gsi UPM

OSLC for DevOps event-based automation

Guillermo García Grao

Agenda

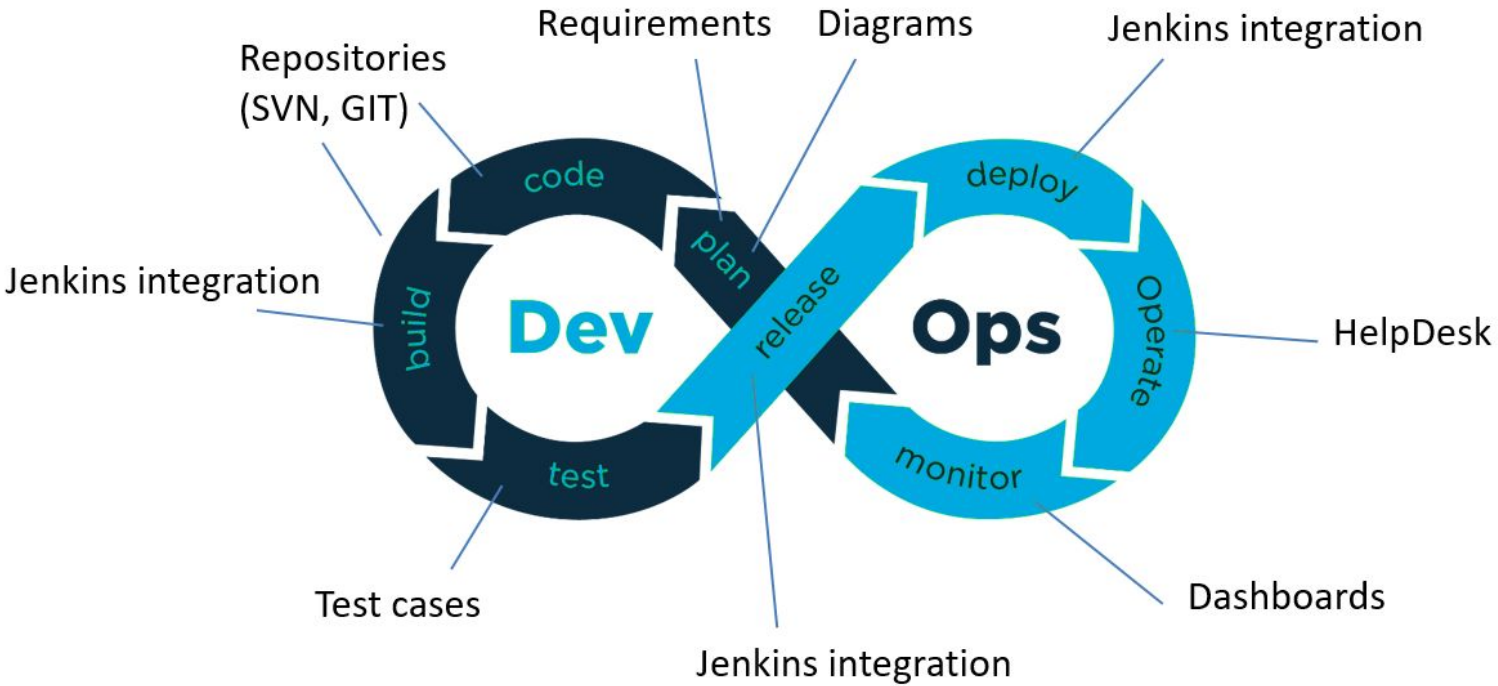
1. Introduction and Context
2. Integrating DevOps Services
3. Automation as a Service
4. Standardizing Event-based Automation
5. Case study

1.

Introduction and Context

DevOps: definition and challenges

Context

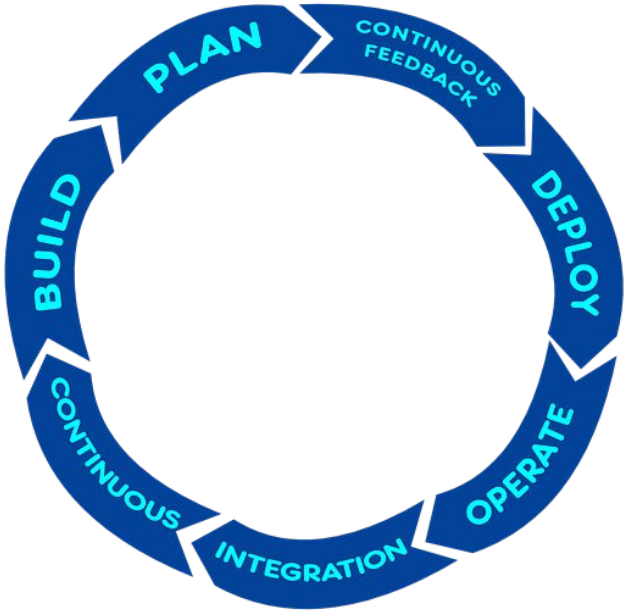

Requirements
Diagrams
Jenkins integration
Repositories
(SVN, GIT)
code
plan
deploy
Jenkins integration
build
Dev
release
Ops
operate
HelpDesk
test
monitor
Test cases
Dashboards
Jenkins integration

“A **collaborative** and **multidisciplinary** effort within an organization to **automate continuous delivery** of new software versions, while guaranteeing their **correctness** and **reliability**.”

Leite et. al. (2019). A survey of DevOps concepts and challenges.

