# Connectors unleashed

## Crafting future-proof low-code solutions

CAMUNDA CON 2024

Thomas Heinrichs - *BPM Craftsman, Miragon*
Calvin Robbins - *Product Manager, Camunda*

# Your Hosts

**Thomas Heinrichs**
BPM Consultant
Augsburg, Germany

MIRAGON

**Calvin Robbins**
Senior Manager of Product Management
Exeter, UK

CAMUNDA

# Agenda

- Getting you ready for the workshop
- What are Connectors and why should you bother about them?
- Best practices on using and building Connectors
- Follow-along workshop:
  - Using domain specific Connectors in combination with the REST Connector
  - Differentiating between domain and integrative processes
  - Implementing a hexagonal domain Connector

# Getting you ready for the workshop

## What do you need to follow along?

- Access our Camunda 8 organisation, by adding your mail to the Form

For following the “pro-code” approach to connectors:

- An IDE of your choice
- Java or Node.js runtime
- Access to GitHub

Form for C8 access

Github

# What are Connectors and why should you bother about them?

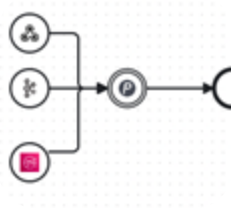
# Why bother with Low-Code?

- **Shorter development cycles:** Low-Code platforms facilitate fast development, allowing businesses to quickly build scalable projects.
- **Democratize Software Development:** Low-code technology democratizes the process of software development, making it accessible to a wider range of people.
- **Overcoming Developer Shortage:** The limited resources are always the developers. Low-code and no-code technologies enable organizations of any size to address the shortage of developers.
- **Better Communication:** Model-based development makes it possible to have an abstraction to the code and thus ensures a common language that both business and IT can understand.

# The Problem Connectors solve

- Most use cases, Processes needs to connect to outside systems to orchestrate work
- Additional effort and cost for customers to implement and maintain connectivity
- Camunda solely providing out-of-the-box connectivity would be limited

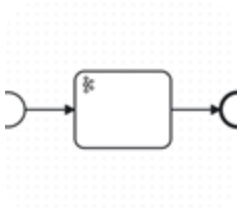
# Out of the Box Connectors



Inbound

Enable workflows to **receive data** or messages from **external systems** or **services**

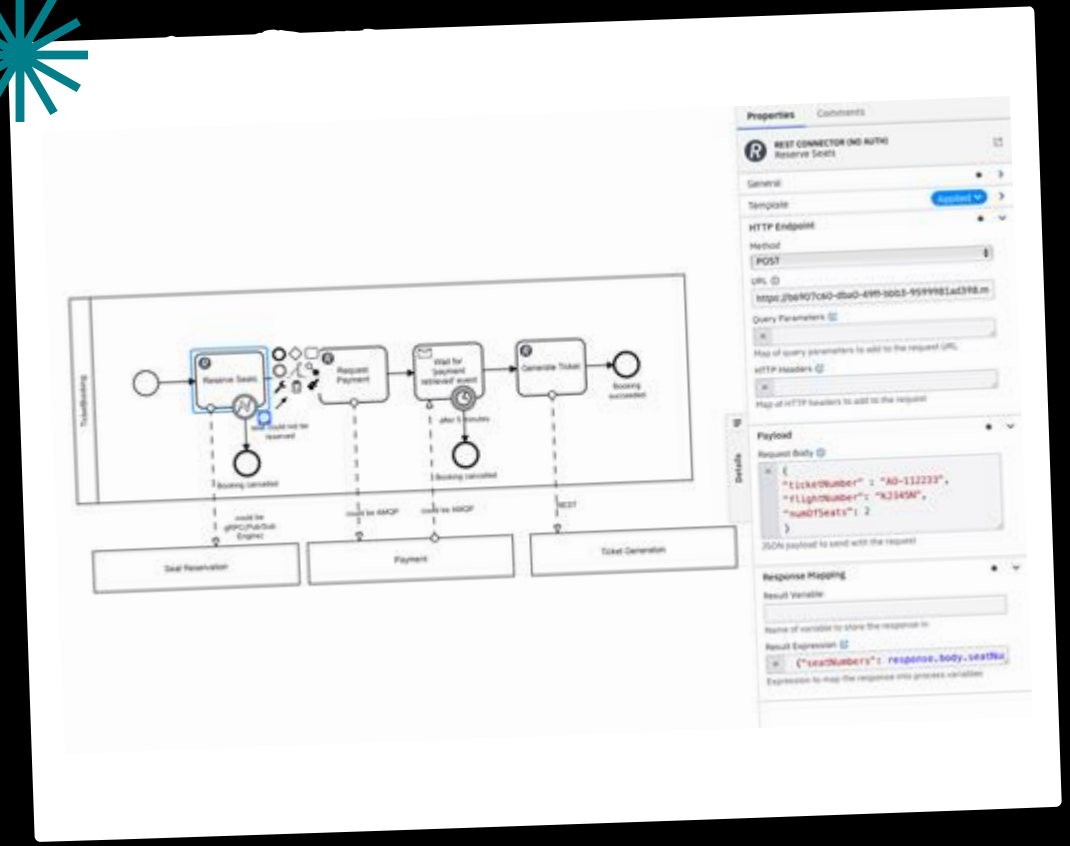
Integrates workflows into a wider business process or system architecture



