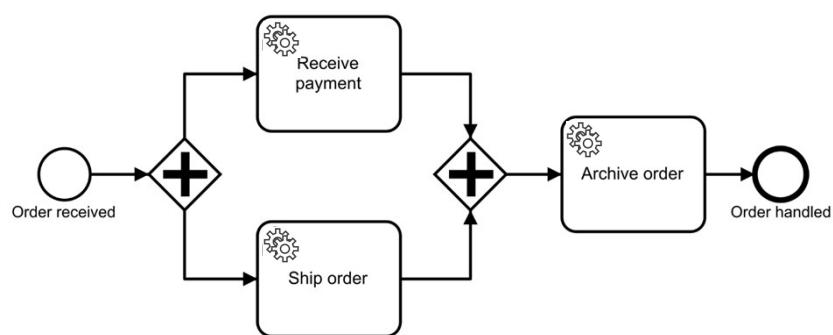
How Many Transactions? (The Other Way Around)

Hidden solution



182

How Many Transactions?



What if all tasks were Service Tasks?



183

How Many Transactions?

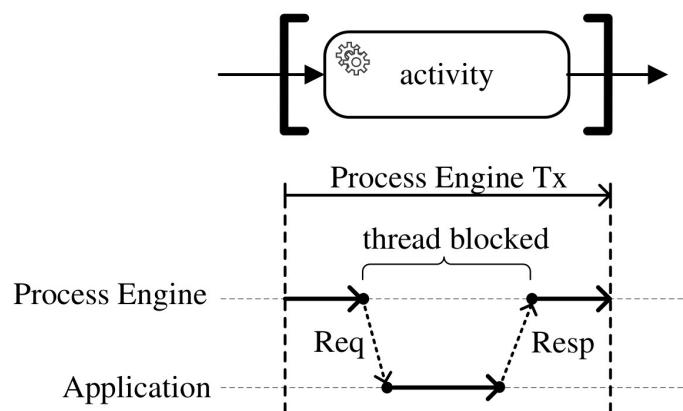
Hidden solution

What if all tasks were Service Tasks?



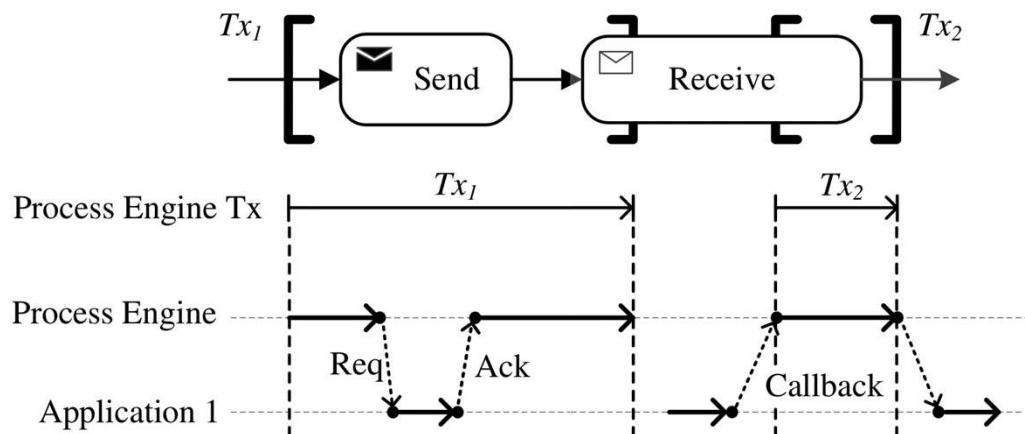
184

Synchronous Service Invocation: Request / Response

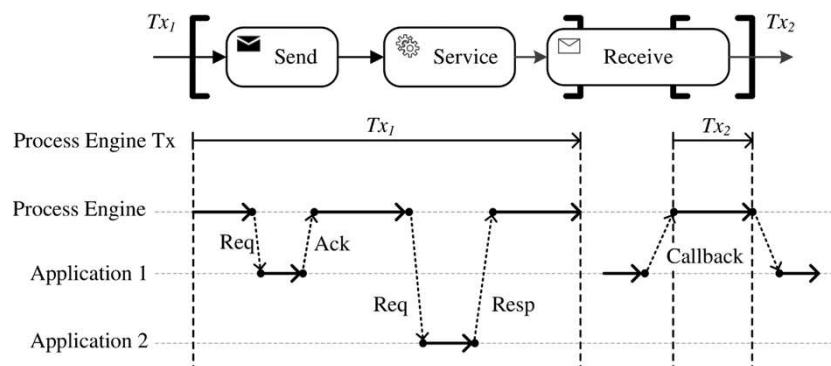




Asynchronous Service Invocation: Req / Ack / Callback

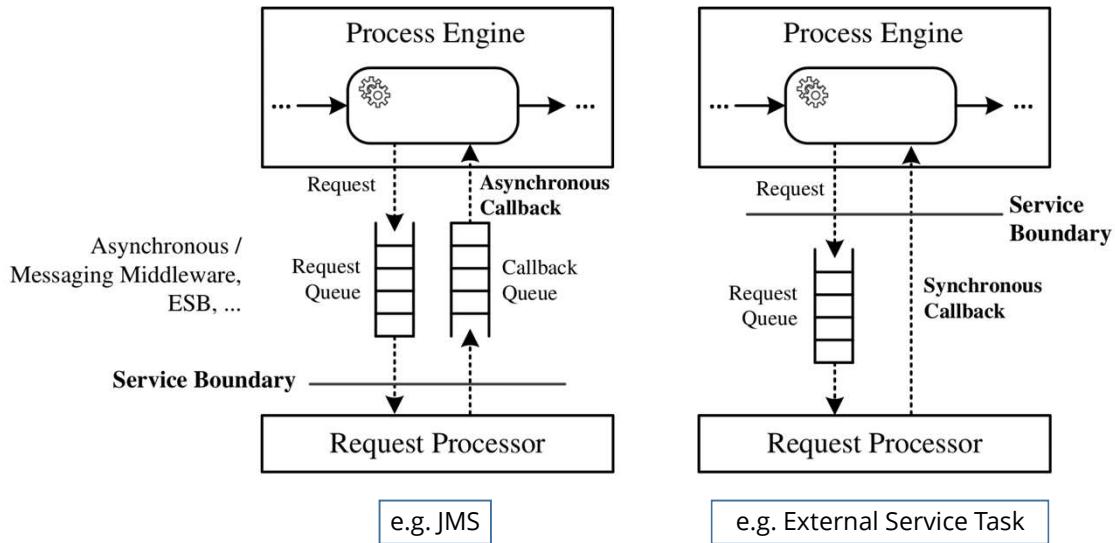


Asynchronous & Synchronous Invocation Combined

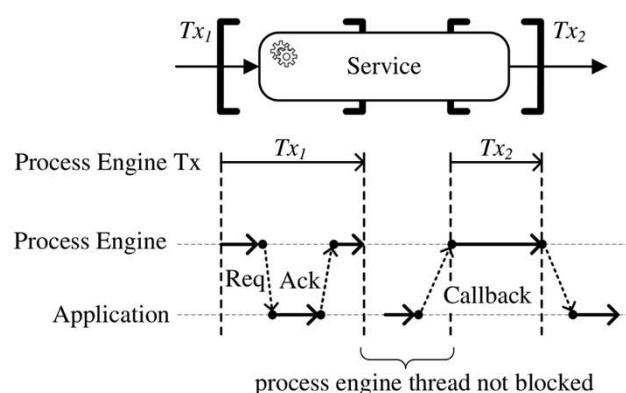




Queuing and Callbacks



External / Asynchronous Service Task



External Service Task:

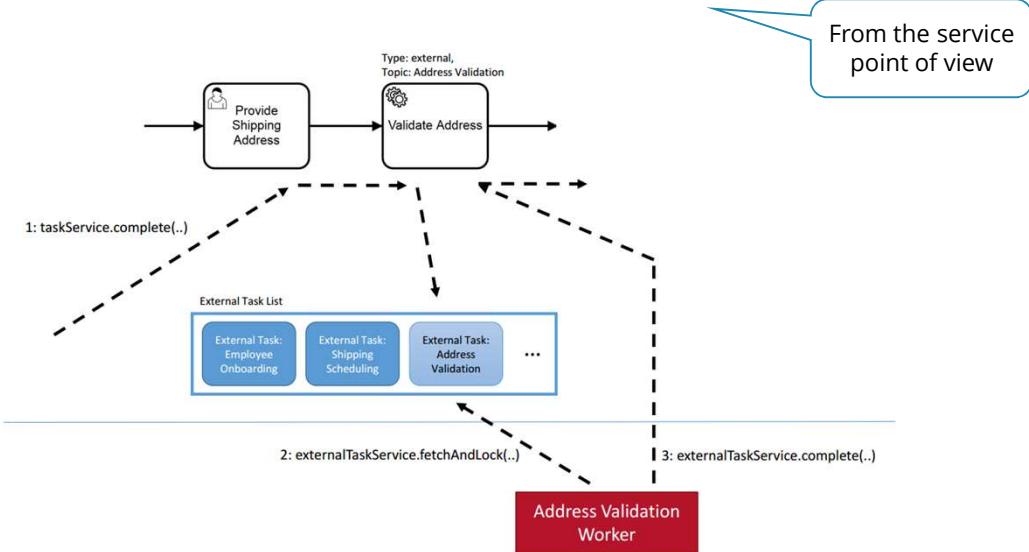
<https://docs.camunda.org/manual/latest/user-guide/process-engine/external-tasks/>

Asynchronous Service Task:

<https://github.com/camunda/camunda-bpm-examples/tree/master/servicetask/service-invocation-asynchronous>



External Service Task: Don't Call Us! We Call You!



<https://docs.camunda.org/manual/latest/user-guide/process-engine/external-tasks/>



Event Coverage

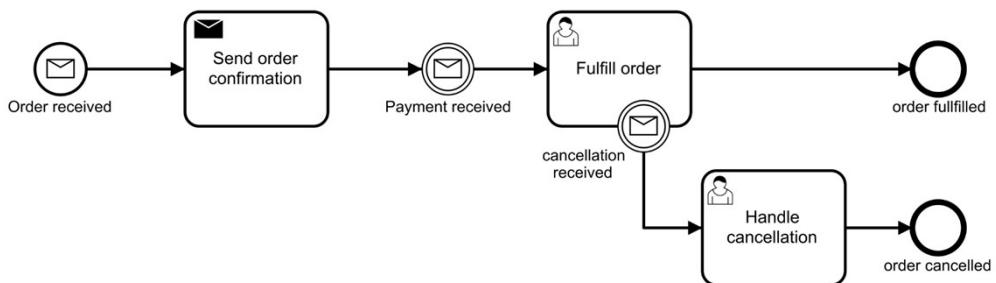
Type	Start			Intermediate				End
	Normal	Event Subprocess	Event Subprocess non-interrupt	catch	boundary	boundary non-interrupt	throw	
None	○						○	○
Message	✉	✉	✉	✉	✉	✉	✉	✉
Timer	⌚	⌚	⌚	⌚	⌚	⌚	⌚	⌚
Conditional	⤒	⤒	⤒	⤒	⤒	⤒	⤒	⤒
Link				⤓			⤓	
Signal	△	△	△	△	△	△	△	△
Error	✗				✗			✗
Escalation	Ⓐ	Ⓐ		Ⓐ	Ⓐ	Ⓐ	Ⓐ	Ⓐ
Termination							●	
Compensation	✖			✖		✖	✖	✖
Cancel					✖		✖	
Multiple	○○	○○	○○	○○	○○	○○	○○	○○
Multiple Parallel	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕

See:
docs.camunda.org/manual/latest/reference/bpmn20/#events



191

Message Events

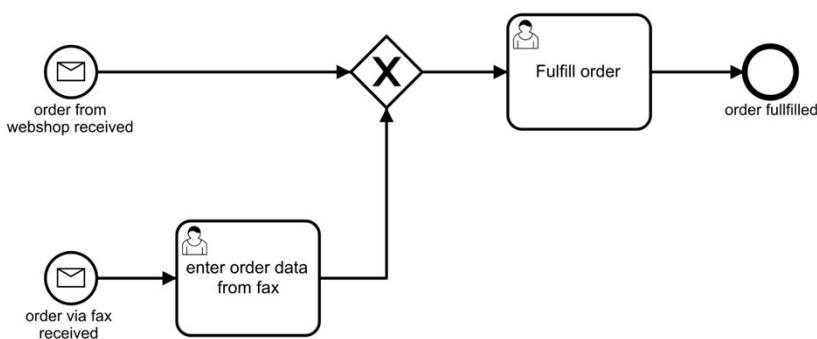


```
runtimeService
    .createMessageCorrelation("messageName")
    .processInstanceVariableEquals("myVariable", "myValue")
    .setVariable("myPayload", "myPayloadValue")
    .correlateWithResult();
```



192

Multiple Message Start Events Possible

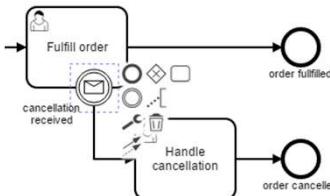


```
// correlate the message
runtimeService
    .createMessageCorrelation("messageName")
    .processInstanceBusinessKey("AB-123")
    .setVariable("payment_type", "creditCard")
    .correlateWithResult();
```



193

Configuring Message Names



BoundaryEvent_14qb0su

General	Listeners	Extensions
---------	-----------	------------

General

Id: BoundaryEvent_14qb0su

Name: cancellation received

Details

Message: orderCancelMessage (id=Message_04dfh6o)

Message Name: orderCancelMessage

Asynchronous Continuations

Asynchronous Before

Asynchronous After

Documentation

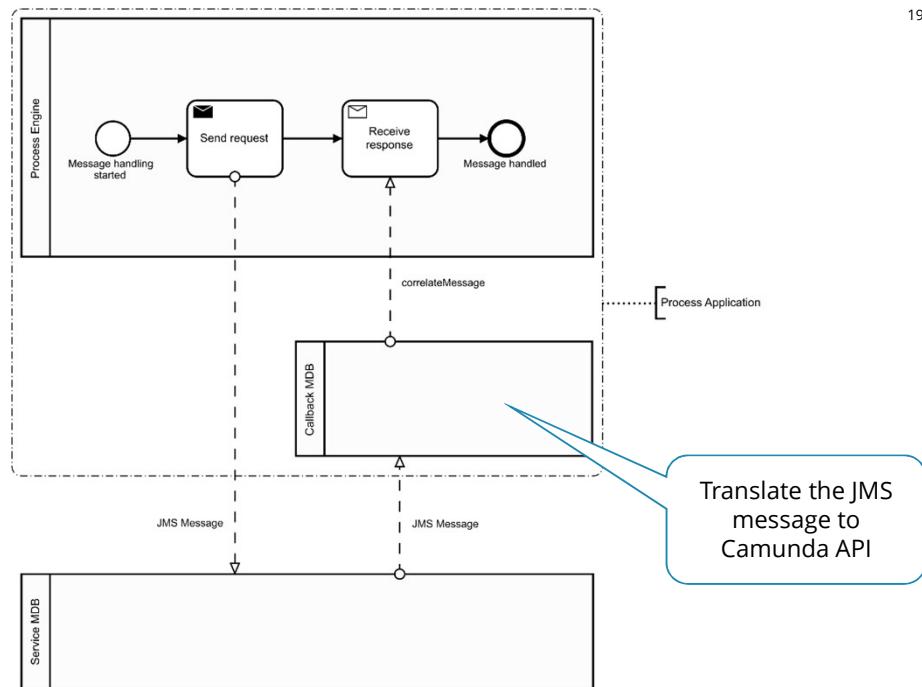
Element Documentation

```
<bpmn:process id="OrderHandlingProcess" isExecutable="true">
  ...
  <bpmn:boundaryEvent id="BoundaryEvent_14qb0su"
    name="cancellation received" attachedToRef="fulFillOrderUserTask">
    <bpmn:messageEventDefinition messageRef="Message_04dfh6o" />
  </bpmn:boundaryEvent>
  <bpmn:userTask id="fulFillOrderUserTask" name="Fulfill order">
  </bpmn:userTask>
  ...
</bpmn:process>
<bpmn:message id="Message_04dfh6o" name="orderCancelMessage" />
```



