

Berlin, 11-13 October 2017



TESTING BIG DATA APPLICATIONS AT DESIGN TIME AND RUNTIME WITH DICE

Matej Artač (XLAB), Ismael Torres (Prodevelop),
Vasilis Papanikolaou (ATC), Giorgos Giotis (ATC)

Berlin, 11-13 October 2017



MOTIVATION

Why DICE?

Outline

- About the DICE project
 - DevOps and testing
- DICE demonstrators
 - Posidonia Operations from Prodevelop
 - News Asset from ATC
- Conclusions

Berlin, 11-13 October 2017



ABOUT THE DICE PROJECT

Developing Data-Intensive Cloud Applications with
Iterative Quality Enhancements

Internet of Things feeds Big Data

- IoT is a growing source of data
 - sensors
 - data feeders
 - low power processing
- Big Data
 - storing large volumes of data
 - streaming and off-line analysis
 - finding value in data

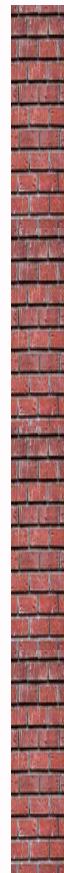
What problems EU SMEs face?

Learning curves → 

Initial prototyping → 

Risk of failure → 

Fast-paced market → 



Customers with legacy data now ask for Big Data technologies



(+ others...)



DICE: Quality-Aware DevOps for Big Data



Design



Prototype



*“Quality-Aware DevOps
for Big Data”*

Enhance

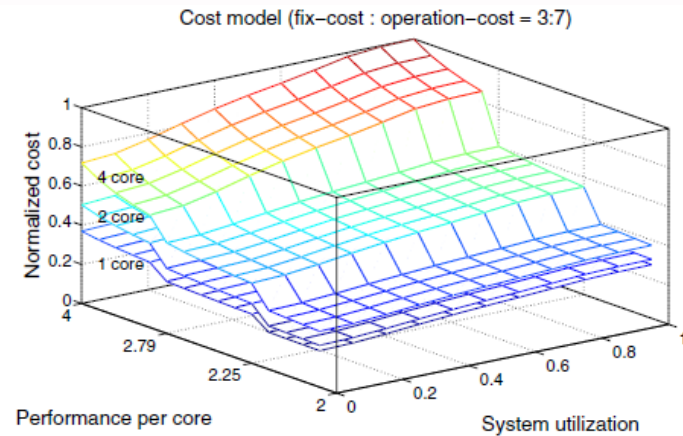
Deploy

Monitor



What do we mean by Quality?

- Reliability
 - Availability
 - Fault-tolerance
- Efficiency
 - Performance
 - Costs
- Correctness
 - Privacy & security
 - Temporal metrics

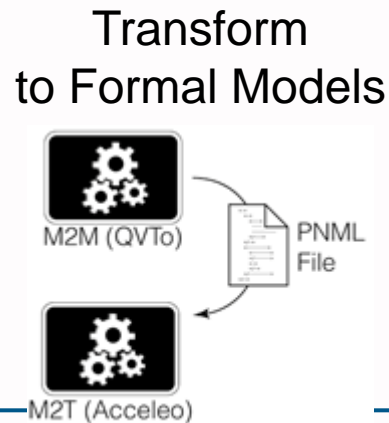
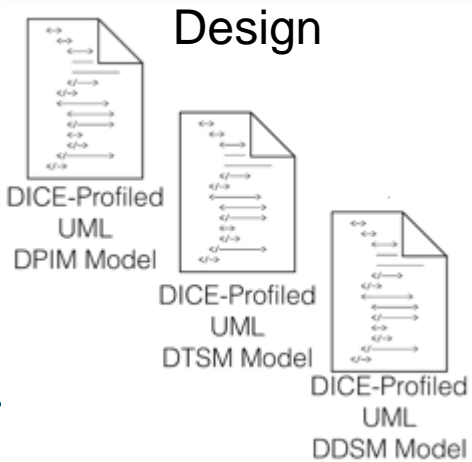
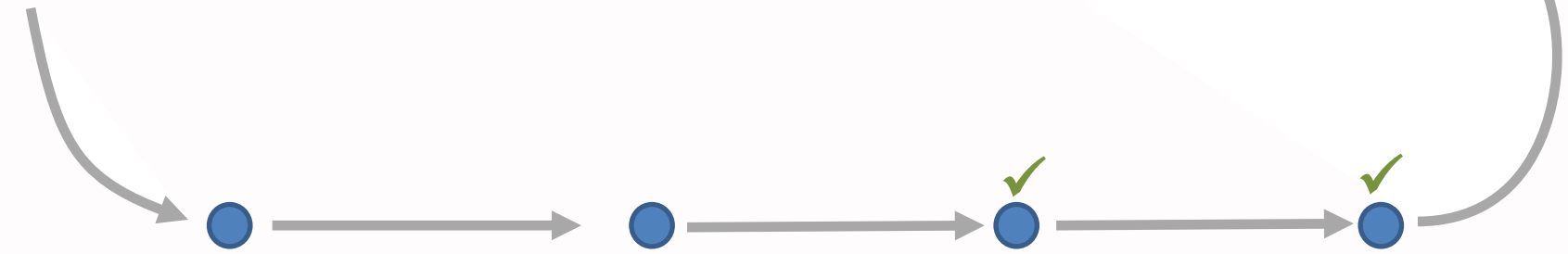


DICE Workflow - Dev

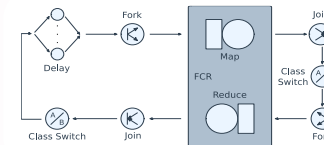


DICE Eclipse IDE

Enhance



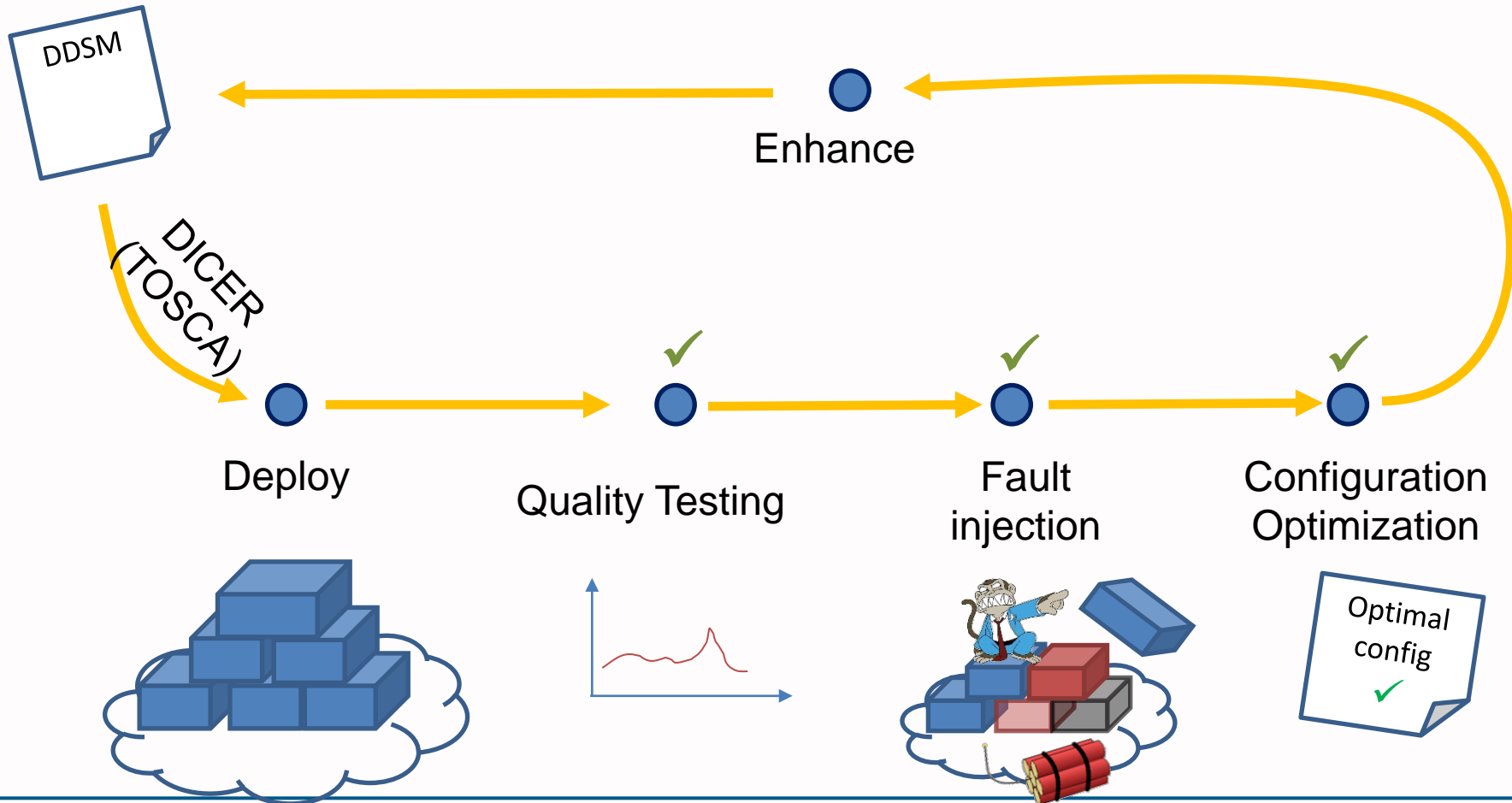
Simulate & Verify



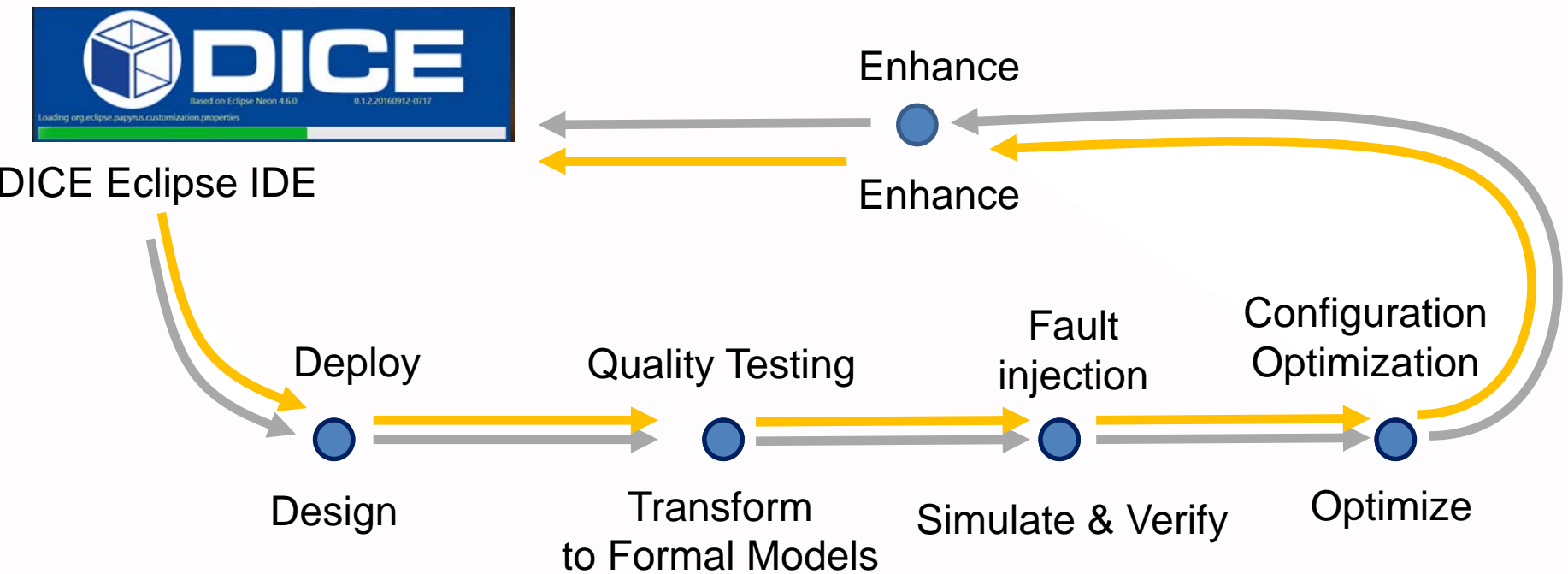
Optimize



DICE Workflow - Ops



DICE Workflow: Unified Toolchain



Berlin, 11-13 October 2017



DICE DEMONSTRATORS

Tools for real SMEs with
real use cases

Demonstrators

pro²DEVELOP
Integración de tecnologías



Posidonia Operations



Maritime
Sector

ATC
ATHENS TECHNOLOGY CENTER



News Asset



News&Media
Market

NETFECTIVE
Creative Technology

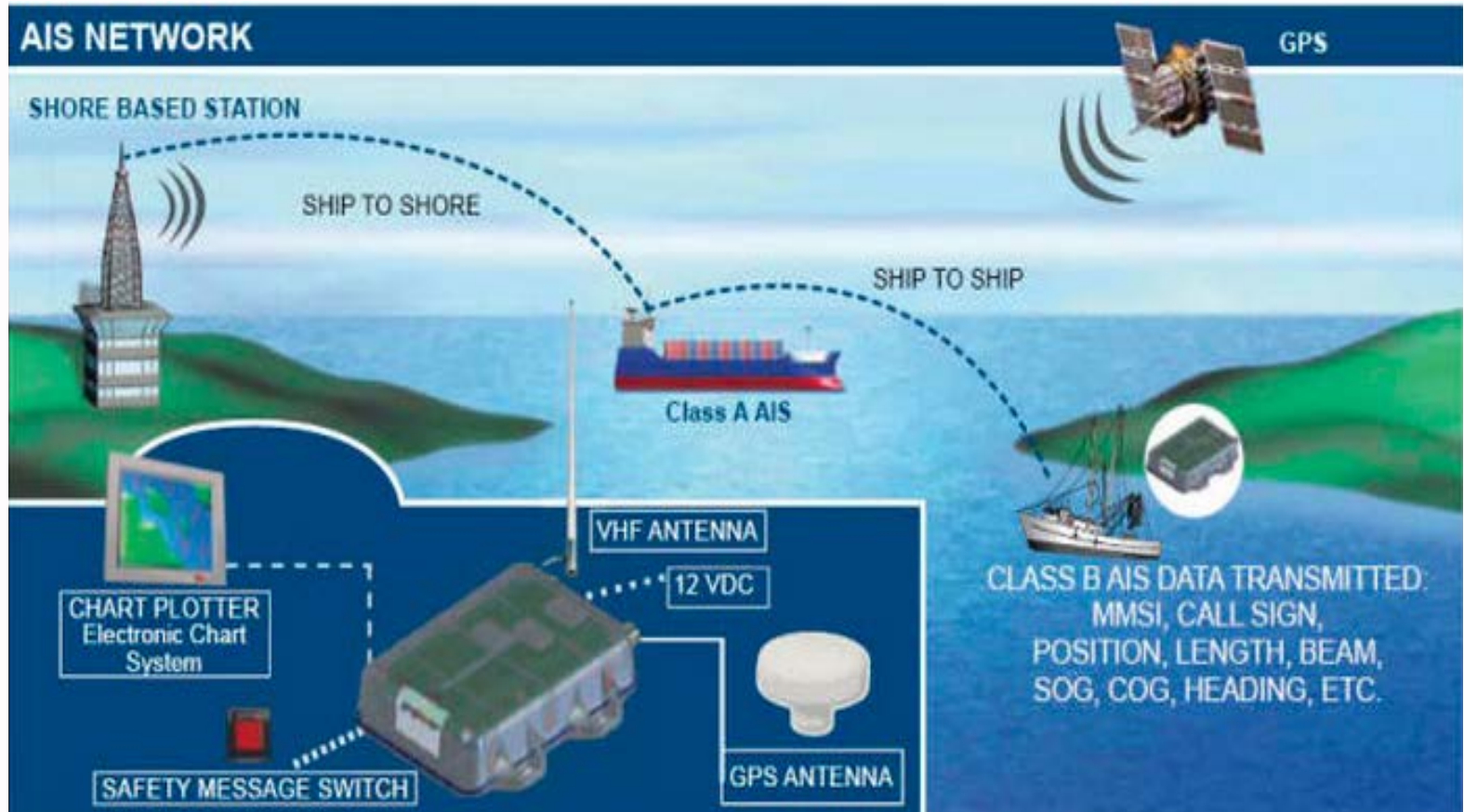


Tax Fraud Detection
Application



e-Government
Market

Tutorial 1: Posidonia Operations



Tutorial 2: News Asset Platform

Content Creation & Editorial workflow



Berlin, 11-13 October 2017



DICE- POSIDONIA DEMONSTRATOR

Ismael Torres

Posidonia
Port Solutions Suite



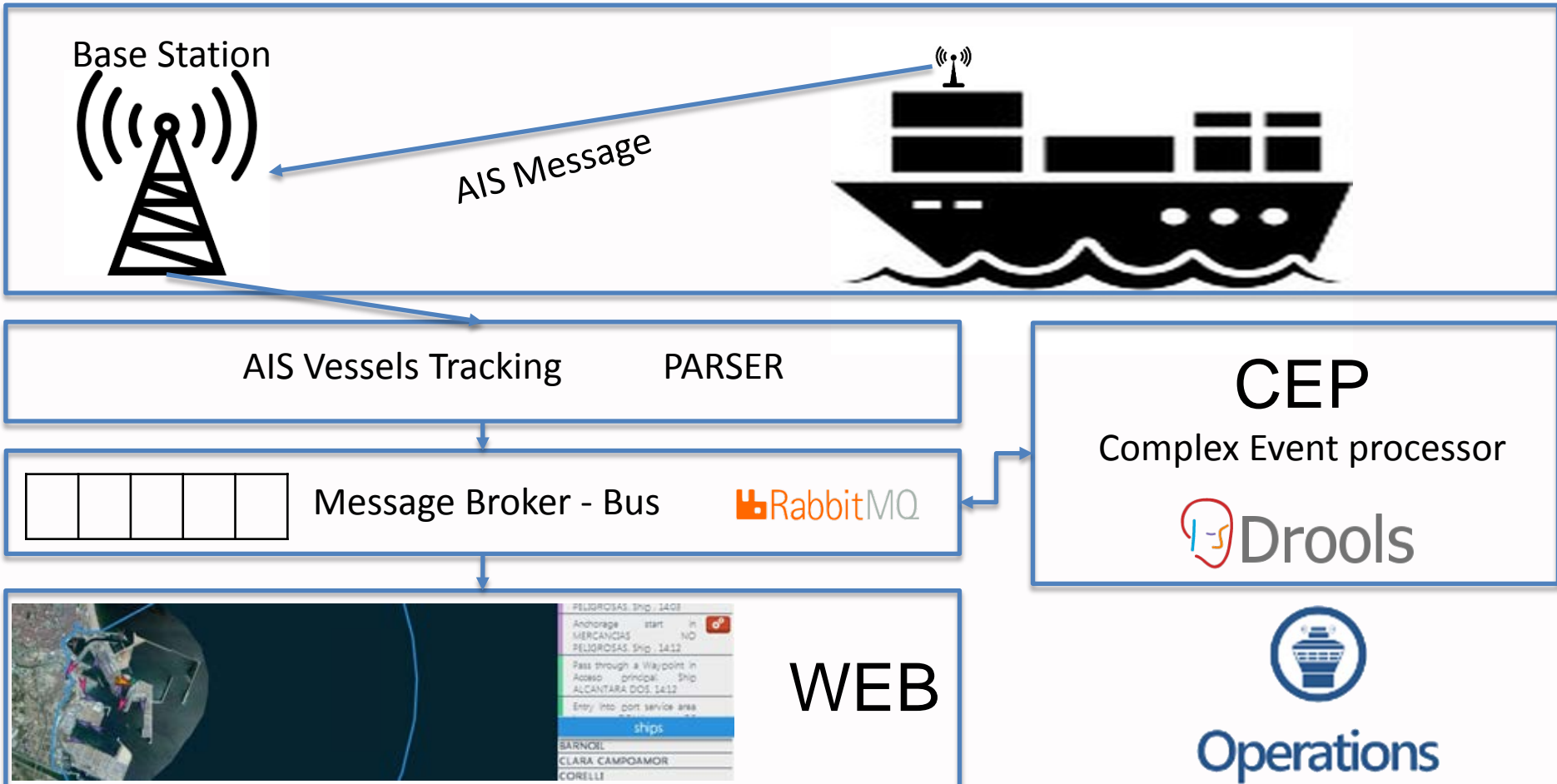
About Prodevelop

- Spanish SME with more than 20 years. 80 engineers
- Sectors (civil service, ports, agrairan, insurances)
- R&D (multiplatform mobility, Geospatial technologies, IoT, software engineering)
- 2011 start intenationalization (sales delegation Casablanca and Sao paulo)

