



# Camunda BPM and Microservices

A polyglot environment

Version 2021-01

## Agenda

### Day 1:

1. Process Modeling with BPMN 2.0
2. Camunda BPM Platform
3. Deploying to Camunda Engine
4. Work with External Service Tasks
5. Data Objects, Gateways & Expressions

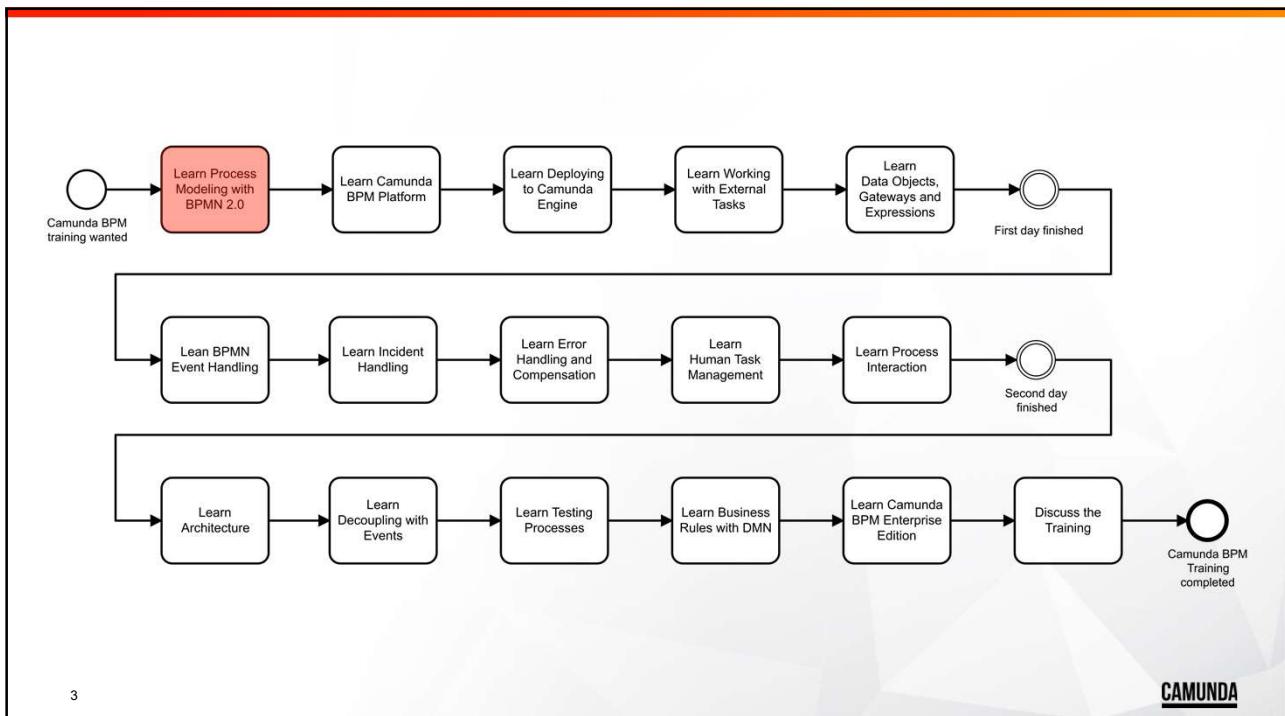
### Day 2:

6. BPMN Event Handling
7. Incident Handling
8. Error Handling and Compensation
9. Human Task Management
10. Process Interaction

### Day 3:

11. Architecture
12. Decoupling with Events
13. Testing Processes
14. Business Rules with DMN
15. Camunda BPM Enterprise Edition
16. Wrap up

CAMUNDA



## 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

**NIST**

**ORACLE**

**CAMUNDA**

**SAP**

**IBM**

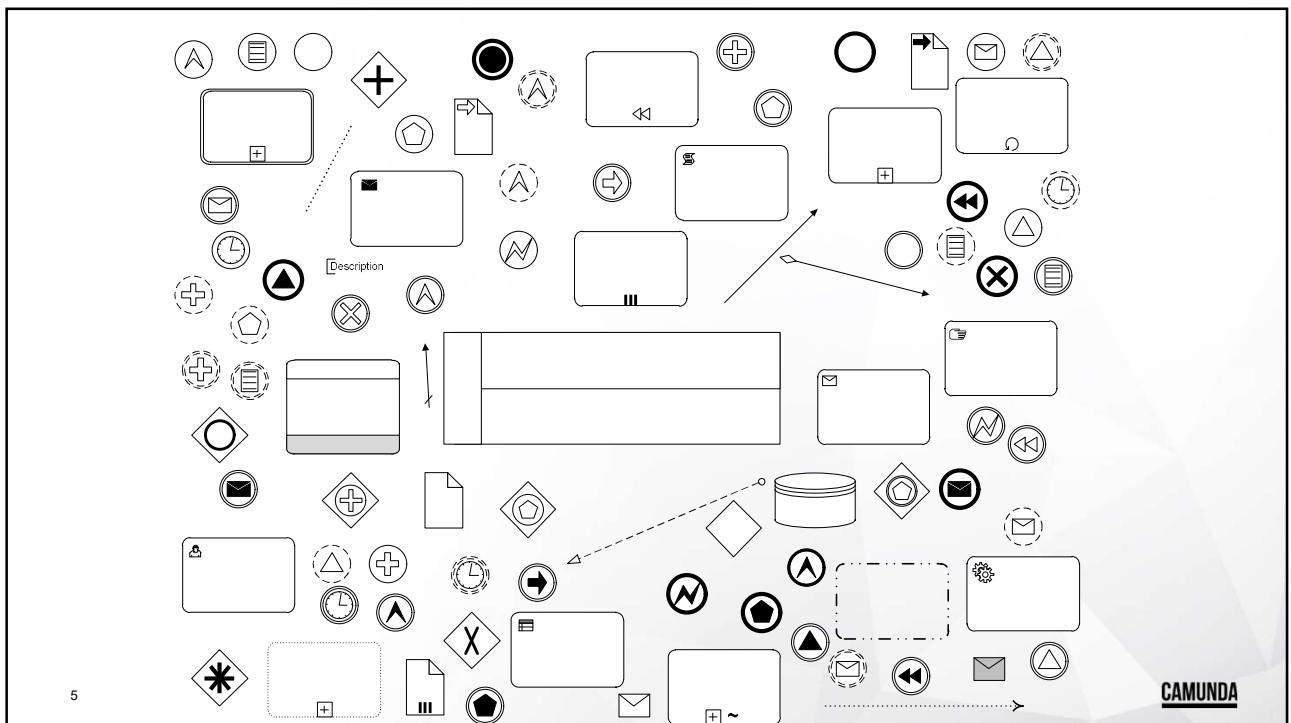
**Trisotech**

**Global360**

**TIBCO**

**FUJITSU**

**INTALIO**



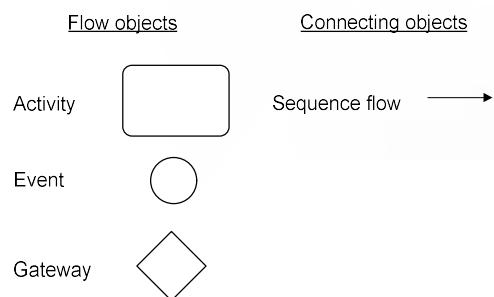
## Our First BPMN Process Model



6

CAMUNDA

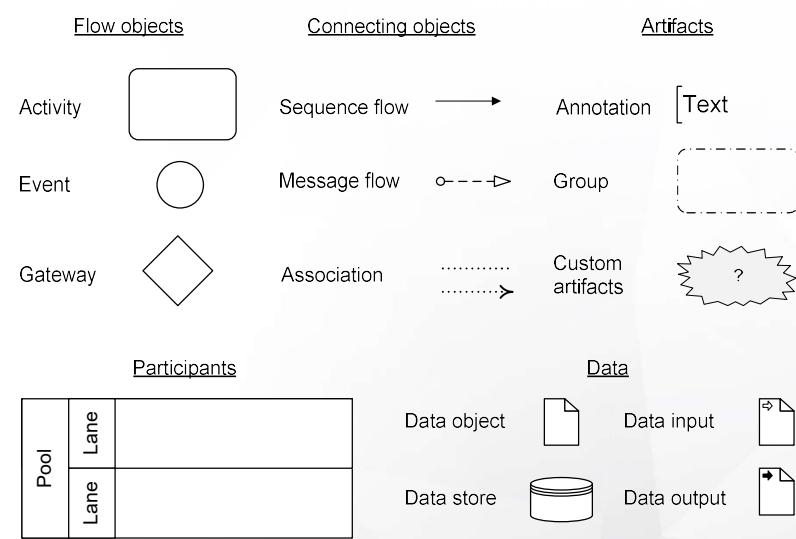
## Basic Elements



7

CAMUNDA

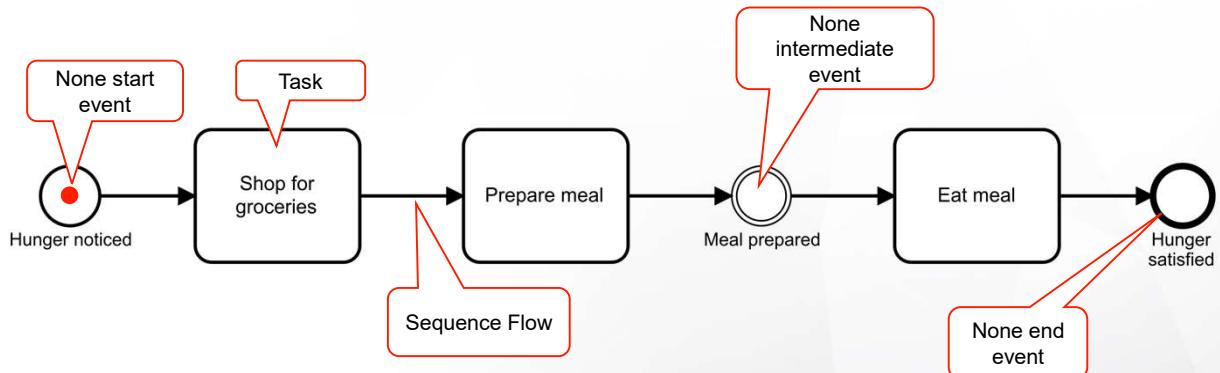
## Basic Elements - Documentation



8

CAMUNDA

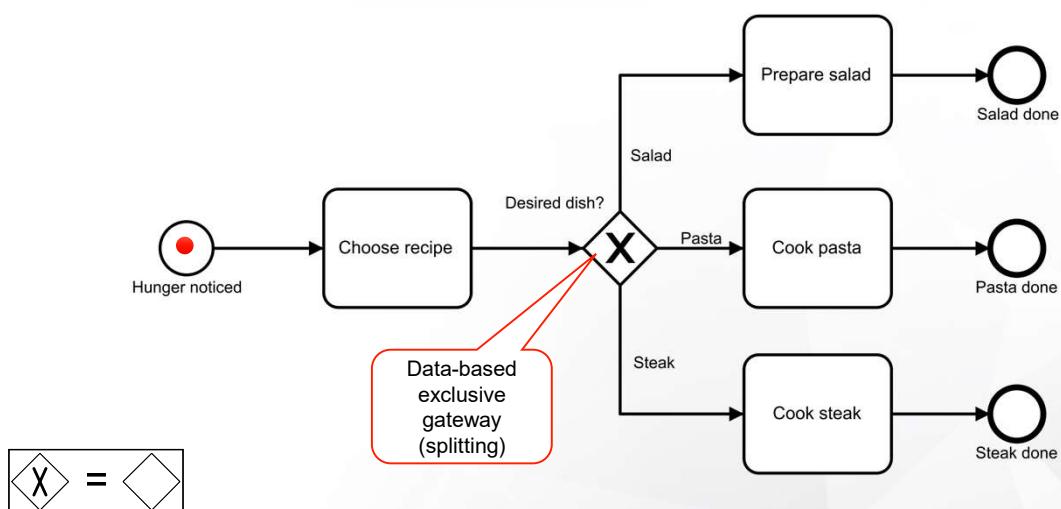
## Tasks and events



9

CAMUNDA

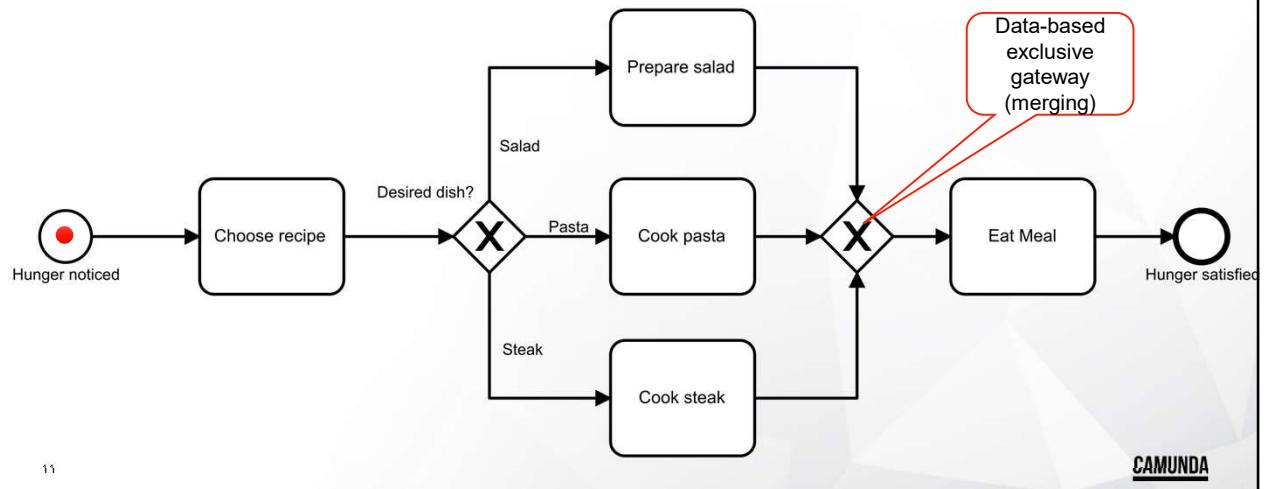
## The XOR-Gateway



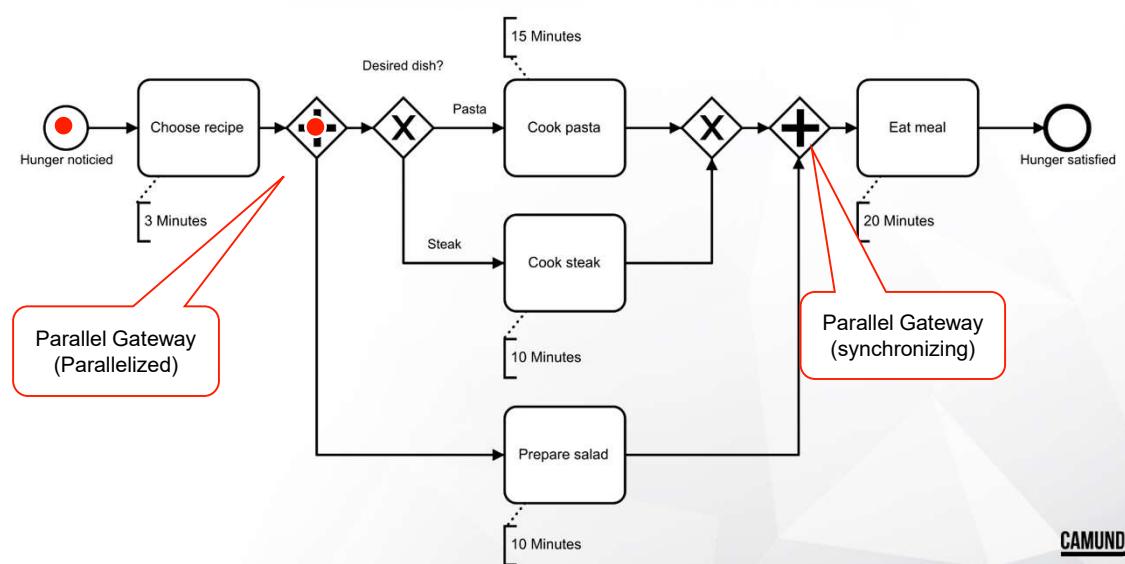
10

CAMUNDA

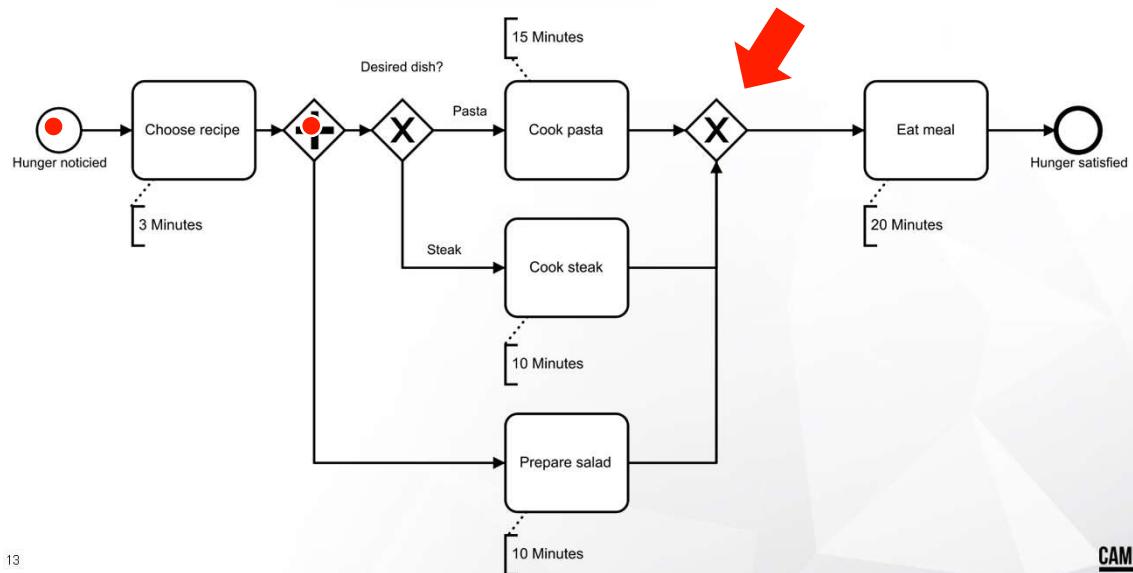
## XOR-Gateways Can Also Merge



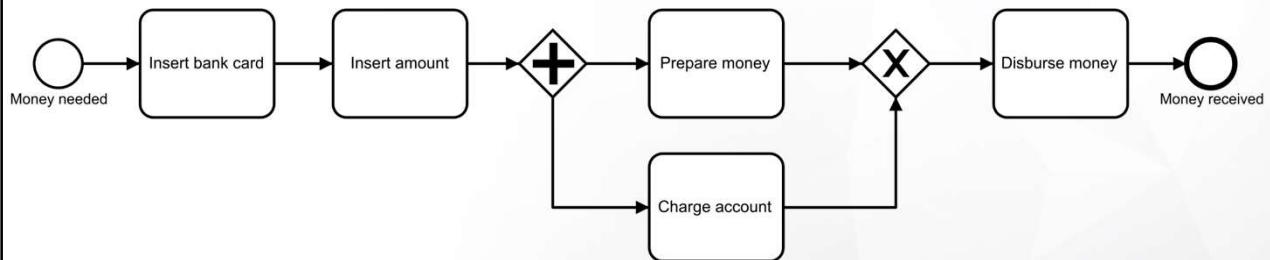
## Preparing Salad and Main Course at the Same Time



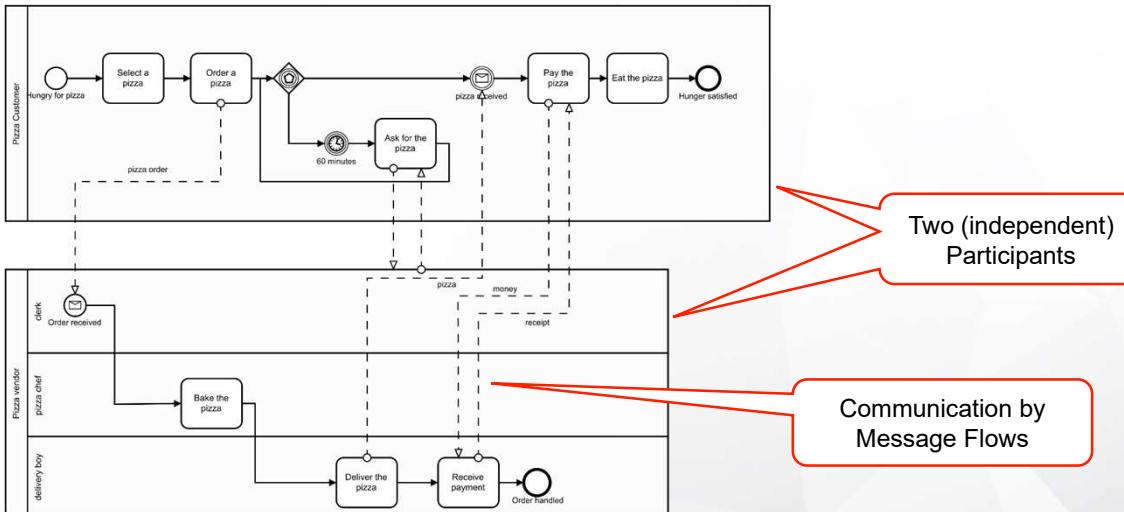
## What Happens in This Process?



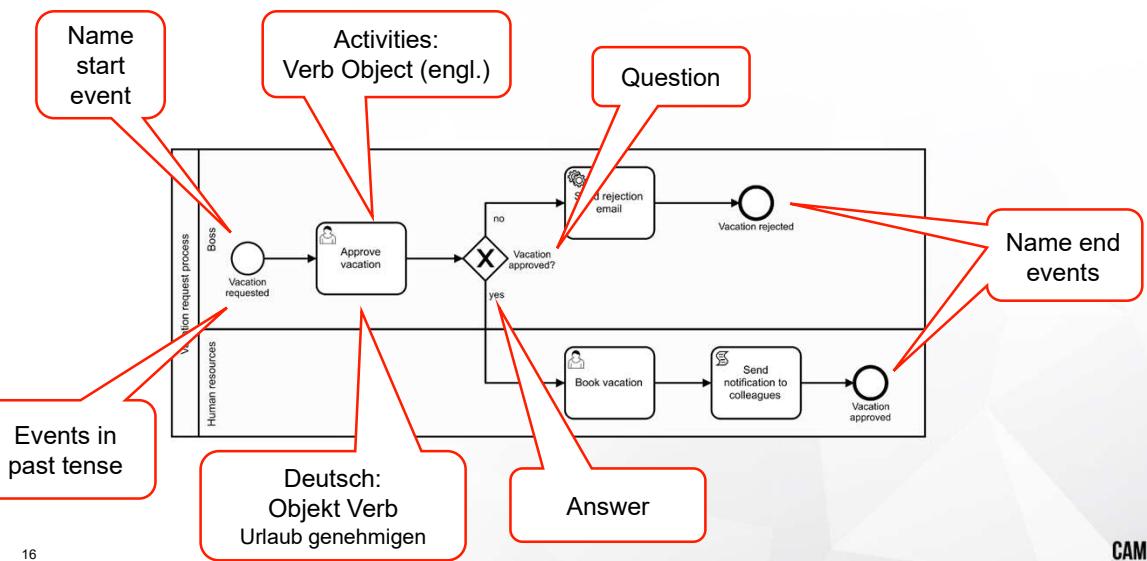
## Example: ATM

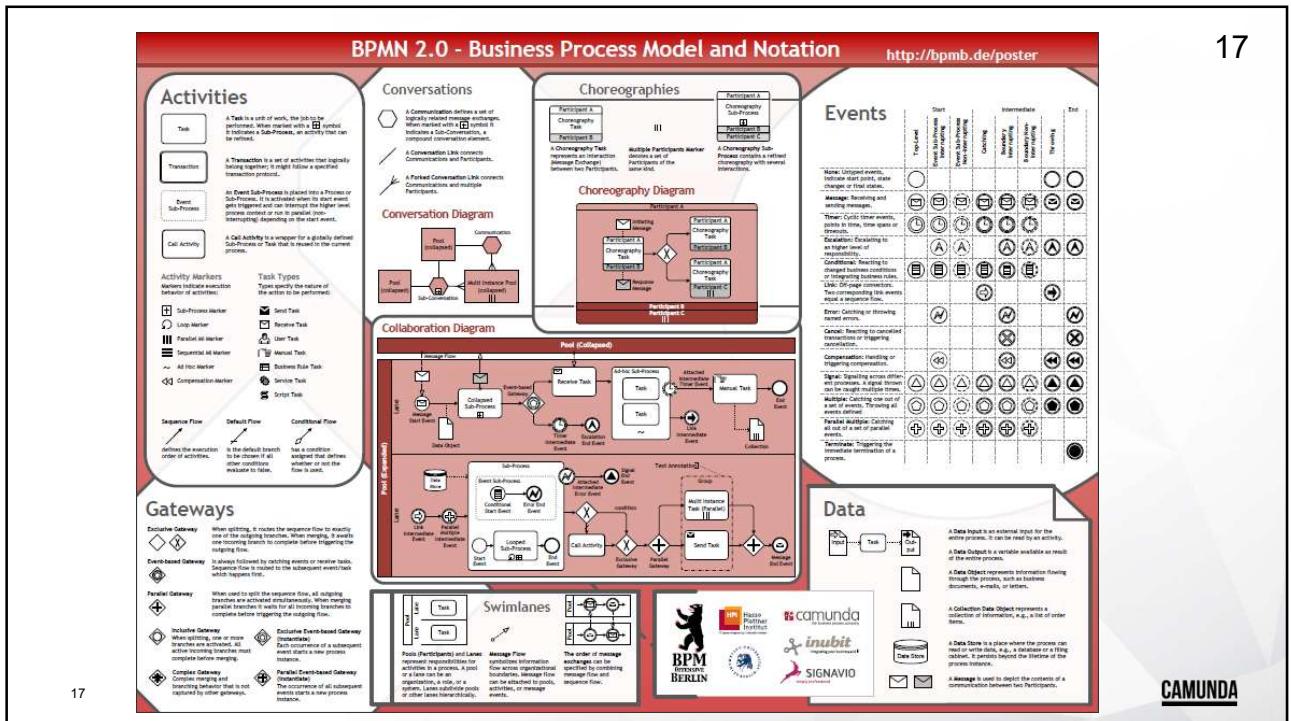


## Collaboration Diagram



## Create Readable Process Models



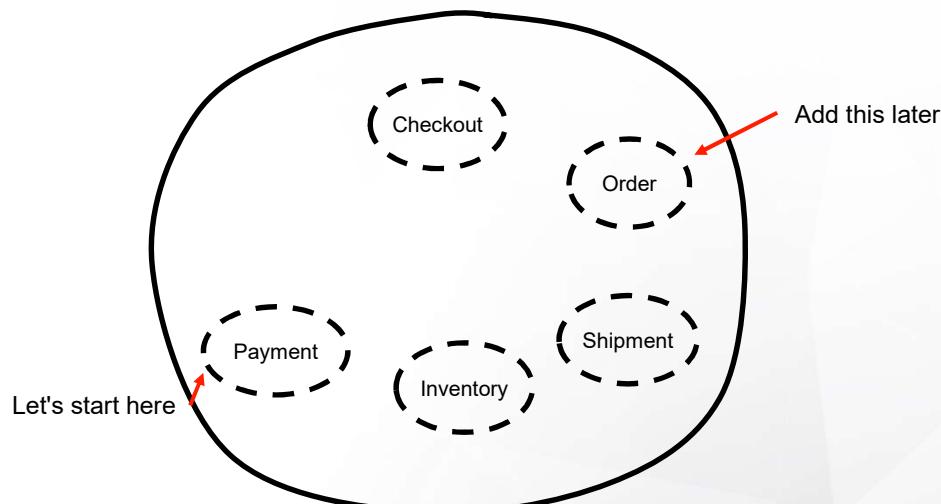


CAMUNDA

## Scenario for Exercises



## A Simple Order Process



**Domain:** a web shop

**Bounded Contexts:**

- Order
- Shipment
- Payment
- Checkout
- Inventory

A lot of freedom here,  
It's just an example

CAMUNDA

## Let's Do the Payment Process

- The customer can have a credit with us
- We use credit card payment
- If the credit is not sufficient we charge the remaining amount

20

CAMUNDA



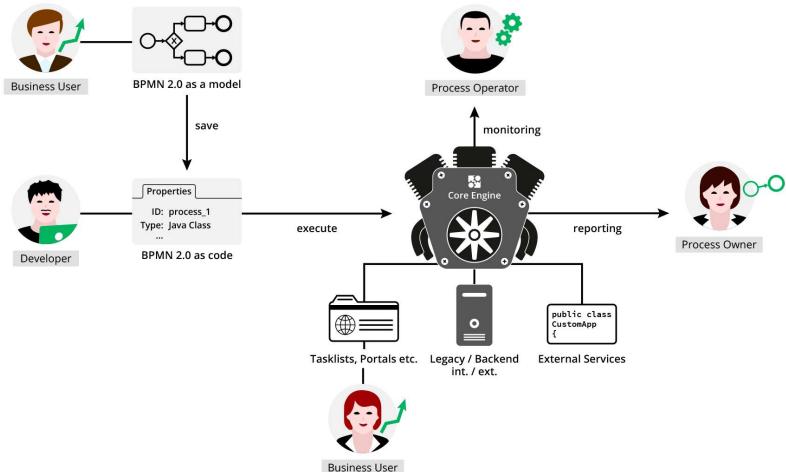
## Group Exercise 0

Use a whiteboard or flipchart  
to model a payment process



## Payment Process

## What's So Cool About BPMN 2.0



23

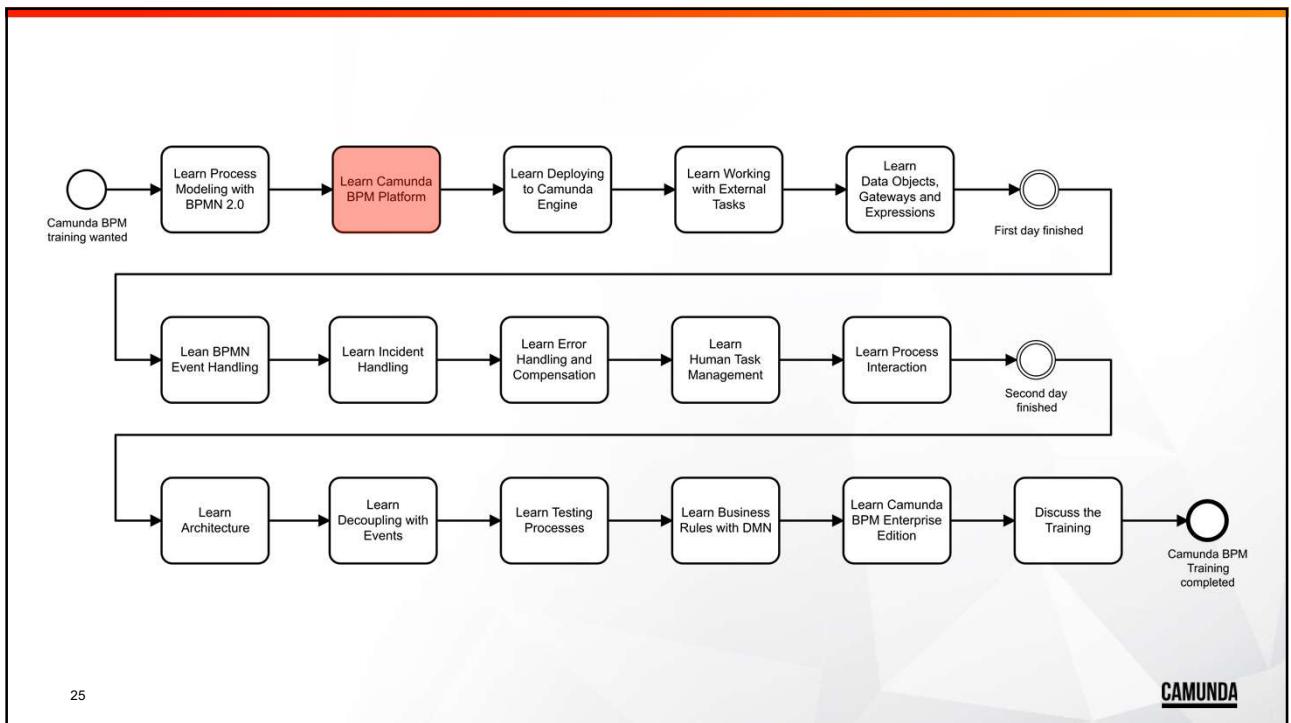
CAMUNDA

## Why Using a Process Engine?

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

24

CAMUNDA



25

**CAMUNDA**

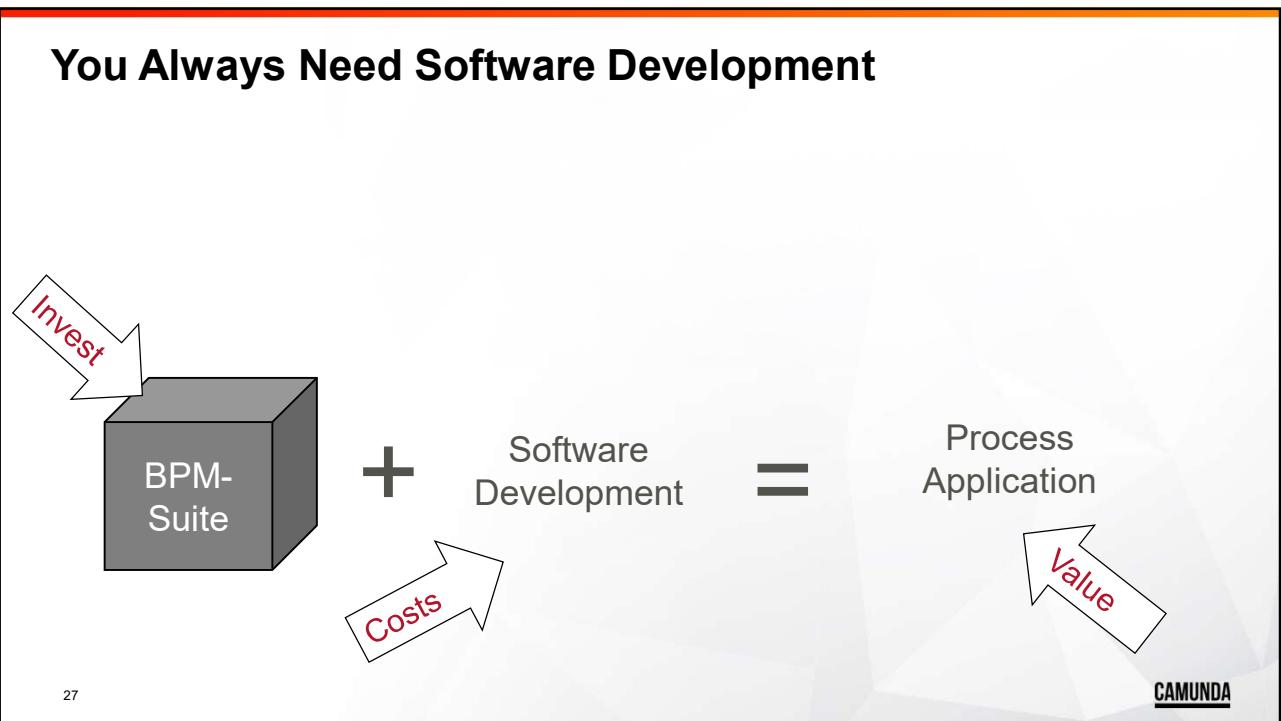
## Shiny BPM Suites?



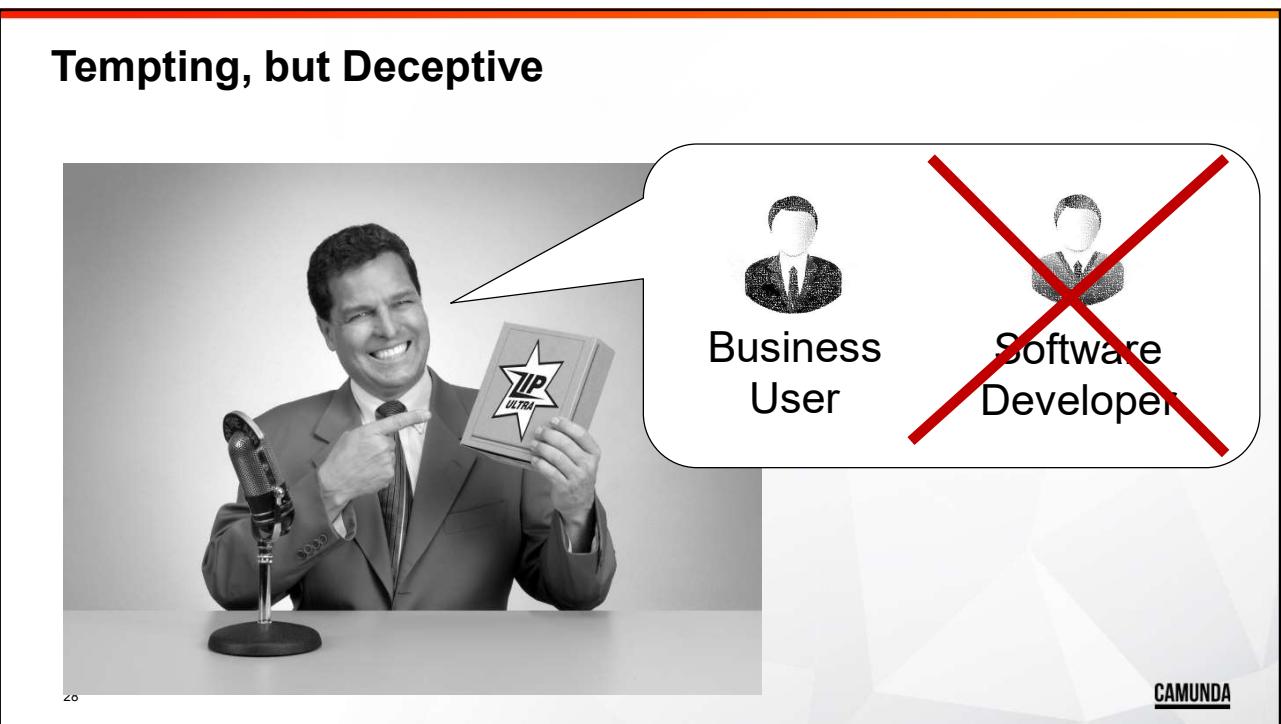
26

**CAMUNDA**

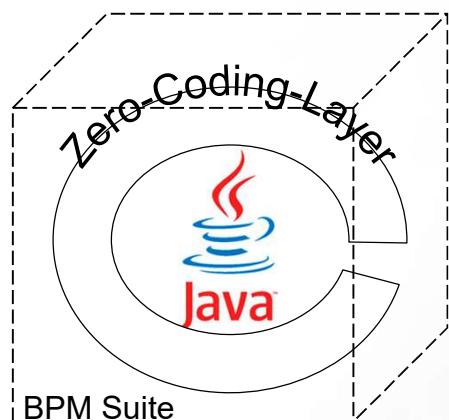
## You Always Need Software Development



## Tempting, but Deceptive



## A Fundamentally Wrong Approach\*



complicated  
restrictive

restrictive  
proprietary



Business User



Software  
Developer

CAMUNDA

<sup>29</sup> \*for automating core business processes

## Our Approach: Developer Friendly BPM

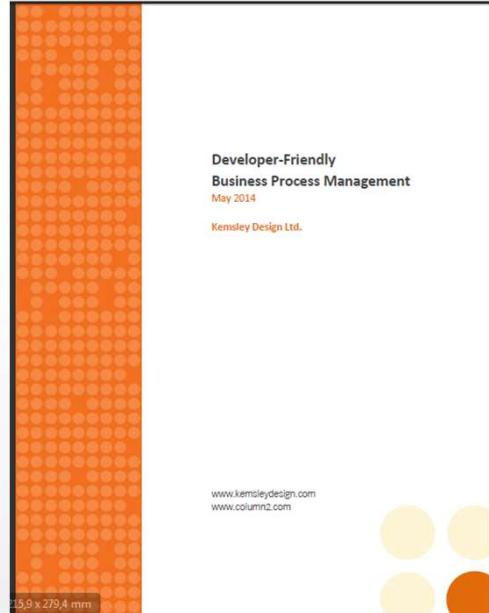


## Developer-Friendliness

- A fundamentally different approach compared to Zero-Code Suites!
- It's not about „empowering Business Analysts“. Even with low code approaches coding is still required
- Camunda helps developers be more productive („Less code“)
- Camunda is microservice orchestration ready

<https://camunda.com/learn/whitepapers/developer-friendly-bpm/>

31



## In short

Camunda

Developer-Friendly  
System Integration

VS.

