DICE Fault Injection Tool

Craig Sheridan, Darren Whigham

Flexiant Limited

Matej Artač

XLAB d.o.o.

Overview

o Motivation

o Fault Injection Tool

o DICE project and DevOps

o Conclusion

Motivation

o Infrastructure equipment fails

o Expensive approach:

- Making hardware completely reliable
- Making sure that software never fails

o Reasonable approach:

- Make systems resilient to failures

o Need for on-demand failures
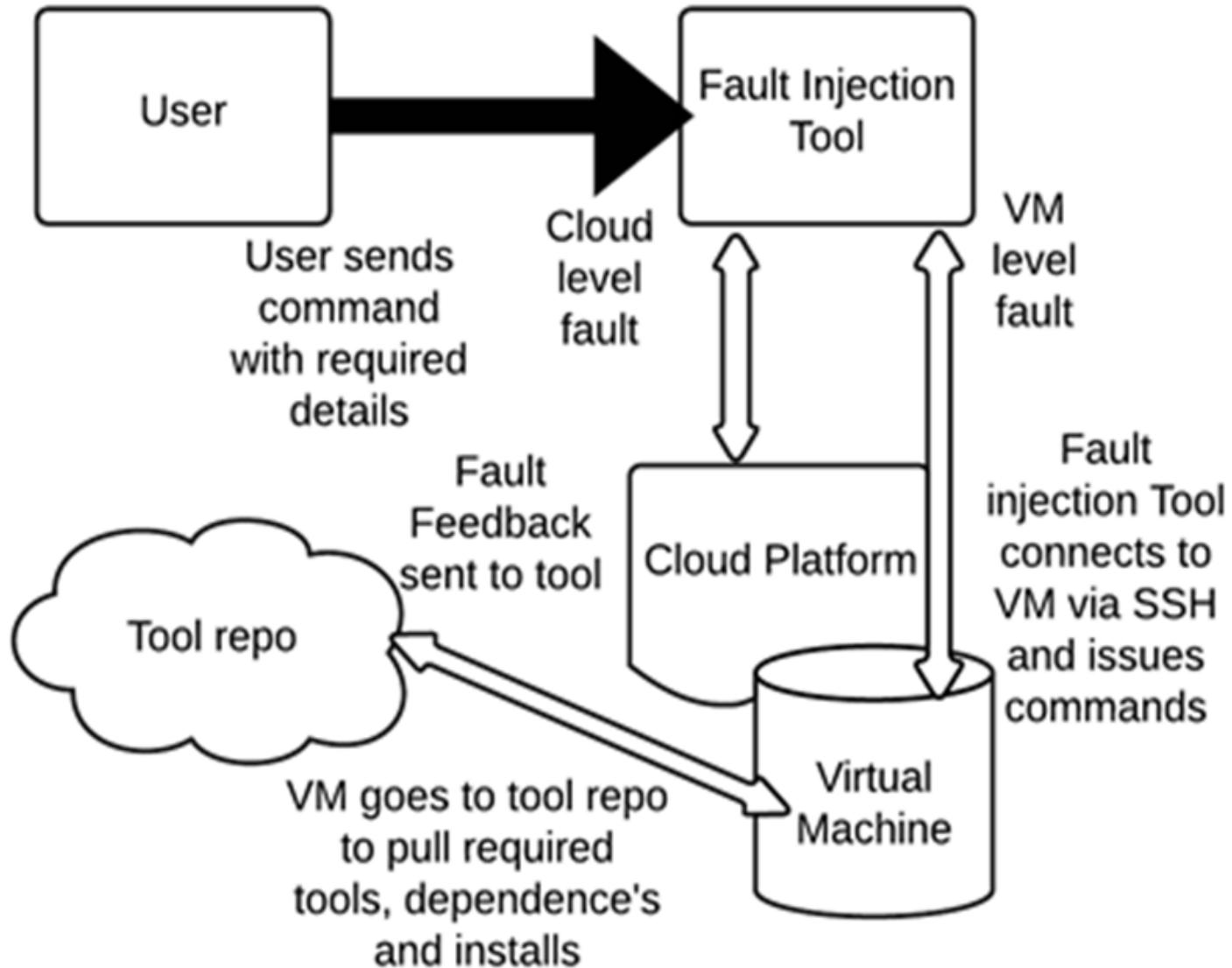
Fault Injection Tool

