5th UCAAT User Conference on Advanced Automated Testing

ETSI

TESTING BIG DATA APPLICATIONS AT DESIGN TIME AND RUNTIME WITH DICE

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

DICE

MOTIVATION

Why DICE?

DICE

Outline

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

5th UCAAT User Conference on Advanced Automated Testing

ETSI

ABOUT THE DICE PROJECT

Developing Data-Intensive Cloud Applications with Iterative Quality Enhancements

DICE

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

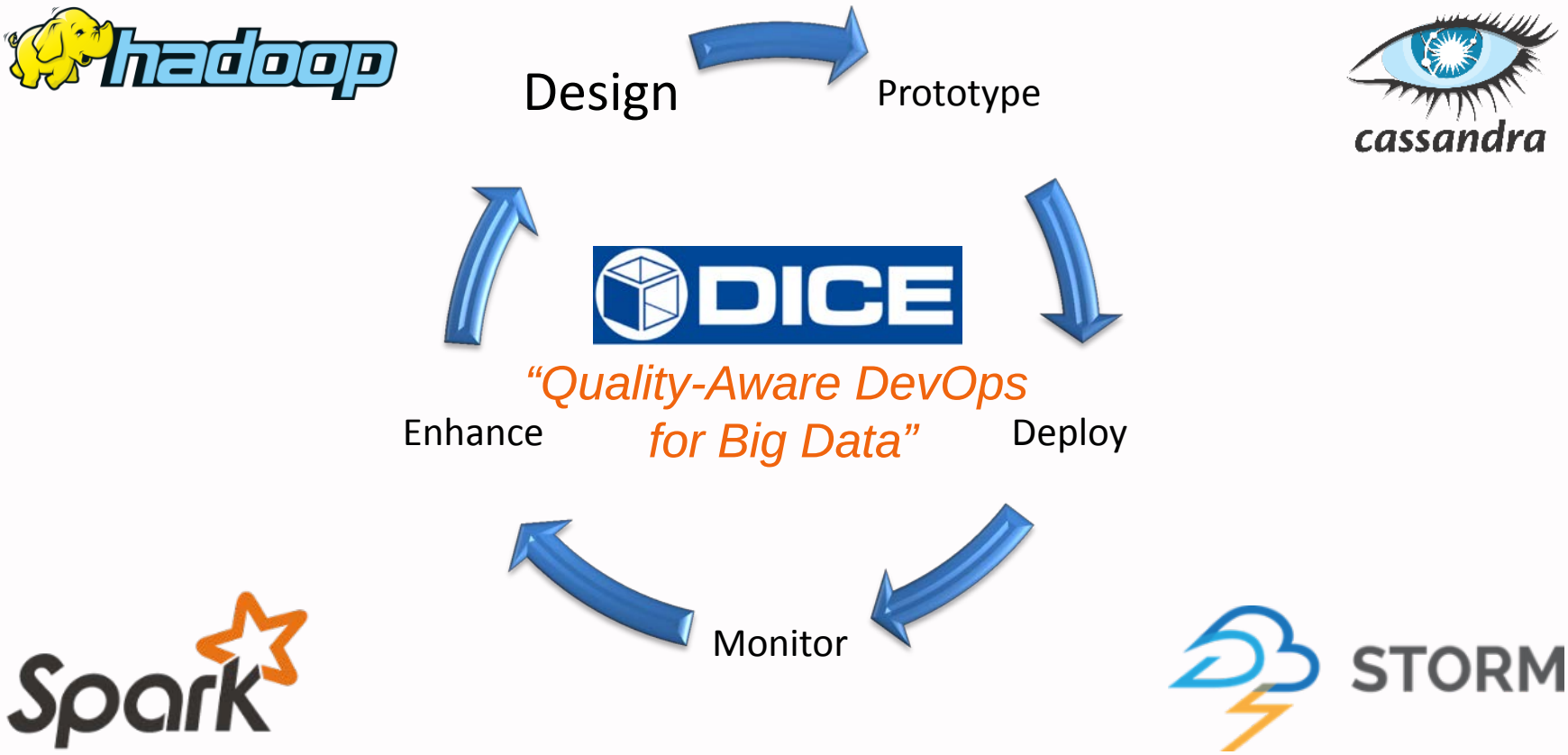
hadoop MapReduce

Spark

cassandra

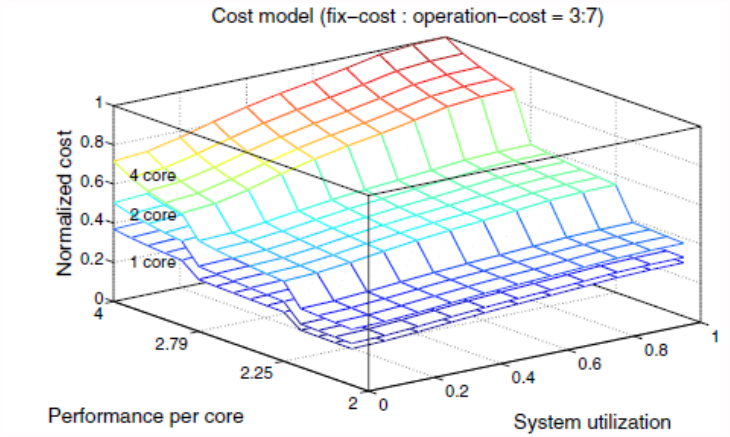
(+ others…)

DICE: Quality-Aware DevOps for Big Data



What do we mean by Quality?

