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

- Repositories (SVN, GIT)
- Requirements
- Diagrams
- Jenkins integration
- HelpDesk
- Dashboards
- Test cases
- Jenkins integration
“A collaborative and multidisciplinary effort within an organization to automate continuous delivery of new software versions, while guaranteeing their correctness and reliability.”

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

Microservices

Challenge I
Integration

Challenge II
Automation

Challenge III
DevOps Challenges

- Microservices
- Integration
- Automation

Challenges:
- Challenge I
- Challenge II
- Challenge III
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

Challenge I
Integration

Microservices

Challenge II
Automation

Challenge III
3. Challenge II

Automation as a Service
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:
- *zapier*
- IFTTT
- AutomateIt

DevOps:
- StackStorm
Challenge II - Current State

**CX Pack**
Integration pack for Aruba’s AOS-CX switch.
- **v1.0.2**
  - Aruba Networks, a Hewlett Packard Enterprise company
  - Supported Python versions: Python 3.x

**aws**
ST2 content pack containing Amazon Web Services integrations.
- **v2.0.1**
  - StackStorm, Inc.
  - Supported Python versions: Python 3.x

**azure**
Microsoft Azure integrations.
- **v1.0.0**
  - StackStorm, Inc.
  - Supported Python versions: Python 3.x

**beerTab**
StackStorm pack for keeping track of people who owe you beer.
- **v1.0.1**
  - StackStorm, Inc.
  - Supported Python versions: Python 3.x

**cassandra**
Apache Cassandra Integrations

**astral**
Triggers for sunrise/sunset information
- **v1.0.9**
  - Tim Inkly
  - Supported Python versions: Python 3.x

**aws_boto3**
AWS actions using boto3
- **v1.0.0**
  - StackStorm, Inc.
  - Supported Python versions: Python 3.x

**aws_s3**
AWS S3-specific actions
- **v2.0.3**
  - StackStorm, Inc.
  - Supported Python versions: Python 3.x

**backups**
Pack to backup StackStorm databases
- **v2.0.8**
  - StackStorm, Inc.
  - Supported Python versions: Python 3.x

**couchbase**

**check_mk**
Check_MK integrations

**bolt**
StackStorm integration pack for Puppet Bolt
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

ewe:Rule → ewe:executes → ewe:Action

ewe:Rule ← ewe:executes ← ewe:Action

ewe:Channel ← ewe:hasAction ← ewe:Action

ewe:Channel ← ewe:generatesEvent ← ewe:Event

ewe:Event ← ewe:triggeredBy ← ewe:Rule
Challenge II - Potential Solution

EWE Tasker

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

- Microservices
- Integration
- Automation

Challenge I

Challenge II

Challenge III
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 vice versa) *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 vice versa) 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

Event → ewetasker → Action

Bug → Bugzilla → Service #1

TAS → Issue → Service #2
Case Study - Scenario

Event

Action

Bug

Bugzilla

Service #1

ewetasker

TAS

GitHub

Issue

Service #2
Case Study - Scenario

**TRS**
Change Event

Event

**Bugzilla**
Service #1

Bug

**ewetasker**
TAS

Action

**GitHub**
Service #2

Issue
Case Study - Scenario

Service #1

Bugzilla

Bug

Event

ewetasker

TAS

Action

ChangeRequest

OSLC

Service #2

GitHub

Issue
Case Study - Scenario

[Diagram showing the flow of events and actions between Bugzilla (Service #1) and GitHub (Service #2) through the process of handling a bug (TAS, Action, Issue)].
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