Fault Injection Tool

o Problem

- When testing, it is difficult to **reproduce faults** happening in production

o Goal

- Causes environmental problems or **failures on demand**

- Lets users **control** the **levels** and seriousness of induced failures

Design

Supported VM-level Faults

- **Shutdown** random VM (black list or white list)
- High **CPU** for VM
- High **Memory** usage for VM
- Block VM external **network** access
- High **Network Bandwidth** usage
- High **Disk I/O** usage
- Stop a **service** running on VM

Supported Cloud-level Faults

o **Shutdown** a Node

o High **CPU** for Node

o High **Memory** usage for Node
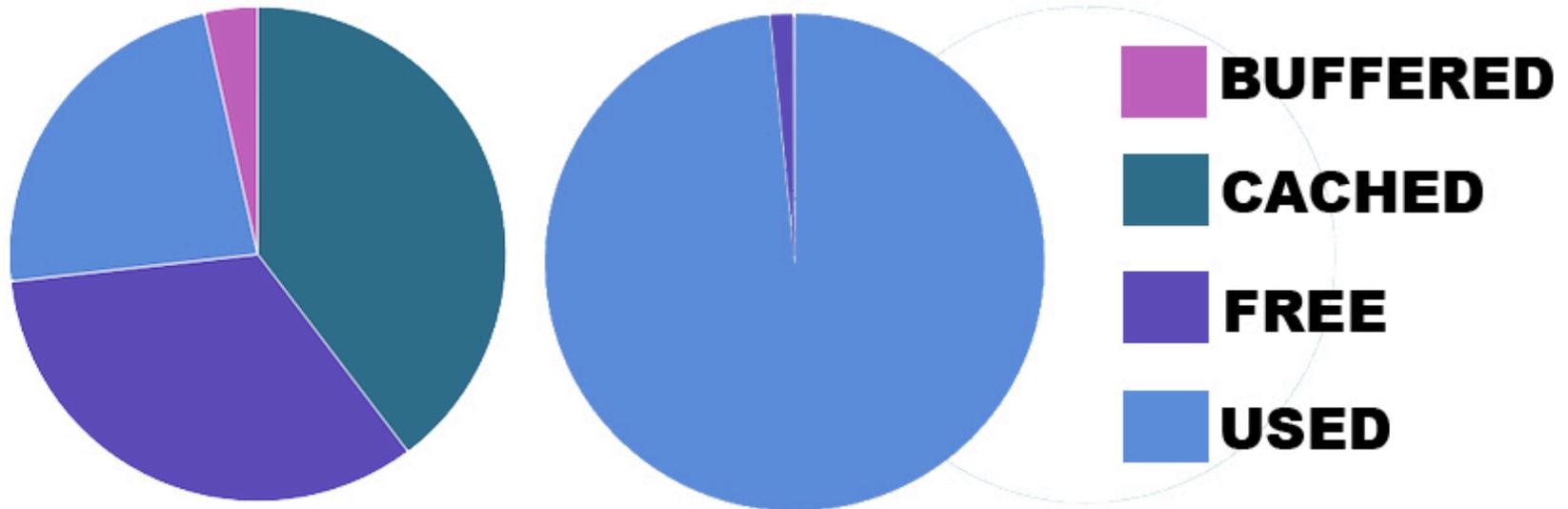
o High **Bandwidth** usage

Usage example command

```
$ fleximonkey --stressmem \
    2 \                        test loops
    2048m  \                   total memory
    ubuntu@109.231.126.101 \   ssh host
    -no \                      no password
    home/ubuntu/SSHKEYS/VMkey.key
```

Usage example result

Innovation

o ChaosMonkey: only accesses AWS & terminates VMs.

o Cocoma: developed but not supported, suffers from high resource usage, complex config & limited extensibility.

o FIT generates various cloud agnostic faults at VM & Cloud.

o Range of functions, greater flexibility to generate multiple faults

o Lightweight and only installs required tools and components on target VMs.

o Future extensibility in mind considering the needs and challenges of cloud service providers such as scalability and resiliency of the cloud consumption marketplace.

