# Cloud provider agnostic Java and Workflow functions

Maciej Swiderski

*OpenEnterprise*

# who am I?

- independent software engineer and consultant
- workflow enthusiast in the field for more than 15 years
- creator of Automatiko project
- write blog posts to share knowledge
- occasionally tweets about workflows and software development

@SwiderskiMaciek

# Cloud Functions and Java

Does it really make sense?

- Java initially was designed for long running (weeks and months) deployments
- Recently a number of initiatives made it more appealing for serverless deployments
  - GraalVM
  - Quarkus
  - Micronaut
  - Spring Native
- Most of the cloud provider serverless offerings support Java as runtime

# Quarkus Funqy

*“Quarkus Funqy is part of Quarkus’s serverless strategy and aims to provide a portable Java API to write functions deployable to various FaaS environments like*

- *AWS Lambda*
- *Azure Functions*
- *Google Cloud Functions*
- *Knative*

*”*

# What is Funqy?

## 1. Define your function

```
1  public class GreetingFunction {
2
3      @Inject
4      GreetingService service;
5
6      @Funq
7      public Greeting greet(Person person) {
8          return service.greet(person.getName());
9      }
10 }
```

## 2. Try your function

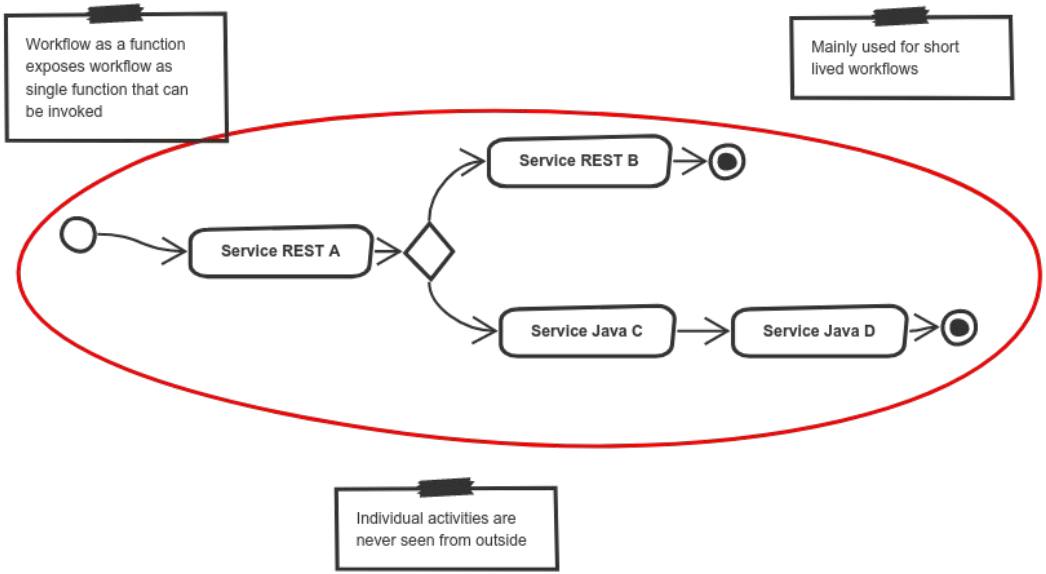
```
// invoke via POST
curl "http://localhost:8080/greet" \
        -X POST \
        -H "Content-Type: application/json" \
        -d '{"name":"john"}'

// invoke via GET
curl "http://localhost:8080/greet?name=john"
```

## 3. Choose Funqy binding and deploy

# Workflow as a Function

- is dedicated for short lived operations that usually last no longer that a (few) second(s).
- typically targets deployments in more constrained environments such as:
  - AWS Lambda
  - Google Cloud Run
  - Azure Functions
- Regardless of the number of activities it contains, the workflow is always exposed as a single function