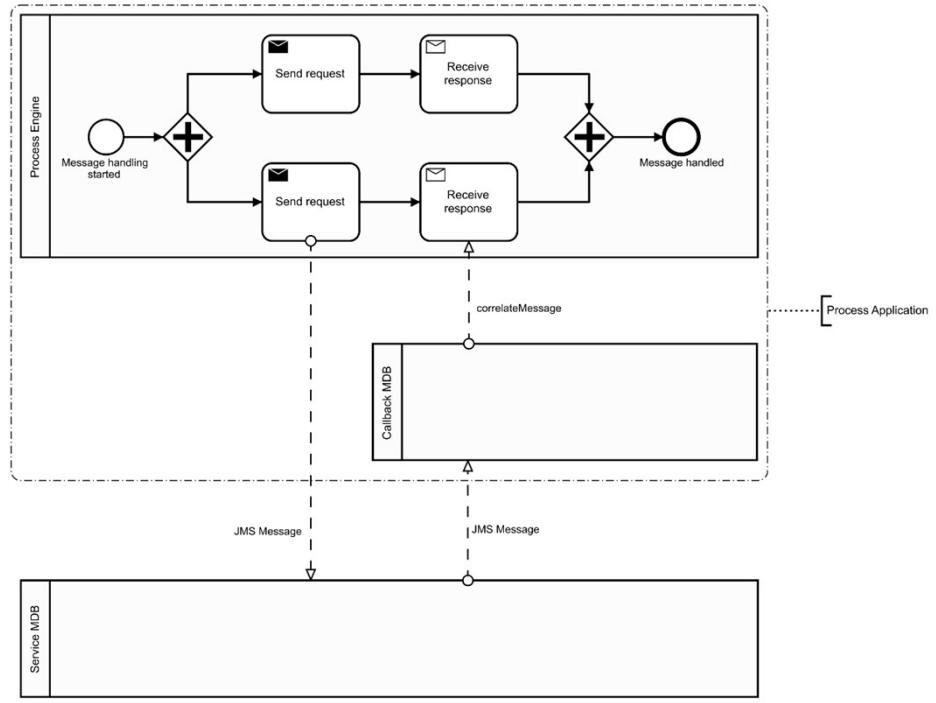
194

JMS



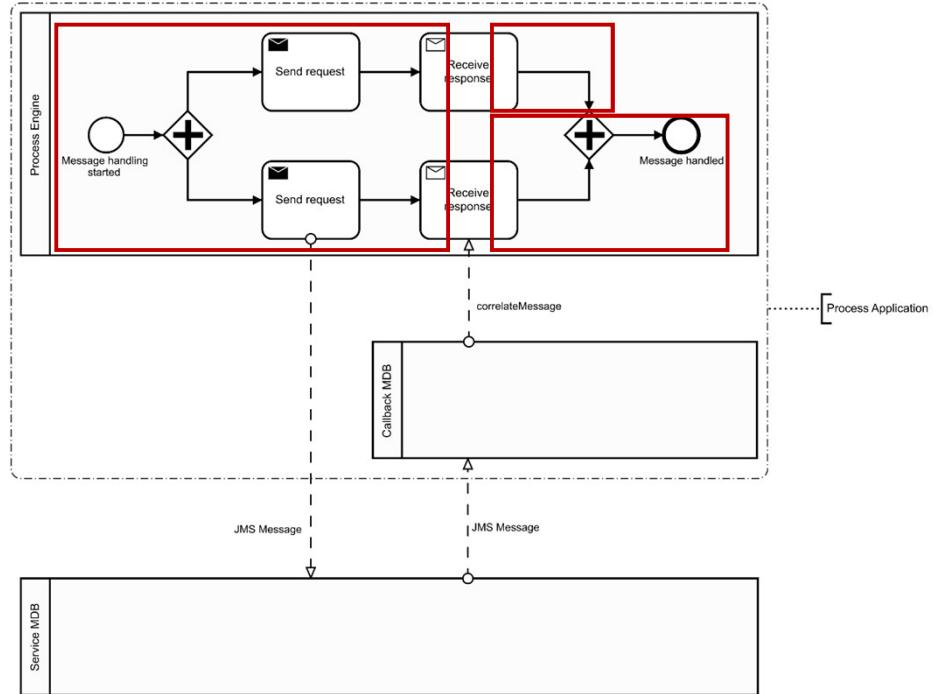
JMS (#TX?)

195



JMS

196





Correlation Keys

Process Instance Level

- Business Key
- Process Variable (e.g. UUID)
- Process Instance ID (not recommended)

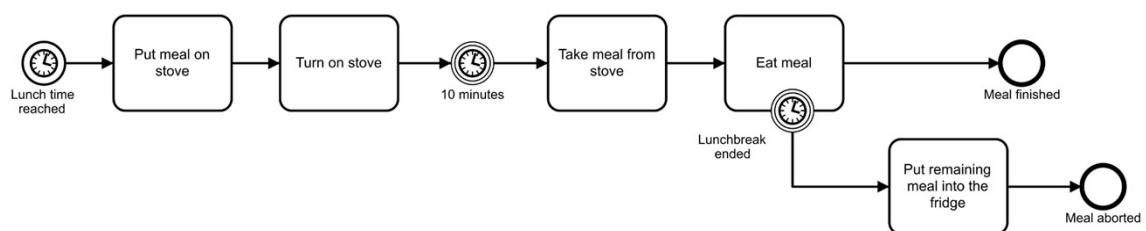
Token Level (when waiting for multiple messages)

- Local Variable (e.g. UUID)
- Combination of Process Instance (see above) and Activity ID
- Execution ID (not recommended)

Alternative option: correlation table



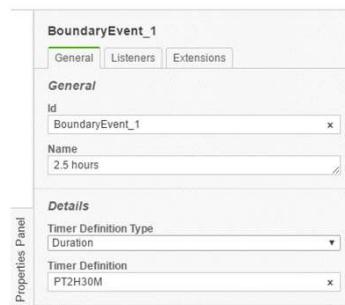
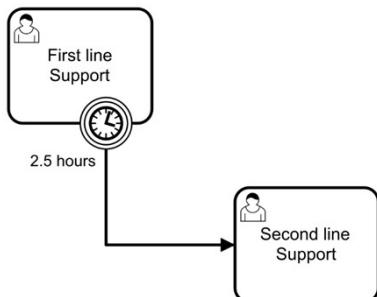
Timer Events





199

Timer Event Configuration



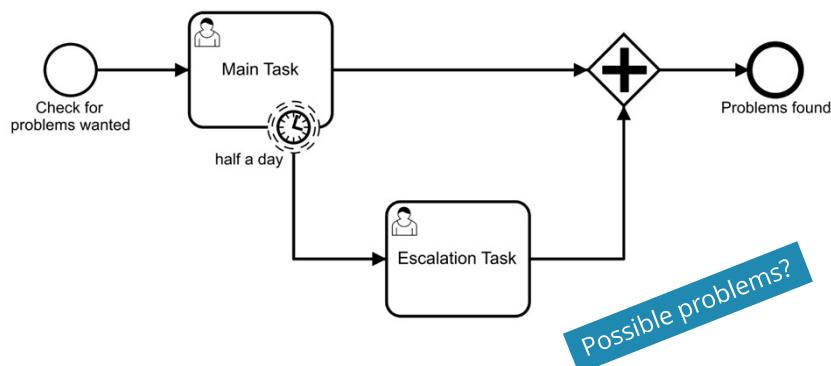
```
<bpmn:process id="Process_1" name="Timer Event Configuration Example" isExecutable="true">
  <bpmn:userTask id="firstLineSupportUserTask" name="First Line Support" />
  <bpmn:boundaryEvent id="BoundaryEvent_1" name="2.5 hours" attachedToRef="firstLineSupportUserTask">
    <bpmn:timerEventDefinition>
      <bpmn:timeDuration xsi:type="bpmn:tFormalExpression">PT2H30M</bpmn:timeDuration>
    </bpmn:timerEventDefinition>
  </bpmn:boundaryEvent>
  <!-- -->
</bpmn:process>
```

ISO8601: https://en.wikipedia.org/wiki/ISO_8601



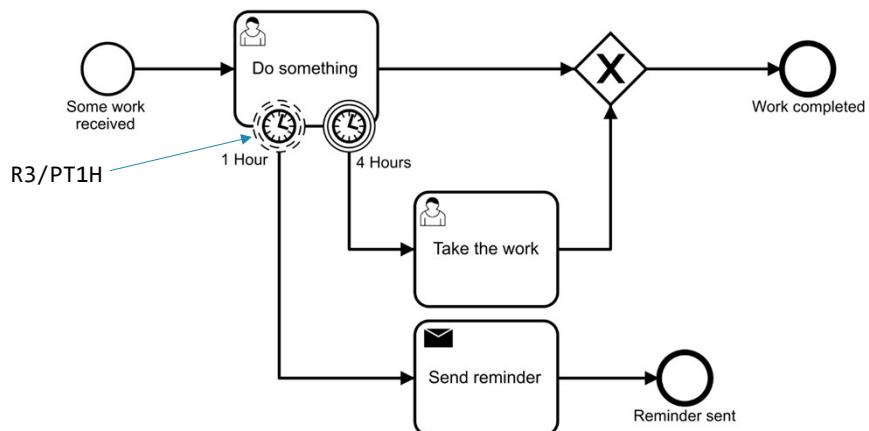
200

Non Interrupting Timer Event

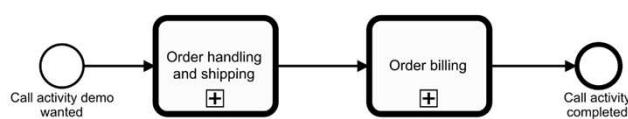




Pattern for Two Stage Escalation



Reusable Sub-Processes: Call Activities



Properties Panel

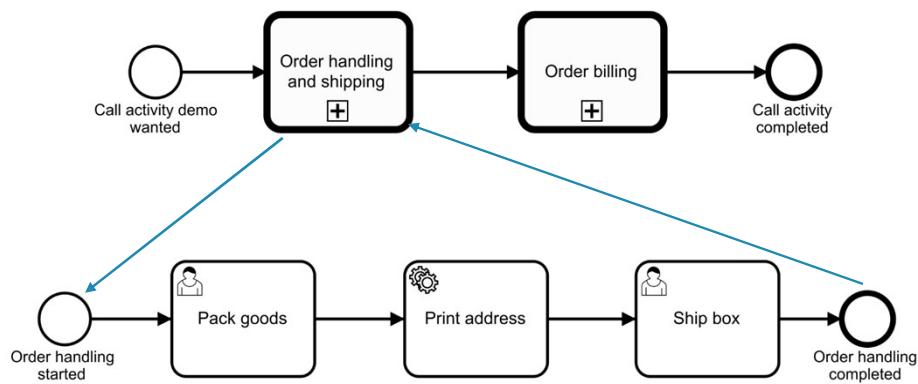
orderHandlingCallActivity

General	Variables	Listeners	Input/Output	Extens:
General				
Id: orderHandlingCallActivity				
Name: Order handling and shipping				
Details				
CallActivity Type: BPMN				
Called Element: orderHandlingProcess				
Binding: latest				
Deployment: deployment				
Version: version				
<input type="checkbox"/> Business Key				
Delegate Variable Mapping				
Properties Panel				
Variables				
In Mapping: 0 : all				
Out Mapping: 0 : orderNumber				
Out Mapping				
Type: Source				
Source: orderNumber				
Target: orderId				
<input type="checkbox"/> Local				

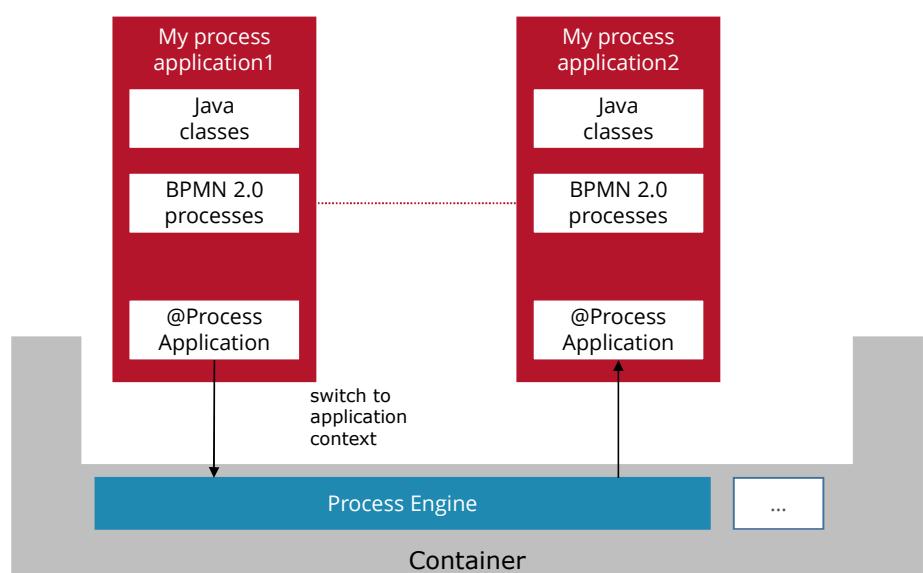
- Key is used to identify called process
- Process is resolved at runtime
- Variables can be mapped