DevOps Goal

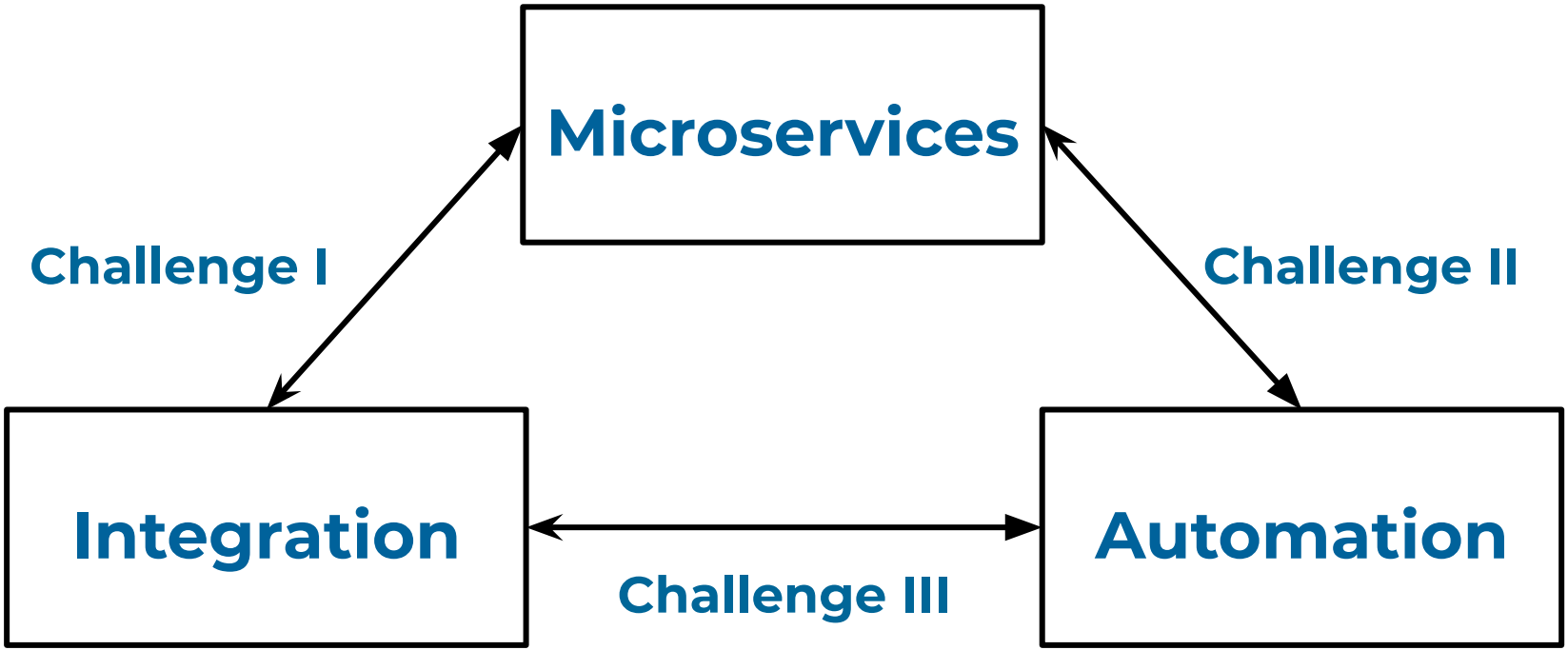
Faster application **production, integration** and **delivery**.



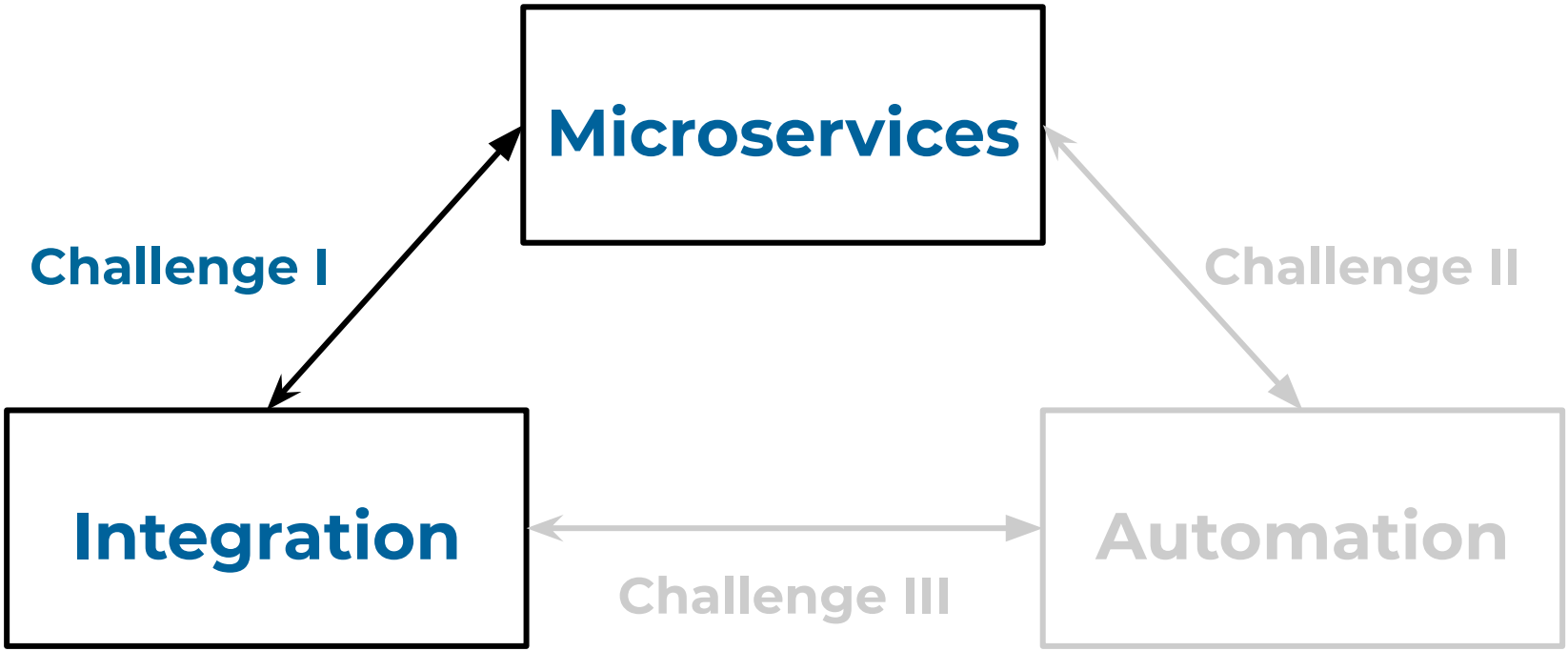
Desired DevOps Features

- **Scalability**
  - Microservices architecture
- **Adaptability** to rapid changes
  - Fast services integration
- **Automation**: building, testing, deployment
  - Continuous Delivery (CD)

DevOps Challenges



DevOps Challenges



2.

Challenge I

Integrating
DevOps
Services

Challenge I - Presentation

Microservices vs Adaptability:

- **Expensive** to migrate APIs
- **Vendor lock-in**
- Lack of **flexibility** and **versatility**

Challenge I - Current State

- Manual migrations
  - Slow
  - Expensive
- Non-standardized APIs
  - Hard to cover every possible tool
  - Lack of flexibility for developers