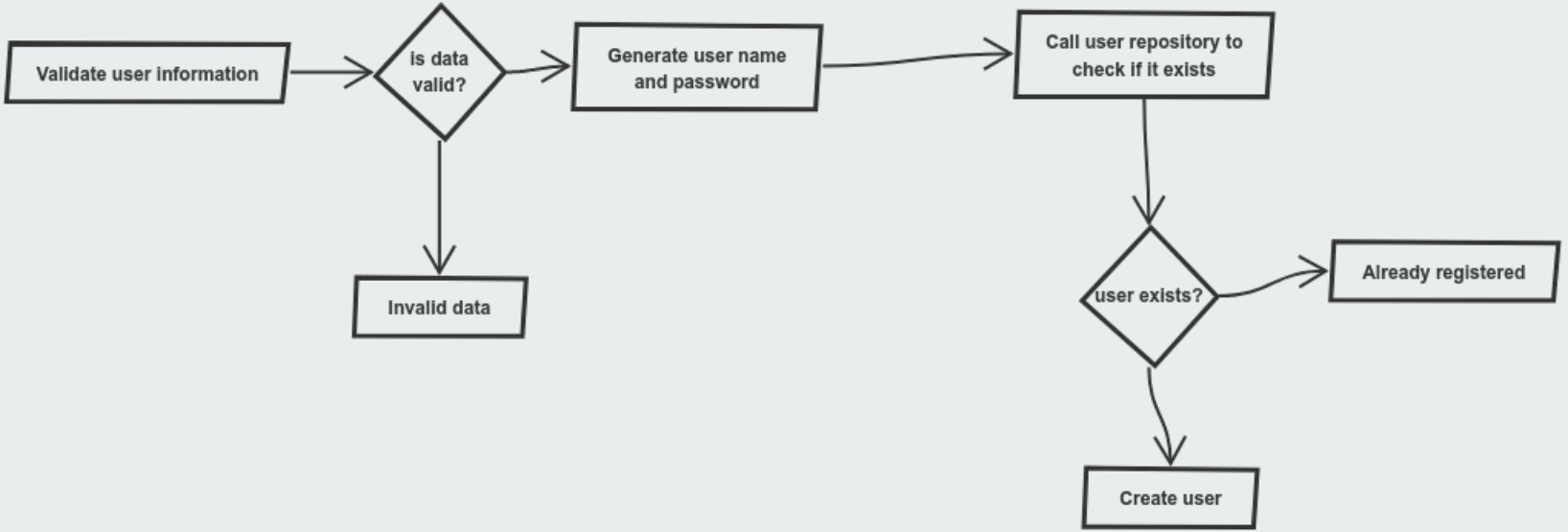
# Workflow as function




- At build time workflow definition is transformed into a **Funqy** function
- Takes advantage of all features of funqy
  - **GET** and **POST** endpoints
  - Bindings to
    - AWS Lambda
    - Azure Functions
    - Google Cloud Functions
    - Knative Eventing
- Completely independent code of the cloud provider though binding selection is done at build time as well

# Use case

Function responsible for registering user in Swagger PetStore service



# Let's see it in action

User registration as function on
AWS Lambda, Google Cloud Functions and Azure Functions