Reusable Sub-Processes: Call Activities

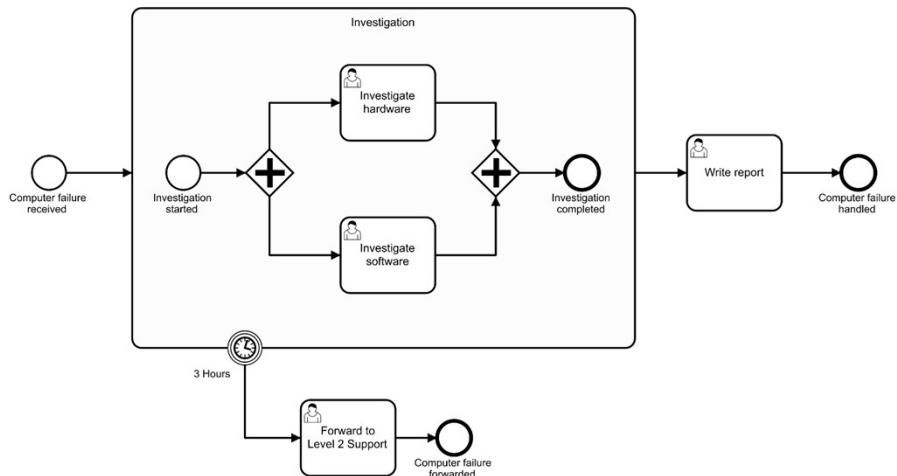


Calling Processes From Other Process Applications





Embedded Sub-Processes

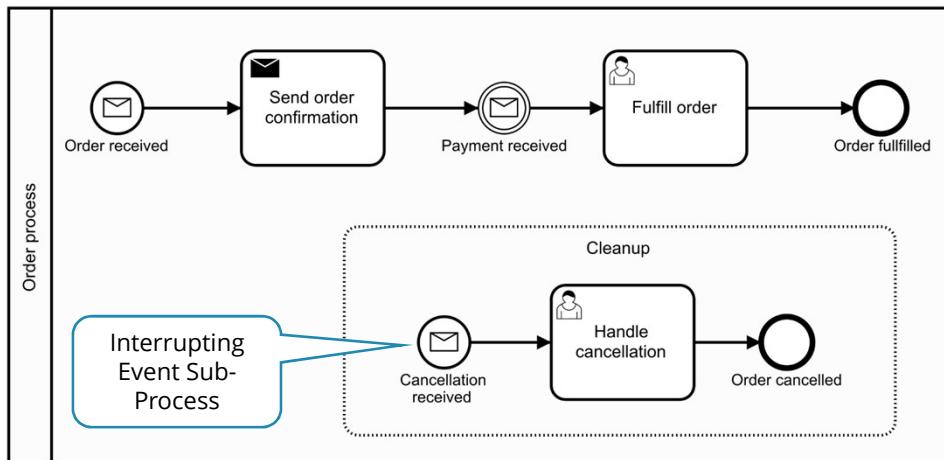


The Sub-Process Is Part of the Main Process

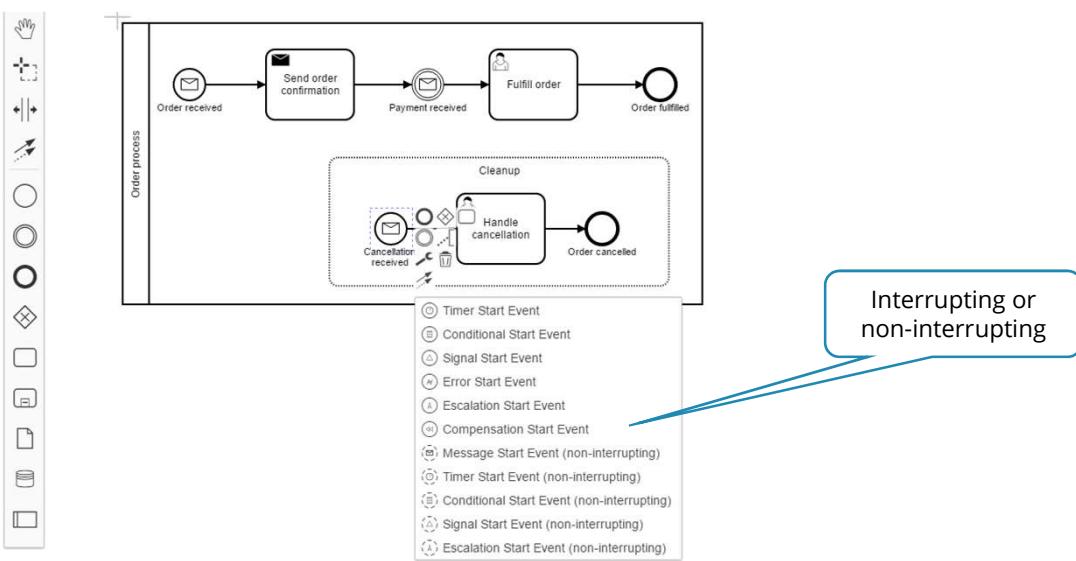
```
<bpmn:process id="Process_1" isExecutable="false">
  <bpmn:startEvent id="StartEvent_1" name="Computer failure received"></bpmn:startEvent>
  <bpmn:sequenceFlow id="SequenceFlow_1ilu0mr" sourceRef="StartEvent_1"
  targetRef="investigationSubprocess" />
  <bpmn:subProcess id="investigationSubprocess" name="Investigation">
    <bpmn:startEvent id="investigationStartedStartEvent" name="Investigation
    started"></bpmn:startEvent>
    <bpmn:parallelGateway id="ParallelGateway_1"></bpmn:parallelGateway>
    <bpmn:userTask id="investigateHardwareUserTask" name="Investigate hardware"></bpmn:userTask>
    <bpmn:userTask id="investigateSoftwareUserTask" name="Investigate software"></bpmn:userTask>
    <bpmn:parallelGateway id="ParallelGateway_2"></bpmn:parallelGateway>
    <bpmn:endEvent id="investigationCompletedEndEvent" name="Investigation
    completed"></bpmn:endEvent>
    <bpmn:sequenceFlow id="SequenceFlow_01s0ans" sourceRef="investigationStartedStartEvent"
    targetRef="ParallelGateway_1" />
    ...
  </bpmn:subProcess>
  ...
</bpmn:process>
```



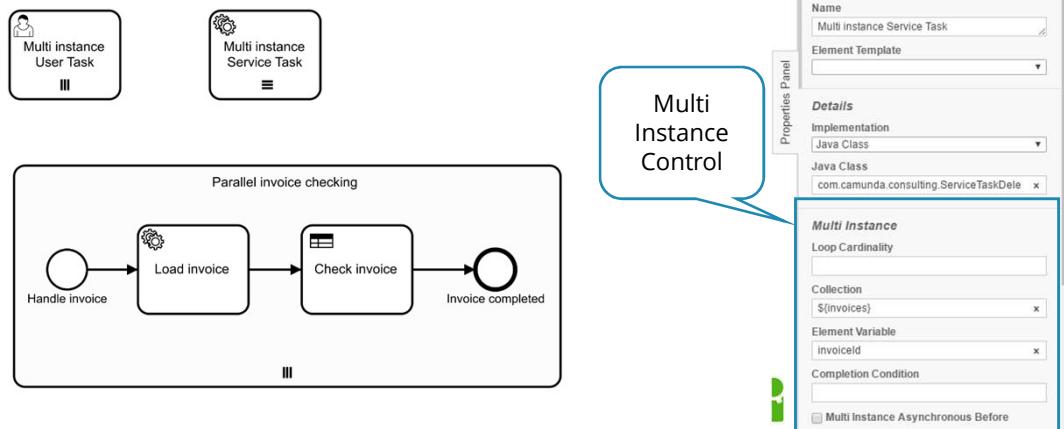
Event Sub-Process



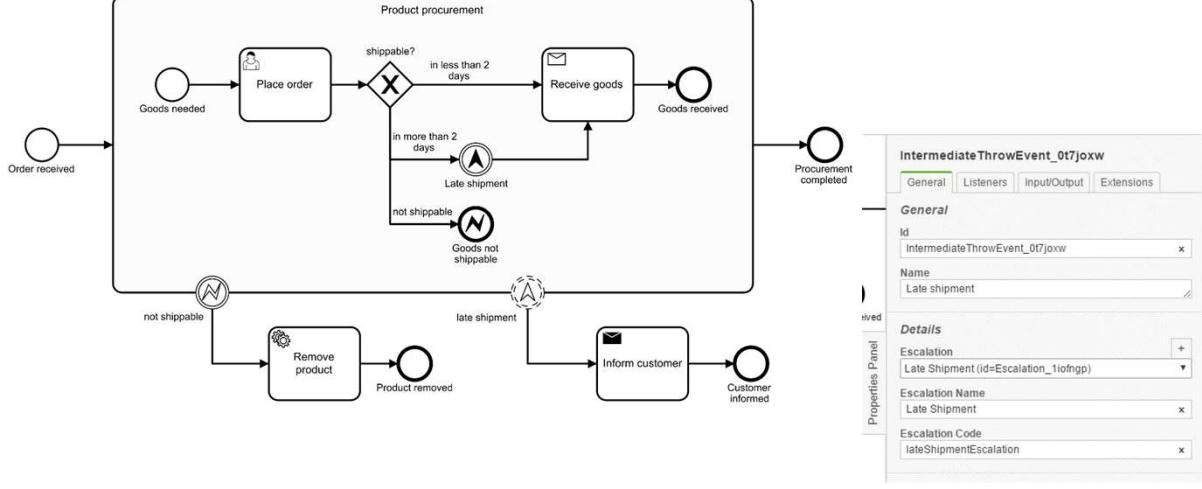
Possible Events to Start an Event Sub-Process



Multiple Instance Marker

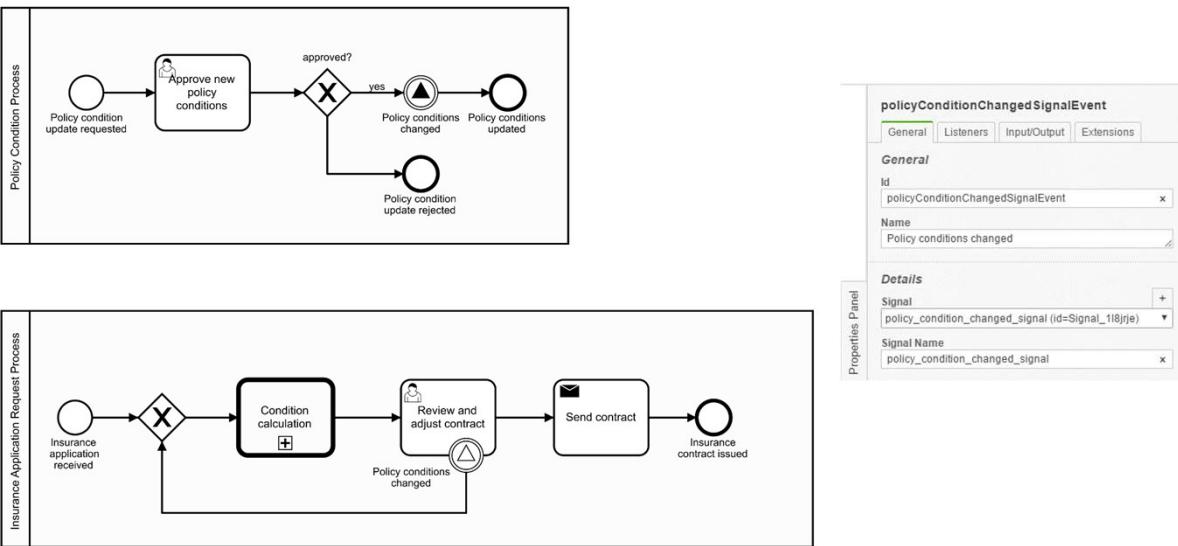


Escalation and Error Events

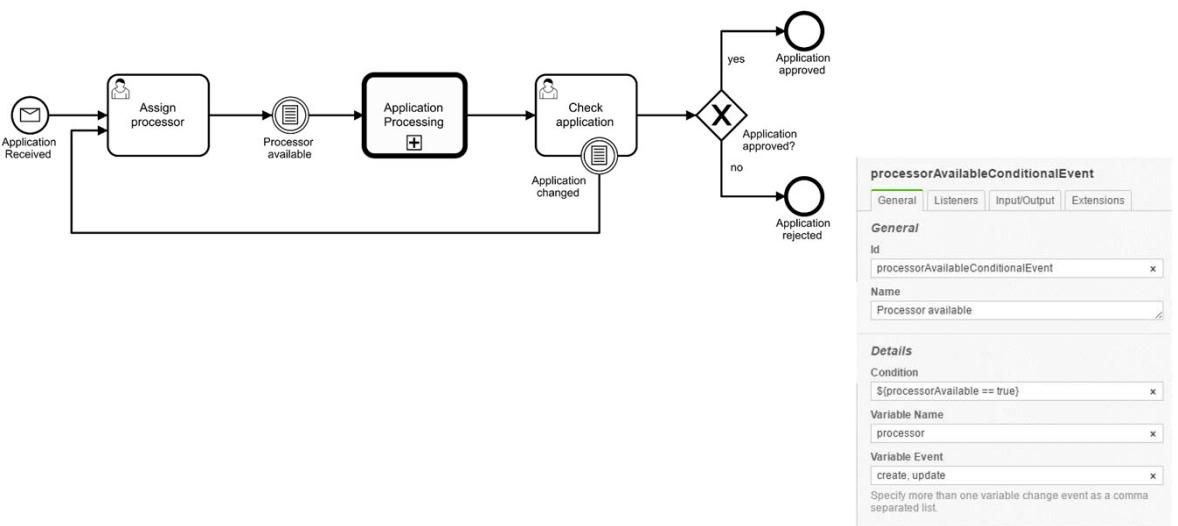




Signal Events – Broadcast to All Process Instances

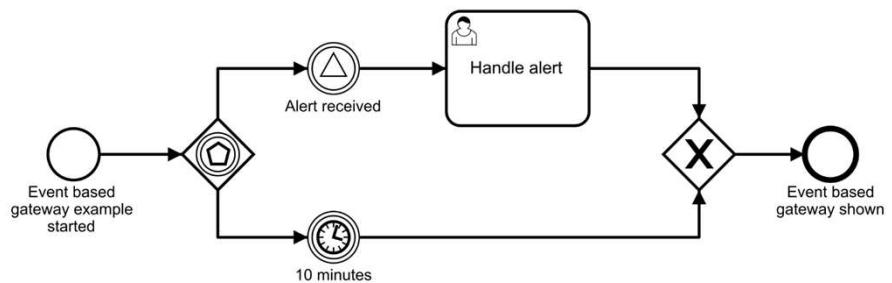


Conditional Events





Event-Based Gateway – Simple Example



Event-Based Gateway – Real World Example

Please model the following process:

If an insurant could be possibly subrogated against, I get information about that. I check that case and if the possibility is really there, I send a request for payment to the insurant and make me a reminder. If recourse is not possible, I close the case.

When we receive the money, I make a booking and close the case. If the insurant disagrees with the recourse, I'll have to check the reasoning of that. If he is right, I simply close the case. If he is wrong, I forward the case to a collection agency.

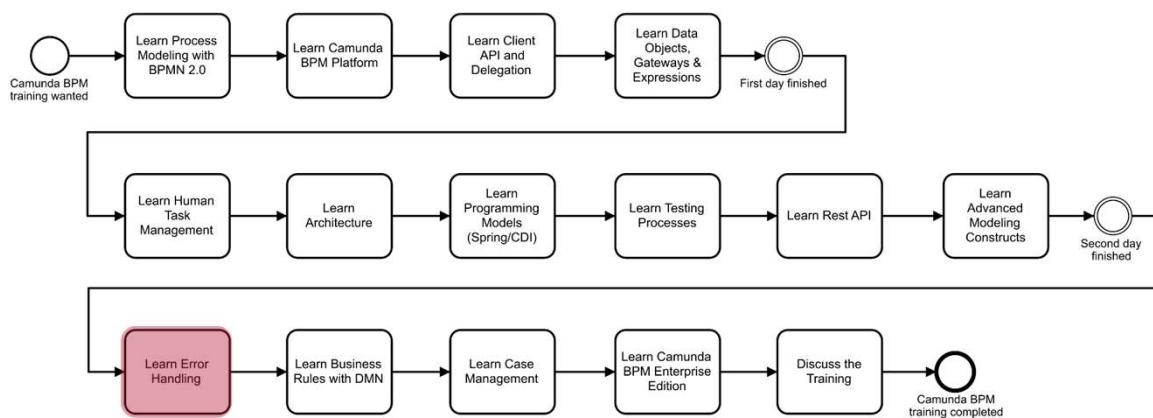
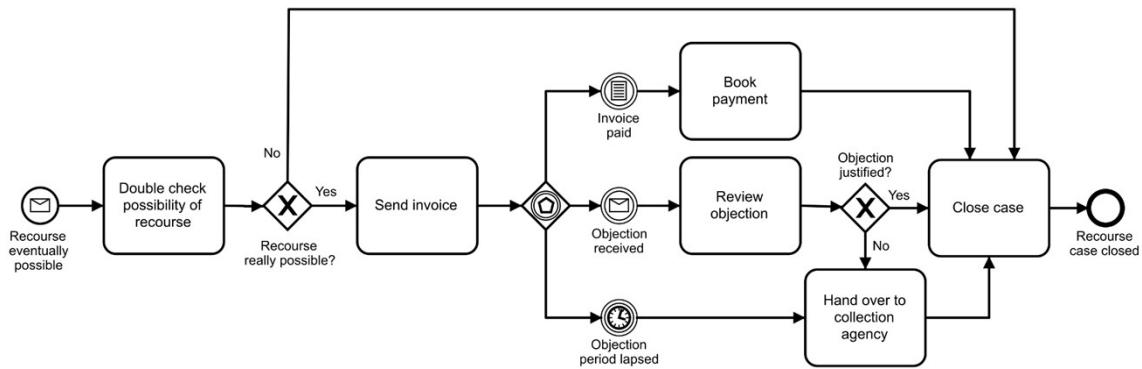
If the deadline for disagreement is reached and we haven't received any money, I forward the case to the collection agency as well.

Background information:

Insurants can be forced to pay back money they received from the insurance company for different reasons. This is called recourse. Here the clerk describes how this process works.