Outbound

Enables workflows to **trigger** with **external systems** or **services**

Integrates workflows with other parts of a business process or system architecture




# Out of the Box Connectors Usage in Camunda 8 SaaS

- **Accelerate solution** implementation by providing pre-built, ready-to-use connectors.
- **Simplify integration** with external systems, reducing the need for coding expertise and lowering the barrier to entry for less technical users.
- **Eliminate concerns** about connector configuration details, such as API endpoints, authentication, HTTP methods, and headers.

# Connector *x* Job Worker

## What's the difference?

### JOB WORKER

- A job worker is usually part of a Zeebe Client application that can be directly executed to work on jobs.
- Every job worker implementation defines on its own how to handle input data, validating and transforming it
- Deals with environment tasks like handling variables in and out

### CONNECTOR

- A Connector is reusable code, and it is environment agnostic.
- It is not a standalone application; you cannot start it.
- Is, a library and can be used in the Connector runtime environment.
- Has secret management capabilities
- Focusses on core business functionality

# The Connector SDK

The Connector SDK allows you to develop custom Connectors using Java code.

The SDK provides APIs for common Connector operations, such as:

- Fetching and deserializing input data
- Validating input data
- Replacing secrets in input data

Additionally, the SDK allows testing of your Connector behaviour.

```
<dependency>
  <groupId>io.camunda.connector</groupId>
  <artifactId>connector-core</artifactId>
  <version>0.2.2</version>
  <scope>provided</scope>
</dependency>
```

```
@OutboundConnector(
    name = "PING",
    inputVariables = {"caller"},
    type = "io.camunda.example.PingConnector:1"
)
public class PingConnector implements OutboundConnectorFunction {

  @Override
  public Object execute(OutboundConnectorContext context) throws Exception {}
}

```

# Connector Templates

1

## What is a Connector Template?

- A Connector template is a pre-configured JSON file in Camunda 8 that defines the appearance and behaviour of connectors in Modeler.

2

## Pre-configured Parameters

- Connector templates allow designers to preset input parameters, control their visibility, and restrict value options, ensuring consistency across different processes.