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

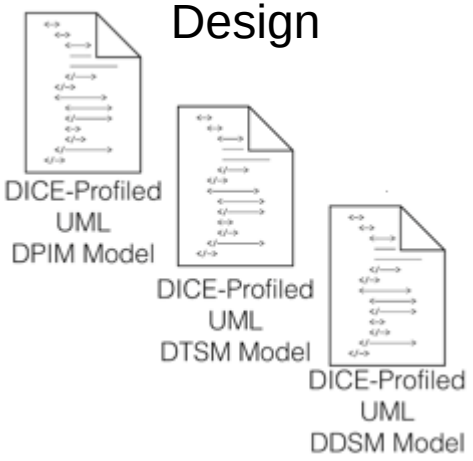


DICE Workflow - Dev

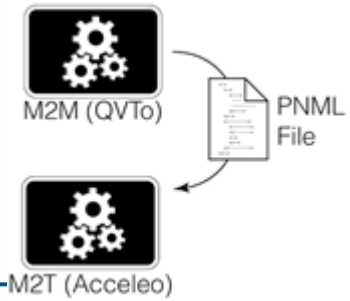
DICE Eclipse IDE

Enhance

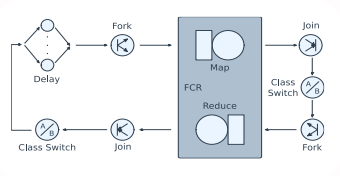
Design



Transform
to Formal Models

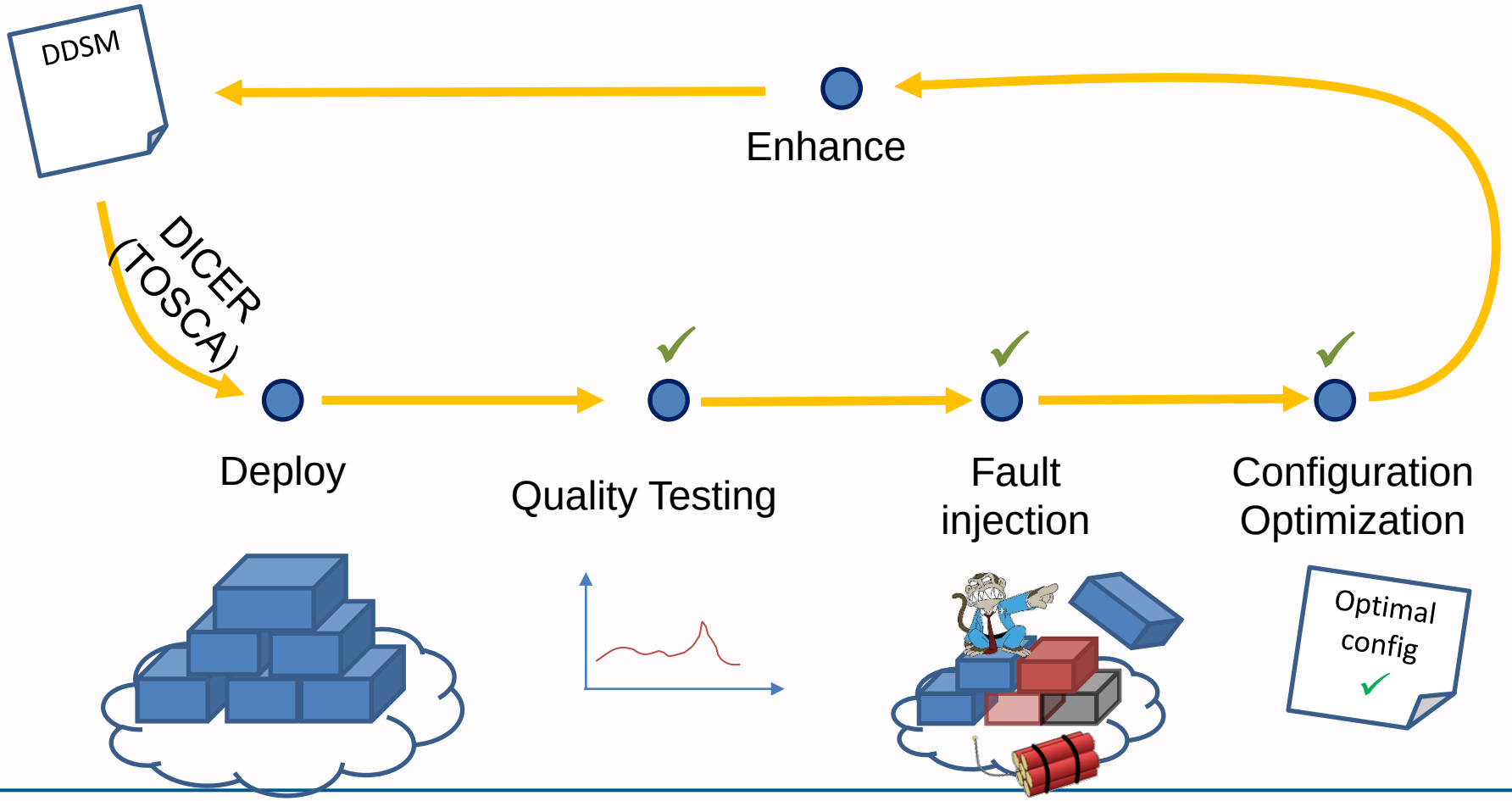


Simulate & Verify

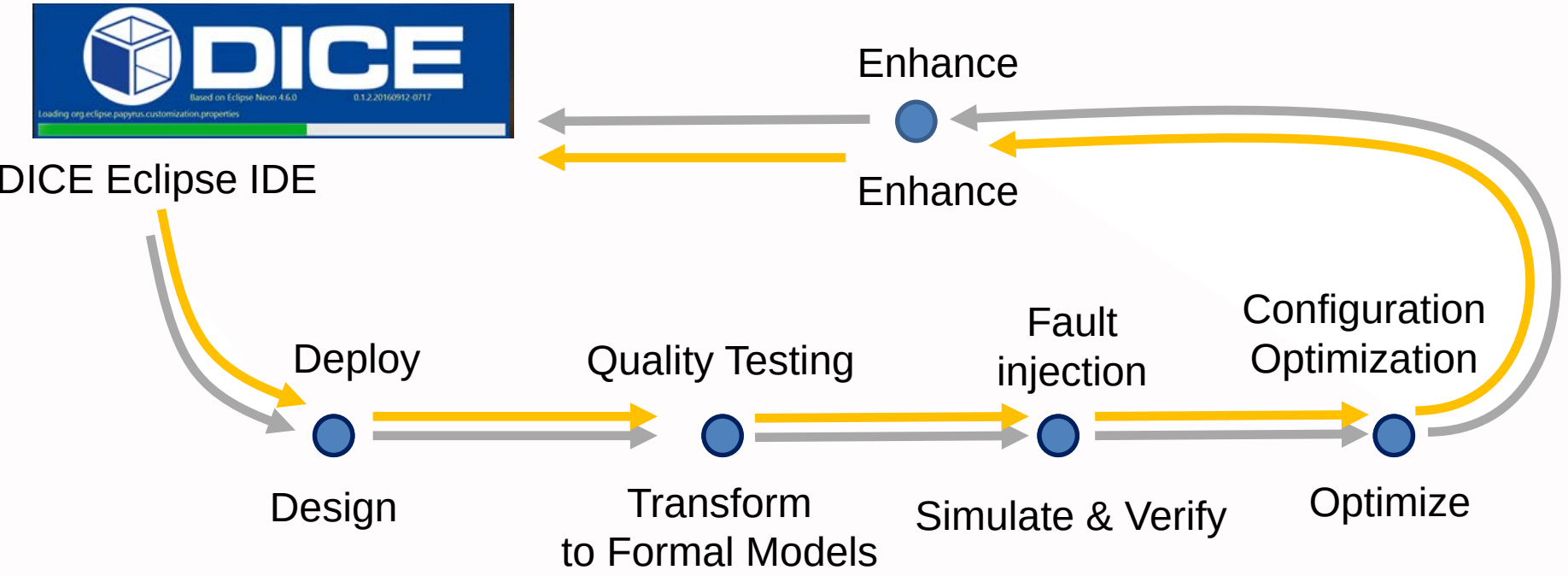


Optimize

DICE Workflow - Ops



DICE Workflow: Unified Toolchain



5th UCAAT User Conference on Advanced Automated Testing

ETSI

DICE DEMONSTRATORS

Tools for real SMEs with real use cases

DICE

Demonstrators

Posidonia Operations

Maritime Sector

News Asset

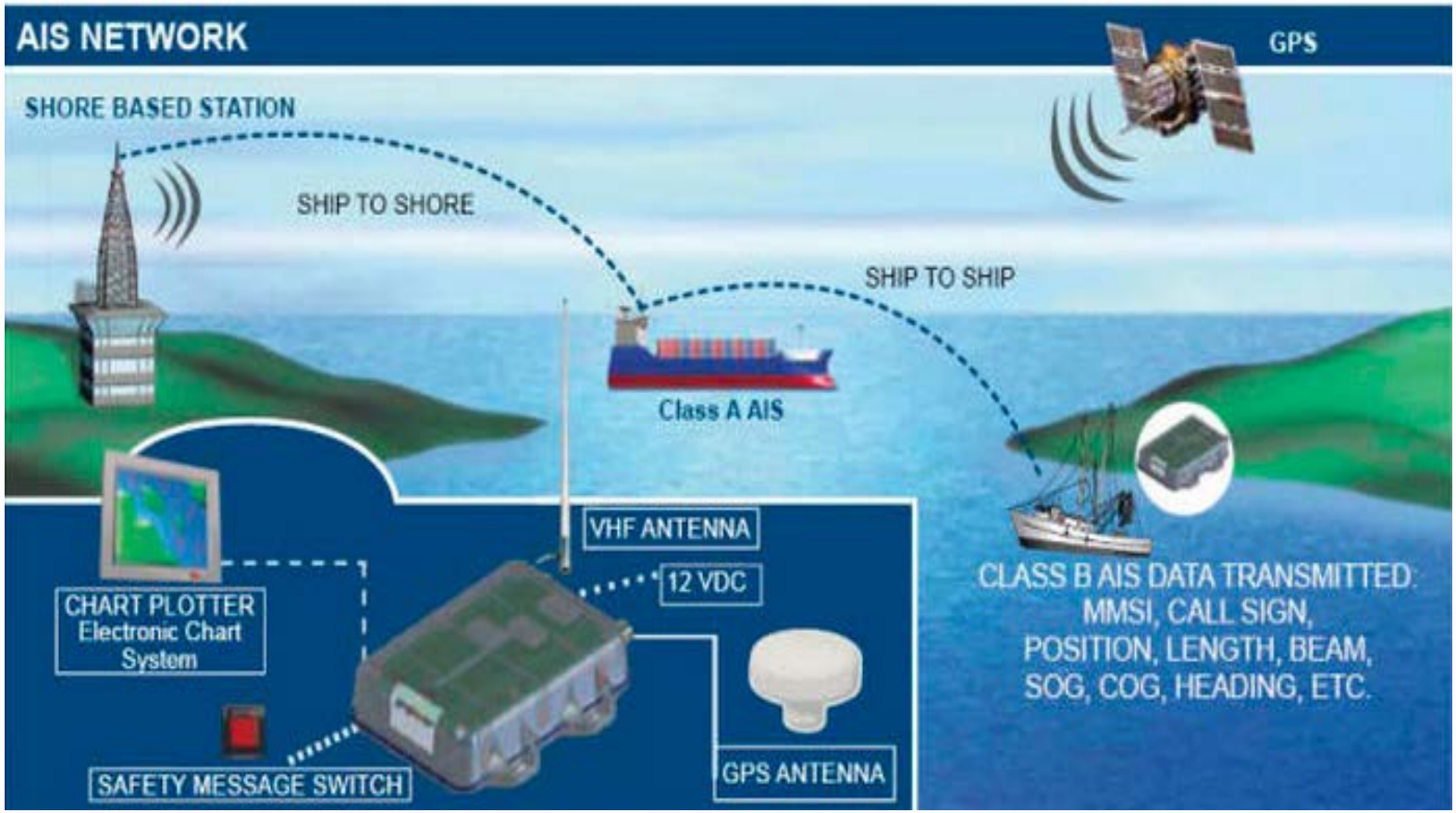
News&Media Market

Tax Fraud Detection Application

e-Goverment Market

Tutorial 1: Posidonia Operations

prodevelop Integración de tecnologías



Tutorial 2: News Asset Platform

5th UCAAT User Conference on Advanced Automated Testing

ETSI

DICE- POSIDONIA DEMONSTRATOR

Ismael Torres

Posidonia
Port Solutions Suite

prodevelop
Integrating technologies

DICE

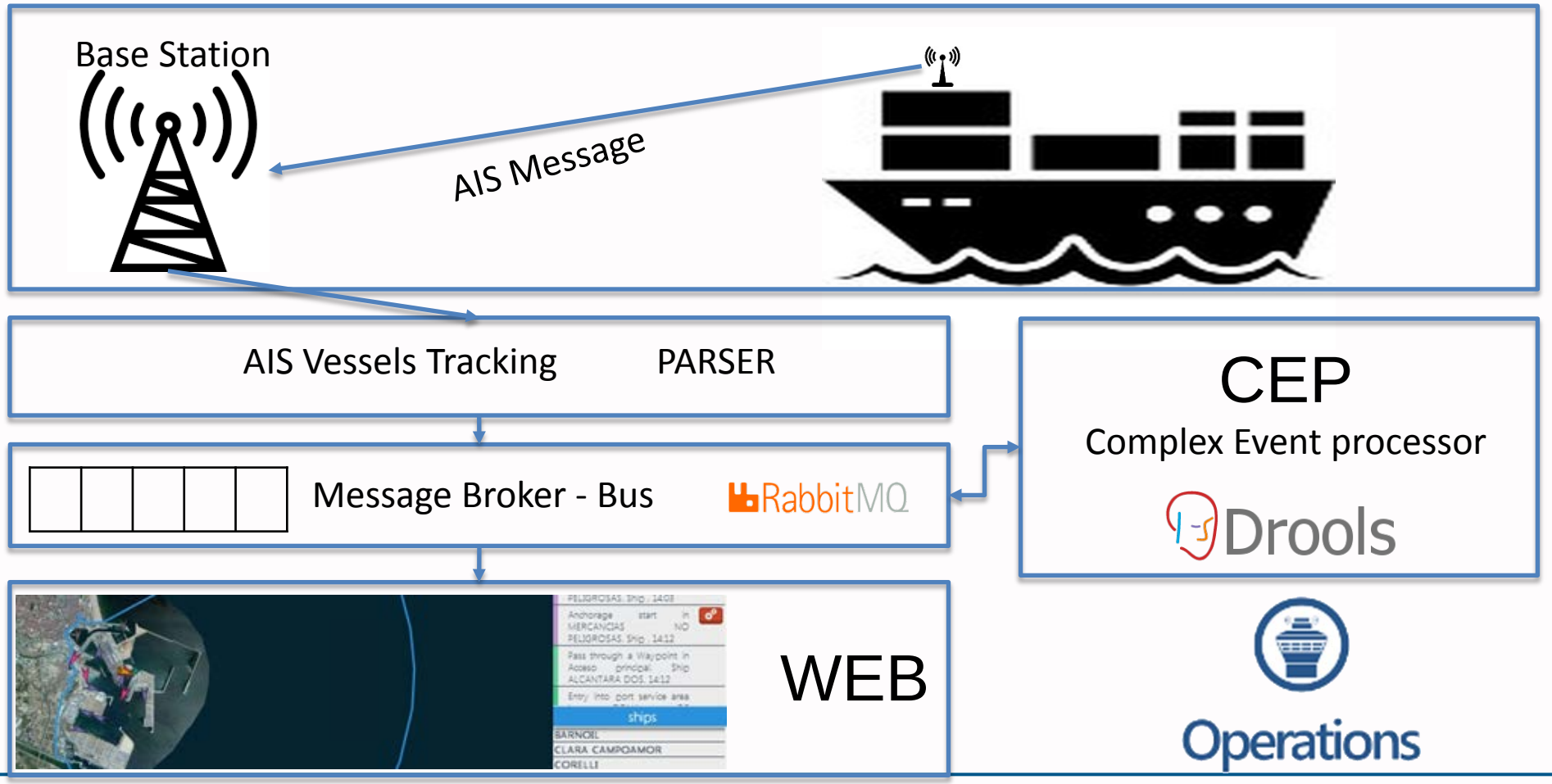
About Prodevelop

- **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)

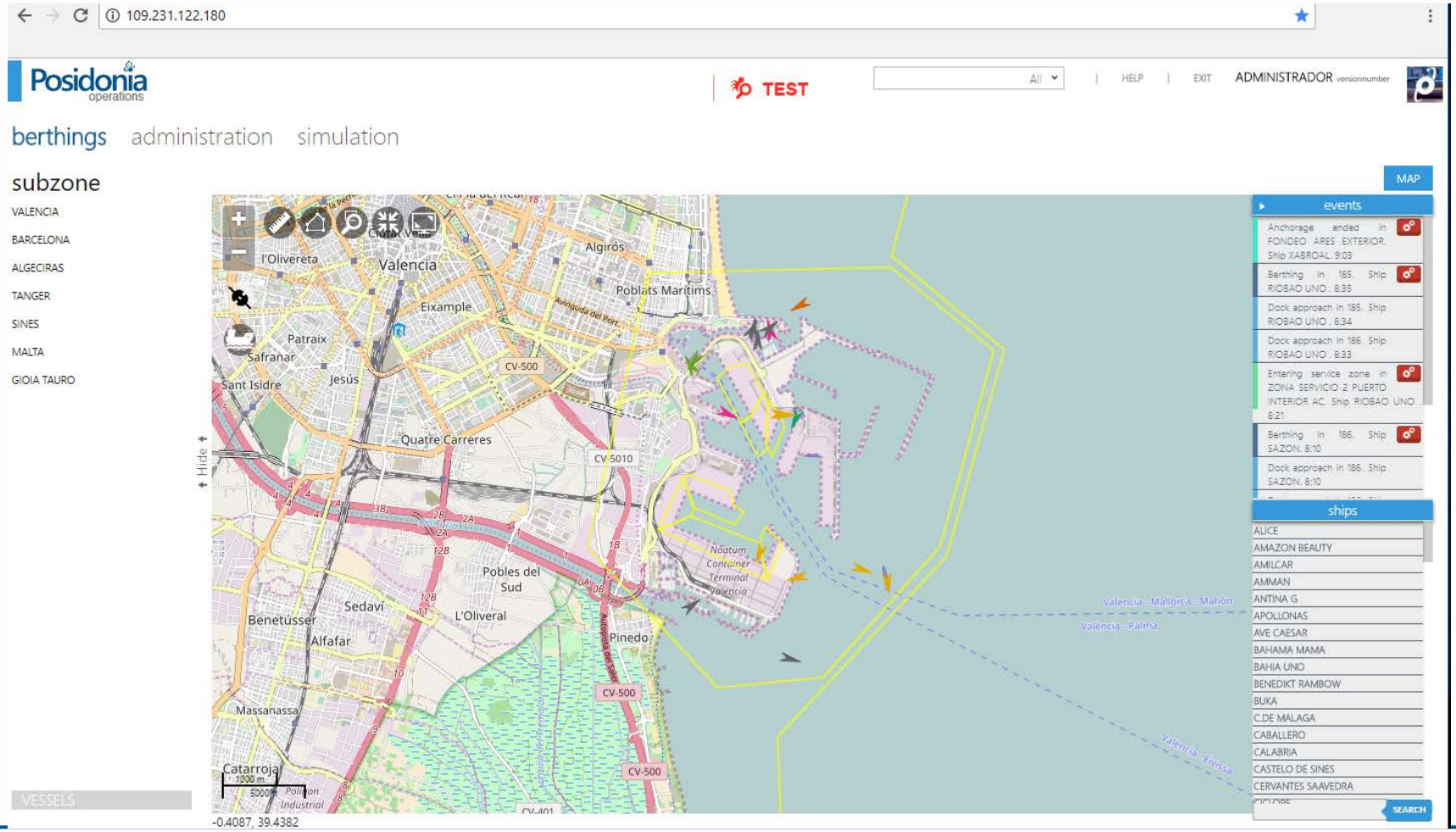
Highly specialized in the Maritime Industry

CMMI SM LEVEL 2
ABS Quality Evaluations, Inc. Management System Certification
ISO/IEC 20000-1:2011
SYSTEM CERTIFICATION UNE 166002 SGS
Posidonia Port Solutions Suite

POSIDONIA OPERATIONS – Architecture Machines



POSIDONIA OPERATIONS



Posidonia Operations

ALREADY IN PRODUCTION

- 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“

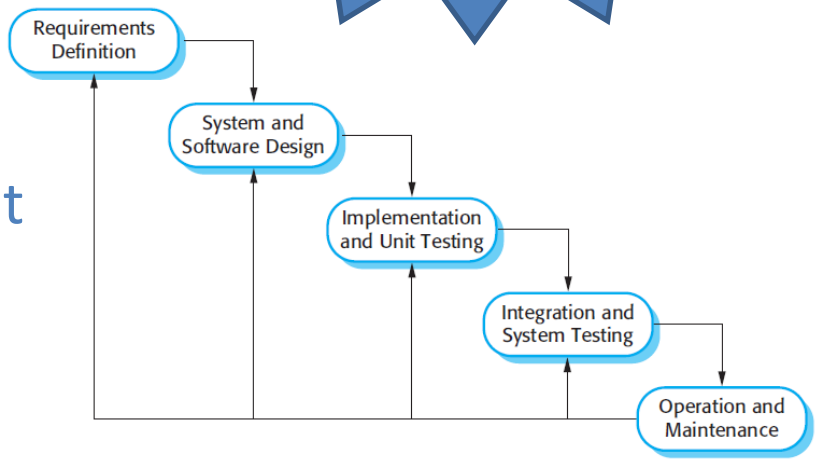
Main reasons of using DICE

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

Manual Test and Deployment



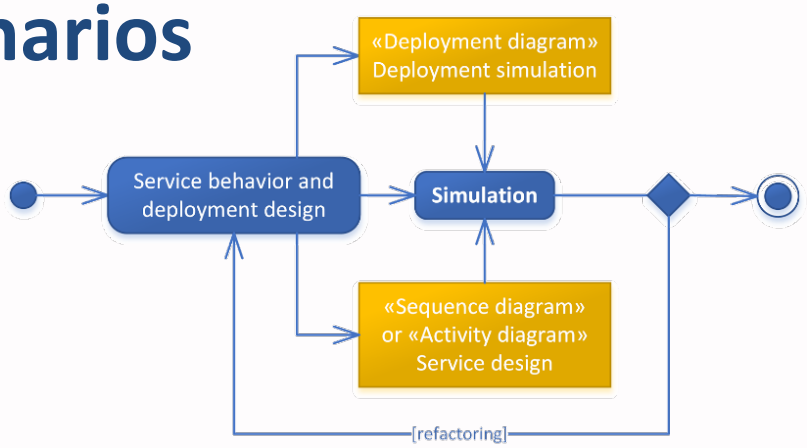
POSIDONIA OPERATIONS - Challenges

- **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

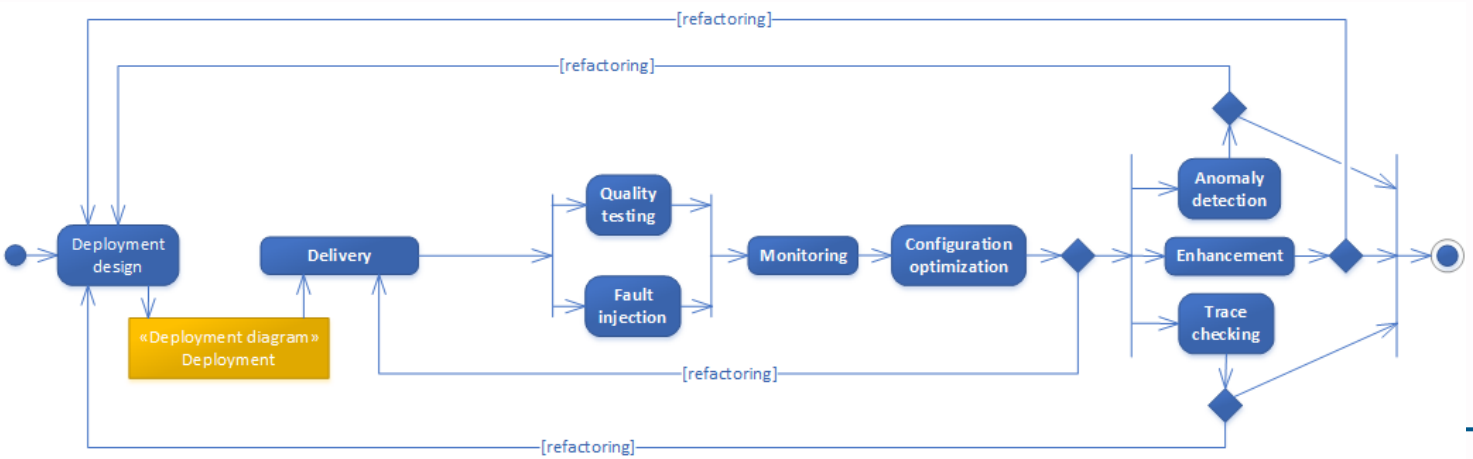
Standalone: Simulation

"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

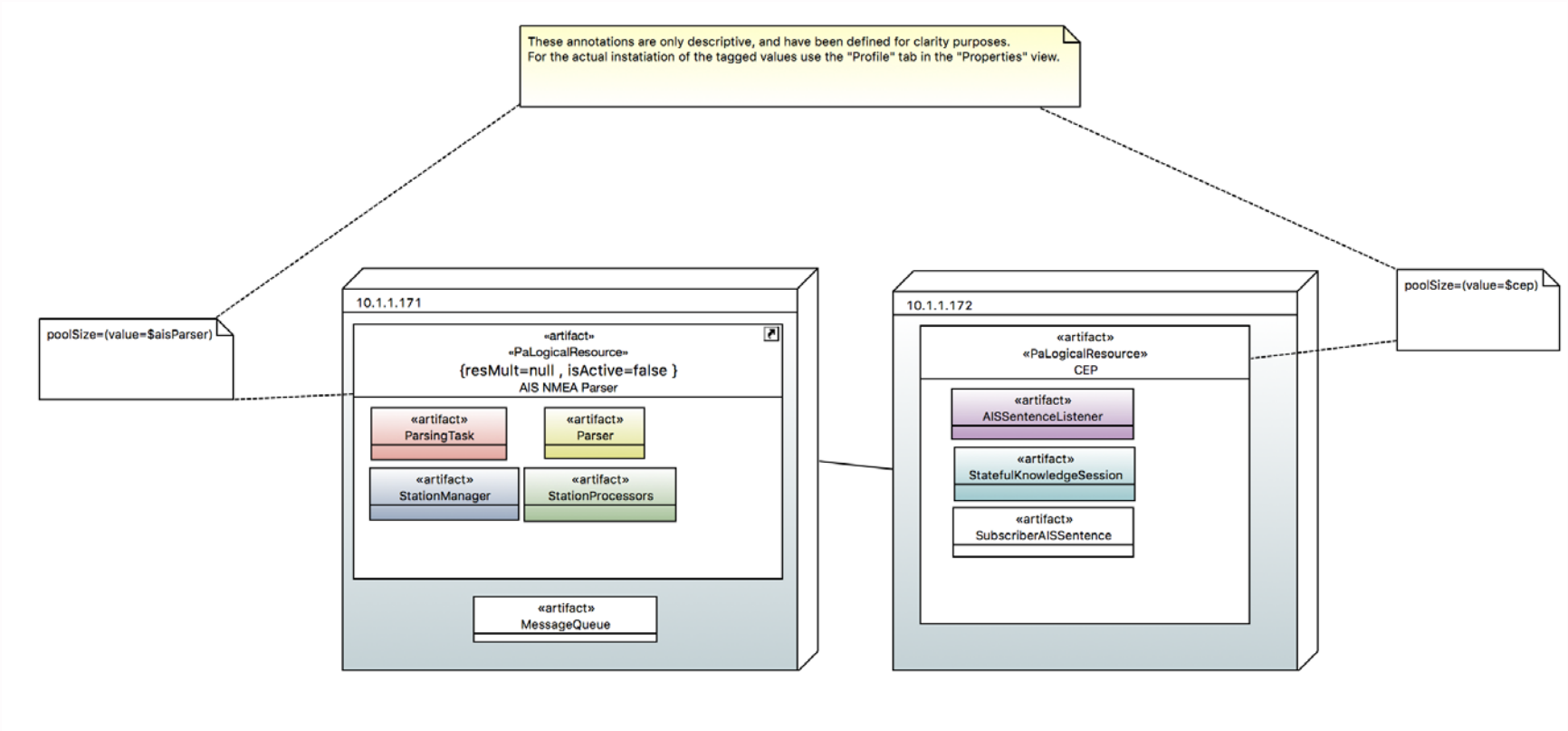
Use a DPIM (Platform Independent Model) to specify the system.

- MDA-UML Models enriched with DICE Profile.