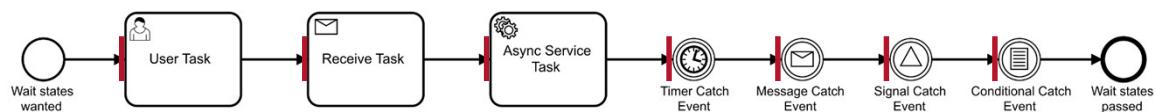
Event-Based Gateway – Real World Example





217

Wait States



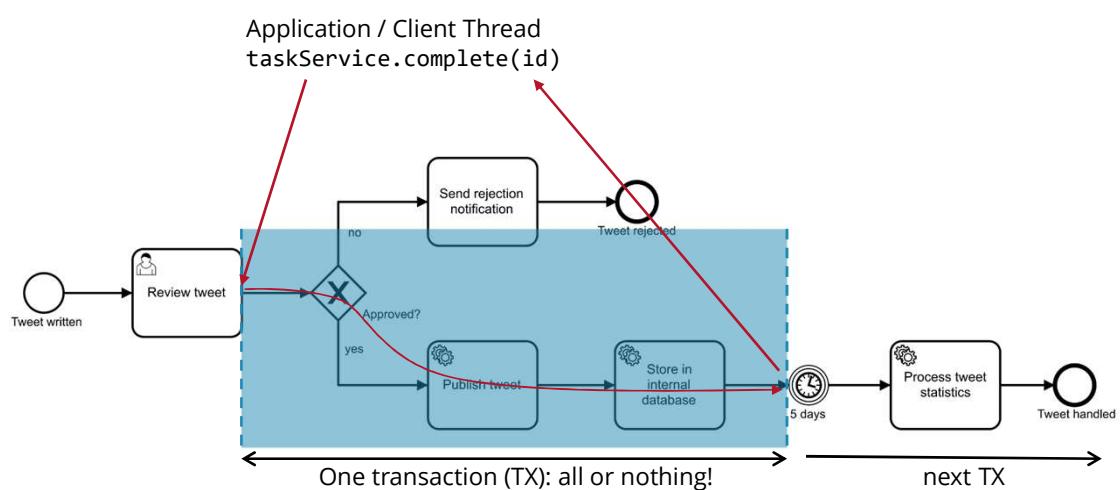
Also called:

- Transaction Boundaries
- Save Points



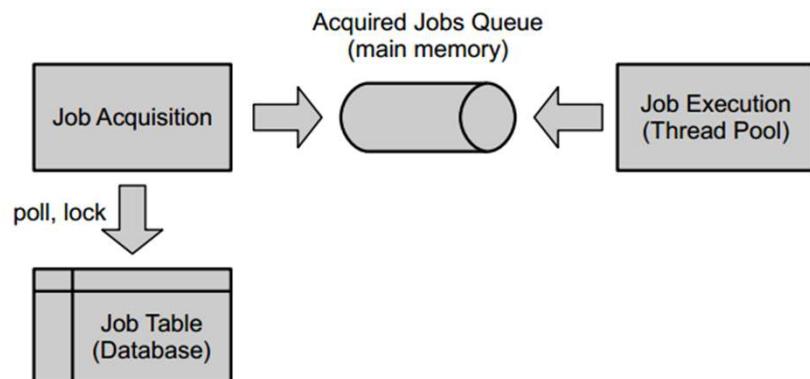
218

Transaction Boundaries





The Job Executor



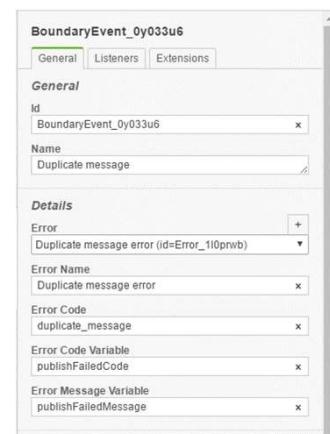
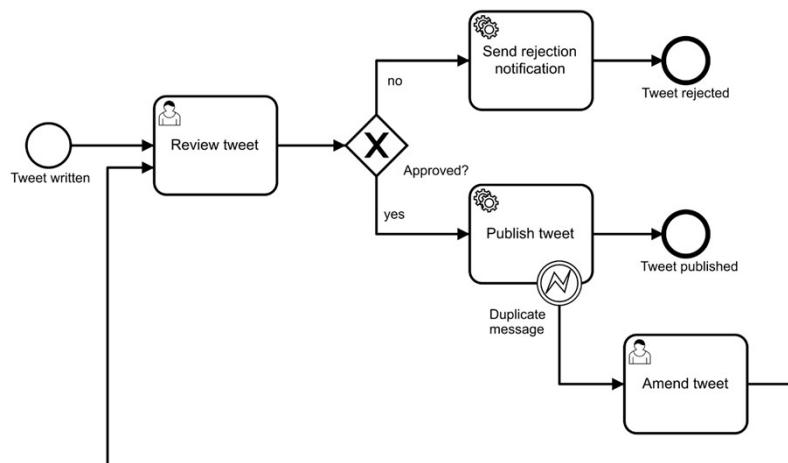
Demo





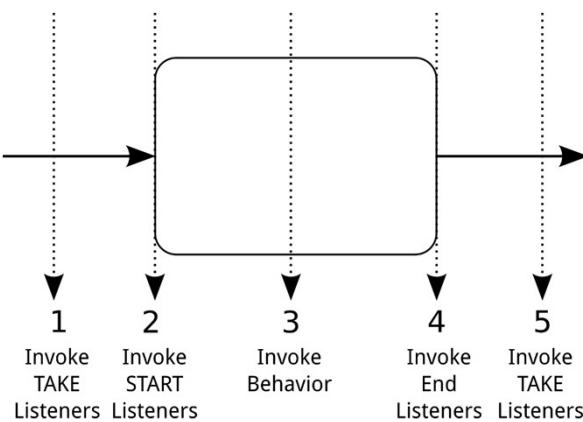
221

Error Handling



222

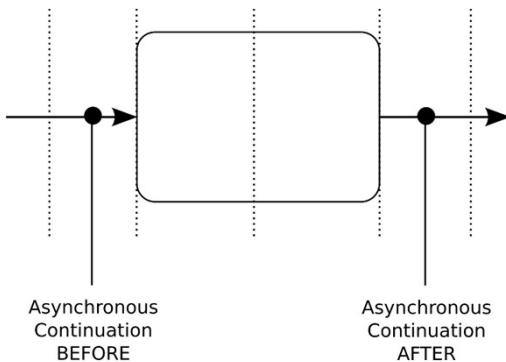
Asynchronous Continuation: Execution of a BPMN Element





223

Asynchronous Continuation

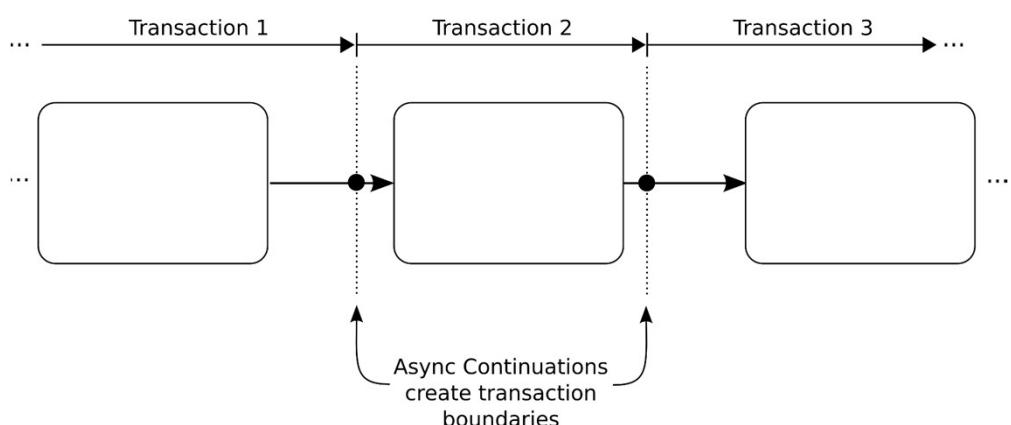


```
<bpmn:serviceTask id="publishTweetServiceTask" name="Publish tweet" camunda:asyncBefore="true" camunda:class="com.camunda.consulting.PublishTweetDelegate"/>
<bpmn:serviceTask id="publishTweetServiceTask" name="Publish tweet" camunda:asyncAfter="true" camunda:class="com.camunda.consulting.PublishTweetDelegate"/>
```



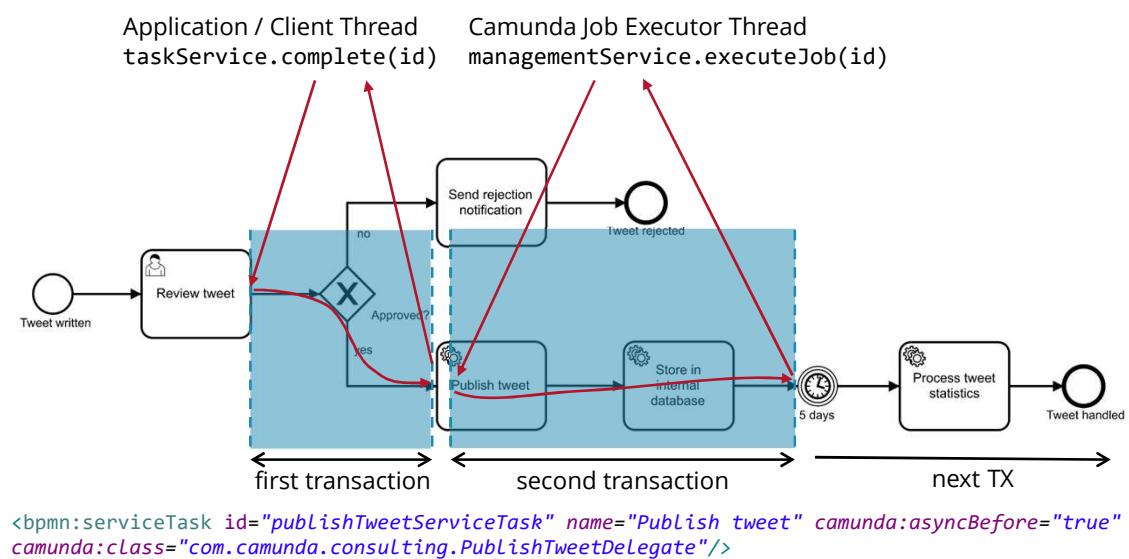
224

Asynchronous Continuation = Transaction Boundaries/Save Points

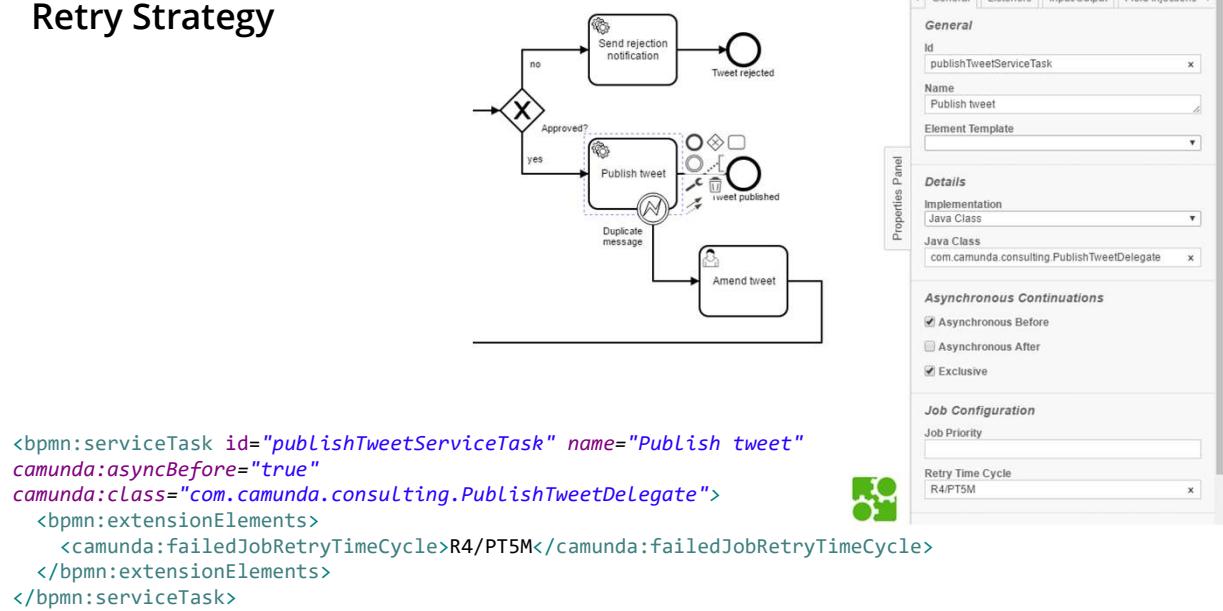


```
<bpmn:serviceTask id="publishTweetServiceTask" name="Publish tweet" camunda:asyncBefore="true" camunda:asyncAfter="true" camunda:class="com.camunda.consulting.PublishTweetDelegate"/>
```

Transaction Boundaries



Retry Strategy





Best Practices for Save Points

- Before start event of process instance
- After user task
- Before invocations of external systems especially over network
- After non-transactional, non-idempotent side-effects
- After expensive external computation
- Before parallel joins like parallel and inclusive gateways or at the end of each instance of parallel multi-instance activities

NOT

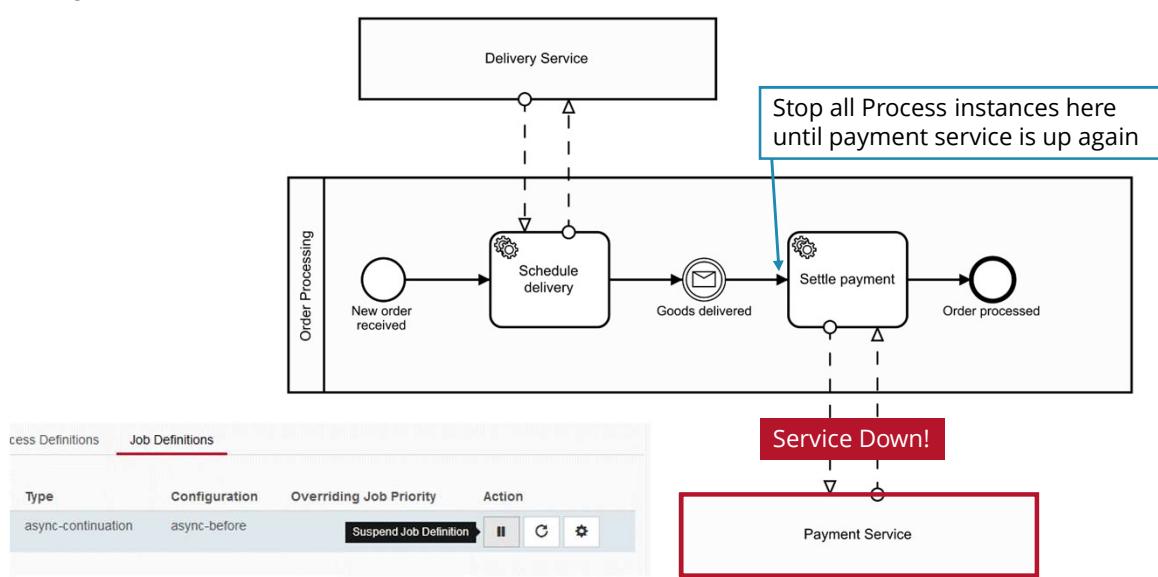
- Before tasks that wait anyway (user tasks, receive tasks, external service tasks, catching events like timer, message, signal)
- Before XOR & Event-based gateways
- Before splitting AND & OR gateways

UNLESS

- Listeners, beans or complex expressions, e.g. using SPIN, that could fail are involved



Suspension

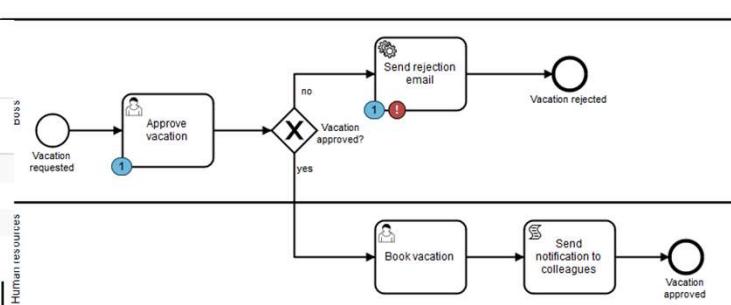


Incidents Overview

4 process definitions deployed

List Previews

State	Running Instances	Name
0	0	Insurance Application
13	13	Invoice Receipt
1	1	TwitterDemoProcess
8	8	Vacation request



Process Instances Called Process Definitions Job Definitions

State	ID	Start Time	Business Key
0	1a2e7549-ab0f-11e6-879c-185e0f710ea8	2016-11-15T09:40:03	
1	9f4d8c60-ab0e-11e6-879c-185e0f710ea8	2016-11-15T09:36:37	

Incidents Details

Camunda Cockpit | Vacation request : 9f4d8c60-ab0e-11e6-879c-185e0f710ea8 : Runtime

Information Filter

Instance ID: 9f4d8c60-ab0e-11e6-879c-185e0f710ea8...
 Business Key: null
 Definition Version: 4
 Definition ID: VacationRequestProcess:4:7568539...
 Definition Key: VacationRequestProcess
 Definition Name: Vacation request
 Tenant ID: null
 Deployment ID: 7564aa0e-ab0b-11e6-879c-185e0f710ea8...
 Related Migration