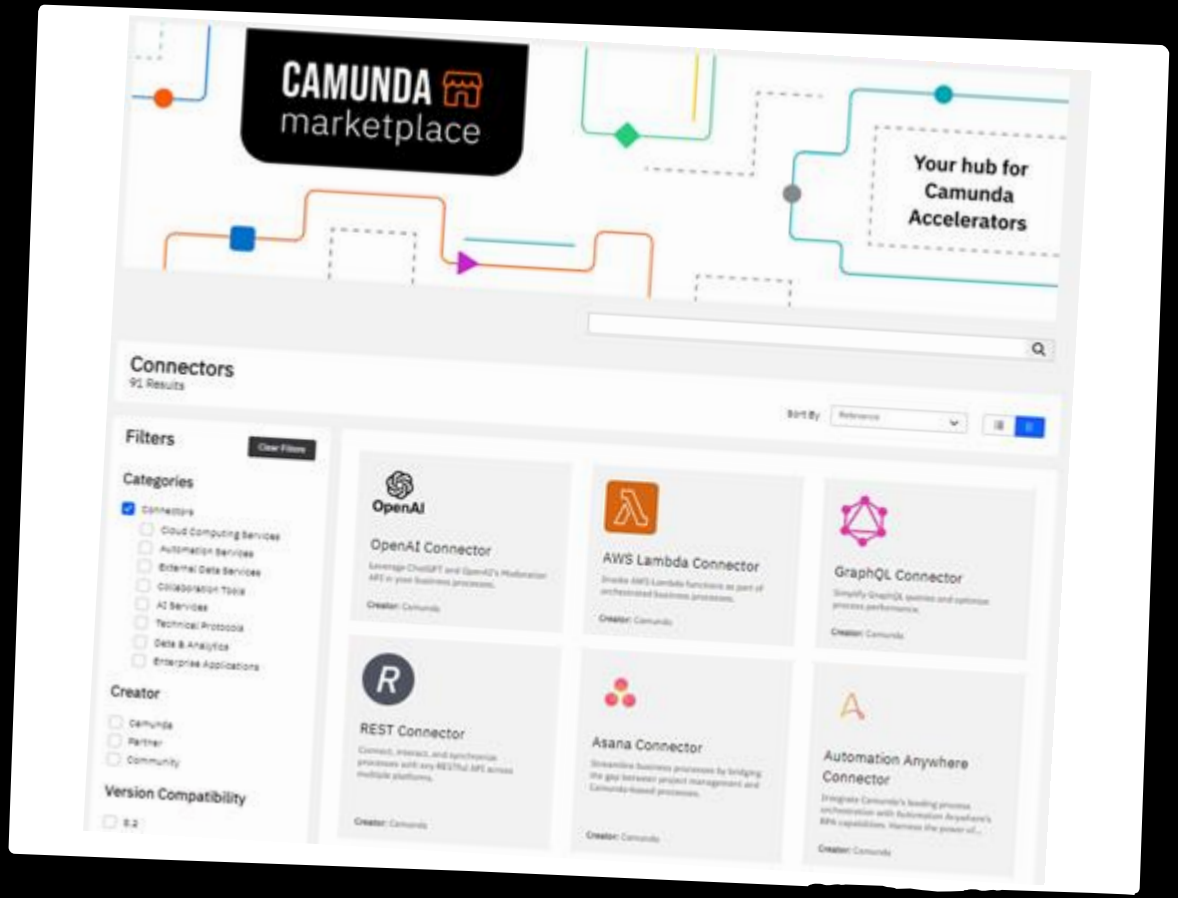
3

## Customizing Connector Templates

- You can either start from scratch or modify an existing template, with the latter being a faster and easier approach for tailoring connectors to specific workflow needs.

# Camunda Marketplace

- **Marketplace for Camunda & Partner Connectors:** Explore a wide range of connectors from both Camunda and its trusted partners, all in one place.
- **Idea Portal for Community Input:** Submit your ideas for new connectors and upvote suggestions from others to shape future releases.
- **Upcoming Connector Releases:** Stay informed about the next connectors planned for release by Camunda, keeping you ahead of the curve.



# Best practices on using and building Connectors


NOW WHAT COULD POSSIBLY GO WRONG?

# Considerations of using Camunda Connectors

## Tight Coupling

Connectors may lead to a more tightly coupled system, where elements like databases become closely integrated with your business processes. Such integration requires updates to the BPMN model following significant changes in the database.

## Vendor Lock-in Concerns

The use of out-of-the-box connectors offers ease and efficiency but might restrict flexibility over time. Transitioning back to standard BPMN models could become more challenging if reliance on specific connectors grows.