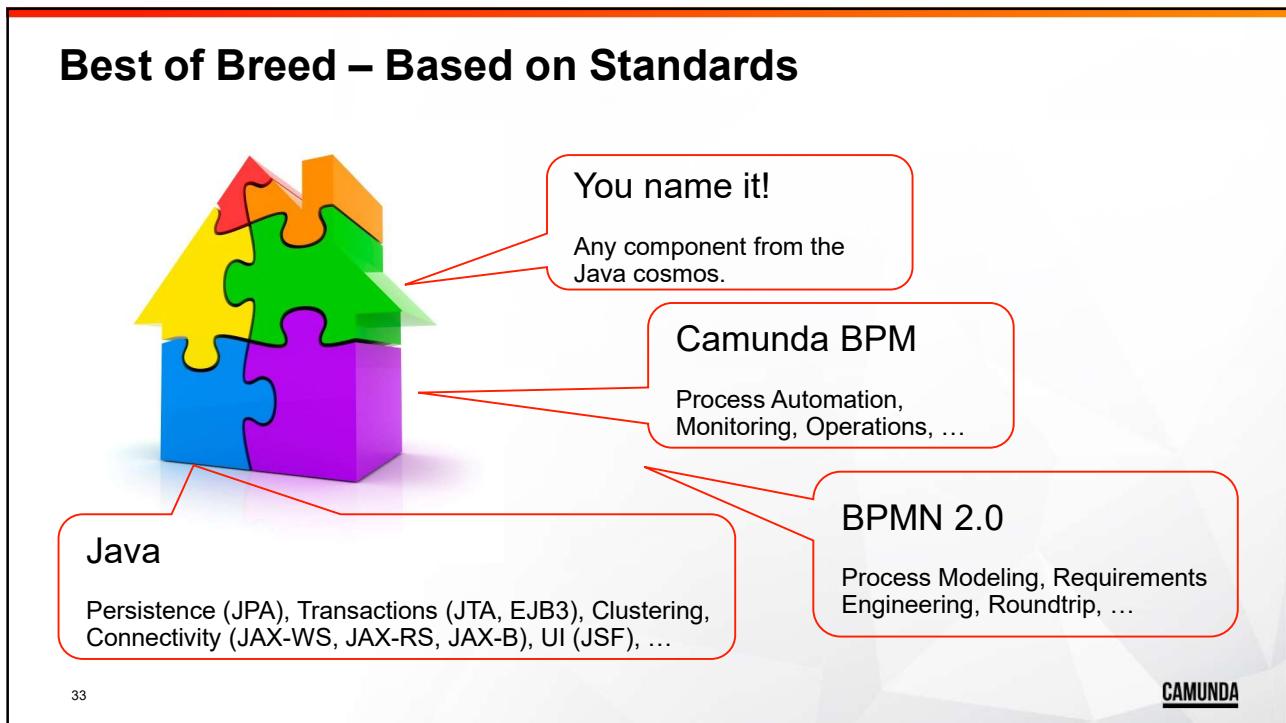
Zero-Code BPM Suites

Death by  
Property Panel

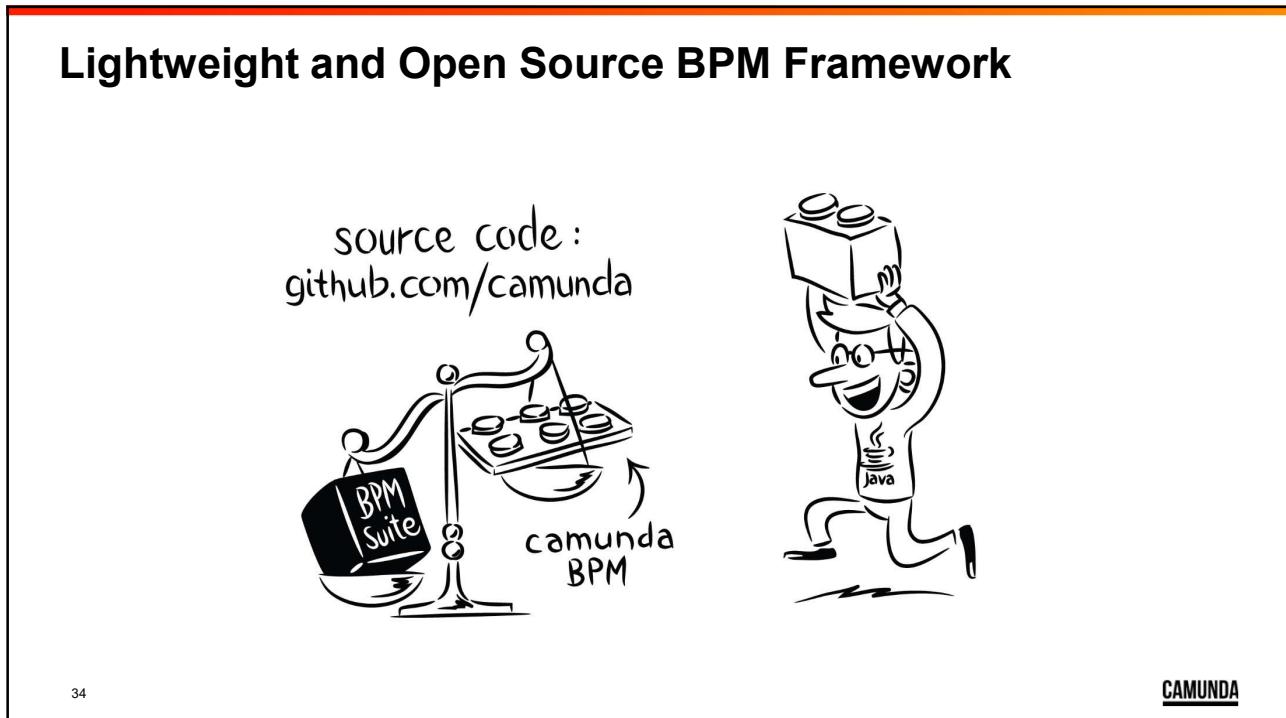
32

CAMUNDA

## Best of Breed – Based on Standards



## Lightweight and Open Source BPM Framework



## ...Plus BPMN Based Business-IT-Alignment

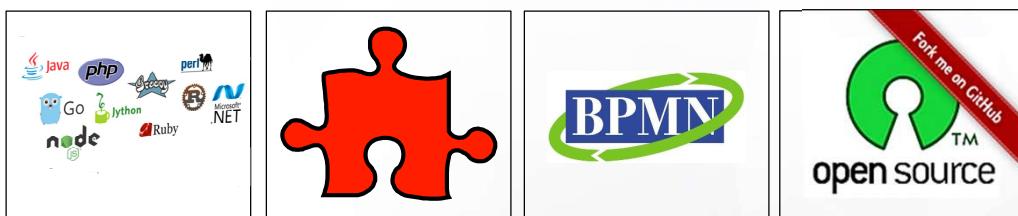
35



CAMUNDA

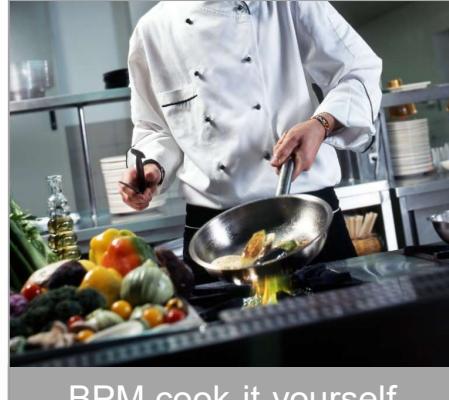
## Summary: Core Concepts of Camunda BPM

36



CAMUNDA

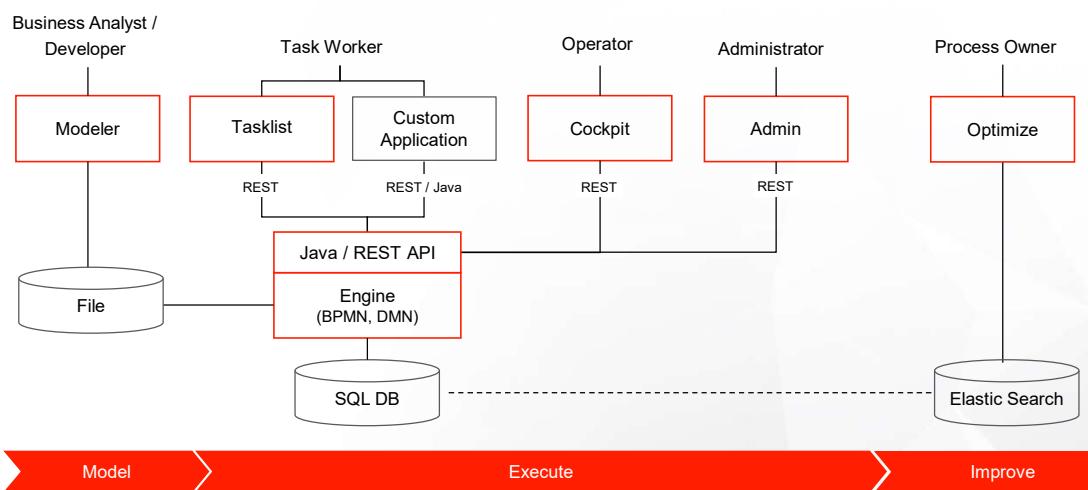
## Decide Yourself!



37

CAMUNDA

## Camunda BPM Components



38

CAMUNDA



# Exercise 1

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks



Install  
Camunda  
BPM



# Demo



Camunda  
BPM  
Platform

## Camunda Cockpit

The screenshot shows the Camunda Cockpit interface. On the left, there is a sidebar with various configuration details such as Definition Version, Definition ID, Definition Key, Definition Name, History Time To Live, Tenant ID, Deployment ID, Instances Running, and Related. The main area features a BPMN process diagram with a heatmap overlay, indicating activity levels across the process steps. Below the diagram is a table titled 'Process Instances' showing three completed instances with their start and end times and business keys.

State	ID	Start Time	End Time	Business Key
completed	1e523c77-67fe-11e8-8725-00155d230939	2018-06-01T17:30:27	2018-06-01T17:30:27	A-00375
completed	1e275b9c-67fe-11e8-8725-00155d230939	2018-06-01T17:05:06	2018-06-01T17:05:06	A-00374
completed	1ddb10ed-67fe-11e8-8725-00155d230939	2018-06-01T16:11:06	2018-06-01T16:11:06	A-00373

**History view and Heatmap Enterprise Feature**

41

CAMUNDA

## Cockpit Features

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

42

CAMUNDA

## Camunda Admin

The screenshot shows the Camunda Admin interface. At the top, there's a navigation bar with links for 'Users', 'Groups', 'Tenants', 'Authorizations', and 'System'. Below the navigation, there are five main sections:

- Users**: Contains links for 'Create New User', 'List of Users', and 'My Profile'.
- Groups**: Contains links for 'Create New Group' and 'List of Groups'.
- Tenants**: Contains links for 'Create New Tenant' and 'List of Tenants'.
- Authorizations**: Contains a single link 'Manage Authorizations'.
- System**: Contains links for 'General', 'Execution Metrics', and 'License Key'.

At the bottom of the page, there are footer links for 'Date and Time displayed in local timezone: Europe/Berlin' and 'Powered by camunda BPM / v7.9.0-ea'. The 'CAMUNDA' logo is also present in the bottom right corner.

43

## Authorizations

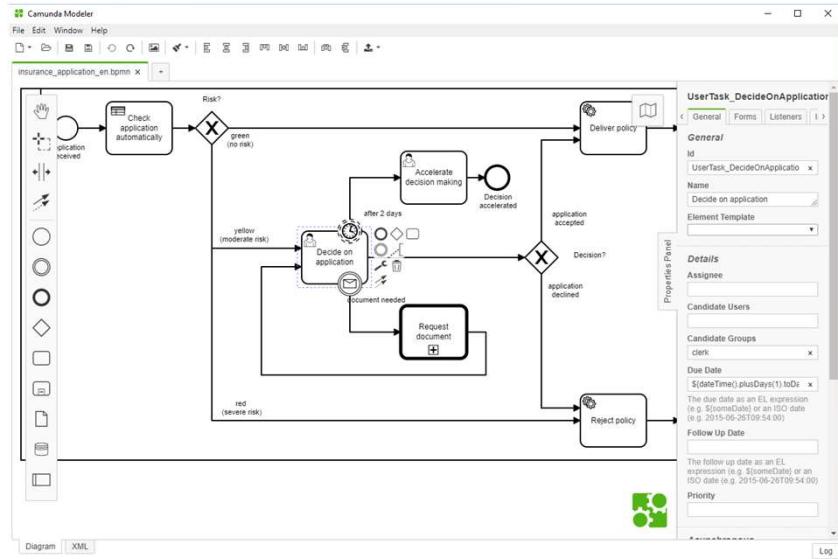
The screenshot shows the 'Authorizations' section of the Camunda Admin interface. On the left, there's a sidebar with a tree view of application components, including 'Process Definition' which is currently selected. The main area displays a table titled 'Process Definition Authorizations'.

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 of the page, there are footer links for 'Date and Time displayed in local timezone: Europe/Berlin' and 'Powered by camunda BPM / v7.9.0-ea'. The 'CAMUNDA' logo is also present in the bottom right corner.

44

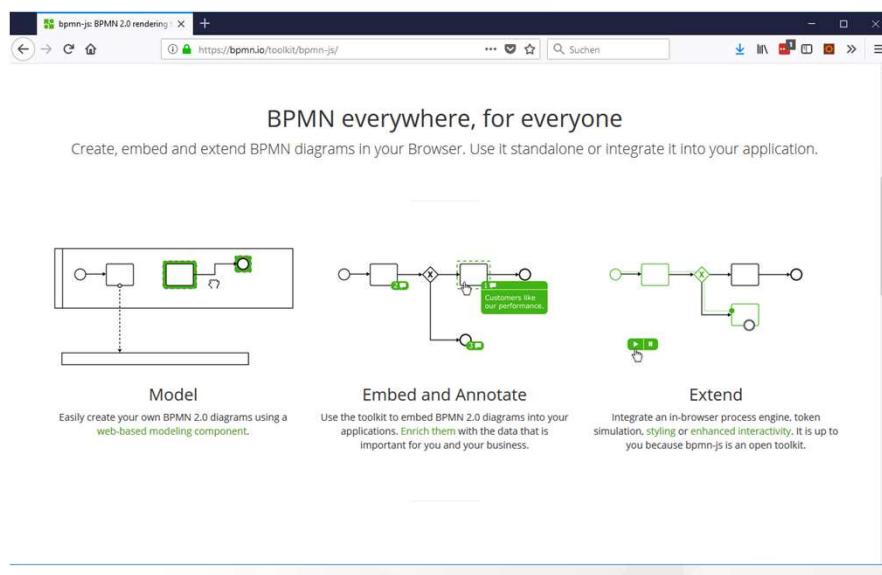
## Camunda Modeler



45

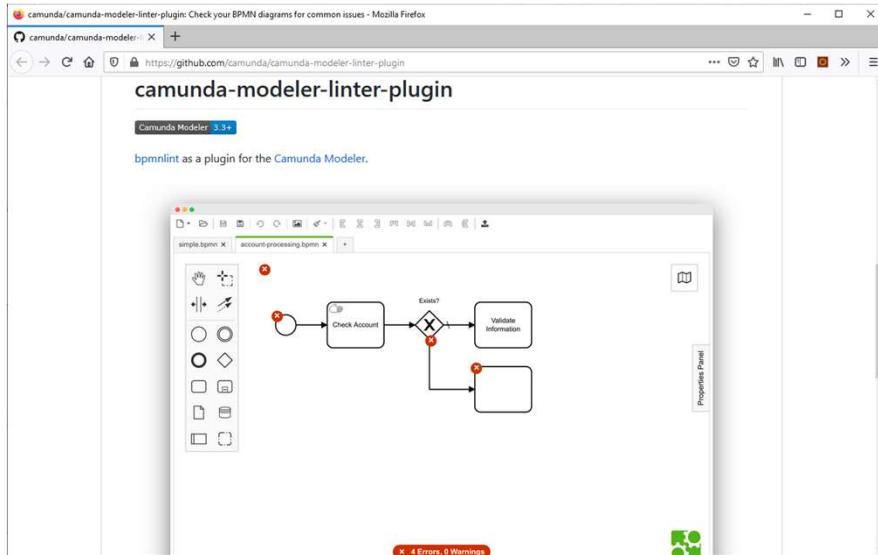
CAMUNDA

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



46

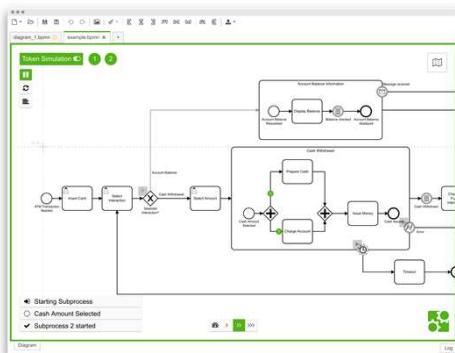
## Extend the Camunda Modeler



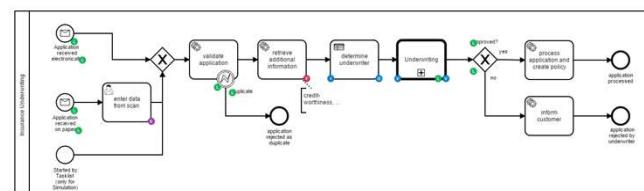
47

CAMUNDA

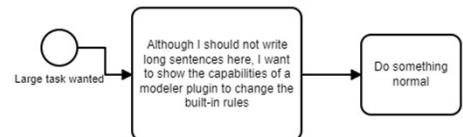
## Camunda Modeler Plugin Examples



Token Simulation Plugin



Property Info Plugin



Resize Tasks Plugin

<https://github.com/camunda/camunda-modeler-plugins>

48

CAMUNDA

## cawemo.com

The screenshot shows the homepage of cawemo.com. At the top, there is a navigation bar with links for "Sign up free" and "Login". Below the navigation, the title "Business Process Modeling" is displayed. To the left of the main content area, there is a list of features:

- BPMN 2.0 standard diagram modeling
- Web-based
- Collaborative
- Integrated
- Workflow-automation-ready

Below this list is a "Sign up free" button. To the right of the list is a BPMN diagram illustrating a process flow:

```
graph LR; Start(( )) --> Send[Send a message]; Send --> Wait{Wait for a response or some event}; Wait --> Decision{Result?}; Decision -- Timed out --> HasNotCalled((Has not called anything)); Decision -- Otherwise --> Call[Call a system or service]; Call --> HasCalled((Has called a system or service));
```

The diagram consists of several nodes: a start node, a "Send a message" task, a "Wait for a response or some event" gateway, a decision diamond labeled "Result?", a timer node labeled "Timed out", a "Call a system or service" task, and two end nodes labeled "Has called a system or service" and "Has not called anything".

49

CAMUNDA

## Camunda Tasklist

The screenshot shows the Camunda Tasklist interface. On the left, there is a sidebar with a tree view of tasks:

- My Tasks
- My Group Tasks (4)
  - Accounting
  - John's Tasks
  - Mary's Tasks
  - Peter's Tasks
- All Tasks

The main area displays a task titled "Approve Invoice". The task details are as follows:

**Invoice Receipt**  
Due in 7 days, Created 7 minutes ago  
Invoice Amount: 50  
Invoice Number: B05-43934

**Creditor**  
Great Pizza for Everyone Inc.

**Amount**  
30

**Invoice Category**  
Travel Expenses

**Invoice Number**  
GPFE-23232323

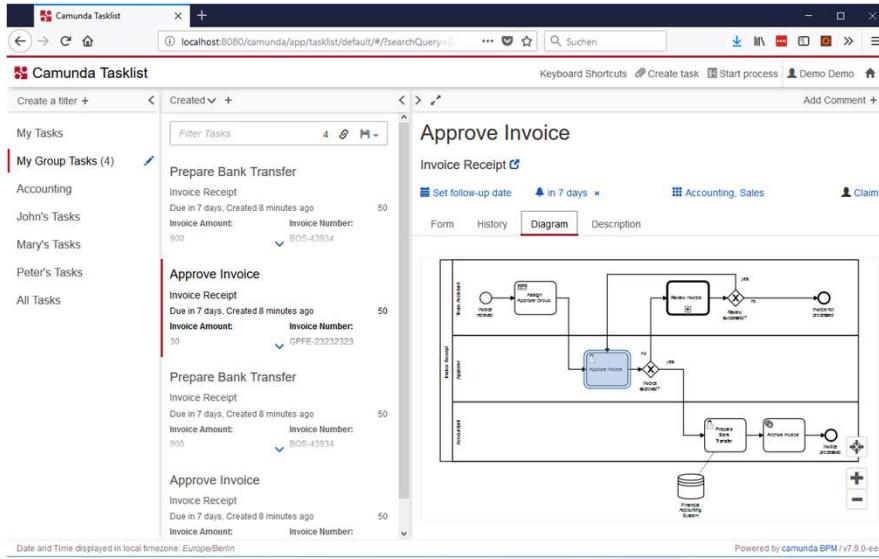
**Do you approve?**

Buttons at the bottom include "Save" and "Complete".

50

CAMUNDA

## Process Overview



51

CAMUNDA

## <https://docs.camunda.org>

The screenshot shows the Camunda BPM documentation website at <https://docs.camunda.org/manual/latest/>. The top navigation bar includes links for 'Get Started', 'BPM Platform', 'Optimize', 'Enterprise', and 'Camunda.org'. A search bar is also present. The main content area features a section titled 'The Camunda BPM Manual' with a brief introduction to the platform. Below this is a 'Getting Started' section with a guide to 'GET STARTED WITH BPMN 2.0'. It lists various BPMN components and their equivalents: BPMN 2.0, CMN 1.1, DMN 1.1; Spring Framework, Java EE 7, Apache Maven. A table provides links to other useful topics like 'Introduction', 'Guides', 'References', 'Overview', 'Installation', 'User Guide', 'REST API', etc. At the bottom, there's a footer with links to 'camunda.org' and 'docs.camunda.org', a privacy statement, and a 'Back to top' button.

52

CAMUNDA

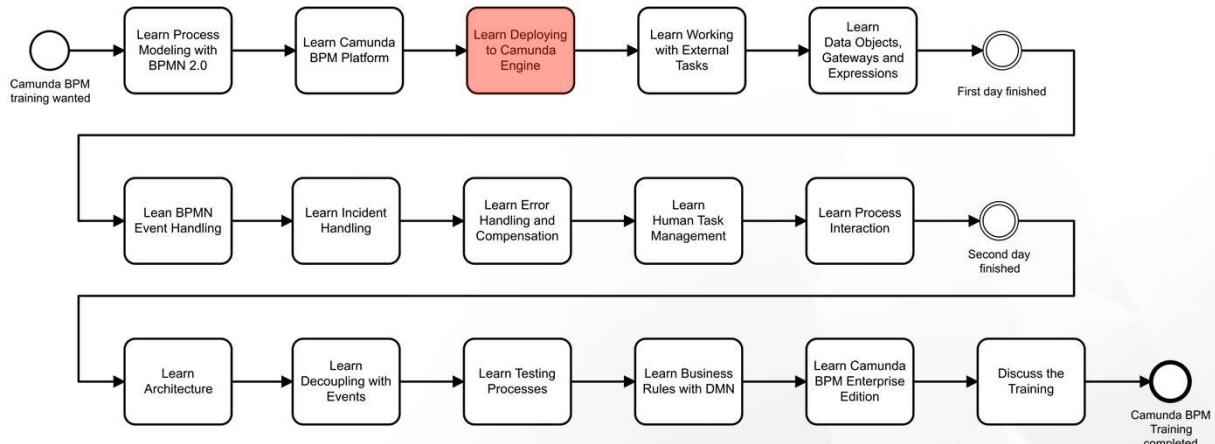
# Camunda Best Practices

The screenshot shows the Camunda Best Practices website. At the top, there's a navigation bar with links for "Search", "For Stakeholders" (which is underlined), "By Alphabet", "By Keywords", and "Customer Success Path". Below the navigation, a section titled "Camunda Best Practices For Stakeholders" lists categories: "Architecture" (Deciding About Your Stack, Deciding About Your Tasklist, Reporting About Processes, Sizing Your Environment, Using a Greenfield Stack) and "Development" (Building a Custom Tasklist/Application, Choosing the DMN Hit Policy, Dealing With Problems and Exceptions, Enhancing Tasklists with Business Data, Handling Data in Processes). A note at the bottom says "Internally a Third Party Toolkit". The footer includes copyright information: "Displayed January 30, 2019. Applies to Camunda 7.10. Any feedback? [best-practices@camunda.com](mailto:best-practices@camunda.com)" and "© Camunda Services GmbH 2019, All Rights Reserved".

53

<https://camunda.com/best-practices/>

CAMUNDA



54

CAMUNDA

## Name the Process

The screenshot shows the Camunda Modeler interface. On the left is the BPMN diagram for a payment process. The process starts with a start event labeled "Payment requested". It flows to a task "Deduct existing credit", then a decision diamond "Credit sufficient?". If "no", it goes to a task "Charge credit card" and then ends at a final event "Payment completed". If "yes", it ends at the final event "Payment completed". A red arrow points from the "Properties Panel" to the "Name" field in the "General" tab of the configuration panel.

Diagram:

```
graph LR; Start(( )) --> Deduct[Deduct existing credit]; Deduct --> Decision{Credit sufficient?}; Decision -- no --> Charge[Charge credit card]; Charge --> End1((( )); Decision -- yes --> End2((( ));
```

Properties Panel (PaymentProcess General):

- General**
- Id**: PaymentProcess
- Name**: Payment
- Version Tag**: (empty)
- Executable

Configuration Panel (PaymentProcess General):

- General**
- Id**: PaymentProcess
- Name**: Payment
- Version Tag**: (empty)
- Executable

CAMUNDA

## Make Process Executable

The screenshot shows the Camunda Modeler interface. On the left is the BPMN diagram for a payment process. The process starts with a start event labeled "Payment requested". It flows to a task "Deduct existing credit", then a decision diamond "Credit sufficient?". If "no", it goes to a task "Charge credit card" and then ends at a final event "Payment completed". If "yes", it ends at the final event "Payment completed". A red arrow points from the "Properties Panel" to the "Executable" checkbox in the "General" tab of the configuration panel.

Diagram:

```
graph LR; Start(( )) --> Deduct[Deduct existing credit]; Deduct --> Decision{Credit sufficient?}; Decision -- no --> Charge[Charge credit card]; Charge --> End1((( )); Decision -- yes --> End2((( ));
```

Properties Panel (PaymentProcess General):

- General**
- Id**: PaymentProcess
- Name**: Payment
- Version Tag**: (empty)
- Executable

Configuration Panel (PaymentProcess General):

- General**
- Id**: PaymentProcess
- Name**: Payment
- Version Tag**: (empty)
- Executable

CAMUNDA

## Conditions

The screenshot shows the Camunda Modeler interface with a BPMN diagram titled "payment-business-description.bpmn". The diagram includes a start event, a "Charge existing credit" task, a decision diamond labeled "Credit sufficient?", a "Charge credit card" task, and an end event labeled "Payment completed". A sequence flow connects the first task to the decision diamond. From the decision diamond, two paths emerge: one labeled "yes" leading to the second task, and one labeled "no" leading to an exclusive gateway. A red arrow points from the "no" path to the properties panel of the sequence flow, which is titled "SequenceFlow\_1ed60tx". The properties panel shows the "Condition Type" as "Expression" and the "Expression" field containing "\${not creditSufficient}".

57

CAMUNDA

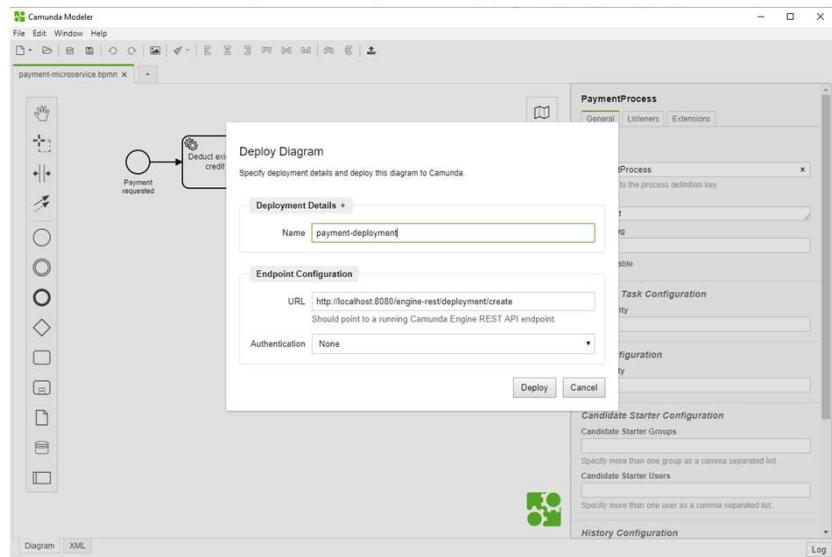
## Scripts

The screenshot shows the Camunda Modeler interface with a BPMN diagram titled "payment-technical.bpmn". The diagram includes a start event, a "Charge existing credit" task, a decision diamond labeled "Sufficient?", a "Charge credit card" task, and an end event labeled "Payment completed". A sequence flow connects the first task to the decision diamond. From the decision diamond, two paths emerge: one labeled "yes" leading to the second task, and one labeled "no" leading to an inclusive gateway. A red arrow points from the "no" path to the properties panel of a task titled "Task\_0g2r169". The properties panel shows the "Script Format" as "javascript", "Script Type" as "Inline Script", and the "Script" field containing `print("charge existing credit...");`. The "Result Variable" field is empty.

58

CAMUNDA

## Deploy From Modeler



59

CAMUNDA

## Deploy the Process

The screenshot shows the Camunda Cockpit interface with a deployment list. One entry is selected, showing deployment details: "Deployment Time" (twitter-qa), "Deployment Name" (camunda-deployment), and "Process" (tweet\_approval\_process.bpmn). Below this, a "Deploy" button is visible. To the right, the Postman application is open, showing a POST request to "http://localhost:8080/engine-rest/deployment/create". The "Body" tab is selected, showing a JSON payload with fields: "deployment-name": "postman", "deployment-source": "camunda-deployment", and "process": "tweet\_approval\_process.bpmn". The "JSON" tab shows the raw JSON code. The response from Postman indicates a successful deployment: "Status: 200 OK", "Time: 809 ms", and "Size: 104 B".

```
curl -X POST \
  http://localhost:8080/engine-rest/deployment/create \
  -F deployment-name=camunda-deployment \
  -F deployment-source=curl \
  -F 'process=@C:\Users\Ingo Richtsmeier\Documents\ca054\tweet_approval_process.bpmn'
```

60

CAMUNDA

## Include Additional Resources

- Include: forms, subprocesses, DMN diagrams, script files
- Reference resources in the BPMN diagram with deployment://

The left side shows the Camunda BPMN deployment interface. It displays a list of files being deployed, including `invoice.v2.bpmn`, `invoiceBusinessDecisions.dmn`, `approve-invoice.html`, `assign-reviewer.html`, `prepare-bank-transfer.html`, `review-invoice.html`, and `start-form.html`. A "Deploy" button is at the bottom.

The right side shows a Postman API request for deployment. The URL is `http://localhost:8080/engine-rest/deployment/create`. The "Body" tab shows a JSON payload:

```
[{"id": "1", "deploymentName": "postman-example", "deploymentSource": "rest-client", "processId": "process-1", "dmnId": "dmn-1", "startForm": "start-form.html", "approveForm": "approve-invoice.html"}]
```

Below the table, the JSON payload is shown again:

```
{"id": "1", "deploymentName": "postman-example", "deploymentSource": "rest-client", "processId": "process-1", "dmnId": "dmn-1", "startForm": "start-form.html", "approveForm": "approve-invoice.html"}
```

61

CAMUNDA

## Camunda REST API

your custom  
tasklist



your mobile  
app



REST  
API

process engine

Tasklist



Cockpit

CAMUNDA

62

## Documentation of REST API

The screenshot shows a browser window displaying the Camunda REST API documentation. 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 the following structure:

- Manual: latest (7.6)
  - Introduction
  - User Guide
  - Reference
    - Rest Api
      - Overview
      - Authorization
      - Batch
      - Case Definition
      - Case Execution
      - Case Instance
      - Decision Definition
      - Decision Requirements Definition
      - Deployment
      - Engine
      - Execution
      - External Task
      - Filter
      - Group
      - History
        - Activity Instance
        - Batch
  - OPTIONS

The main content area contains the following sections:

- Method**: GET /history/process-instance/count
- Parameters**
- Query Parameters**

Name	Description
processInstanceId	Filter by process instance id.
processInstanceIds	Filter by process instance ids. Must be a comma-separated list of process instance

A red box highlights the "Method" section, and a callout bubble points to it with the text "Most important for beginners".  
A sidebar on the right lists "ON THIS PAGE:" with links to Method, Parameters, Result, Response Codes, and Example.  
A callout bubble from the sidebar points to the "Method" section with the same text "Most important for beginners".  
A button in the sidebar says "Get up to 24/7 support with the Camunda Enterprise Edition".  
The footer includes the Camunda logo and copyright information: camunda.org and docs.camunda.org are part of camunda BPM | Built by camunda and contributors — Privacy Statement — camunda Services GmbH © 2015.

## Start Process Instances

Use any REST client

Base URL: <http://localhost:8080/engine-rest>

POST /process-definition/key/{key}/start

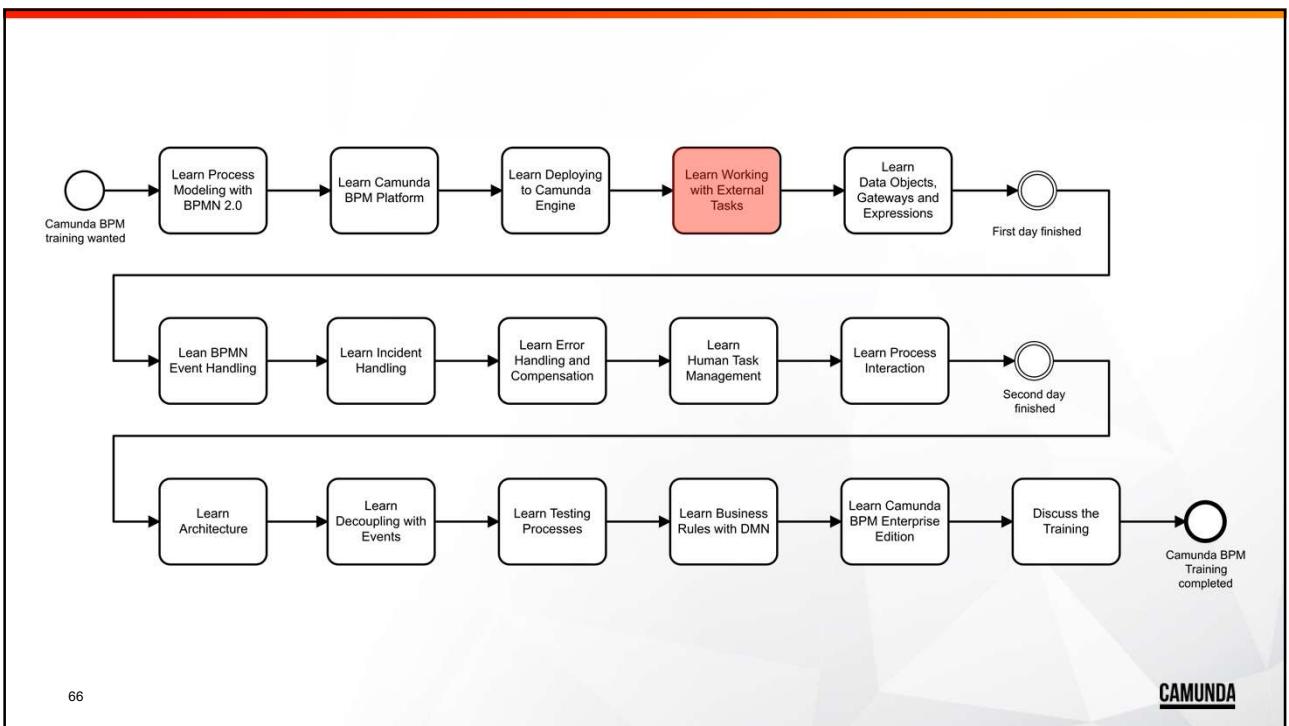
Payload: {"variables": {"creditSufficient": {"value": "true"}}}

<https://docs.camunda.org/manual/latest/reference/rest/process-definition/post-start-process-instance/>

CAMUNDA

**Exercise 2**

<https://training.camunda.com/microservices>  
 user: microservices  
 password: camundarocks



## 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	

67

CAMUNDA

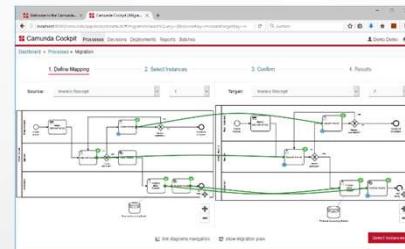
## Start New Process Instances

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

POST /process-definition/key/VacationRequestProcess/start {}

If needed, process instance migration is possible



68

CAMUNDA

## Key vs. ID vs. Name

```
POST /process-definition/key/VacationRequestProcess/start
```

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

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

69

The screenshot shows the 'General' tab of a participant configuration screen. The participant is named 'Participant\_041airq'. The 'Name' field contains 'Vacation request process'. The 'Process Id' field contains 'VacationRequestProcess'. The 'Process Name' field contains 'Vacation request'. A checked checkbox labeled 'Executable' is present.