Variables Incidents Called Process Instances User Tasks Modify

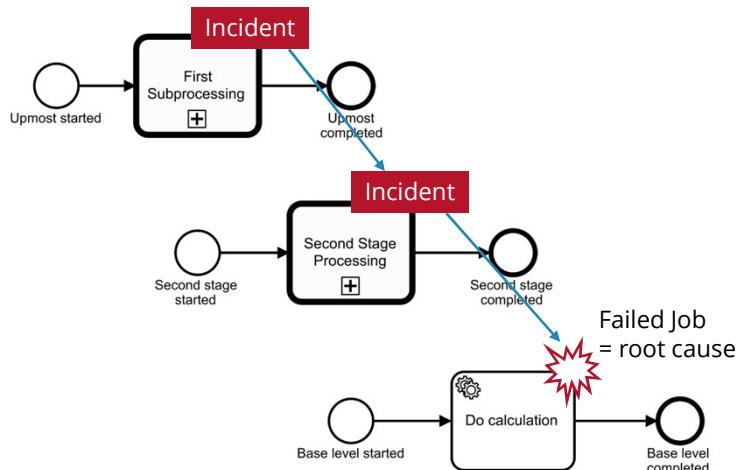
Message	Timestamp	Activity	Cause Process Instance Id	Root Cause Process Instance Id	Type	Action
Could not send notification...	2016-11-15T09:38:18	Send rejection email			Failed Job	

Retry failed job



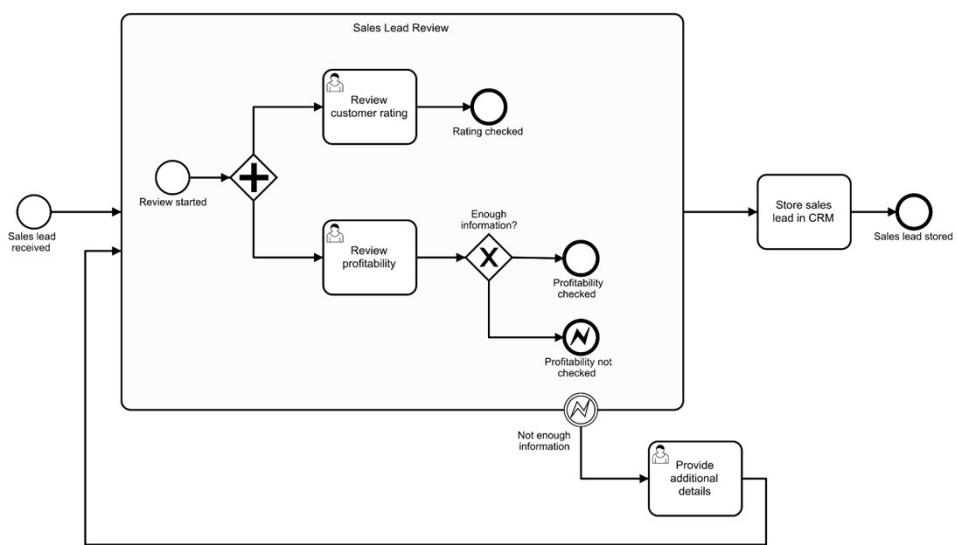
231

Drill Down



232

Error Event





What if No Error Handler Is Defined?

Quotes from BPMN 2.0.1:

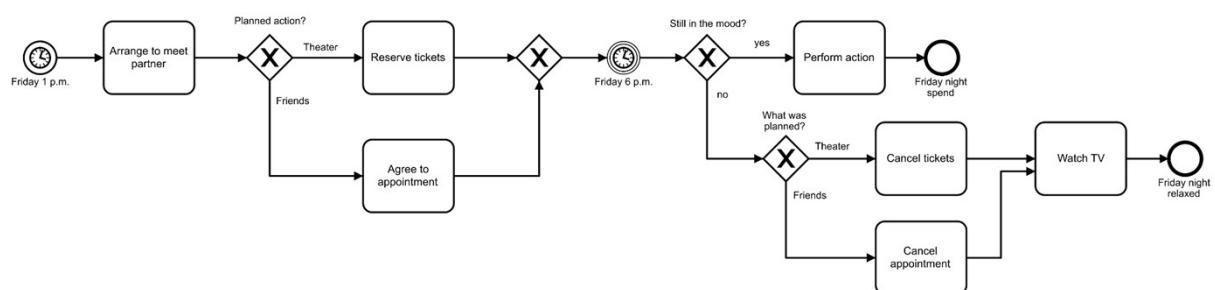
"Error triggers are critical and **suspend execution** at the **location of throwing**."

"If no catching Event is found for an error or escalation trigger, this trigger is **unresolved**."

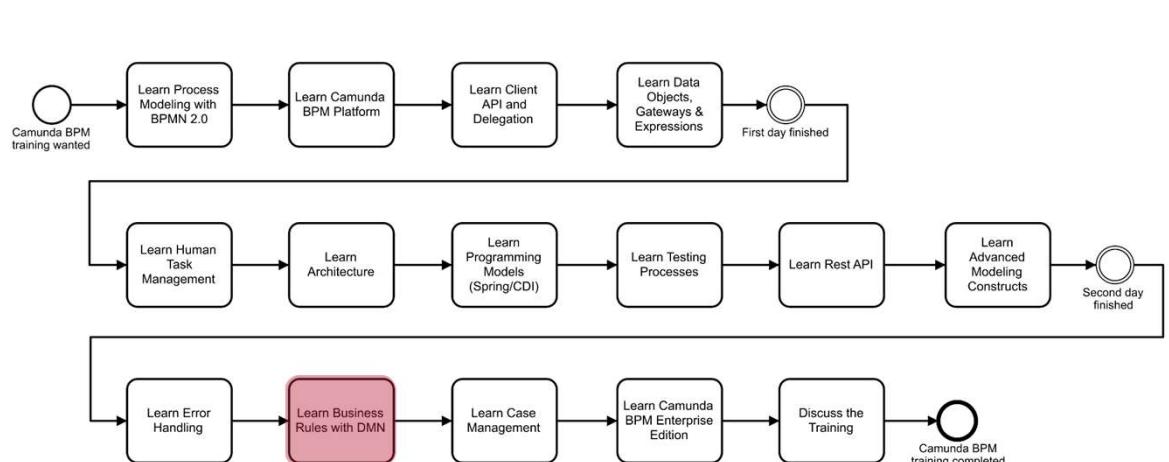
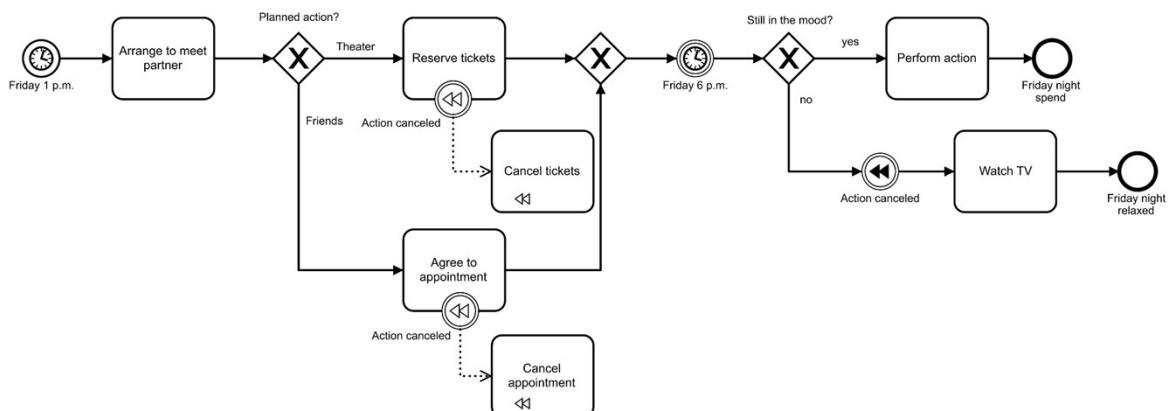
"The behavior of the Process is **unspecified** if no Activity in the hierarchy has such an Error Intermediate Event. The system executing the process can define additional Error handling in this case, a common one being **termination of the Process instance**."



Compensation



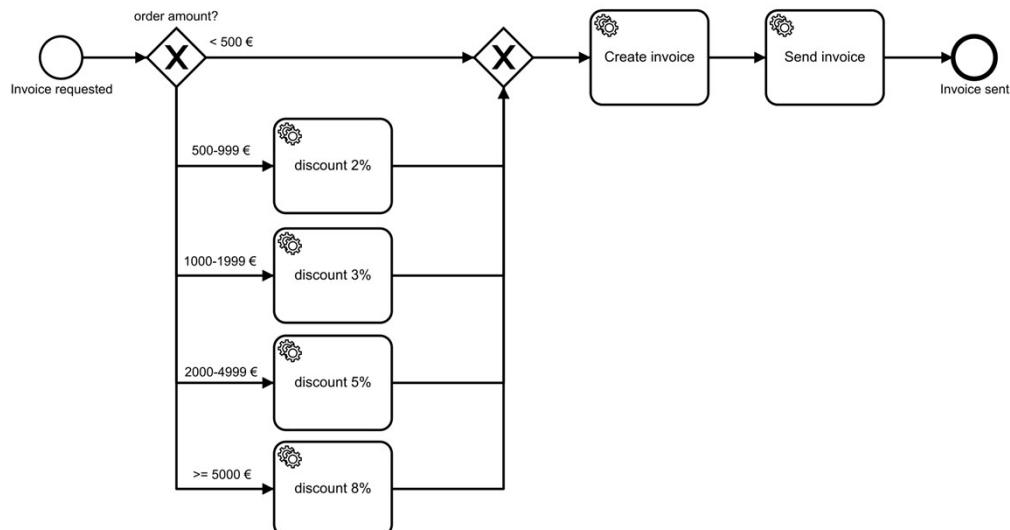
Compensation Events





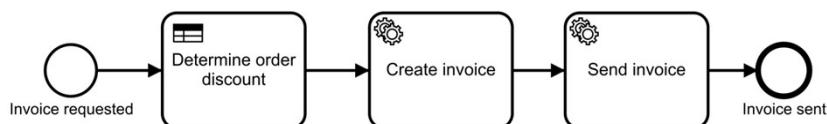
237

Is This a Good Process Model?



238

An Improved Version





Discount Rules as DMN Decision Table

The screenshot shows the Camunda Modeler application window. The title bar says "Camunda Modeler". The menu bar includes "File", "Edit", "Window", and "Help". A toolbar with various icons is at the top. Below the toolbar, there's a tab bar with "order-discount.dmn x" and a "+" button. The main area displays a "Decision Table" titled "Order Discount". The table has three columns: "Input +" (with a dropdown menu showing "long"), "Output +" (with a dropdown menu showing "integer"), and "Annotation". There are five rows of data:

	Input +	Output +	Annotation
1	<500	0	-
2	[500..999]	2	-
3	[1000..1999]	3	-
4	[2000..4999]	5	-
5	>=5000	8	-

At the bottom of the table view, there are buttons for "Table" and "XML". On the right side of the table view, there's a "Log" button.

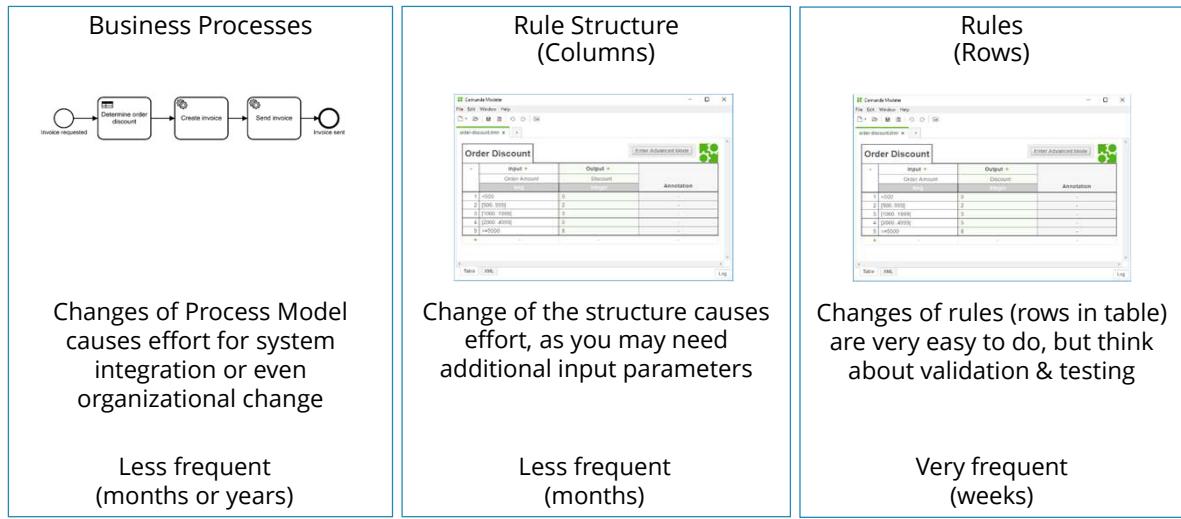


The Standard DMN

- Decision Model and Notation
- Is currently in publications by OMG for Decisions/Business Rules
- Specification: „The primary goal of DMN is to provide a **common notation** that is readily understandable by all **business users**, from the business analysts needing to create initial decision requirements and then more detailed decision models, to the **technical developers** responsible for automating the decisions in processes, and finally, to the business people who will manage and monitor those decisions. DMN creates a standardized **bridge for the gap between the business decision design and decision implementation**. DMN notation is designed to be **useable alongside the standard BPMN** business process notation.”
- Camunda shipped DMN 1.1 support with 7.4



Typical Frequency of Changes



Rules Expressed as Decision Table

Dish				Enter Advanced Mode
u	Input + Season string	Output + Dish string	Annotation	
1 "Fall"	"Spareribs"		-	
2 "Winter"	"Roastbeef"		-	
3 "Spring"	"Steak"		-	
4 "Summer"	"Light Salad and a nice Steak"		Hey, why not?	

Decision name

Input expression

Hit Policy „Unique“

Each row = single rule

Input entry

Output entry = Result of rule

Optional remarks. Typically used for motivation of rule

Multiple Inputs

Input entries can have various data formats

Enter Advanced Mode

Annotation

Dish		Input +	Output +		
U		Season string	Vegetarian Guests boolean	Dish string	Annotation
1	"Fall"	No	▼	"Spareribs"	-
2	"Winter"	No	▼	"Roastbeef"	-
3	"Spring"	No	▼	"Steak"	-
4	"Summer"	No	▼	"Light Salad and a nice Steak"	Hey, why not?!
5	-	Yes	▼	"Pasta"	-
+		-	-	-	-

Multiple inputs always follow „AND“ logic

Technical Details

Decision Id

Input parameter

Type definition

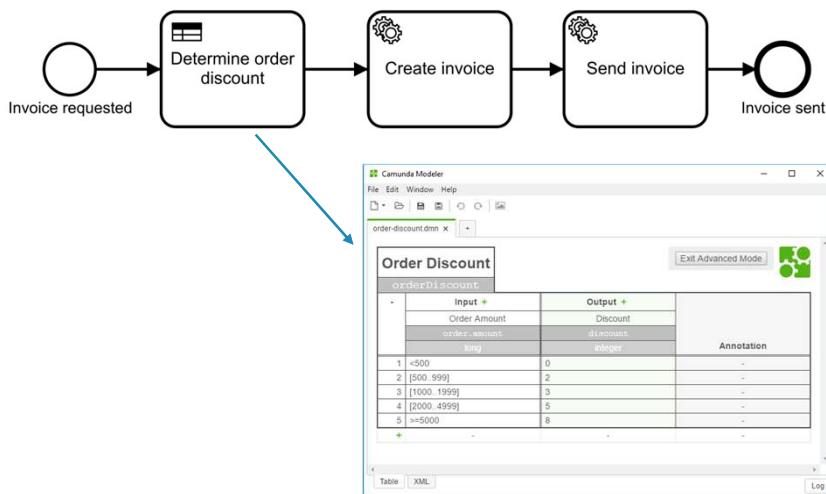
Exit Advanced Mode

Annotation

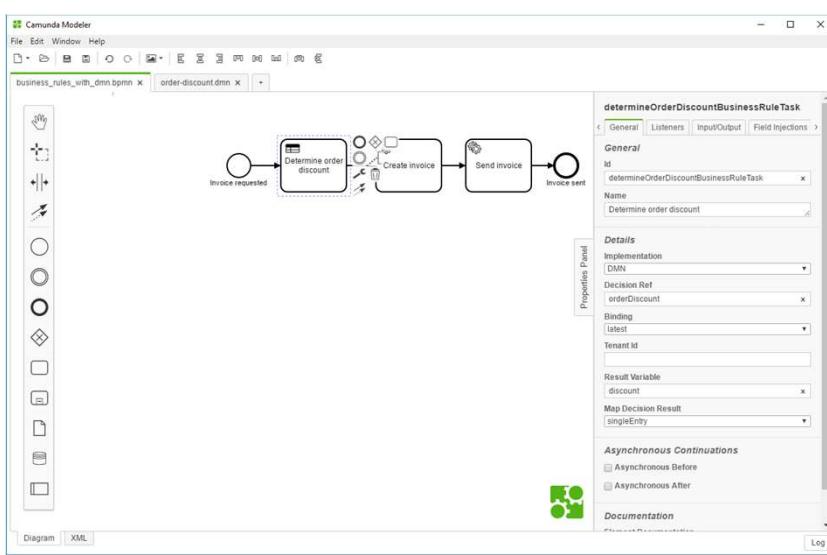
Dish		Input +	Output +		
U	dish-decision	Season string	How many guests guestCount integer	Dish desiredDish string	Annotation
1	"Fall"	<=8		"Spareribs"	-
2	"Winter"	<=8		"Roastbeef"	-
3	"Spring"	<=4		"Dry Aged Gourmet Steak"	-
4	"Spring"	[5..8]		"Steak"	Save money
5	"Fall", "Winter", "Spring"	> 8		"Stew"	Less effort
6	"Summer"	-		"Light Salad and a nice Steak"	Hey, why not?!
+		-	-	-	-