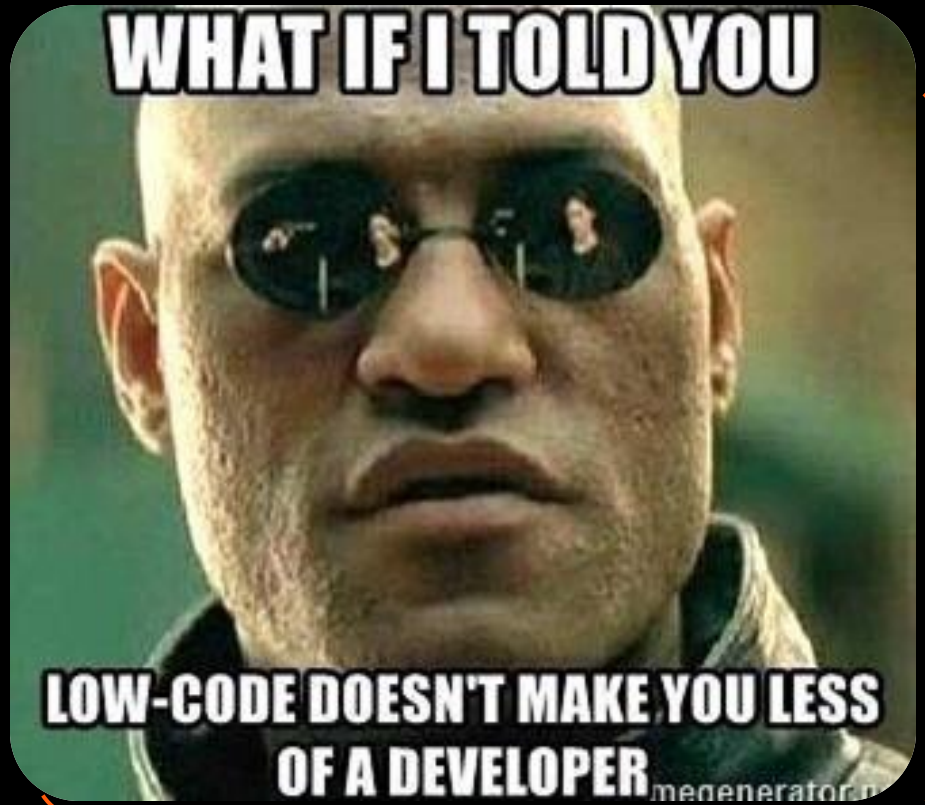
## Increased Data Flow

Connectors might necessitate a larger flow of data through process instances, adding parameters that are essential only for the functioning of these connectors.

# Requirements from a Developer Perspective

## As a developer I want:

- Traceable Changes and Versioning
- Unit Testing
- Regulated Deployments and Stages
- Usage of Open Standards
- The Possibility to do Impact Analysis



# Use domain specific Connectors

REST CONNECTOR
Send SMS

General
Name
Send SMS
ID
Activity_0ehpj4f

Template Applied

Authentication
Type
None
Choose the authentication type. Select 'None' if no authentication is necessary

HTTP endpoint
Method
GET
URL fx
Must not be empty.

TWILIO CONNECTOR
Send SMS via Twilio

General
Name
Send SMS via Twilio
ID
Activity_18f7zju

Template Applied

Operation
Operation type

Authentication
Authentication type

Response mapping
Result variable
Name of variable to store the response in. Details in the documentation

SEND SMS
ServiceTask

General

Template

Input

Message Text

Must not be empty.

The content of the message that will be sent

To Number

Must not be empty.

The recipient's phone number

Protocol level

Domain level

# Relying on Connectors?
# - it depends on the use case

## DO IT YOURSELF

- Simple
- Local automations with little criticality
- No governance or quality assurance needed

## GUIDED

- Medium complexity
- Medium cirticality
- Some governance required
- Some guidance necessary

## PROFESSIONAL DEVELOPMENT

- High complexity
- High criticality
- Complicance and regulatory requirements
- Version control
- Automated testing