@automatiko_io

# Any one thing more ...

User registration as function flow on
Knative eventing

@automatiko_io

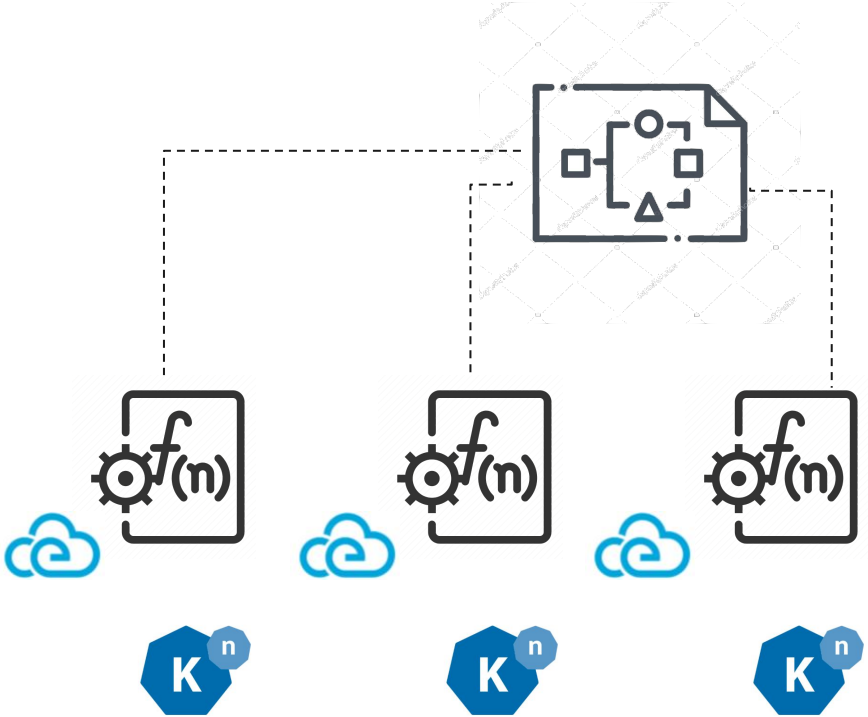
# Workflow as a function flow

Your complete business logic executed as functions

Workflow as a function flow allows to define complete business logic that in most of the cases is more complex than just a single function but execute it as set of functions that are

- self contained
- independent
- invokable at any time
- scalable

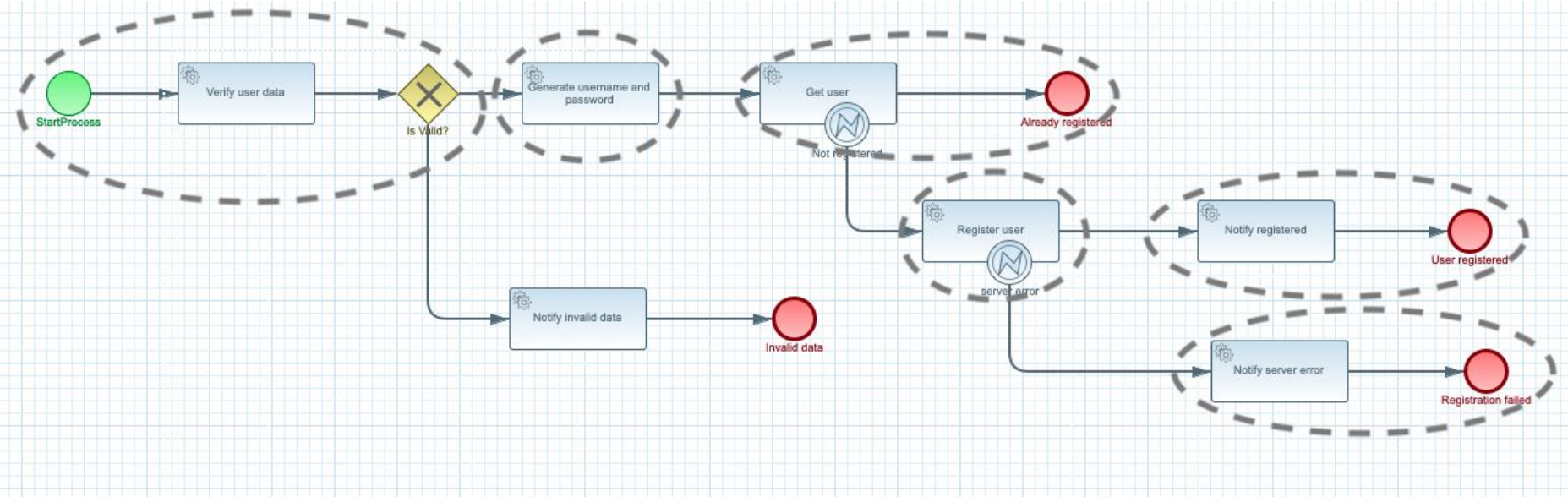
# Functions in the workflow




- Workflow definition is sliced into functions that
  - will be a **dedicated entry point**
  - will have **input**
  - can produce **one or more outputs**
    - becomes a sink
  - will have Knative **trigger** associated with it

- All of these steps are performed based on **workflow definition at build time** (generates both code and manifest file)

# User registration workflow as function flow



# Questions?

# Want to know more?

- Get in touch on twitter
  - **@SwiderskiMaciek**
  - **@automatiko_io**
- Visit website
  - https://automatiko.io
  - https://quarkus.io/guides/funqy
  - https://knative.dev
  - https://knative.dev/blog/articles/workflow-as-function-flow/

**If you are interested and would like to explore more don't hesitate to reach out!**