DICE Framework

Use in DevOps
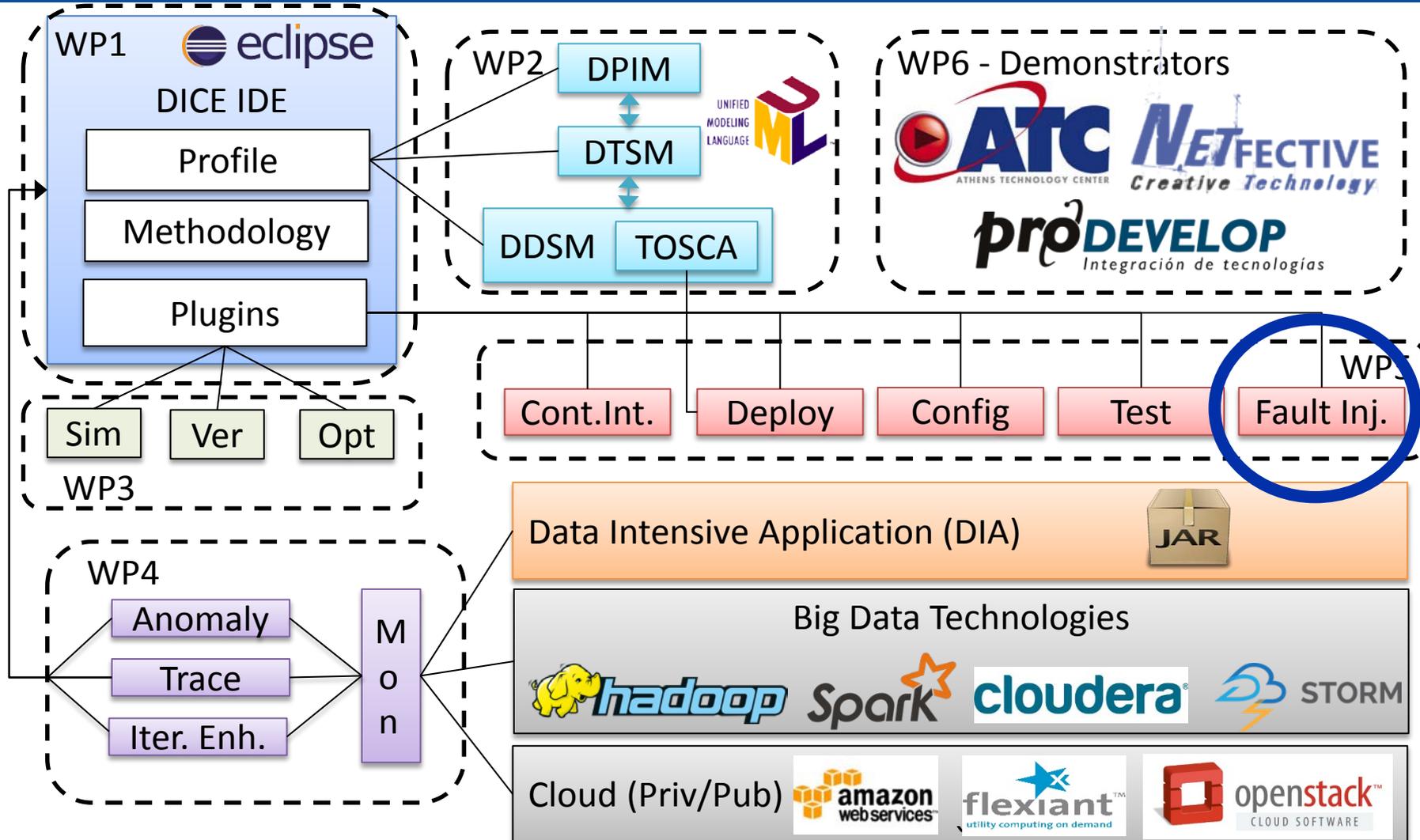
o Continuous Integration/Testing

- Regular resilience tests

o Anomaly detection

- Induce faults, check for anomalies
- Help with quality enhancement

Conclusion and future work

o Aimed at **developers** using **quality driven DevOps** approach

o Helps **cloud operators** perform stress analysis

o Difficulties and possibilities of **extensibility** by external users and investigating limitations

o Consider different topologies, operating systems and vendor agnostic cloud provider infrastructure

o Evaluate the **overhead** of operation

o **Containerised environments** will be a future target to help understand effect on microservices when injecting faults to the underlying host

Release: February 2017

https://github.com/dice-project/DICE-Fault-Injection-Tool