Highly specialized in the Maritime Industry



POSIDONIA OPERATIONS – Architecture Machines



POSIDONIA OPERATIONS

← → ↻ ⓘ 109.231.122.180

Posidonia operations

TEST

ADMINISTRADOR versionnumber

berthings administration simulation

subzone

- VALENCIA
- BARCELONA
- ALGECIRAS
- TANGER
- SINES
- MALTA
- GIOIA TAURO

The screenshot displays the Posidonia Operations web application interface. At the top, there is a navigation bar with the Posidonia logo, a 'TEST' button, and a user profile 'ADMINISTRADOR versionnumber'. Below the navigation bar, there are tabs for 'berthings', 'administration', and 'simulation', with 'berthings' selected. A 'subzone' dropdown menu is open, showing a list of port locations including VALENCIA, BARCELONA, ALGECIRAS, TANGER, SINES, MALTA, and GIOIA TAURO. The main content area features a map of the Valencia port area, showing various berthing zones and ship movements. On the right side, there is a 'MAP' button and two panels: 'events' and 'ships'. The 'events' panel lists several events such as 'Anchorage ended in FONDEO ARES EXTERIOR, Ship XABROAL 9:03' and 'Berthing in 185, Ship RIOBAO UNO : 8:35'. The 'ships' panel lists various ship names like ALICE, AMAZON BEAUTY, AMILCAR, AMMAN, ANTINA G, APOLLONAS, AVE CAESAR, BAHAMA MAMA, BAHIA UNO, BENEDIKT RAMBOW, BUKA, C/DE MALAGA, CABALLERO, CALABRIA, CASTELO DE SINES, and CERVANTES SAAVEDRA. A 'SEARCH' button is located at the bottom of the ships list.

Posidonia Operations

- 18 ports monitored
- 2 cloud deployments
- 5 on-premises
- +30 artifacts running



POSIDONIA OPERATIONS – New Product

Goal for 2018

- New Product "POSIDONIA Operations ON THE CLOUD"

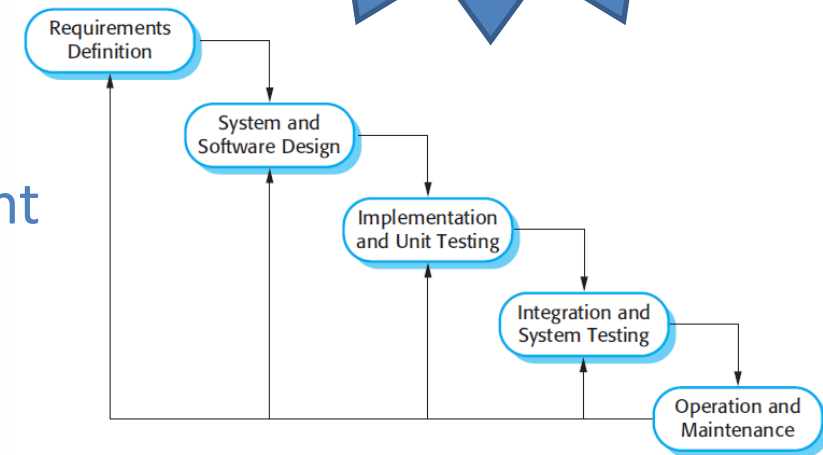
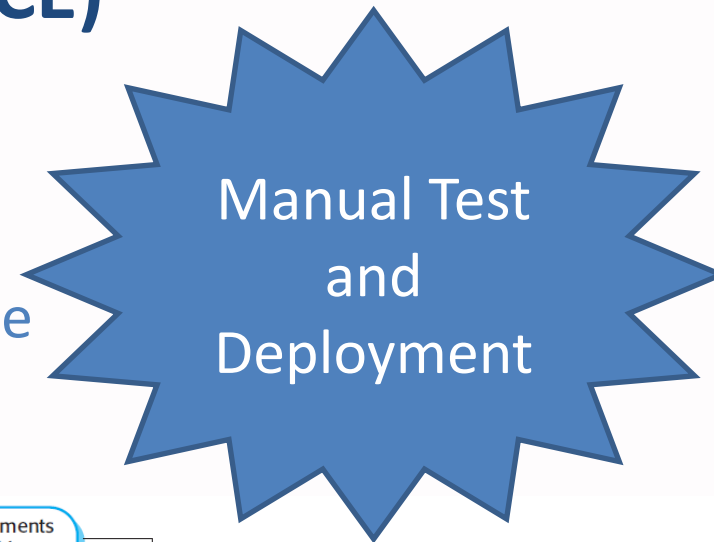


Main reasons of using DICE

1. Automatic deployment on the cloud
2. Automatic configuration
3. Improve business rules validation (semi-automatic)

HOW WE MADE IT (without DICE)

1. We take some AIS logs
2. Develop a new CEP geo-fencing rule
3. Evaluate (**manually**) the correctness rate
4. Evaluate (**manually**) the performance
5. Integrate and deploy in pre-prod
6. Check the throughput (**manually**)
7. Make some fixes in the rule
8. **Iterate** from 3. until pro deployment



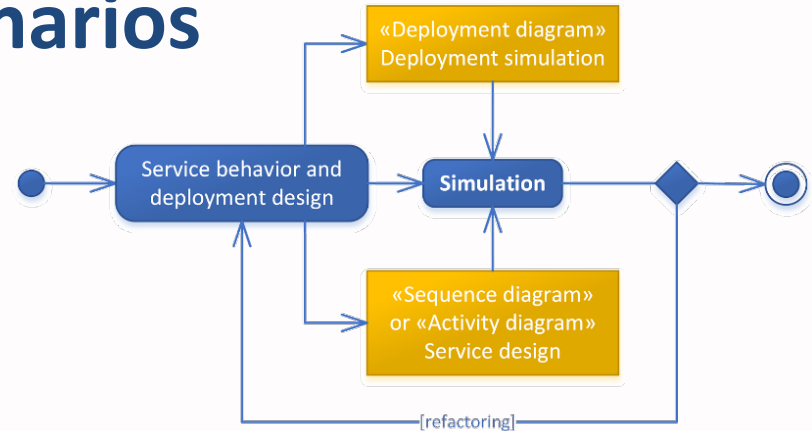
POSIDONIA OPERATIONS - Challenges

- **Continuous delivery**
 - Continuous evolutionary maintenance
 - Different deployments and port configurations
- **Test - Quality**
 - Ensure reliability of the results, performance, etc.
 - Calculate the maximum throughput to evaluate scalability (*what happens if the marine traffic increase? what if we add a new rule?*)
 - Easily create test deployments to improve quality testing (*now, manually and operator dependent*)
 - Monitor system and application metrics to evaluate performance (*need of application metrics*)
 - Calculate correctness rate to evaluate reliability (*are we detecting the events correctly?*)

DICE METHODOLOGY- Scenarios

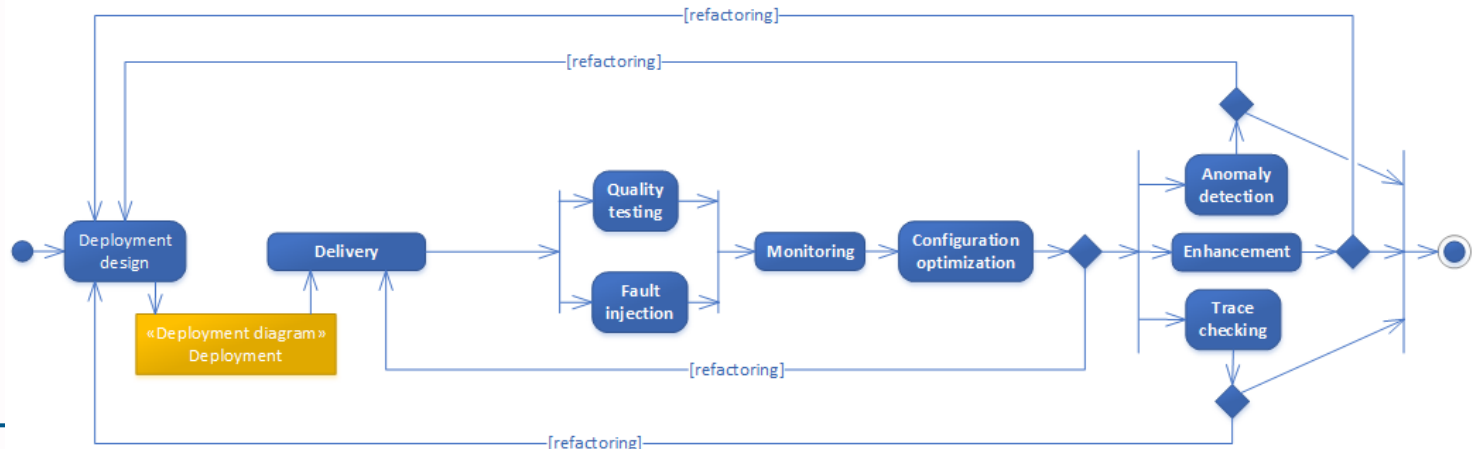
Standalone: Simulation

“I wish to focus on a specific DICE tool”



Partial Devops: Deployment + Fault Injection + Monitoring

I want to quickly deploy and tune the DIA



Simulation Tool – Problem to solve

- What happens if we add a new rule/increase marine traffic?
- Which is the maximum throughput of the current configuration?

Simulation Tool – Model

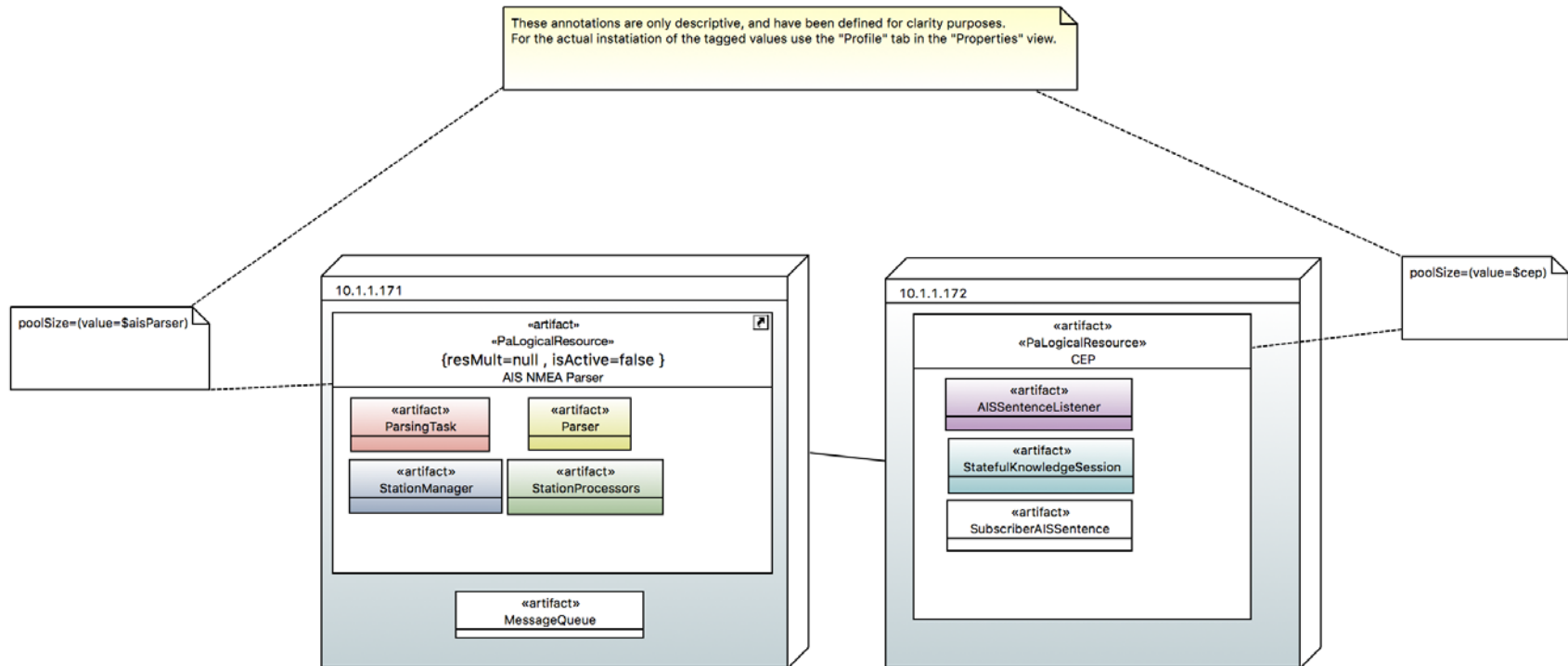
Use a DPIM (Platform Independent Model) to specify the system.

- MDA-UML Models enriched with DICE Profile.

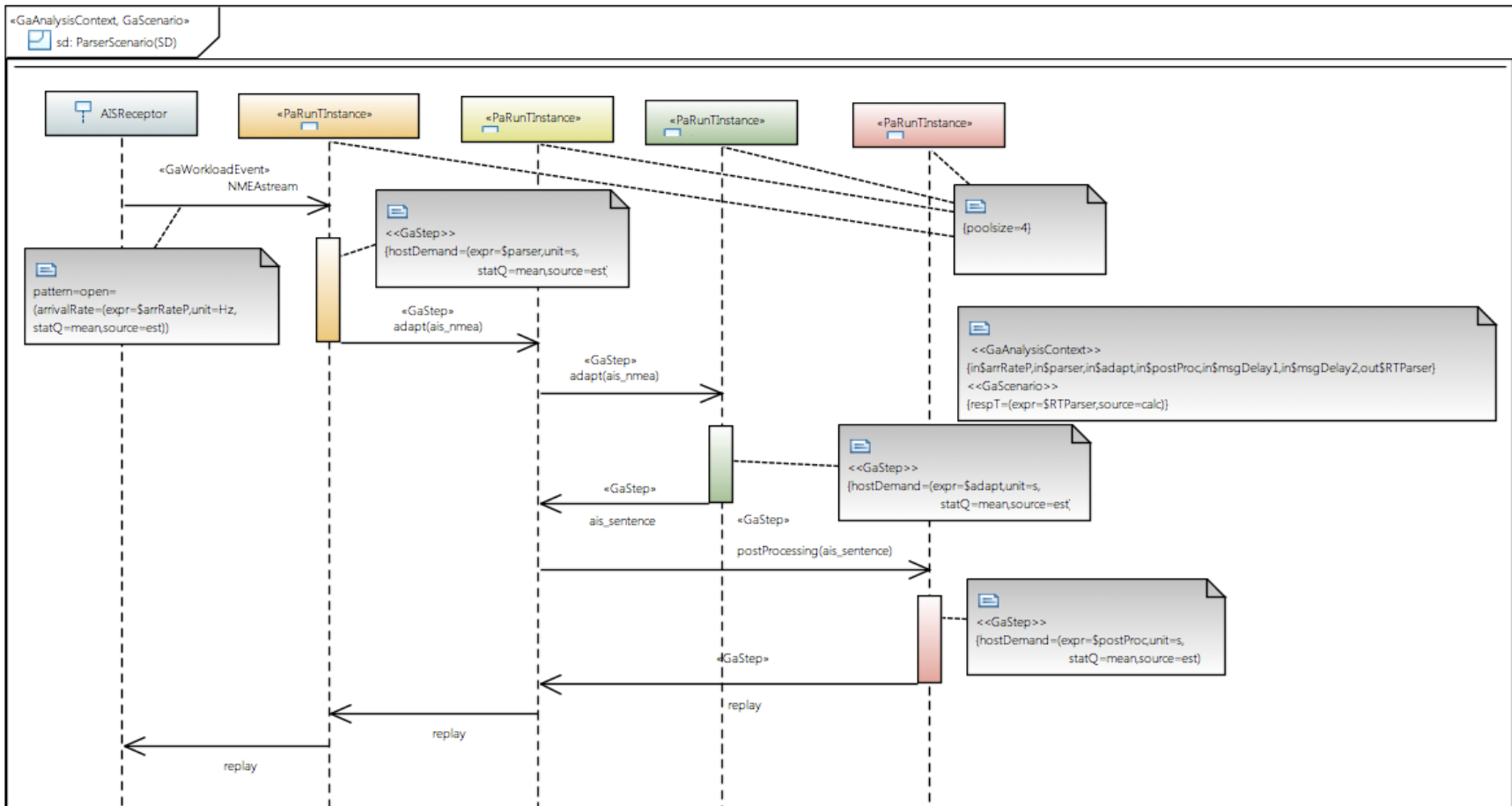
Given a DPIM model, the simulation tool tells us the maximum throughput

- The model specifies:
 - Messages per second
 - Number of AIS Parser and CEP instances
 - Number and cost of each rule