## Process Variables

Are persisted with the process instance

### Client Code:

```
POST /process-definition/key/aProcessDefinitionKey/start
```

```
{variables:  
    {"aVariableName": {"value": "a variable value", "type": "String"},  
     "bVariableName": {"value": true, "type": "Boolean"}  
    }  
}
```

More details later

70

CAMUNDA

## Business Key

- Domain specific identifier of a process instance
- Optional
- Set at the start
- Set or change in an Execution Listener
- Displayed prominently in Cockpit and Tasklist
- Query for process instances

Candidates for  
the Business Key  
in the Payment

POST /process-definition/key/aProcessDefinitionKey/start

{"businessKey": "my business key"}

Start process

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

Business Key			
Add a varia... +	Name	Type	Value
Rem... x	content	String	My first tweet from Twitter QA

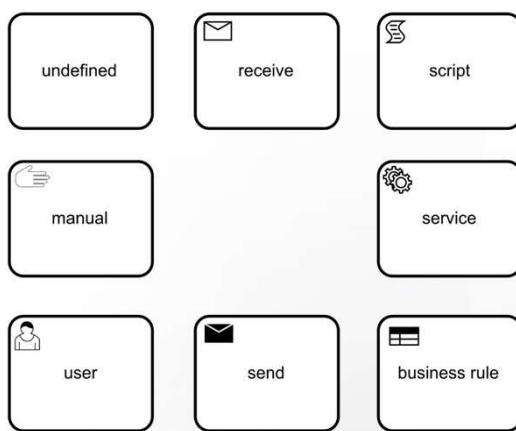
Back

Close Start

CAMUNDA

71

## BPMN 2.0 Task Types



72

CAMUNDA

## Service Task With External Worker

The diagram illustrates a BPMN process and its configuration in Camunda.

**BPMN Process:**

```
graph LR; Start(( )) --> BookVacation[Book vacation]; BookVacation --> SendNotification[Send notification to colleagues]; SendNotification --> VacationApproved((Vacation approved));
```

**Service Task Configuration:**

**Camunda BPMN:**

- General:** Id: sendRejectionEmailServiceTask, Name: Send rejection email, Element Template: [dropdown]
- Details:** Implementation: External, Topic: rejection-mail
- External Task Configuration:** Task Priority: [input field]
- Asynchronous Continuations:** Asynchronous Before, Asynchronous After

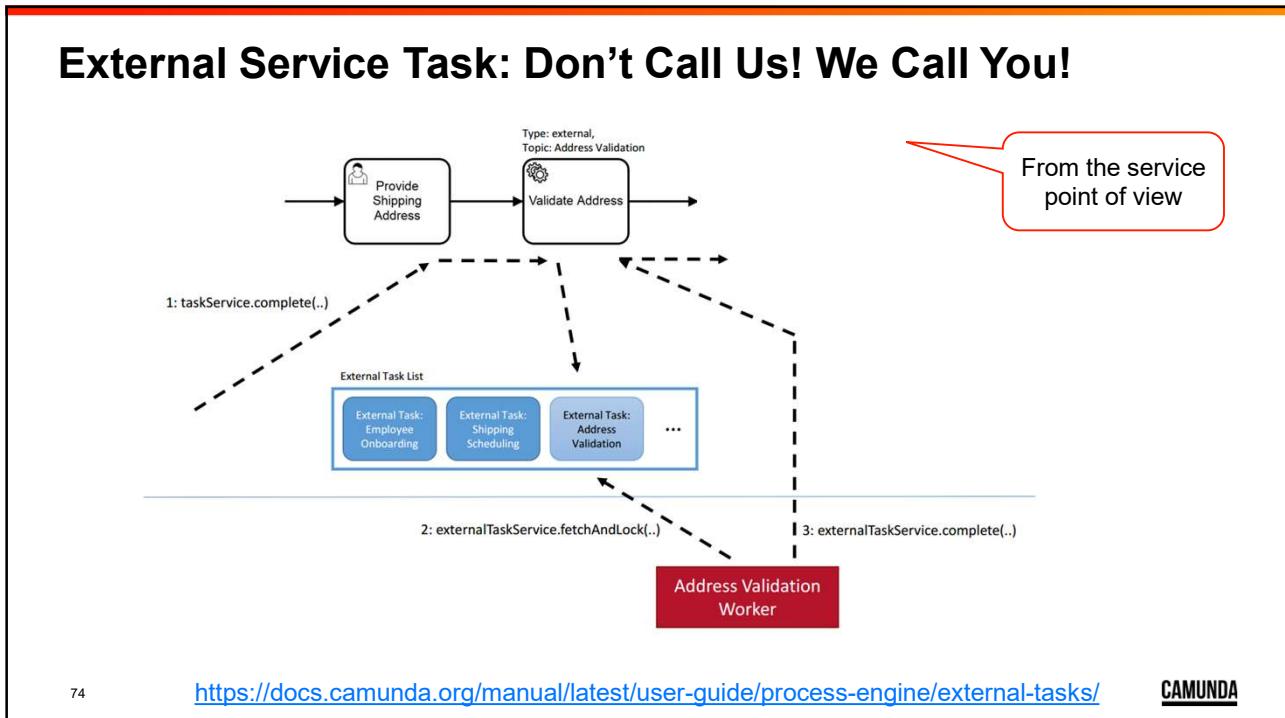
**Camunda Engine:**

- General:** Id: sendRejectionEmailServiceTask, Name: Send rejection email, Element Template: [dropdown]
- Details:** Implementation: External, Topic: rejection-mail
- External Task Configuration:** Task Priority: [input field]

73

CAMUNDA

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



## Naming Convention for Topics

- Name it from the workers point of view:
  - tweet-rejection
  - tweet publication
  - rejection-publication
  - rejection-mailing
  - ...
- What can I do as a worker?

75

CAMUNDA

## Exercise 3

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks



## Worker

Two rest calls and some code inbetween

```
1. POST /external-task/fetchAndLock
{ "workerId": "mailer",
  "maxTasks": 2,
  "topics": [
    {"topicName": "rejection-mail",
     "lockDuration": 60000}
  ]
}
```

Reponse:

```
{ ...
  "id": "anExternalTaskId",
  ...
}
```

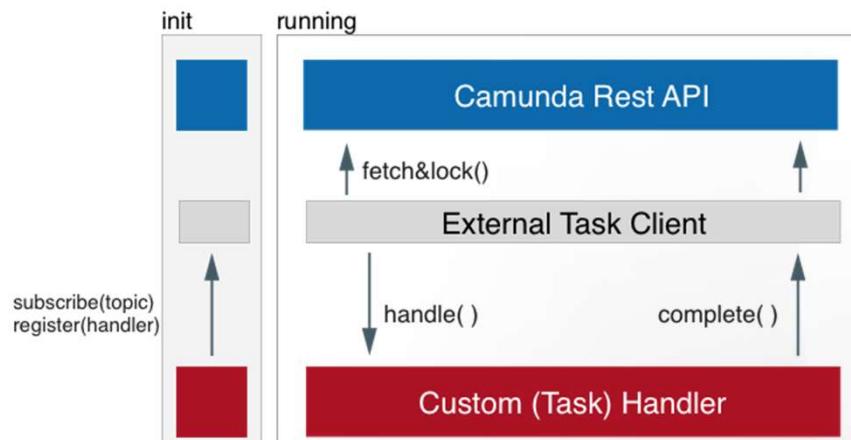
2. Do what you  
need to do

```
3. POST /external-task/anExternalTaskId/complete
{
  "workerId": "mailer"
}
```

77

CAMUNDA

## External Task Client



78

CAMUNDA

## External Task Client in JavaScript

The screenshot shows a browser window with the URL [https://github.com/camunda-external-task-client-js](https://github.com/camunda/camunda-external-task-client-js). The page title is "camunda-external-task-client". It includes a green "build passing" badge. Below it, there's a section titled "Installing" with two command-line examples: "npm install -s camunda-external-task-client-js" and "yarn add camunda-external-task-client-js". Under "Usage", there are four steps: 1. Make sure to have Camunda running. 2. Create a simple process model with an External Service Task and define the topic as 'topicName'. 3. Deploy the process to the Camunda BPM engine. 4. In your NodeJS script: followed by a code snippet.

```
const { Client, logger } = require("camunda-external-task-client-js");

// configuration for the Client:
// - 'baseUrl': url to the Workflow Engine
// - 'logger': utility to automatically log important events
const config = { baseUrl: "http://localhost:8080/engine-rest",
  use: logger,
  maxTasks: 1};

// create a Client instance with custom configuration
const client = new Client(config);

// susbscribe to the topic: 'creditScoreChecker'
client.subscribe("credit-charge", async function({ task, taskService }) {
  console.log("checking credit score");
  // complete the task
  await taskService.complete(task);
});
```

<https://github.com/camunda/camunda-external-task-client-js>

CAMUNDA

## Run External Task Client in JavaScript

- Install: `npm install -s camunda-external-task-client-js`
  - Code:

```
const { Client, logger } = require("camunda-external-task-client-js");

// configuration for the Client:
// - 'baseUrl': url to the Workflow Engine
// - 'logger': utility to automatically log important events
const config = { baseUrl: "http://localhost:8080/engine-rest",
  use: logger,
  maxTasks: 1};

// create a Client instance with custom configuration
const client = new Client(config);

// susbscribe to the topic: 'creditScoreChecker'
client.subscribe("credit-charge", async function({ task, taskService }) {
  console.log("checking credit score");
  // complete the task
  await taskService.complete(task);
});
```
- Run: `node my-client.js`

80

CAMUNDA

## Camunda Client in C#

Install:

```
dotnet new console -o myExternalTaskClient
cd myExternalTaskClient
dotnet add package BerndRuecker.Sample.CamundaClient --version 0.1.1
dotnet add package Newtonsoft.Json --version 12.0.0.0
```

Code Program.cs:

```
using System;
using CamundaClient;

namespace myExternalTaskClient
{
    class Program
    {
        static void Main(string[] args)
        {
            CamundaEngineClient camunda = new CamundaEngineClient();
            camunda.Startup(); // Deploys all models and Start all found ExternalTask-Workers
            Console.ReadLine(); // wait for ANY KEY
            camundaShutdown();
        }
    }
}
```

81

CAMUNDA

## External Task Worker in C#

Code Worker.cs:

```
using System;
using System.Collections.Generic;
using CamundaClient.Dto;
using CamundaClient.Worker;

namespace myExternalTaskWorker
{
    [ExternalTaskTopic("credit-charge")]
    class ExampleWorker : IExternalTaskAdapter
    {

        public void Execute(ExternalTask externalTask, ref Dictionary<string, object> resultVariables)
        {
            Console.WriteLine($"checkin credit score");
        }
    }
}
```

Run: dotnet run

82

CAMUNDA

## External Task Client in Python

Run: pip install requests

PythonWorker.py:

```
import requests
import json
import time

url = 'http://localhost:8080/engine-rest/external-task/fetchAndLock'

fetchAndLockPayload = {"workerId": "myExampleWorker",
    "maxTasks": 1,
    "usePriority": "true",
    "topics": [
        {"topicName": "credit-charge",
            "lockDuration": 30000
        }
    ]
}

...
https://github.com/camunda-consulting/code/blob/master/snippets/python-external-worker/PythonWorker.py
```

CAMUNDA

## External Task Client in Python

```
try:
    while True:

        # Fetch and Lock
        try:
            res = requests.post(url, json=fetchAndLockPayload)
            print('Fetch and lock status code: ', res.status_code)

            body = res.text
            print('Fetch and lock response: ', body)

        except:
            print('Engine is down')

        while body == '[]':
            time.sleep(5)
            res = requests.post(url, json=fetchAndLockPayload)
            body = res.text
            print ('Fetch and lock response: ', body)

            if body !='[]':
                break

        # Put your Business Logic here
        print('Deduct existing credit')

    ...
https://github.com/camunda-consulting/code/blob/master/snippets/python-external-worker/PythonWorker.py
```

```
#Prepare url with task id and the response body
response = json.loads(body)
taskId = response[0]['id']
taskId = str(taskId)

complete_url = ('http://localhost:8080/'
    'engine-rest/external-task/'
    + taskId + '/complete')

response = {
    "workerId": "myExampleWorker",
}

# API call (Complete)
try:
    complete = requests.post(complete_url, json=response)
    print('complete status code: ', complete.status_code)

except:
    print('failed')

except KeyboardInterrupt:
    pass
```

Run: python PythonWorker.py

CAMUNDA

# Awesome Camunda External Task Clients

Here is a list with all the code Snippets in different languages that helps you to create an external task client for Camunda.

## Supported Code by Camunda

- JavaScript
- Java

## Unsupported

- JavaScript
- .Net
- Spring Boot
- Python
- Ruby and a lot of more! amazing

It is just a beginning! If you already have something, create a pull request with a link to your code.

85

<https://github.com/camunda/awesome-camunda-external-clients>

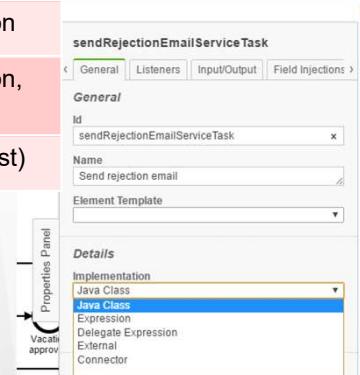
CAMUNDA

# Other Options for Service Tasks

Implementation	Description
Java Class	Implement the service in Java
Expression	Resolve the implementation from an expression
Delegate Expression	Resolve the implementation from an expression, that implements JavaDelegate
Connector	Configure the call to a web service (soap or rest)

More details come later

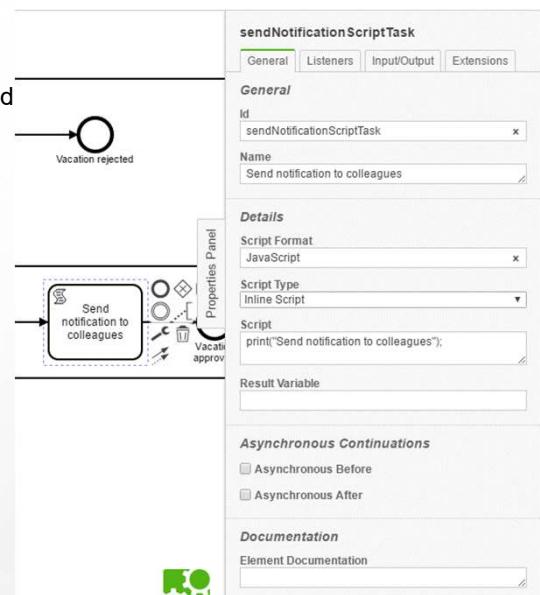
86



CAMUNDA

## Script Task

- Every JSR-223 compliant scripting engine can be used
- Script sources can be externalized
- Scripts get compiled and cached
- Available variables in all script languages:
  - all process variables
  - execution
  - task



87

CAMUNDA

## Script Task Examples

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

Special key word

Process variables

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

Load script from deployment

Store script result

88

CAMUNDA

## Business Rule Task

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



89

CAMUNDA

## Execution Listeners

The screenshot shows a Camunda BPM process diagram and its configuration interface. On the left, there is a process flow starting with a 'Book vacation' task, followed by a 'Send notification to colleagues' task, which ends with a 'Vacation approved' event. This is connected to another parallel gateway, which then leads to a 'Send rejection email' task and ends with a 'Vacation rejected' event. The 'Send rejection email' task has an outgoing arrow pointing to a 'Delete' icon.

The central part of the interface is the 'Properties Panel'. It displays two execution listeners:

- sendRejectionEmailServiceTask**:
  - Listeners**: An 'Execution Listener' named 'start : Script' is selected. The 'Listener Type' is 'Script' and the 'Event Type' is 'start'. The 'Script Format' is 'javascript' and the 'Script Type' is 'Inline Script'. The script content is:

```
var str = execution.getVariable("someText");
var res = str.toUpperCase();
execution.setVariable("someText", res);
```
  - Execution Listener**: Another 'Execution Listener' for the same task, with the same configuration as the first one.

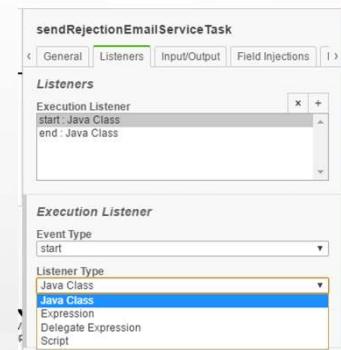
90

CAMUNDA

## Events for Execution Listeners

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

Listener Type
Script
Expression
Delegate Expression
Java Class



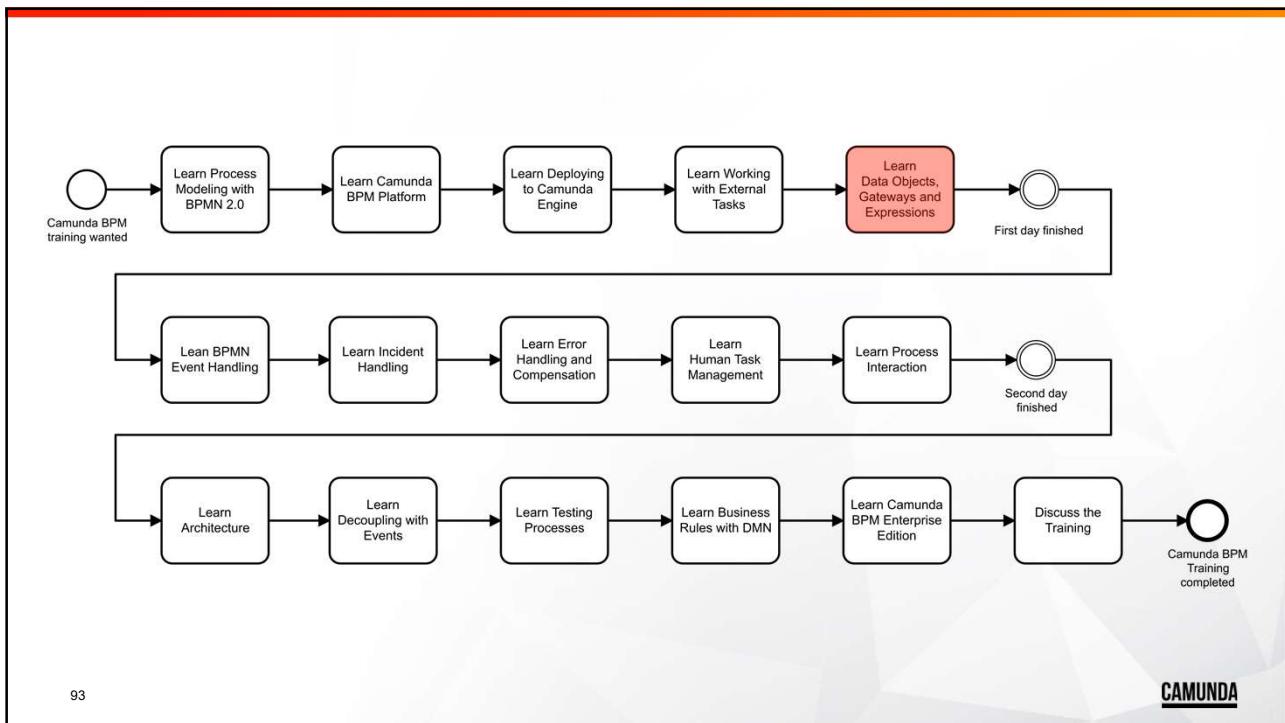
CAMUNDA

91

## Exercise 4

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks





## Set Process Variables

- When starting a process instance:

```

POST /process-definition/key/aProcessDefinitionKey/start
{
  "variables": {
    "aVariable" : {
      "value" : "aStringValue",
      "type": "String"
    },
    "anotherVariable" : {
      "value" : true,
      "type": "Boolean"
    }
  }
}

```

- On running process instances:

```

PUT /process-instance/aProcessInstanceId/variables/aVarName
{"value" : "someValue", "type": "String"}

```

## Accessing Process Variables

Retrieve variables with external-task/fetchAndLock

```
{  
    "workerId": "emailer",  
    "maxTasks": 2,  
    "usePriority": true,  
    "topics":  
        [{"topicName": "rejection-mail",  
         "lockDuration": 10000,  
         "variables": ["employee"]}  
     ]  
}
```

request

```
[{  
    ...  
    "id": "anExternalTaskId",  
    ...  
    "variables": {  
        "employee": {  
            "type": "String",  
            "value": "Jim",  
            "valueInfo": {}  
        }  
    }  
}]
```

response

Set variables with external-task/anExternalTaskId/complete

```
{  
    "workerId": "emailer",  
    "variables":  
        {"mailText": {"value": "aStringValue"},  
         "sendAt": {"value": "2018-02-20T15:42:36"}  
}
```

or at process instance start

CAMUNDA

## Process Variables

- Are not modeled in BPMN 2.0
- Behave like a 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>



vacation request

## Access Variables in External Task Client

- In the handler (Node.js):

```
client.subscribe("credit-charge", async function({ task, taskService }) {  
    const amount = task.variables.get("amount");  
    const resultVars = new Variables();  
  
    resultVars.set("creditSufficient", false);  
    await taskService.complete(task, resultVars);  
})
```

97

CAMUNDA

## Object Variables

Examples from the external task client for Java Script

Saving values:

```
variables.set("fullName", { first: "John", last: "Doe" });  
  
variables.setTyped("test", {  
    value: "<test id=2 />",  
    type: "XML",  
    valueInfo: {}  
});
```

Serialize  
explicitly  
(except JSON)

Reading values:

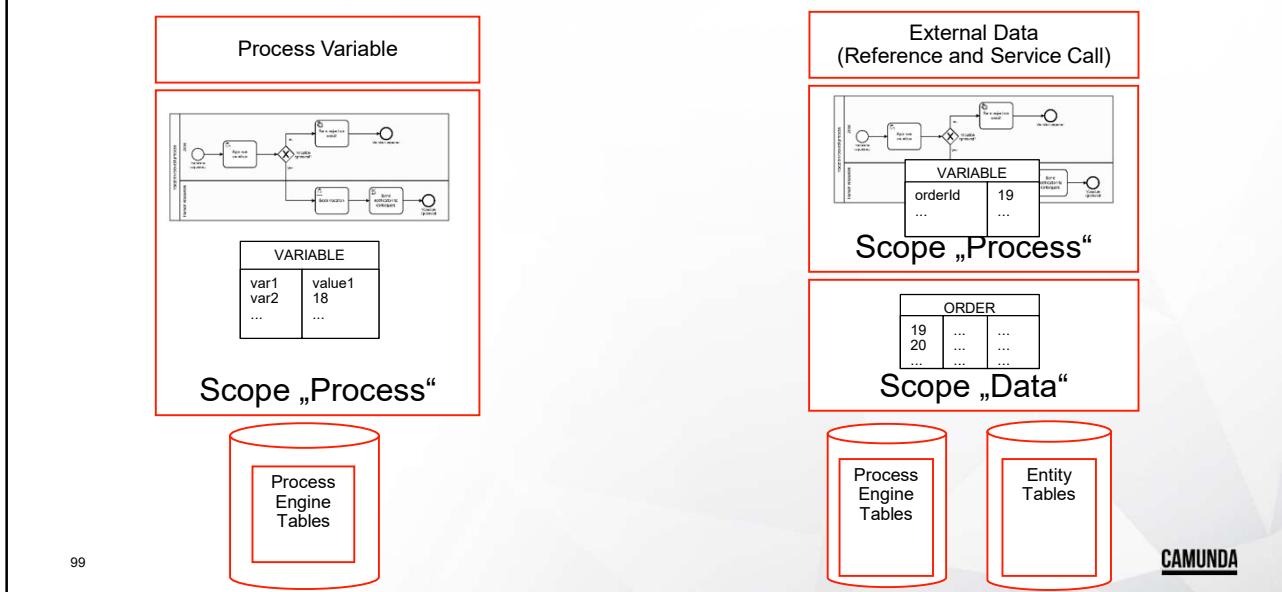
```
variables.get("fullName");   Output: { first: "John", last: "Doe" }  
variables.get("test");       Output: <test id=2 />
```

Deserialize  
transparently

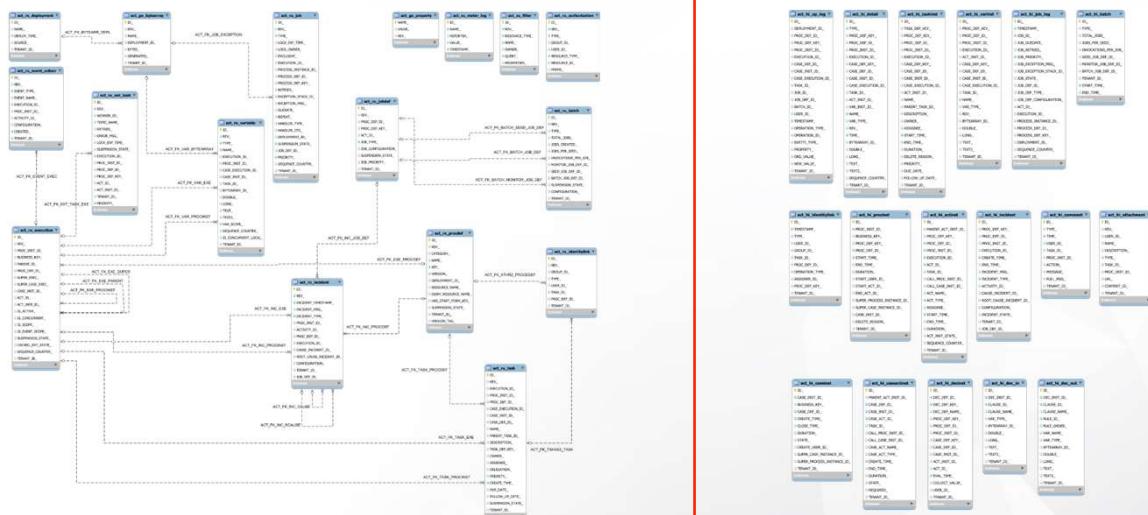
98

CAMUNDA

# Storage



## Database Schema



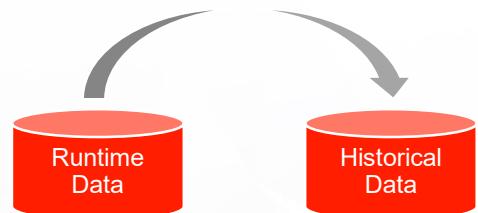
<https://docs.camunda.org/manual/latest/user-guide/process-engine/database/#database-schema>

100

CAMUNDA

## History Configuration

finished process instance are removed from runtime tables



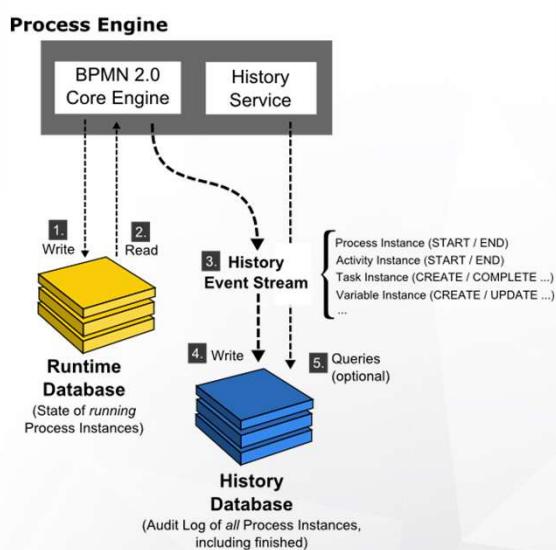
### Configuration options:

- 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

101

CAMUNDA

## Asynchronous History



102

CAMUNDA

## History Cleanup

- historyTimeToLive for each process definition: Number of days
- defined by business needs
- either in the process model
- or by API-Call
- or in the Cockpit

The screenshot shows the Camunda Cockpit interface. On the left, the 'History Configuration' page displays the 'History Time To Live' setting at 180 days. On the right, the 'Cleanable Data' page lists process definitions with their versions, tenant IDs, and history time to live settings. A red callout box labeled 'Cleanup View Enterprise Feature' points to the 'Cleanup Now' button at the bottom of the page. A message at the bottom left indicates the next cleanup will run in 7 hours.

Process Definition	Version	Tenant ID	Finished	Cleanable	History Time To Live
Insurance Application	3		364	0	365 days
Document Request	2		84	0	365 days
Vacation request	1		2	0	180

- run periodically, configure by weekdays
- run manually

103

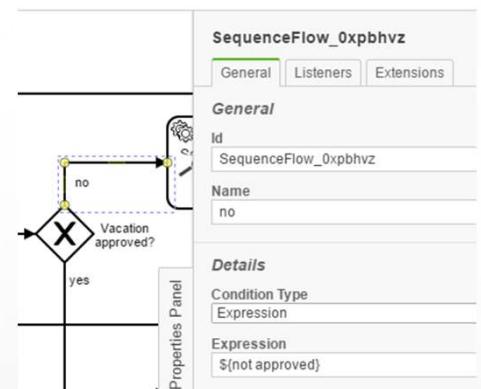
CAMUNDA

## 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)}`



104

CAMUNDA

## Unified Expression Language (UEL)

EL Expression	Result
<code> \${1 &gt; (4/2)}</code>	false
<code> \${4.0 &gt;= 3}</code>	true
<code> \${100.0 == 100}</code>	true
<code> \${((10*10) ne 100)}</code>	false
<code> \${'a' &lt; 'b'}</code>	true
<code> \${'hip' gt 'hit'}</code>	false
<code> \${4 &gt; 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.

EL Expression	Result
<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.
<code> #{customer.calcTotal}</code>	The return value of the method calcTotal of the customer bean.

from  
<http://java.sun.com/javaee/5/docs/tutorial/doc/bnagh.html#bna1n>

105

CAMUNDA

## Predefined Expressions

```
 ${currentUser()}
 ${currentUserGroups()}
```

```
from org.camunda.bpm.engine.impl.el.CommandContextFunctionMapper
```

```
 ${now()} -> java.util.Date
 ${dateTime()} -> org.joda.DateTime
```

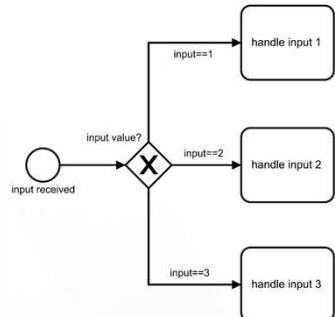
```
from org.camunda.bpm.engine.impl.el.DateTimeFunctionMapper
```

<https://docs.camunda.org/manual/latest/user-guide/process-engine/expression-language/#internal-context-functions>

106

CAMUNDA

## 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>
```

107

CAMUNDA

## Camunda Spin Project Example

```
{
  "name" : "jonny",
  "address" : {
    "street" : "12 High Street",
    "post code" : 1234
  }
}
```

Access in an expression:

```
 ${JSON(customer).prop("address").prop("post code").numberValue() == 1234}
```

Imagine a response from a web service call

Access JSON variable value in an expression:

```
 ${customer.jsonPath("$.address.post code").numberValue() == 1234}
```

Maybe a condition on a sequence flow

108

CAMUNDA

# 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/>

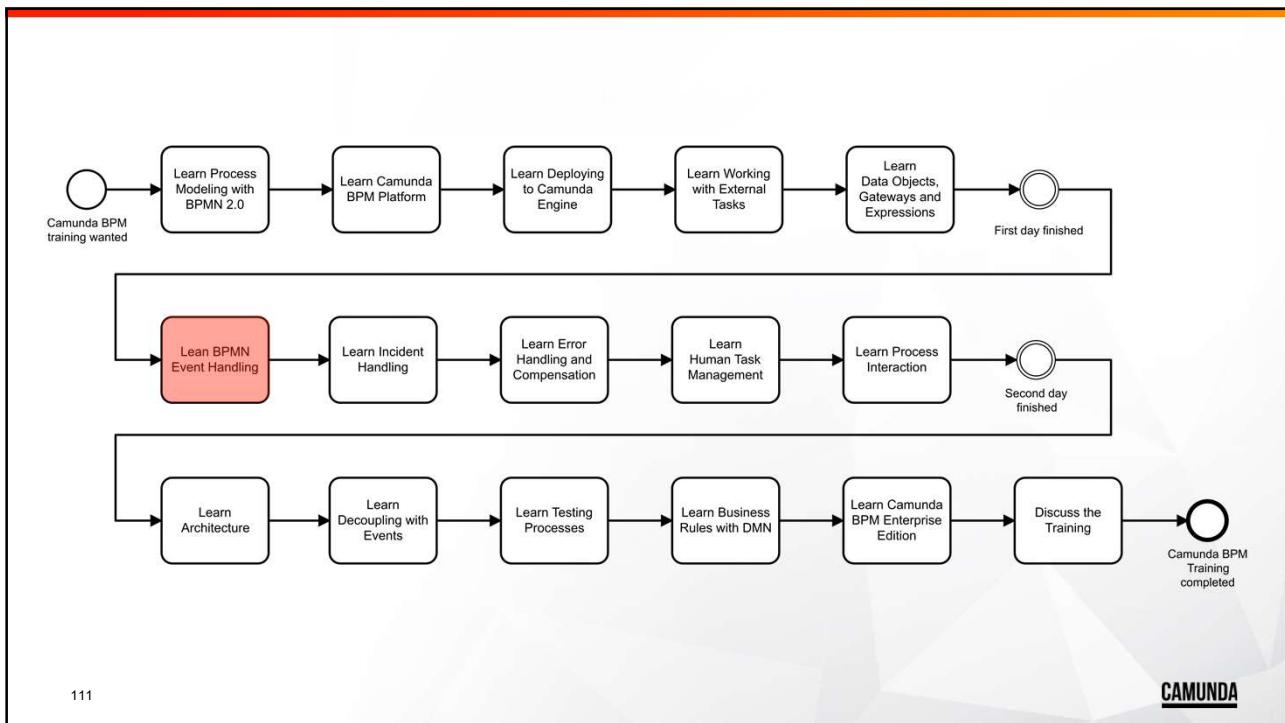
109

CAMUNDA

## Exercise 5

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks





## Event Coverage

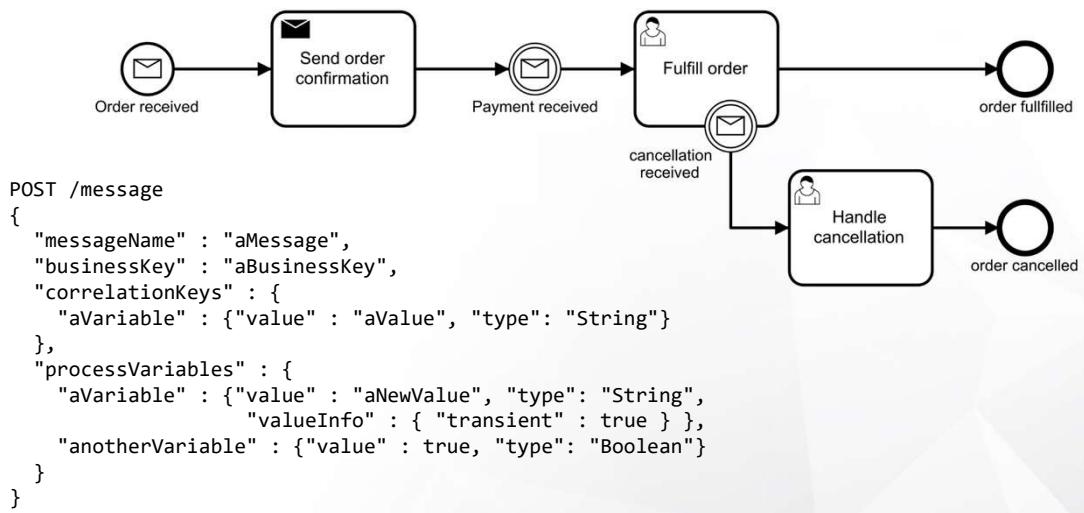
Type	Start			Intermediate			End	
	Normal	Event Subprocess	Event Subprocess non-interrupt	catch	boundary	boundary non-interrupt	throw	
None	○						○	○
Message	✉	✉	✉	✉	✉	✉	✉	✉
Timer	⌚	⌚	⌚	⌚	⌚	⌚		
Conditional	☰	☰	☰	☰	☰	☰		
Link				➡			➡	
Signal	△	△	△	△	△	△	△	△
Error	✗				✗		✗	✗
Escalation	Ⓐ	Ⓐ		Ⓐ	Ⓐ	Ⓐ	Ⓐ	Ⓐ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

See:  
[docs.camunda.org/manual/latest/reference/bpmn20/#events](https://docs.camunda.org/manual/latest/reference/bpmn20/#events)

112

CAMUNDA

## Message Events



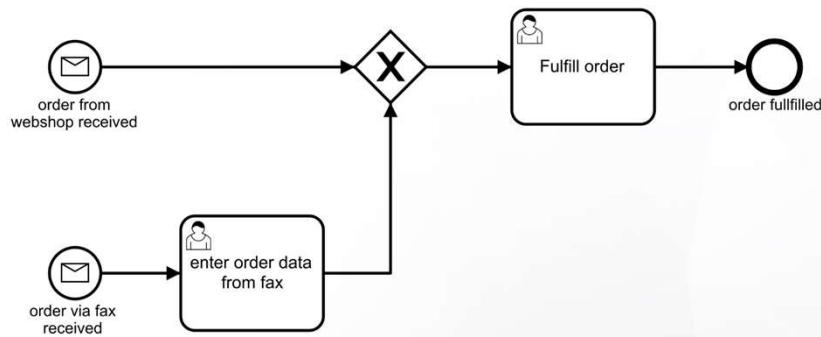
113

CAMUNDA

Demo



## Multiple Message Start Events Possible

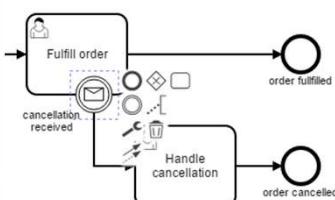


115

CAMUNDA

## Configuring Message Names