Business Rule Task Reference a DMN Decision Table



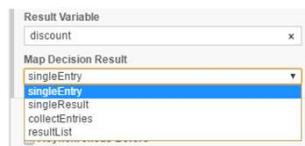
Reference in Camunda Modeler



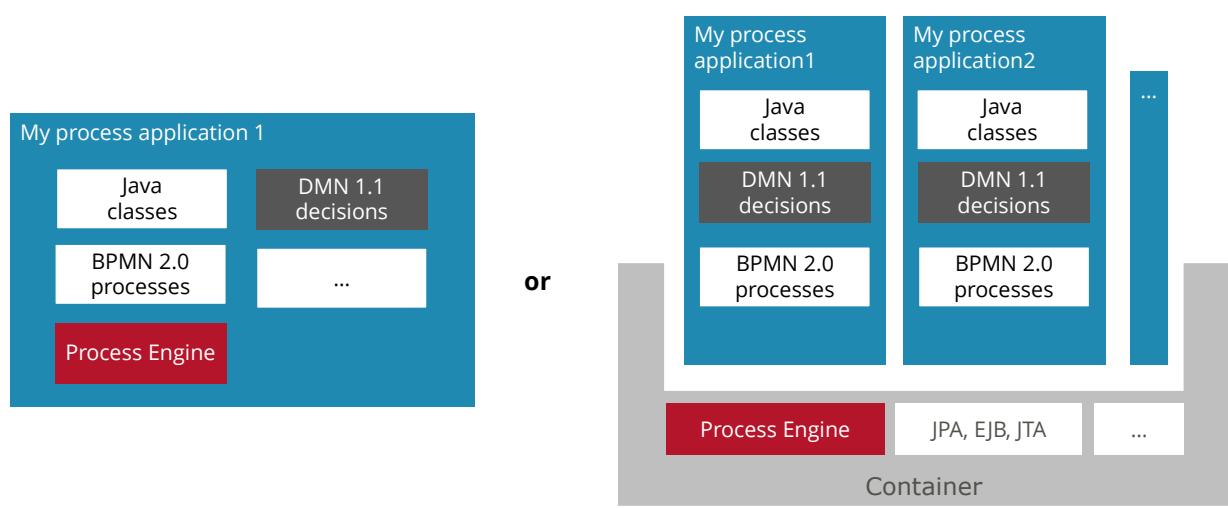


Map the Result to the Process

Mapper	Result	Is suitable for decision tables with
singleEntry	TypedValue	No more than one matching rule and only one output
singleResult	Map<String, Object>	No more than one matching rule and multiple outputs
collectEntries	List<Object>	Multiple matching rules and only one output
resultList	List<Map<String, Object>>	Multiple matching rules and multiple outputs



DMN Is Part of the BPM Platform





Standalone Stateless Decision Engine

- You can use the core DMN engine as stateless Rule Engine separately from the BPM Platform if you like
- Single Maven Dependency:

```
<dependency>
  <groupId>org.camunda.bpm.dmn</groupId>
  <artifactId>camunda-engine-dmn</artifactId>
</dependency>
```



Standalone (Stateless) Decision Engine

```
// create configuration
DmnEngineConfiguration configuration = DmnEngineConfiguration
    .createDefaultDmnEngineConfiguration();

// configure if needed...

// build engine
DmnEngine dmnEngine = configuration.buildEngine();

// load DMN file
InputStream inputStream = ...;

//create and add variables
VariableMap variables = Variables.createVariables();

// create decision
DmnDecision decision = dmnEngine.parseDecision("decision", inputStream);

// evaluate decision
DmnDecisionTableResult tableResult = dmnEngine
    .evaluateDecisionTable(decision, variables);
DmnDecisionRuleResult ruleResult = tableResult.getSingleResult();
String firstEntry = ruleResult.getEntry("desiredDish");
```



Decision Service in the Platform

```
DmnDecisionTableResult result = processEngine  
    .getDecisionService()  
    .evaluateDecisionTableByKey(decisionDefinitionKey, variables);
```

Also available as REST-Service

```
POST /decision-definition/key/aDecisionDefinitionKey/evaluate
```

Request body:

```
{  
    "variables" : {  
        "amount" : { "value" : 600, "type" : "Double" },  
        "invoiceCategory" : { "value" : "Misc", "type" : "String" }  
    }  
}
```

Response:

```
[  
    {  
        "result": { "value" : "management", "type" : "String", "valueInfo" : null }  
    }  
]
```



JUnit for Decisions

```
@Test  
public void test() {  
    DmnEngine dmnEngine = dmnEngineRule.getDmnEngine();  
  
    // load DMN file  
    InputStream inputStream = ...;  
  
    // create decision  
    DmnDecision decision = dmnEngine.parseDecision("decision", inputStream);  
  
    // evaluate decision  
    DmnDecisionTableResult tableResult = dmnEngine  
        .evaluateDecisionTable(  
            decision,  
            withVariables(key, value, furtherKeyValuePairs));  
    assertThat(tableResult.getSingleResult()).containsEntry(key, expectedValue);  
}
```

For embedded
decision engine



JUnit for Decisions

```
@Test  
@Deployment(resources = {"myRule.dmn"})  
public void testWithProcessEngine() {  
    Map<String, Object> variables = withVariables("orderAmount", 1999.90);  
    DmnDecisionTableResult result = processEngine()  
        .getDecisionService()  
        .evaluateDecisionTableByKey("my_decision", variables);  
    assertThat(result.getFirstResult()).containsEntry("discount", 5);  
}
```

For decisions in the process engine



Hit Policies

- Single
 - **U(nique)**: Only one rule can match
 - **A(ny)**: Multiple rules may match – must have same output
 - **P(riority)**: Rule with highest priority (output field) is selected
 - **F(irst)**: First matching rule (order in table!) is selected
- Multiple
 - **O(utput Order)**: List of rules in order of priority (output field)
 - **R(ule Order)**: List of rules in order of table
 - **Collect**: List of all hits without order, might be combined with operator + (sum), < (min), > (max), # (count)

Italic hit policies are not yet supported by the Camunda DMN engine



Examples From the Spec

Applicant Risk Rating			
U	Applicant Age	Medical History	Applicant Risk Rating
1	> 60	good	Medium
2		bad	High
3	[25..60]	-	Medium
4		good	Low
5	< 25	bad	Medium

Student Financial Package Eligibility				
R	Student GPA	Student Extra-Curricular Activities Count	Student National Honor Society Membership	Student Financial Package Eligibility List
1	> 3.5	>= 4	Yes	20% Scholarship
2	> 3.0	-	Yes	30% Loan
3	> 3.0	>= 2	No	20% Work-On-Campus
4	<= 3.0	-	-	5% Work-On-Campus



Friendly Enough Expression Language (FEEL)

[date and time(„2016-12-24T00:00:00“)

.. date and time(„2016-12-26T23:59:99“)]

examplesSimpleExpressions

Support for different „endpoint“ data types: number, string, boolean, time, date, date-time, time-duration.

F	Input +				Annotation
	Season	Date	Number of guests	Children	
	season	date	guestCount	children	
1	-	[date and time("2016-12-24T00:00:00"), date and time("2016-12-26T23:59:99")]	-	-	"Christmas Dinner"
2	"Winter"	-	-	true	"Vegetables"
3	"Winter"	-	<=6	-	"Steak"
4	"Winter"	-]6..10[-	"Pizza"
5	"Winter"	-	[10..12]	-	"Huge Pizza"
6	"Winter"	-	12..[13..14]>14	-	"Family Deluxe Pizza"
7	"Spring", "Summer", "Fall"	-		true	"Salad"
8	not("Winter")				
9	not("Spring", "Summer", "Fall")				It is Winter again ;-)

Negation

Support for Comparison (<, <=, >, >=) and Ranges ([x..y]):

- Start included: [
- Start excluded:] or (
- End included:]
- End excluded: [or)



257

JUEL as Expression Language

The screenshot shows the Camunda BPMN editor interface. A table titled "Tweet approval" contains a row for "approveTweet". The "Input" column for this row contains the JUEL expression: `currentTweet.content.matches('someRegEx')`. A tooltip for this expression is open, showing the "Expression Language" dropdown set to "juel". The tooltip also lists other operators: add, copy, remove, clear, and Input.



258

Cockpit for Decisions

The screenshot shows the Camunda Cockpit interface for a decision instance. The "Assign Approver" table has the following data:

R	Input	Output	Annotation
1	≤ 500	"accounting" * accounting	
2	≤ 800	"sales" = sales	
3	> 500	"management"	
4	≤ 60	"team assistance"	
5	[100..250]	"abteilungsleiter"	

Below the table, the "Inputs" tab is selected, showing the input values: Invoice Amount (Double) = 50 and Invoice Category (String) = Travel Expenses.

Enterprise Feature

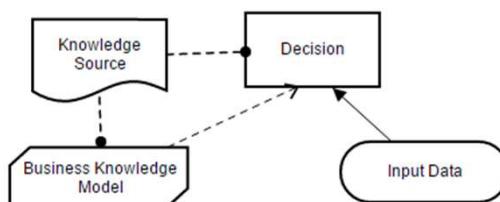
Live Editing of Decision Tables

	Input +	Output +	Approver Group	Annotation
R	Invoice Amount	Invoice Category		
	amount	invoiceCategory	result	
	double	string	string	
1	<= 500	-	"accounting"	-
2	<= 800	Travel Expenses	"sales"	-
3	> 500	-	"management"	-
4	<=60	"Misc"	"team assistance"	-
5	[100..250]	"Misc"	"abteilungsleiter"	-
+				

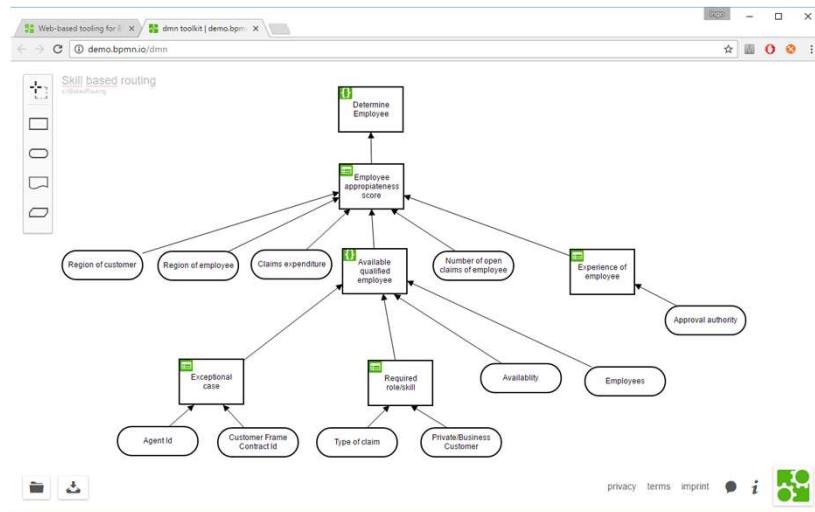


Beyond (Or Before) Decision Tables

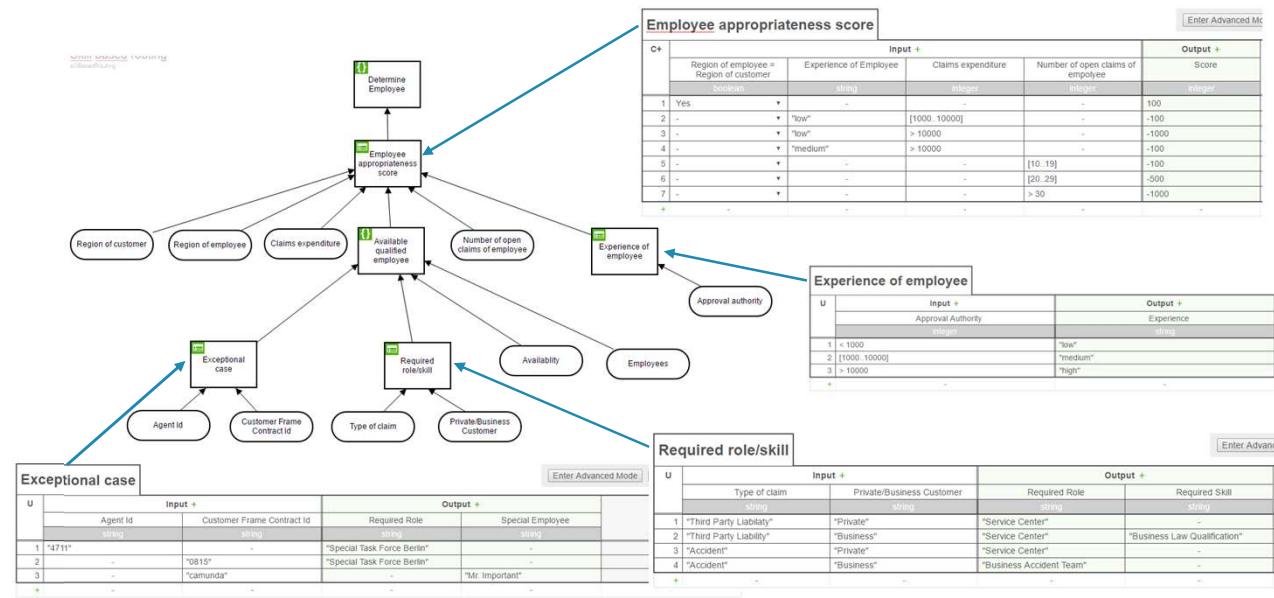
- DMN defines „Decision Requirements Diagram“
- Decision Requirements Diagrams will define the decisions [...], their interrelationships, and their requirements for decision logic
- Starting point for modeling complex decisions
- Elements of Decision Requirements Diagram