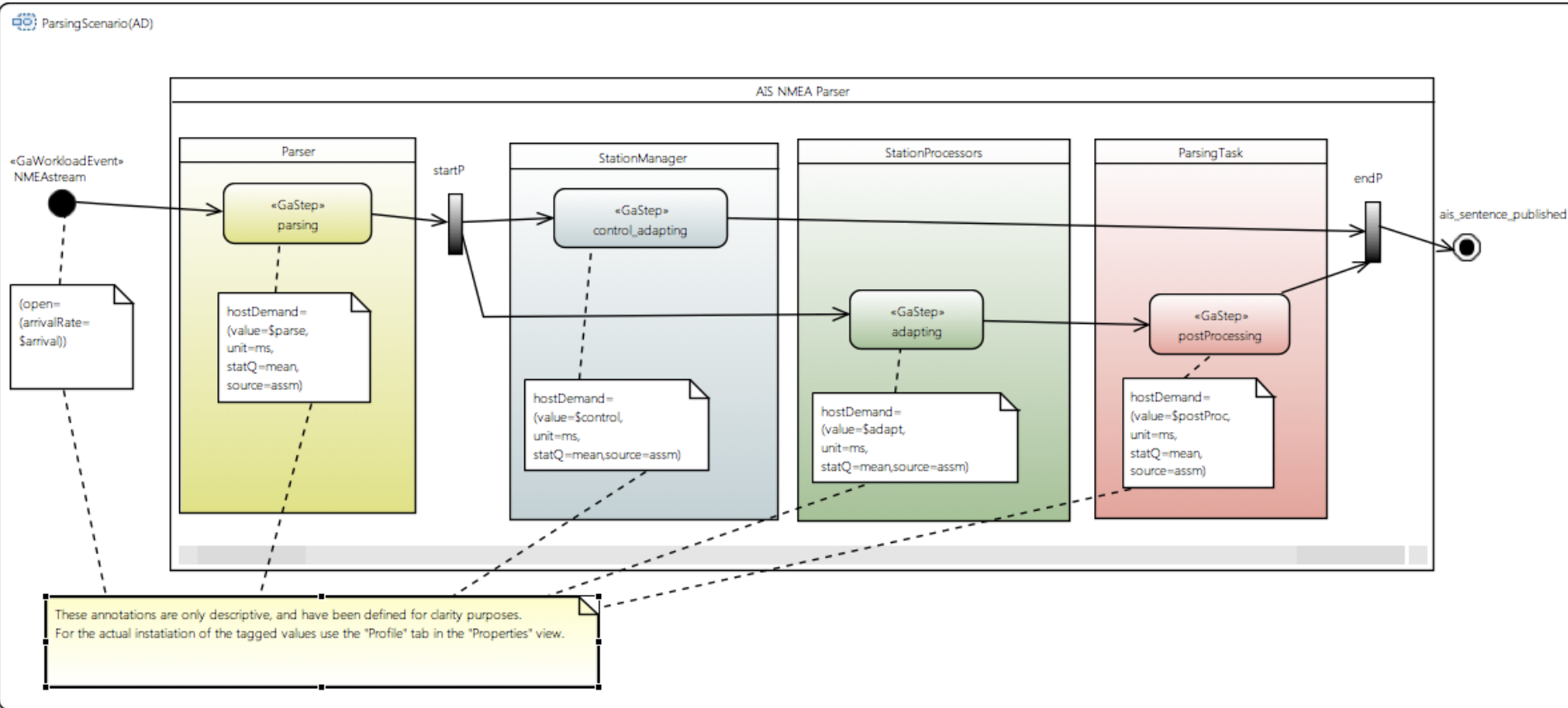
Simulation Tool – Model



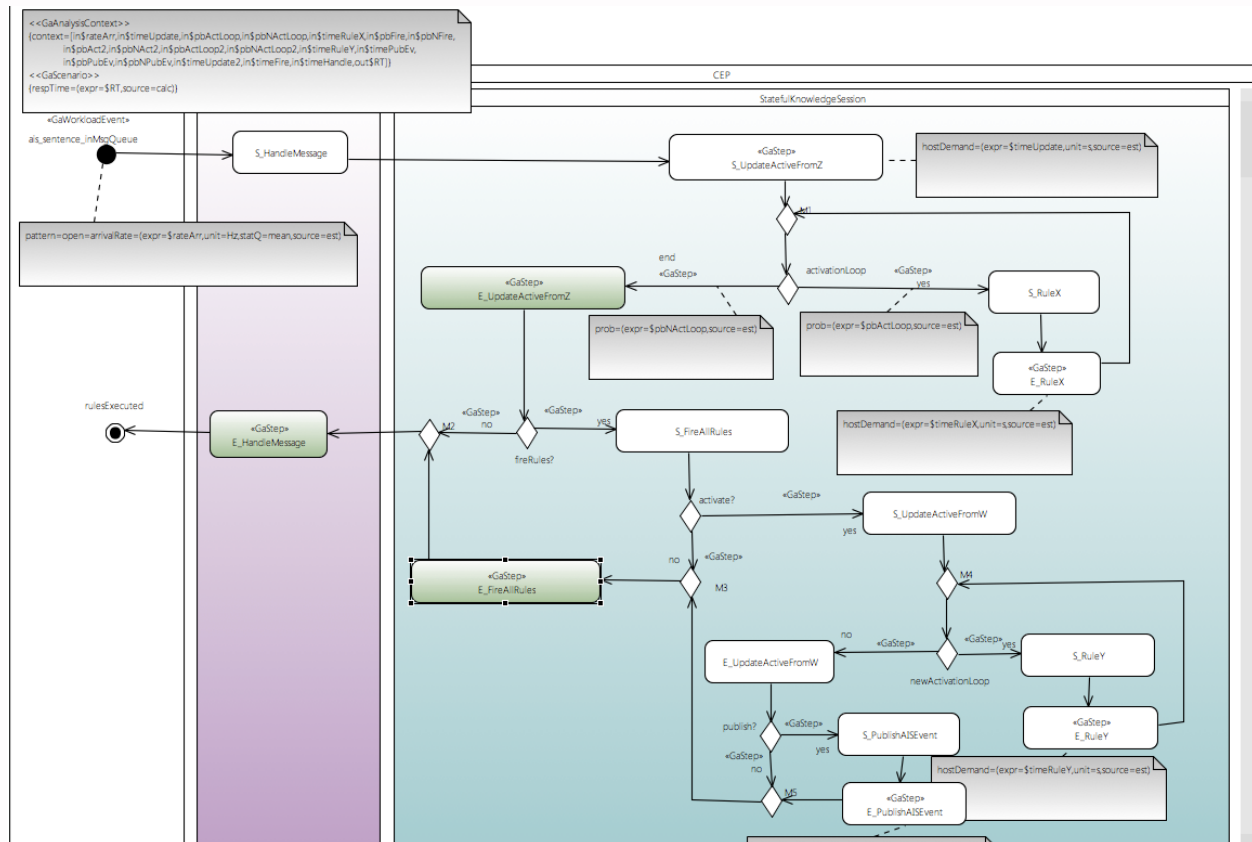
Simulation Tool – Model



Simulation Tool – Model



Simulation Tool – Model



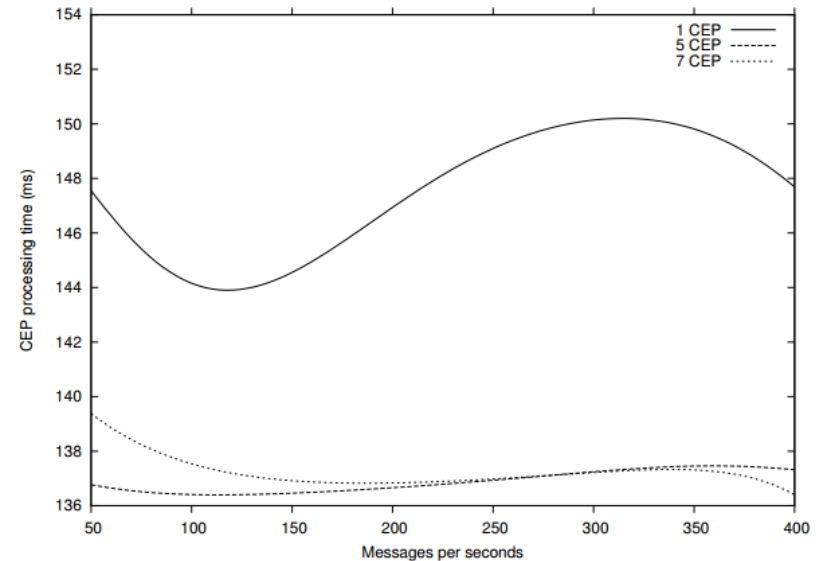
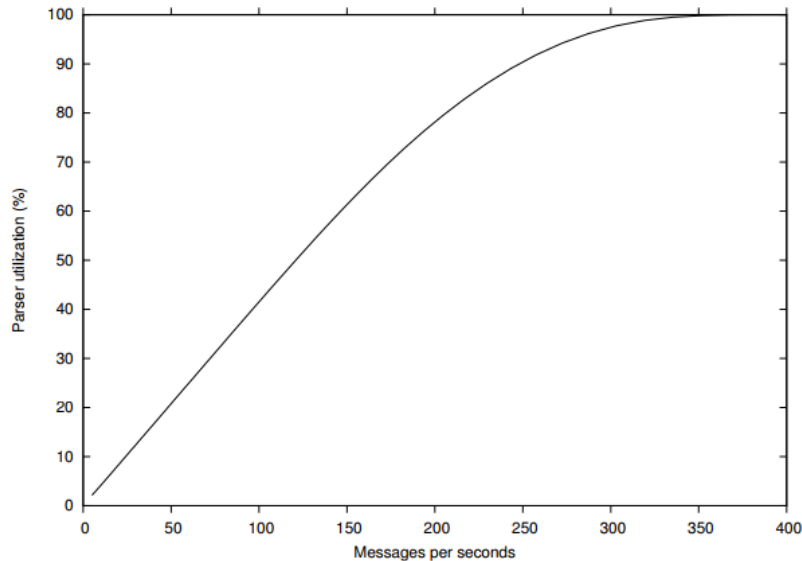
Simulation Tool – Configuration

- SIMULATION PARAMETERS:
 - Cost estimation of each step (the actual cost is produced by the monitoring tool)
 - Message rate

The screenshot shows a software configuration window with a sidebar on the left and a main content area. The sidebar includes a 'parsing' tab and a list of categories: UML, Comments, Marte, Profile, Style, Appearance, Rulers And Grid, and Advanced. The main content area is divided into two panels. The left panel, titled 'Applied stereotypes:', contains a list of stereotypes with checkboxes and expandable icons. The 'hostDemand' stereotype is selected and highlighted in blue. The right panel, titled 'hostDemand', contains a text box with the expression: `(expr=$rate, statQ=mean, source=est)`. Both panels have control icons (up, down, add, delete, edit) at the top right.

Simulation Tool – Configuration

- OUTPUT:
 - Throughput



Simulation Tool – Results

- Scalability prediction at design time
- Iterative enhancement
- Multiple configurations
- Maximum throughput

Simulation Tool – Results

Main objectives of using DICE

1. Automatic deployment on the cloud
2. Automatic configuration
3. Improve business rules validation (semi-automatic)

- GOALS ACHIEVED:

- PO.1 Simulation and predictive analysis of new event detection rules
- PO.2 Scalability analysis

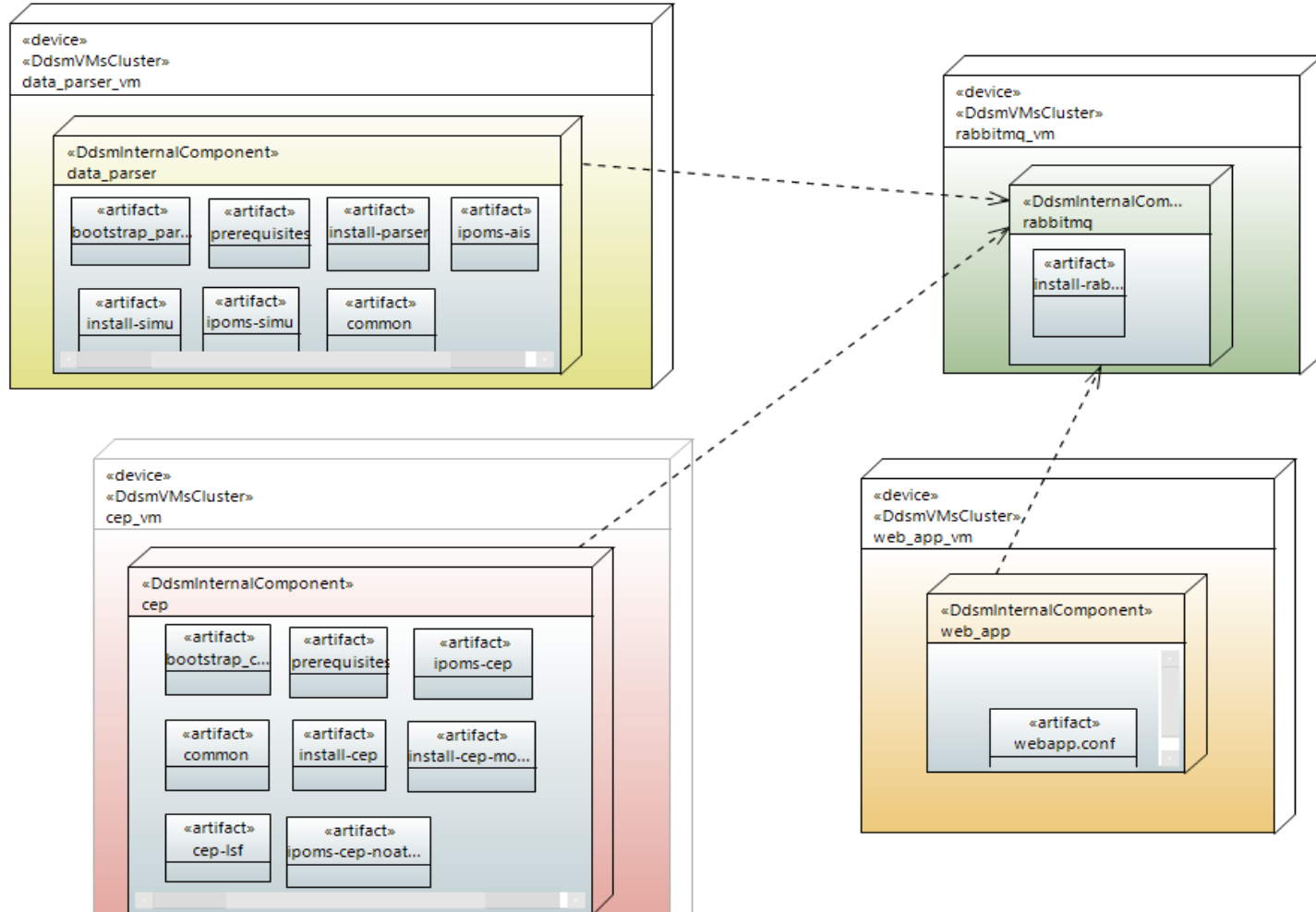
DICER-DEPLOYMENT- Problem to solve

- How to reduce the time required to deploy the application?
- how to make deployment easier and not require much technical knowledge?

DICER-DEPLOYMENT - Model

- **The Deployment Model (DDSM) is based on the simulation results.**
 - Models the physical deployment of artifacts on nodes
- **The model specifies:**
 - Number of nodes (virtual machines)
 - Number of instances of applications (AIS parser, CEP, Rabbit)
 - Network configuration, installation scripts, etc.
- **These tools allow an automatic deployment of the solution, which is essential to perform automatic testing**

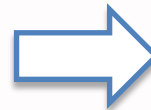
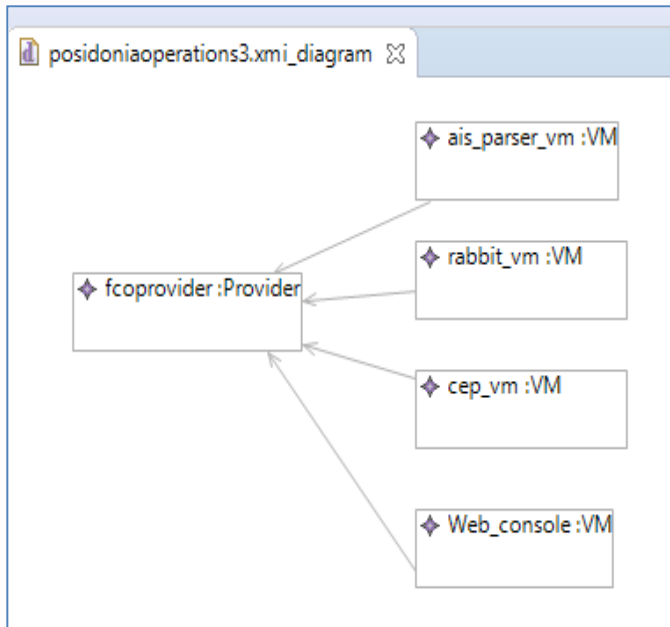
DICER-DEPLOYMENT



Posidonia - DDSM

DICER-DEPLOYMENT

IDE – DICER



Model to Text
Transformation

```
tosca_definitions_version: cloudify_dsl_1_3

imports:
  - http://dice-project.github.io/DICE-Deployment-Clou

node_templates:

  #Data_parser
  data_parser_vm:
    type: dice.hosts.Medium
  data_parser:
    type: dice.components.misc.ScriptRunner
    properties:
      script: scripts/run-install-data-parser.sh
      language: bash
      arguments:
        - get_attribute: [ rabbitmq, fqdn ]
    resources:
      relationships:
        - type: dice.relationships.ContainedIn
          target: data_parser_vm
        - type: dice.relationships.Needs
          target: rabbitmq

  # RabbitMQ
  rabbitmq_firewall:
  rabbitmq_vm:
    type: dice.hosts.Medium
    relationships:

  rabbitmq:
    type: dice.components.misc.ScriptRunner
    properties:
    relationships:

  # cep_vm
  cep_vm:
    type: dice.hosts.Medium

  cep:
    type: dice.components.misc.ScriptRunner
    properties:
    relationships:

outputs:
```

MODEL- DDSM

BLUEPRINT

DICER-DEPLOYMENT

DEPLOYMENT

109.231.122.194

DICE Deployment Containers Hello, dice. Logout

[+ New Container](#)

prode_20170221_3full_1 3ef6c34e-wr12-47d3-wed2-c9201238564

[Undeploy Blueprint](#) [Redeploy](#) [Upload Blueprint](#)

✓ Blueprint successfully deployed
Since Wednesday, February 22, 2017 9:00 AM

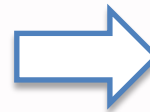
Output Name	Value	Description
rabbitmq_ip	109.231.122.154	IP of the RabbitMQ's host
data_parser_ip	109.231.122.181	IP of the data parser's host
cep_ip	109.231.122.233	IP of the CEP's host

Blueprint ID: c-cf7e4b08-f0d1-4d2b-b32f-4805ef8aa380

```

1 #!/usr/bin/env bash
2 #
3 # This script is used to deploy the DICE containers.
4 #
5 # It will create a VDC, create the containers, and
6 # start them.
7 #
8 # Usage: ./deploy.sh
9 #
10 # Options:
11 # -h: Show this help message
12 # -v: Verbose mode
13 # -d: Debug mode
14 # -s: Skip the VDC creation
15 # -i: Skip the container creation
16 # -t: Skip the container start
17 #
18 # Environment variables:
19 # DICE_VDC_NAME: The name of the VDC to create
20 # DICE_CONTAINER_NAME: The name of the container to create
21 # DICE_CONTAINER_IP: The IP address of the container
22 # DICE_CONTAINER_CPU: The number of CPUs for the container
23 # DICE_CONTAINER_RAM: The amount of RAM for the container
24 # DICE_CONTAINER_DISK: The amount of disk space for the container
25 #
26 #
27 #
28 #
29 #
30 #
31 #
32 #
33 #
34 #
35 #
36 #
37 #
38 #
39 #
40 #
41 #
42 #
43 #
44 #
45 #
46 #
47 #
48 #
49 #
50 #
51 #
52 #
53 #
54 #
55 #
56 #
57 #
58 #
59 #
60 #
61 #
62 #
63 #
64 #
65 #
66 #
67 #
68 #
69 #
70 #
71 #
72 #
73 #
74 #
75 #
76 #
77 #
78 #
79 #
80 #
81 #
82 #
83 #
84 #
85 #
86 #
87 #
88 #
89 #
90 #
91 #
92 #
93 #
94 #
95 #
96 #
97 #
98 #
99 #
100 #
    
```