```
<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" />
```



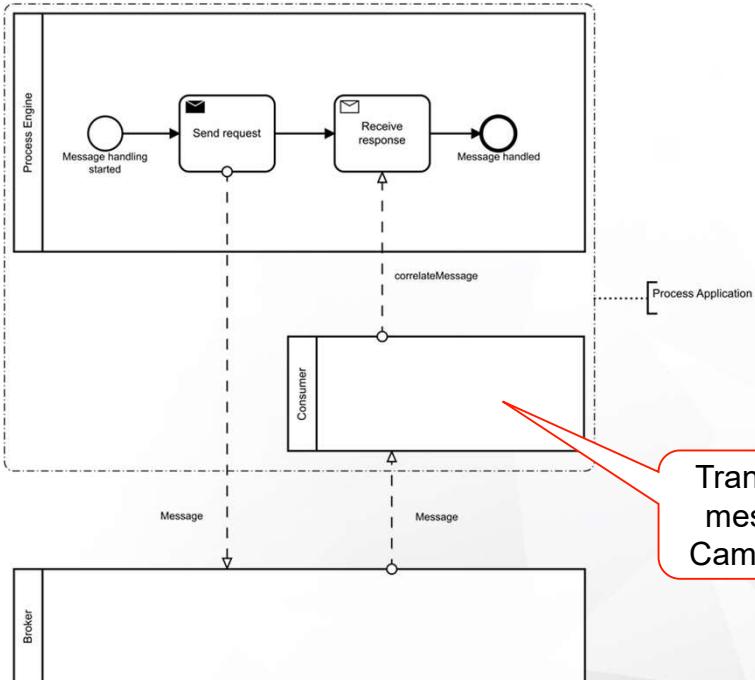
Properties Panel

BoundaryEvent_14qb0su	General	Listeners	Extensions
<b>General</b>			
<b>Id</b>	BoundaryEvent_14qb0su		
<b>Name</b>	cancellation received		
<b>Details</b>			
<b>Message</b>	orderCancelMessage (id=Message_04dfh6o)		
<b>Message Name</b>	orderCancelMessage		
<b>Asynchronous Continuations</b>	<input type="checkbox"/> Asynchronous Before <input type="checkbox"/> Asynchronous After		
<b>Documentation</b>	Element Documentation		

116

CAMUNDA

## Messaging



117

Translate the  
message to  
Camunda API

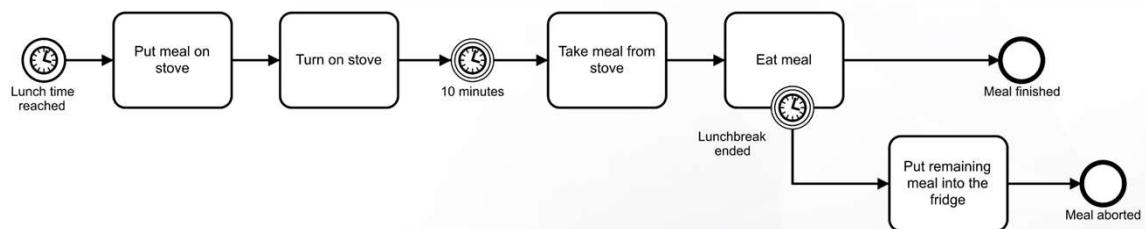
CAMUNDA

## Exercise 6

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks



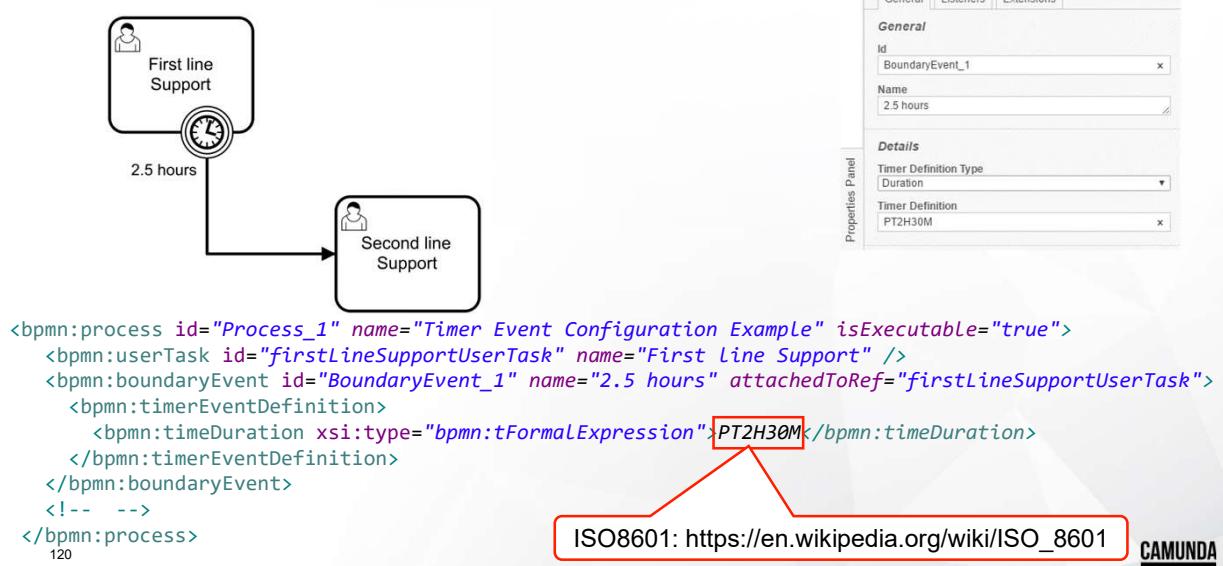
## Timer Events



119

CAMUNDA

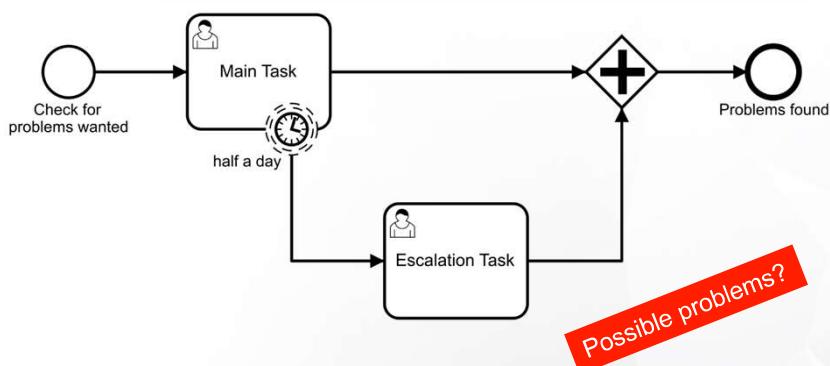
## Timer Event Configuration



120

CAMUNDA

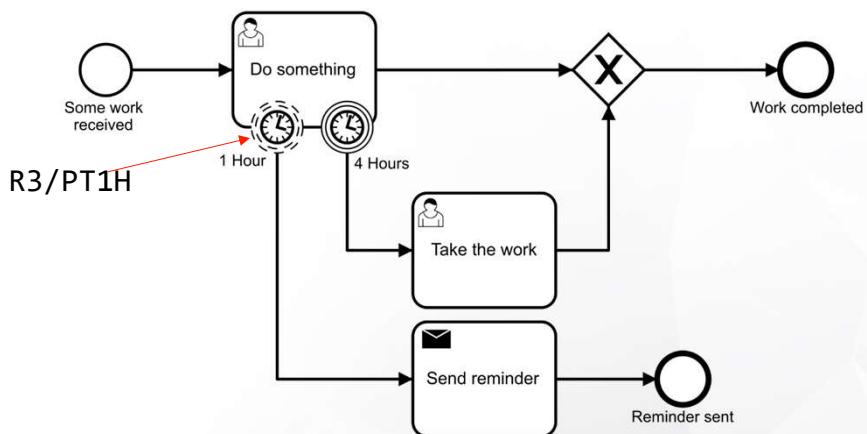
## Non Interrupting Timer Event



121

CAMUNDA

## Pattern for Two Stage Escalation



122

CAMUNDA

## Time-triggered Events

The screenshot shows a Camunda BPMN process model. A task named "Do something" is shown with an incoming arrow labeled "Some work received" and an outgoing arrow labeled "Work completed". The task has several outgoing events, including a timer event. The properties panel on the right shows the task configuration. A red diagonal banner with the text "To reduce the clutter in the process model" is overlaid on the process diagram.

Properties Panel

Task Listener

Task Listener

Event Type: timeout

Listener Id: automaticReassign

Listener Type: Java Class

Java Class: com.camunda.training.ReassignTask

Timer Definition Type: Duration

Timer Definition: PT2H30M

Field Injection

CAMUNDA

## Conditional Events

The screenshot shows a Camunda BPMN process model. An event "Order received" leads to a task "Check CAD model". From this task, an arrow goes to a conditional event "3D printer available?". From this event, an arrow goes to a task "Start printing". Below the process, two examples of conditional events are shown: one triggered by a timer ("two hours") and another triggered by an external event ("payment received").

Condition Type: Expression

Expression: \${availablePrinterNumber != null}

Trigger evaluation:

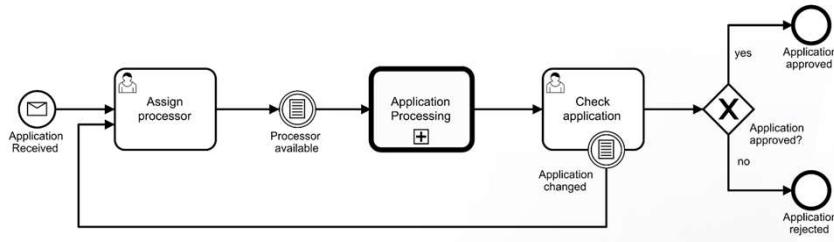
- via API
- in a Java Delegate (e.g. on a parallel branch)

Determined inside the process instance

Determined outside of the process instance

CAMUNDA

## Conditional Events



**processorAvailableConditionalEvent**

- General**
  - Id**: processorAvailableConditionalEvent
  - Name**: Processor available
- Details**
  - Variable Name**: processorAvailable
  - Variable Event**: create, update
  - Condition Type**: Expression
  - Expression**: \${processorAvailable == true}

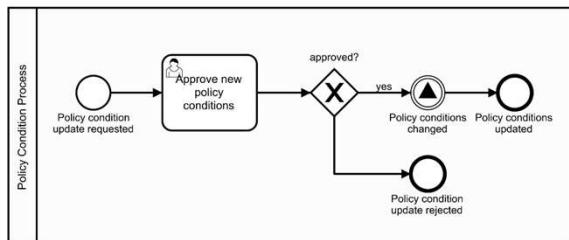
125

Limit evaluation to process variable

Limit evaluation to events

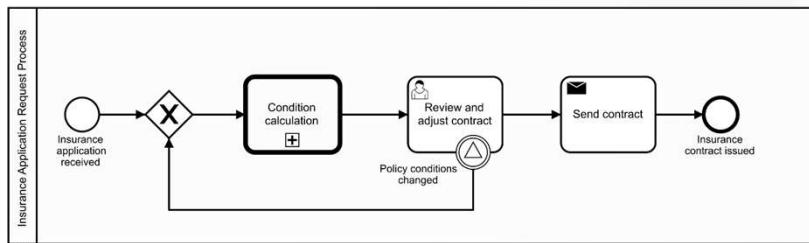
CAMUNDA

## Signal Events – Broadcast to All Process Instances



**policyConditionChangedSignalEvent**

- General**
  - Id**: policyConditionChangedSignalEvent
  - Name**: Policy conditions changed
- Details**
  - Signal**: policy\_condition\_changed\_signal (id=Signal\_118jje)
  - Signal Name**: policy\_condition\_changed\_signal



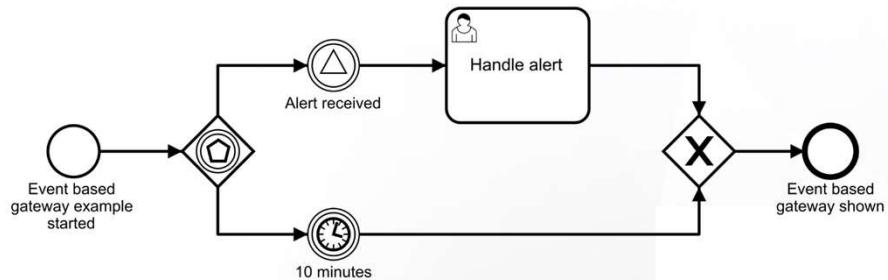
**Properties Panel**

- Signal**: policy\_condition\_changed\_signal (id=Signal\_118jje)
- Signal Name**: policy\_condition\_changed\_signal

CAMUNDA

126

## Event-Based Gateway – Simple Example



127



## 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.

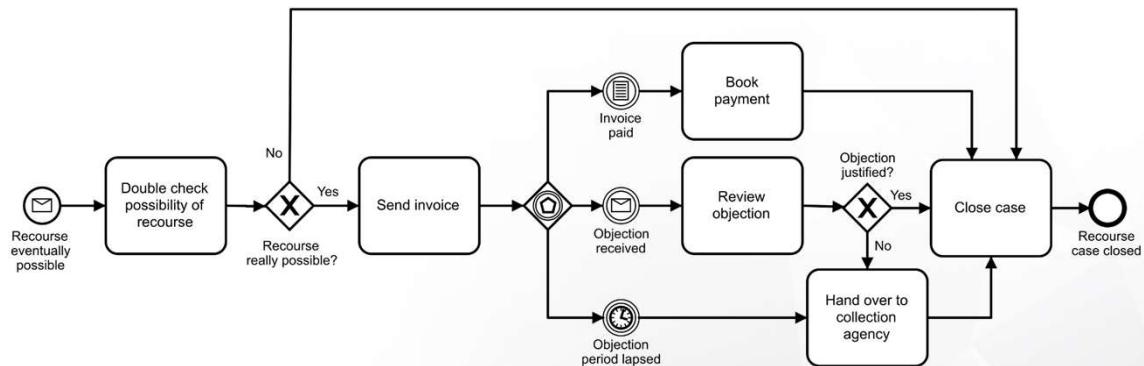
### Background information:

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.

128

CAMUNDA

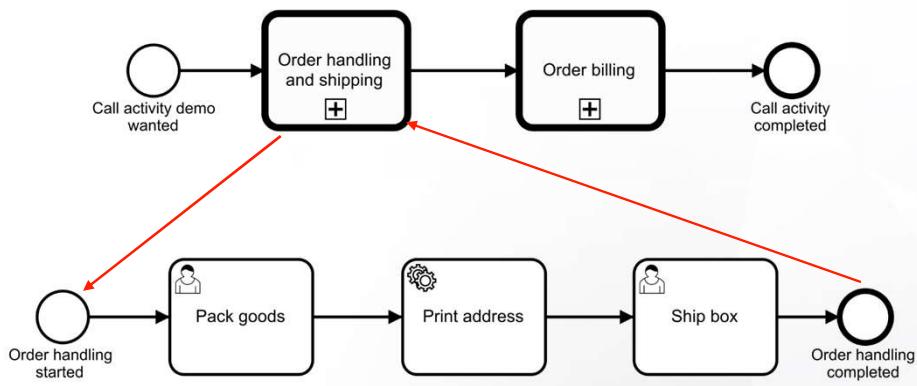
## Event-Based Gateway – Real World Example



129

CAMUNDA

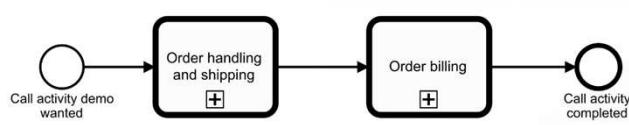
## Reusable Sub-Processes: Call Activities



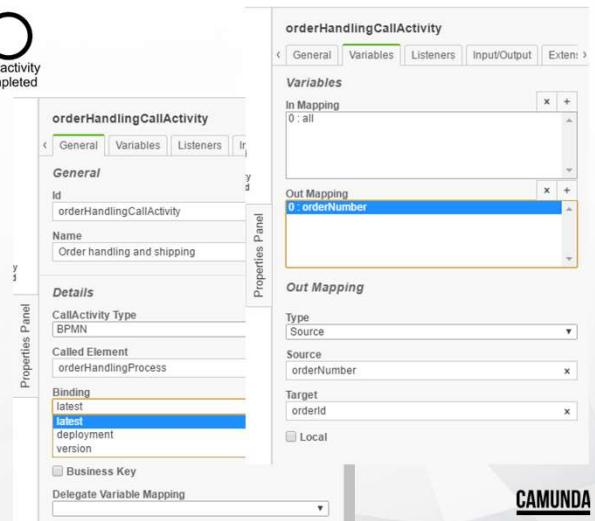
130

CAMUNDA

## Reusable Sub-Processes: Call Activities



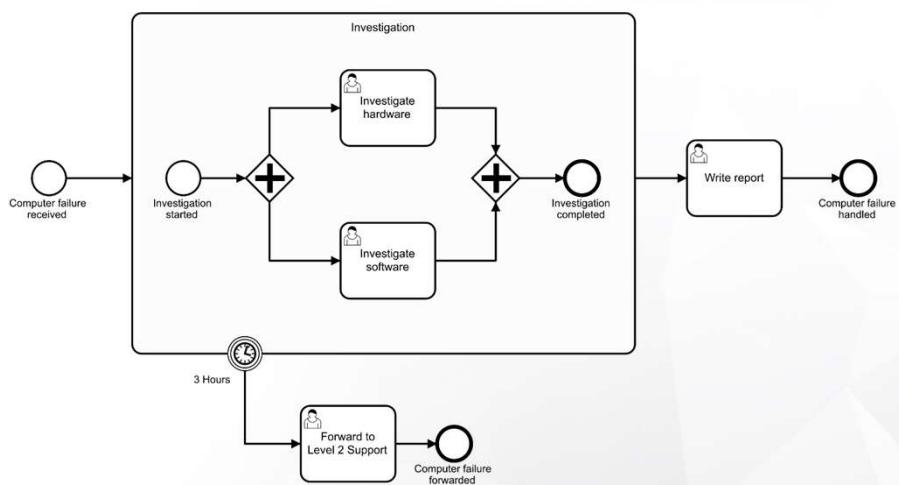
- Key is used to identify called process
- Process is resolved at runtime
- Variables can be mapped



131

CAMUNDA

## Embedded Sub-Processes



132

CAMUNDA

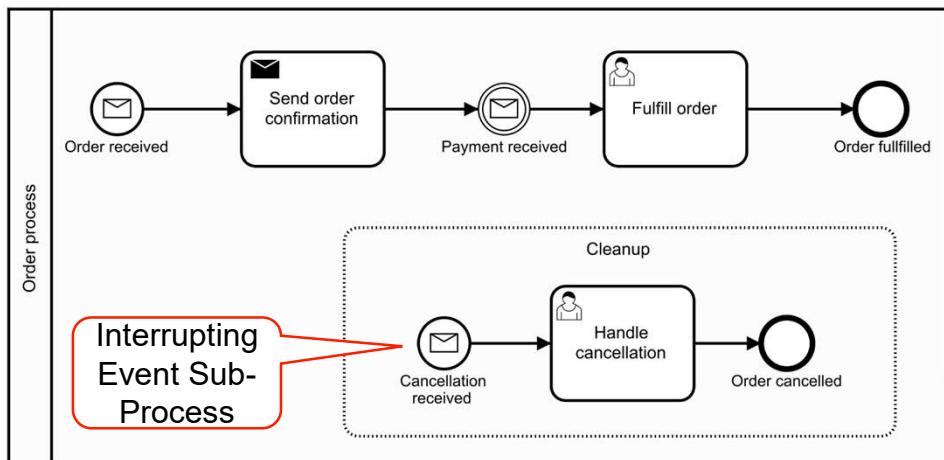
## The Sub-Process Is Part of the Main Process

```
<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>
```

133

CAMUNDA

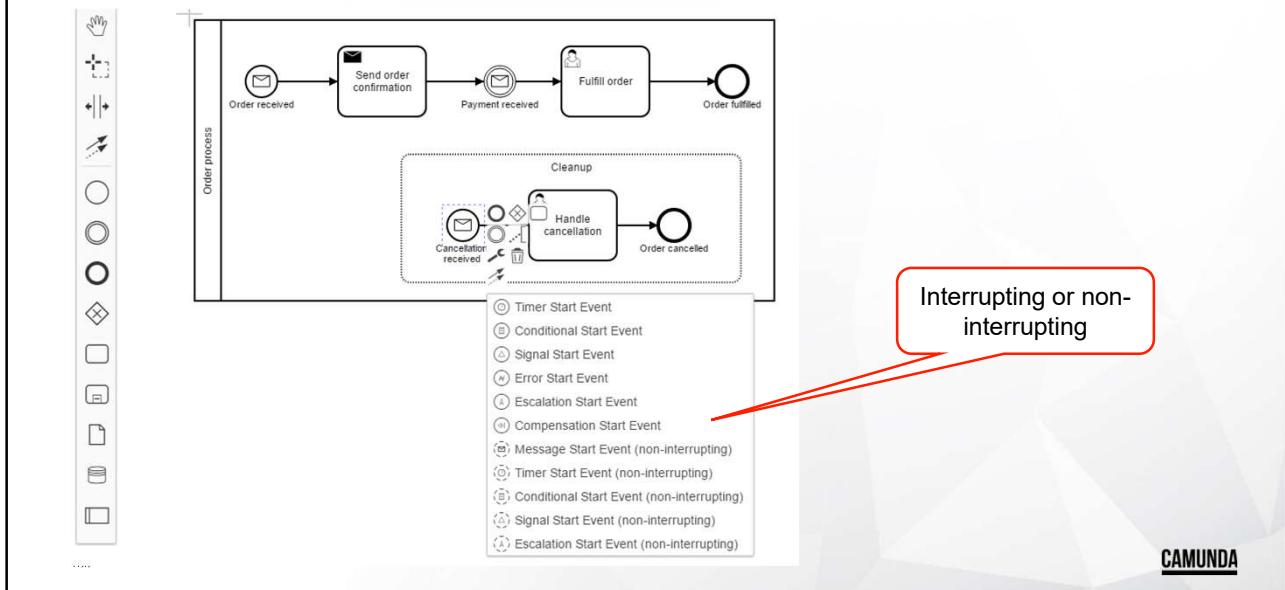
## Event Sub-Process



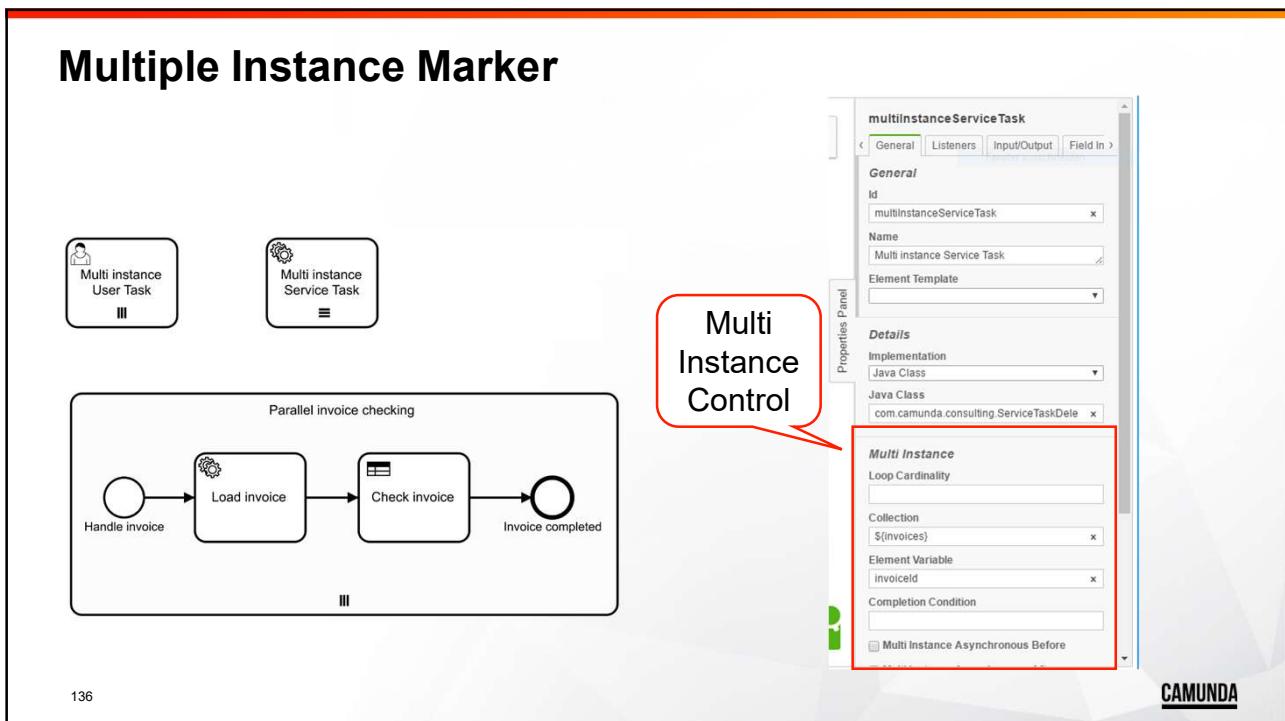
134

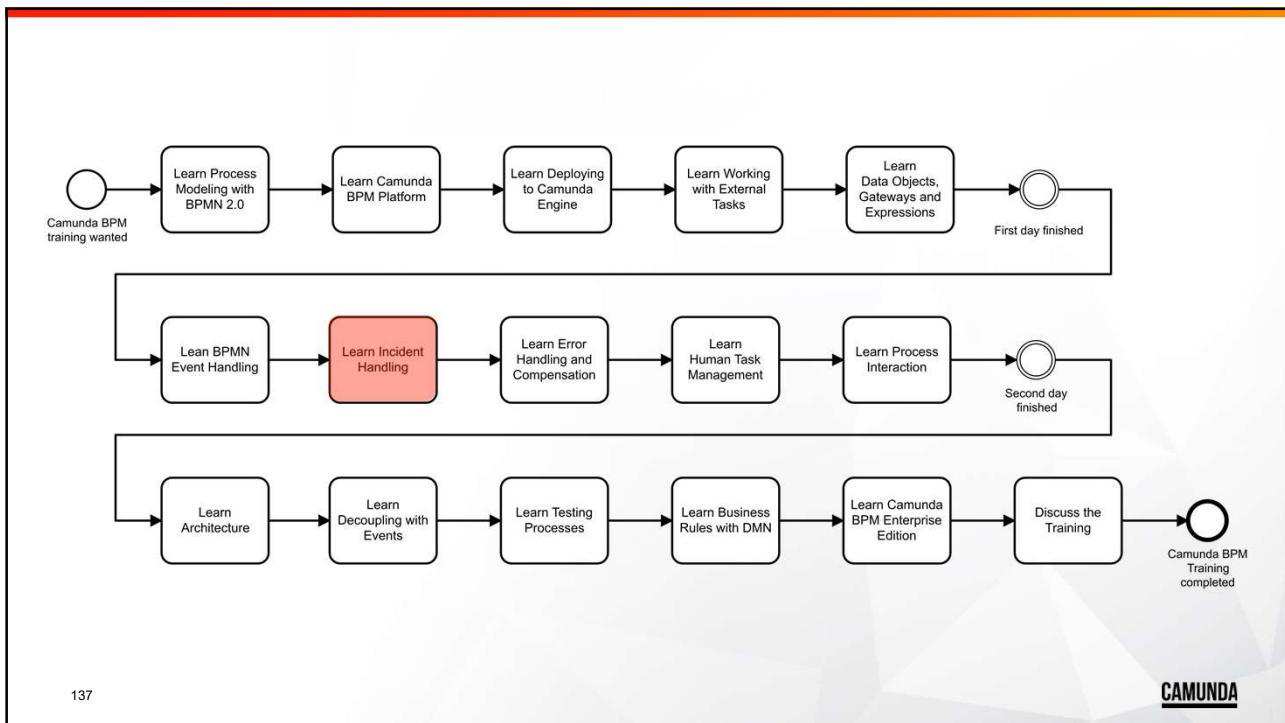
CAMUNDA

## Possible Events to Start an Event Sub-Process



## Multiple Instance Marker





## Something Wrong in the Worker

- Reply with
- POST /external-task/{id}/failure

```
{
  "workerId": "aWorker",
  "errorMessage": "Does not compute",
  "retries": 3,
  "retryTimeout": 60000
}
```

- Handle retries in the client
- Return the open retries in response



## Demo

### Archive Invoice failed

Incidents

## Retry Strategy

- Manage retries in the worker

POST /external-task/{id}/failure

```
{  
  "workerId": "aWorker",  
  "errorMessage": "Does not compute",  
  "retries": retryVariable - 1,  
  "retryTimeout": 60000  
}
```

**retries** A number of how often the task should be retried. Must be  $\geq 0$ . If this is 0, an incident is created and the task cannot be fetched anymore unless the retries are increased again. The incident's message is set to the errorMessage parameter.

**retryTimeout** A timeout in milliseconds before the external task becomes available again for fetching. Must be  $\geq 0$ .

## Implement Retries

```
client.subscribe("credit-charge", async function({ task, taskService }) {
  var remainingRetries = task.retries; // Get the retries
  if (remainingRetries == null) remainingRetries = 3; // initialize if empty
  remainingRetries = remainingRetries - 1; // decrement
  const possibleFailure = task.variables.get("possibleFailure");
  if (!possibleFailure && possibleFailure) {
    await taskService.handleFailure(task, {
      errorMessage: "sorry this failed",
      errorDetails: "because of my implementation",
      retries: remainingRetries,
      retryTimeout: 10000 });
  } else {
    const resultVars = new Variables();
    resultVars.set("remaining", amount - 50.00);
    await taskService.complete(task, resultVars);
  }
});
```

Javascript  
example

141

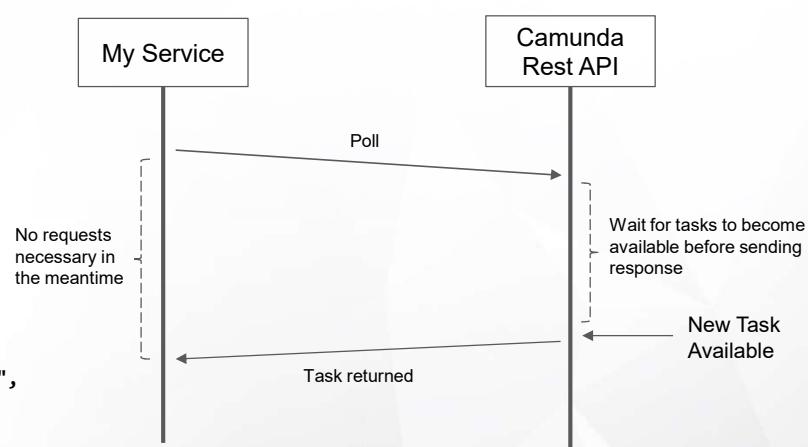
CAMUNDA

## Long Polling

Reduce latency between  
create and work

POST /external-task/fetchAndLock  
Request Body:

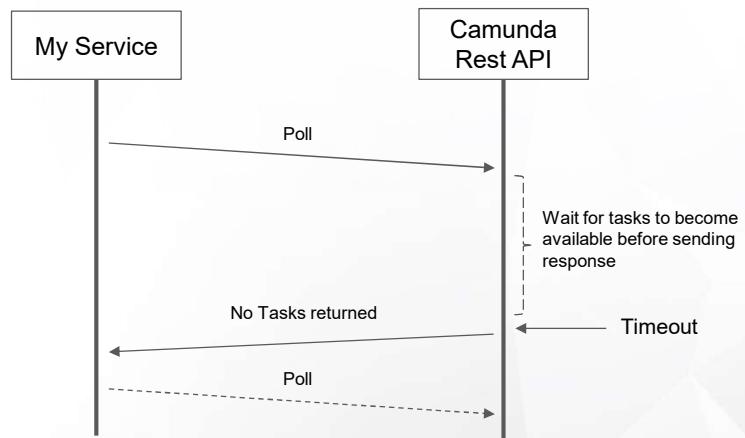
```
{
  "workerId": "myService",
  "asyncResponseTimeout": "60000",
  "maxTasks": 1,
  "usePriority": true,
  "topics": [{"topicName": "createOrder",
  "lockDuration": 10000, "variables": ["orderId"]}]
}
```



142

CAMUNDA

## What if no task is created?



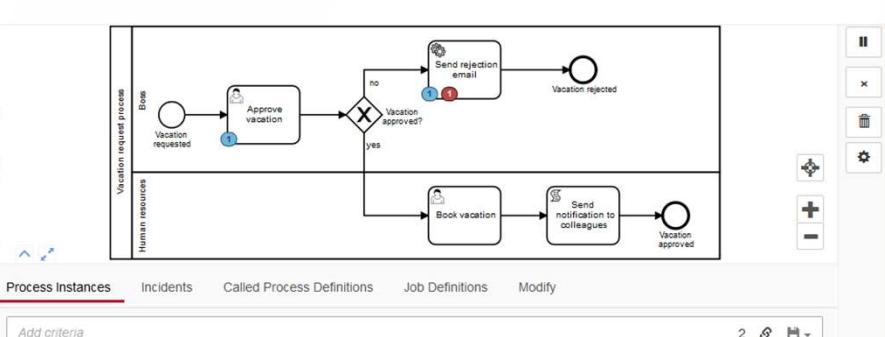
143

CAMUNDA

## Incidents Overview

5 process definitions deployed

State	Incidents	Running Instances
✓	0	7
✓	0	13
✗	1	7
✓	0	1
✗	1	2



144

e/Berlin

Powered by camunda BPM / v7.9.0-e

## Incidents Details

The screenshot shows the Camunda Cockpit interface. On the left, there is a sidebar with various process details:

- Instance ID: 4f0b931c-689e-11e8-8725-00155d230939
- Business Key: null
- Definition Version: 1
- Definition ID: VacationRequestProcess:1:89192dff...
- Definition Key: VacationRequestProcess
- Definition Name: Vacation request
- Tenant ID: null
- Deployment ID: 890fb81b-6893-11e8-8725-00155d2...
- Super Process Instance ID: null
- Related: Migration

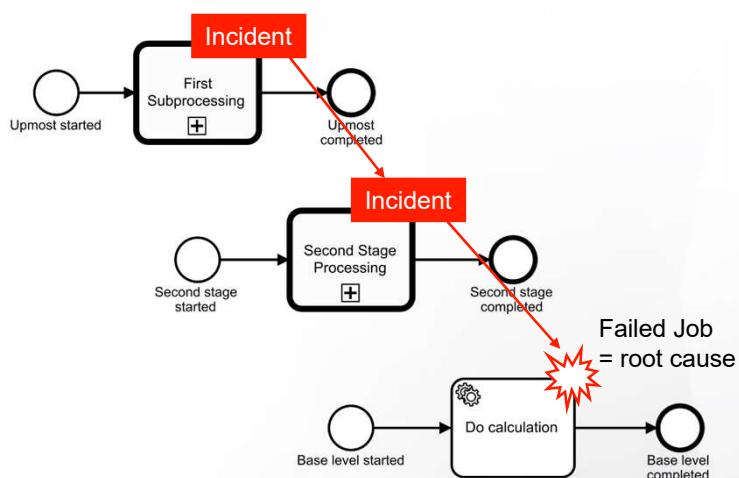
The main area displays a process diagram titled "Vacation request process". The diagram starts with a "Boss" node, followed by a task "Approve vacation", a decision diamond "Vacation approved?", and two outgoing paths: "no" leading to "Send rejection email" and "yes" leading to "Book vacation" and "Send notification to colleagues". The process ends with "Vacation rejected" and "Vacation approved".

A red callout box points to a row in the "Incidents" table:

Message	Timestamp	Activity	Cause Process Instance ID	Root Cause Process Instance ID	Type	Action
Could not send notification...	2018-06-05T11:24:01	Send rejection email			Failed Job	<input type="button" value="Retry failed Job"/>

At the bottom right, the Camunda logo is visible.

## Drill Down



146

CAMUNDA



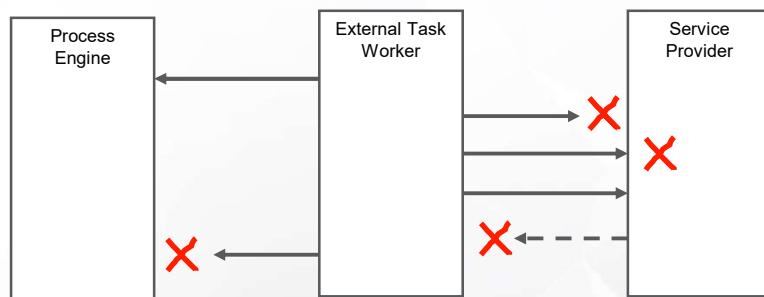
## Exercise 7

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks

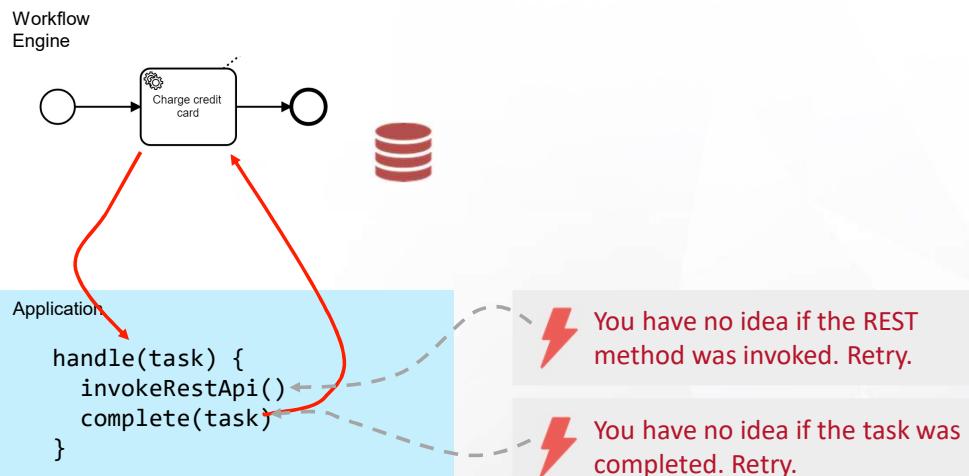


### Automatic Retry of External Service Tasks

- The worker dies
  - When? Maybe between work and complete
- Lock is timed out
- Next worker picks up again
  - Work is done again



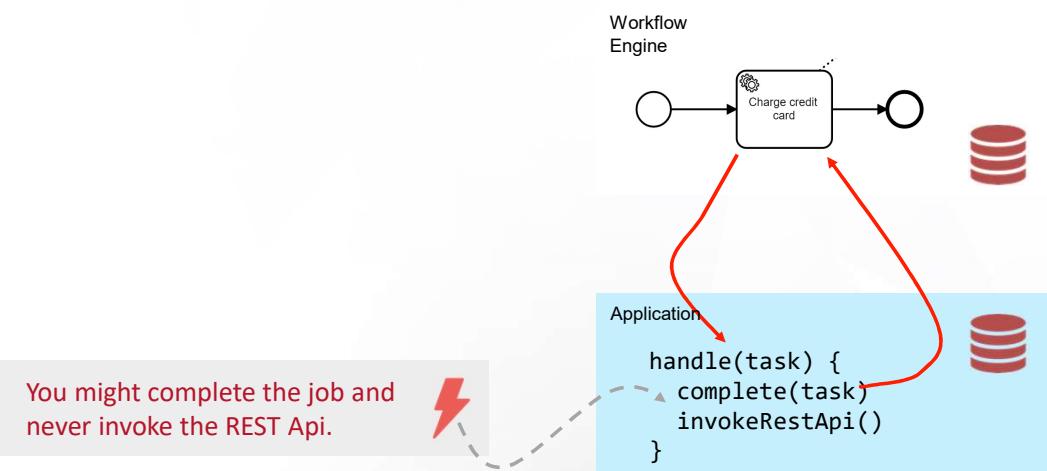
## „At Least Once“



149

CAMUNDA

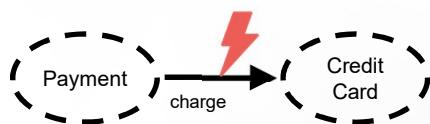
## „At Most Once“



150

CAMUNDA

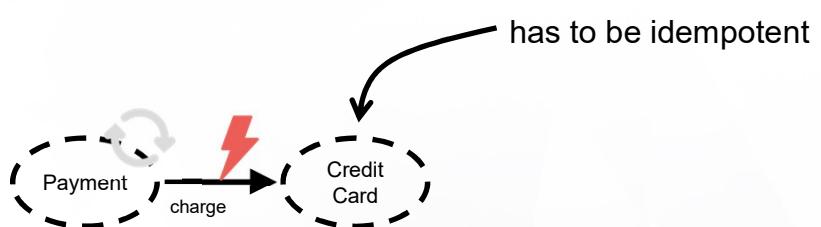
## Network Problems



151

CAMUNDA

## Strategy: Retry



Charge Credit Card  
cardNumber  
amount

← Not idempotent

Charge Credit Card  
cardNumber  
amount  
transactionId

← Idempotent

152

CAMUNDA

## Support for Idempotency

- State of process instance: Process variables
- Unique Key
  - Business Key
  - Transaction ID
  - ...
- Create early
- Retry is possible

153

CAMUNDA

## Remember the Retry Strategy

- Manage retries in the worker

POST /external-task/{id}/failure