It’s workshop time! *

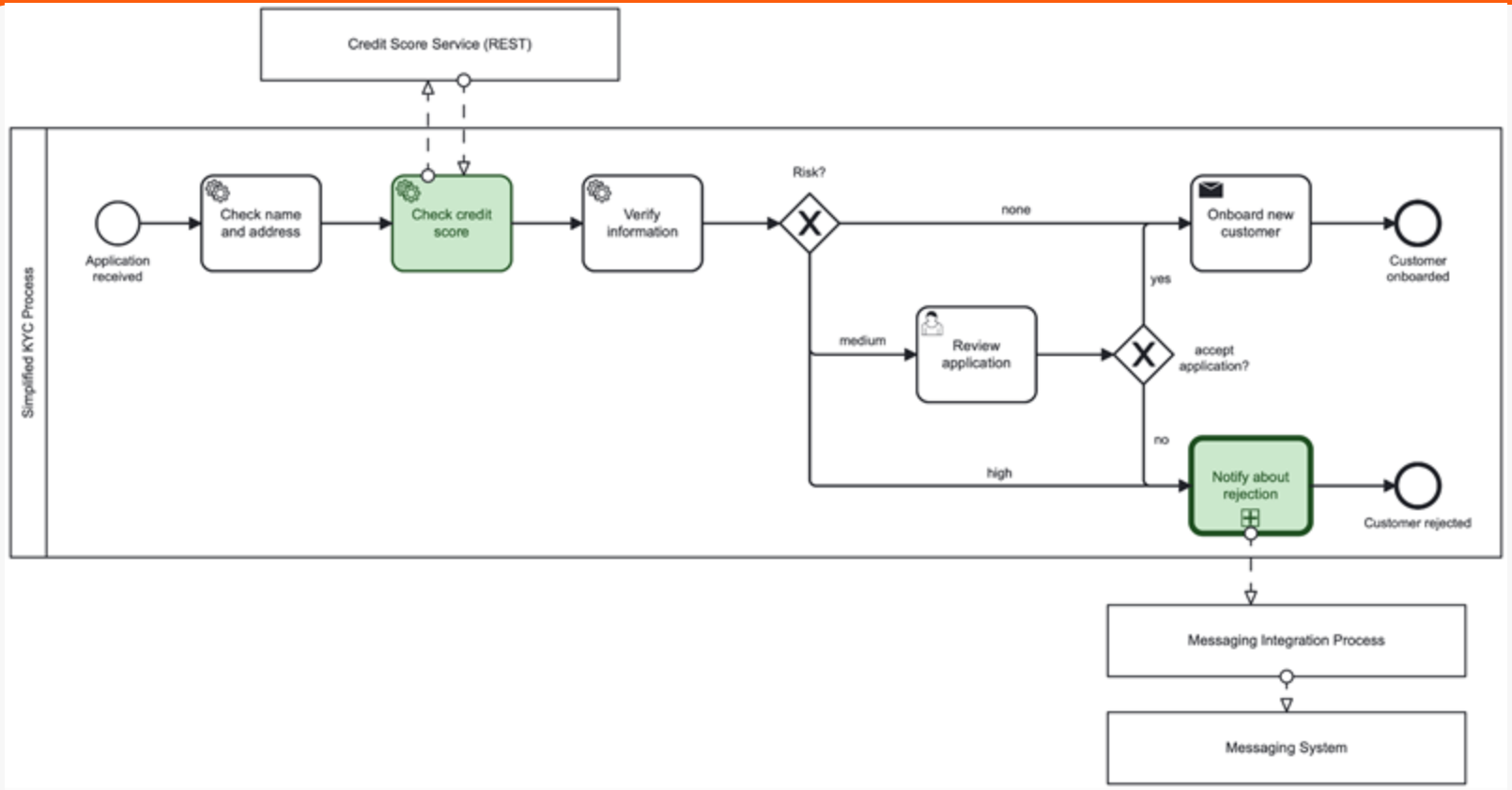
# How to workshop?

Sit back and relax

Follow along as you wish

Or take it home and follow the guide

# It starts with a process model



# Building sustainable connectors in a low code fashion

# Building connectors in a pro-code fashion

# Writing Hexagonal Integrations

## Ports and Adapters

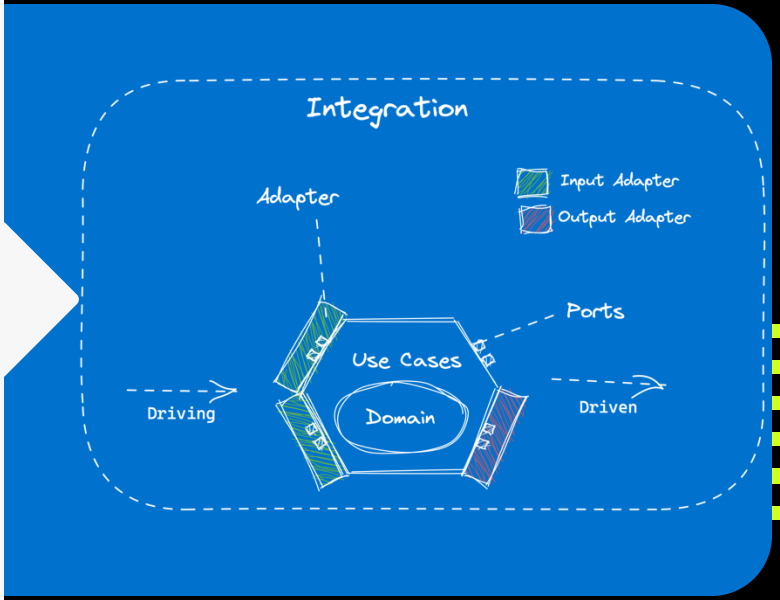
- We organise the hexagonal architecture into layers
- The outermost constist of adapters that translate between the application and other systems

## No outgoing dependencies

- All dependencies point toward the center
- The Domain has no dependency towards the Use-Case or an Adapter