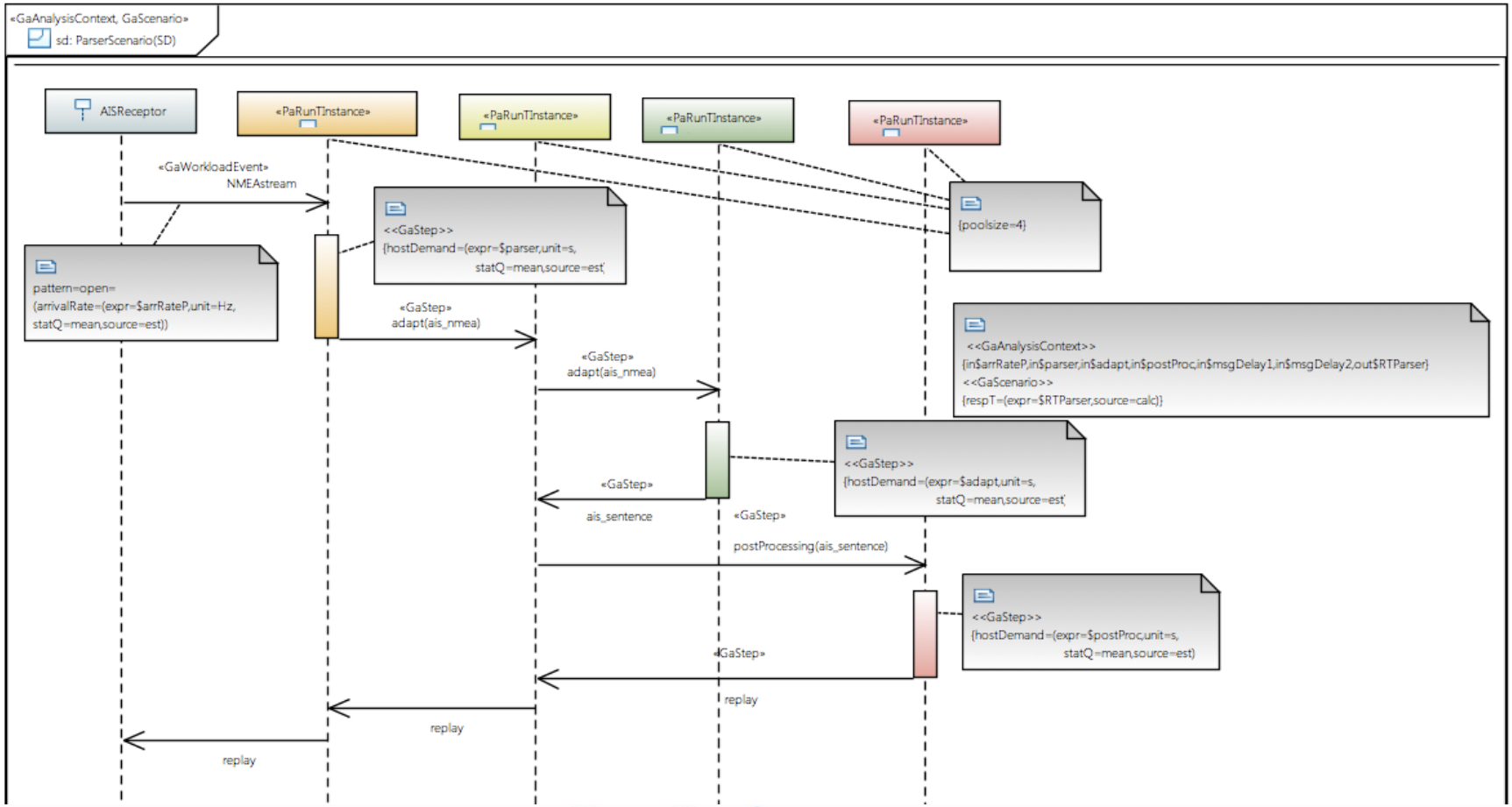
Given a DPIM model, the simulation tool tells us the maximum throughput

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

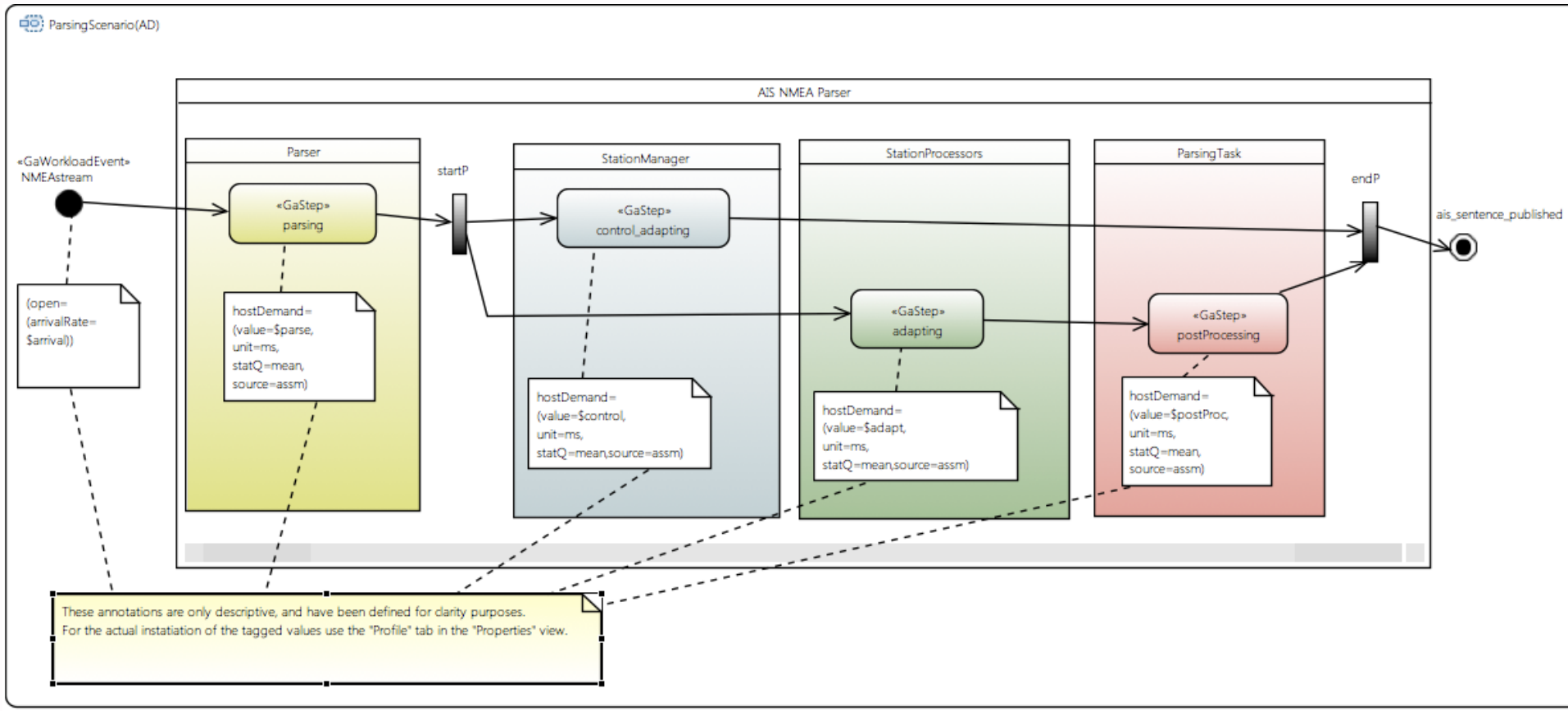
Simulation Tool – Model



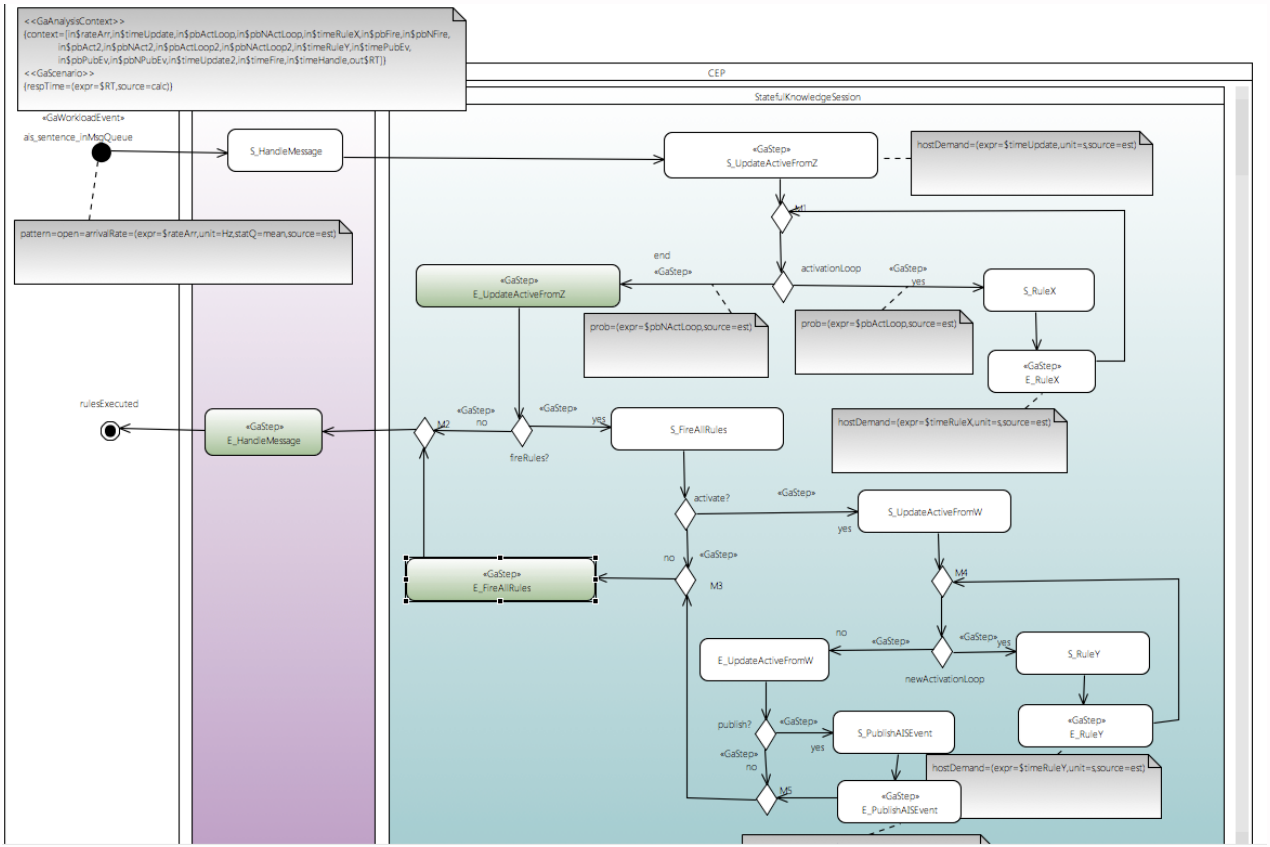
Simulation Tool – Model



Simulation Tool – Model

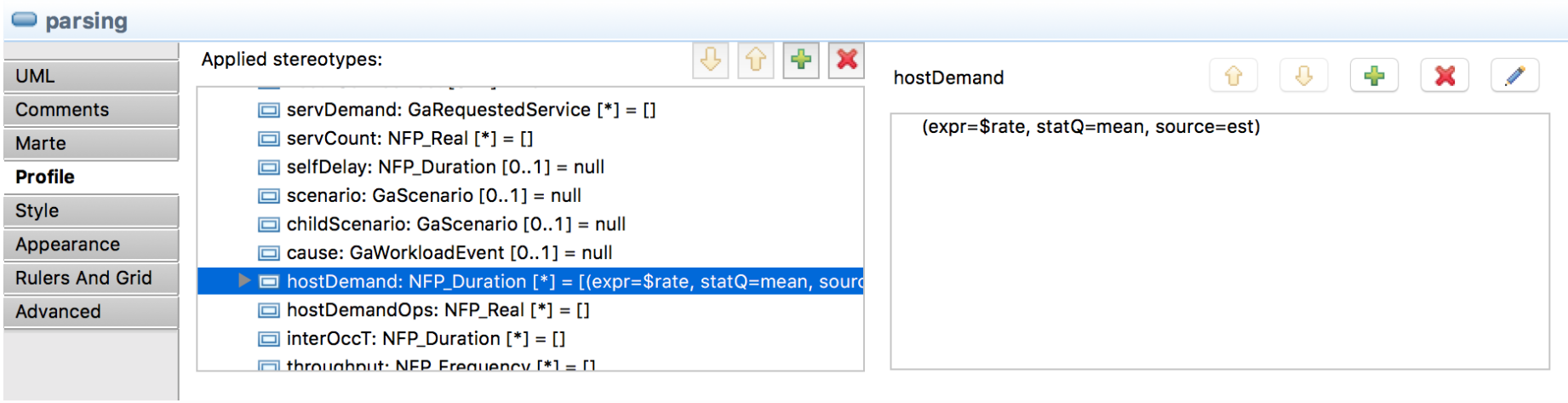


Simulation Tool – Model



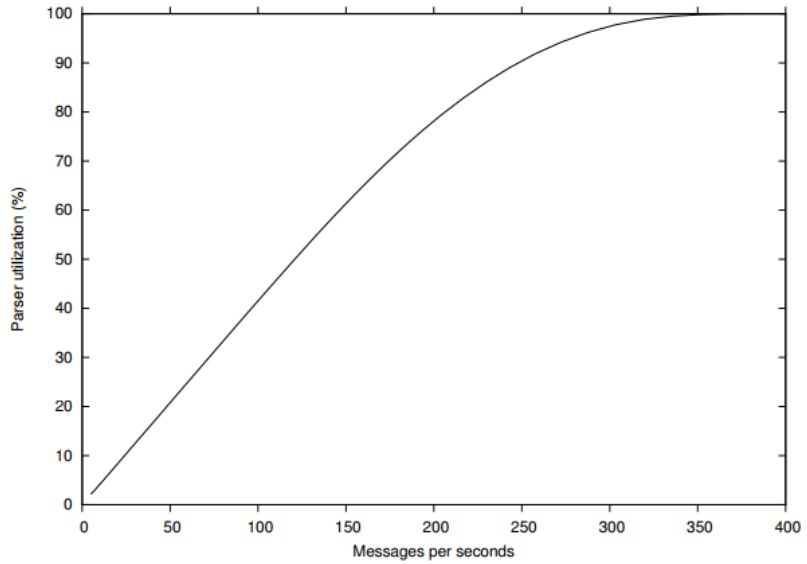
Simulation Tool – Configuration

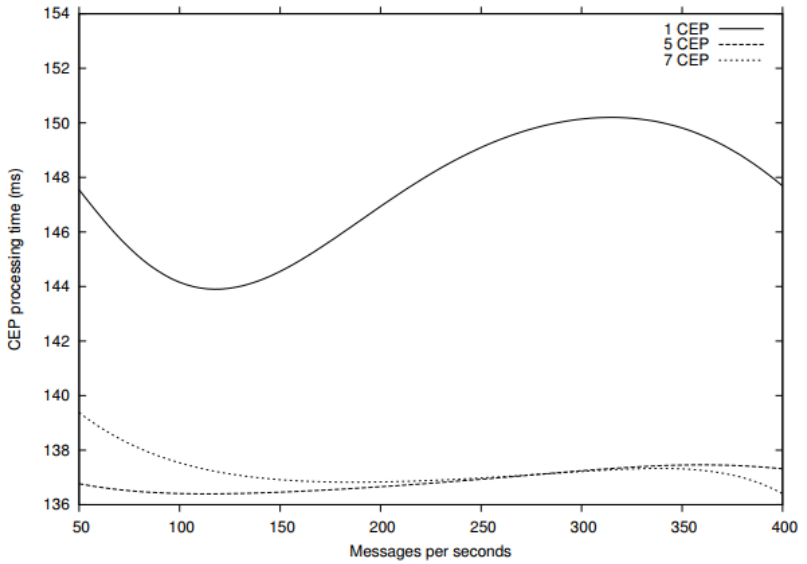
- SIMULATION PARAMETERS:
  - Cost estimation of each step (the actual cost is produced by the monitoring tool)
  - Message rate



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

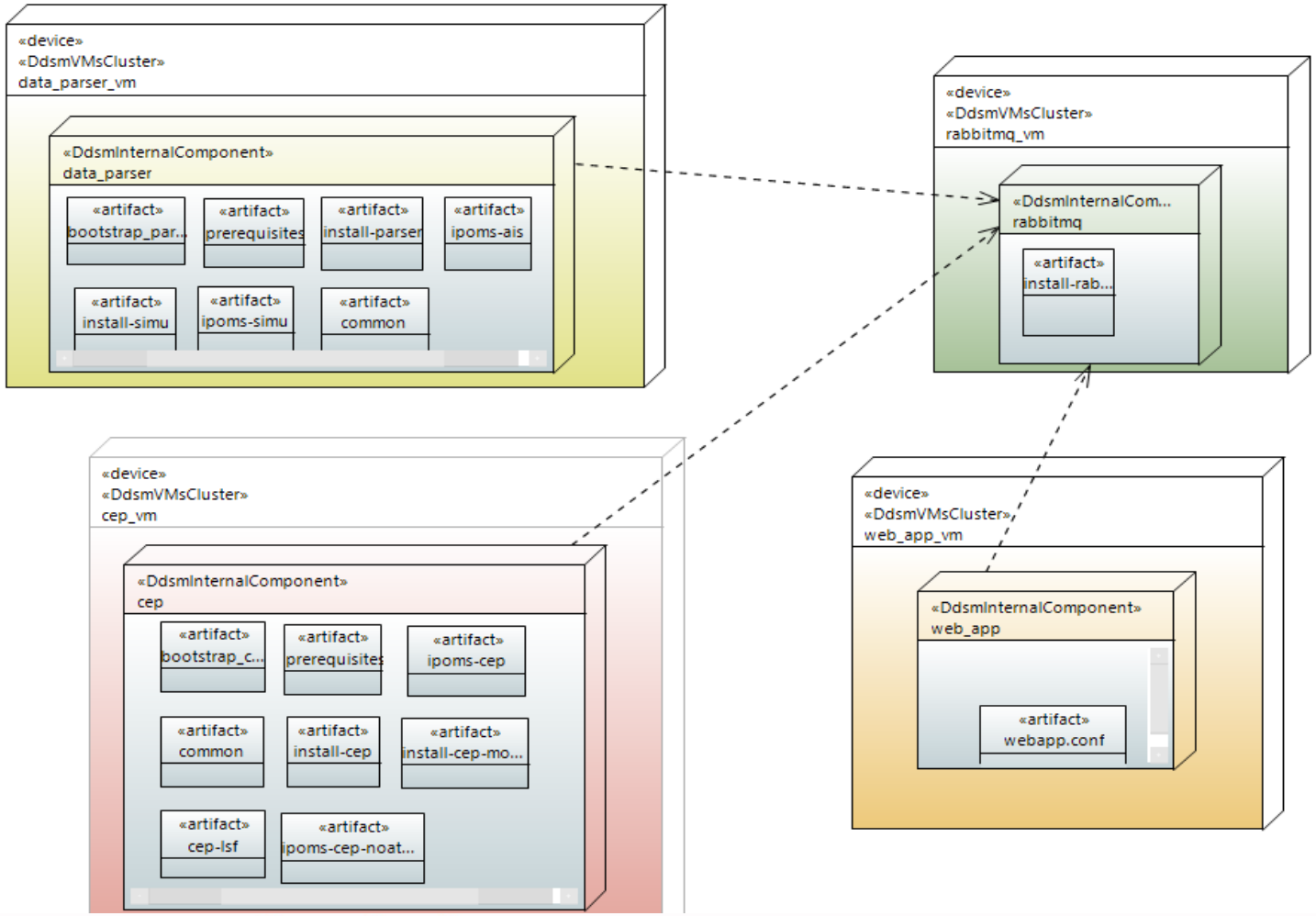
- How to reduce the time required to deploy the application?
- how to make deployment easier and not require much technical knowledge?