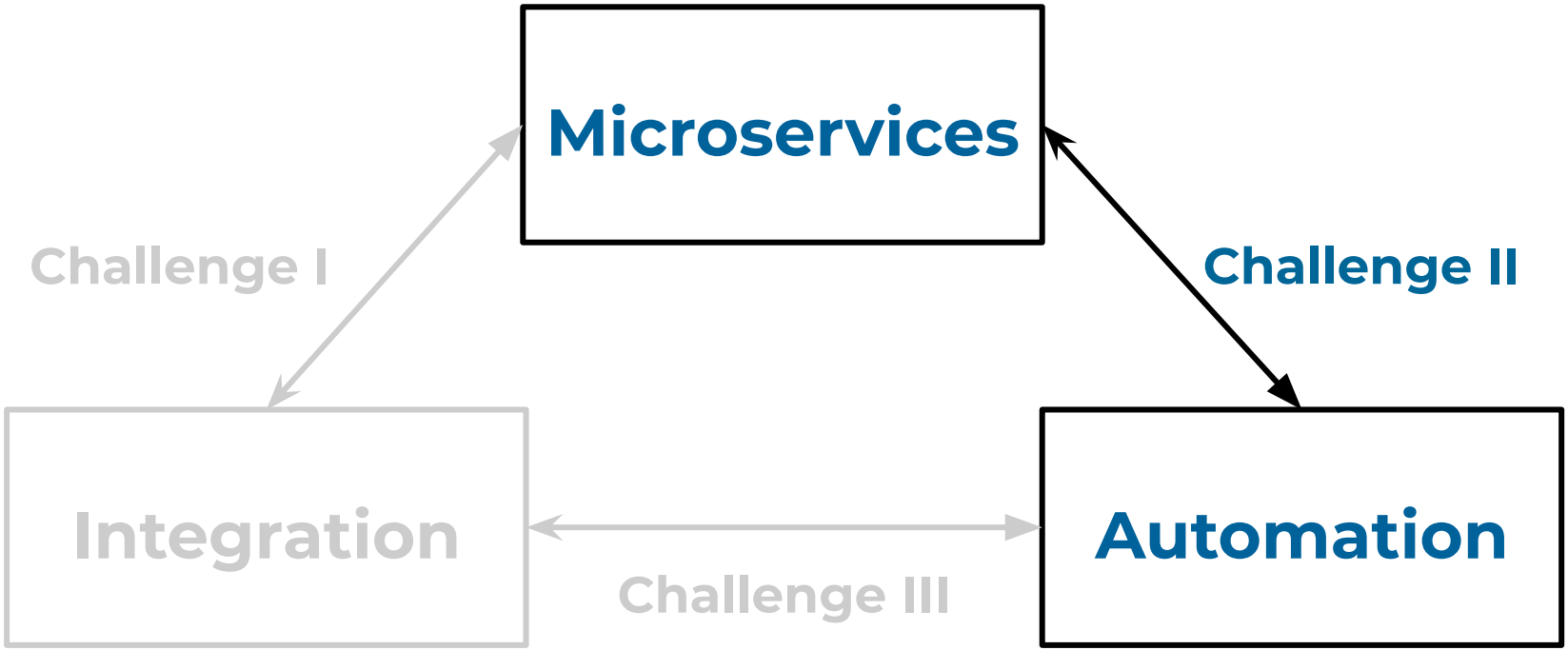
Challenge I - Potential Solution

OSLC

Challenge I - Potential Solution

- **Seamless integration** between compliant tools
- **No assumptions** about tool's internal structure
- Flexible model: **Linked Data**

DevOps Challenges



3.

Challenge II

Automation as
a Service

Challenge II - Presentation

Microservices and Automation:

- Service that **provides automation**
- Concept: **Automation as a Service**
- Implementation: **Task Automation Server**

Challenge II - Current State

Task Automation Servers (TAS)

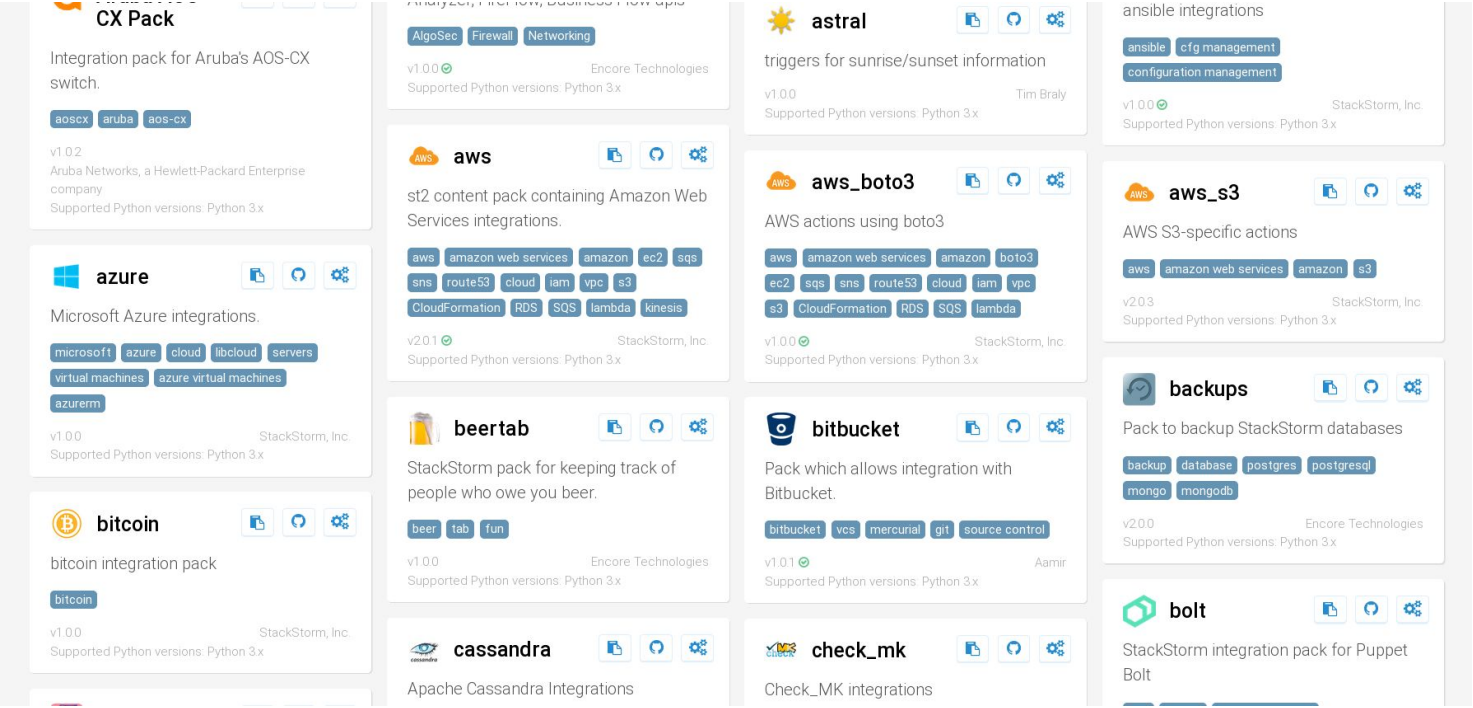
- Provide **automation capabilities** to other services
- Based on the **ECA model**
  - **Event** triggered
  - **Condition** evaluated
  - **Action** executed

Challenge II - Current State

User applications:

DevOps:

Challenge II - Current State



Challenge II - Current State

Task Automation Servers (StackStorm)

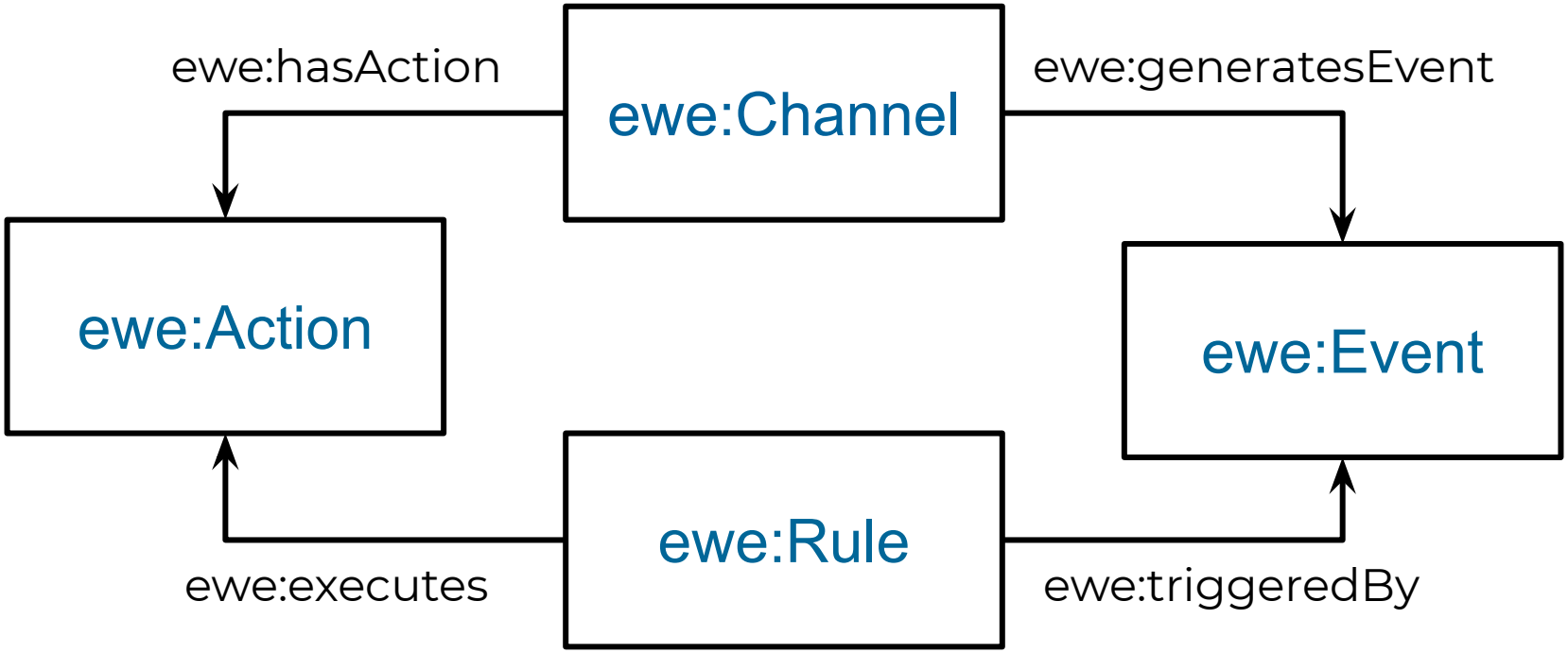
- How to migrate automations?
- **Vendor lock-in**... again

Challenge II - Potential Solution

- **Semantic** model for TAS
- Rule **interoperability**
- **EWE** (Evented WEb) ontology

Challenge II - Potential Solution

- **Semantic** model for TAS
- Rule **interoperability**
- **EWE** (Evented WEb) ontology
  - Linked Data flexibility
  - Reasoning capabilities



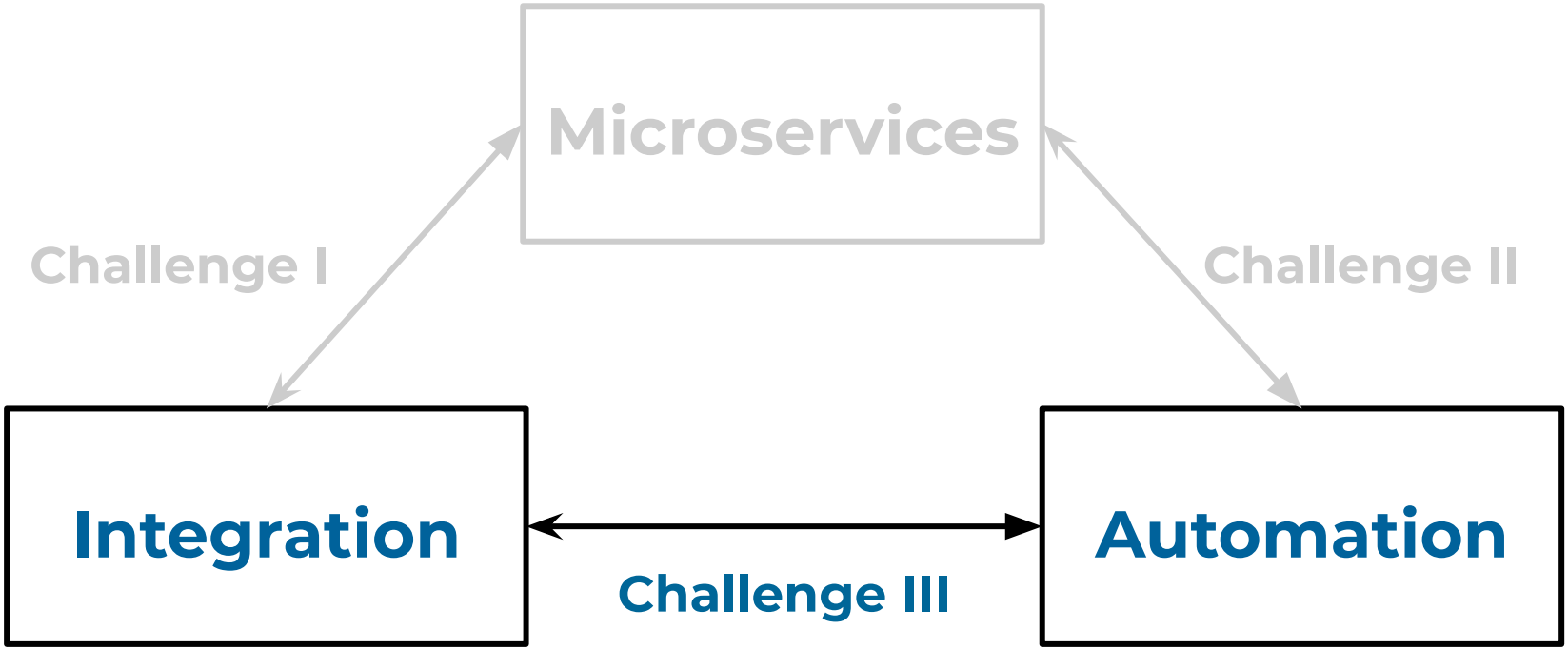
EWE model simplified

Challenge II - Potential Solution

EWE Tasker

- TAS implementation **based on EWE** semantic model
- Rule engine: **EYE reasoner**

DevOps Challenges



4.