Upload



flexiant Cloud Orchestrator Resources Alberto Romeu DICE

Dashboard Servers Disks Networks Deployments Snapshots Firewalls Images Blueprints VDCs Load balancers

Resources

- Dice, Fault, Injection, VM (1) 4 GB / 2 CPU
- PCP Agent Server (1) 2 GB / 2 CPU
- DarrenTest2 (1) 2 GB / 2 CPU
- DarrenCluster2Test (1) 4 GB / 3 CPU
- Darrentest (1)

Servers

Server name	VDC name	CPU	RAM (MB)	IP address	Status text with
<input type="checkbox"/> c-cf7e4b08-f0d1-4d2b-b32f-4805ef8aa380	*_CloudifyManag	1	2048	109.231.122.154	▶ Running
<input type="checkbox"/> c-cf7e4b08-f0d1-4d2b-b32f-4805ef8aa380	*_CloudifyManag	1	2048	109.231.122.233	▶ Running
<input type="checkbox"/> c-cf7e4b08-f0d1-4d2b-b32f-4805ef8aa380	*_CloudifyManag	1	2048	109.231.122.181	▶ Running

Automatic Deployment

DICER-DEPLOYMENT – Results

- Deployment from scratch: 5 hours (it includes the initial learning curve, create the model, etc.)
- Next deployments: 20 minutes
- Effective time for a new deployment: minutes (automation)
- Continuous Deployment (productivity)
- Deployment diagrams (helps understand the system)
- Automatic Deployment (No experts needed, less error, faster, less cost)

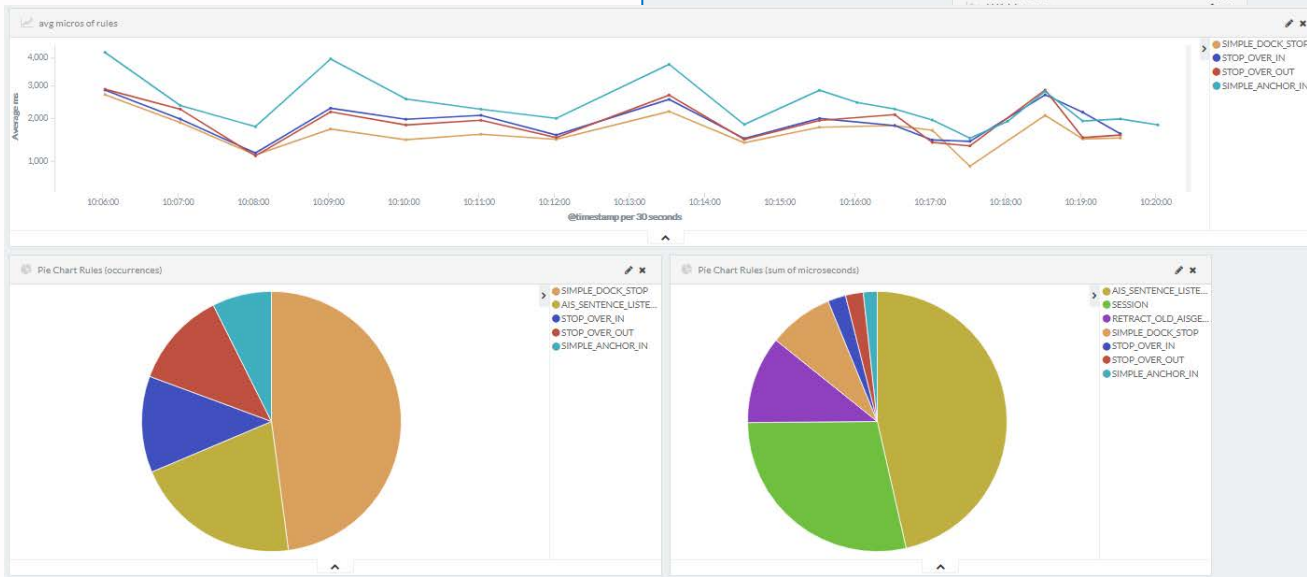
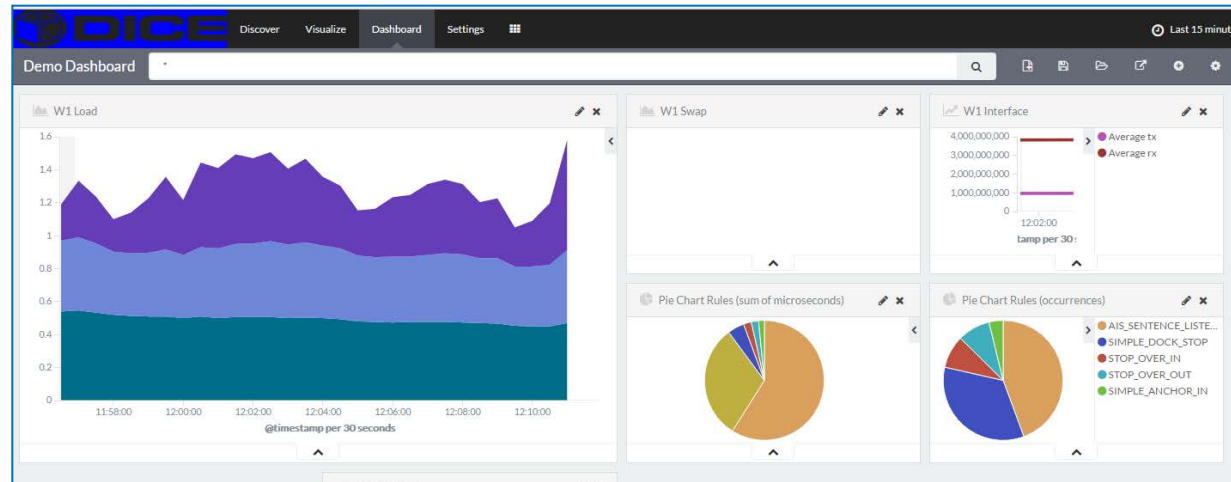
Monitoring- Problem to solve

- How to control the resources used in Real time?
- How to control the Posidonia throughput in real time?
- How to analyze vessels routes inside the port?
- How to use the data obtained to improve the system and to make test?

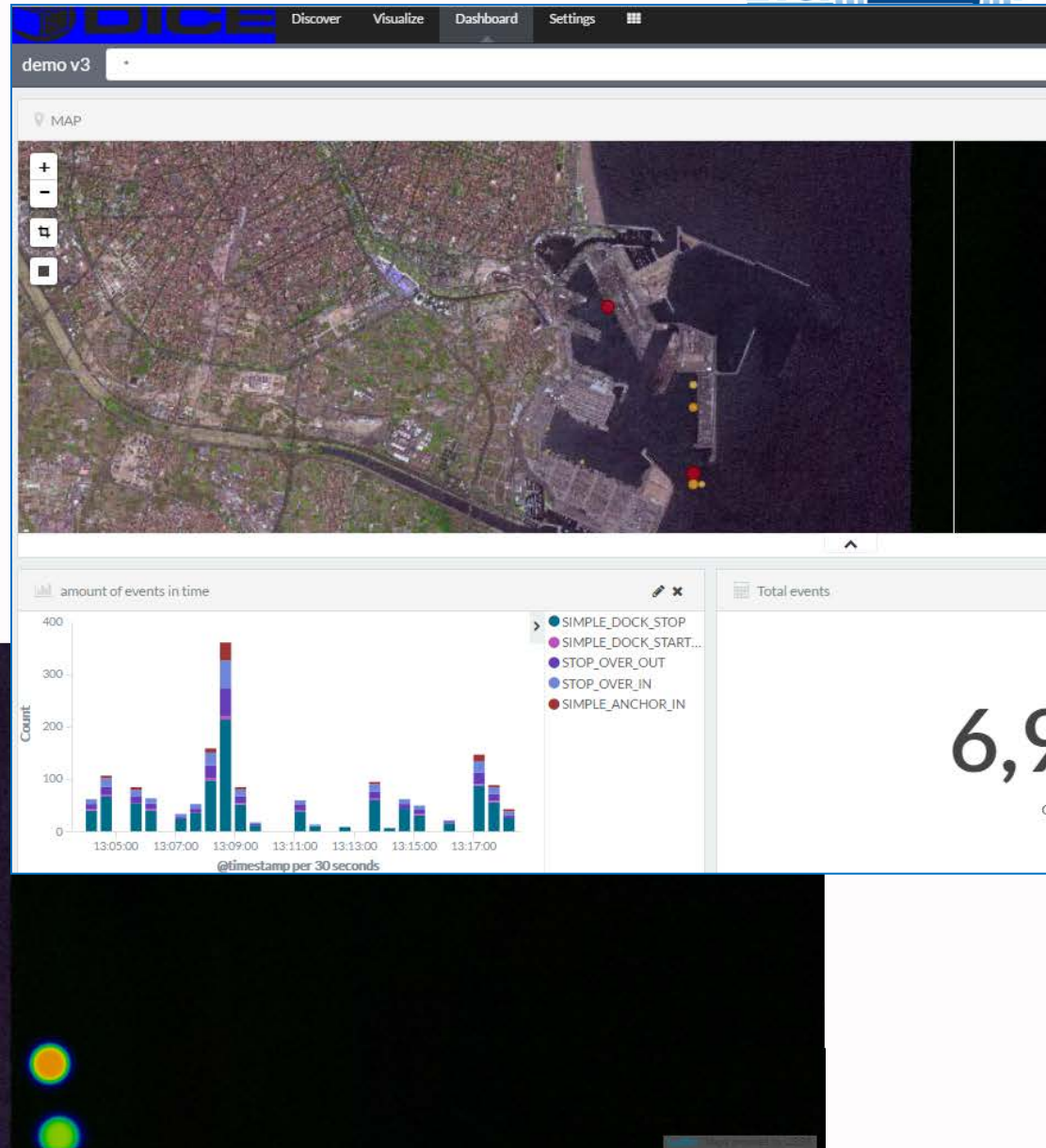
Monitoring

- DICE monitoring platform (DMon) collects, stores, indexes and visualizes monitoring data in real-time from applications running on Big Data frameworks.
- Monitoring Posidonia Operations “running” provides performance metrics.
- These metrics can be used to redesign the simulation model to improve the results

Monitoring



Monitoring



Monitoring – Results

- Control about resources used in real time
- Use log files to visualise/control the system in real time
- Use log files as an input for other DICE tools to automatize some test and improve the quality of the system

Fault injection tool- Problem to solve

- What happens to our product, if the system is overload?
- What happens if one of our components stops?

Fault injection tool

- DICE Fault Injection tool (FIT) is used to generate faults within VM.
- FIU installs a third party tool in the VM to produce faults.

DICE FIT High CPU

Fault Started : 109.231.122.157 ubuntu100 100

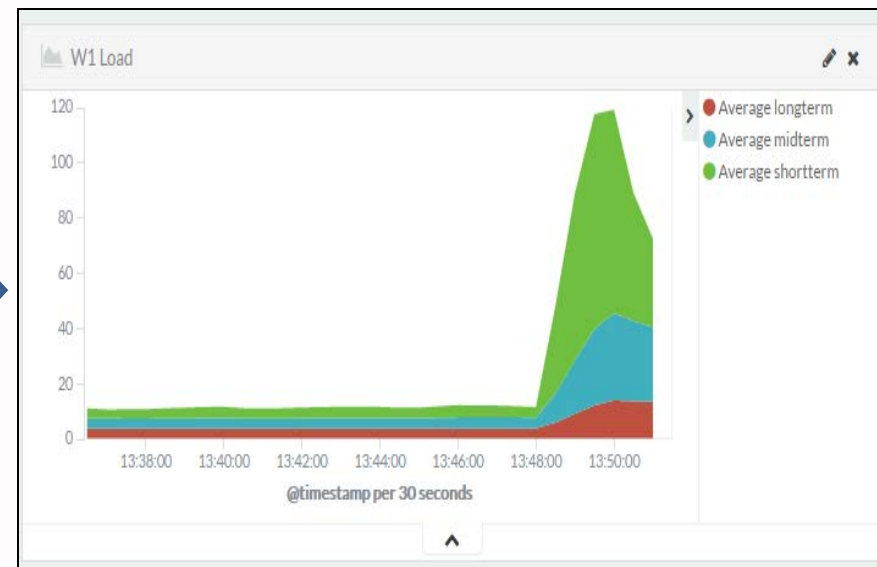
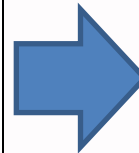
IP Address *

Username *

Password *

CPU count *

Time *



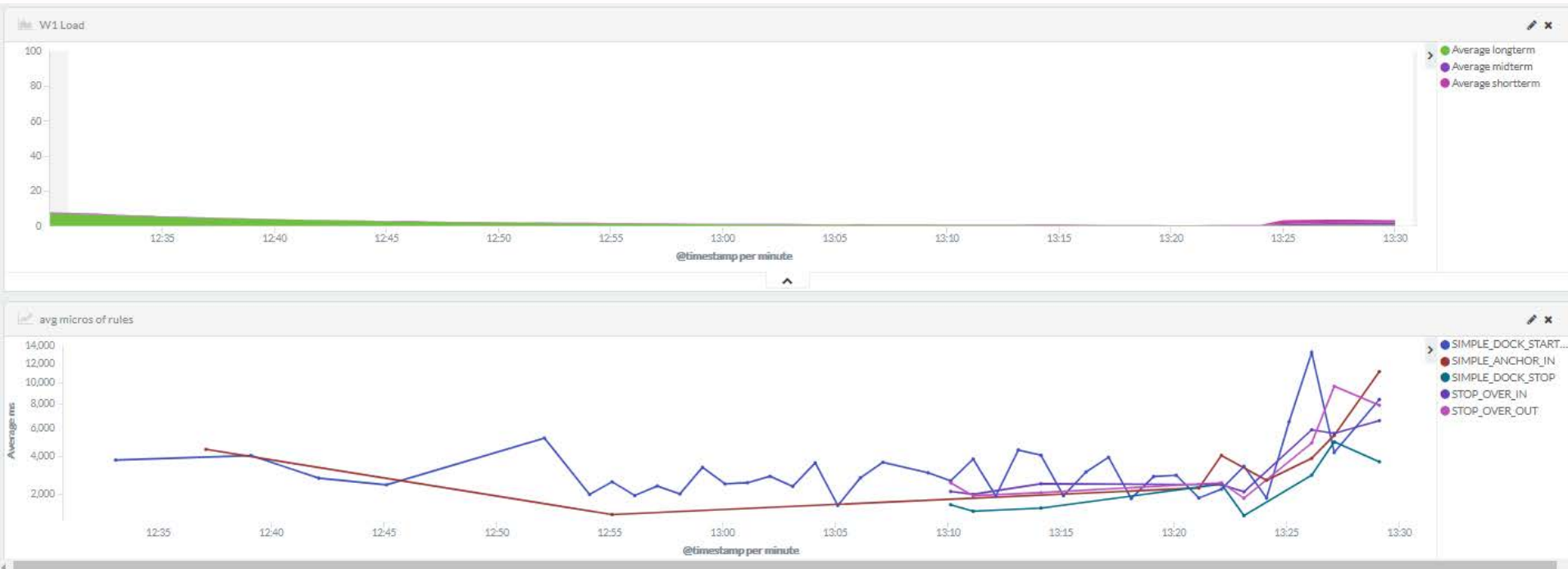
Fault injection tool

- CPU load



Fault injection tool

- Memory Load



Fault injection tool - Results

- Test the system behaviour generating different types of faults (CPU and memory overloads)

Anomaly detection tool - Problem to solve

- *Is the time required to execute the different rules normal?*
- *Is there any anomaly in the detection of events?*

Anomaly detection tool

- **Anomaly detection** is the identification of items, events or observations which do not conform to an **expected pattern** or other items in a dataset.
- Anomaly detection is used in the UC to detect anomalies related with the **processing time required** for the different events (rules) that the system analyses

Anomaly detection tool

Examples of the Rules execution cost

Component	key	method	ms	ship
AIS_SENTENCE_LISTENER	2017-06-22T11:53:04.278Z	HANDLE_MESSAGE	209	211636100
SIMPLE_ANCHOR_OUT	2017-06-22T11:53:04.272Z	UPDATE_ACTIVE	710	305965000
STOP_OVER_OUT	2017-06-22T11:53:04.271Z	UPDATE_ACTIVE	652	305965000
SIMPLE_DOCK_STOP	2017-06-22T11:53:04.270Z	UPDATE_ACTIVE	293	305965000
STOP_OVER_IN	2017-06-22T11:53:04.270Z	UPDATE_ACTIVE	680	305965000
SIMPLE_DOCK_STOP	2017-06-22T11:53:04.269Z	UPDATE_ACTIVE	295	305965000
AIS_SENTENCE_LISTENER	2017-06-22T11:53:04.263Z	HANDLE_MESSAGE	9483	305965000
SIMPLE_ANCHOR_OUT	2017-06-22T11:53:04.262Z	UPDATE_ACTIVE	755	305965000
STOP_OVER_OUT	2017-06-22T11:53:04.261Z	UPDATE_ACTIVE	618	305965000
STOP_OVER_IN	2017-06-22T11:53:04.260Z	UPDATE_ACTIVE	686	305965000
SIMPLE_DOCK_STOP	2017-06-22T11:53:04.259Z	UPDATE_ACTIVE	303	305965000
AIS_SENTENCE_LISTENER	2017-06-22T11:53:04.246Z	HANDLE_MESSAGE	199	225366000
AIS_SENTENCE_LISTENER	2017-06-22T11:53:04.245Z	HANDLE_MESSAGE	766	224161160

Definition of anomalies

AIS_SENTENCE_LISTENER ms > 20000
 RETRACT_OLD_AISGEOMDATA ms > 60000
 SIMPLE_ANCHOR_IN ms > 30000
 SIMPLE_ANCHOR_OUT ms > 2000
 SIMPLE_DOCK_STOP ms > 2000
 STOP_OVER_IN ms > 2000
 STOP_OVER_OUT ms > 2000
 FIRE_ALL_RULES ms > 30000
 HANDLE_MESSAGE ms > 30000
 RETRACT ms > 30000

Anomaly detection tool - Results

- The ADT detects anomalies related with the cost execution, this cost impact directly in the performance of the system.
- The ISF method did not detect all the anomalies (15.5 % from the original 22.4 %) but it had a relatively small false positive count (accuracy of 93.4%).
- It is possible in increase the accuracy of the method by considering a bigger set of data.