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**

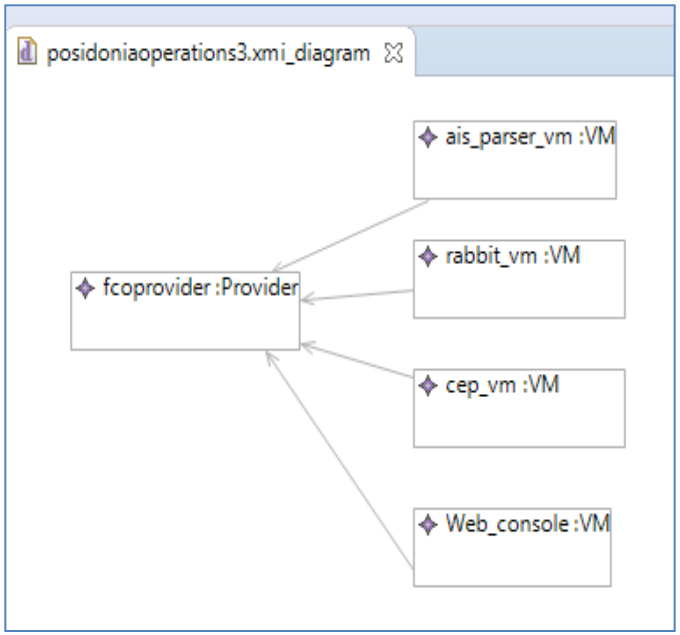
DICER-DEPLOYMENT

Posidonia - DDSM



DICER-DEPLOYMENT

IDE – DICER



Model to Text Transformation

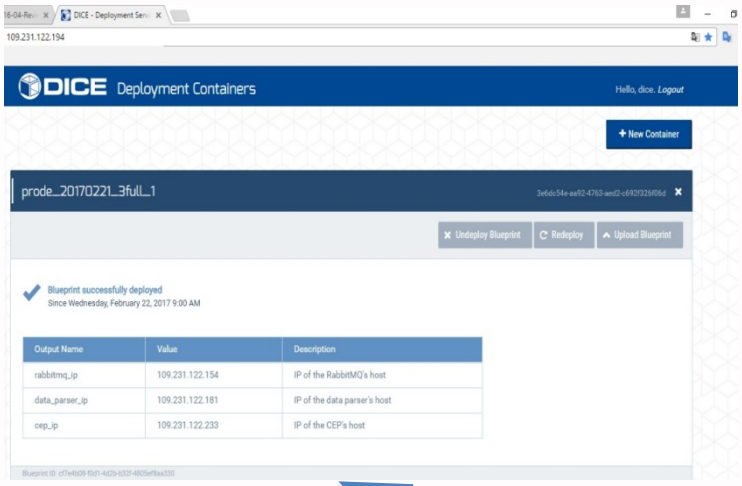
```
tosca_definitions_version: cloudify_dsl_1_3

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

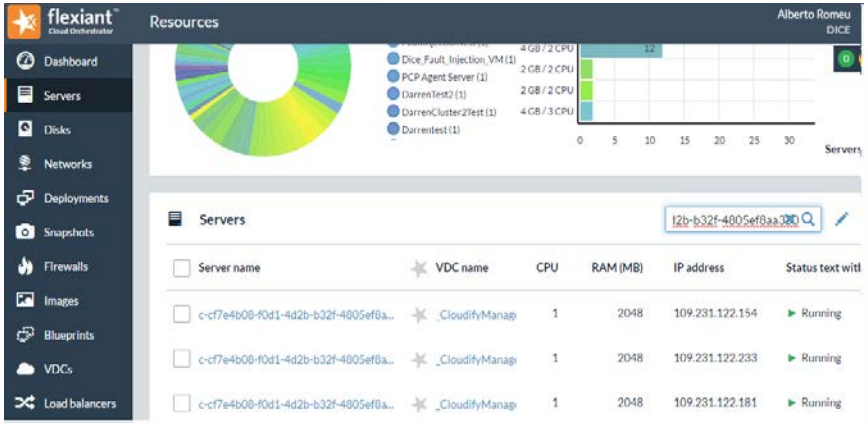
DEPLOYMENT



Upload

flexiant Cloud Orchestrator

amazon web services



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)
- Continuos Deployment (productivity)
- Deployment diagrams (helps understand the system)
- Automatic Deployment (No experts needed, less error, faster, less cost)

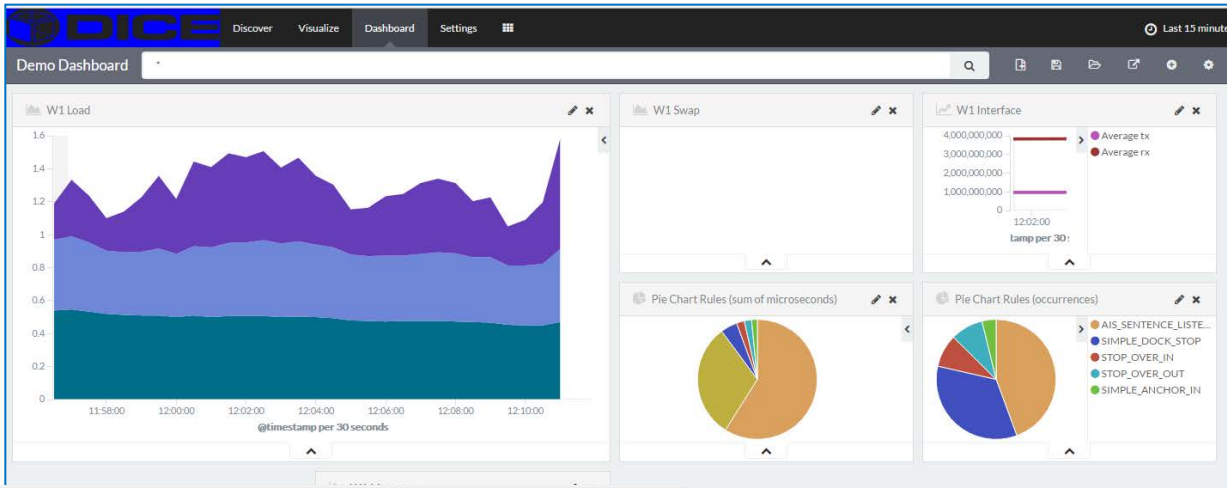
Monitoring- Problem to solve

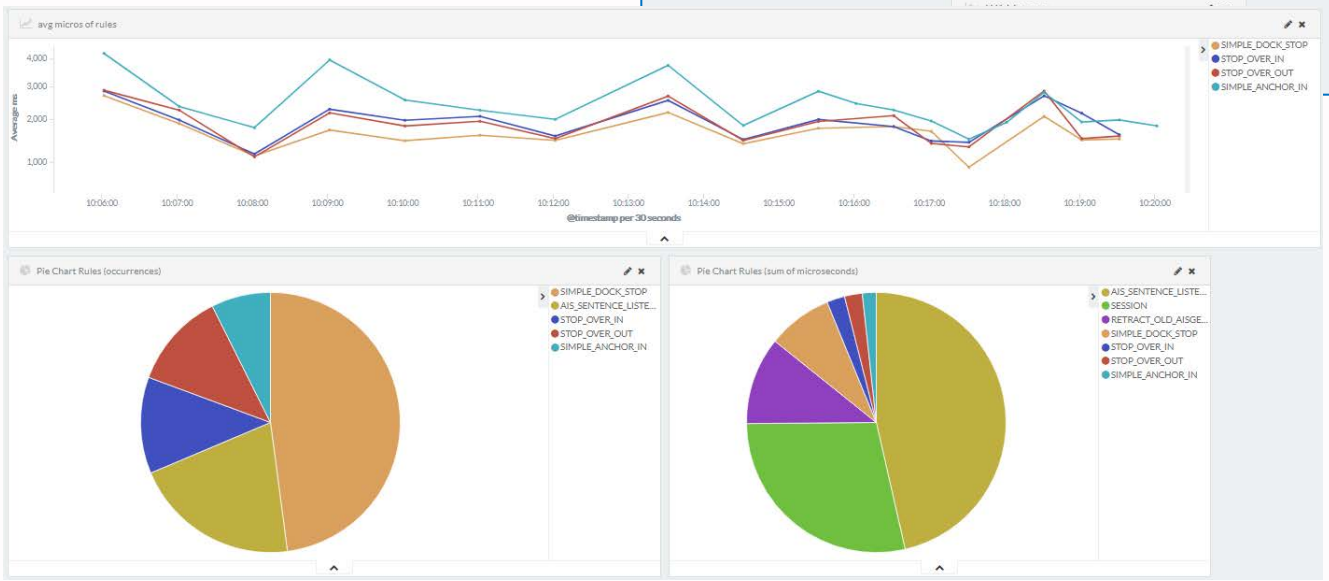
- How to control the resources used in Real time?
- How to control the Posidonia throughput in real time?
- How to analyze vessels routes inside the port?
- How to use the data obtained to improve the system and to make test?