## Benefits

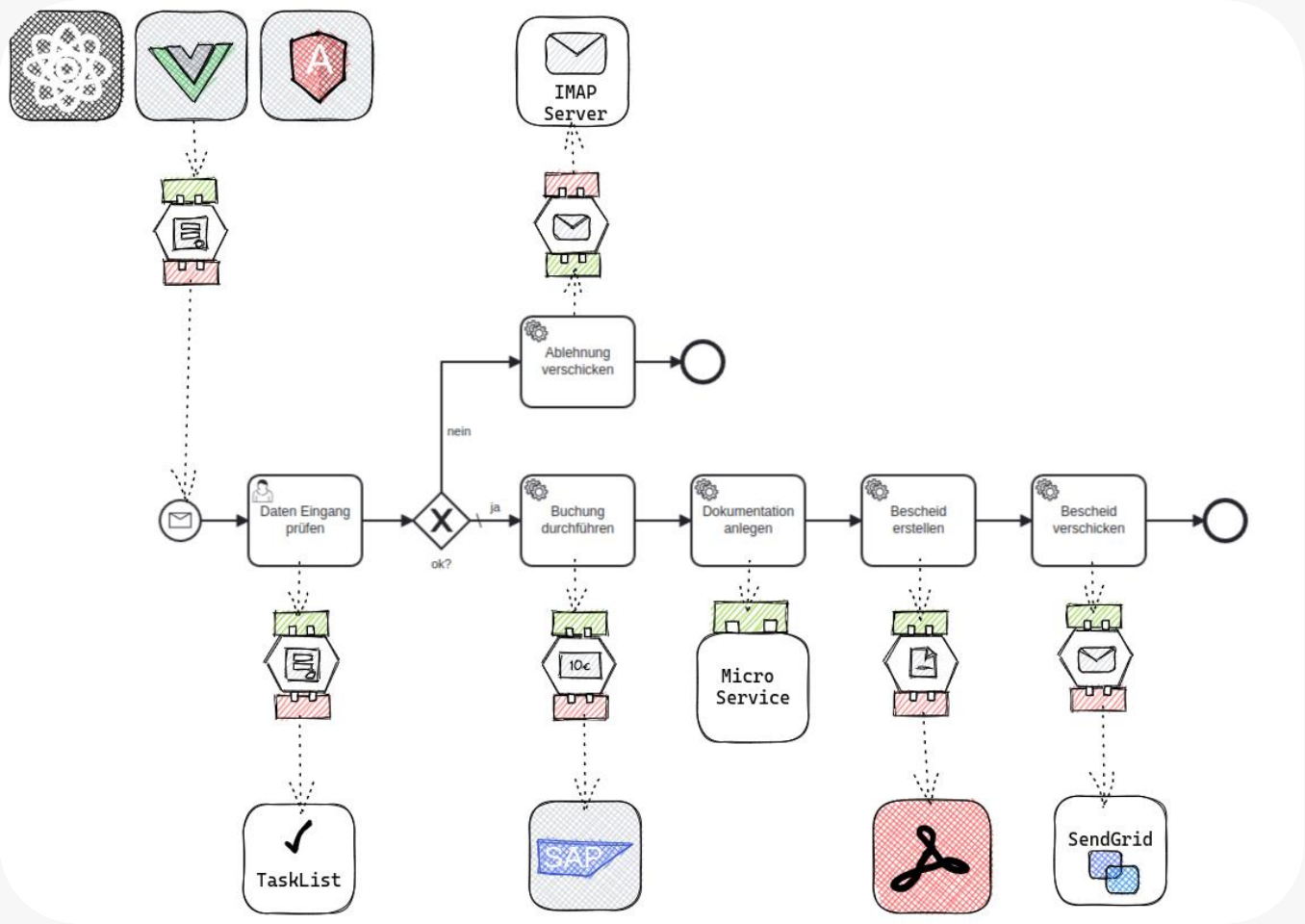
- Truly technology neutral application core
- Easily adaptable to new technical surroundings
- Far easier maintainable




**Integration Component**

Using Hexagonal Architecture

Reading recommendation: [Get your hands dirty with clean architecture]


IMAP Server
Ablehnung verschicken
nein
Daten Eingang prüfen
ok?
ja
Buchung durchführen
Dokumentation anlegen
Bescheid erstellen
Bescheid verschicken
10€
Micro Service
TaskList
SAP
SendGrid

MIRAGON

# Thank You

thomas.heinrichs@miragon.io
calvin.robbins@camunda.com

Thomas Heinrichs
Calvin Robbins

Miragon.io
Camunda.com

CAMUNDA