Metric	CEP
Labelled anomalies	1447
Detected anomalies	999
False positives	58
Good Anomalies	941
Percentage labelled	22,4%
Percentage detected	15,5%
Accuracy	93,4%

CONCLUSIONS – Benefits of using DICE

1. Automatic deployment on the cloud
 - **DICER + Deployment**
 - Fully automated deployment in minutes Automatic configuration
2. Automatic configuration
 - **Simulation + Monitoring:**
 - See the effect of different configuration at Design phase
3. Improve business rules validation (semi-automatic)
 - **Anomaly Detection + Trace Checking Tool**
 - Detect unexpected patterns in the events detected.
 - **Fault Injection Tool**
 - Evaluate evaluate system stability against unexpected failures

Berlin, 11-13 October 2017



THE NEWS ASSET DEMONSTRATOR

Vasilis Papanikolaou, George Giotis

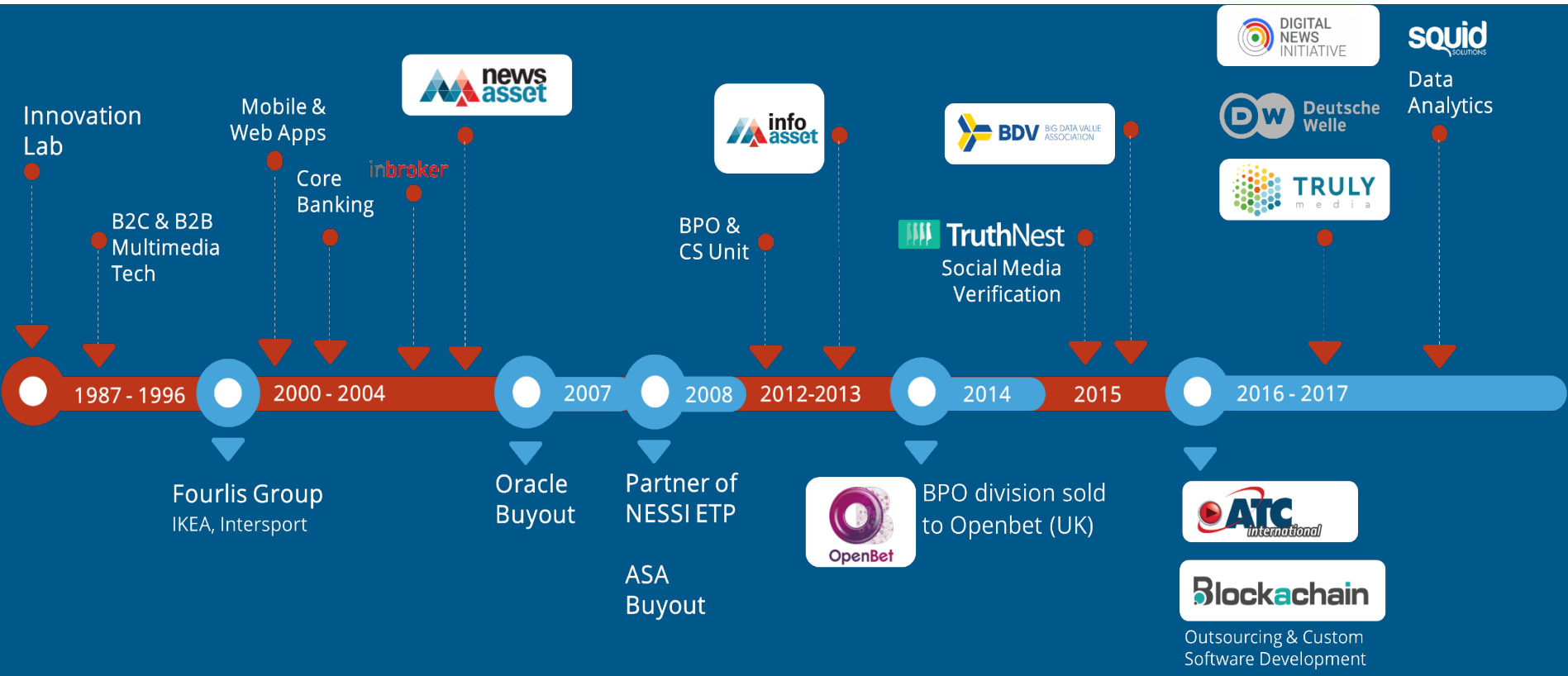
Berlin, 11-13 October 2017



ATHENS TECHNOLOGY CENTER

Who we are

ATC MILESTONES: over the last 30 years



Departments & Activities

Media & Content
Management



Web & Mobile
Applications

Innovation
Lab



Custom IT
Services

Berlin, 11-13 October 2017



NEWS ASSET



What News Asset is

News Asset Platform Description

An end-to-end multimedia cross-channel suite for an evolving News Agency

Handles large volumes of media asset (real time and/or archived news content)

Datasets composed of media items such as text, images, reports, articles, videos, etc.



Concept

Services that supports the whole life-cycle of managing a news item:

- planning,
- creating, gathering and searching,
- editing,
- producing, distributing and archiving



Workflow

Old –fashioned non-cloud software

Based on .Net

Storage intensive

Data intensive

Computation services

Centralized architecture



Keywords

Fat-Client, Business Server and Data Base Server

News reception may vary from 5000 to 20000 items per day

News distribution may be 50000 to 200000 items per day

The default import format is based on standard NewsML format



Technical insight

Challenges

Uncountable media items are produced by heterogeneous sources

Media eye witnessing reports (e.g. Twitter)

Immediate access and management of real time news items is crucial

Handle streams of data, serve a wide range of workloads



Challenges

- Refactoring of the old-fashioned engine
 - related to cloud processing and Big Data technologies
- Reconfiguration
 - revise obsolete architecture with respect to quality-driven metrics
- Manage complexity
 - real-time responsiveness for temporal peaks of high computational demand