DRD Example: Skill Based Routing



Decisions Reference Decision Tables



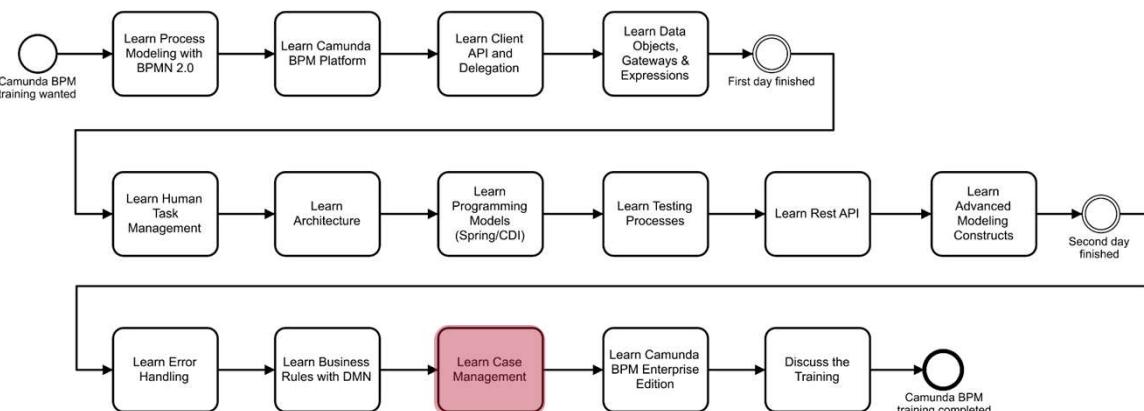


Exercise 9

Follow the provided instructions for exercise 9

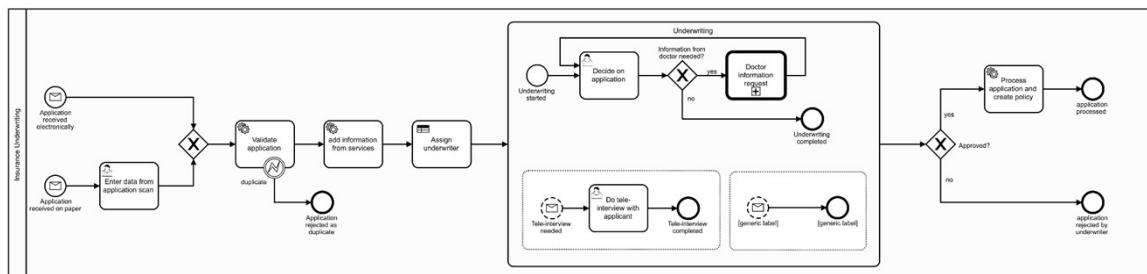


Camunda BPM training wanted

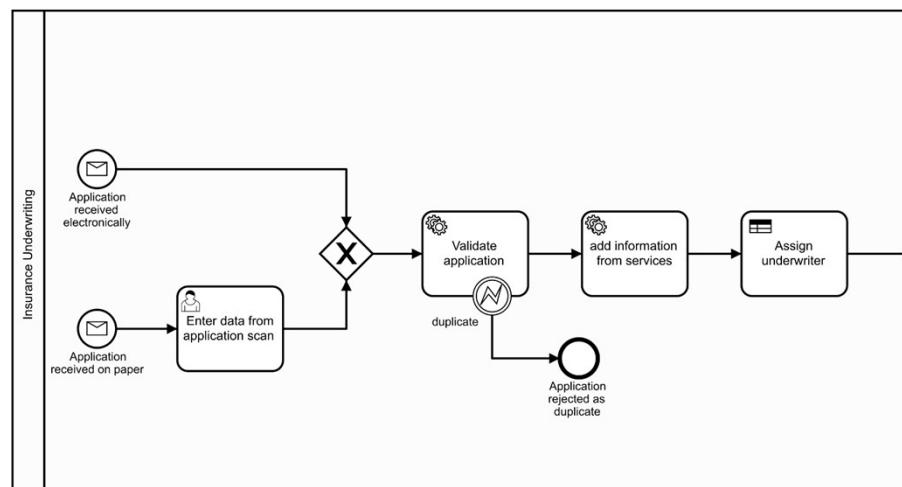




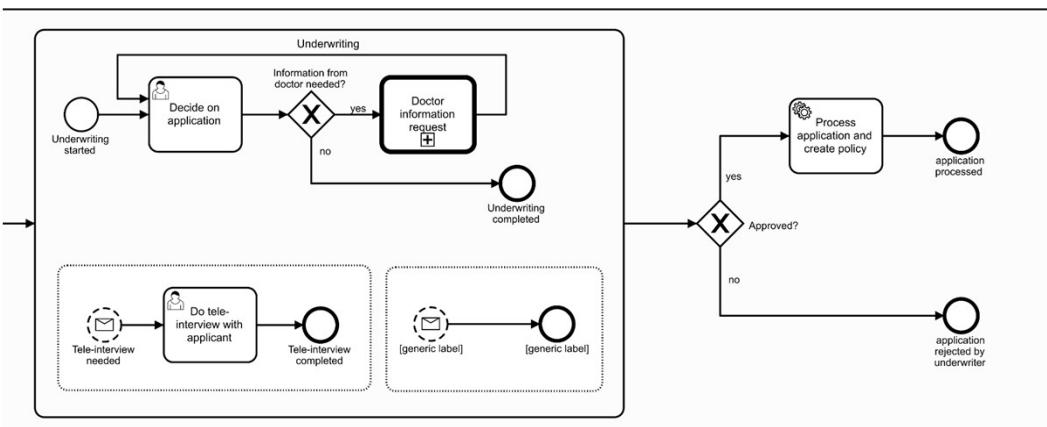
Example „Underwriting“



Example „Underwriting“

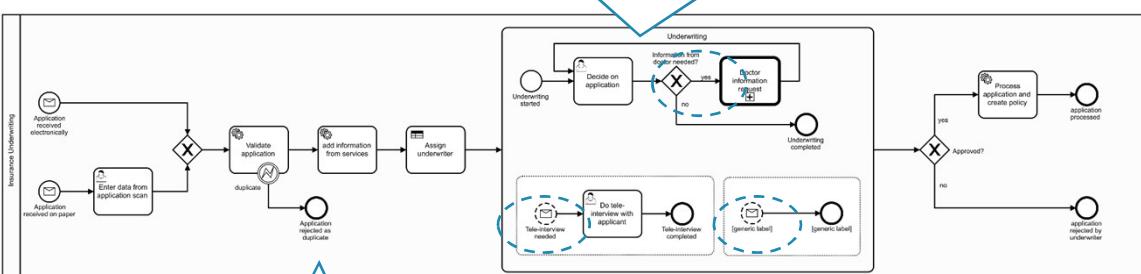


Example „Underwriting“



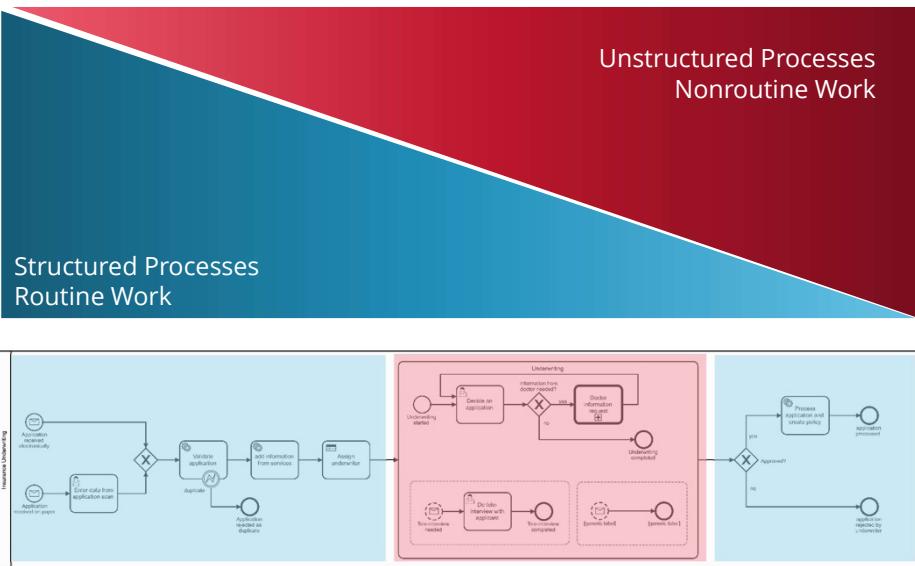
Example „Underwriting“

It is hard to give a stable ordering of the tasks! Flexibility needed.



There are workarounds – but this is hard to model in BPMN. You need more flexibility for the „Knowledge Worker“

Structured vs. Unstructured Work

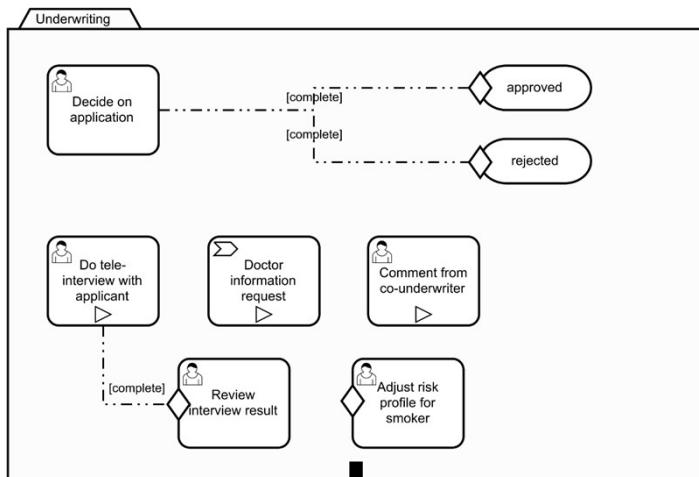


CMMN – Case Management Model and Notation

- First published by OMG in May 2014 as the BPMN-equivalent for „Cases“
- „Activities that are not so predefined and repeatable, but instead depend on evolving circumstances and ad hoc decisions by knowledge workers regarding a particular situation“
- Typical Use Cases:
 - application and claim processing in insurance
 - patient care and medical diagnosis in healthcare
 - exception handling, e.g. problem resolution in call centers, invoice discrepancy handling, data validation errors, ...
- See <http://www.omg.org/spec/CMMN/1.1/>



CMMN as Alternative



CMMN Explained

Human Task – automatically activated

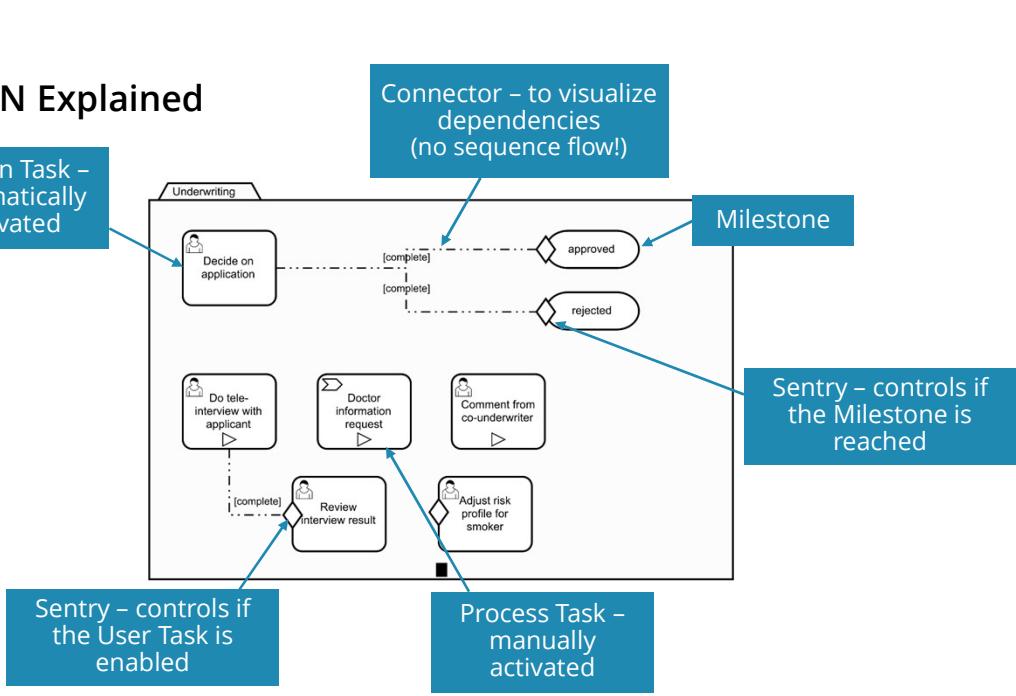
Connector – to visualize dependencies
(no sequence flow!)

Milestone

Sentry – controls if the Milestone is reached

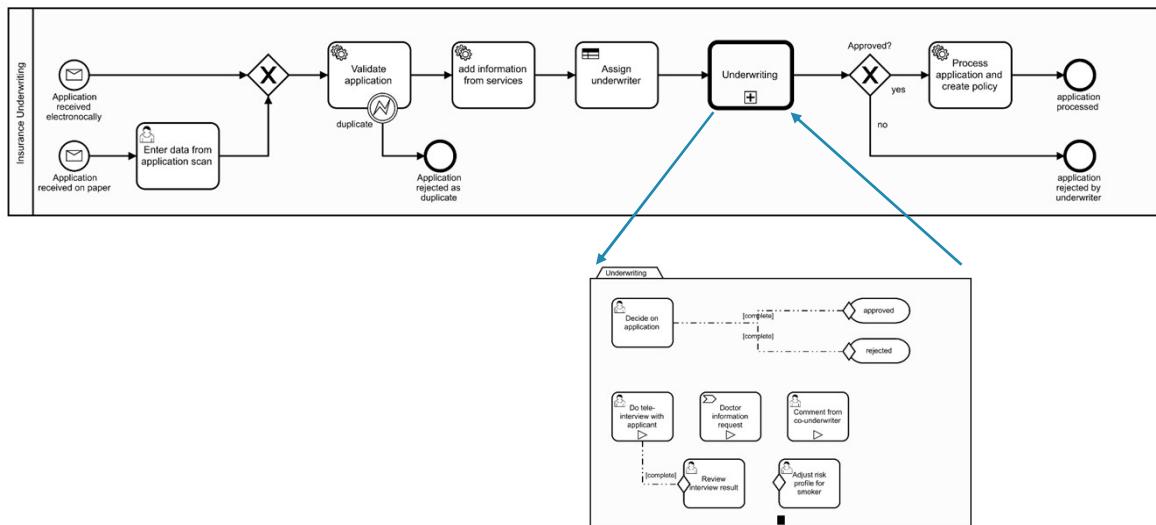
Sentry – controls if the User Task is enabled

Process Task – manually activated

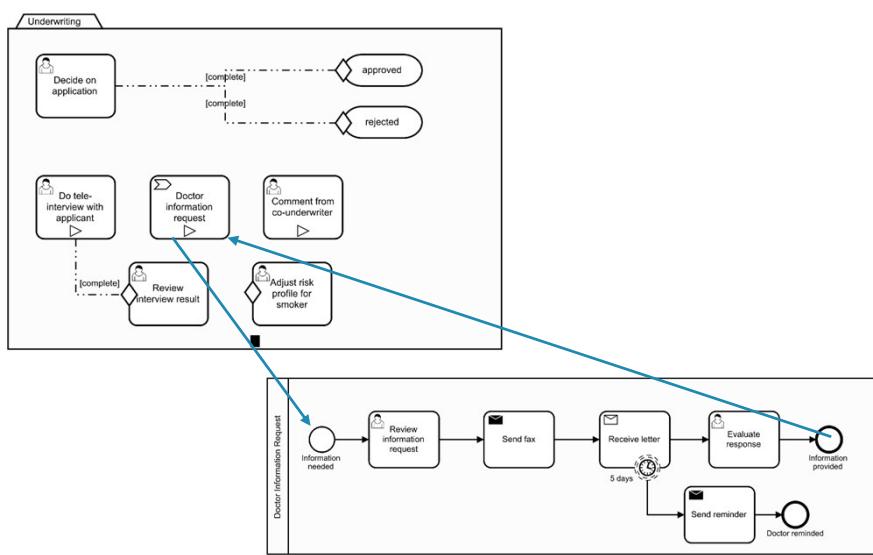




Stable and Easier Process



CMMN Calls BPMN





275

Case UI

The screenshot shows the Camunda BPM Underwriting Cockpit interface. At the top, there's a header with the title "Underwriting Cockpit" and navigation links for "New", "Cases", and "Tasks". Below the header is a search bar. The main content area is titled "Underwriting" and displays a task form for "Task: decide on application". The task form includes fields for "Customer ID: 47110815", "Customer Name: Bernd Ruecker", and "Evaluation Comments" which state "Patient has a great health status. Works much - but loves what he does. No concerns!". There's also an "Approve?" dropdown set to "Yes" with buttons for "Complete Task" and "Save". To the right of the task form are three panels: "Active Activities (Running)" containing "decide on application"; "Enabled Activities (Available)" containing "doctor information request" and "tele-interview with applicant"; and "Completed Activities" containing "comment from co-underwriter". The footer of the interface says "powered by Camunda BPM".



276

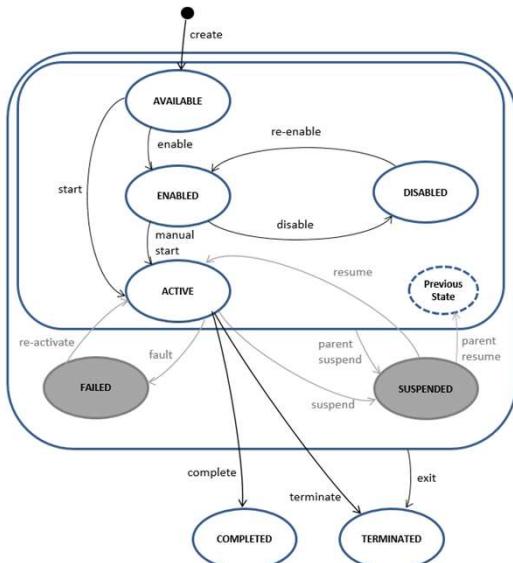
Example for Case Instance

This screenshot is similar to the one above, showing the Camunda BPM Underwriting Cockpit. It highlights specific components with callout boxes: "Task form" points to the central "Underwriting" task form; "Case context" points to the top-level "Underwriting" header; and "Case activities" points to the sidebar panels on the right. The task form itself contains the same "decide on application" task and its associated comments and approve options.