Challenge III

Standardizing
Event-based
Automation

Challenge III - Presentation

Standardized interfaces between services

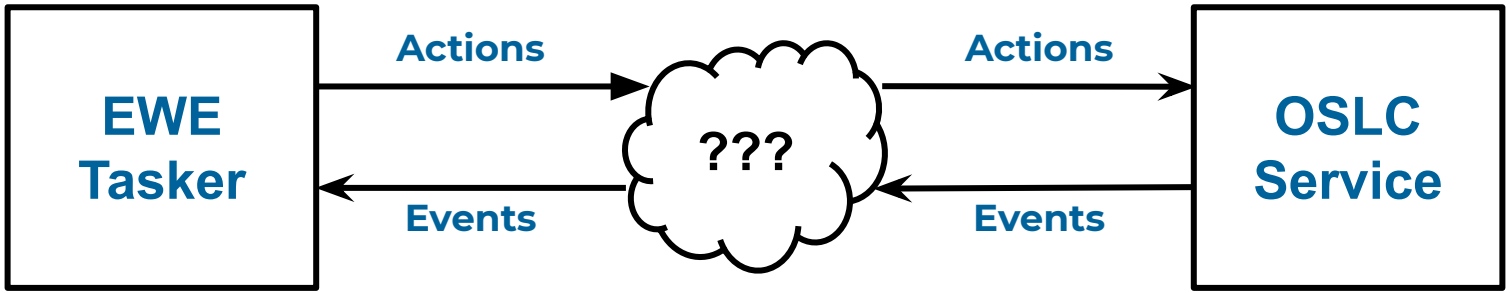
Semantic modeling for Task Automation Servers

OSLC

EWE

Challenge III - Presentation

- Interaction between **TAS** and **OSLC Services**



Challenge III - Presentation

- Interaction between **TAS** and **OSLC Services**

Challenge III - Current State

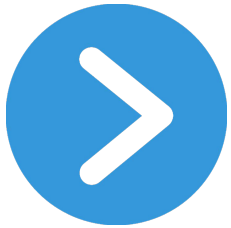
OSLC Automation:

- Should be **extended**
- OSLC **interface for TAS**

Challenge III - Current State

Events in OSLC

Actions in OSLC

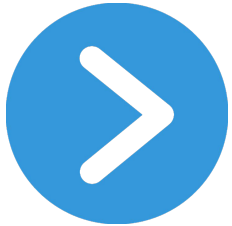


Challenge III - Current State

Events in OSLC

- **Tracked Resource Set** (TRS)
  - Creation, Modification, Deletion
  - TRS Patch for more complex events (TRS 3.0)
- **New spec** (?)

Challenge III - Current State

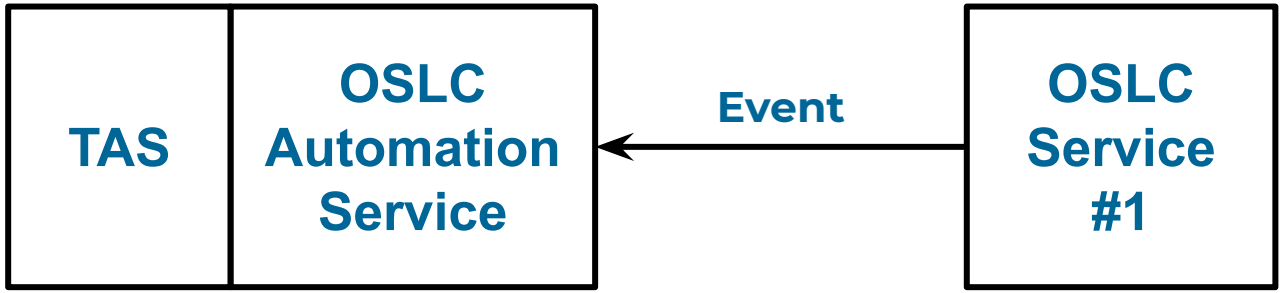


Actions in OSLC

- **CRUD** (Create, Read, Update, Delete)
  - HTTP POST, PUT and DELETE methods
- **OSLC Actions** (?)