```
{  
  "workerId": "aWorker",  
  "errorMessage": "Does not compute",  
  "retries": retryVariable - 1,  
  "retryTimeout": 60000}
```

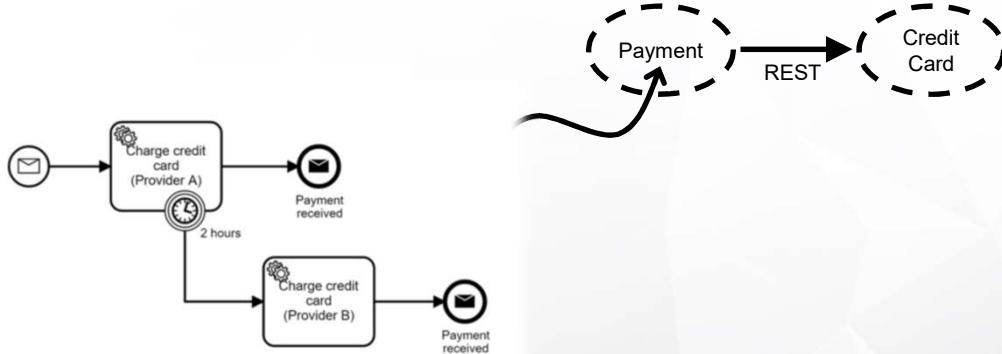
**retries** A number of how often the task should be retried. Must be  $\geq 0$ . If this is 0, an incident is created and the task cannot be fetched anymore unless the retries are increased again. The incident's message is set to the errorMessage parameter.

**retryTimeout** A timeout in milliseconds before the external task becomes available again for fetching. Must be  $\geq 0$ .

154

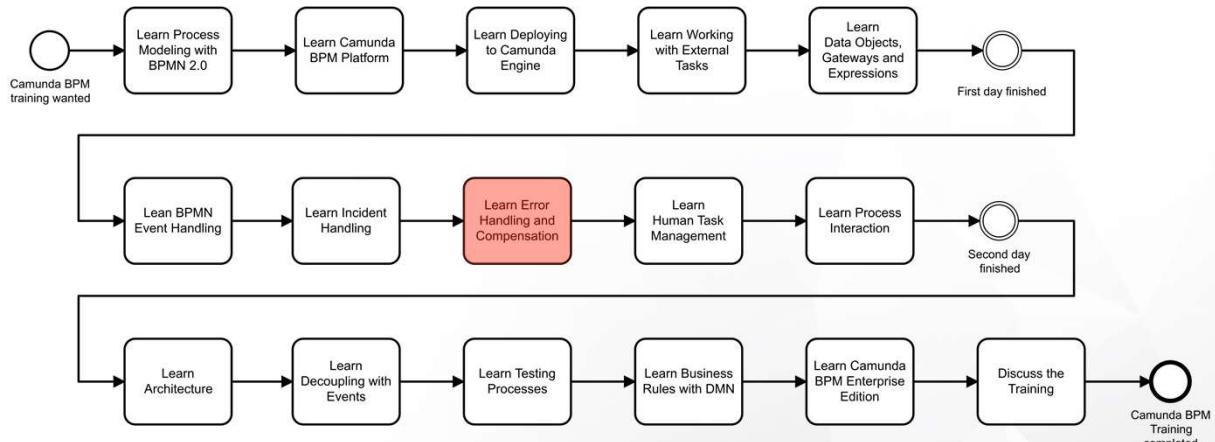
CAMUNDA

## Fallbacks Increase Resilience



155

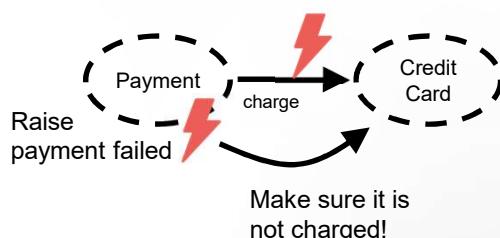
CAMUNDA



156

CAMUNDA

## Strategy: Cleanup

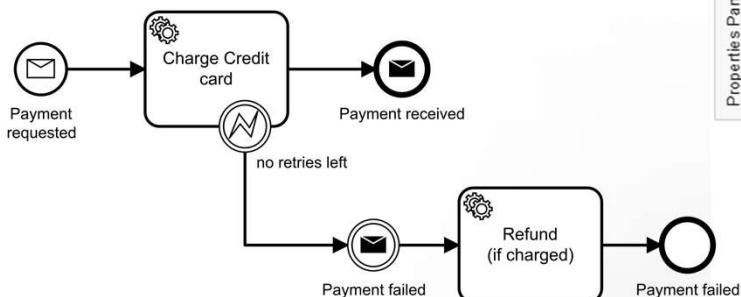


```
Cancel charge  
cardNumber  
amount  
transactionId
```

157

CAMUNDA

## Error Handling



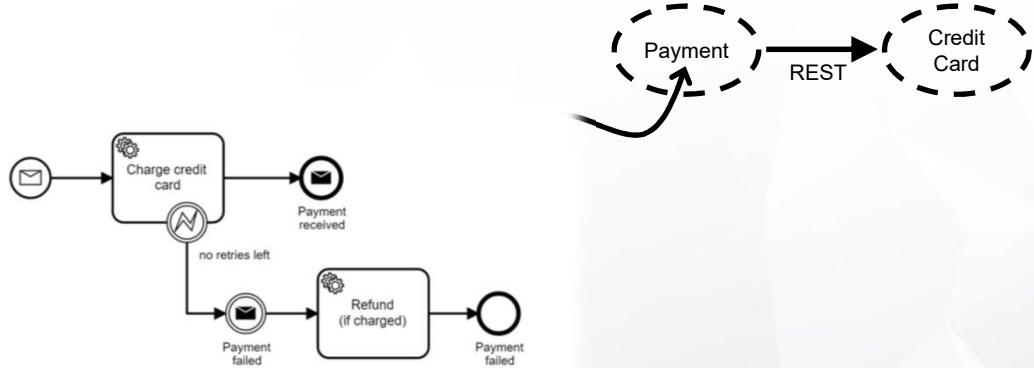
Properties Panel

Details	
Error	+ Charging error (id=Error_1n8mxyh) ▾
Error Name	X Charging error
Error Code	X chargingError
Error Message	
Error Code Variable	errorCode
Error Message Variable	errorMessage

158

CAMUNDA

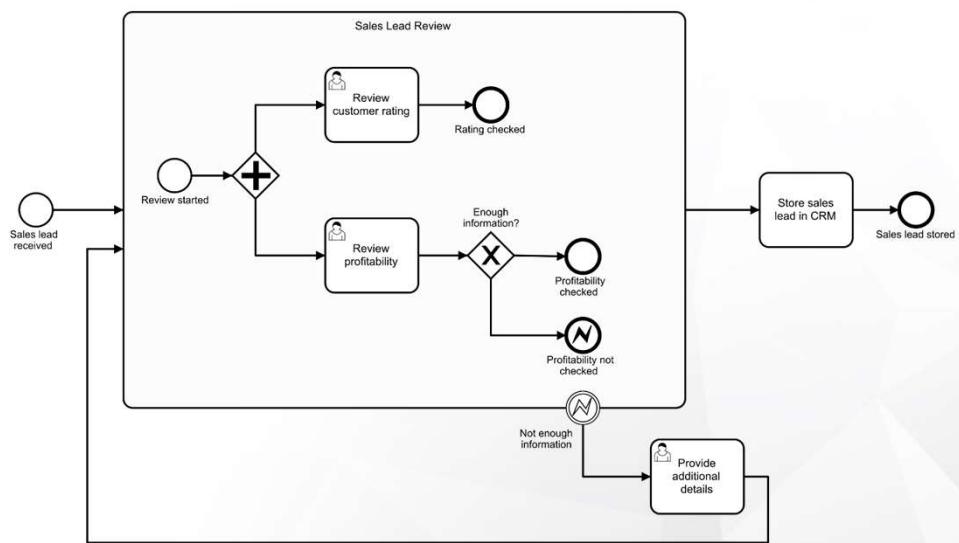
## Example



159

CAMUNDA

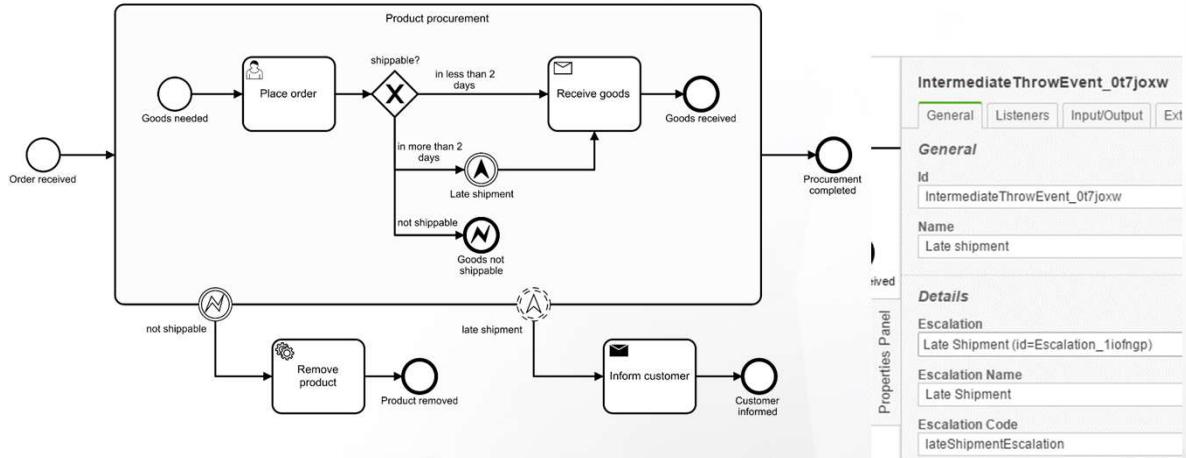
## Error Event



160

CAMUNDA

## Escalation and Error Events

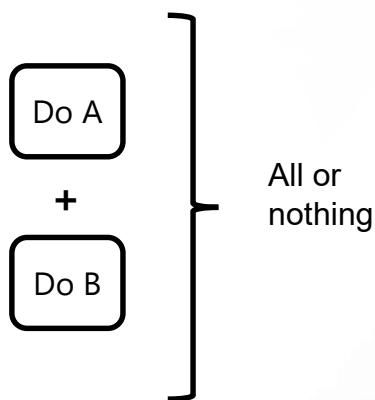


161

CAMUNDA

## Transactions on Central Servers

Once upon a time:



```
try {  
    tx.begin();  
    doA();  
    doB();  
    tx.commit();  
} catch (Exception e) {  
    tx.rollback();  
}
```

Or simply:

```
@Transactional  
public void createCustomer(Customer cust)  
{  
    // ...  
}
```

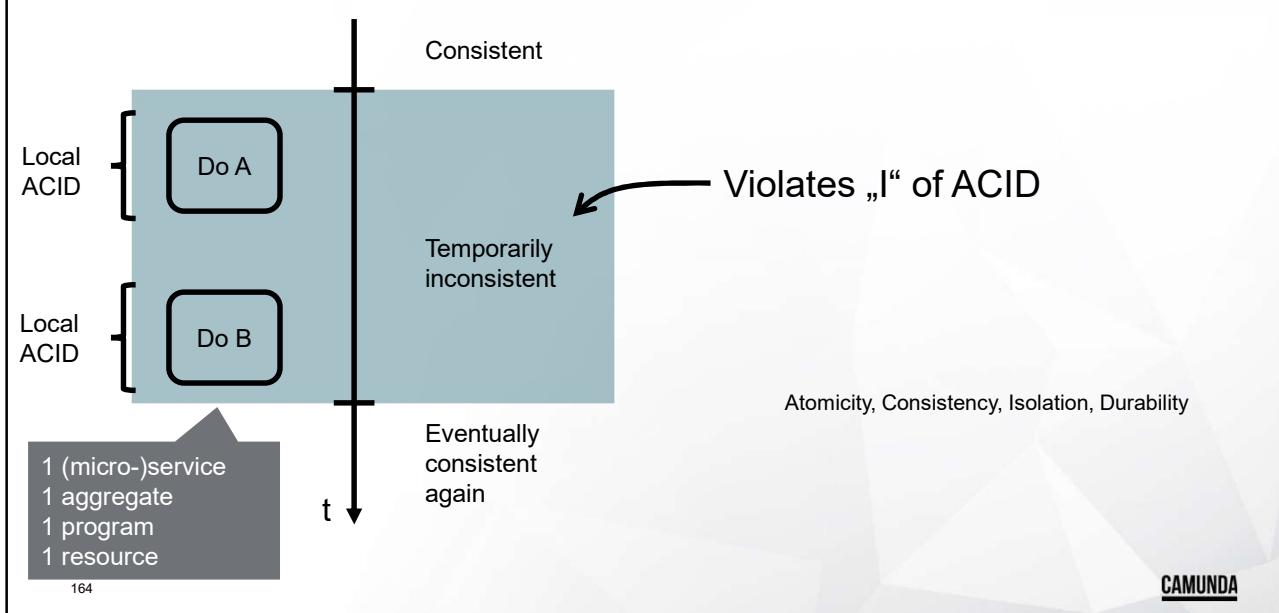
162

CAMUNDA

## Distributed Systems

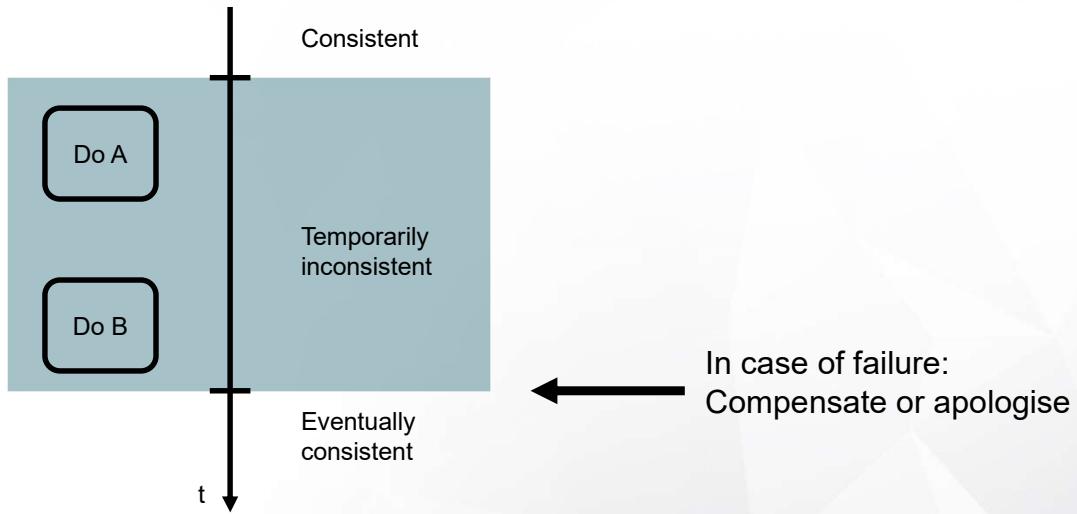


### That Means



CAMUNDA

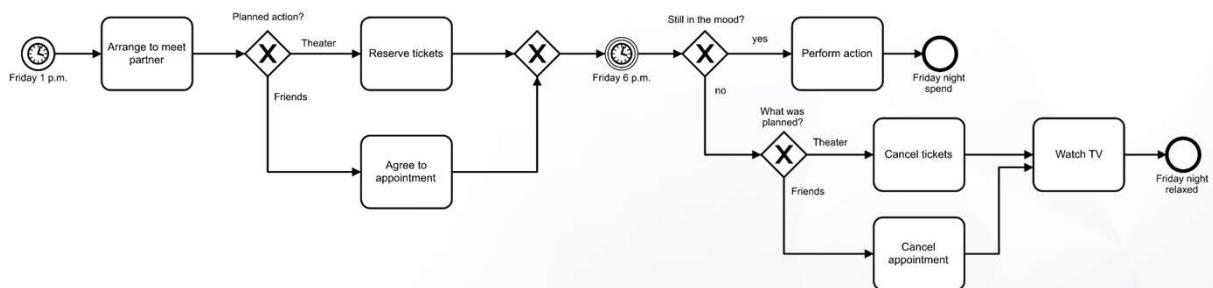
## That Means



165

CAMUNDA

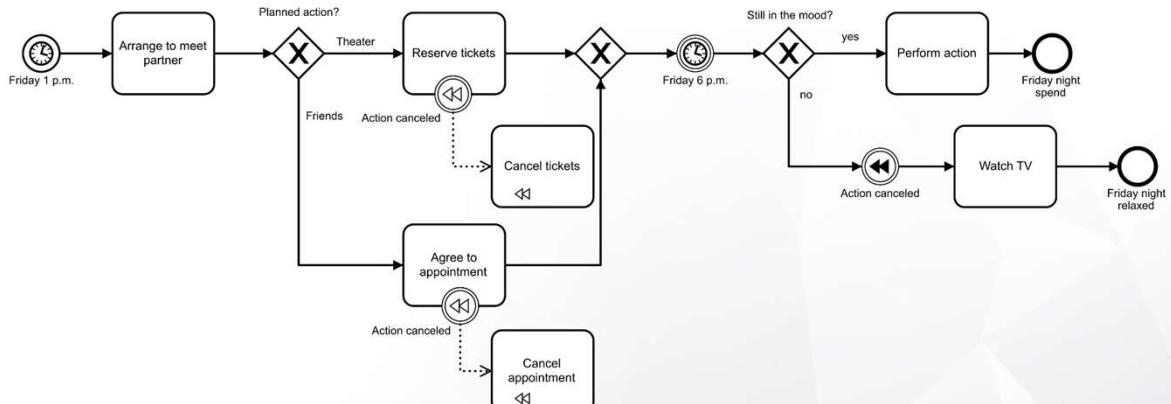
## Compensation



166

CAMUNDA

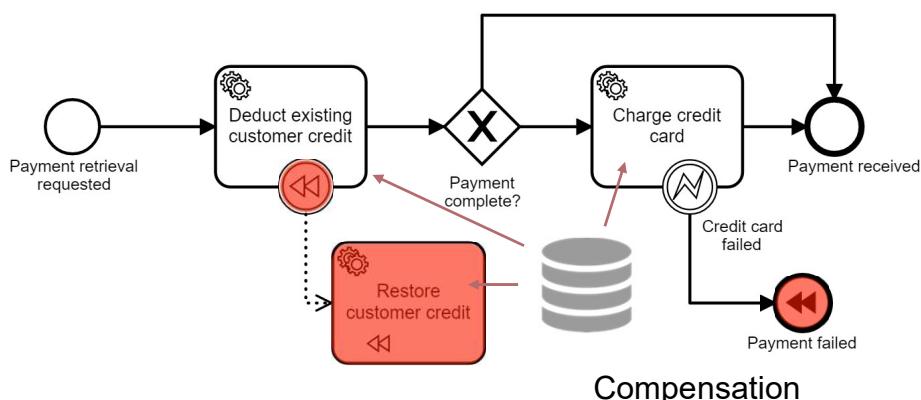
## Compensation Events



167

CAMUNDA

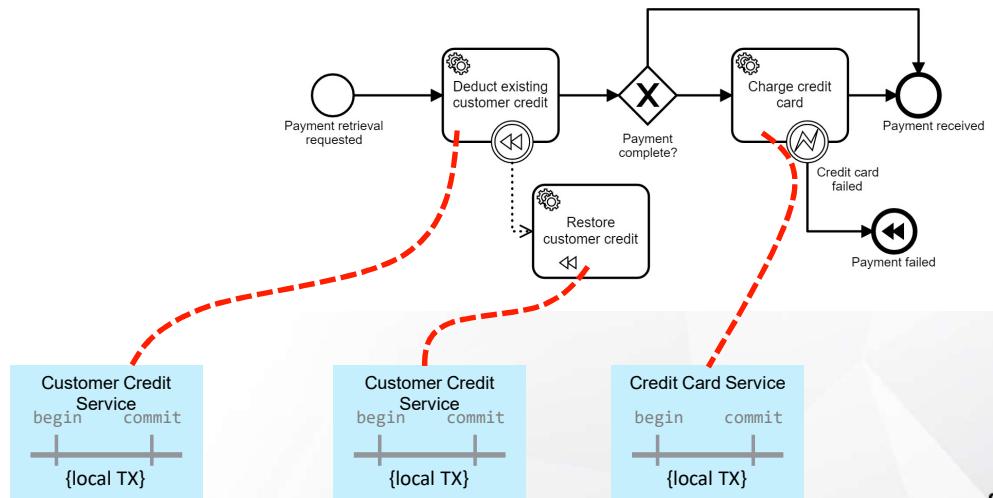
## Distributed Transactions Using Compensation \*



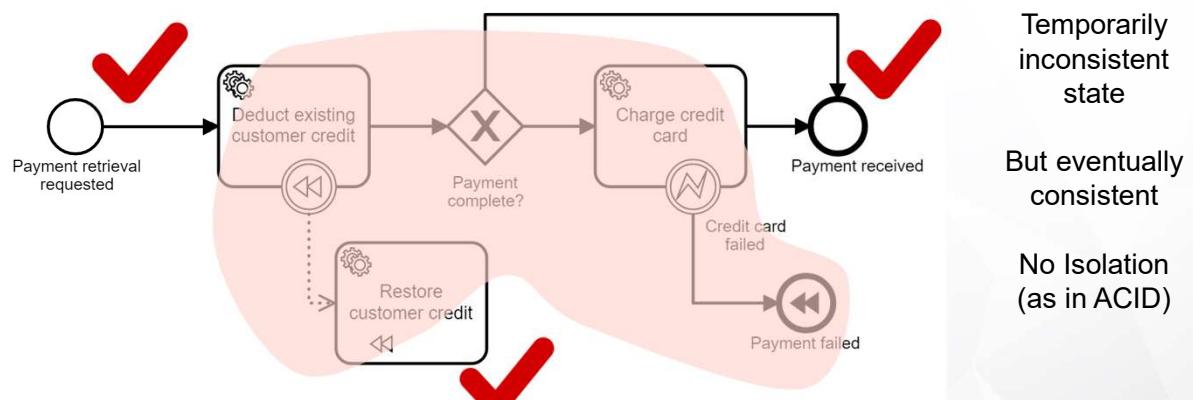
168

CAMUNDA

## A Saga Coordinator

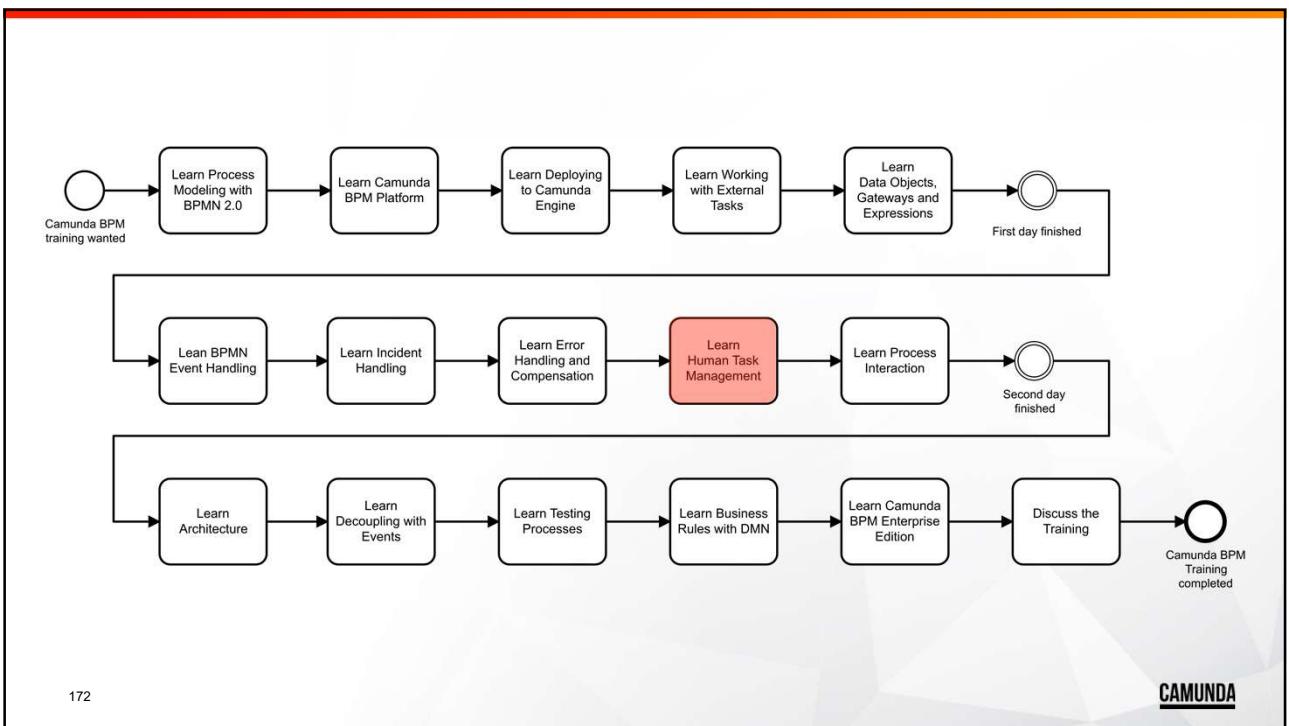


## Relaxed Consistency

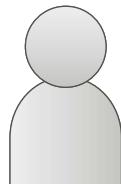
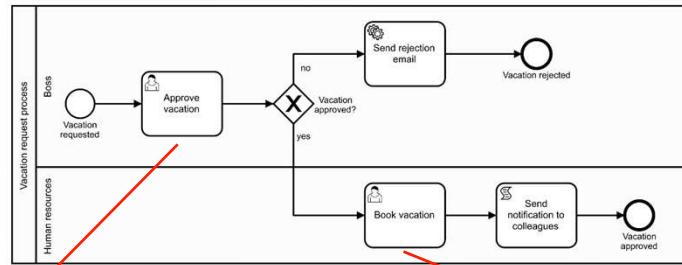


**Exercise 8**

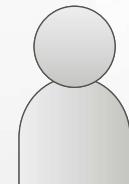
<https://training.camunda.com/microservices>  
 user: microservices  
 password: camundarocks



## Task Management



173

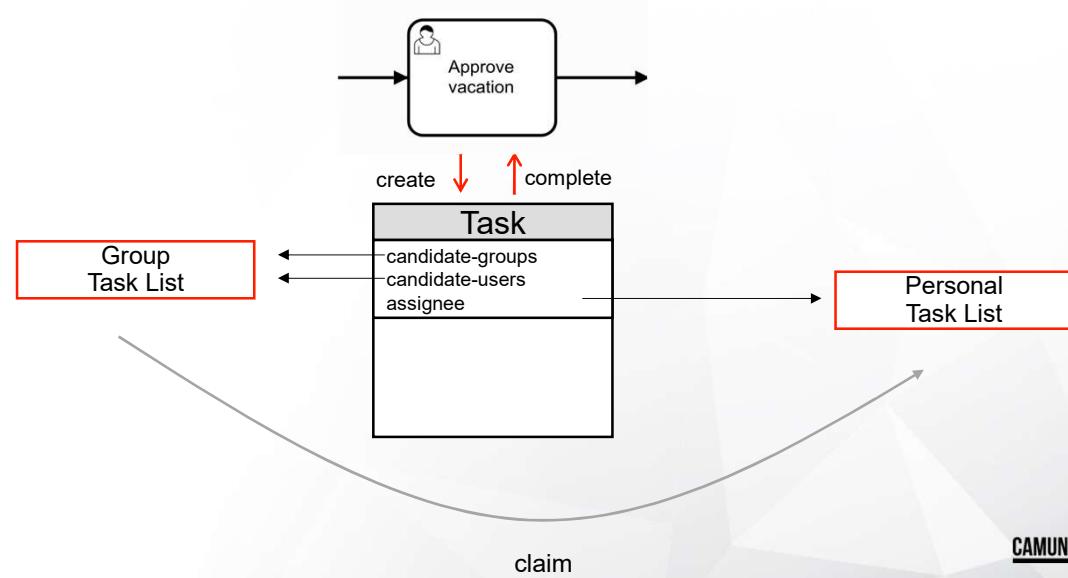


Human Resources



CAMUNDA

## Task Management



174

CAMUNDA

## Tasklist in Camunda BPM

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with navigation options like 'My Tasks', 'My Group Tasks (7)', 'Accounting', 'John's Tasks', 'Mary's Tasks', 'Peter's Tasks', and 'All Tasks'. A red circle with the number '1' is overlaid on this sidebar area. The main panel shows a list of tasks. Two specific tasks are highlighted with red circles and numbered '2': 'Approve vacation' and 'Approve Invoice'. The 'Approve vacation' task is selected, showing its details: 'Vacation request', 'Created 2 minutes ago', and a form titled 'Approve vacation' with fields for 'Employee' (set to 'John'), 'From' (set to '2018-07-17T00:00:00'), and 'Until' (set to '2018-07-24T00:00:00'). A red circle with the number '3' is overlaid on the approval form. At the bottom right of the interface, the text 'Powered by camunda BPM / v7.9.0-ee' and the Camunda logo are visible.

175

CAMUNDA

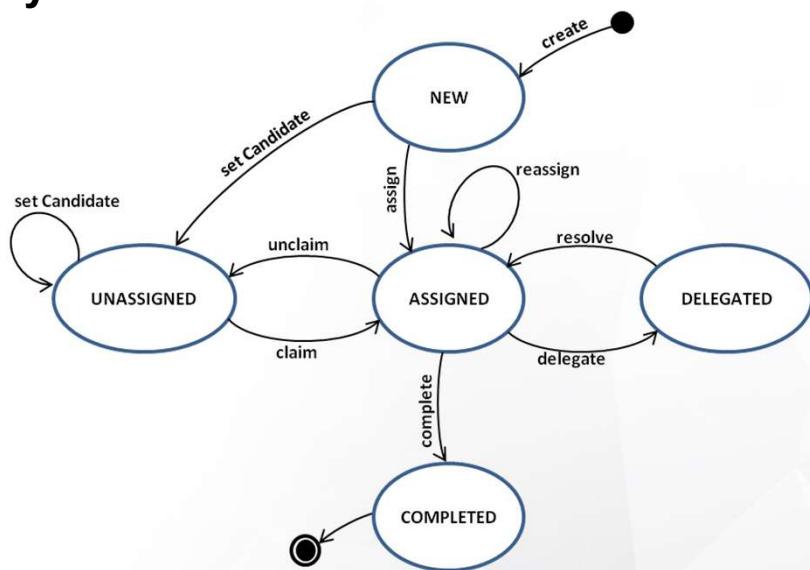
## Groups & Personal Task List (Via Filters)

The screenshot shows the 'Edit filter' dialog box. The left sidebar has the same navigation as the previous screenshot. The main area is titled 'Edit filter' and contains a 'General' tab with a 'Criteria' section. It includes a note: 'This section is aimed to set the parameters used to filter the tasks. Keys marked with a \* accept expressions as value.' Below this are two criteria: 'Unassigned' (Key: Rem..., Value: Unassigned) and 'Candidate Groups\*' (Key: Rem..., Value: \${currentUserGroups()}). A note next to the second criterion says: 'List of values separated by comma or an expression which evaluates to a list. E.g.: "camunda-admin, accounting" or "\${currentUserGroups}"'. There are checkboxes for 'Include assigned tasks' (unchecked) and 'ALL' of the criteria are met. Below these are tabs for 'Permissions' and 'Variables', and buttons for 'Delete filter', 'Close', and 'Save'.

176

CAMUNDA

## Task Lifecycle

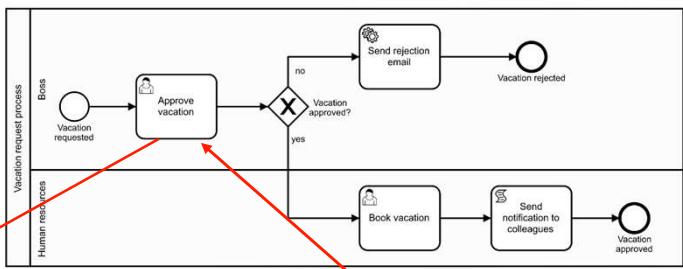


177

CAMUNDA

## Task Management API

GET /task



GET /task/{id}/form

Approve vacation

Vacation request

Set follow-up date Set due date Management

Form History Diagram Description

Approve the vacation request

Employee

Mary

From

2016-12-20T00:00:00

Until

2016-12-23T23:59:00

Approve?

Add Comment +

> <

Approve?

Save Complete

CAMUNDA

## 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());
```

179

CAMUNDA

## 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);
```

180

CAMUNDA

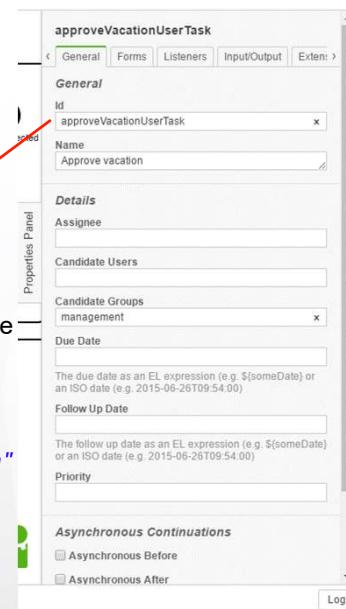
## Key vs. ID vs. Name

GET /task

- task.id: Task Instance ID
- task.taskDefinitionKey: BPMN User Task ID
- task.name: BPMN User Task Name

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

181



CAMUNDA

## Task Listener

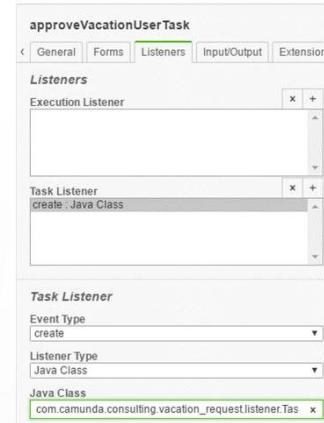
### Typical Use Cases:

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

### Available events:

create  
update  
assignment  
complete  
delete

182



### Listener Type

Script
Expression
Java Class
Delegate Expression

CAMUNDA

## Task Forms

The screenshot illustrates a vacation request process and its associated task form.

**Start process:** A form with fields for "Your Name", "Start date", "Last day of vacation", and a "Back" button. A red arrow points from the "Vacation requested" event in the process diagram to this form.

**Human resources:** A vertical column containing the process diagram and the task form.

**Process Diagram:**