Monitoring

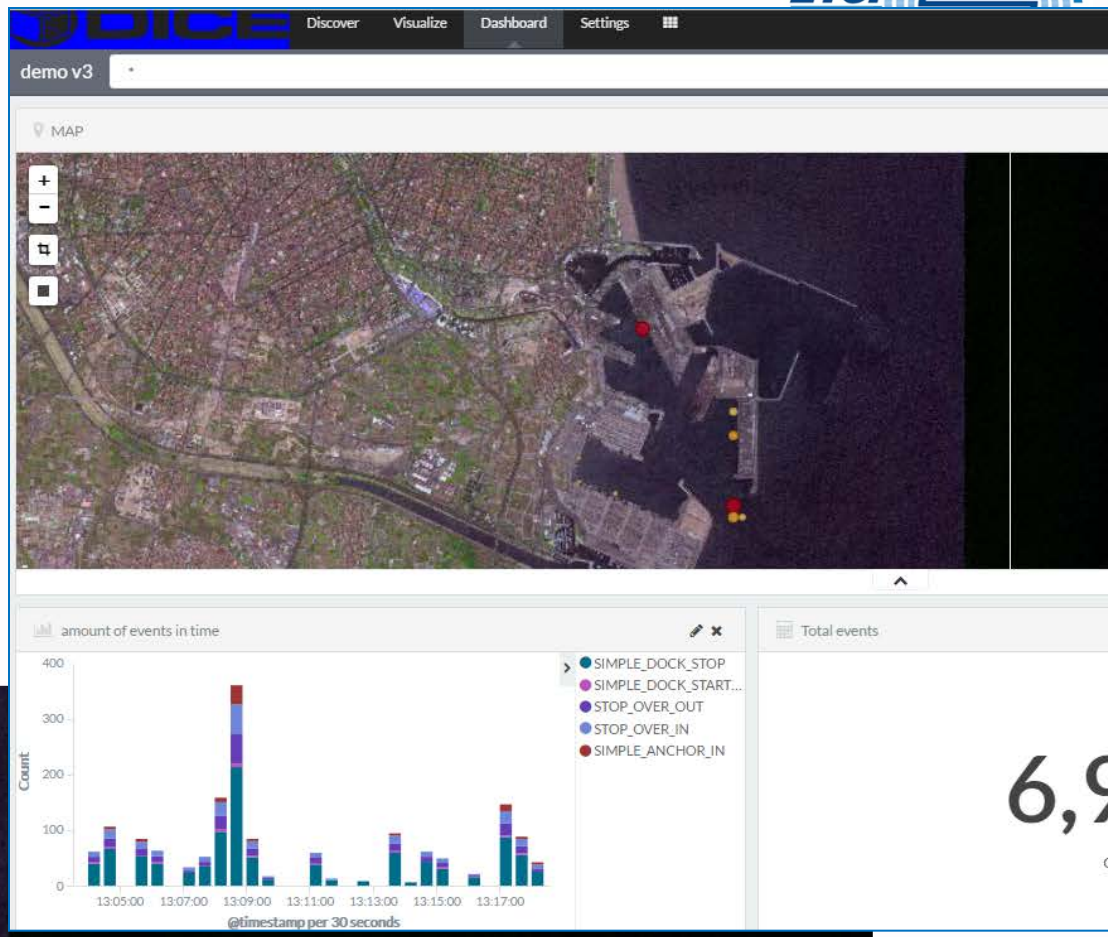
- DICE monitoring platform (DMon) collects, stores, indexes and visualizes monitoring data in real-time from applications running on Big Data frameworks.
- Monitoring Posidonia Operations “running” provides performance metrics.
- These metrics can be used to redesign the simulation model to improve the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

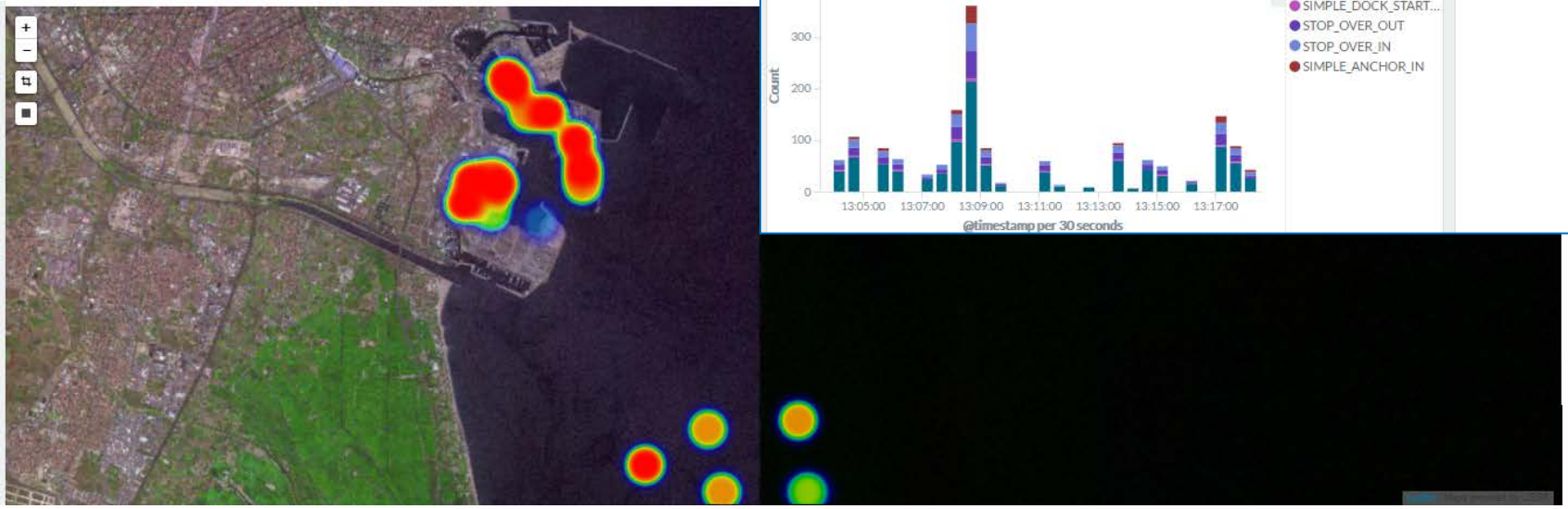
Monitoring





Monitoring





Monitoring – Results

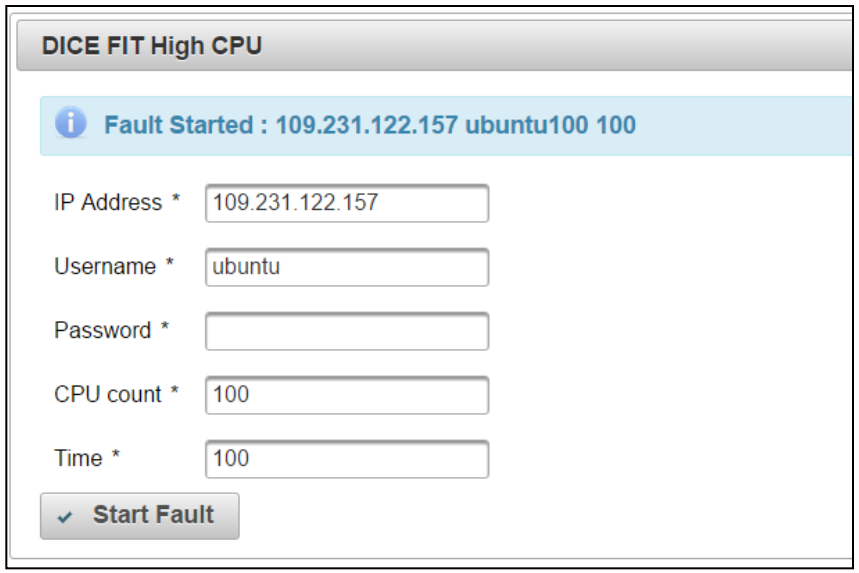
- Control about resources used in real time
- Use log files to visualise/control the system in real time
- Use log files as an input for other DICE tools to automatize some test and improve the quality of the system

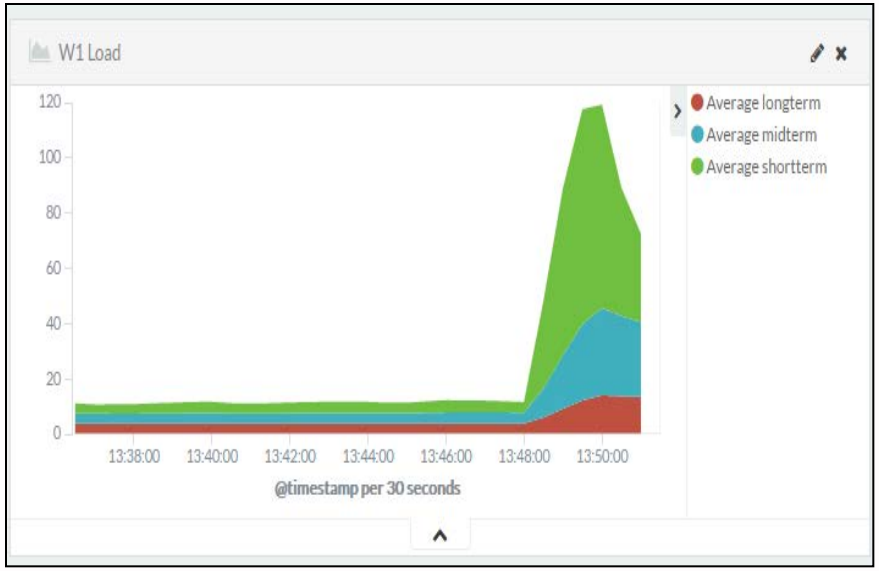
Fault injection tool- Problem to solve

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

Fault injection tool

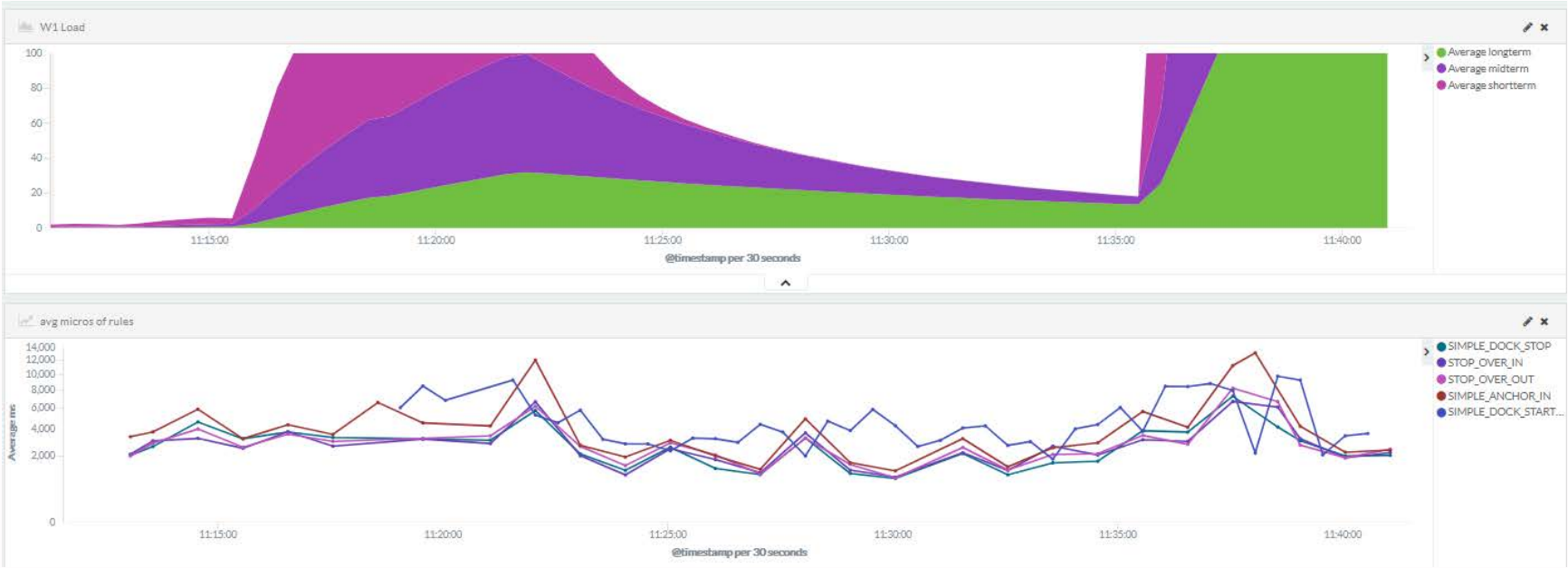