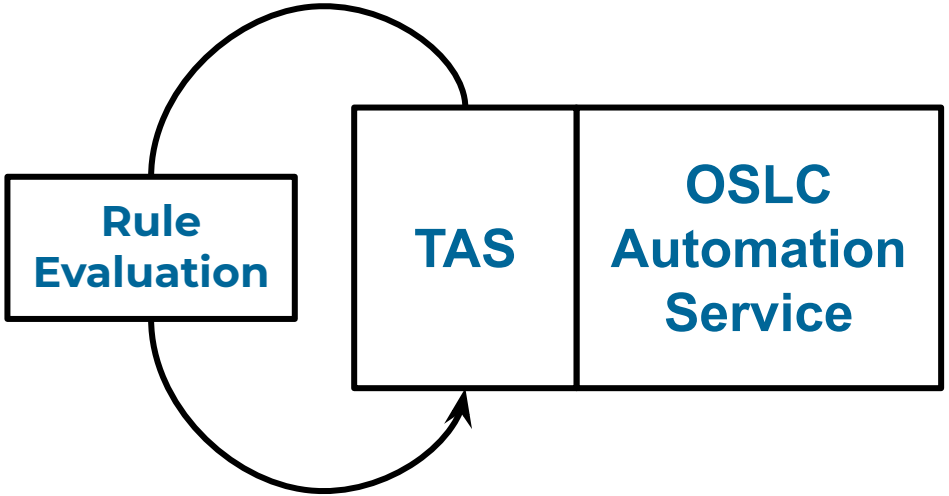
Challenge III - Potential Solution

OSLC Automation + **TAS**



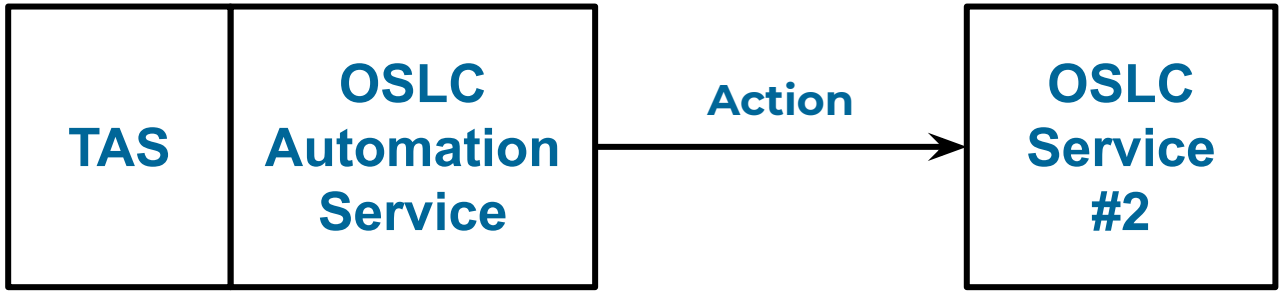
Challenge III - Potential Solution

OSLC Automation + **TAS**



Challenge III - Potential Solution

OSLC Automation + **TAS**



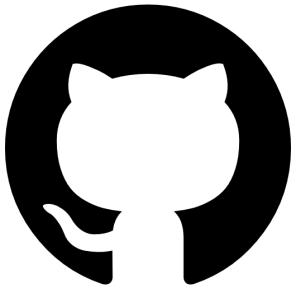
4.

Case study

Github - Bugzilla integration

Case Study - Proposal

Development team uses **GitHub**



Testing team uses **Bugzilla**

Case Study - Goals

- Replicate the bugs from Bugzilla as issues in GitHub (and viceversa) **automatically**

Case Study - Goals

- Replicate the bugs from Bugzilla as issues in GitHub (and vice versa) **automatically**
- Using **OSLC interfaces** between the services

Case Study - Goals

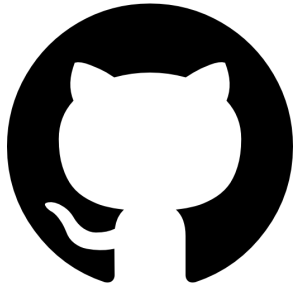
- Replicate the bugs from Bugzilla as issues in GitHub (and viceversa) **automatically**
- Using **OSLC interfaces** between the services
- TAS with **semantic rules** support

Case Study - OSLC Interfaces

Bugzilla

- oslc:ServiceProviders - **Products**
- Resources - **Bugs**
- Domain - **ChangeManagement**