```

graph LR
    Start(( )) --> Boss((Boss))
    Boss --> Approve[Approve vacation]
    Approve --> Decision{Vacation approved?}
    Decision -- no --> Rejection[Send rejection email]
    Decision -- yes --> End(( ))
  
```

**Task Form:** An "Approve vacation" form with tabs for "Form", "History", "Diagram", and "Description". It includes fields for "Employee" (Mary), "From" (2016-12-20T00:00:00), "Until" (2016-12-23T23:59:00), and an "Approve?" checkbox. Buttons for "Save" and "Complete" are at the bottom.

**User Task Configuration:** A modal window titled "approveVacationUserTask" showing the "Forms" tab, where "Form Key" is set to "embedded:app:forms/embedded/task-form.html".

Page number: 183

CAMUNDA

## Task Forms in Camunda Tasklist

The screenshot shows four examples of task forms in the Camunda Tasklist:

- Generic form:** A "Second User Task" form under "Generic Form Process". It includes tabs for "Form", "History", "Diagram", and "Description". A "Business Key" section allows adding variables. A "Complete" button is at the bottom.
- Generated form:** A "Start process" form with fields for "Firstname", "Lastname", and "Date of Birth". A "Close" and "Start" button are at the bottom.
- Embedded form:** An "Approve vacation" form with tabs for "Form", "History", "Diagram", and "Description". It includes fields for "Employee" (Mary), "From" (2016-12-20T00:00:00), "Until" (2016-12-23T23:59:00), and an "Approve?" checkbox. Buttons for "Save" and "Complete" are at the bottom.
- External form:** A screenshot of a web browser displaying a "Hello World" application, which is an external form.

Page number: 184

CAMUNDA

## 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

Form History Diagram Description

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

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

Load Variables ⓘ Complete

CAMUNDA

185

## 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

General Forms Listeners Input/Output Extent: >

Forms

Form Type Form Data

Form Fields Employee from until approvedBy

Properties Panel

ID approvedBy

Type string

Label Approved by

Default Value

Validation

Add Constraint +

Name readonly Config

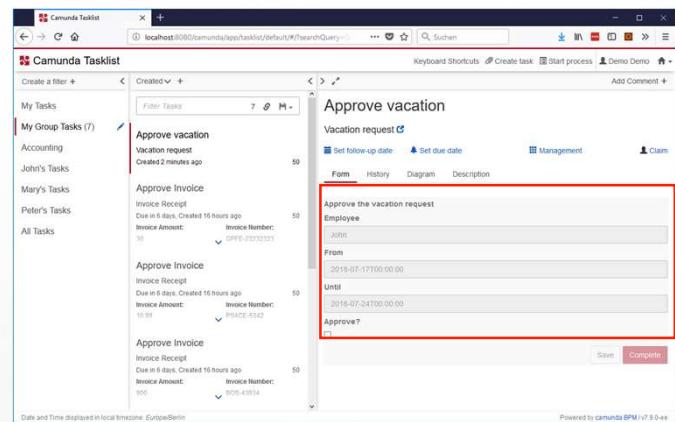
Properties

CAMUNDA

186

## Embedded Task Forms

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

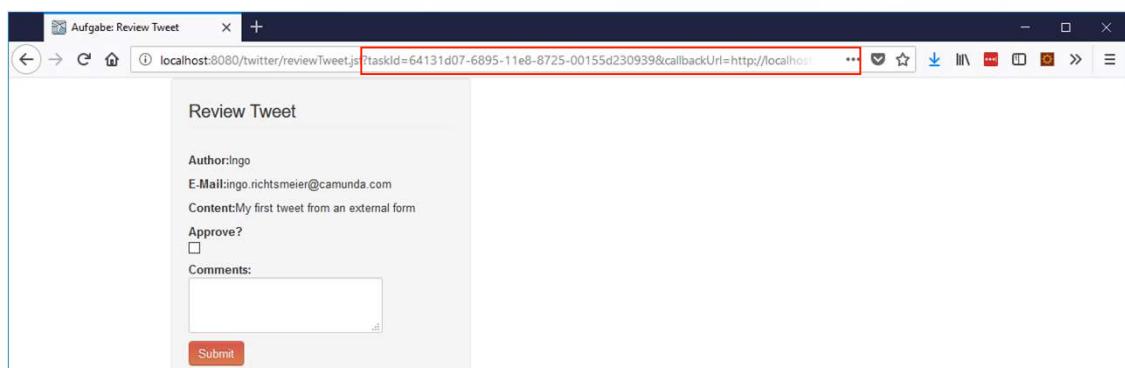


187

CAMUNDA

## External Task Form

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



188

CAMUNDA

## 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
- *Angular JS 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/>

189

CAMUNDA

## Simple Form Example

```
<strong>Here you can request your vacation</strong>

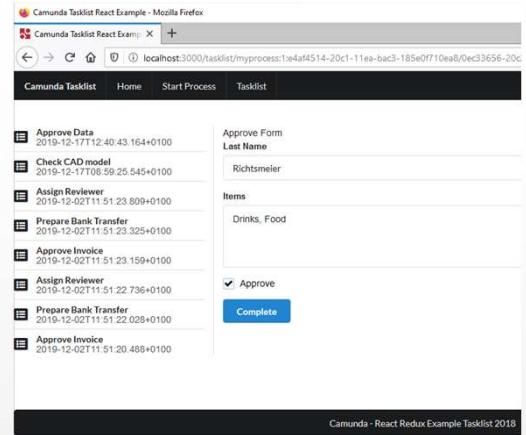
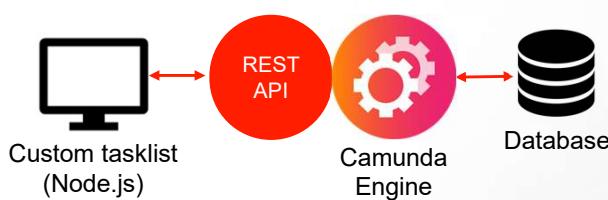
<form class="form-horizontal">
  <div class="control-group">
    <label class="control-label">Your Name</label>
    <div class="controls">
      <input type="text"
            cam-variable-name="employee"
            cam-variable-type="String"
            required
            class="form-control" />
    </div>
  </div>
  <div class="control-group">
    <label class="control-label">Start date</label>
    <div class="controls">
      <input type="text"
            cam-variable-name="from"
            cam-variable-type="Date"
            required
            class="form-control" />
    </div>
  </div>
  <div class="control-group">
    <label class="control-label">Last day of vacation</label>
    <div class="controls">
      <input type="text"
            cam-variable-name="until"
            cam-variable-type="Date"
            required
            class="form-control" />
    </div>
  </div>
  <script cam-script type="text/form-script">
    // custom JavaScript goes here
  </script>
</form>
```

190

CAMUNDA

## Custom Tasklists

- Leverage the REST API
- Can use any technology (Vue.js, React, Angular, ...)
- Be creative with the Form Key



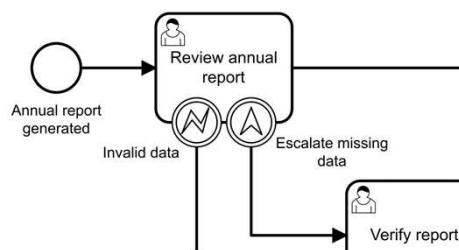
<https://blog.camunda.com/post/2018/02/custom-tasklist-examples/>

<https://github.com/camunda-consulting/code/tree/master/snippets/camunda-tasklist-examples/>

191

CAMUNDA

## Error and Escalation Event on User Tasks



```
<button cam-error-code="bpmn-error-543" cam-error-message="anErrorMessage" class="form-control">Abort</button>
```

```
<button cam-escalation-code="escalation-123" class="form-control">Escalate</button>
```

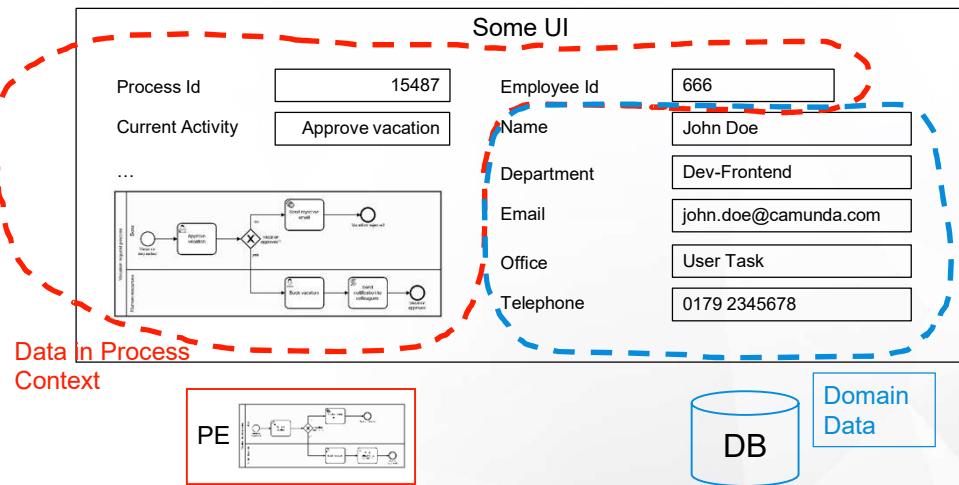
```
POST /task/aTaskId/bpmnError
{
  "errorCode": "bpmn-error-543",
  "errorMessage": "an error message",
  "variables": ...
}

POST /task/aTaskId/bpmnEscalation
{
  "escalationCode": "bpmn-escalation-432",
  "variables": ...
}
```

192

CAMUNDA

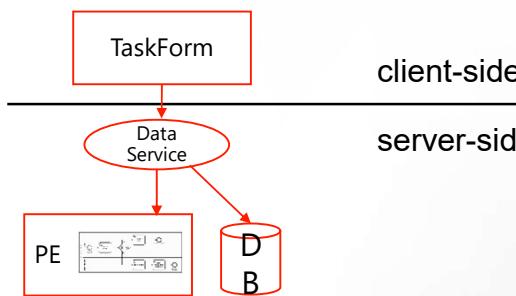
## Typical: Process Data and External Data Are Mixed



193

CAMUNDA

## How to Call Data Service



### Composed Service

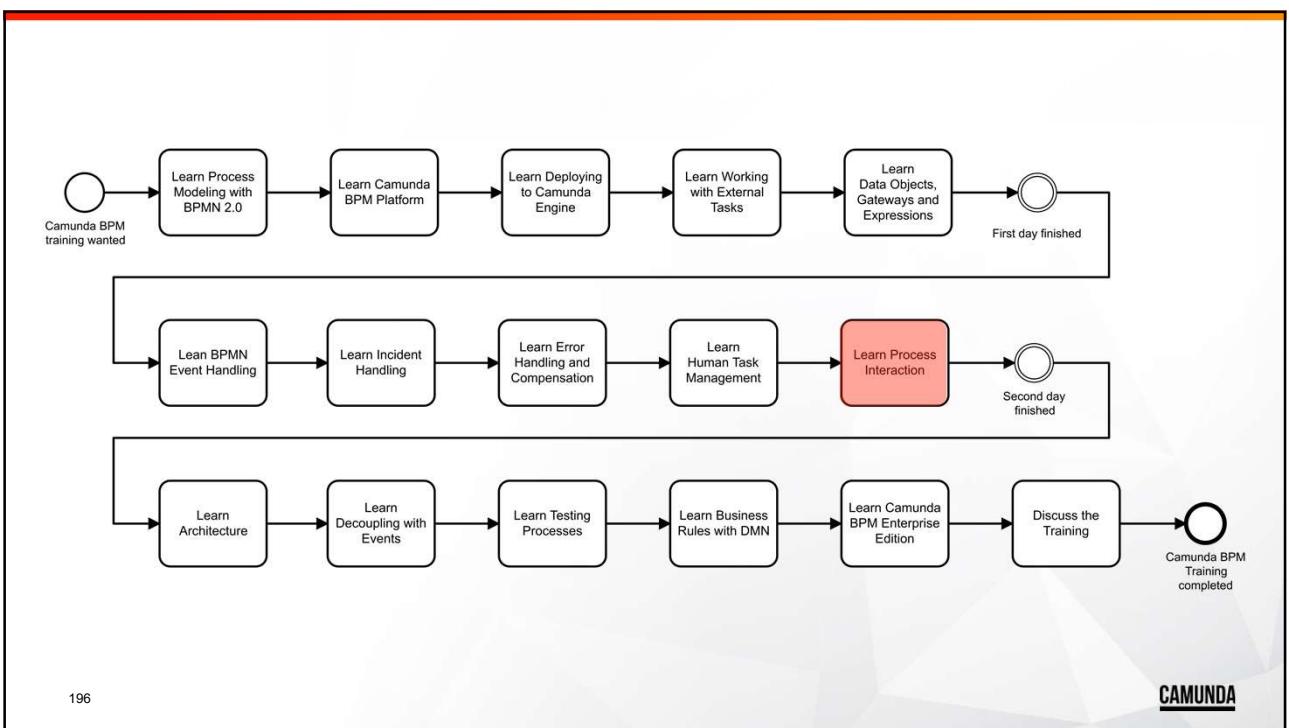
- The UI calls a combined service
- This service calls the Data Service and the Process Service
- TX possible

194

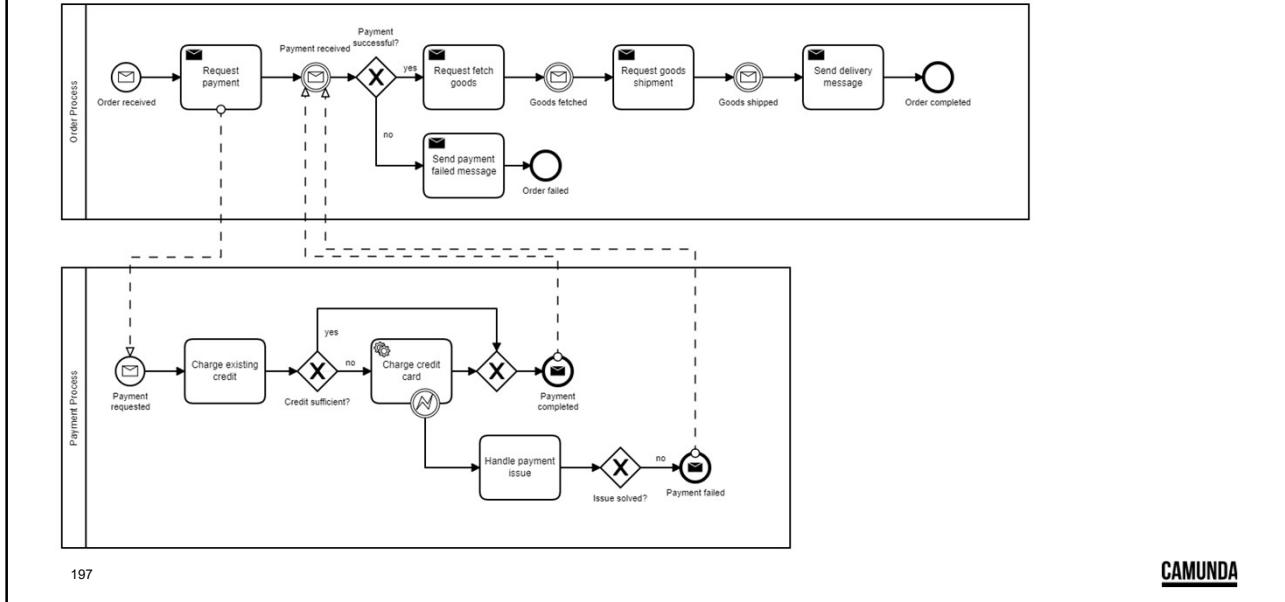
CAMUNDA

**Exercise 9**

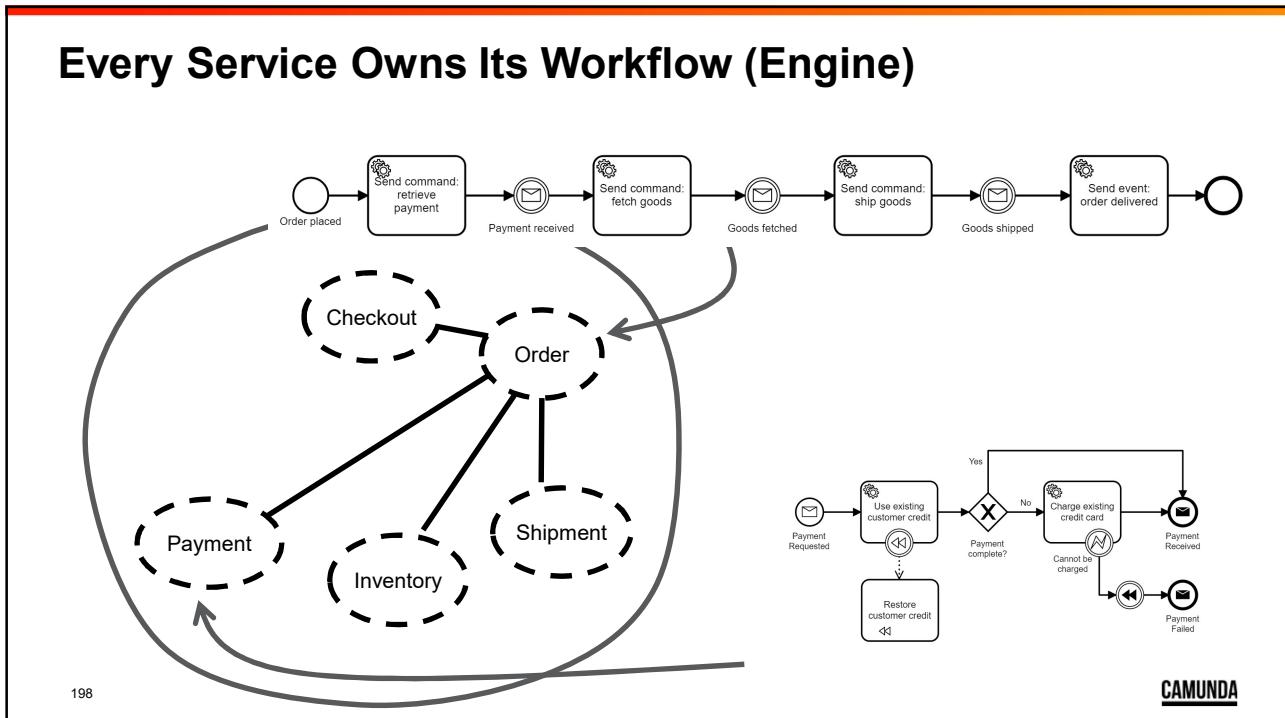
<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks



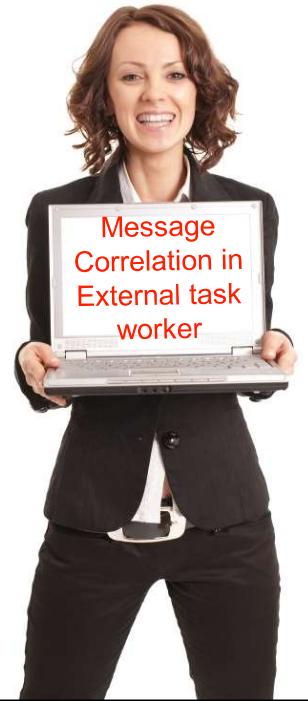
## Collaboration Diagram



## Every Service Owns Its Workflow (Engine)



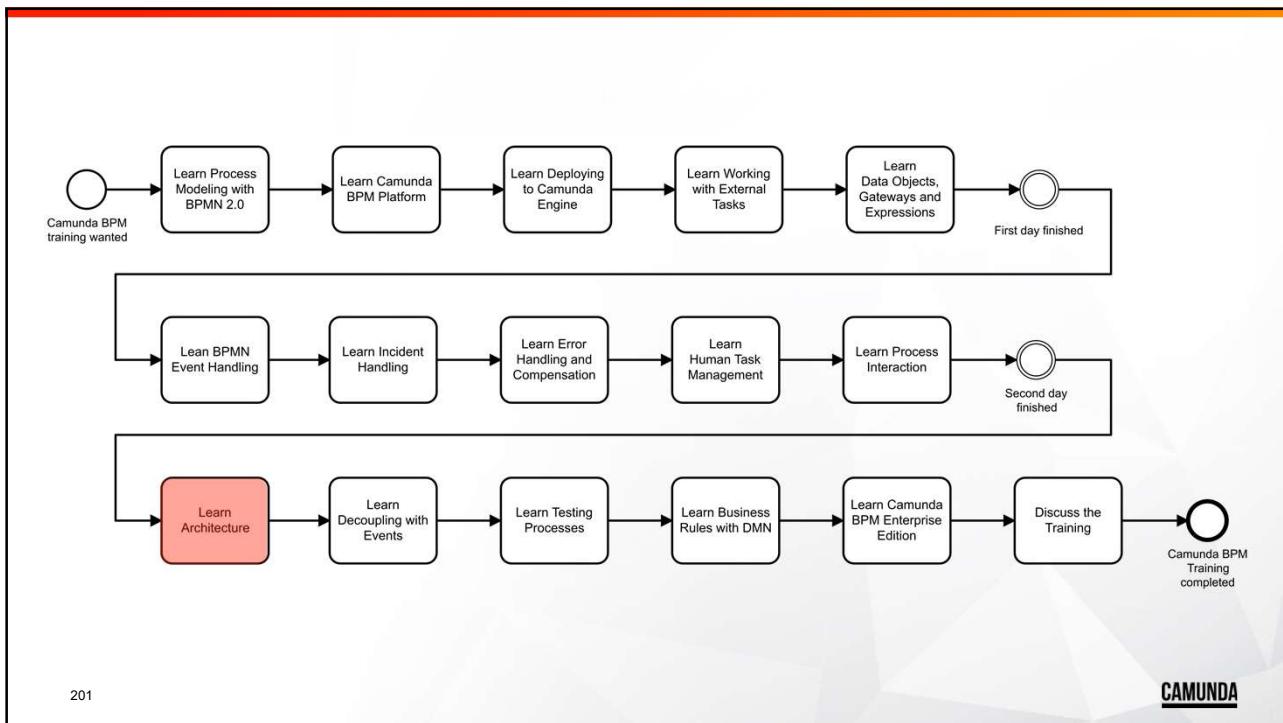
## Demo



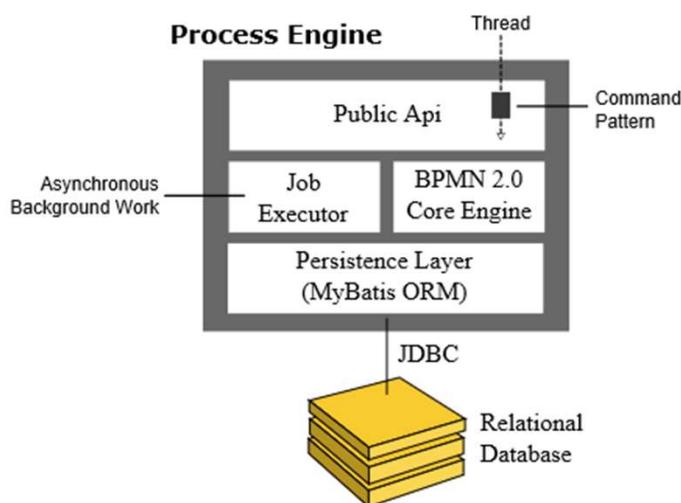
## Exercise 10

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks





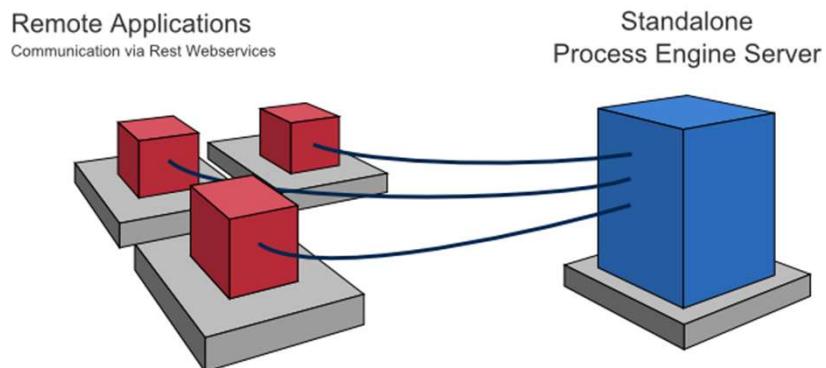
## Process Engine Architecture



202

CAMUNDA

## Standalone / Remote Process Engine Server



203

CAMUNDA

## Other Options for Java Programmers

Various environments possible:



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

204

CAMUNDA

## Process Deployment



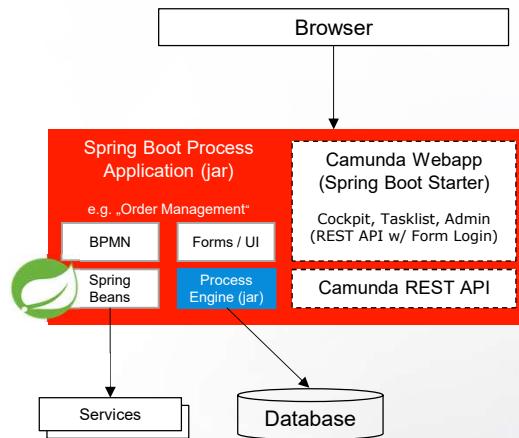
# CAMUNDA



205

CAMUNDA

## Spring Boot: Embedded Process Engine, Webapp & REST API



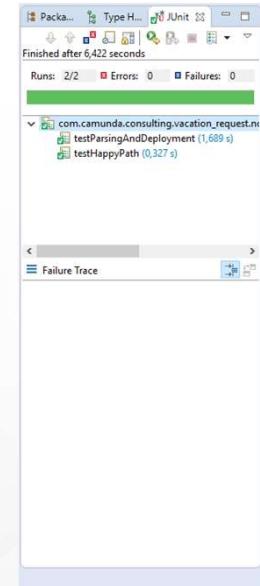
206

CAMUNDA

## 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();  
    }  
}
```

207



CAMUNDA

## 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/>

208

CAMUNDA

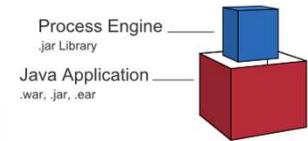
## One Process Application

My process application

Java classes  
Process Engine  
(jar)

BPMN 2.0  
processes

...



209

CAMUNDA

## Multiple Process Applications

My process application

Java  
classes  
Process Engine

BPMN 2.0  
processes

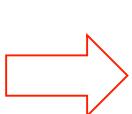
...

My process application2

Java  
classes  
Process Engine

BPMN 2.0  
processes

...



My process  
application

Java  
classes  
BPMN 2.0  
processes

Process Engine

My process  
application2

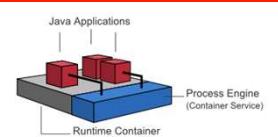
Java  
classes  
BPMN 2.0  
processes

JPA, EJB, JTA

...

Container

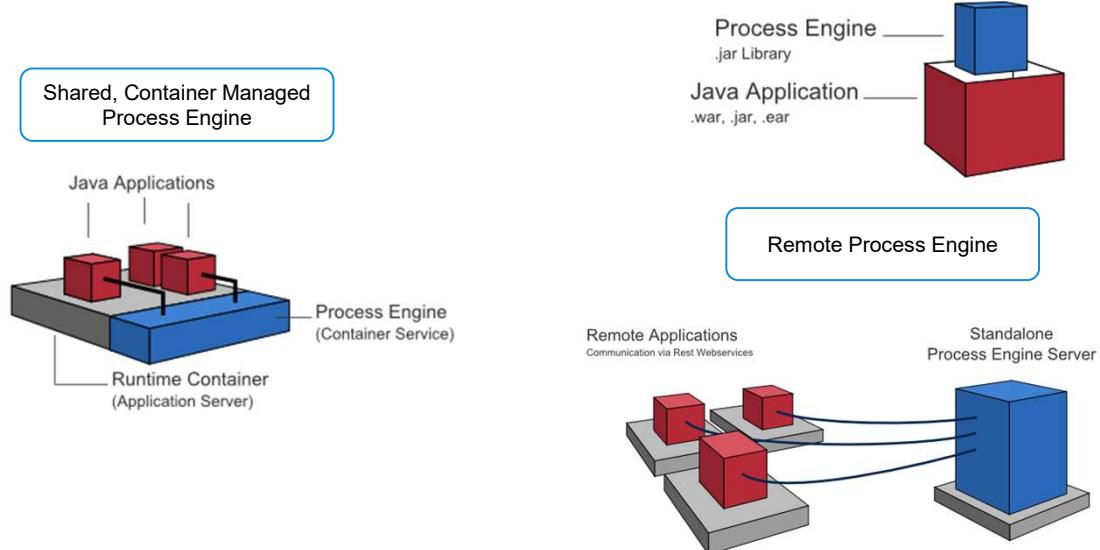
Supported on:  
Tomcat, JBoss AS/EAP, Wildfly AS,  
WebSphere AS and WebLogic



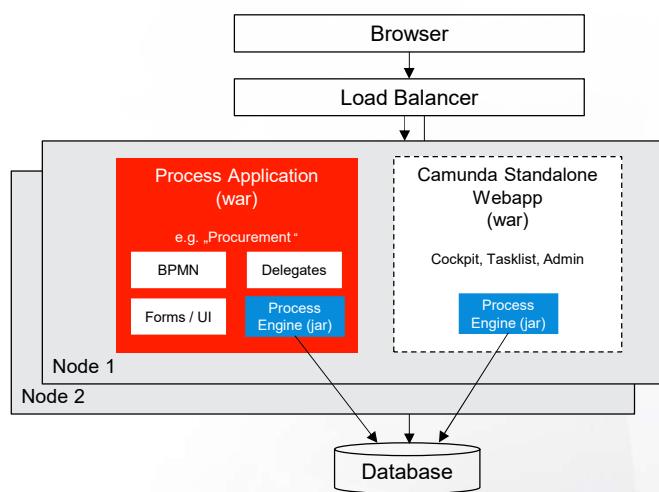
210

CAMUNDA

## Process Engine Modes



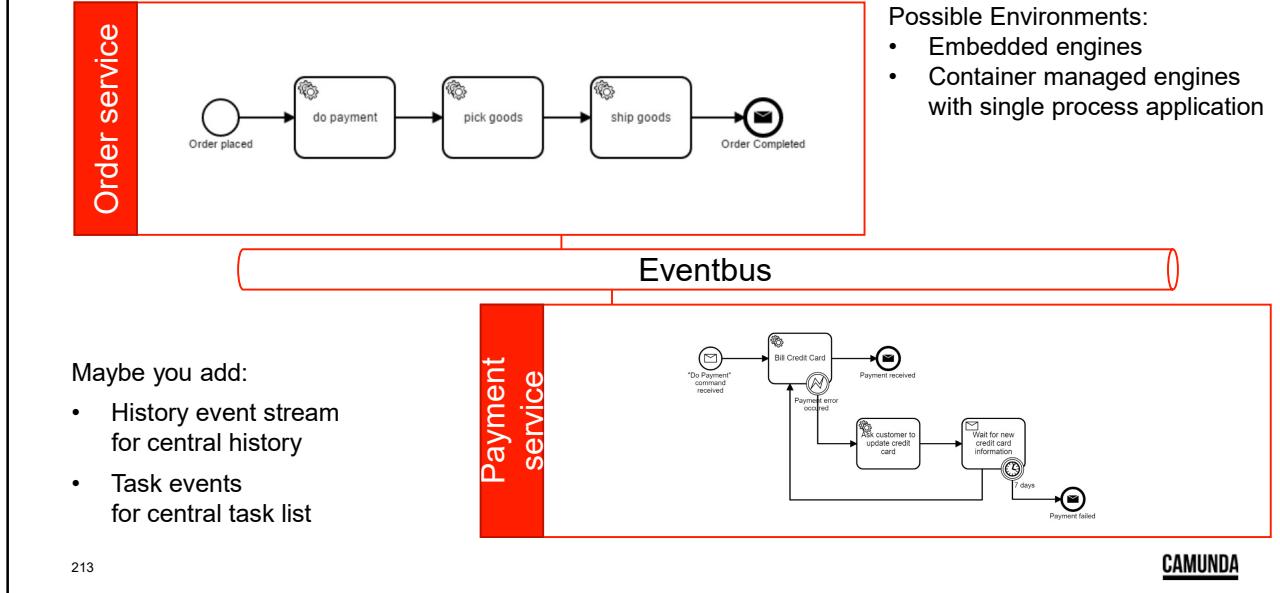
## Example Architecture With Embedded Process Engines