- DICE Fault Injection tool (FIT) is used to generate faults within VM.
- FIU installs a third party tool in the VM to produce faults.





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 Rulues 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 minutesAutomatic 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

5th UCAAT User Conference on Advanced Automated Testing

ETSI

THE NEWS ASSET DEMONSTRATOR

Vasilis Papanikolaou, George Giotis

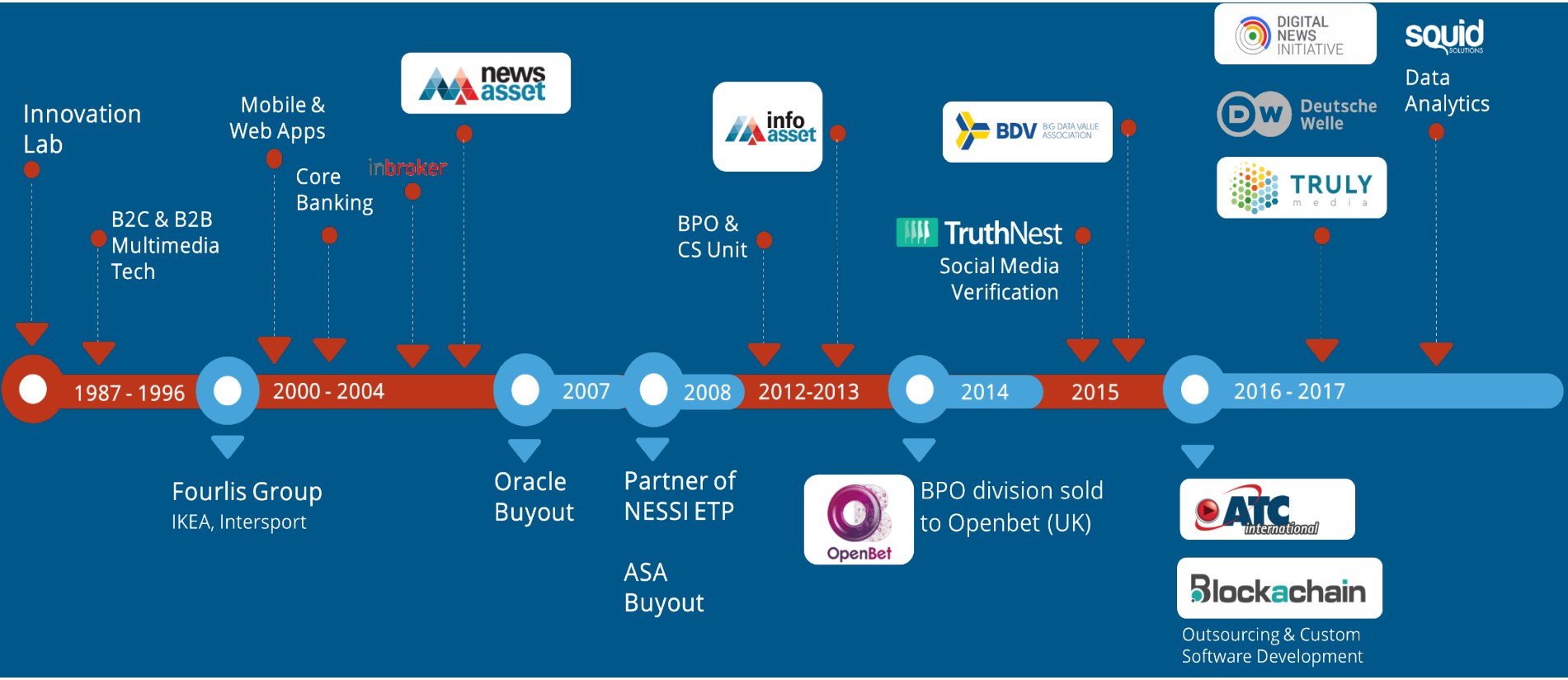
DICE

ATHENS TECHNOLOGY CENTER

Who we are

DICE

ATC MILESTONES: over the last 30 years



Departments & Activities

Media & Content Management

Web & Mobile Applications

Innovation Lab

Custom IT Services

5th UCAAT User Conference on Advanced Automated Testing

ETSI

NEWS ASSET

What News Asset is

news asset

DICE

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

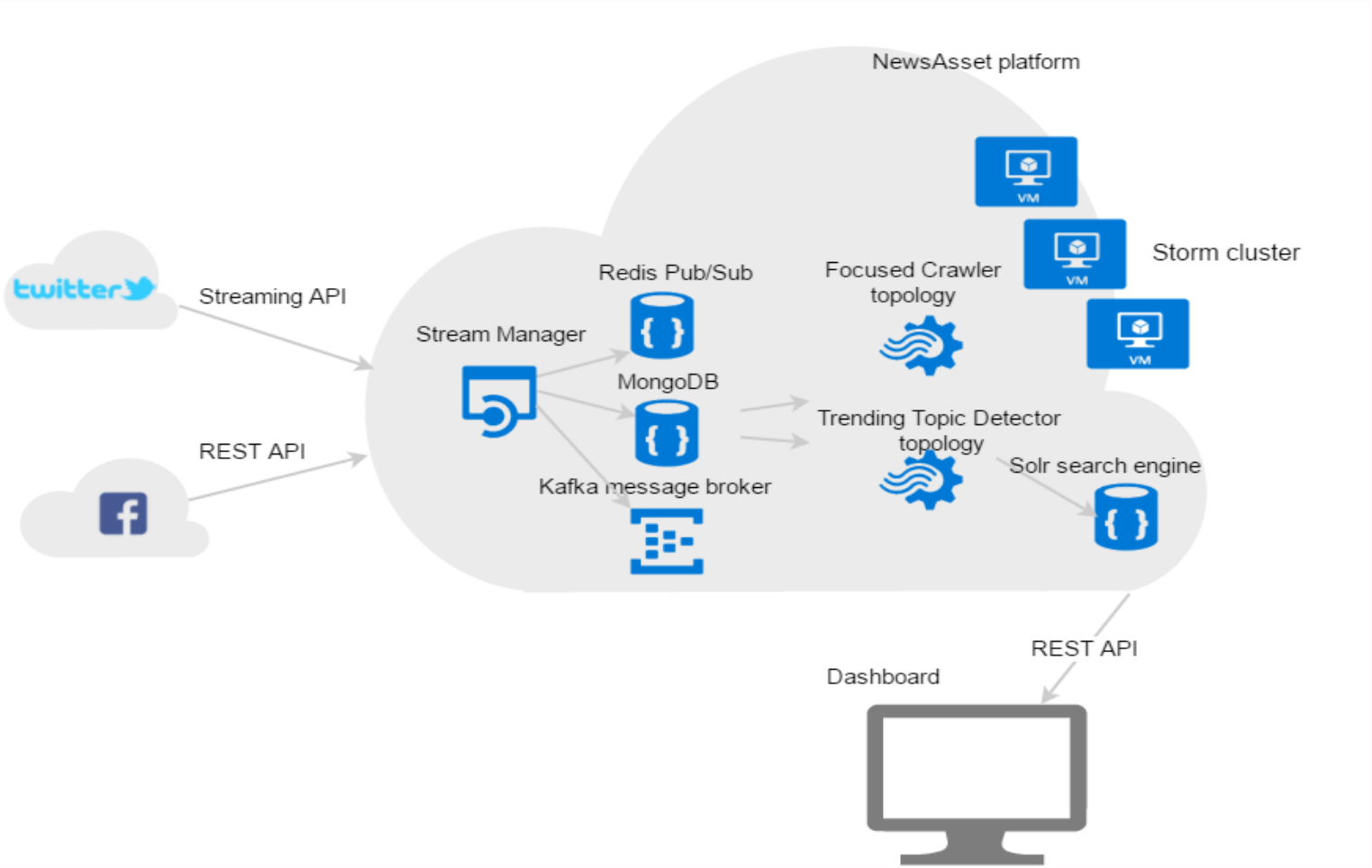
- Refactoring of the old-fashioned engine