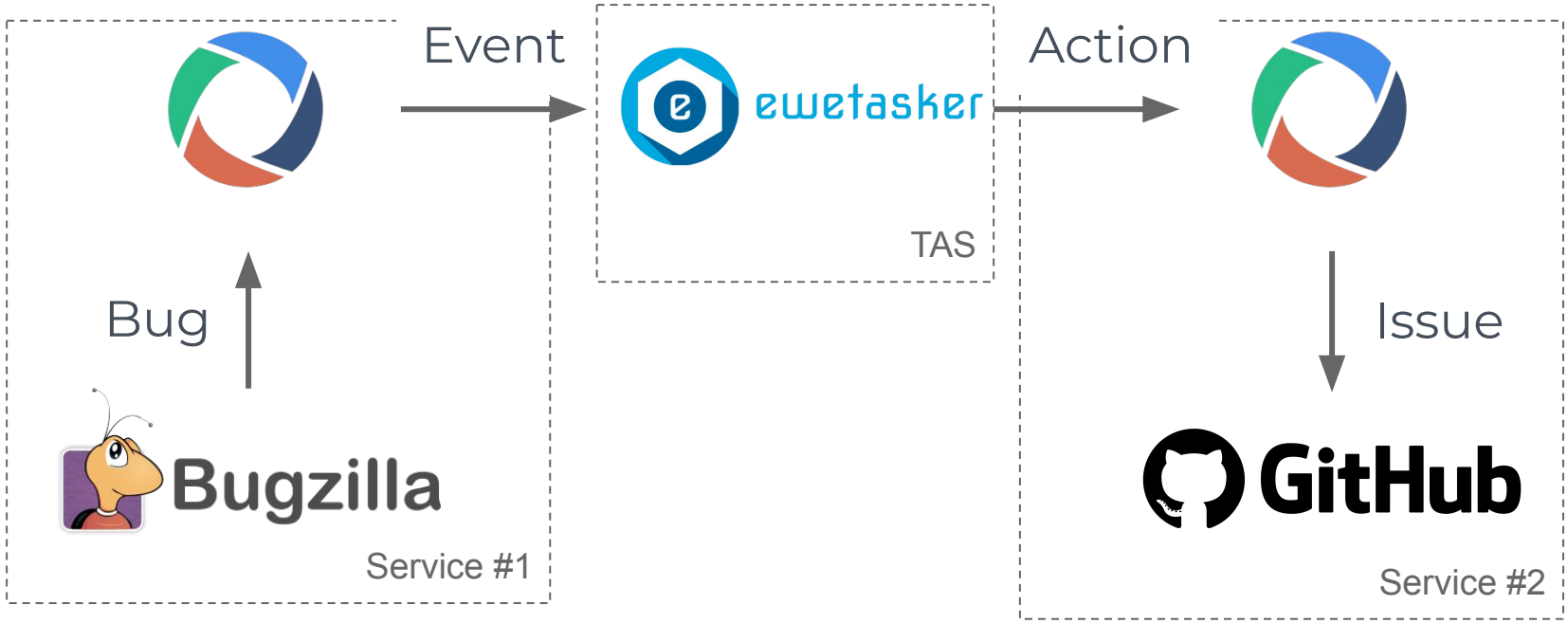
Case Study - OSLC Interfaces



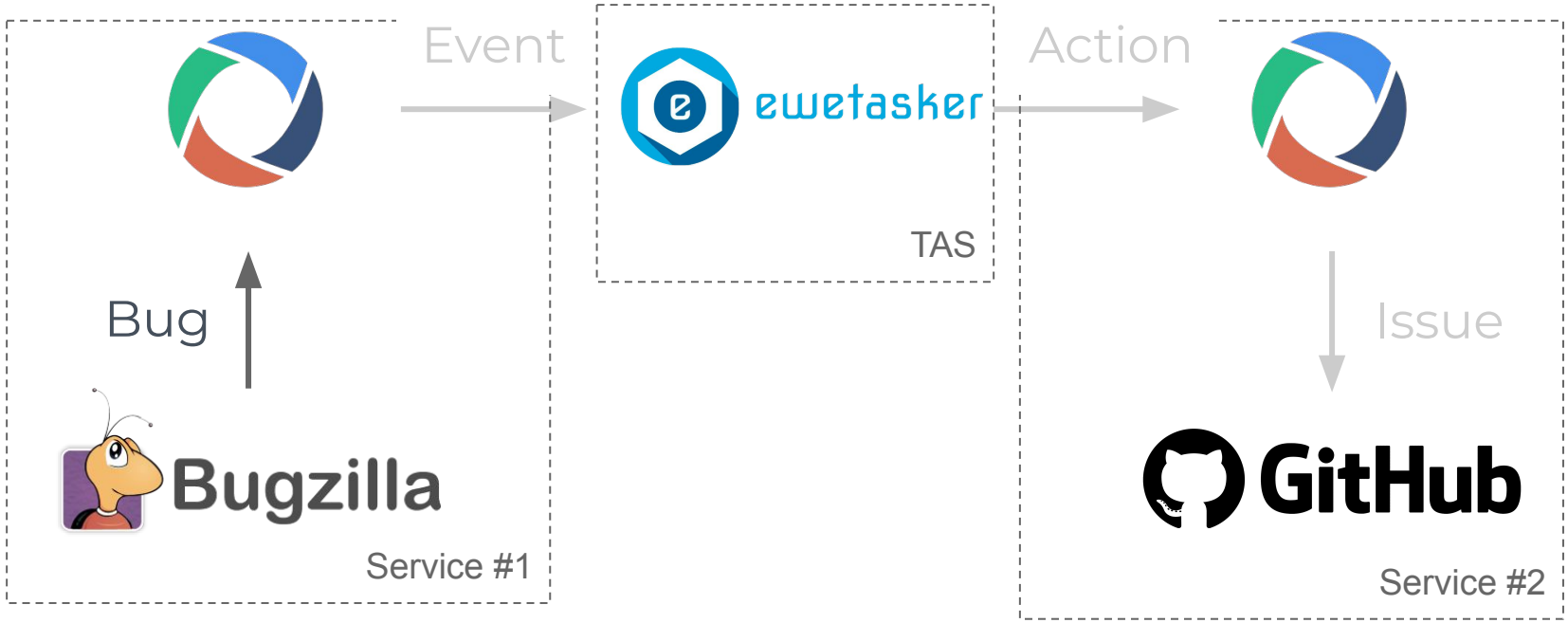
GitHub

- oslc:ServiceProviders - **Repositories**
- Resources - **Issues**
- Domain - **ChangeManagement**