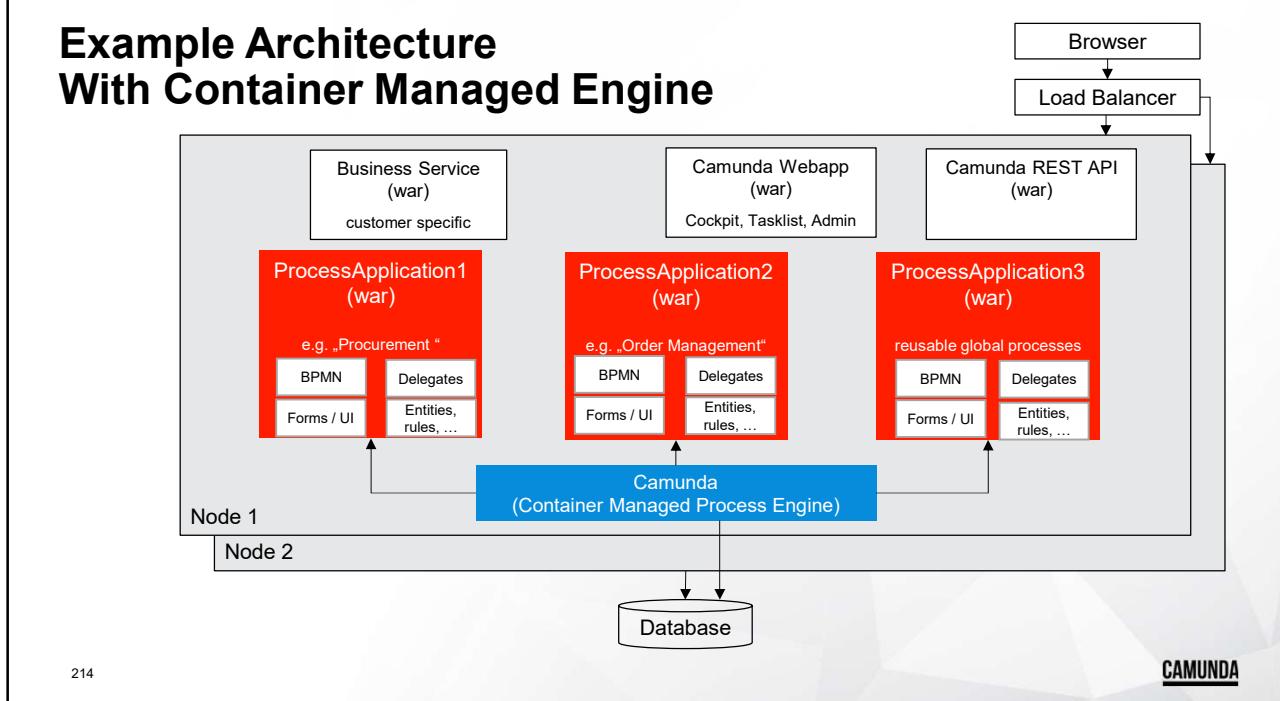
212

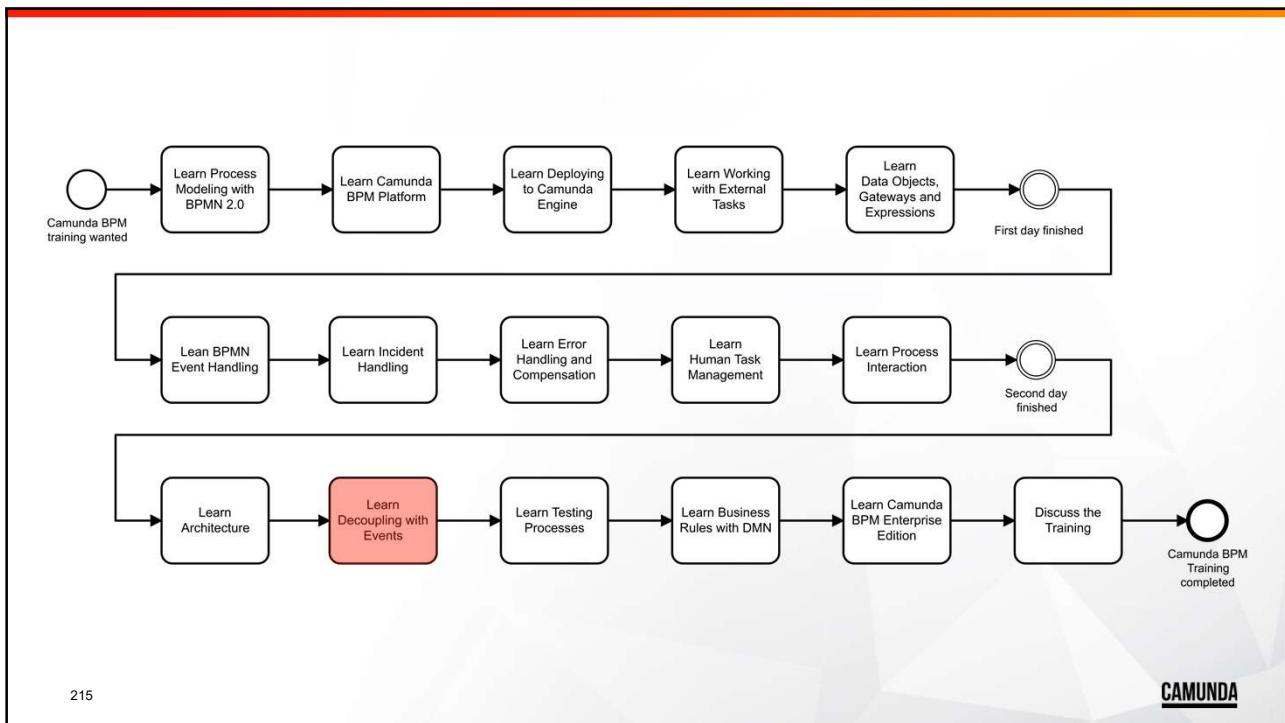
CAMUNDA

## Orchestration With Microservices

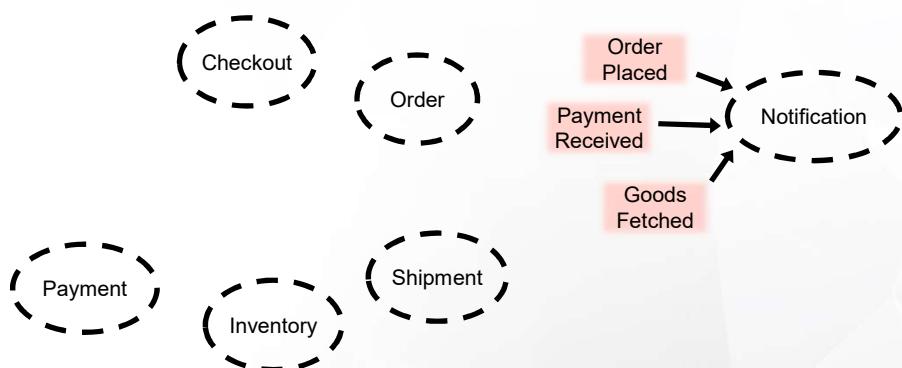


## Example Architecture With Container Managed Engine





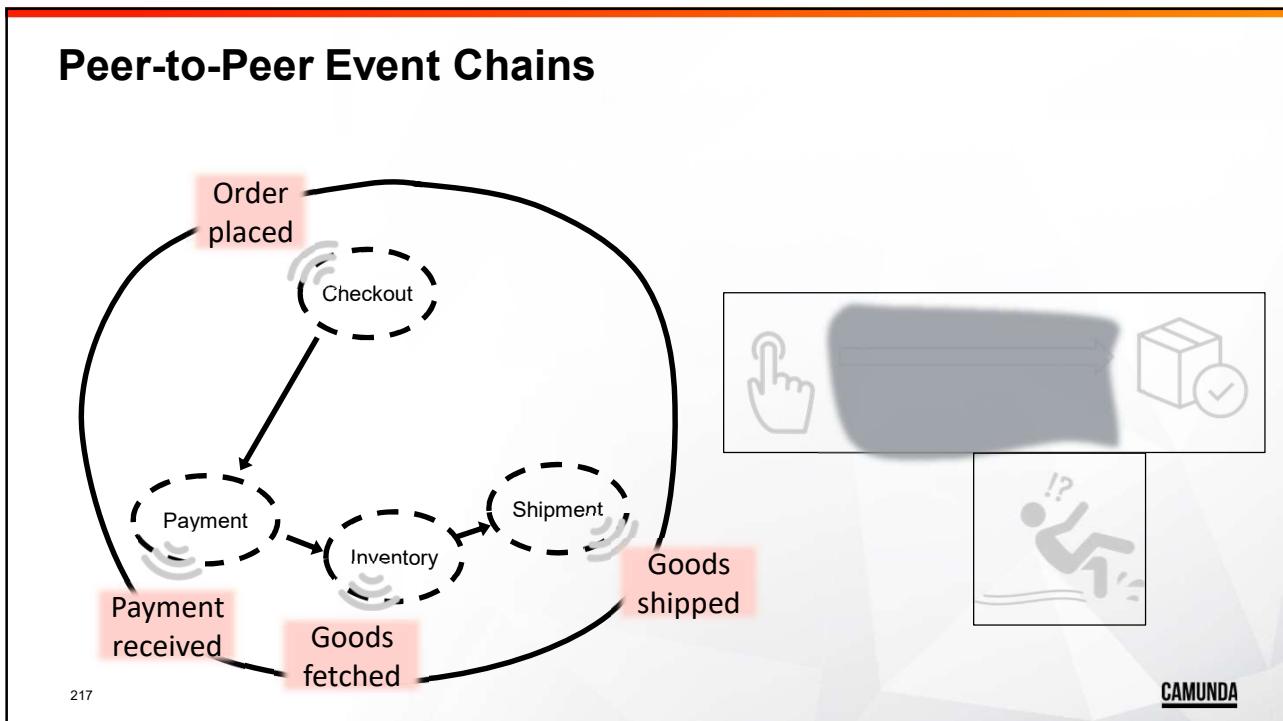
## Event-Driven Architecture



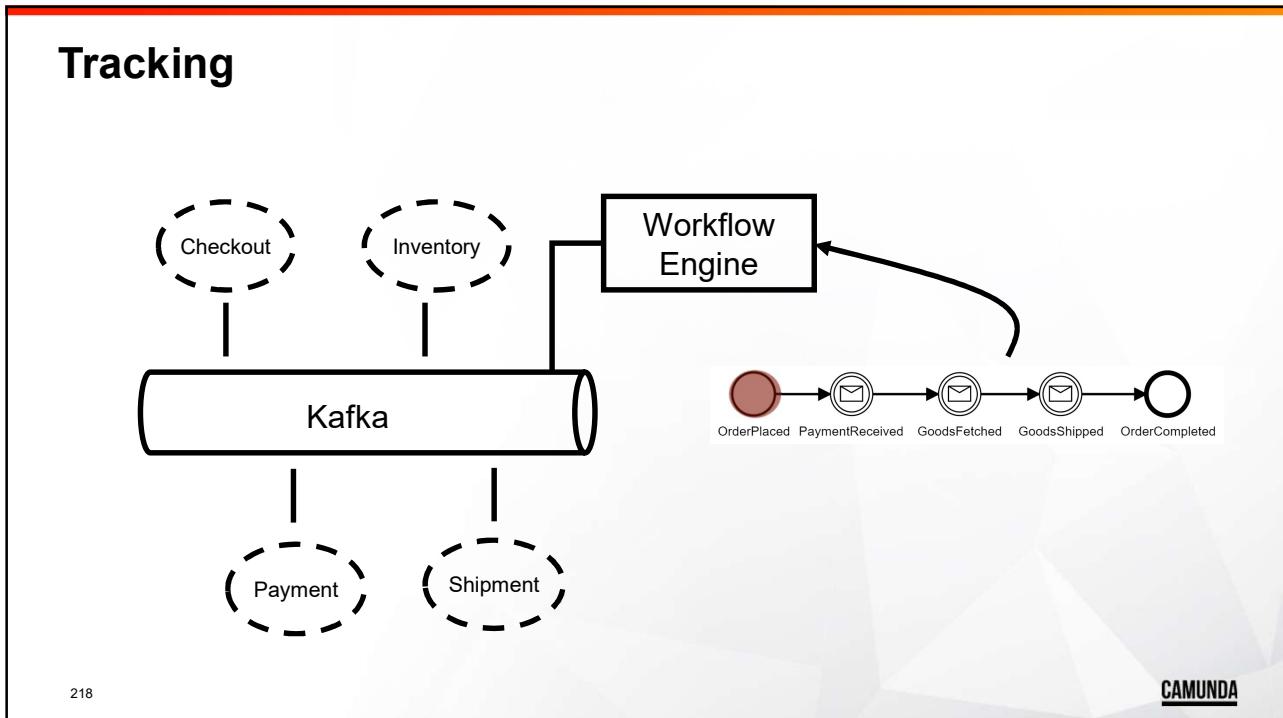
216

CAMUNDA

## Peer-to-Peer Event Chains



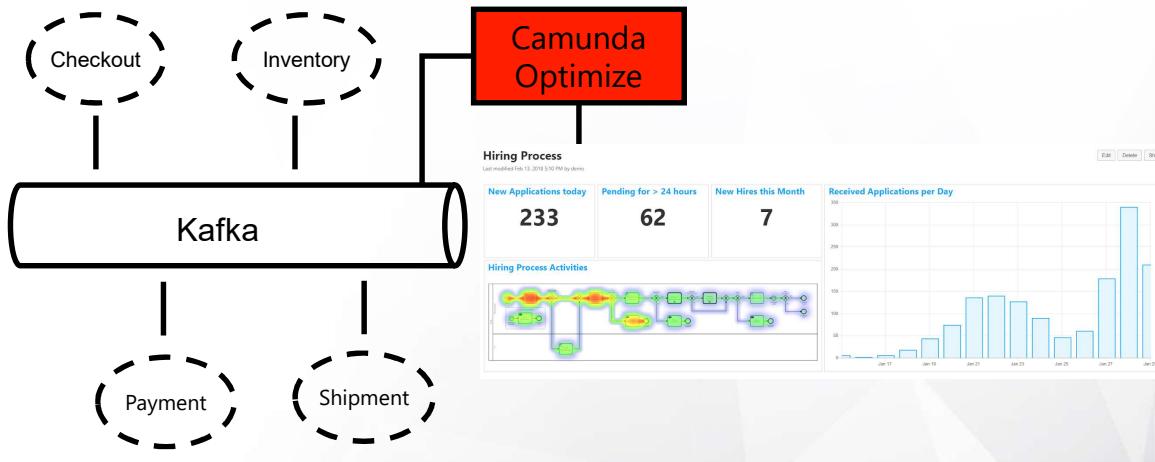
## Tracking



218

CAMUNDA

## Event Based Processes

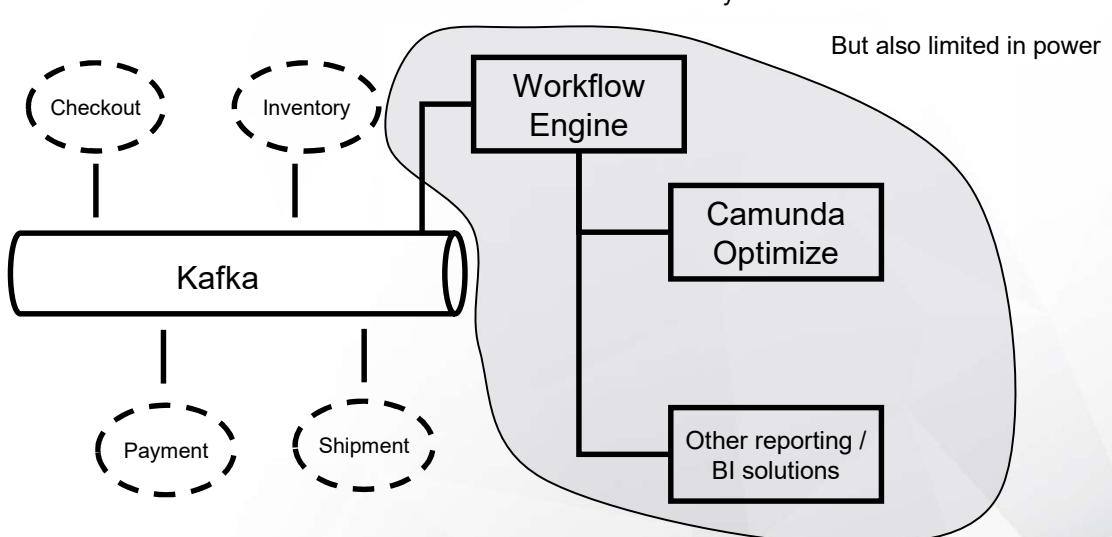


219

<https://docs.camunda.org/optimize/latest/user-guide/event-based-processes/>

CAMUNDA

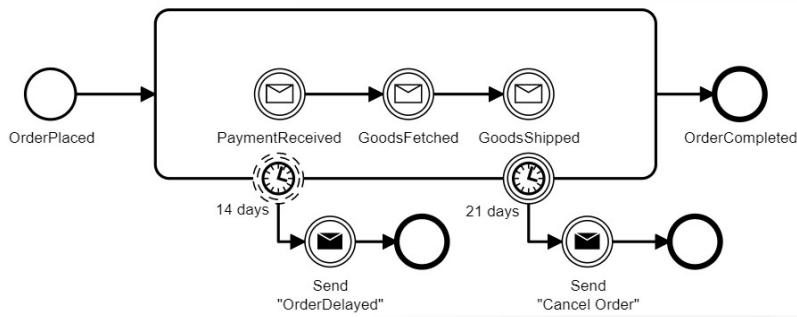
## Tracking + BI



220

CAMUNDA

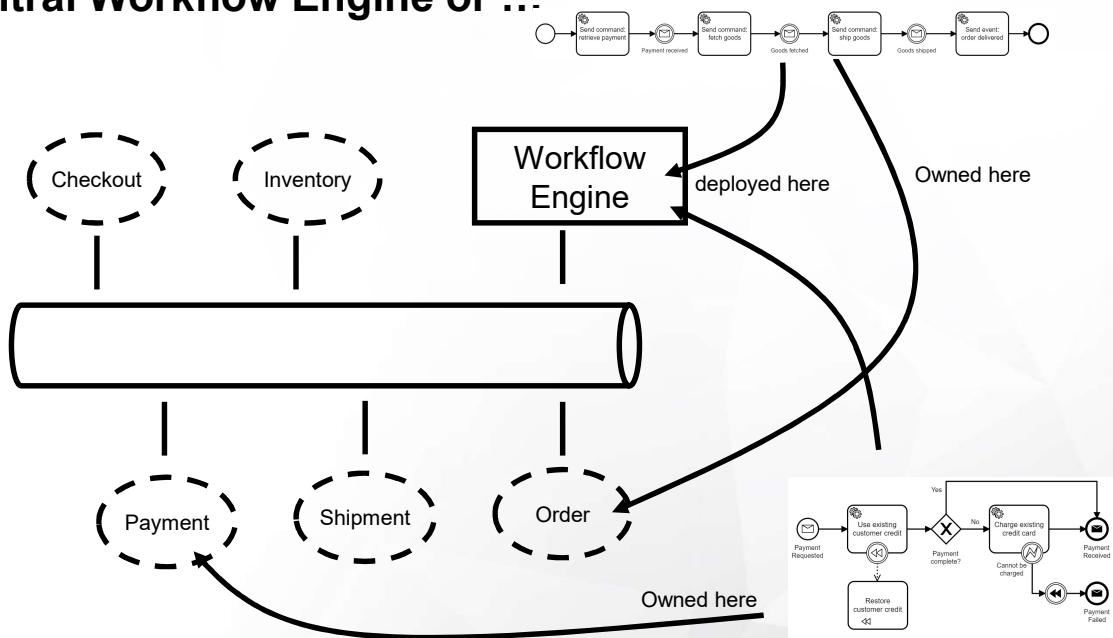
## Start Acting Upon Certain Events



221

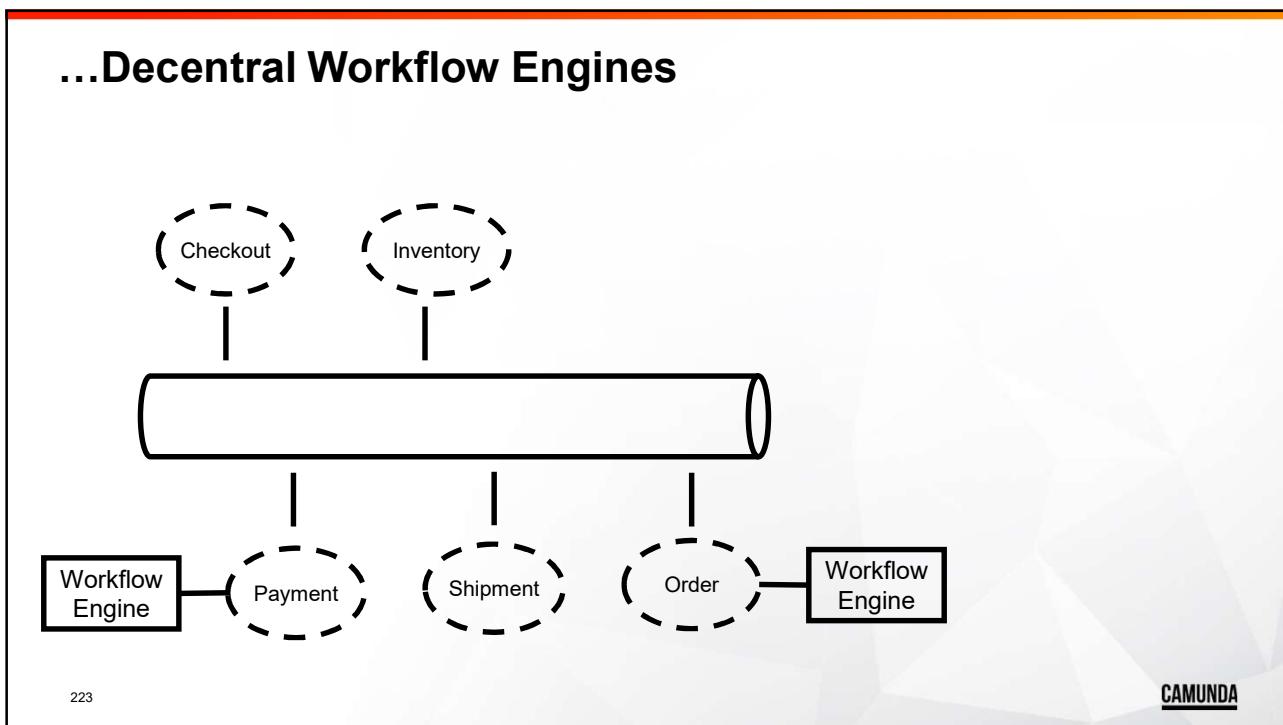
CAMUNDA

## Central Workflow Engine or ...



222

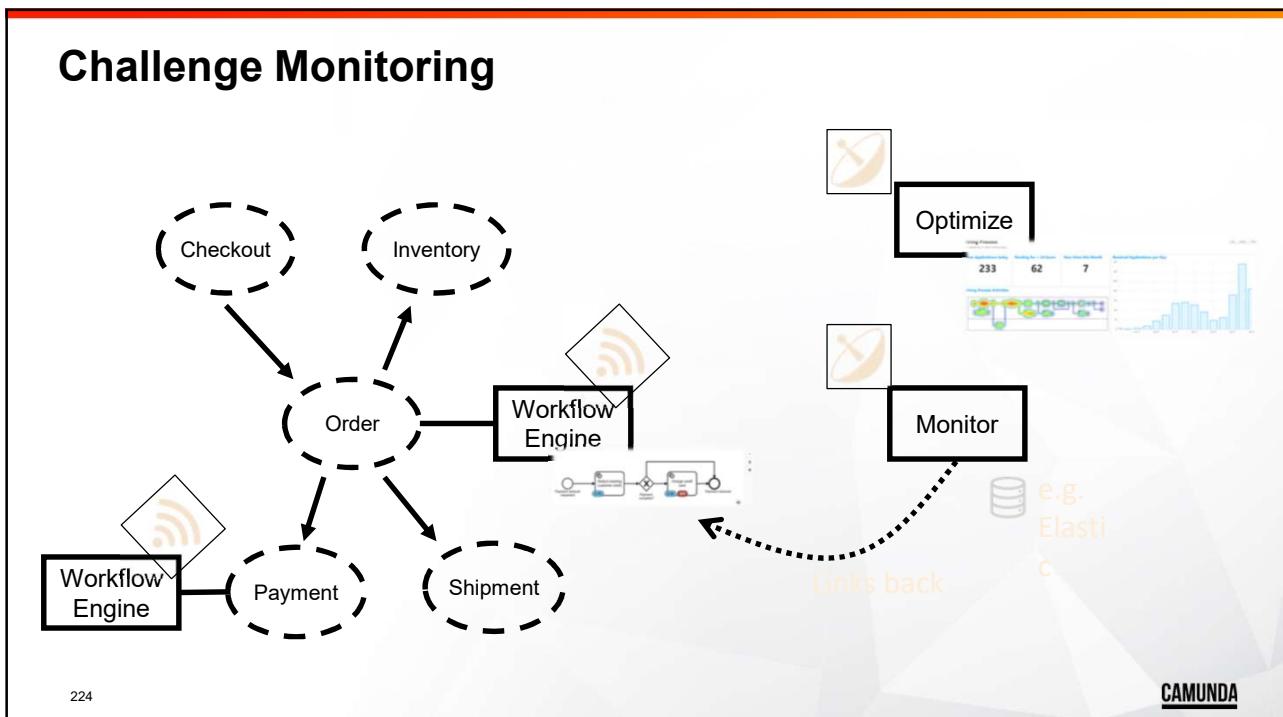
## ...Decentralized Workflow Engines



223

CAMUNDA

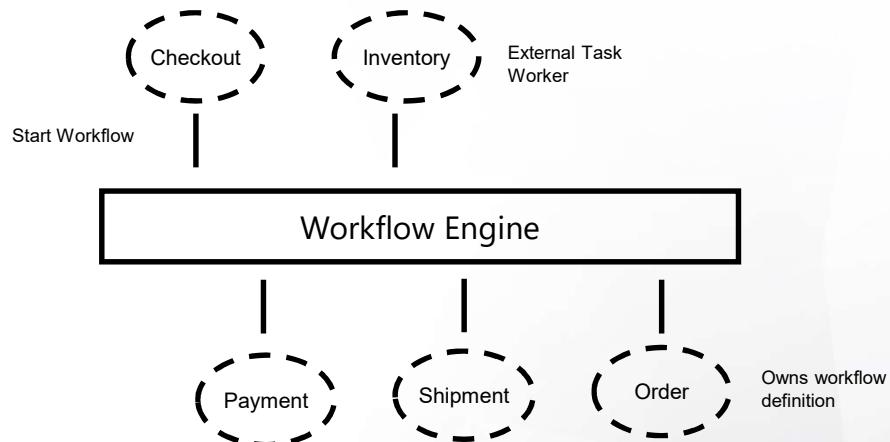
## Challenge Monitoring



224

CAMUNDA

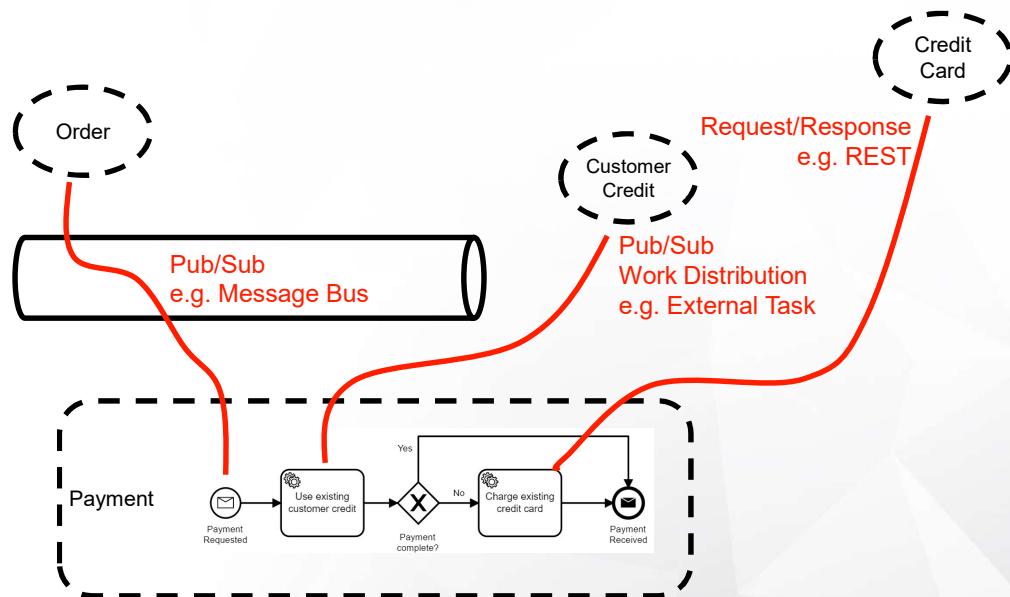
## Work Distribution via Workflow Engine



225

CAMUNDA

## Hybrid Architectures



226

CAMUNDA

## Recap

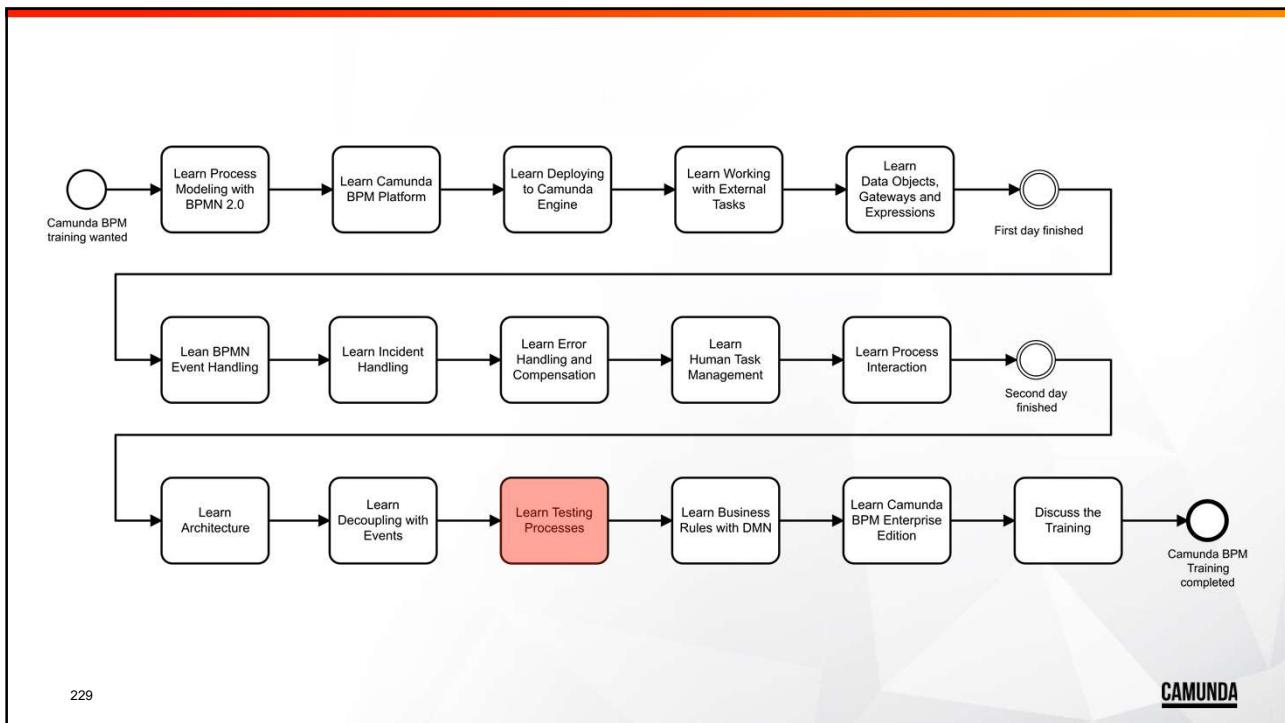
- Microservices have its merits
- You need to balance orchestration and choreography
- Distributed systems need stateful orchestration
- Visibility is essential to survive
- Track, monitor or manage the flow – depending on your scenario
- Workflow automation is an essential building block,
- make sure to use BPMN

227

CAMUNDA

Demo

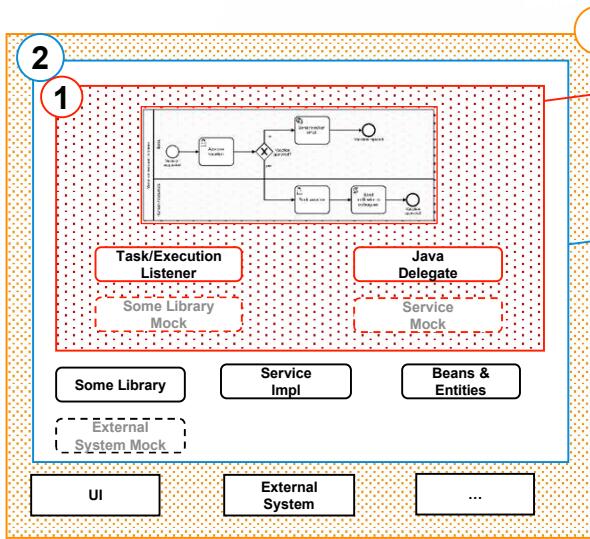




229

CAMUNDA

## Testing Scopes



Goal: Process Model with Data, EL & Adapter Logic

- In Memory
- Single Thread
- No JobExecutor

Goal: Close to Real-Life environment

- Container
- Arquillian & co

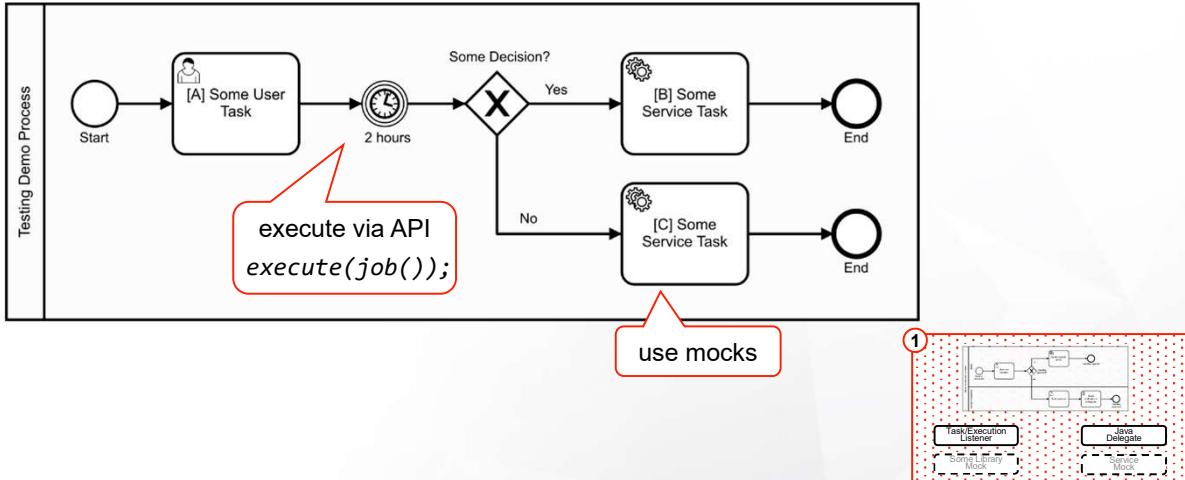
Goal: It REALLY works

- Integration Tests
- Human Driven

230

CAMUNDA

## Example



## Unit Test Support

- In memory database: url: jdbc:h2:mem:process-engine
- Use the REST API or Java API
- Reproduceable scripted tests
  - Postman collections (REST API)
  - Jest (Javascript, Node.js)
  - ... (your programming language)
  - JUnit + camunda-bpm-assert (Java)

## Postman Collection

- Set up an engine with
  - in-memory-database
  - Disable jobExecutor
- Run API calls
- Assert the results
- Transfer variables from response to further requests
- Wrap them in a deployment
- Use collection runner

233

CAMUNDA

Demo

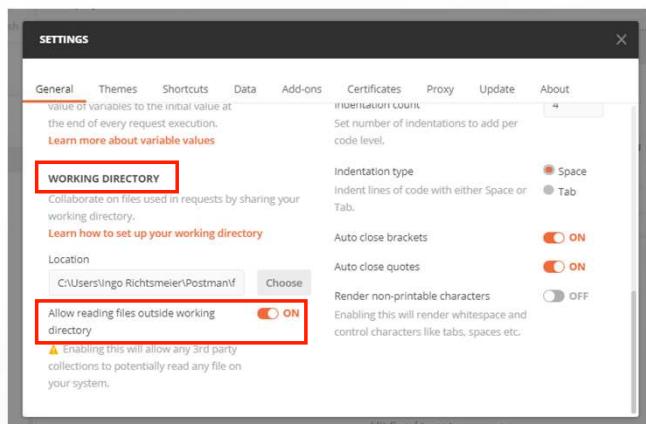
Testing with  
Postman  
Collection

## Settings for process deployment call

To avoid:

Form param `process-model` , file load error: PPERM:  
insecure file access outside working directory

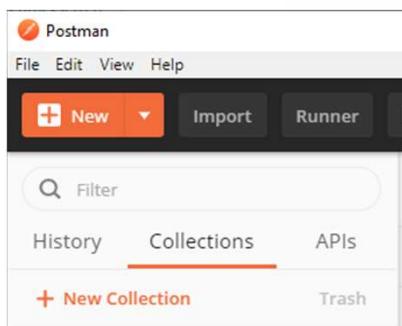
Check the settings:



235

CAMUNDA

## Import Collection



236

CAMUNDA

## Edit Requests

The screenshot shows the Postman application interface. On the left, there's a sidebar with a collection named "mircoservices-training-te..." containing 13 requests, including "deploy", "start process with message", "GET waiting at deduct credit", "fetch and lock", "complete deduct credit", "GET waiting at charge credit card", "fetch and lock", "complete charge credit card", "GET waiting at payment comple...", "fetch and lock", "complete payment comple...", and "process instance finished". The main panel shows a "GET waiting at deduct credit" request with the URL `http://localhost:8080/rest/external-task/processinstanceid={{processinstanceid}}`. The "Params" tab is selected, showing a single parameter `processinstanceid` with the value `{{processinstanceid}}`. There are tabs for Authorization, Headers, Body, Pre-request Script, Tests, Settings, Cookies, and Code. A "Send" button is visible at the top right. Below the URL, there's a "Query Params" table with one row: `processinstanceid` with value `{{processinstanceid}}`. A "Response" section is present but empty. At the bottom, there's a toolbar with icons for search, refresh, and other functions.

237

CAMUNDA

## Edit Assertions

The screenshot shows the Postman application interface. On the left, there's a sidebar with a collection named "mircoservices-training-te..." containing 13 requests, including "deploy", "start process with message", "GET waiting at deduct credit", "fetch and lock", "complete deduct credit", "GET waiting at charge credit card", "fetch and lock", "complete charge credit card", "GET waiting at payment comple...", "fetch and lock", "complete payment comple...", and "process instance finished". The main panel shows a "POST fetch and lock" request with the URL `http://localhost:8080/rest/external-task/fetchAndLock`. The "Tests" tab is selected, showing a JavaScript test script:

```
1 - pm.test("Status code is 200", function () {
2     pm.response.to.have.status(200);
3 });
4 - pm.test("external task id match", function () {
5     var externalTaskId = pm.environment.get("externalTaskId");
6     console.log("External task fetched: " + externalTaskId);
7     var jsonData = pm.response.json();
8     pm.expect(jsonData[0].id).to.eq(externalTaskId);
9});
```

Below the test script, there's a "Test scripts are written in JavaScript, and are run after the response is received." note and a "Learn more about tests scripts" link. To the right, there's a "SNIPPETS" section with links to "Get an environment variable", "Get a global variable", "Get a variable", "Set an environment variable", "Set a global variable", "Clear an environment variable", and "Clear a global variable".

238

CAMUNDA

## Transfer Results

The screenshot shows the Postman interface with a collection named "start process with message". A specific POST request is selected, which has the URL `http://localhost:8080/rest/message`. The "Tests" tab is active, displaying the following JavaScript code:

```
1 pm.test("Status code is 200", function () {
2   pm.response.to.have.status(200);
3 });
4
5 var jsonData = pm.response.json();
6 var processInstanceId = jsonData[0].processInstanceId;
7 console.log(processInstanceId);
8 pm.collectionVariables.set("processInstanceId", processInstanceId);
```

A tooltip is visible on the right side of the screen, providing information about test scripts. The tooltip includes links to learn more about test scripts, snippets, and various environment variable operations like getting, setting, and clearing both environment and global variables.