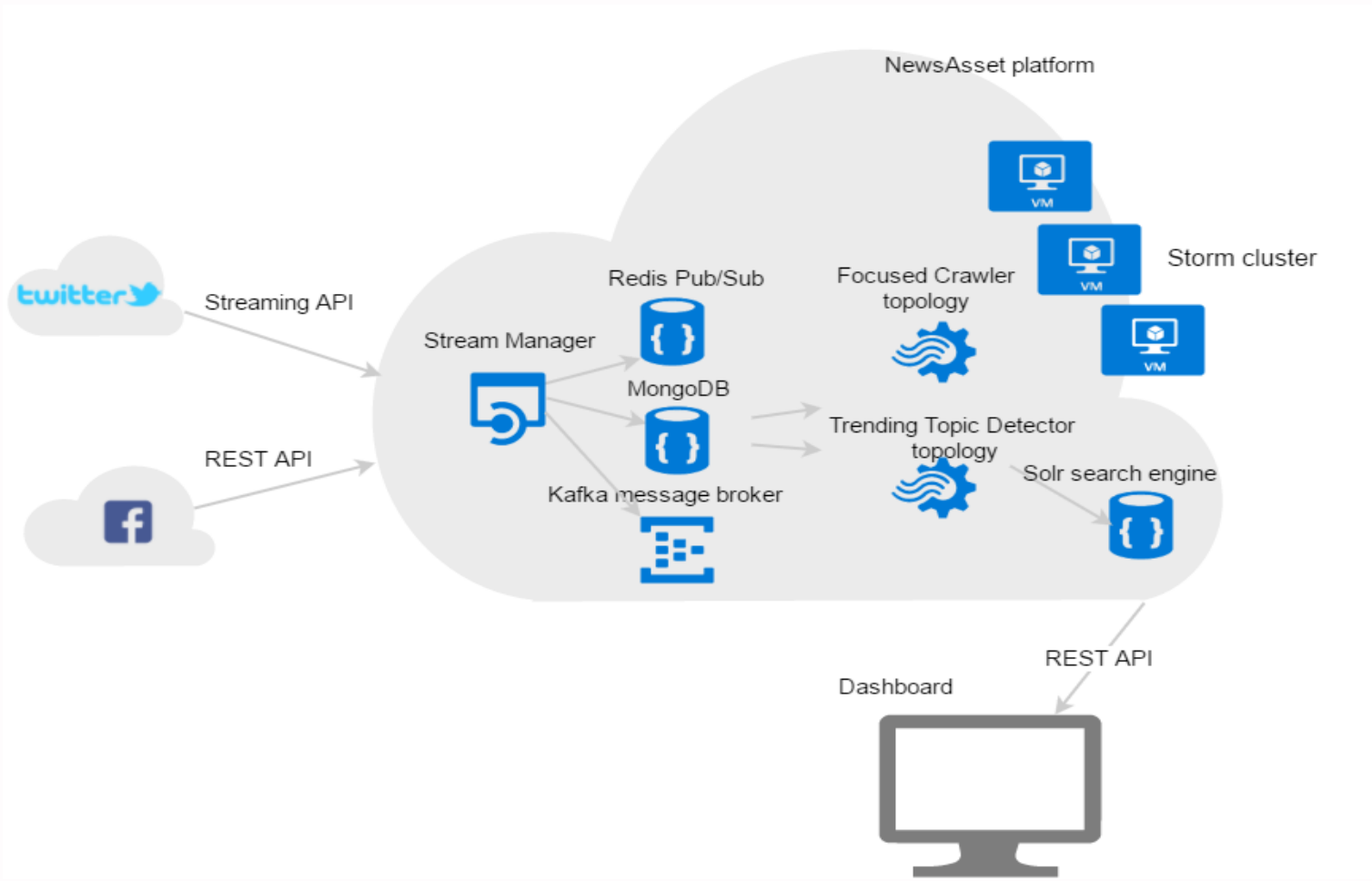
Berlin, 11-13 October 2017



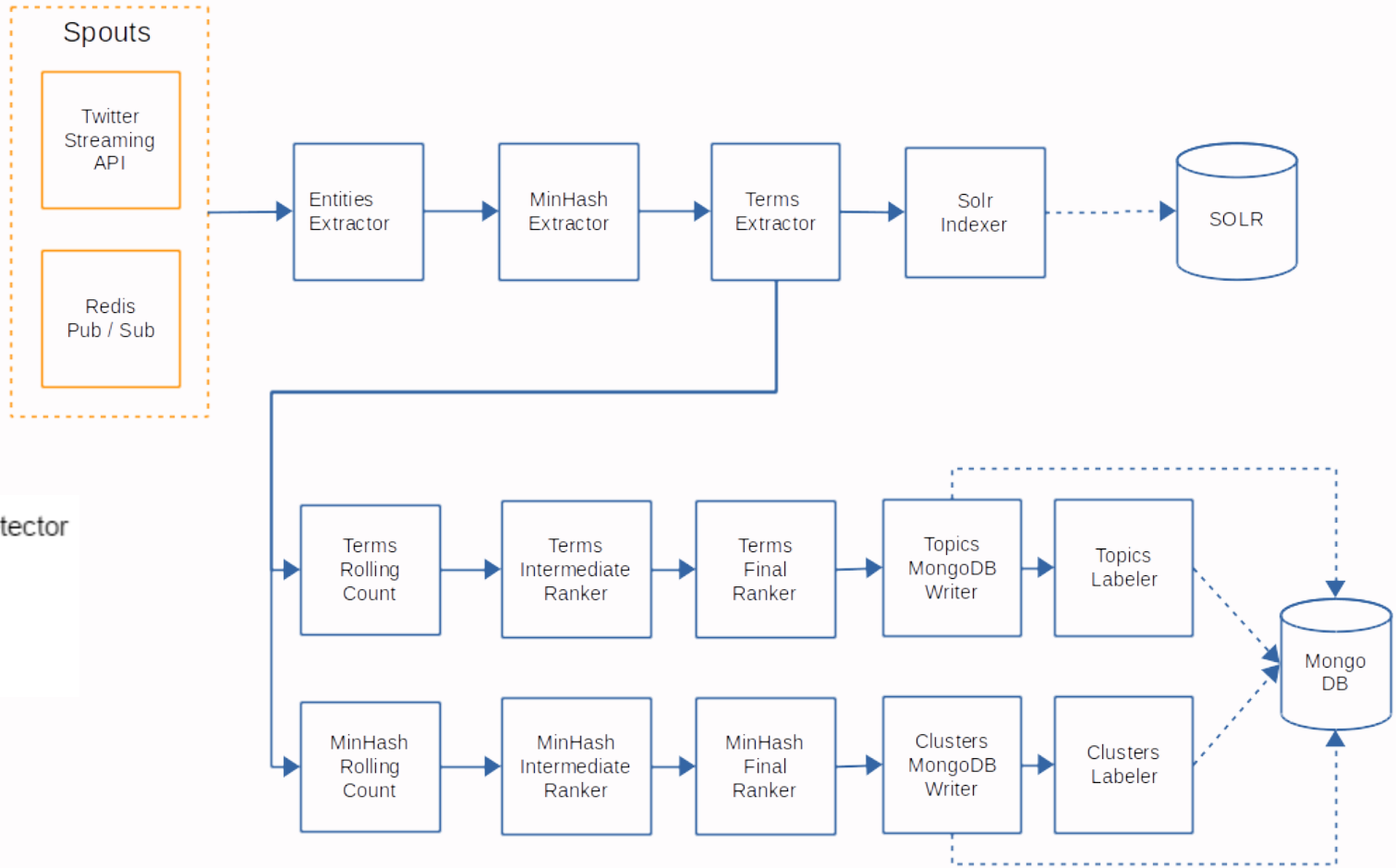
NEWS ORCHESTRATOR

The Solution

News Orchestrator Architecture



Trending-Topic Detector Storm Topology



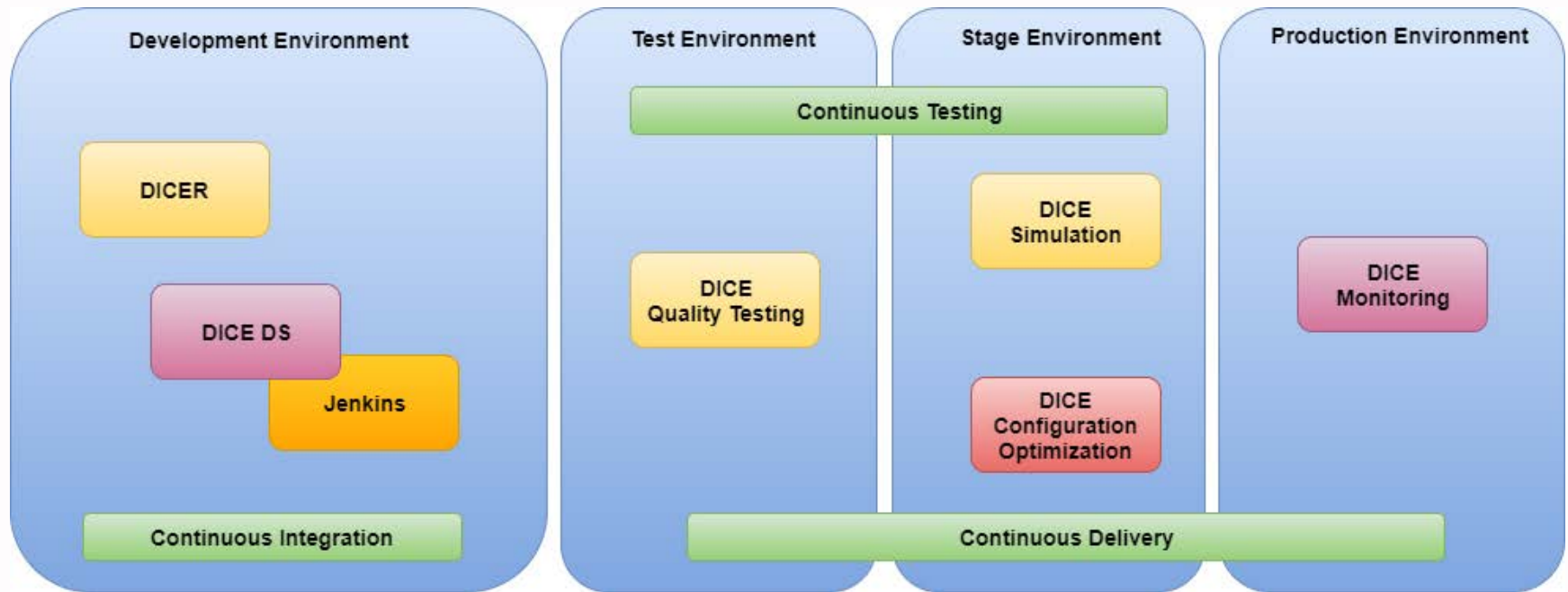
Berlin, 11-13 October 2017



NEWS ASSET TUTORIAL

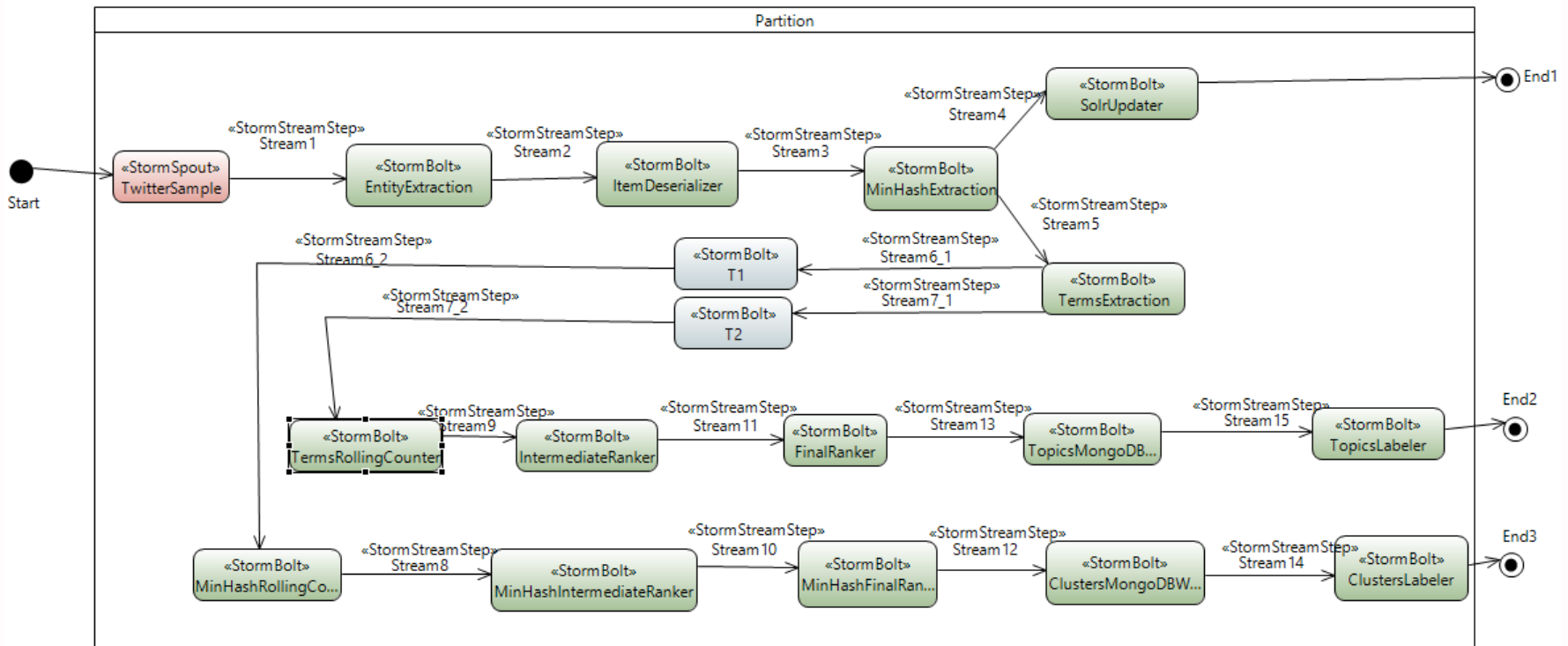
News Asset Experience with DICE

DevOps Lifecycle

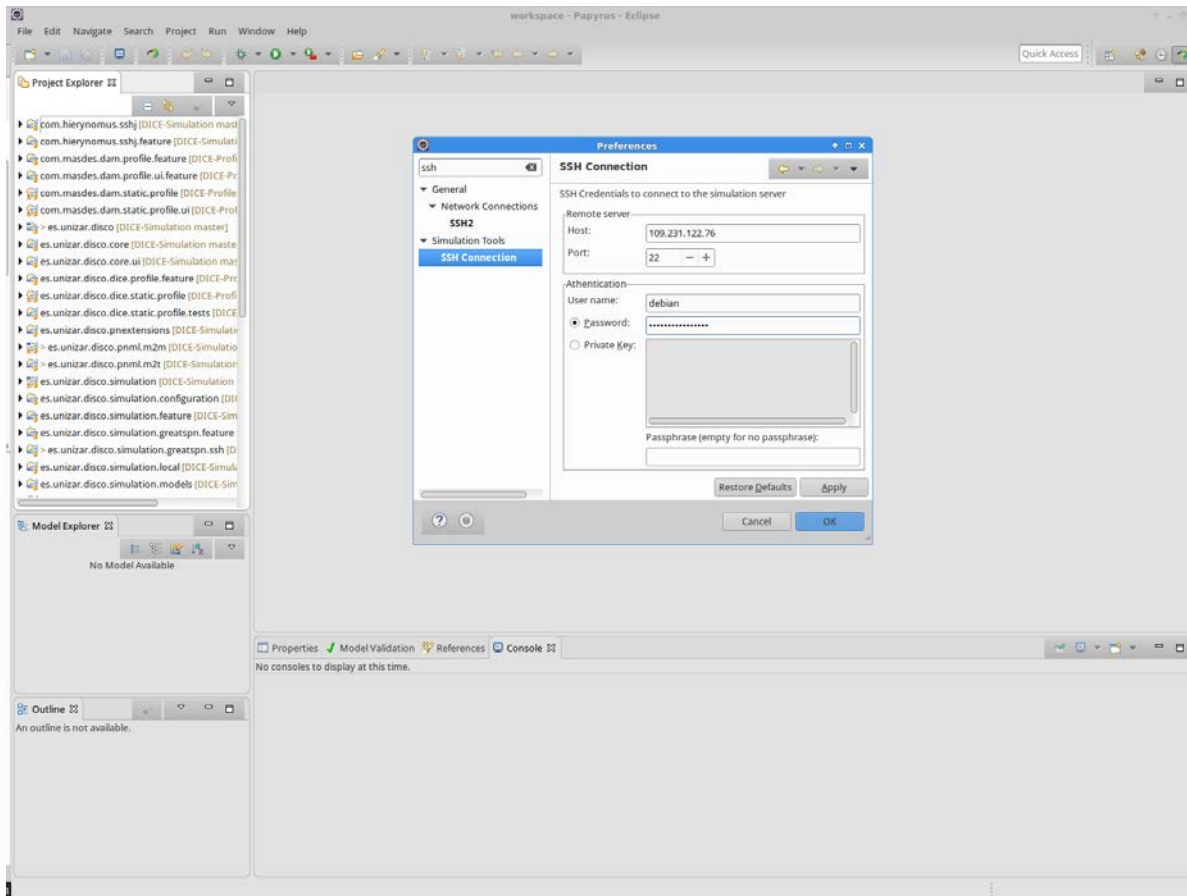


Simulation

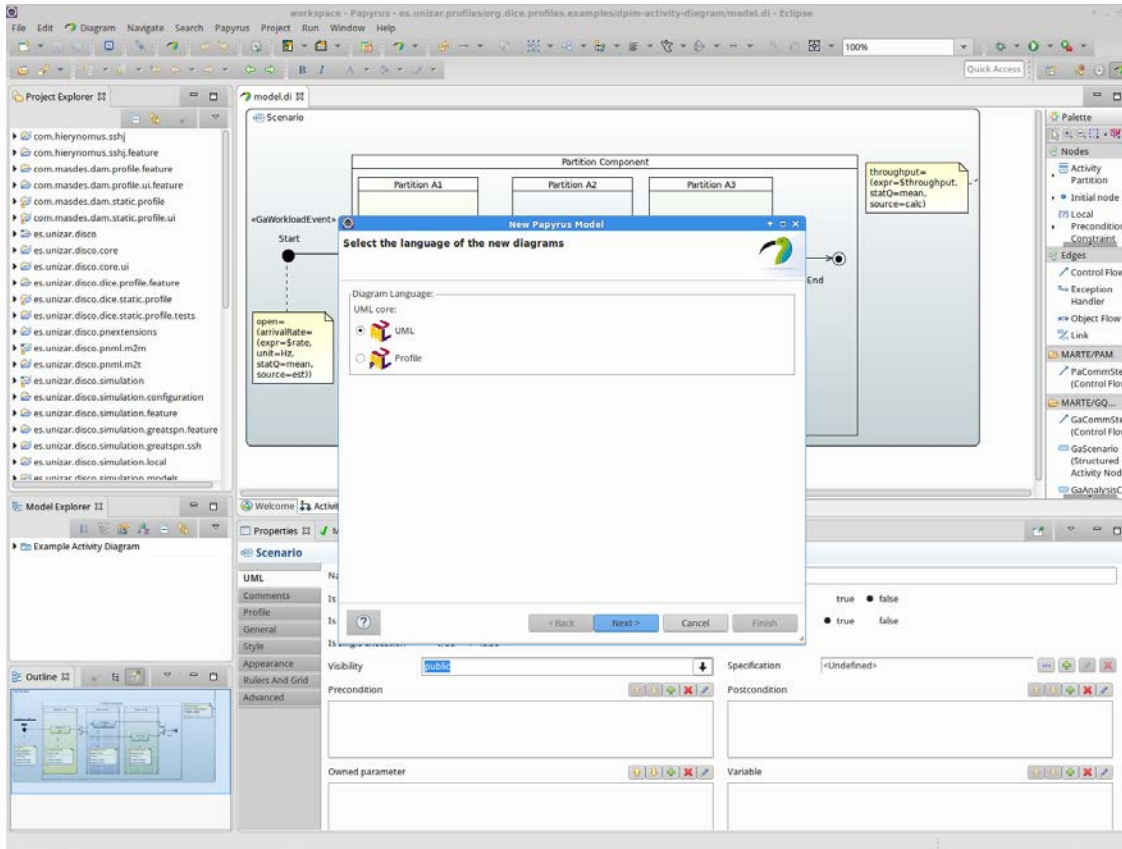
Motive: Predict the behavior of the system prior to the deployment in the cloud.



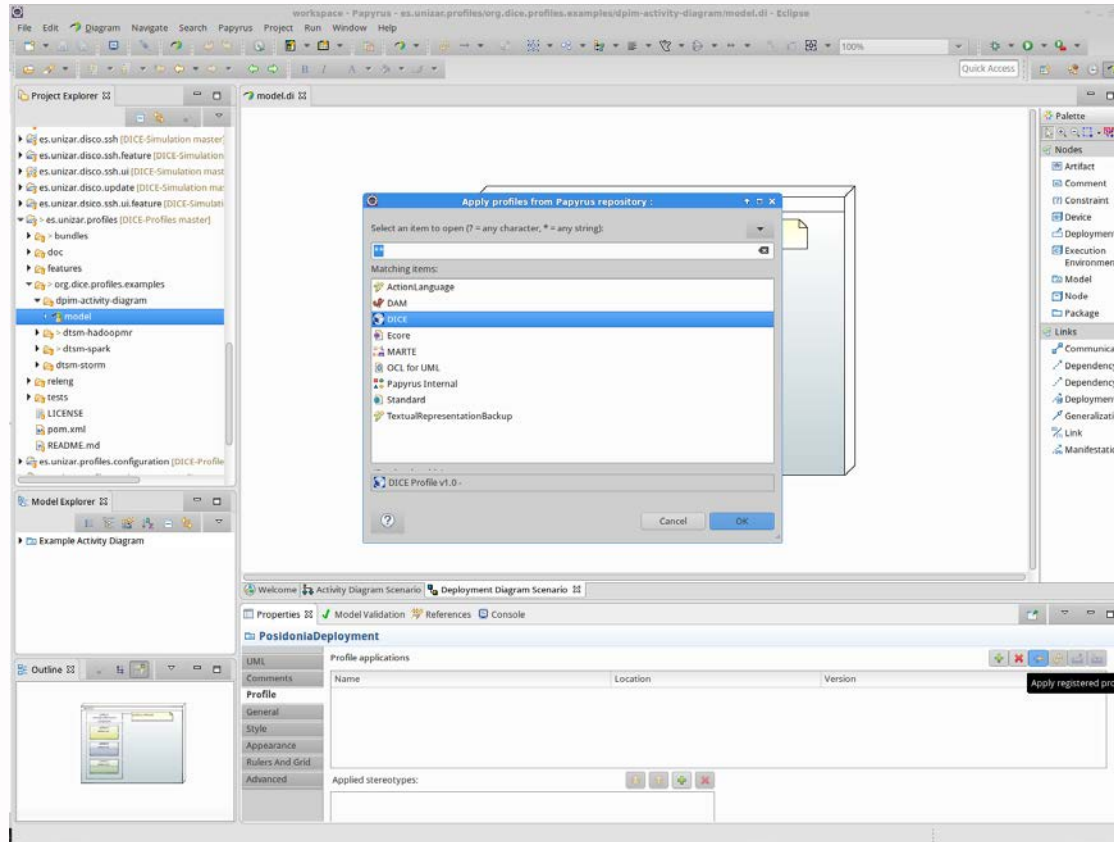
Simulation workflow (1)



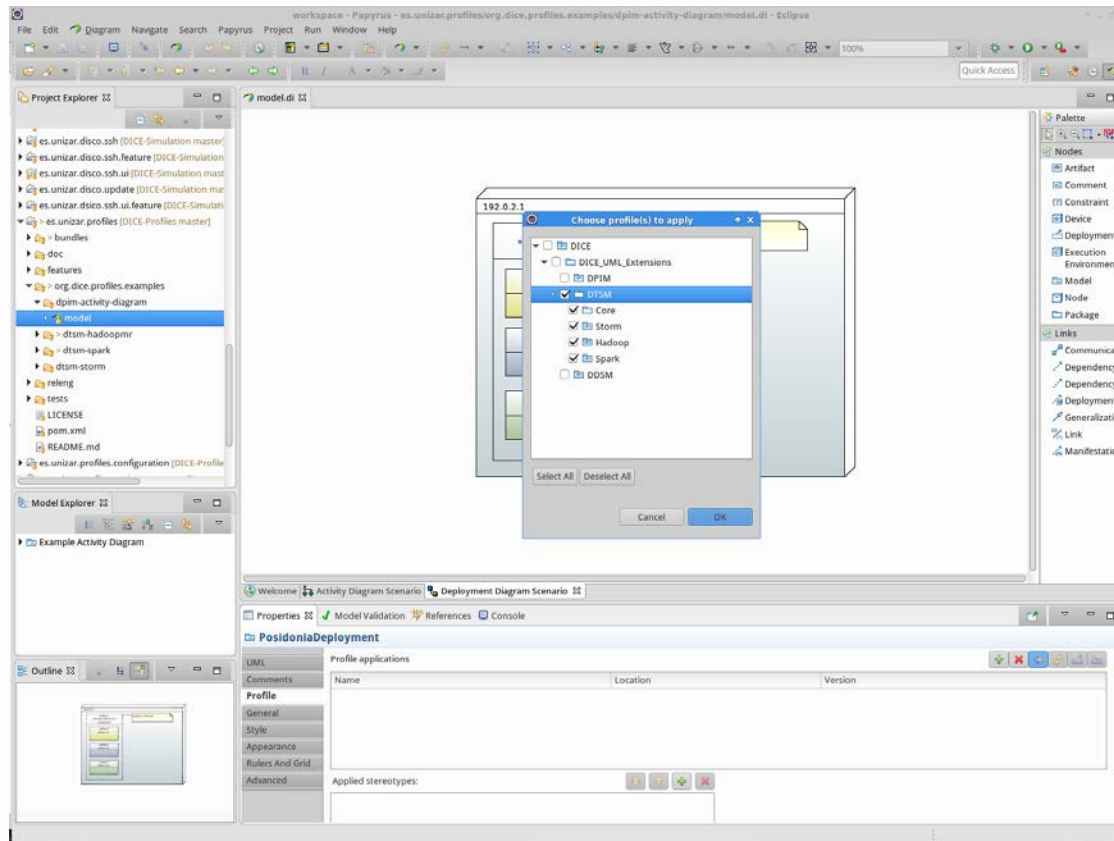
Simulation workflow (2)



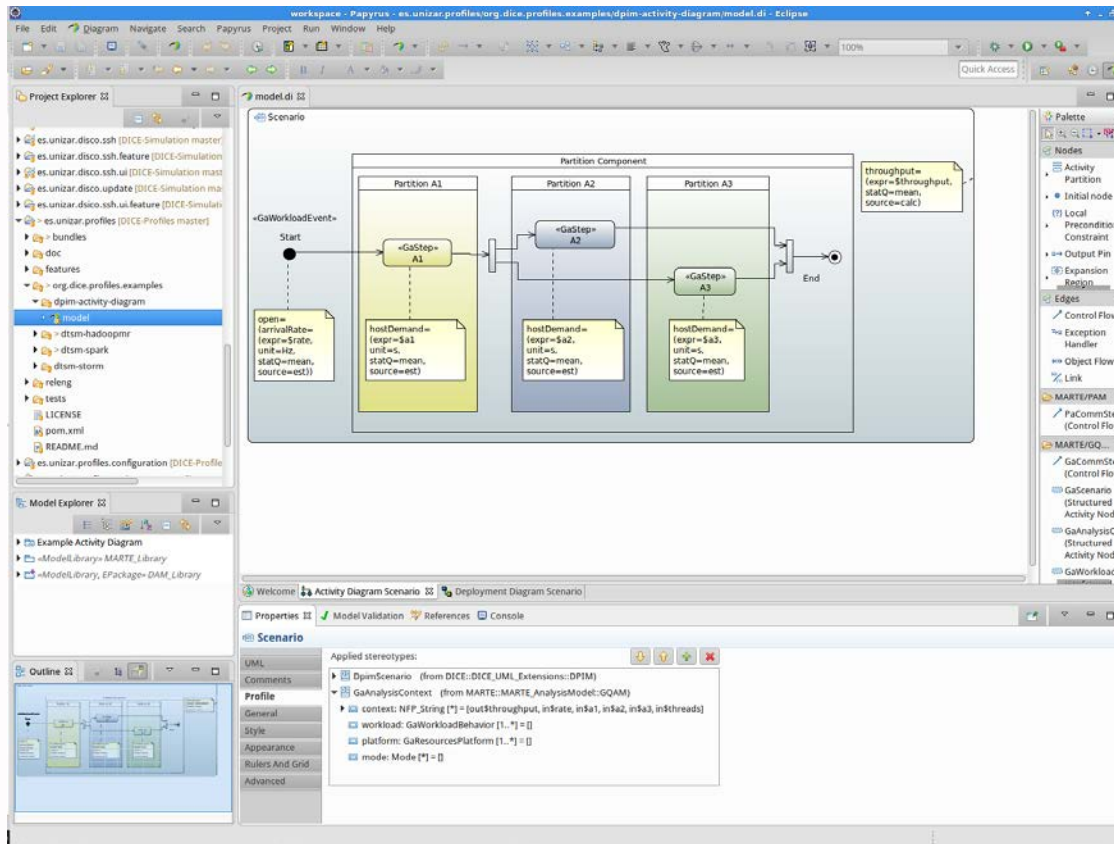
Simulation workflow (3)



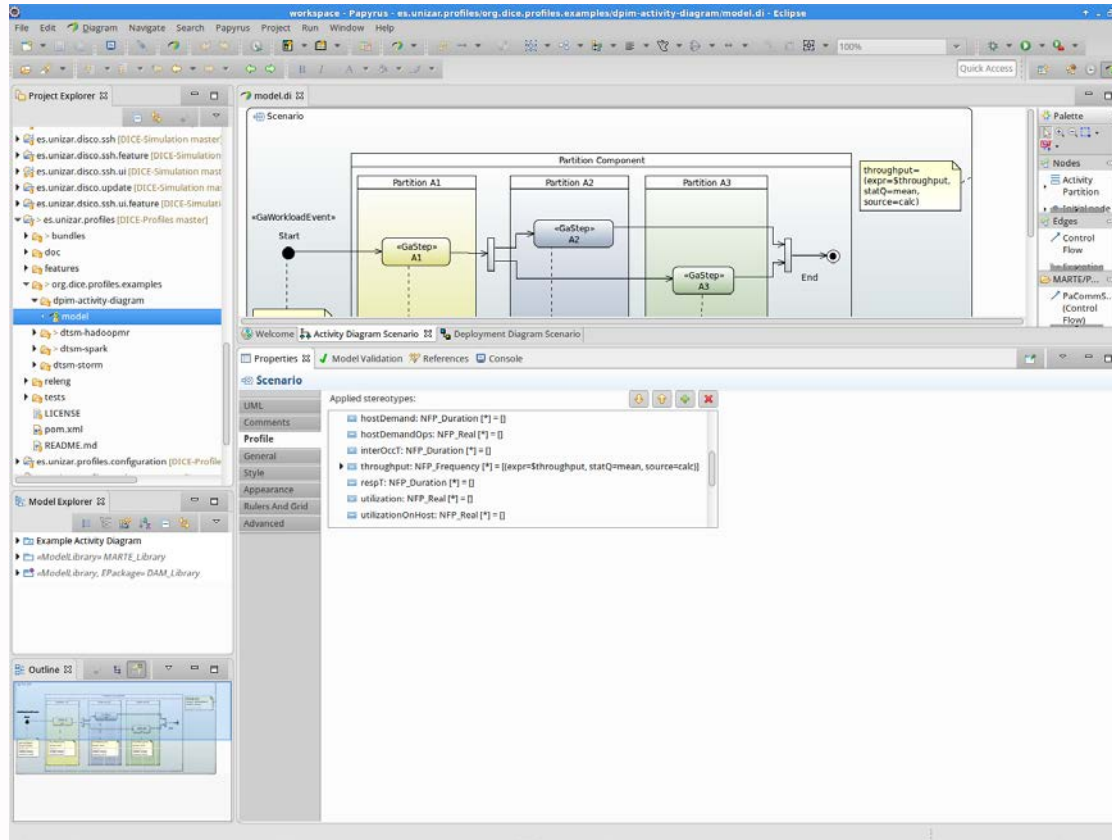
Simulation workflow (4)