  - related to cloud processing and Big Data technologies
- Reconfiguration
  - revise obsolete architecture with respect to quality-driven metrics
- Manage complexity
  - real-time responsiveness for temporal peaks of high computational demand

NEWS ORCHESTRATOR

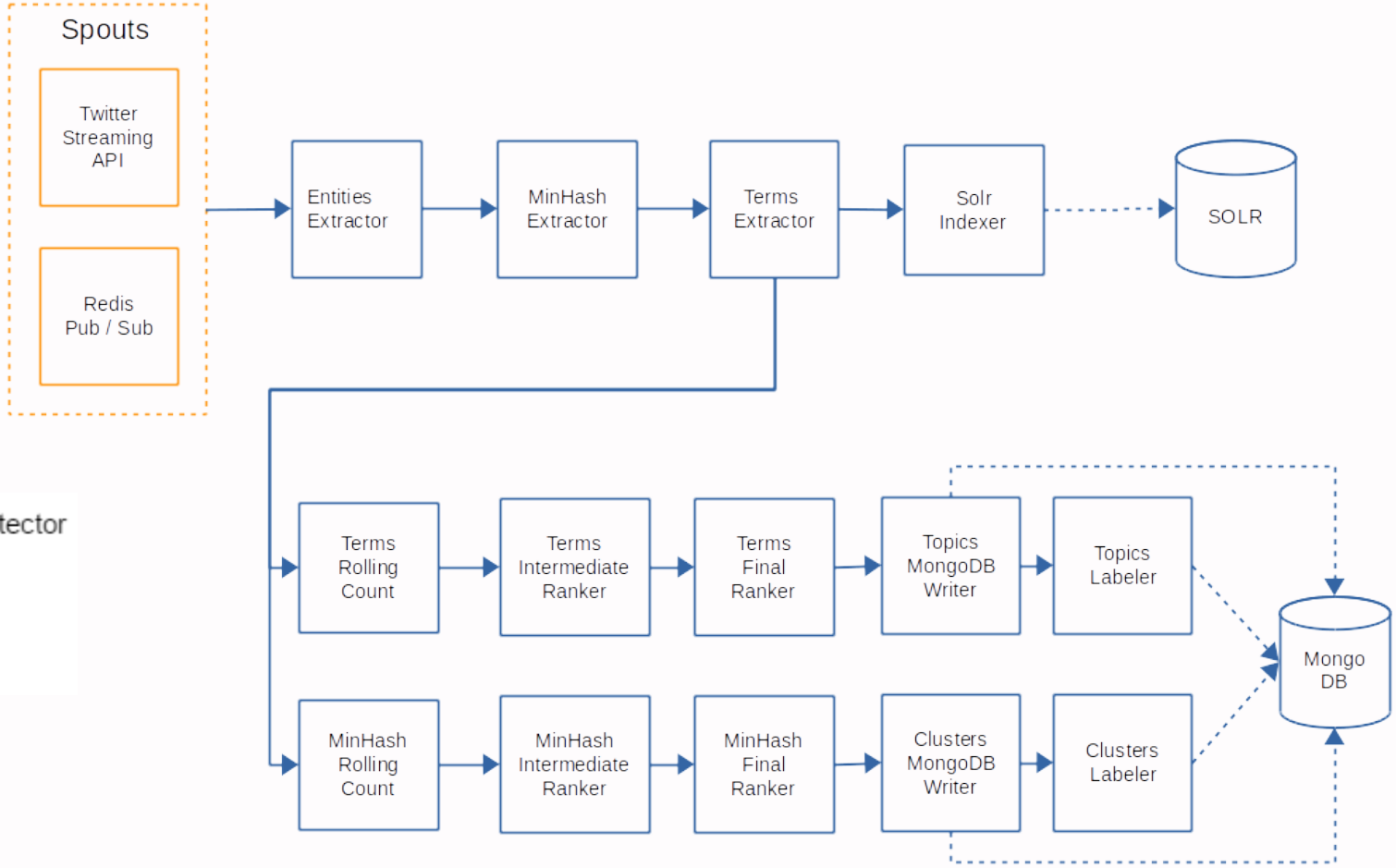
The Solution

DICE

News Orchestrator Architecture



Trending-Topic Detector Storm Topology



Trending Topic Detector topology

5th UCAAT User Conference on Advanced Automated Testing

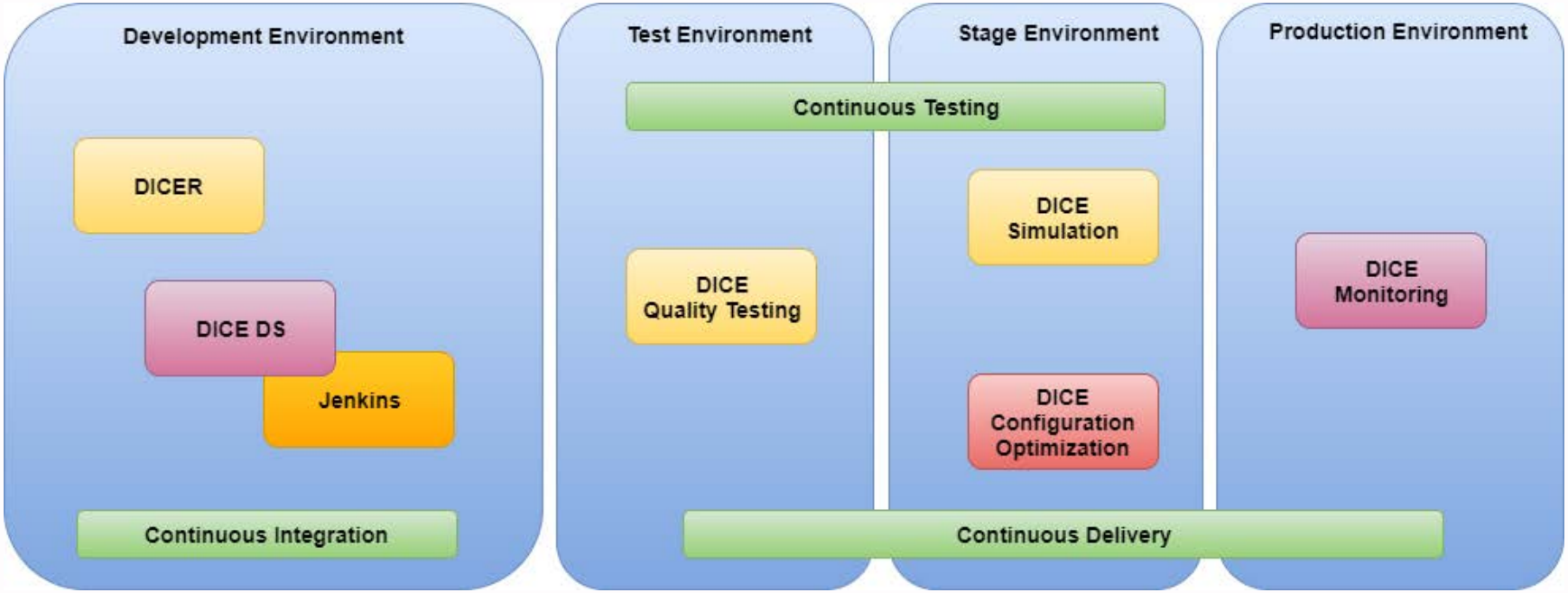
ETSI

NEWS ASSET TUTORIAL

News Asset Experience with DICE

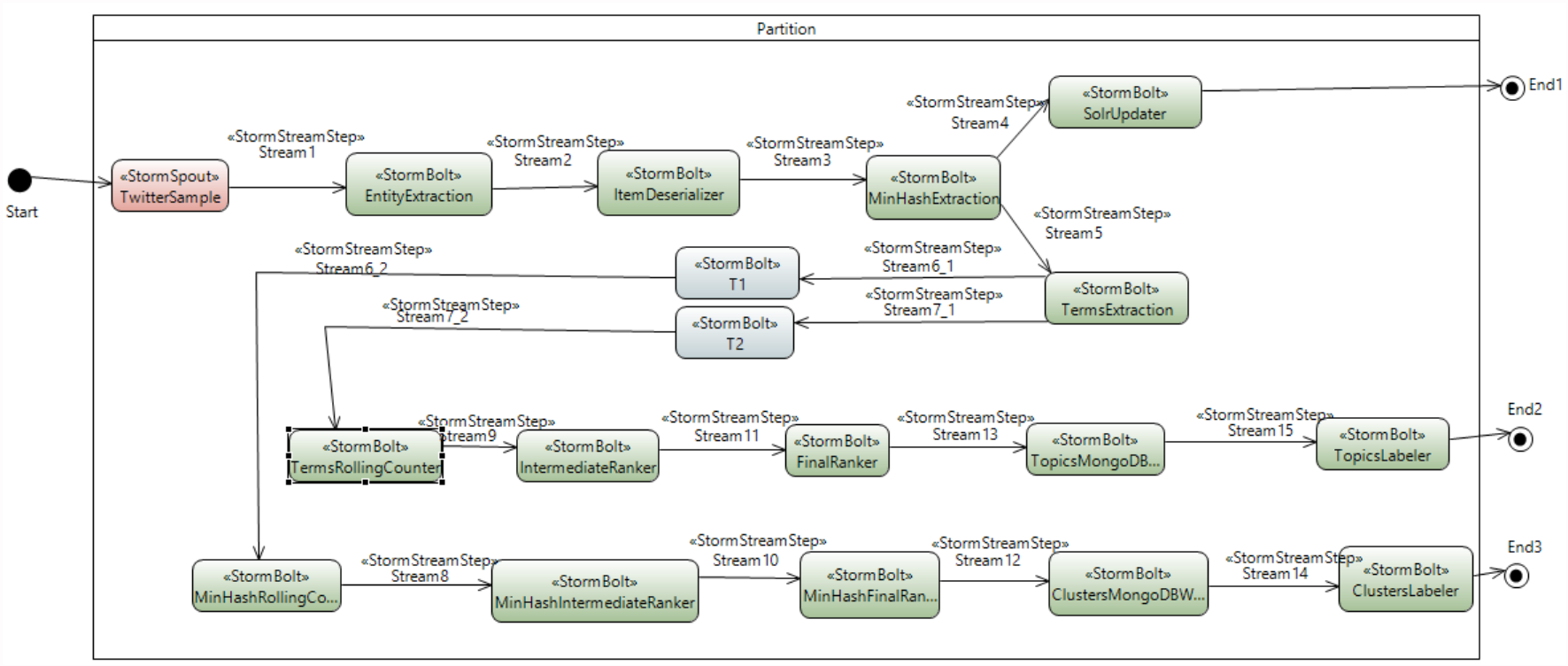
DICE

DevOps Lifecycle

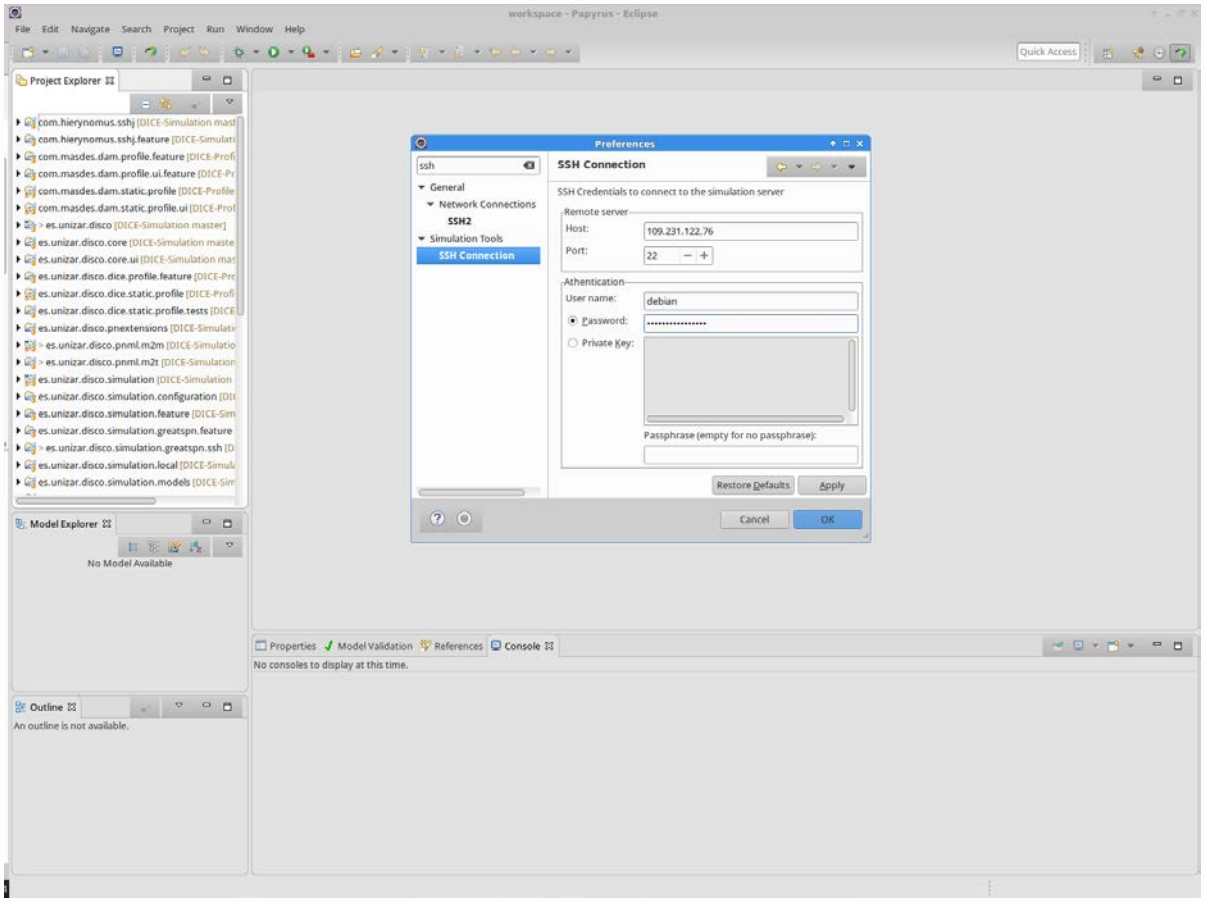


Simulation

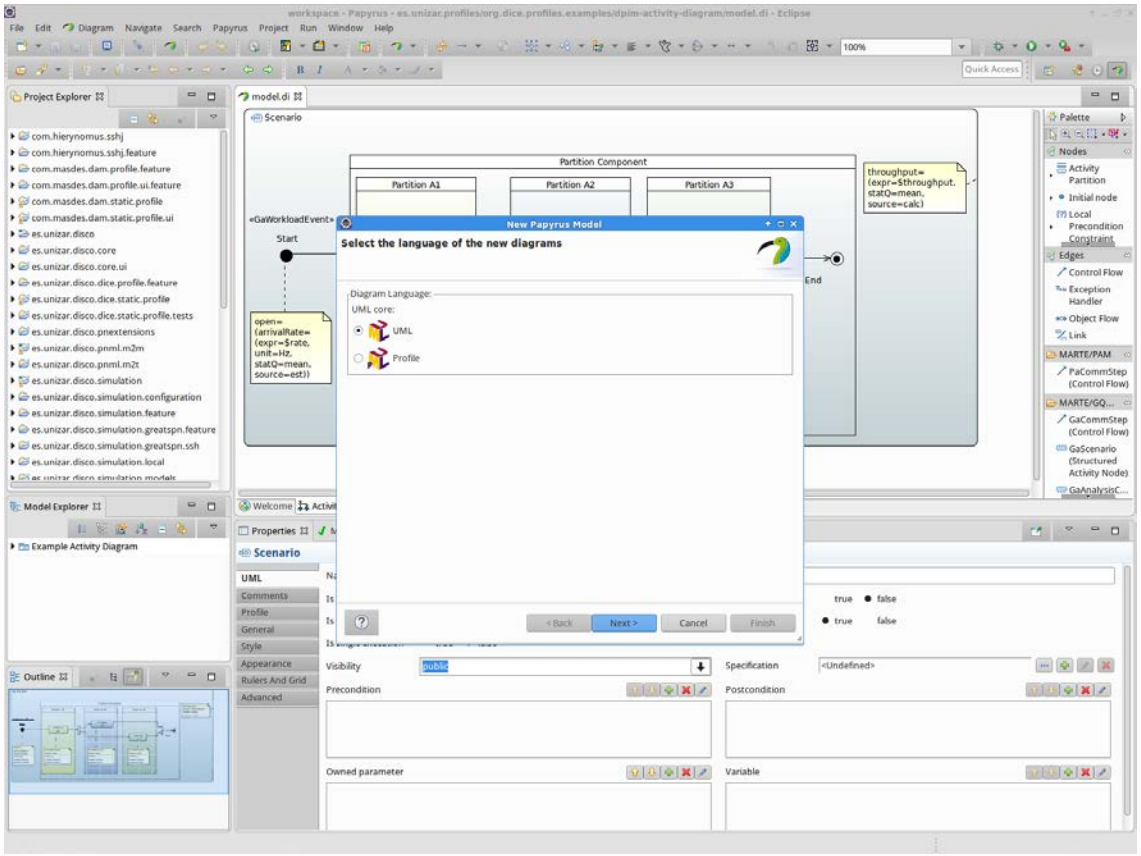
Motive: Predict the behavior of the system prior to the deployment in the cloud.



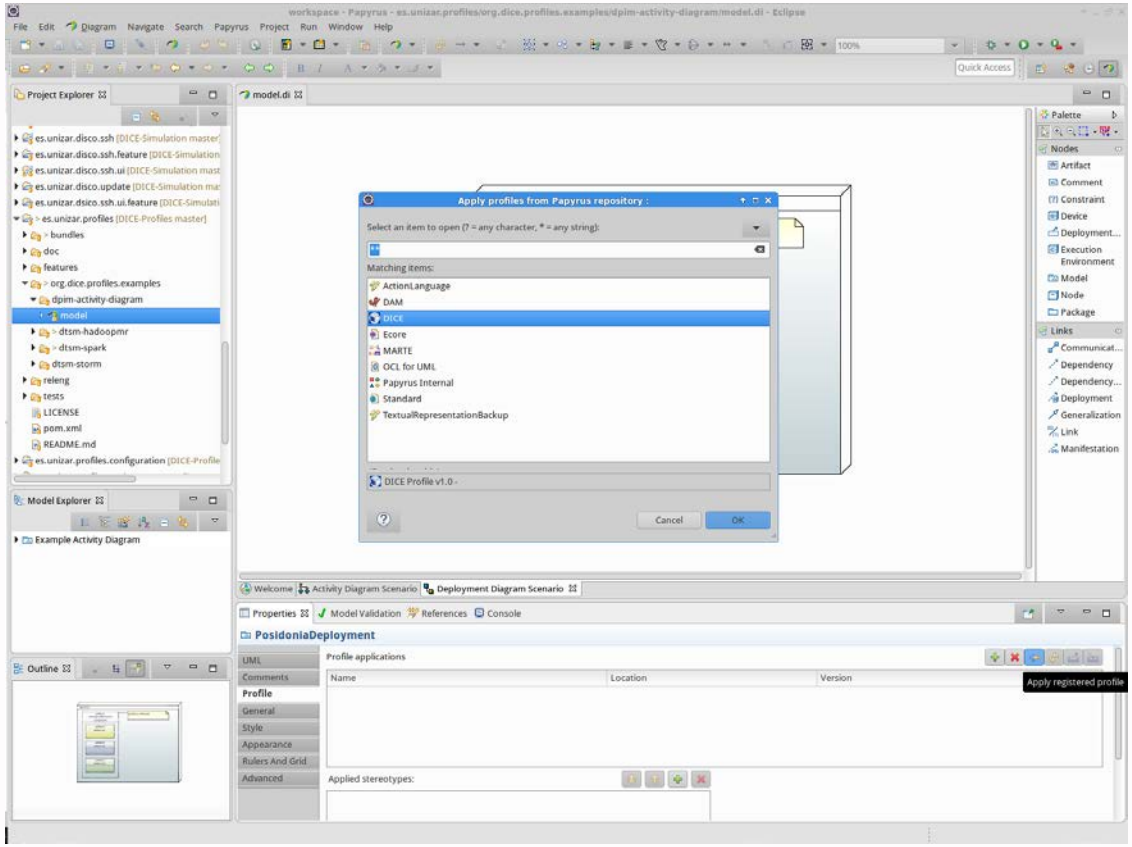
Simulation workflow (1)



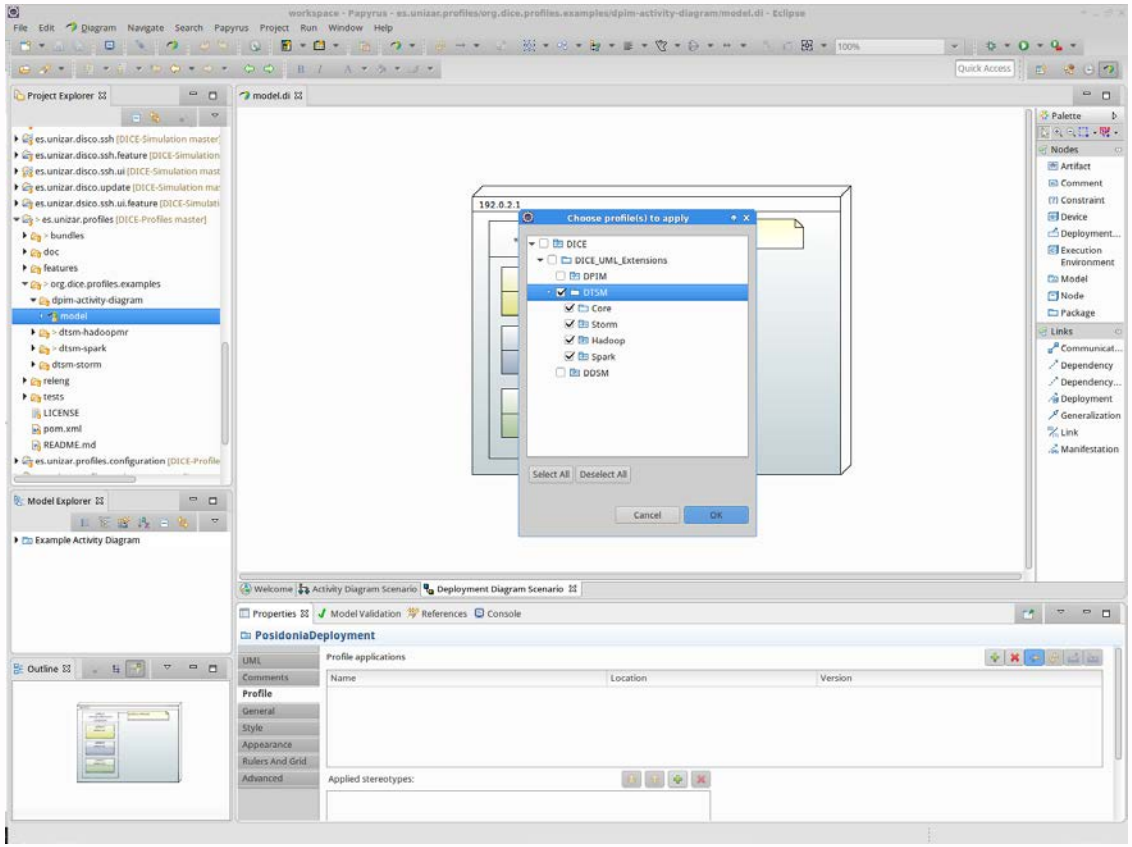
Simulation workflow (2)



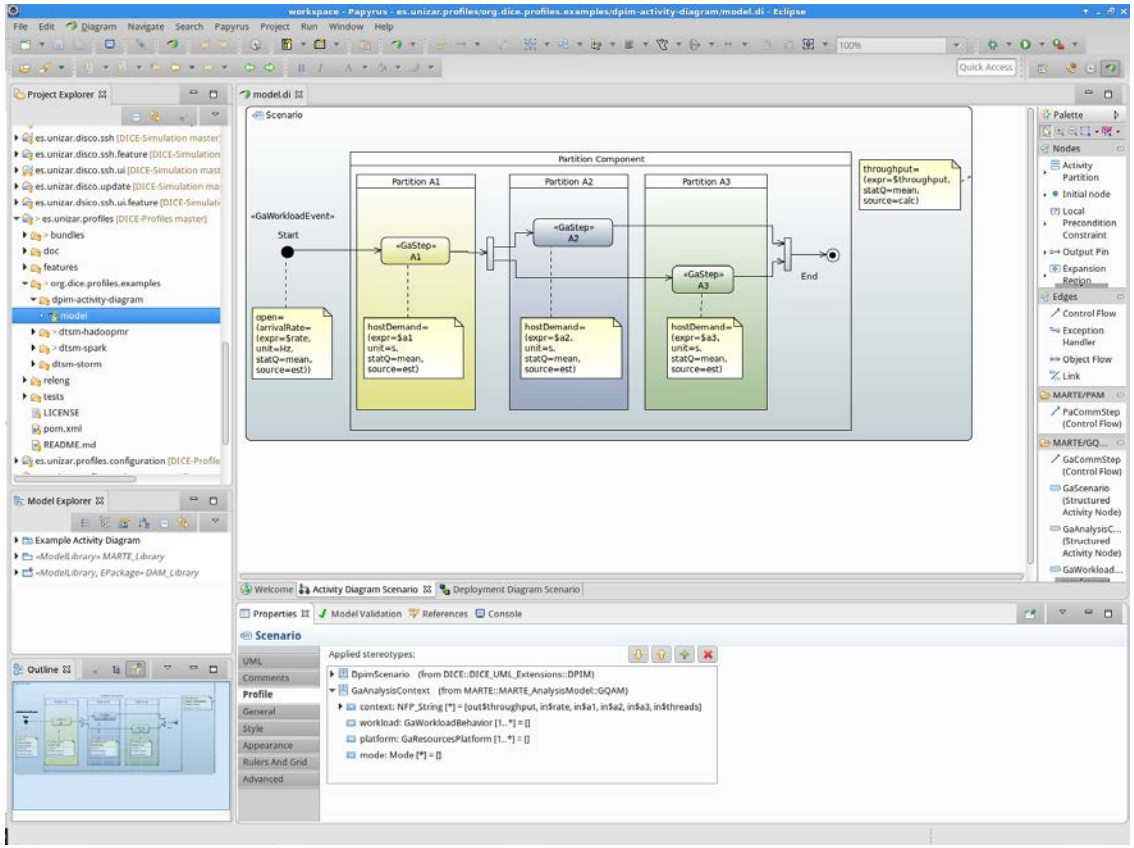