239

CAMUNDA

## Run Collection

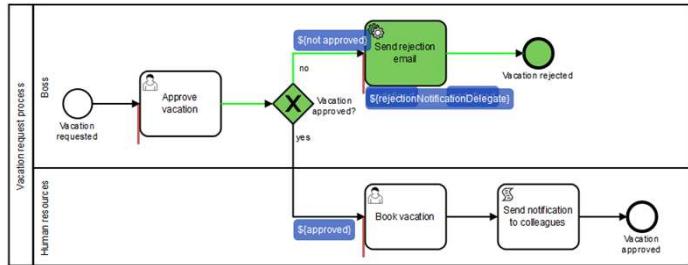
The screenshot shows the Camunda Collection Runner interface. A collection named "mircoservices-training-tests" is being run. The summary indicates 17 passed and 0 failed tests. The results are displayed in a table format under "Iteration 1", showing the following test details:

Test Step	Method	URL	Description	Status	Time	Size
1	POST	localhost:8080/rest/depl...	...lcoservices-training-tests / deploy	200 OK	194 ms	960 B
2	POST	localhost:8080/res...	...ttests / start process with message	200 OK	55 ms	401 B
3	GET	localhost:8080/res...	...ing-tests / waiting at deduct credit	200 OK	27 ms	742 B
4	assert external task credit deducting					
5	POST	localhost:8080/res...	...lces-training-tests / fetch and lock	200 OK	25 ms	889 B
6	external task id match					
7	POST	localhost:8080/rest/ext...	...ring-tests / complete deduct credit	204 No Content	30 ms	64 B
8	GET	localhost:8080/res...	...sts / waiting at charge credit card	200 OK	13 ms	746 B
9	assert external task Charge credit card					

240

CAMUNDA

## Real Unit Test



POST /process-definition/key/VacationRequestProcess/start

```
{"variables": {"from" : {"value" : "2019-12-20T00:00:00.000+0100", "type": "Date"},  
    "until" : {"value" : "2019-12-23T23:59:59.000+0100", "type": "Date"}},  
"startInstructions" :  
[  
    {  
        "type": "startAfterActivity",  
        "activityId": "approveVacationUserTask",  
        "variables": {  
            "approved": {"value": false, "local": false, "type": "Boolean"}  
        }  
    }]  
}
```

241

Process Instance  
Modification

CAMUNDA

## Unit Testing in Java



```
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-bpmn-assert

242

CAMUNDA

## Mock the Worker

- `assertThat(processInstance).externalTask("myServiceTask").hasTopic("expectedTopic");`
- `complete(externalTask(), withVariables("name1", "value1", "name2", 123));`

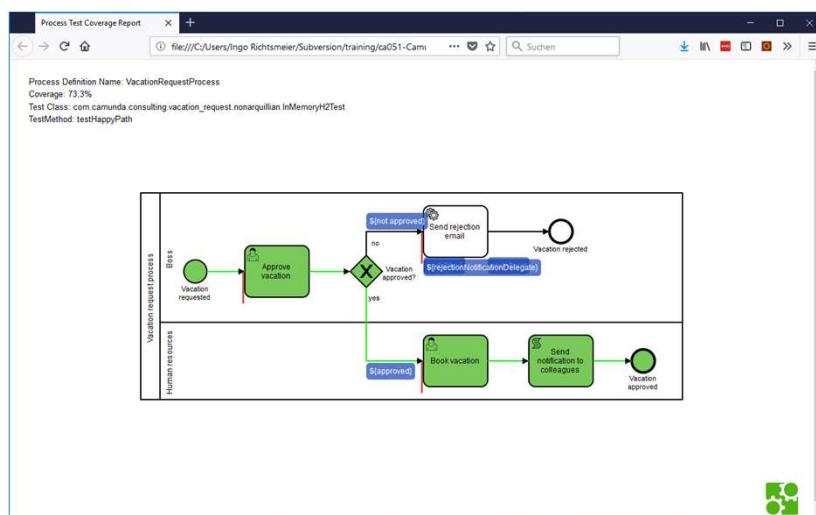
<https://docs.camunda.org/manual/latest/user-guide/testing/#camunda-assertions>

[https://github.com/camunda/camunda-bpm-assert/blob/master/docs/User\\_Guide\\_BPMN.md](https://github.com/camunda/camunda-bpm-assert/blob/master/docs/User_Guide_BPMN.md)

243

CAMUNDA

## Benefit: Process Test Coverage



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

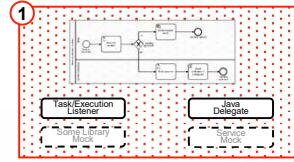
244

CAMUNDA

## Testing on Scope 1 for Java Developers

### We recommend

- JUnit
- Camunda Rule (included in camunda-engine)
- Camunda-bpm-assert (<https://github.com/camunda/camunda-bpm-assert>)
- Mock-Provider of your choice, e.g. Mockito or EasyMock or Jmockit (maybe + PowerMock)
- 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”!

245

CAMUNDA

## Testing on Scope 2: Integration Tests

Five microservice testing strategies for startups

1. The documentation-first strategy
2. The full stack in-a-box strategy
3. The AWS testing strategy
4. The shared testing instances strategy
5. The stubbed services strategy

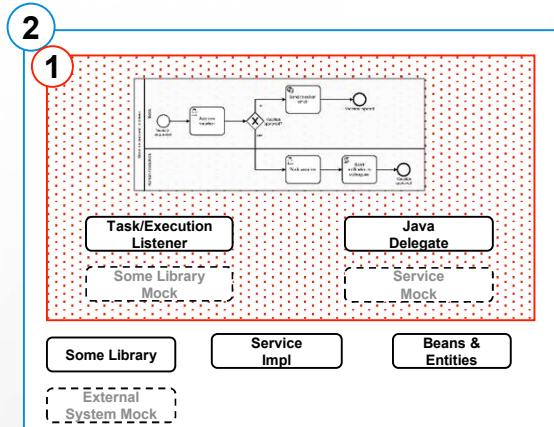
<https://medium.freecodecamp.org/these-are-the-most-effective-microservice-testing-strategies-according-to-the-experts-6fb584f2edde>

246

CAMUNDA

## Testing on Scope 2 with Java

- 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



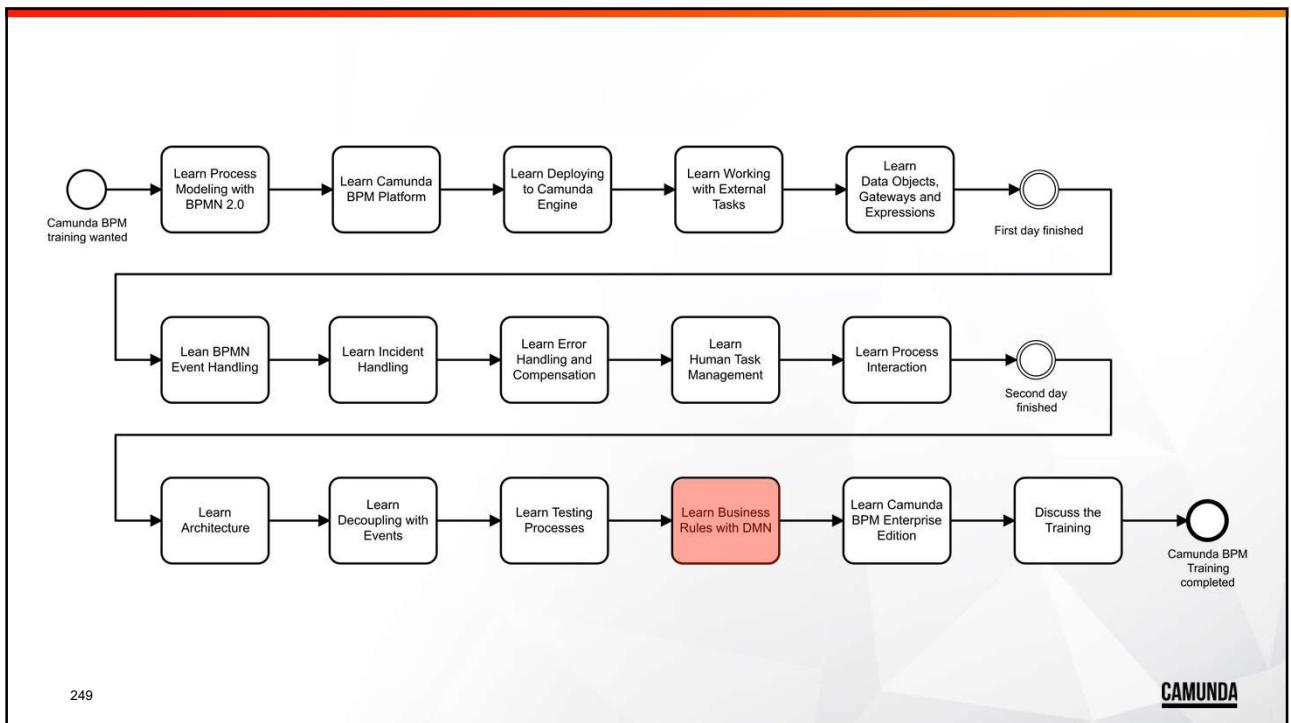
247

CAMUNDA

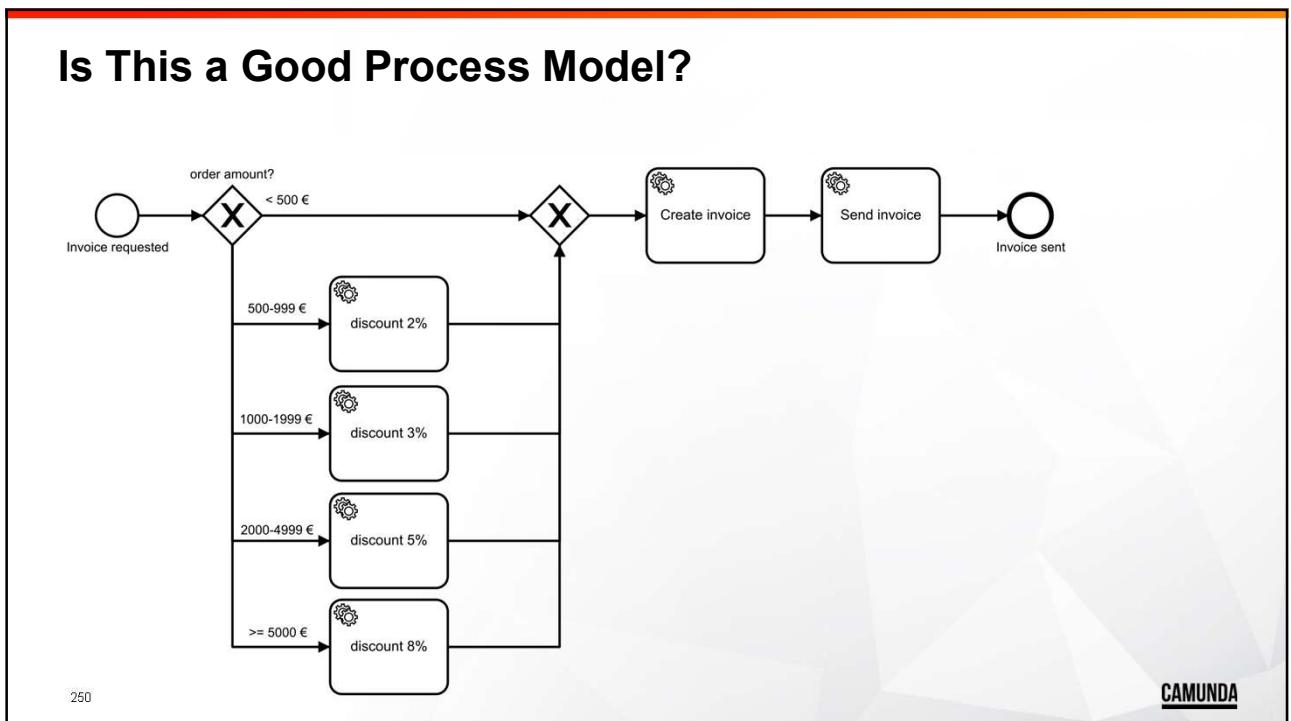
## Exercise 11

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks

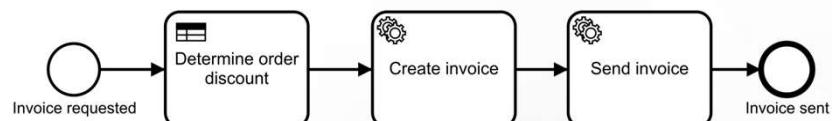




## Is This a Good Process Model?



## An Improved Version



251

CAMUNDA

## Discount Rules as DMN Decision Table

The screenshot shows the Camunda Modeler interface with a DMN decision table titled "Order Discount". The table has a "Hit Policy" set to "Unique". It contains five rows of rules, each with a condition column ("When") and an annotation column ("Then").

When	Then	Annotations
Order Amount <500	Discount 0	
[500..999]	2	
[1000..1999]	3	
[2000..4999]	5	
>=5000	8	
+ -		

Below the table, there are tabs for "Diagram" and "XML", and a "Log" button.

252

CAMUNDA

## The Standard DMN

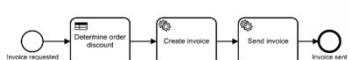
- Decision Model and Notation
- Is currently in publications by OMG for Decisions/Business Rules
- Specification: „The primary goal of DMN is to provide a **common notation** that is readily understandable by all **business users**, from the business analysts needing to create initial decision requirements 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.”

253

CAMUNDA

## Typical Frequency of Changes

Business Processes



Changes of Process Model causes effort for system integration or even organizational change

Less frequent (months or years)

Rule Structure (Columns)

Order Discount	With Price	Unique	Annotations
1.00%	Discount	1.00%	
2.00%	Discount	2.00%	
3.00%	Discount	3.00%	
4.00%	Discount	4.00%	
5.00%	Discount	5.00%	
6.00%	Discount	6.00%	
7.00%	Discount	7.00%	
8.00%	Discount	8.00%	

Change of the structure causes effort, as you may need additional input parameters

Less frequent (months)

Rules (Rows)

Order Discount	With Price	Unique	Annotations
1.00%	Discount	1.00%	
2.00%	Discount	2.00%	
3.00%	Discount	3.00%	
4.00%	Discount	4.00%	
5.00%	Discount	5.00%	
6.00%	Discount	6.00%	
7.00%	Discount	7.00%	
8.00%	Discount	8.00%	

Changes of rules (rows in table) are very easy to do, but think about validation & testing

Very frequent (weeks)

254

CAMUNDA

## Rules Expressed as Decision Table

The diagram shows a decision table titled "Dish" with a "Hit Policy: Unique" dropdown set to "Unique". The table has columns for "When" (Season), "And" (Vegetarian Guests), and "Then" (Dish). The "Annotations" column contains optional remarks like "Hey, why not?!".

- Decision name:** A red box points to the title "Dish".
- Hit Policy „Unique“:** A red box points to the dropdown menu.
- Input expression:** A red box points to the "When" column header.
- Each row = single rule:** A red box points to the first row of the table.
- Input entry:** A red box points to the value "Fall" in the "Season" column of the first row.
- Output entry = Result of rule:** A red box points to the value "Spareribs" in the "Dish" column of the first row.
- Optional remarks.** Typically used for motivation of rule: A red box points to the annotation "Hey, why not?!".

255

CAMUNDA

## Multiple Inputs

The diagram shows a decision table titled "Dish" with a "Hit Policy: Unique" dropdown set to "Unique". The table has columns for "When" (Season), "And" (Vegetarian Guests), and "Then" (Dish). The "Annotations" column contains optional remarks like "Hey, why not?!".

- Input entries can have various data formats:** A red box points to the "Vegetarian Guests" column header, which is of type "boolean".
- Multiple inputs always follow „AND“ logic:** A red box points to the fifth row, where the "Vegetarian Guests" value is "true".

256

CAMUNDA

## Technical Details (1)

The screenshot shows the Camunda Decision Requirements Diagram (DRD) editor. A red box highlights the 'Input parameter' section on the left, which contains a table with columns 'Season' and 'Expression'. A red arrow points from this box to a red box labeled 'Type definition' at the bottom right of the main table area. Another red arrow points from the 'Input parameter' box to a red box labeled 'Decision Id' at the top right of the main table area. The main table has columns 'And' and 'Then'. The 'And' column contains conditions like ' $\leq 8$ ', ' $\leq 8$ ', ' $\leq 4$ ', '[5..8]', and ' $> 8$ '. The 'Then' column contains corresponding dish names: 'Spareribs', 'Roastbeef', 'Dry Aged Gourmet Steak', 'Steak', 'Stew', and 'Light Salad and a nice Steak'. Annotations are provided for some entries: 'Save money' for 'Dry Aged Gourmet Steak', 'Less effort' for 'Stew', and 'Hey, why not?' for 'Light Salad and a nice Steak'. The bottom right corner of the editor window features the 'CAMUNDA' logo.

257

CAMUNDA

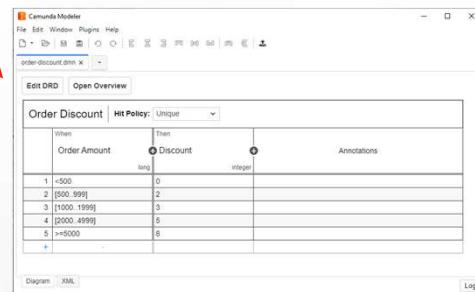
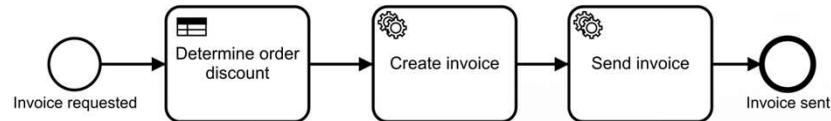
## Technical Details (2)

The screenshot shows the Camunda Modeler interface. A red box highlights the 'decision ID' field in the Properties Panel on the right, which contains the value 'dish-decision'. A red arrow points from this box to a red box labeled 'decision ID' at the bottom right of the panel. The central workspace shows a decision element named 'Dish' with a dashed border. The bottom left of the workspace has tabs for 'Diagram' and 'XML'. The bottom right corner of the interface features the 'CAMUNDA' logo.

258

CAMUNDA

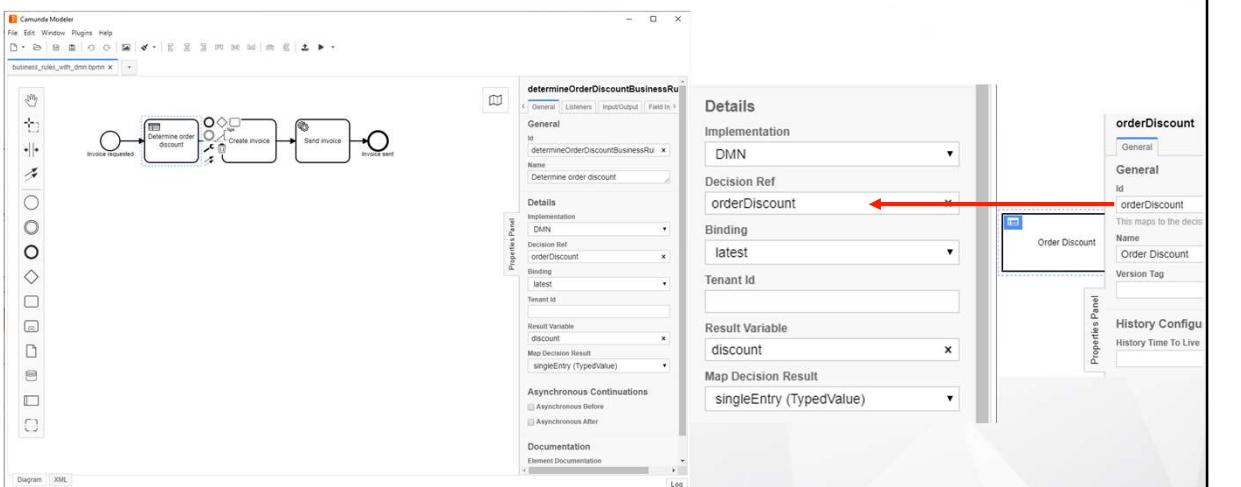
## Business Rule Task Reference a DMN Decision Table



259

CAMUNDA

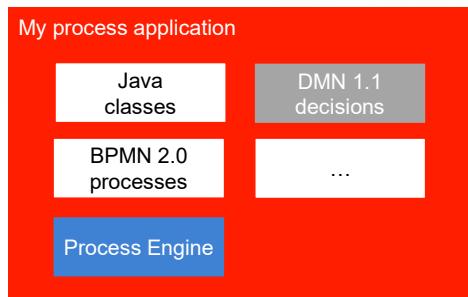
## Reference in Camunda Modeler



260

CAMUNDA

## DMN Is Part of the BPM Platform

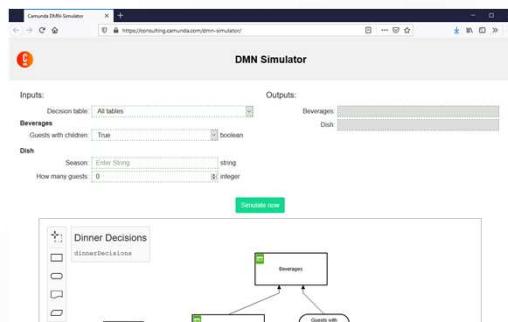


261

CAMUNDA

## Standalone Stateless Decision Engine

- You can use the core DMN engine as stateless Rule Engine separately from the BPM Platform if you like
- Available as library
- Example:  
DMN Simulator: <https://consulting.camunda.com/dmn-simulator/>



262

CAMUNDA

## 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 }  
    }  
]
```

263

CAMUNDA

## 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*

264

CAMUNDA

## 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	< 25	good	Low
5		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

265

CAMUNDA

## Map the Result to the Process

Decision 1 | Hit Policy: Unique

	When Input 1	And Input 2	Then Output 1	And Output 2
	string	string	string	string
1	"value 1"	-	"result 1"	"reason 1"
2	"value 2"	"value 3"	"result 2"	"reason 2"
3	"value 4"	-	"result 3"	"reason 3"
4	"value 5"	"value 7"	"result 4"	"reason 4"
+	-	-		

Map Decision Result

singleEntry (TypedValue)

singleEntry (TypedValue)

singleResult (Map<String, Object>)

collectEntries (List<Object>)

resultList (List<Map<String, Object>>)

266

CAMUNDA

## 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



267

CAMUNDA

## Friendly Enough Expression Language (FEEL)

[date and time(„2016-12-24T00:00:00“)  
..  
date and time(„2016-12-26T23:59:99“)]

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

Simple Expressions		Hit Points	Annotations			
When	And	Date	Number of guests	Children	Dish	string
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” Same as (6..10)
5	“Winter”	-	[10..12]	-	-	=“Huge Pizza”
6	“Winter”	-	12,[13..14]>14	-	-	=“Family Deluxe Pizza”
7	“Spring”, “Summer”, “Fall”	-	-	true	-	=“Salad”
8	not(“Winter”)	-	-	false	-	=“Salad with Goat Cheese”
9	not(“Spring”, “Summer”, “Fall”)	-	-	-	-	=“Vegetables” It is Winter again :-)
+/-	-	-	-	-	-	-

Negation

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

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

268

CAMUNDA

## FEEL Functions

- Full support for FEEL (since 7.13)
- Built-in functions (see <https://camunda.github.io/feel-scala/>):
- Example expressions and functions:

```
contains("foobar", "of")  
  
applicant.monthly.income * 12  
  
if applicant.maritalStatus in ("M","S") then "valid" else "not valid"  
  
sum( [applicant.monthly.repayments, applicant.monthly.expenses] )  
  
sum( credit_history[record_date > date("2011-01-01")].weight )  
  
some ch in credit_history satisfies ch.event = "bankruptcy"
```

<https://github.com/camunda/feel-scala>

269

CAMUNDA

## Scripting as Expression Language

The screenshot shows a Camunda BPMN editor interface. A table titled 'Tweet approval' is displayed under the 'Hit Policy: First' dropdown. The table has three columns: 'When' (labeled 'Tweet Conditions'), 'Then' (labeled 'Annotations'), and an empty column. There are four rows in the table:

When	Then	Annotations
1 lastTweet.content == currentTweet.content	false	You cannot tweet the same content twice
2 currentTweet.content.contains('#ibm')	false	Please do not tweet about competition
3 currentTweet.content.matches('someRegEx')	-	rejection reason found

A modal dialog titled 'Expression Language' is open over the third row. It contains a dropdown menu with the following options: FEEL, JUEL, JavaScript, Groovy, Python, and JRuby. The input field in the dropdown shows the prefix 'juel|'. A tooltip 'rejection reason found' is visible near the bottom right of the dialog.

270

CAMUNDA

## Cockpit for Decisions

The screenshot shows the Camunda Cockpit interface for a Risk Check decision. The top navigation bar includes links for Processes, Decisions, Cases, Human Tasks, and More. A user profile for Paula Pepperman is visible. The main content area displays the 'Risk Check' decision table with the key 'checkRisk\_en'. The table has columns for Input (Age, Employment, Category, Score) and Output (Risk, Risk Level). Rows define rules based on age and employment status, such as 'Unemployed' leading to 'Won't Pay In Time' with a risk level of 'red'. Below the table, there are sections for 'Inputs' and 'Outputs' with their corresponding values. The bottom right corner indicates the cockpit is 'Powered by camunda BPM / v7.13.0-ee'.

271

CAMUNDA

## Live Editing of Decision Tables

The screenshot shows the Camunda Cockpit interface for editing the 'Risk Check' decision table. A red banner in the top right corner says 'Enterprise Feature'. The 'Edit DMN' section allows users to choose a DMN file or upload one. The decision table 'checkRisk\_en' is displayed with its columns: Input (Age, Employment, Category, Score) and Output (Risk, Risk Level). The table rows show the same logic as the cockpit view. At the bottom, there's a 'Definitions' section listing the decision key 'Risk Check' and version '1'. The bottom right corner indicates the cockpit is 'Powered by camunda BPM / v7.13.0-ee'.

272

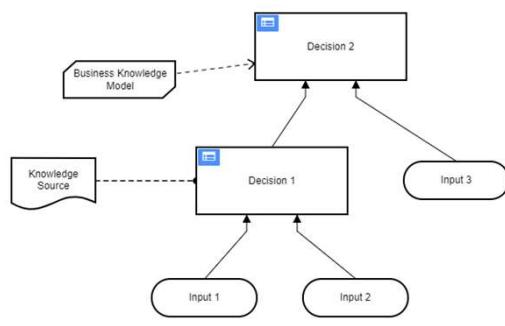
CAMUNDA

## 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

273

CAMUNDA

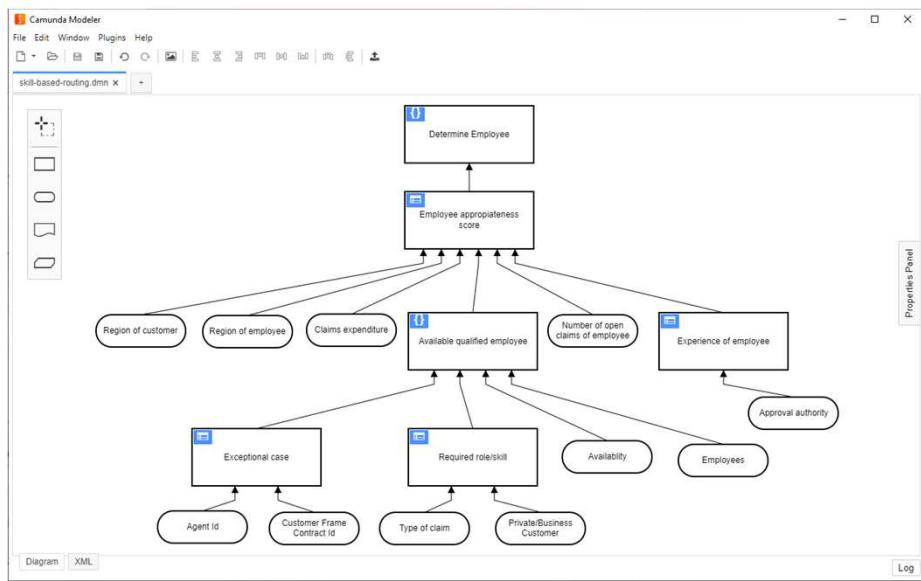


## DRD Example: Skill Based Routing

274

Log

CAMUNDA



## Decisions Reference Decision Tables

The screenshot shows the Camunda Modeler interface with a decision model. Red arrows highlight specific parts of the model:

- A red arrow points from the "Employee appropriateness score" decision table to the "Employee appropriateness score" decision element in the process diagram.
- Another red arrow points from the "Experience of employee" decision table to the "Experience of employee" condition in the "Required role/skill" decision table.
- A third red arrow points from the "Exceptional case" decision table to the "Exceptional case" decision element in the process diagram.

**Employee appropriateness score**

When	And	And	And	Then
Region of employee = Region of customer boolean	Experience of employee "low", "medium"	Claims expenditure integer	Number of open claims of employee integer	100 -1000
1 true	-	-	-	100
2 -	"low"	[1000, 5000]	-	-1000
3 -	"low"	> 5000	-	-1000
4 -	"medium"	> 10000	-	-100
5 -	"medium"	-	[10, 19]	-100
6 -	"medium"	-	[20, 29]	-500
7 -	"medium"	-	>= 30	-1000
+	-	-	-	-

**Experience of employee**

When	Then
Approval Authority integer	Experience "low", "medium", "high"
1 < 1000	"low"
2 [1000, 10000]	"medium"
3 > 10000	"high"
+	-

**Required role/skill**

When	And	Then	And
Type of claim "Third Party Liability", "Accident"	Private/Business Customer "Private", "Business"	Required Role "Service Center", "Business Acc. string	Required Skill
1 "Third Party Liability"	"Private"	"Service Center"	-
2 "Third Party Liability"	"Business"	"Service Center"	"Business Law Qualification"
3 "Accident"	"Private"	"Service Center"	-
4 "Accident"	"Business"	"Business Accident Team"	-
+	-	-	-

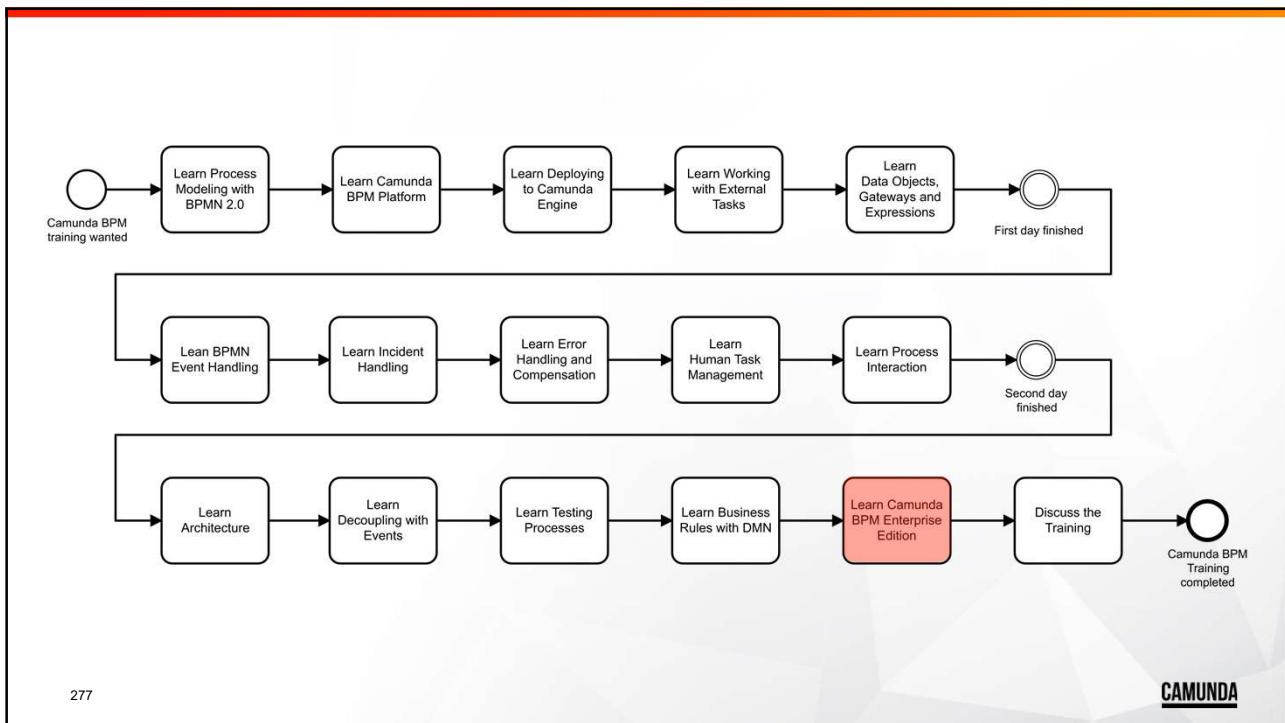
275

CAMUNDA

## Exercise 12

Add Decision Logic about Discount

<https://training.camunda.com/microservices>  
user: microservices  
password: camundarocks



## 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

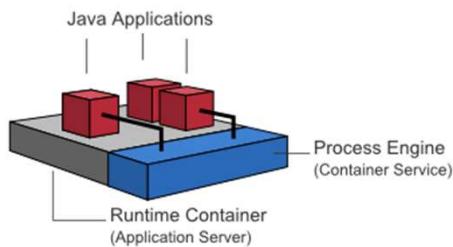
<https://camunda.com/bpm/enterprise/#cockpit>

278

**CAMUNDA**

## Runtime Support

- Integration with IBM WAS
- Integration with Oracle WLS



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

279

CAMUNDA

## SLA Based Support

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

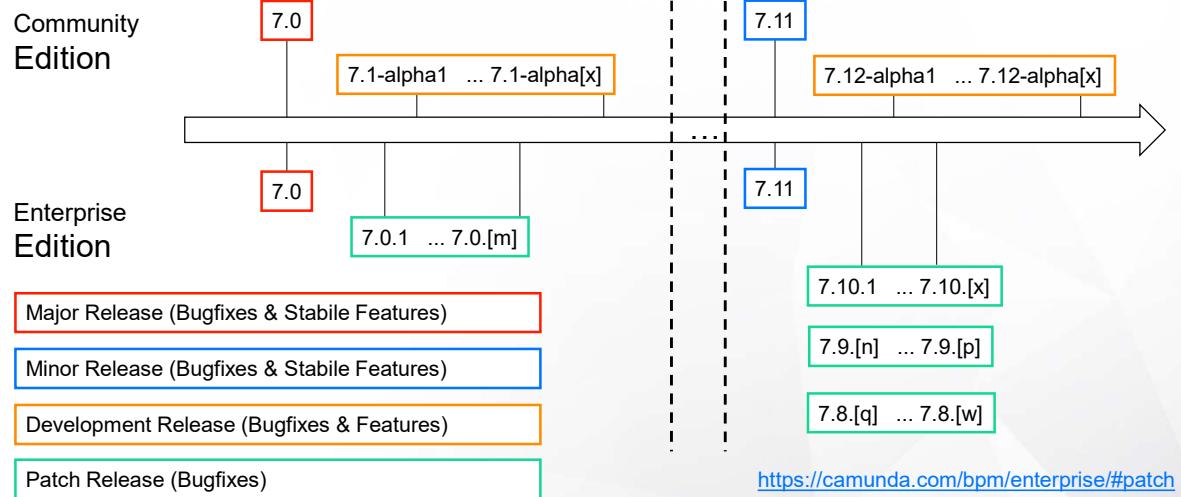
Level	Name	Description	Availability		Max. response time	
			Standard SLA	Advanced SLA	Standard SLA	Advanced SLA
L1	Blocker	Core components (i.e., process engine) of Camunda BPM do not work at all/produce critical errors that prevent usage in production mode.	8x5	24x7	8 business hours	2 hours
L2	Critical	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>

280

CAMUNDA

## Patch Releases



## Please Plan your Updates

- Check patch releases: Bugs that bother you?
- Minor release:
  - 6 Months before End of Support you will receive a mail
  - Check if you run the version mentioned here
  - If yes, start planning your update

## Optimize

The screenshot shows the Camunda Optimize interface with a dark header bar. The main title is "Hiring Dashboard" and it was last modified on Feb 1, 2019 at 1:02 AM by admin. The dashboard features several key metrics and visualizations:

- Total Applications 2018:** 4,743
- New Hires 2018:** 39
- New Applications per Day 2018:** A line chart showing daily application counts from January 1 to December 31, 2018.
- Most executed steps:** A heatmap visualization showing the frequency of execution for various process steps.
- Average Duration per Step:** A BPMN diagram illustrating the duration of different steps in the process.

At the bottom right, there is a copyright notice: "© Camunda Services GmbH 2019. All Rights Reserved. | 2.3.0".

283

CAMUNDA

## References

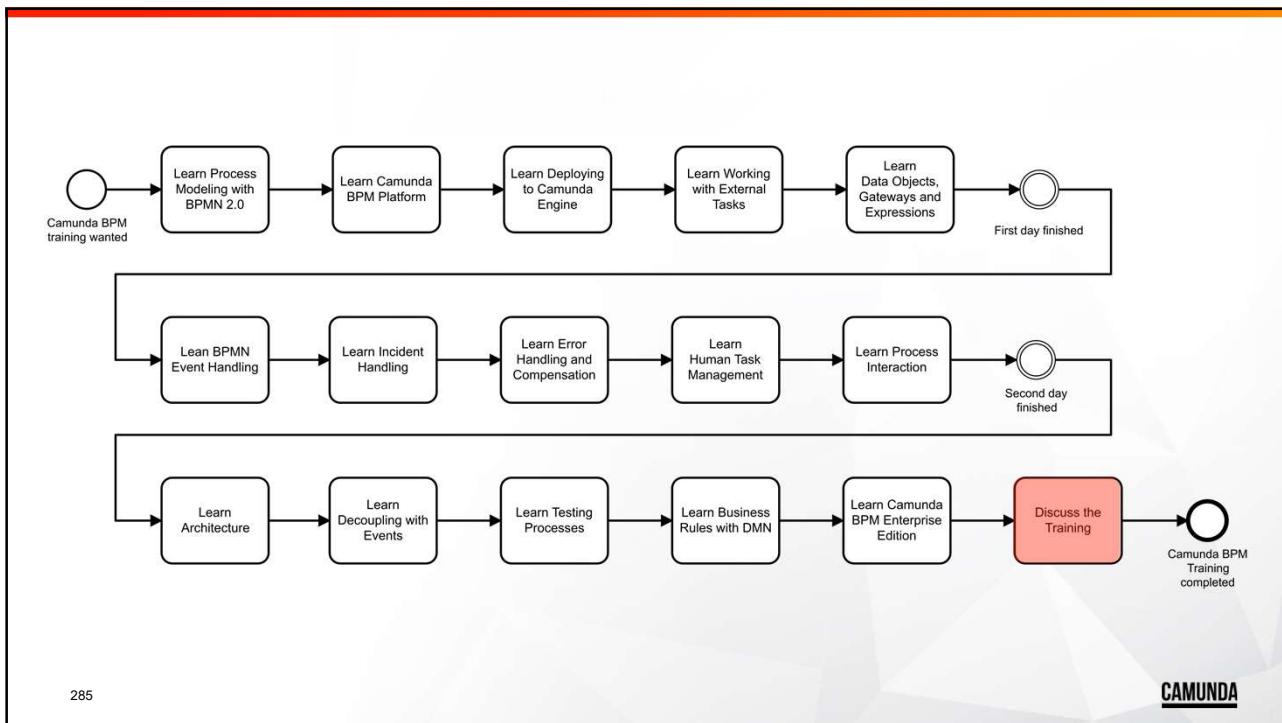
The screenshot shows the Camunda website's case studies section. The header includes the Camunda logo and navigation links for Products, Solutions, Services, Customers, and Learn. The main content area is titled "Camunda Case Studies" and features a sub-headline: "Organizations all over the world are using Camunda to automate their core business processes. Here are some examples." Below this, there are three cards for featured case studies:

- Deutsche Telekom:** Transforming from a monolithic application to an agile, microservices-based architecture. [Learn more](#)
- 24 Hour Fitness:** Executing more than 20M BPMN workflow instances and 18M DMN decision instances per day. [Learn more](#)
- What users are saying about Camunda:** A portrait of a man with glasses and a play button icon.

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

284

CAMUNDA



285

CAMUNDA

## How to Get Updates From the Team?

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



### Whitepapers:

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

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

286

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

Topic	Replies	Views	Activity
DMN make decision to assign task for candidate group	2	25	1h
Multiple use of Modeler	4	6	1h
Multiple use of embedded subprocess	6	14	2h
Pass custom header to service task	9	25	2h
Spring Boot Starter and Postgres	2	7	3h
SAP HCI / CPI Cloud Platform Integration support	9	133	3h
Whole process due date	19	25	4h
Dependency from multi instance call activity to main process	1	8	4h
Unable to use assertThat in Test class to assert BPMN process	5	353	4h
Storing run-time audit information	3	21	5h



**Follow us on Twitter:**  
@camundabpm

CAMUNDA

# Camunda Training Offering

## Overview trainings

### 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 DMN

### DMN (1 day)

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

## 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

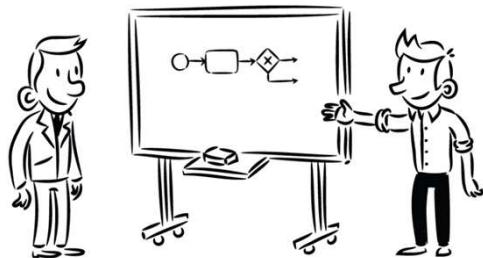
You'll find further information under  
[camunda.com/de/services/training](http://camunda.com/de/services/training)

287

CAMUNDA

# Want to Learn More BPMN?

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



CAMUNDA

## Feedback

Please fill out this form:

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

The screenshot shows a web browser window titled "Training Feedback | Camunda". The URL in the address bar is "camunda.com/services/training/feedback/". The page has a red header with the "CAMUNDA" logo and navigation links for Products, Solutions, Services, Customers, Learn, Contact, Download, DE, and EN. Below the header, there is a sidebar with links for Support, Consulting, and Training. The main content area contains the following fields:

- \* 1. Trainer: A dropdown menu.
- \* 2. Training / Workshop: A dropdown menu.
- \* 3. Type of training (delivery mode): A dropdown menu.
- \* 4. Date of training:  
Date / Time  
Date:
- \* 5. Your role: A dropdown menu.
- 6. Training Management: A horizontal scale with four options: Excellent, Good, Not Great, and Terrible. The "Good" option is highlighted.

289

CAMUNDA