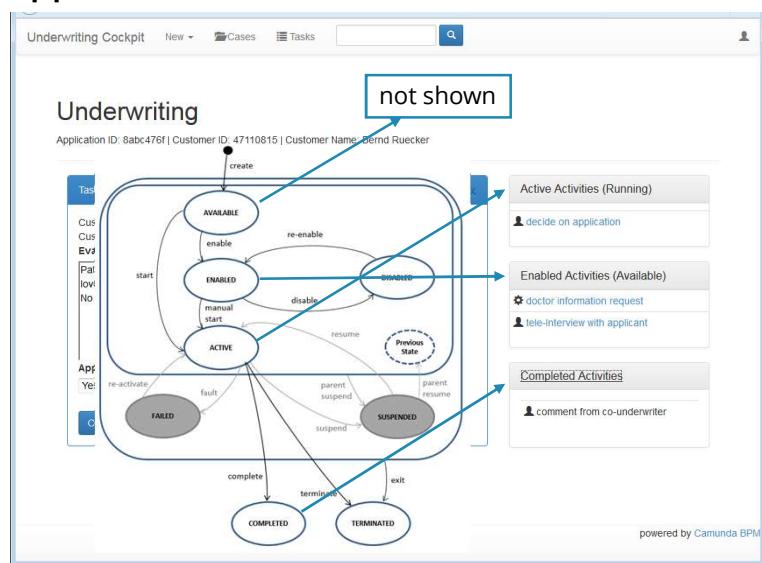
277

Task Lifecycle



278

Lifecycle Mapped on UI

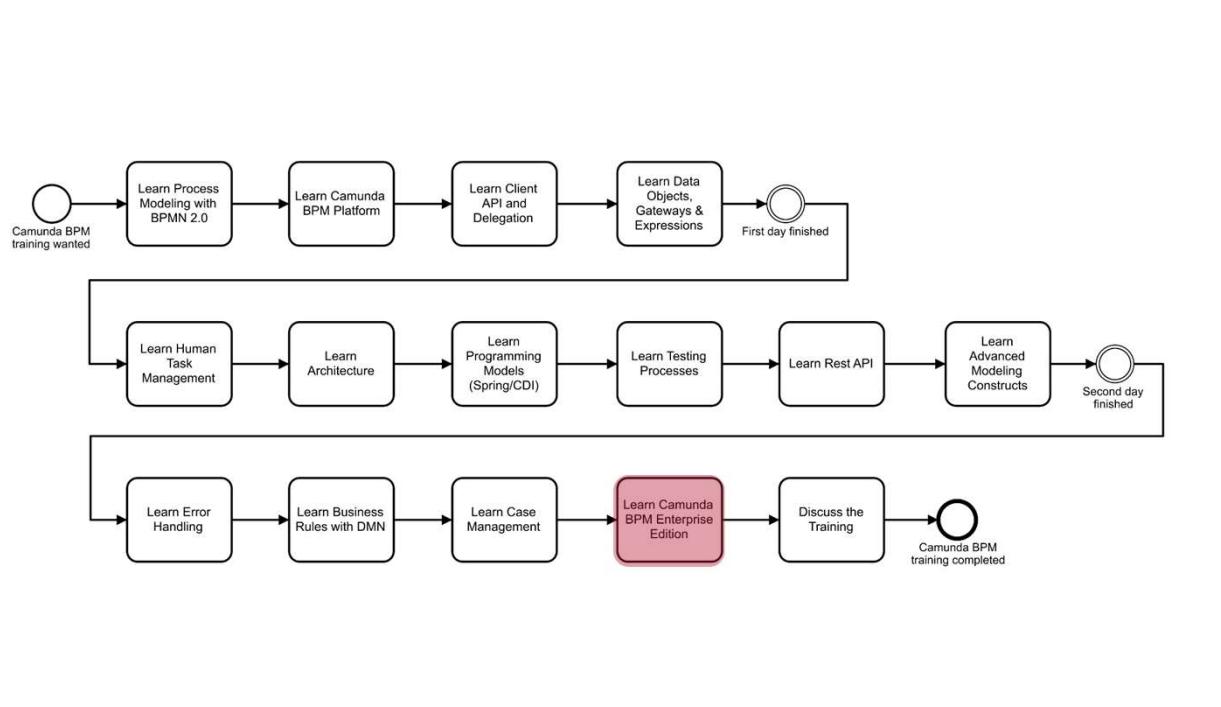


Enterprise Feature

Cockpit for Cases

The screenshot shows the Camunda Cockpit interface for a case instance. On the left, there's a sidebar with details like Instance ID, Case State (completed), Business Key (A-1639), and Definition Version (1). The main area displays a process diagram titled "Application check" with various states and transitions. Below the diagram is an "Audit Log" table showing activity instances:

State	Activity	Start Time	End Time	Activity Instance ID
terminated	Application decision	2017-01-16T12:14:26	2017-01-16T12:16:51	f0f11c37-dbdc-11e6-b25f...
terminated	Application rejected	2017-01-16T12:14:26	2017-01-16T12:16:51	f0f11c38-dbdc-11e6-b25f...
completed	Application accepted	2017-01-16T12:14:26	2017-01-16T12:16:51	f0f11c39-dbdc-11e6-b25f...
terminated	Approve decision	2017-01-16T12:14:26	2017-01-16T12:16:51	f0f11c3a-dbdc-11e6-b25f...
terminated	Application rejected	2017-01-16T12:14:26	2017-01-16T12:16:51	f0f11c3b-dbdc-11e6-b25f...
terminated	Application accepted	2017-01-16T12:14:26	2017-01-16T12:16:51	f0f11c3c-dbdc-11e6-b25f...





Enterprise Exclusive Cockpit Features

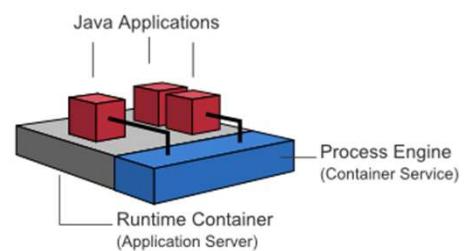
- Process Definition History
- Process Definition Heatmap
- Process Instance History
- Process Instance Migration
- Process Instance Modification
- Bulk Retry and Bulk Cancel
- Decision Table Live Editing
- Process Duration Report
- Redeploy Definitions

<https://camunda.com/bpm/enterprise/#cockpit>



Runtime Support

- Integration with IBM WAS
- Integration with Oracle WLS



<https://camunda.com/bpm/enterprise/#as>



SLA Based Support

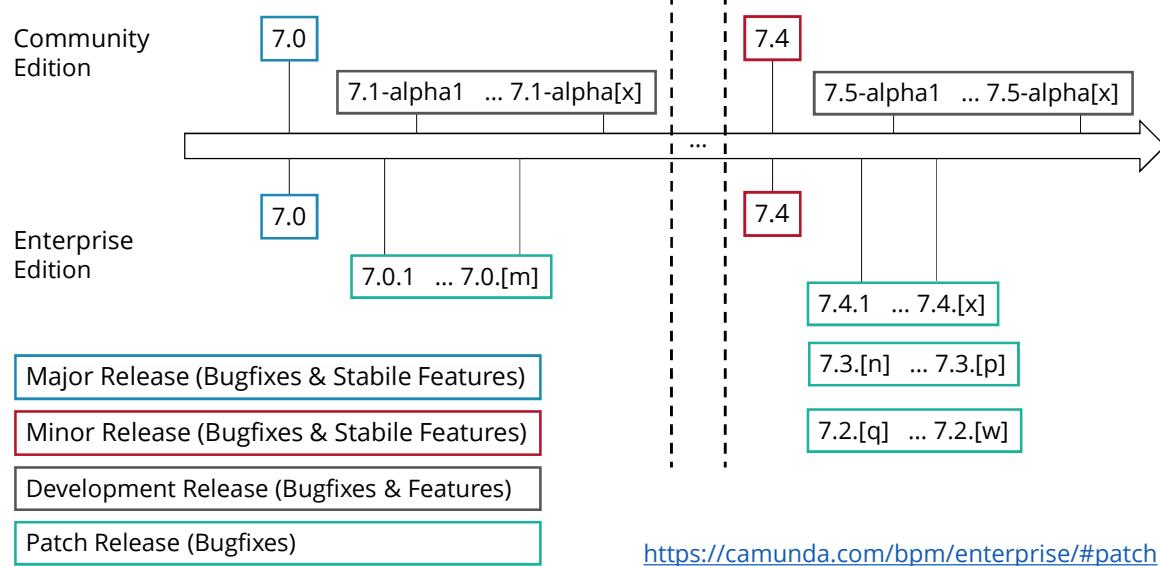
- Reliable support infrastructure with direct contact to core developers
- Service Level Agreements (SLA)

Level	Name	Description	Availability		Max. response time	
			Standard SLA	Advanced SLA	Standard SLA	Advanced SLA
L1	Blocker	Core components (i.e., process engine) of Camunda BPM do not work at all/produce critical errors that prevent usage in production mode.	8x5	24x7	8 business hours	2 hours
L2	Critical	Usage of Camunda BPM seriously affected, a workaround is urgently needed.	8x5		8 business hours	
L3	Help Request	Non-critical errors, Help Requests, Feature Requests.	8x5		16 business hours	

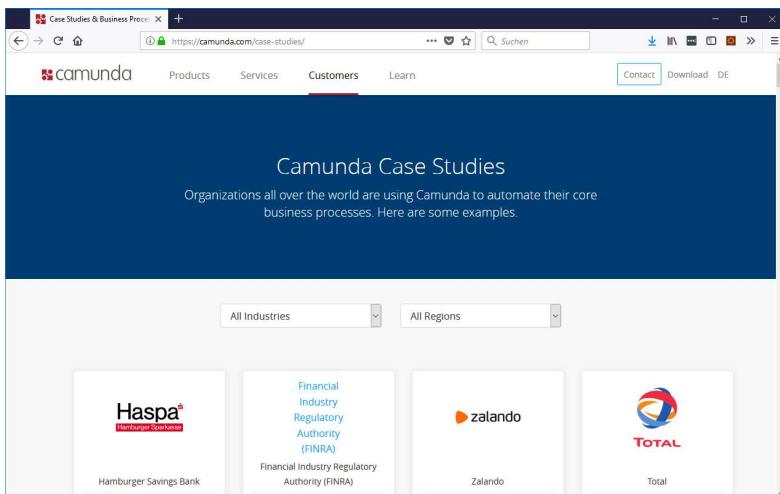
<https://camunda.com/bpm/enterprise/#support>



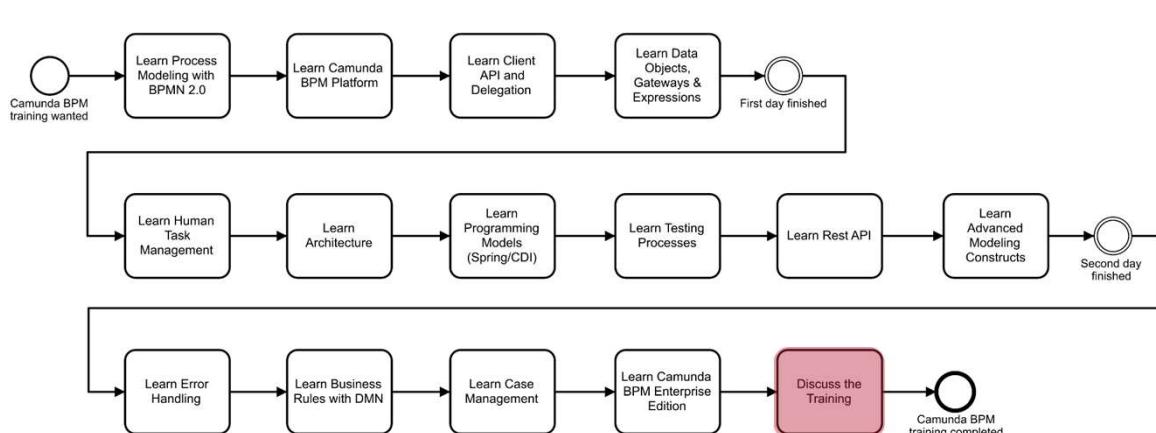
Patch Releases



References



<https://camunda.com/case-studies/>





287

How to Get Updates From the Team?

Blog: <https://blog.camunda.org/>



Whitepapers: <https://camunda.com/learn/whitepapers/>

Events: <https://camunda.com/events/>

Where to meet us

Whether it's at an exhibition booth, a local Meetup or our own event, we're always keen to have a chat with you so drop by and say hello!

DE Roadshow 2018
Meet our team in Germany, Switzerland and Austria.

CamundaCon
Camunda's annual 2-day user Conference in Berlin.

Meetups
Local Meetups, organized by our amazing community.

Conferences
Visiting a developer's conference? There's a good chance we're there.

[More Info and Dates](#) [CamundaCon](#) [Meetups](#) [Conferences](#)

Forum: <https://forum.camunda.org/>

The screenshot shows a forum interface with a header for 'all categories' and various filter options like 'Latest', 'New', 'Unread (85)', 'Top', and 'Categories'. Below the header is a table listing forum topics. Each topic row includes a small profile picture, the category name (e.g., 'Modeler', 'Process Engine', 'Meta'), and the number of replies. The topics listed include: 'Process not found on Camunda platform after setup/deployment', 'Link to external documentation?', 'Get the task instance from a execution listener', 'REST API request to fill a form', 'Bug using Camunda Date', 'Minor typo in BPMN tutorial', 'JsonLogic in Camunda', 'Single-Sign-On in Camunda', and 'Camunda - Getting certificate error while calling an https REST service through connector'.

Follow us on Twitter: @camundabpm



288

Camunda Training Offering

Overview trainings

BPM-trio overview (2 days)

- Values of BPM
- Introduction into BPMN, CMMN, DMN
- Benefits of the BPM trio
- Outlook: Process automation

Camunda BPM Overview (1 day)

- Introduction into BPMN
- Hands on Camunda BPM
- Monitoring & Reporting
- How to start with Camunda

OCEB training (2 days)

- Prepare for the OCEB BPM certification

Modeling trainings

BPMN (3 days)

- BPMN in detail
- Implementing BPM-projects with BPMN
- BPMN real world examples
- BPMN in context of CMMN & DMN

DMN (1 day)

- DMN in detail
- Decision design
- Complex decisions with DRDs
- DMN in context of BPMN & CMMN

CMMN (1 day)

- Why CMMN?
- CMMN in detail
- CMMN real world examples
- CMMN in context of BPMN & DMN

Development trainings

Camunda BPM for developers (3 days)

- Introduction BPMN
- Camunda BPM Architecture
- Automating processes with Camunda
- Testing, deployment, versioning

Camunda BPM DevOps (2 days)

- Camunda BPM installation
- Monitoring & alarming
- Process version migration
- Advanced DevOps topics

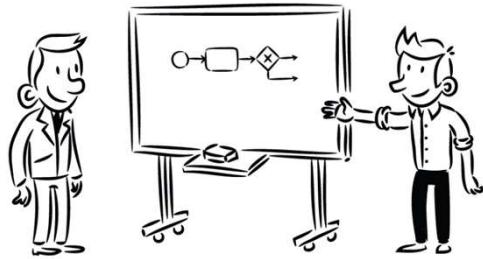
Details see

<https://camunda.com/services/training/>



Want to Learn More BPMN?

- 3-day training
- Learn in-depth BPMN 2.0
 - Understand BPMN
 - Apply BPMN
 - Introduce BPMN



Feedback

Please fill out this form:

<https://camunda.com/services/training/feedback/>

The screenshot shows a web browser window with the URL <https://camunda.com/services/training/feedback/>. The page title is "Training Feedback". The form contains the following fields:

- * 1. Trainer: A dropdown menu.
- * 2. Training: A dropdown menu set to "Camunda BPM Basic".
- * 3. Date of training: A date input field with placeholder "Date / Time" and separate fields for DD, MM, and YYYY.
- 4. What is your role?: A text input field.
- * 5. Training Management: A horizontal rating scale with five options: Excellent, Good, Not Great, and Terrible.