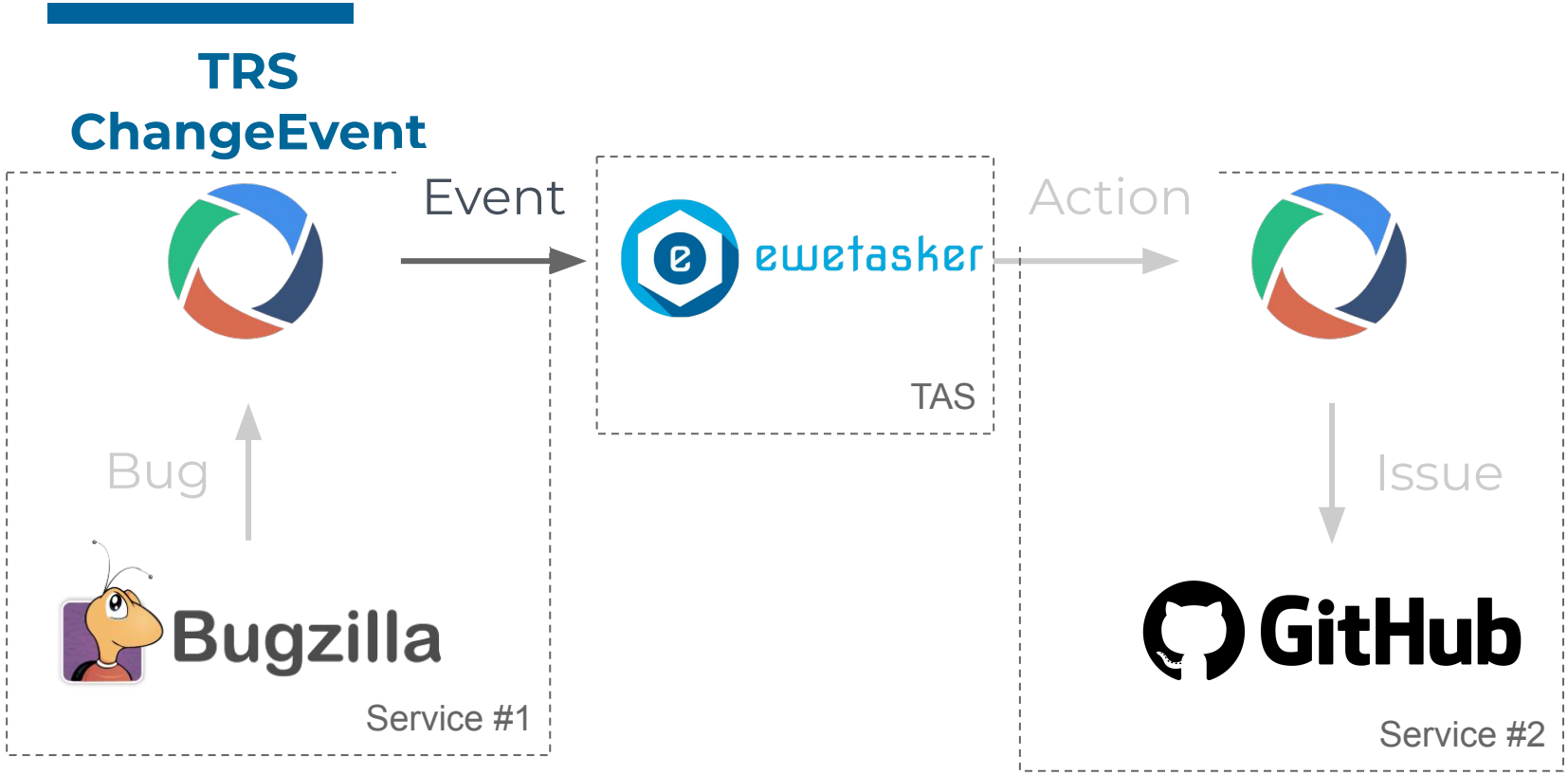
Case Study - Scenario



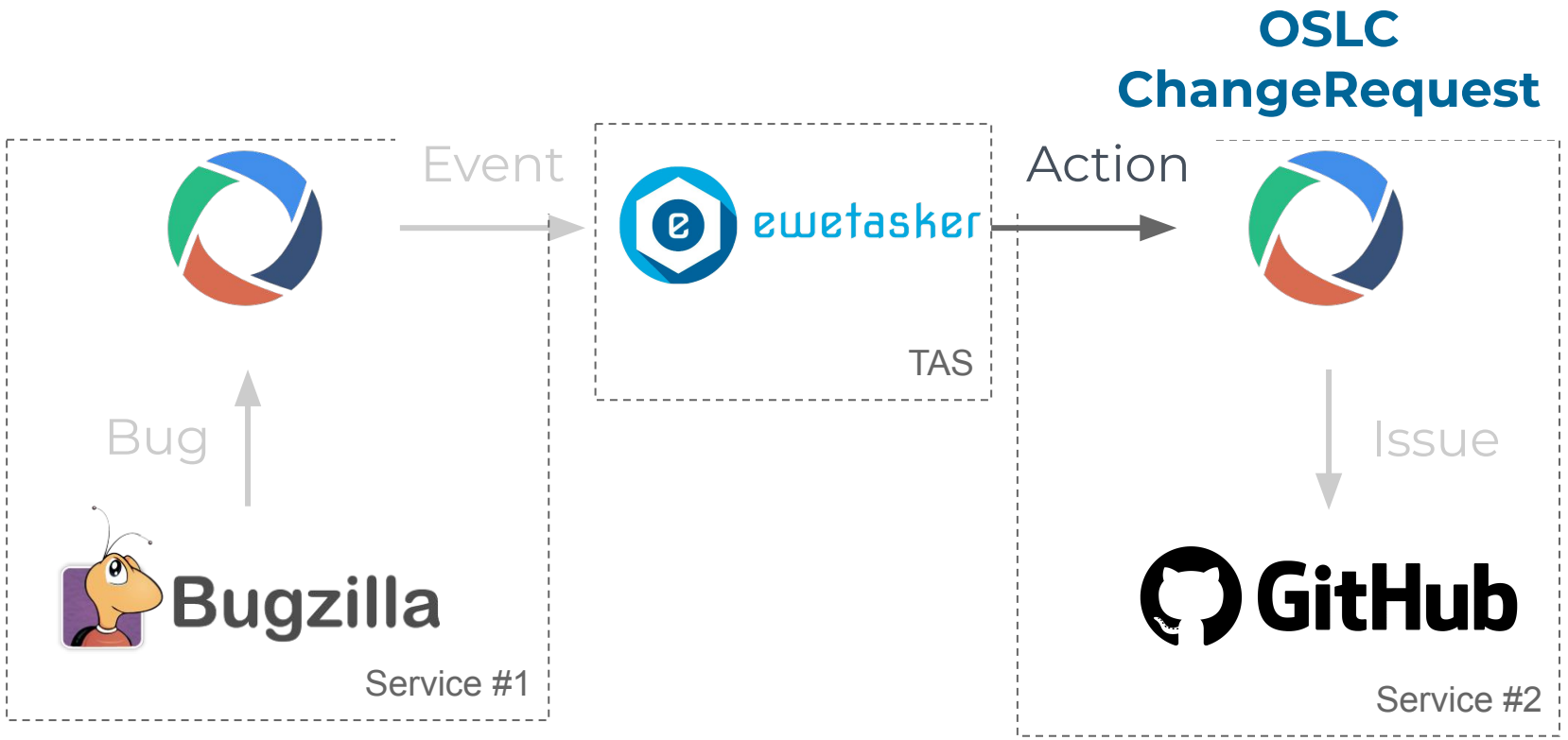
Case Study - Scenario



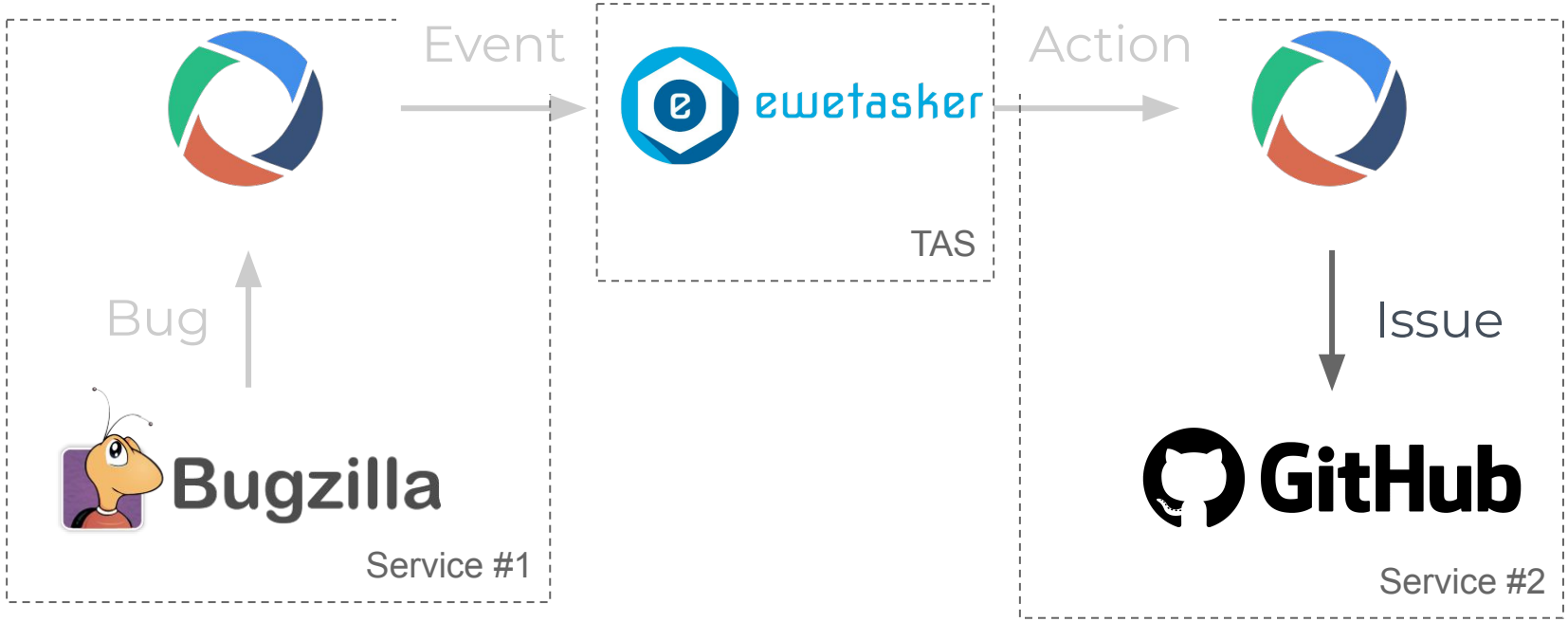
Case Study - Scenario



Case Study - Scenario



Case Study - Scenario



Case Study - Results

Desired **DevOps features**

- Supports **microservices architectures**
- **Fast integration** with new services
- Provide **automation features**

Case Study - Future Work

- Extension for **OSLC Automation** covering TAS features
- **OSLC Events** (new spec?)
- **OSLC Actions** (reactivate?)

Questions?

GSI contact email - gsi@autolistas.upm.es

Guillermo García-Grao - g.ggrao@upm.es