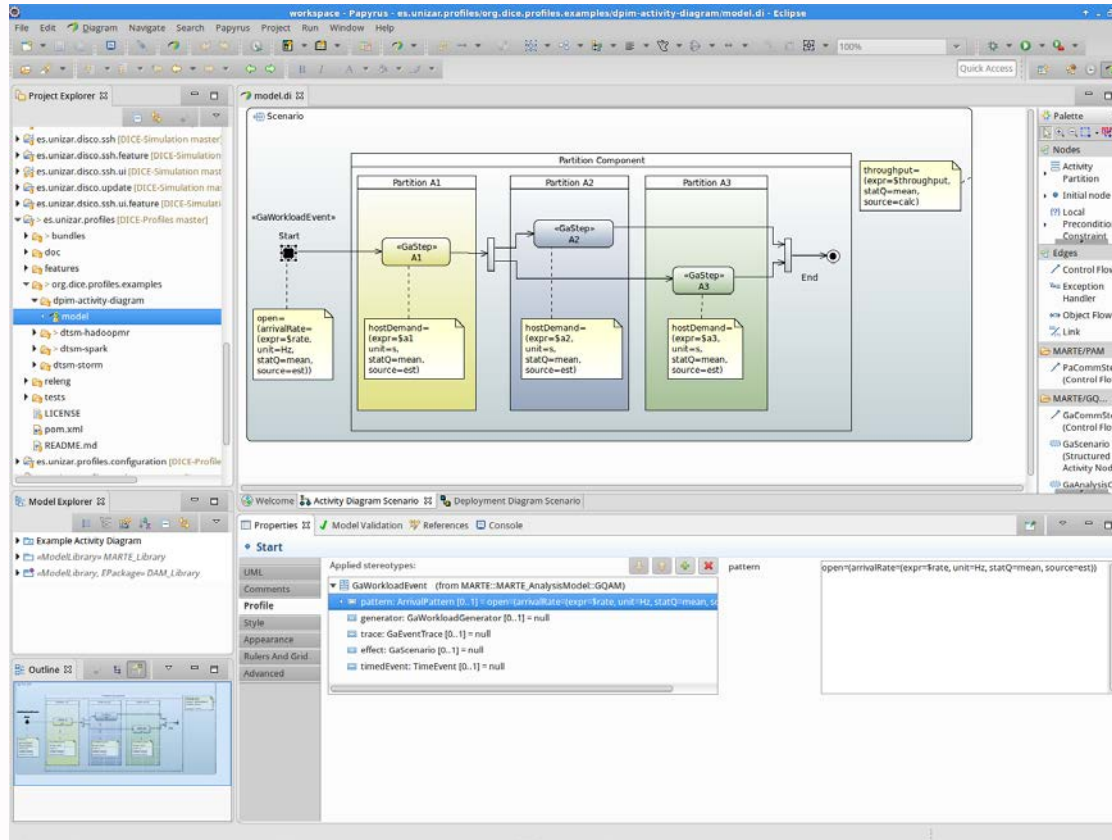
Simulation workflow (5)



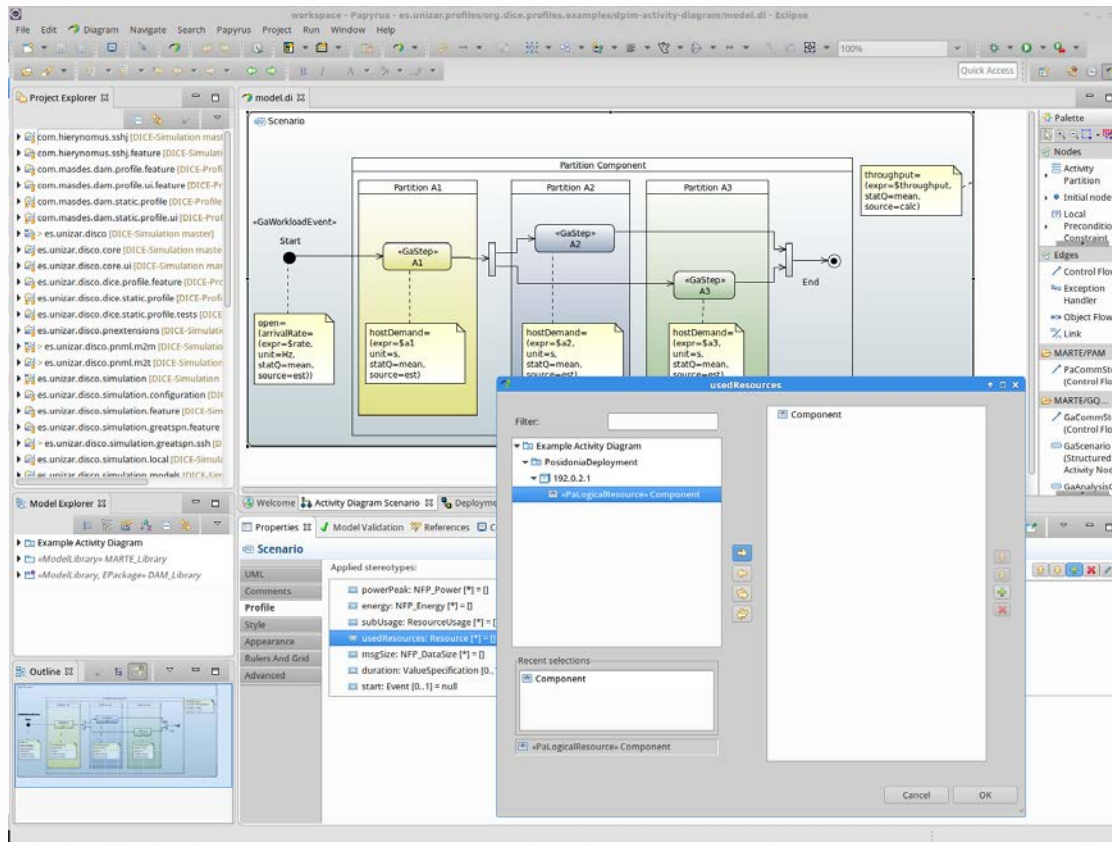
Simulation workflow (6)



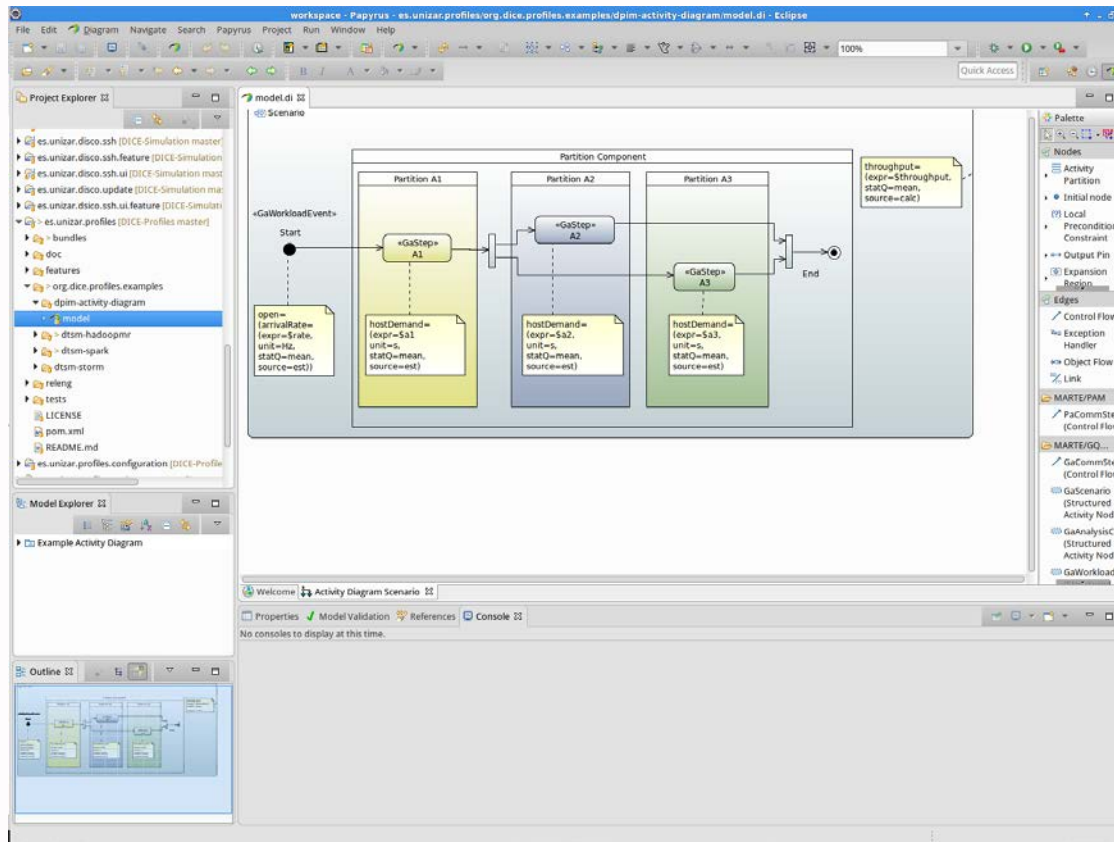
Simulation workflow (7)



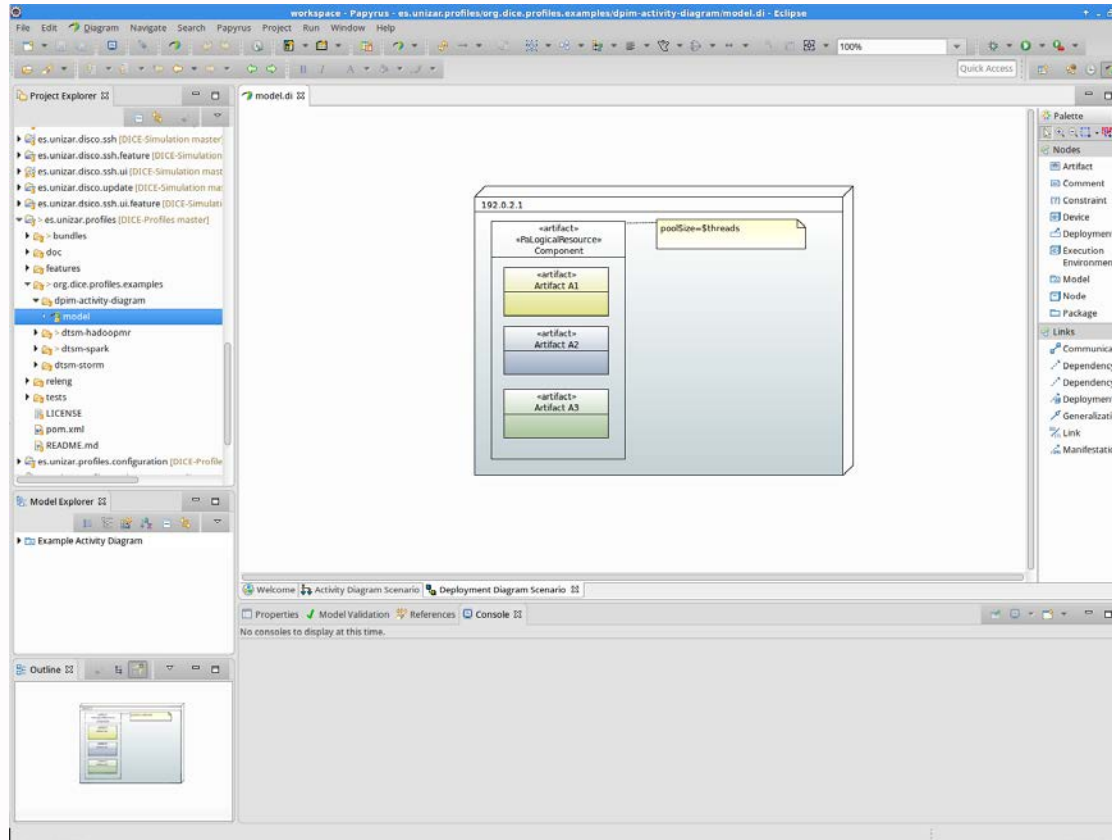
Simulation workflow (8)



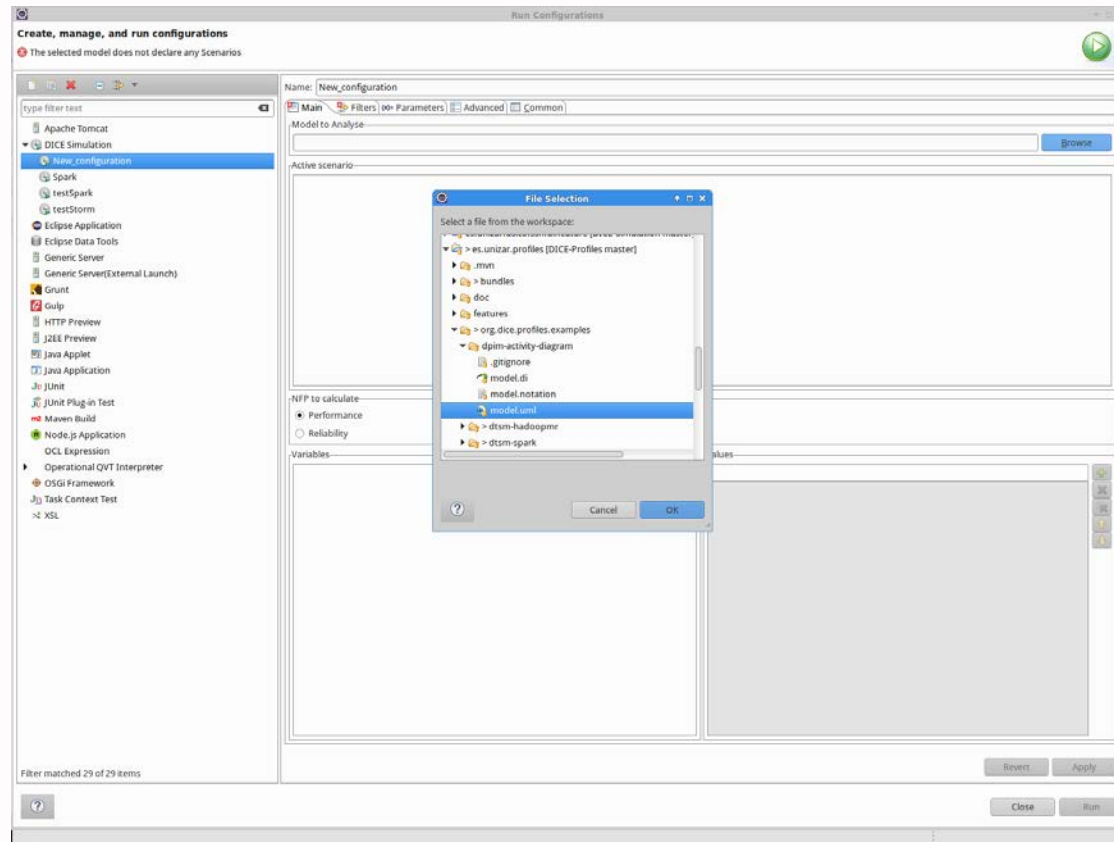
Simulation workflow (9)



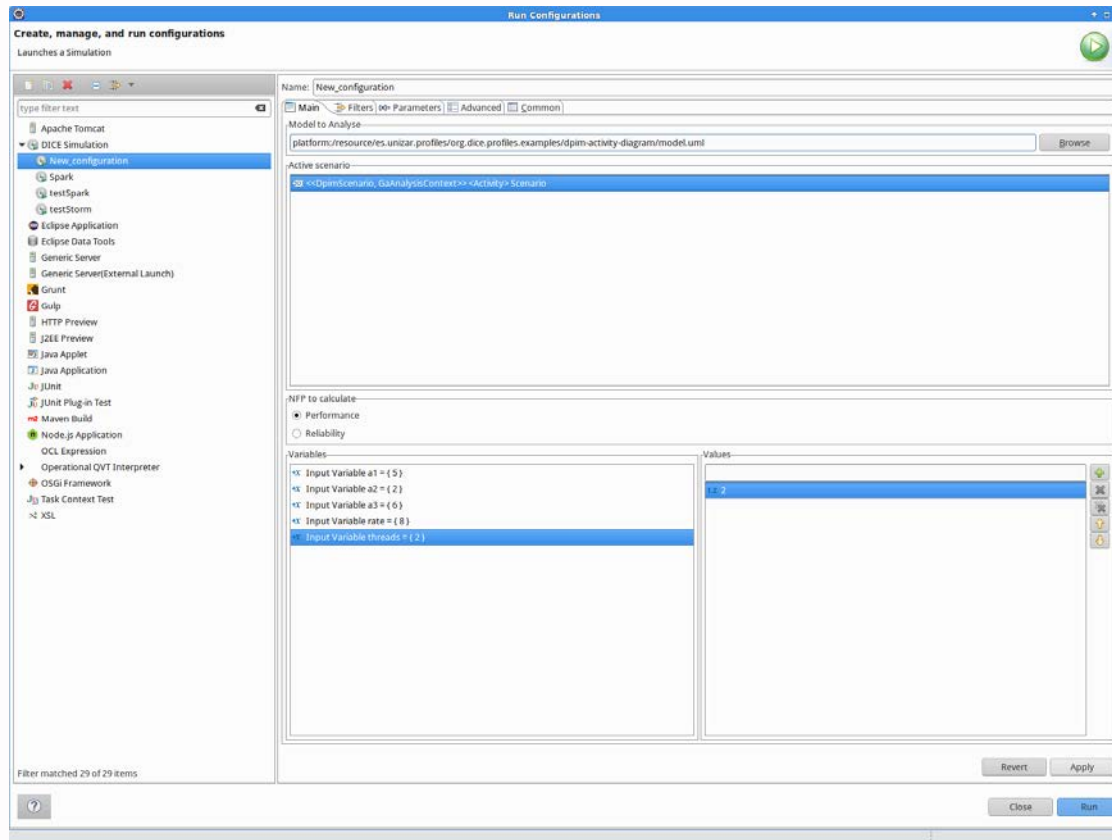
Simulation workflow (10)



Simulation workflow (11)



Simulation workflow (12)



Simulation workflow (13)

The screenshot displays the Eclipse IDE interface for a simulation workflow. The top section shows the Console window with the following log output:

```

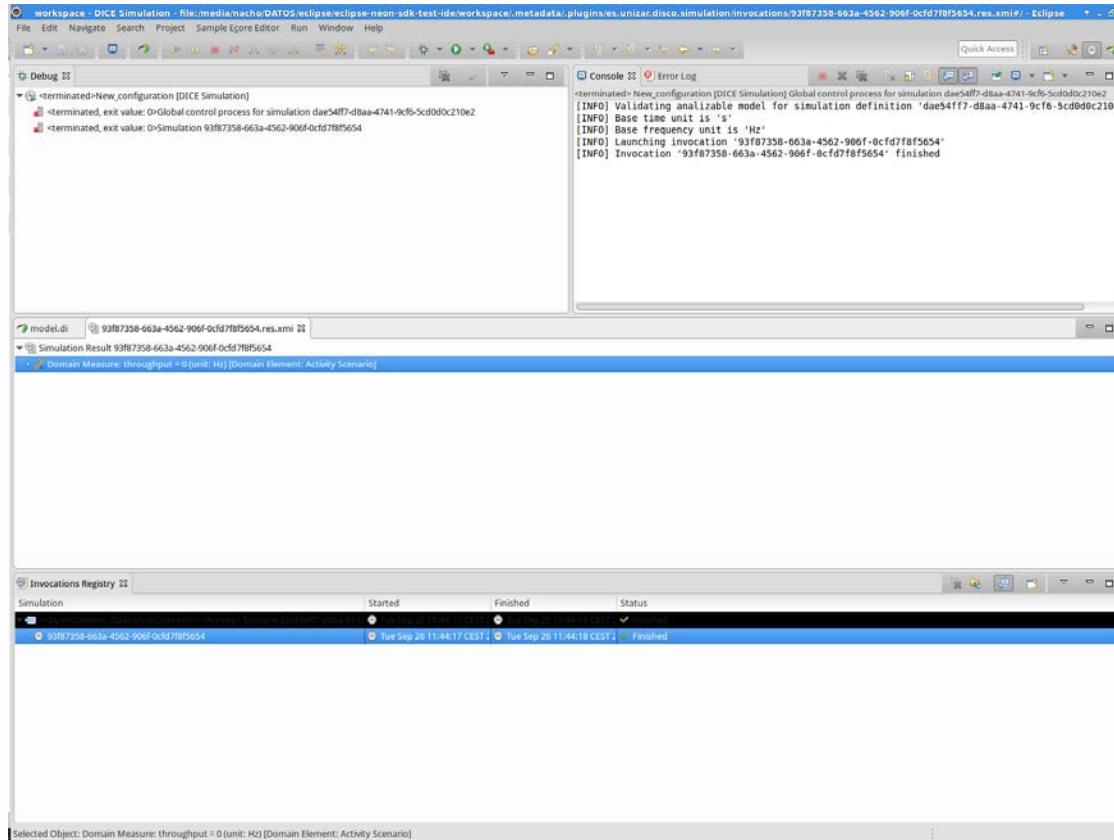
-terminated> New_configuration [DICE Simulation] Global control process for simulation dae54ff7-d8aa-4741-9cf6-5cd9d0c210e2
[INFO] Validating analyzable model for simulation definition 'dae54ff7-d8aa-4741-9cf6-5cd9d0c210e2'
[INFO] Base time unit is 's'
[INFO] Base frequency unit is 'Hz'
[INFO] Launching invocation '93f87358-663a-4562-906f-bcf7f8f5654'
[INFO] Invocation '93f87358-663a-4562-906f-bcf7f8f5654' Finished
    
```

The middle section shows the Activity Diagram Scenario, which includes a Partition Component with three parallel partitions: Partition A1, Partition A2, and Partition A3. Each partition contains a GasStep (A1, A2, and A3 respectively). The workflow starts with a Start node, followed by a parallel fork leading to the three GasSteps, which then merge at an End node. A note on the right indicates a throughput calculation: `throughput = (expr=$throughput, statQ=mean, source=calc)`.

The bottom section shows the Invocations Registry with the following table:

Simulation	Started	Finished	Status
93f87358-663a-4562-906f-bcf7f8f5654	Tue Sep 26 11:44:17 CEST	Tue Sep 26 11:44:18 CEST	Finished

Simulation workflow (14)



Simulation Benefit

- Configure the Storm design to specific execution context
- Detect performance bottlenecks
- Less than 20% with regard to the prediction error on the utilization metric

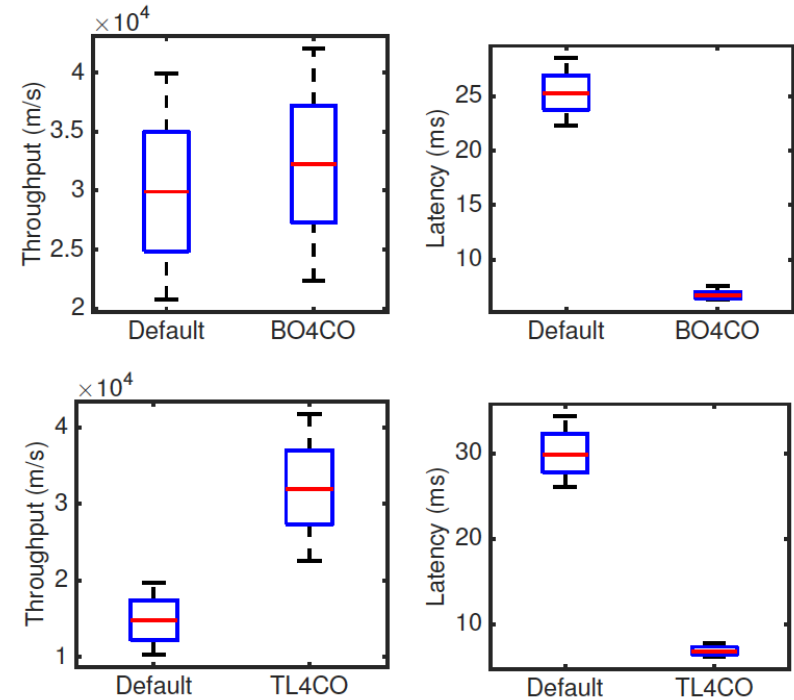
```

model_sigma_real.di  8a7106ce-2039-46d6-bca3-f0ac7ee01776.res.xml
Simulation Result 8a7106ce-2039-46d6-bca3-f0ac7ee01776
  Domain Measure: utilization = 28.4685330135 (unit: percentage) [Domain Element: Opaque Action EntityExtraction]
  Domain Measure: utilization = 15.7485661315 (unit: percentage) [Domain Element: Opaque Action MinHashExtraction]
  Domain Measure: utilization = 95.38756225944 (unit: percentage) [Domain Element: Opaque Action TwitterSample]
  Domain Measure: throughput = 47.7215246951 (unit: Hz) [Domain Element: Opaque Action TwitterSample]
  Domain Measure: utilization = 24.1331819745 (unit: percentage) [Domain Element: Opaque Action TermsExtraction]
  Domain Measure: utilization = 98.87464554563 (unit: percentage) [Domain Element: Opaque Action SolrUpdater]
  Domain Measure: throughput = 19.4930223776 (unit: Hz) [Domain Element: Opaque Action SolrUpdater]
  Domain Measure: utilization = 36.2803247215 (unit: percentage) [Domain Element: Opaque Action TermsRollingCounter]
  Domain Measure: utilization = 0.102745618 (unit: percentage) [Domain Element: Opaque Action IntermediateRanker]
  Domain Measure: utilization = 0.0259525088 (unit: percentage) [Domain Element: Opaque Action FinalRanker]
  Domain Measure: utilization = 0.9203768593 (unit: percentage) [Domain Element: Opaque Action TopicsMongoDBWriter]
  Domain Measure: utilization = 0.2313554488 (unit: percentage) [Domain Element: Opaque Action TopicsLabeler]
  Domain Measure: throughput = 0.0109760002489 (unit: Hz) [Domain Element: Opaque Action TopicsLabeler]
  Domain Measure: utilization = 13.02005980525 (unit: percentage) [Domain Element: Opaque Action MinHashRollingCounter]
  Domain Measure: utilization = 0.3261115673 (unit: percentage) [Domain Element: Opaque Action MinHashIntermediateRanker]
  Domain Measure: utilization = 0.0353575097 (unit: percentage) [Domain Element: Opaque Action MinHashFinalRanker]
  Domain Measure: utilization = 0.6906373801 (unit: percentage) [Domain Element: Opaque Action ClustersMongoDBWriter]
  Domain Measure: utilization = 0.2180832876 (unit: percentage) [Domain Element: Opaque Action ClustersLabeler]
  Domain Measure: throughput = 0.00991380667644 (unit: Hz) [Domain Element: Opaque Action ClustersLabeler]
  Domain Measure: utilization = 1.123342844 (unit: percentage) [Domain Element: Opaque Action ItemDeserializer]
  Domain Measure: utilization = 78.005780761625 (unit: percentage) [Domain Element: Device Core_1]
  
```

Configuration Optimization (CO)

drpc.invocations.threads	64
drpc.max_buffer_size	1048576
drpc.port	3772
drpc.queue.size	128
drpc.request.timeout.secs	600
drpc.worker.threads	64
java.library.path	"/usr/local/lib:/opt/local/lib:/usr/lib"
logs.users	null
logviewer.appender.name	"A1"
logviewer.childopts	"-Xmx128m"
logviewer.cleanup.age.mins	10000
logviewer.max.per.worker.logs.size.mb	2048
logviewer.max.sum.worker.logs.size.mb	4096
logviewer.port	8000
nimbus.blobstore.class	"org.apache.storm.blobstore.LocalFsBlobStore"
nimbus.blobstore.expiration.secs	600
nimbus.childopts	"-Xmx1024m"
nimbus.cleanup.inbox.freq.secs	600
nimbus.code.sync.freq.secs	120
nimbus.credential.renewers.freq.secs	600
nimbus.file.copy.expiration.secs	600
nimbus.impersonation.authorizer	"org.apache.storm.security.auth.authorizer.ImpersonationAuthorizer"
nimbus.inbox.jar.expiration.secs	3600
nimbus.monitor.freq.secs	10
nimbus.queue.size	100000
nimbus.seeds	["10.68.80.67"]
nimbus.supervisor.timeout.secs	60
nimbus.task.launch.secs	120
nimbus.task.timeout.secs	30
nimbus.thrift.max_buffer_size	1048576

Motive: Find optimal configuration settings wrt computational budget restrictions



Plugin Configuration, Parameter Selection

Properties Console SVN Repositories Git Repositories Invocations Registry DICE Configuration View

Plugin Config Parameter Selection Service Config Experiment Config App Config Experiments

Save current configurations

Load saved configurations

Username:

Password:

Jenkins Url:

Job Name:

Build Token:

Properties Console SVN Repositories Git Repositories Invocations Registry DICE Configuration View

Plugin Config Parameter Selection Service Config Experiment Config App Config Experiments

storm

Parameter	Description
topology.error.throttle.interval.secs	
topology.trident.batch.emit.interval.millis	
topology.disruptor.wait.timeout.millis	
topology.disruptor.batch.size	
topology.disruptor.batch.timeout.millis	
topology.disable.loadaware.messaging	
topology.state.checkpoint.interval.ms	
topology.max.spout.pending	
topology.ack.executors	
topology.tick.tuple.freq.secs	

Add Parameters

Parameter	Type	Min	Max	Step	Options
topology.executor.receive.buffer.size	Integer	1024	2048	1	
topology.min.replication.count	Integer	1	10	1	
topology.worker.shared.thread.pool.size	Integer	1	20	1	
topology.max.task.parallelism	Integer	1	50	1	

Service & Experiment Configuration

The screenshot displays the DICE Configuration View interface, which is divided into two main sections: Service Configuration and Experiment Configuration.

Service Configuration Panel:

- Plugin Config:** Parameter Selection, Service Config, Experiment Config, App Config, Experiments
- Service 1:**
 - servicename: Kafka Broker
 - URL: http://10.151.64.59:9092
 - username: giopnd
 - password: *****
 - Remove button
- Service 2:**
 - servicename: Monitoring
 - URL: http://10.151.64.45:5001
- Options:**
 - servicename URL ip container
 - username password tools storm_client
- Add Service button

Experiment Configuration Panel:

- Plugin Config:** Parameter Selection, Service Config, Experiment Config, App Config, Experiments
- Configuration Parameters:**
 - Noise: 1e-5
 - Number of Iterations: 100
 - Initial Design: 2
 - Save Folder: ./integrated/reports
 - Config Folder: ./integrated/config
 - Summary Folder: ./integrated/summary
 - Blueprint: storm-openstack.yaml
 - Config: topology.yaml
 - Topic: json-topic
 - Sleep Time: 10000
 - Metric Poll: 1000
 - Exp Time: 300000
 - Replication: 4

Application Configuration, Experiments

The screenshot shows the 'DICE Configuration View' window with the following configuration fields:

- CLI file:
- Jar File:
- Jar Path:
- Class:
- Name:
- Args:
- Type:

The screenshot shows the 'DICE Configuration View' window with the following experiment control buttons:

- Run BO4CO
- Check Status
- Retrieve Results

CO Benefit

- **Throughput improvement:**
 - more than twice compared to the default configuration
 - achieved after only 100 iterations
 - took $100 * 10 \text{ min} = 16\text{h}$ to run
- **No expert needed!!**

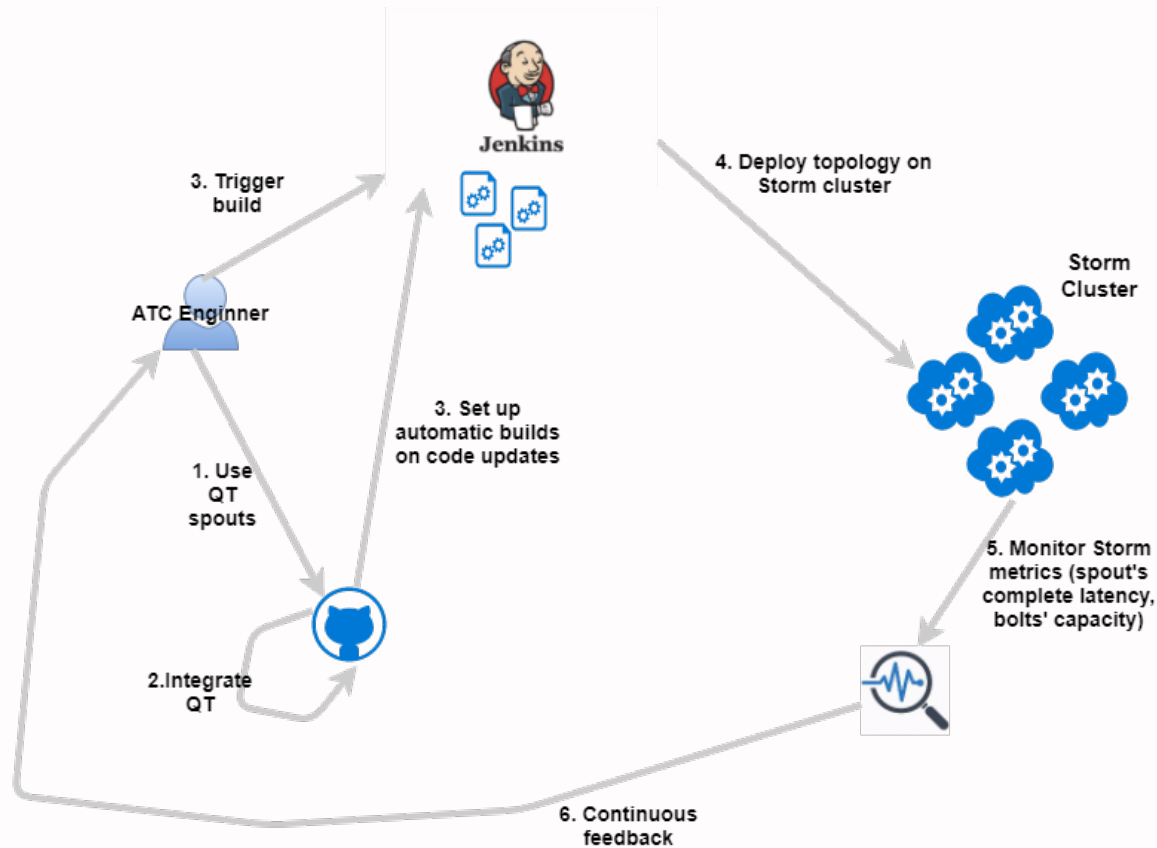
Quality Testing (QT)

- **Motive:** Stress test the capacity of the ‘trending topic detection’ topology
- Run multiple iterations/experiments
 - Inject constantly increasing stream load
 - With respect to ‘complete latency’ metric threshold
- Evaluate the maximum input rate
 - Twitter Streaming API has limitations
 - Twitter Paid API

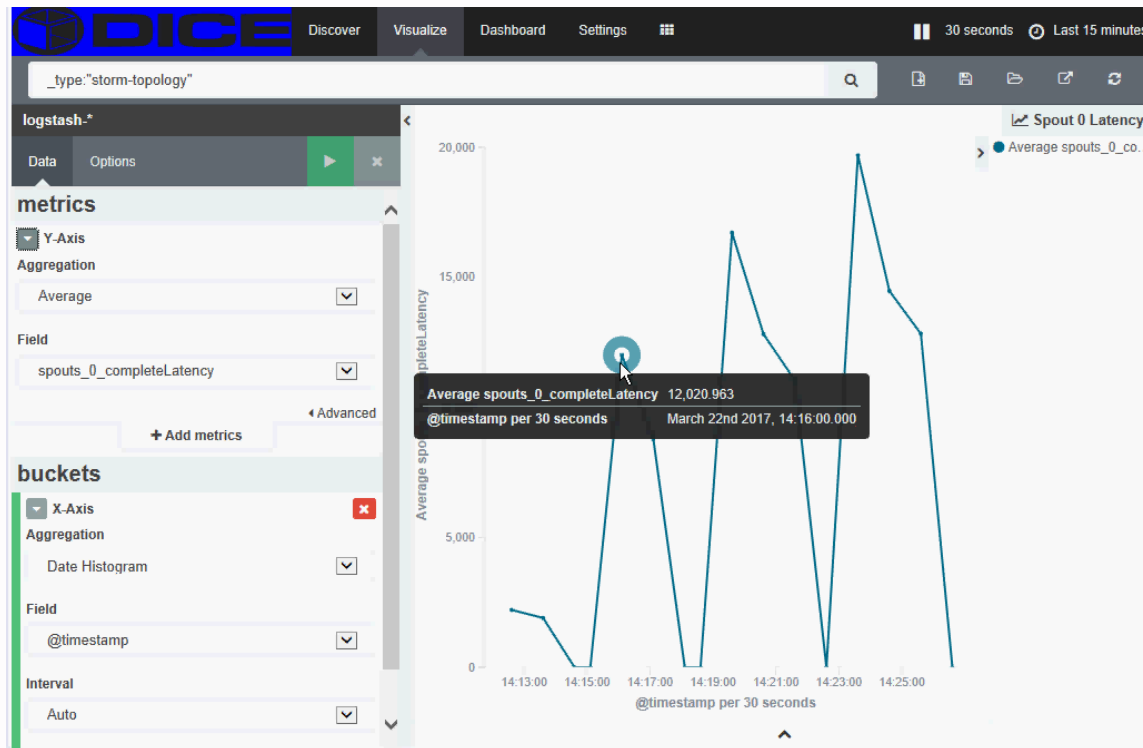
```
TopologyBuilder builder = new TopologyBuilder();

/* Create QT-LIB's Spout factory */
QTLoadInjector qt = new QTLoadInjector();
/* Obtain a spout to inject at prescribed rates specified
RateSpout qtSpout = qt.getRateSpout();
qtSpout.setArrivalMode(RateSpout.ArrivalMode.ParseCount);
qtSpout.setRateScaler(scaler);
qtSpout.setArrivalFile("counts.txt");
qtSpout.setDataMode(RateSpout.DataMode.ParseJSON);
qtSpout.setBinaryBDoc(true);
qtSpout.setDataFile("test.json");
```

Quality Testing (QT) - Scenario



QT Benefit



Berlin, 11-13 October 2017



CONCLUSION

Final thoughts and
remarks

DICE – Automated Testing Encapsulated

- Methodology for using DevOps, testing tools
- Uses a mix of known approaches
 - Formal model analyses (Simulation Tools)
 - Machine Learning approaches (Anomaly Detection, Configuration Optimization)
 - Efficient techniques of optimization
- Works offline on a design and online on the runtime

Will it be useful for you?

DICE Tool	We use models in our design all the time	UML? No, thank you
Simulation	✓	
DICER Deployment	✓	✓
Configuration Optimization	✓	✓
Quality Testing	✓	✓
Fault Injection	✓	✓
Monitoring	✓	✓

Follow us!

- Web page: <http://www.dice-h2020.eu/>
- Code repository: <https://github.com/dice-project>
- Twitter: [@diceh2020](https://twitter.com/diceh2020) Facebook: [DiceH2020](https://www.facebook.com/DiceH2020)
- Matej Artač: matej.artac@xlab.si [@matej_artac](https://www.instagram.com/matej_artac)
- Ismael Torres: itorres@prodevelop.es
- Vasilis Papanikolaou: v.papanikolaou@atc.gr
- George Giotis: g.giotis@atc.gr
- Damian A. Tamburri: damianandrew.tamburri@polimi.it



Funded by the Horizon 2020
Framework Programme of
the European Union



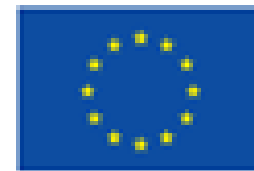
Acknowledgement

DICE

Horizon 2020 Research & Innovation Action

Grant Agreement no. 644869

<http://www.dice-h2020.eu>



Funded by the Horizon 2020
Framework Programme of the European Union