Simulation workflow (3)



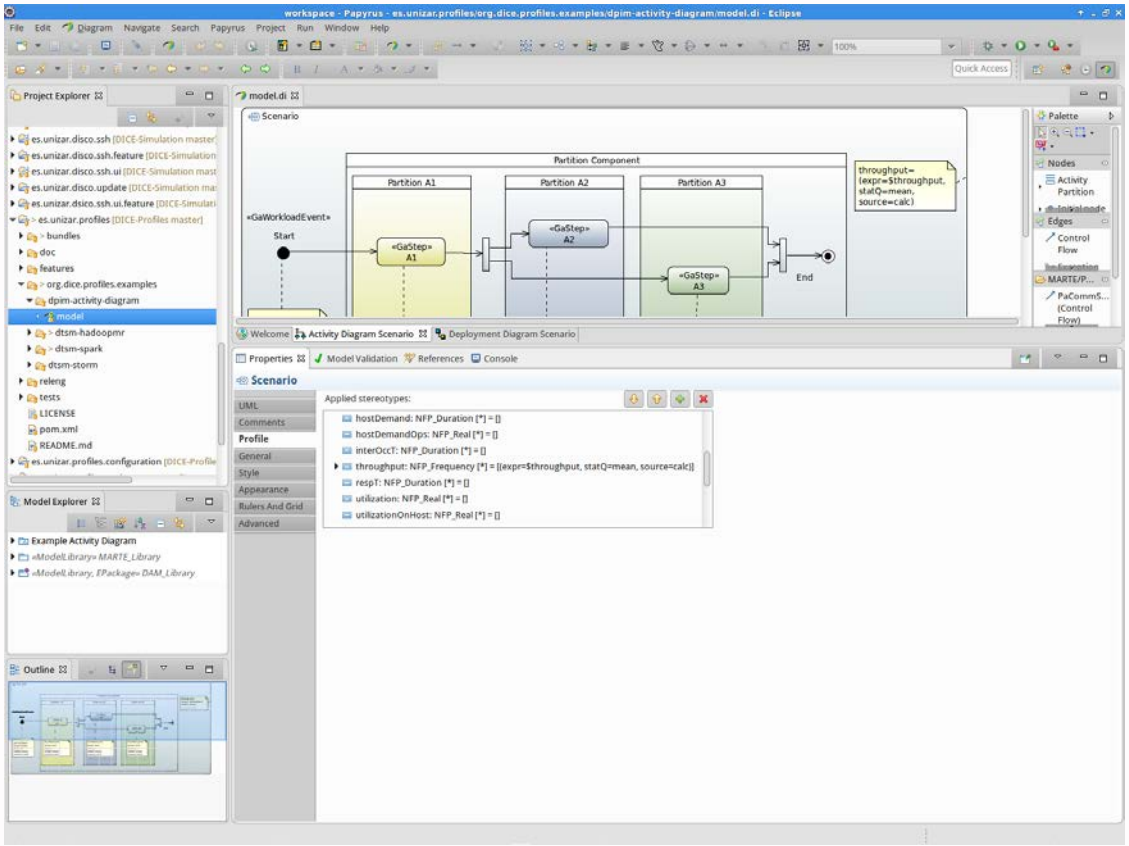
Simulation workflow (4)



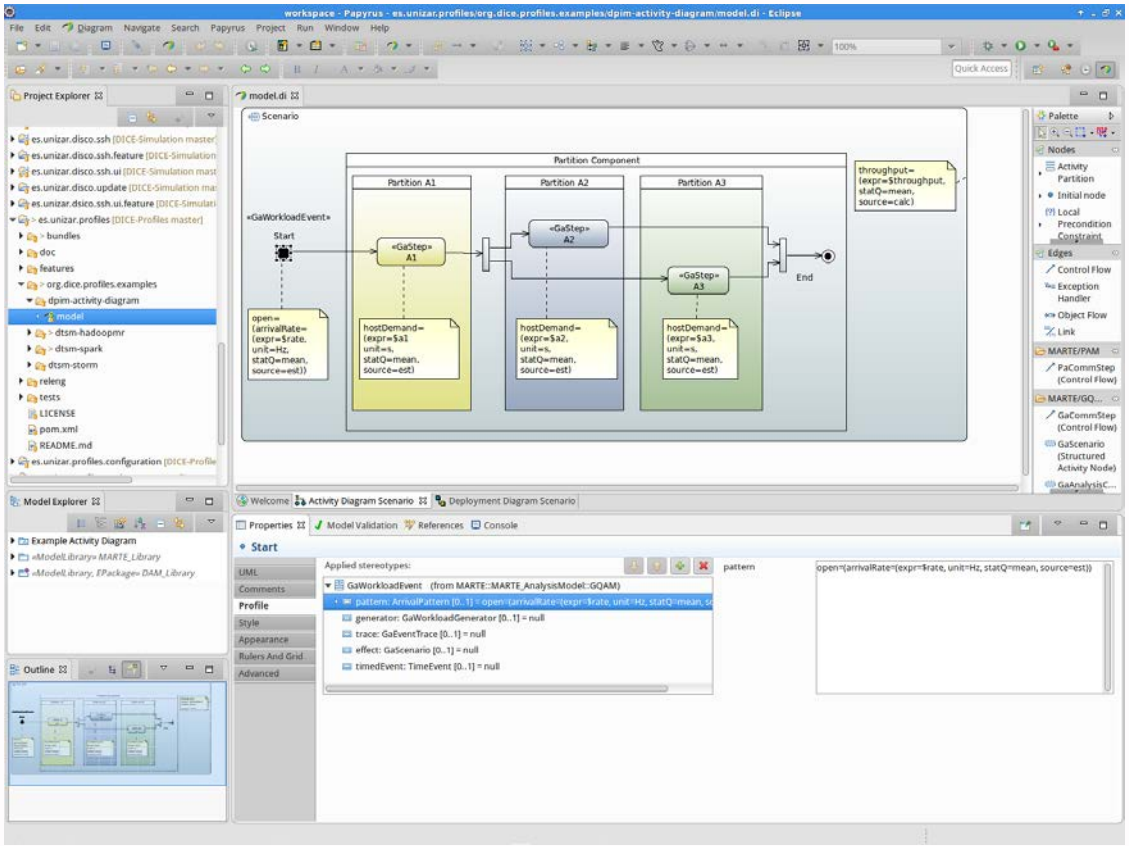
Simulation workflow (5)



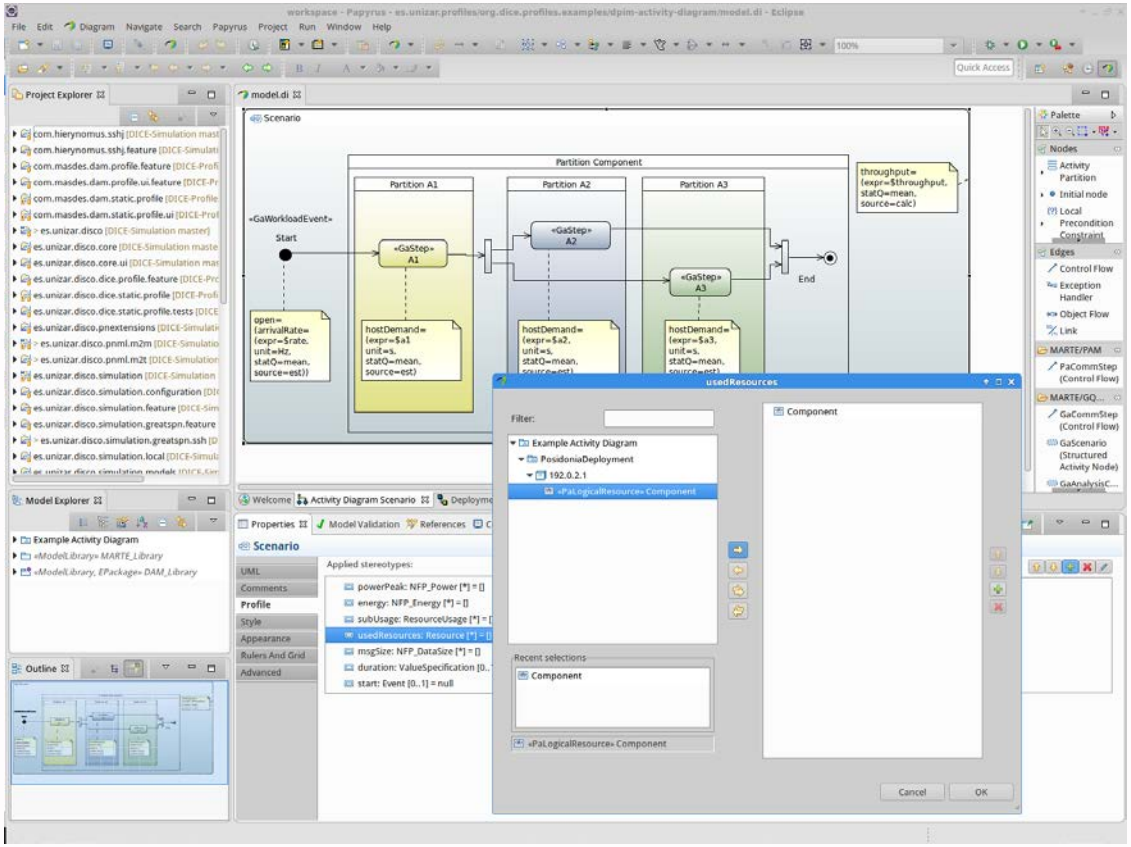
Simulation workflow (6)



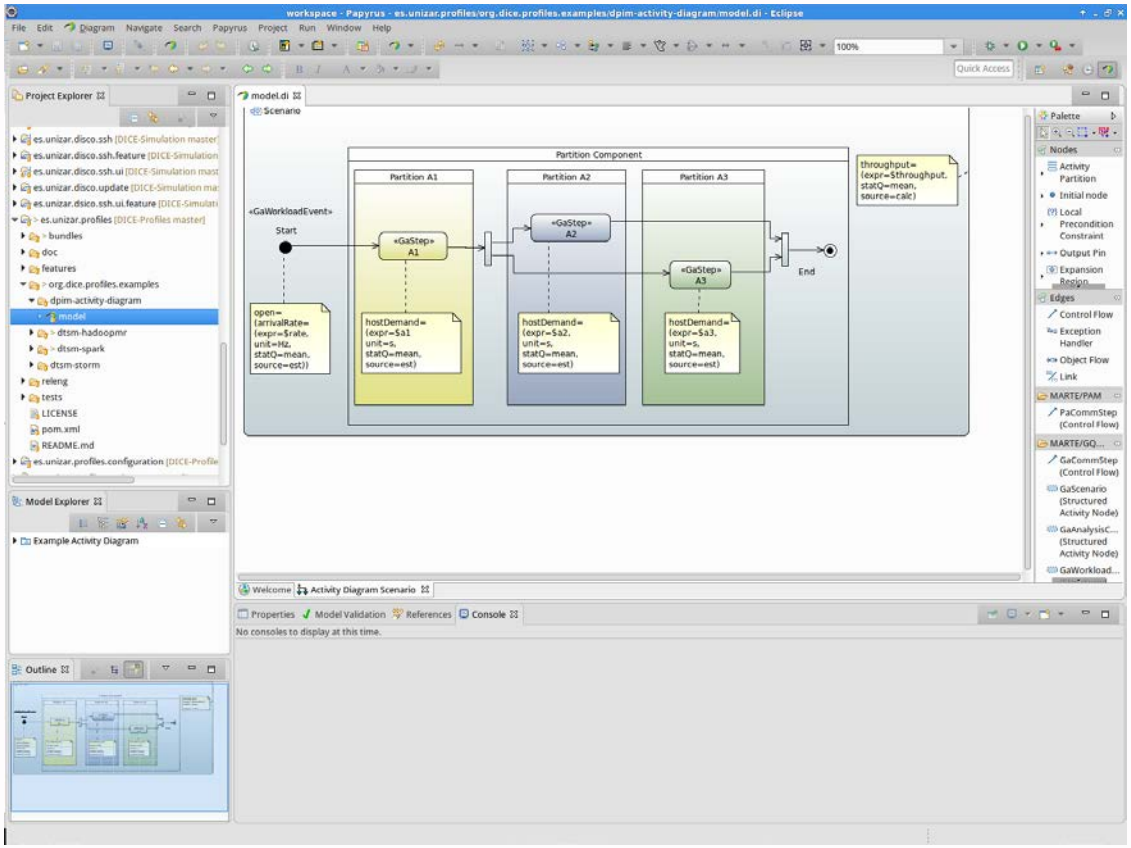
Simulation workflow (7)



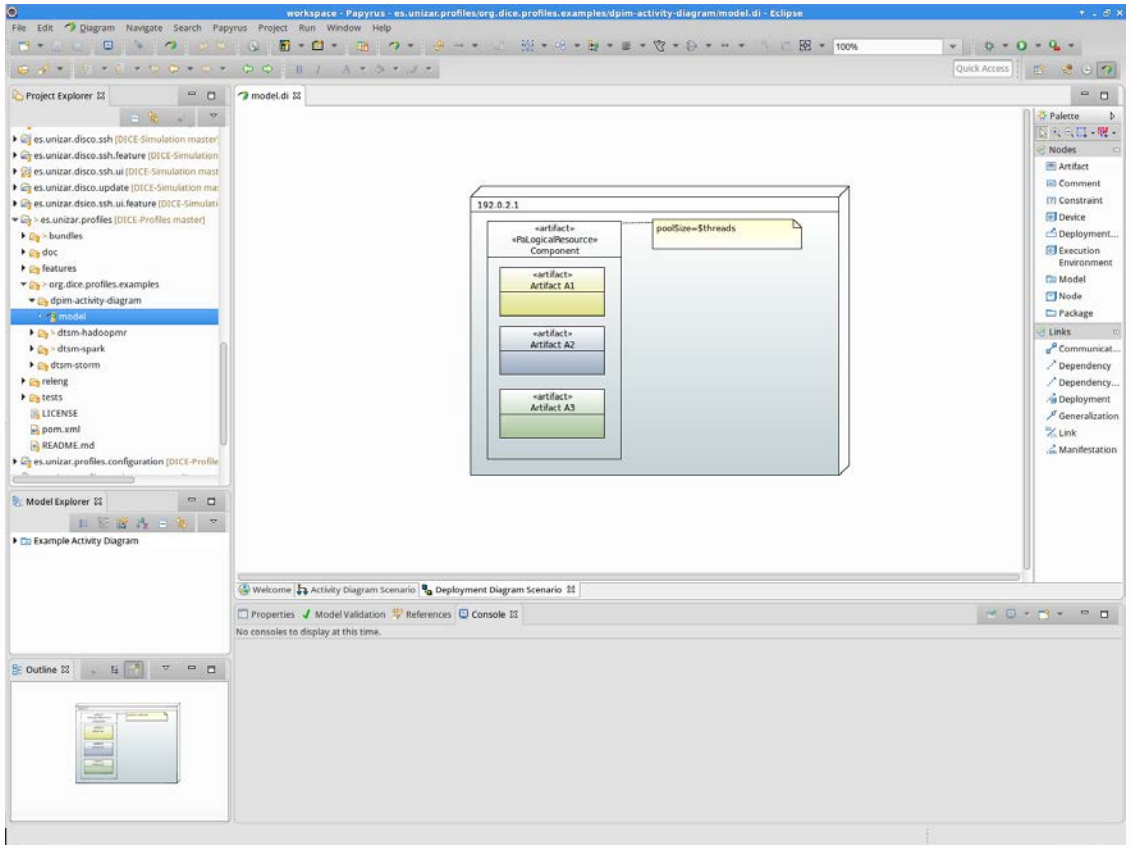
Simulation workflow (8)



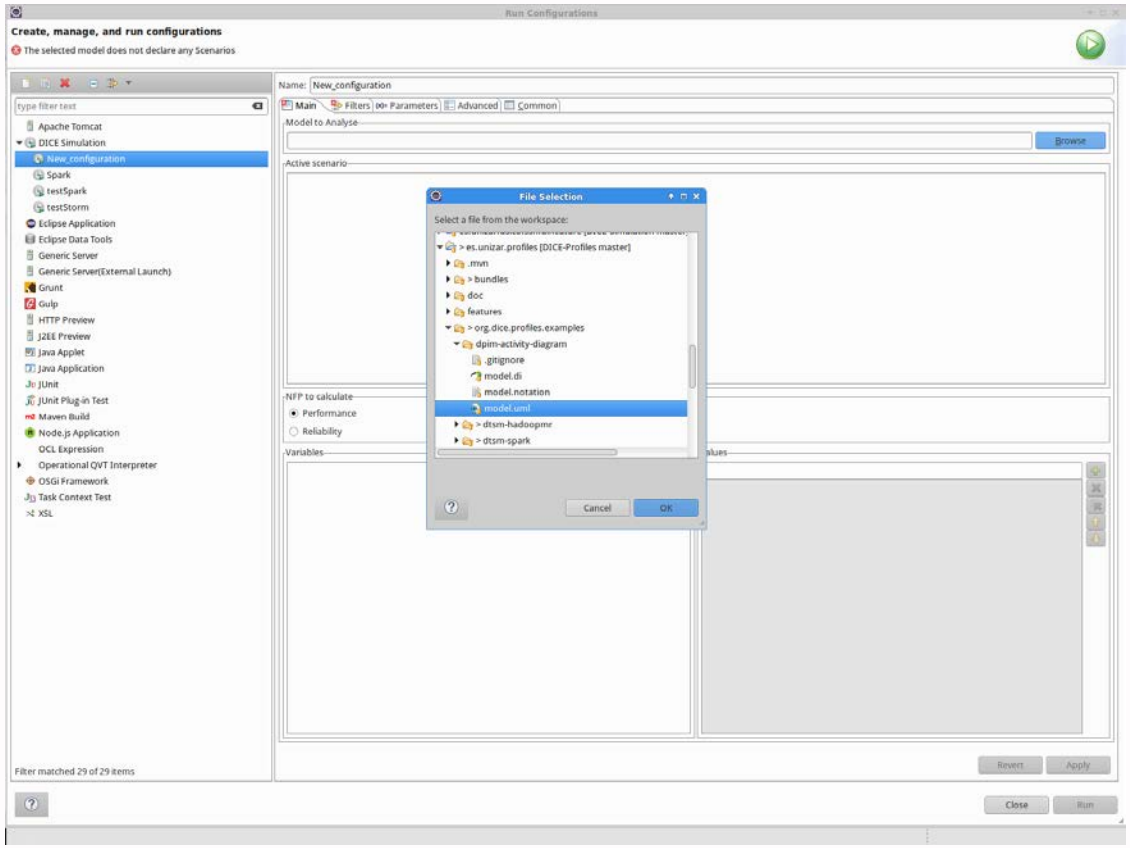
Simulation workflow (9)



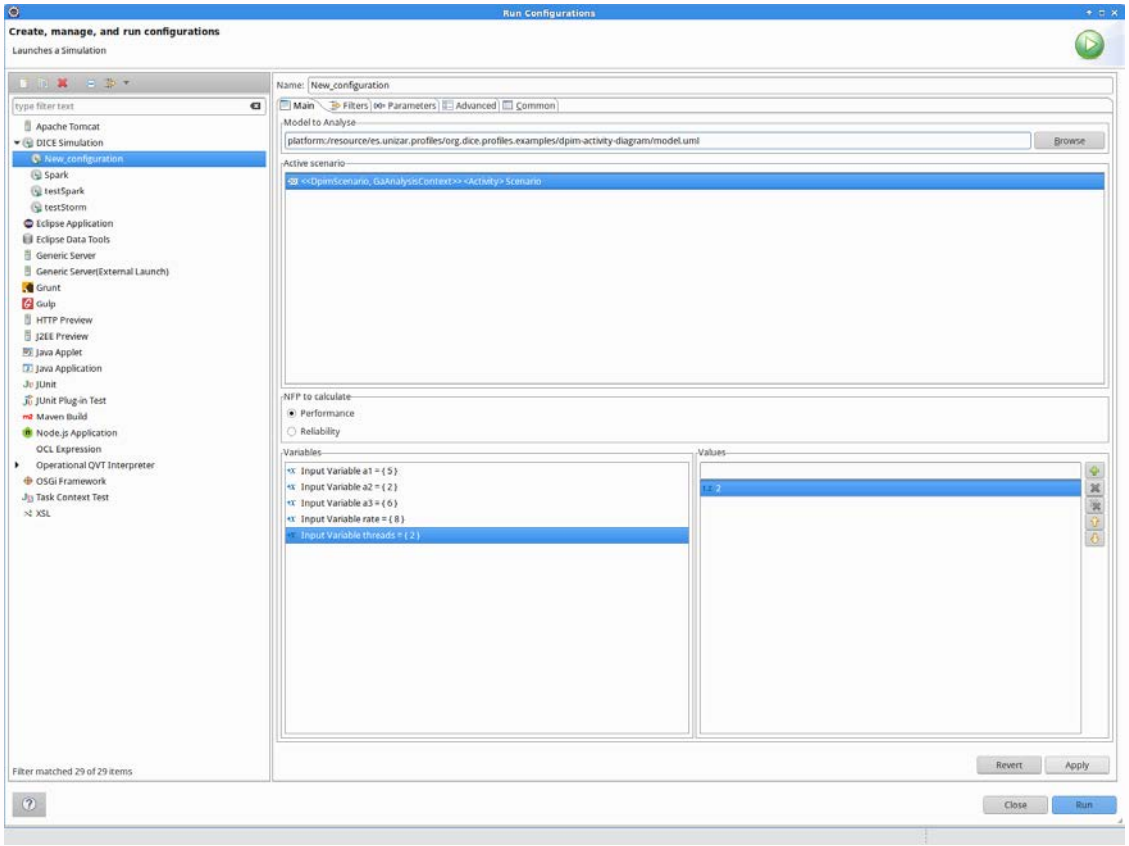
Simulation workflow (10)



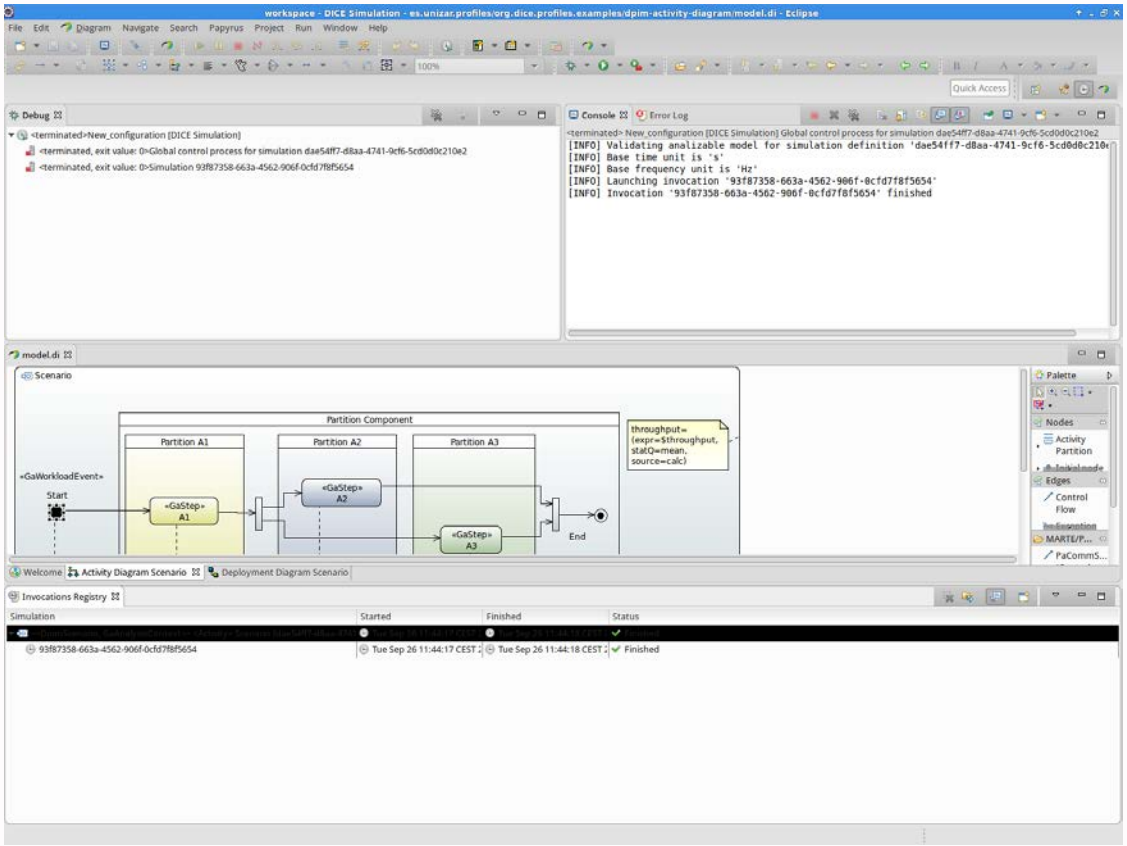
Simulation workflow (11)



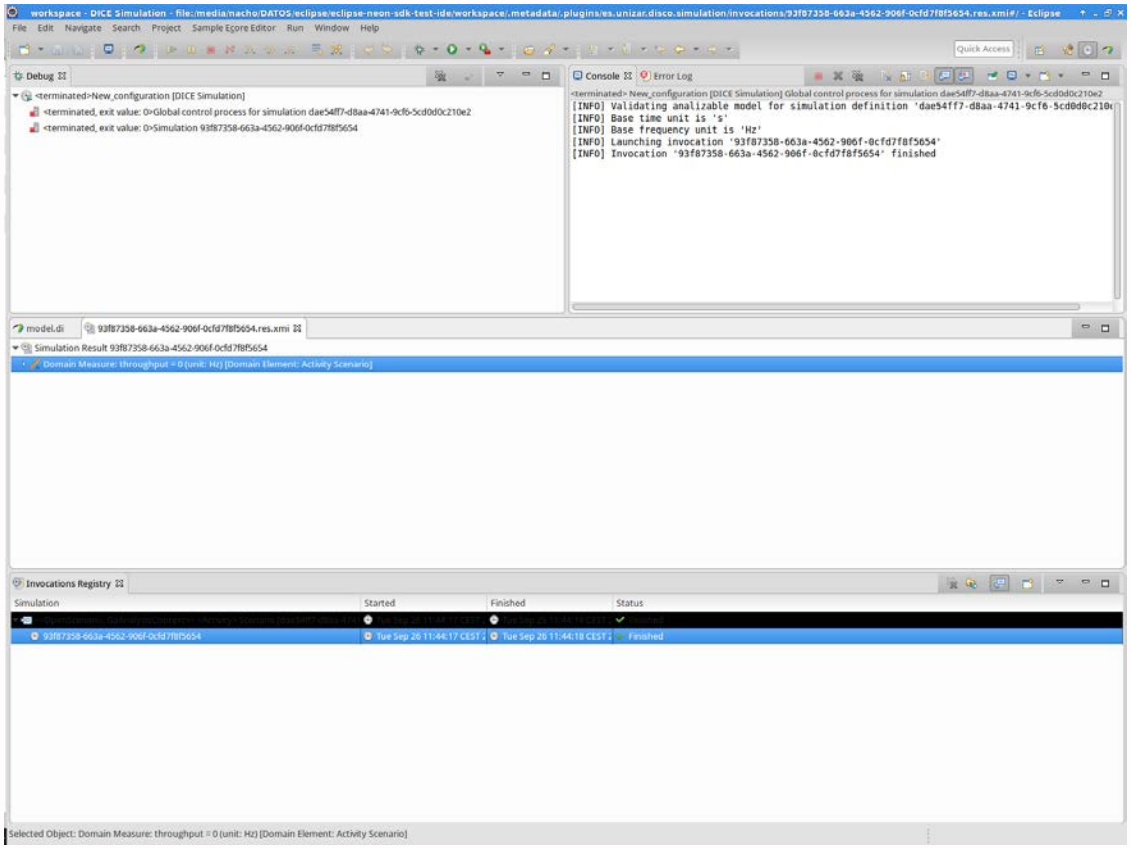
Simulation workflow (12)



Simulation workflow (13)

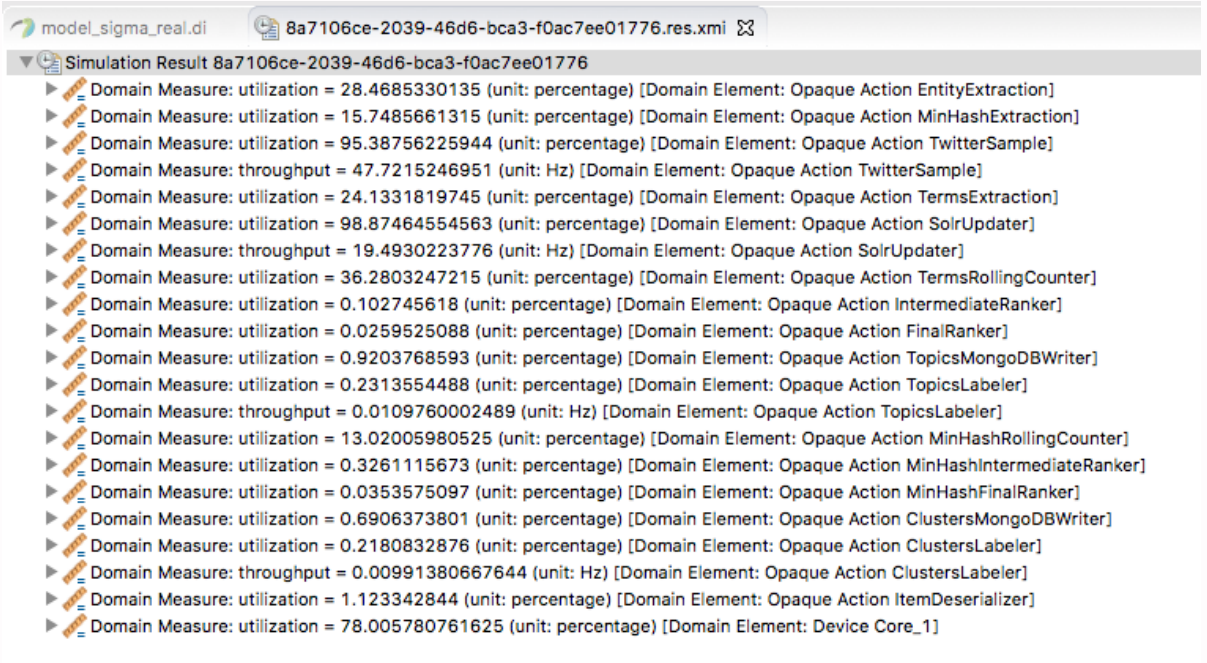


Simulation workflow (14)



Simulation Benefit

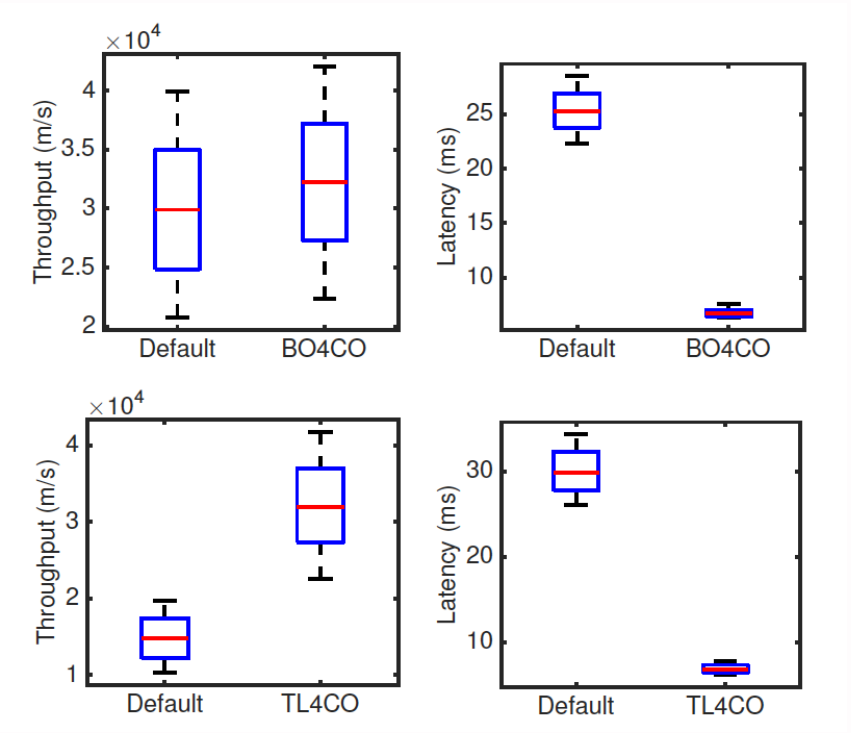
- Configure the Storm design to specific execution context
- Detect performance bottlenecks
- Less than 20% with regard to the prediction error on the utilization metric



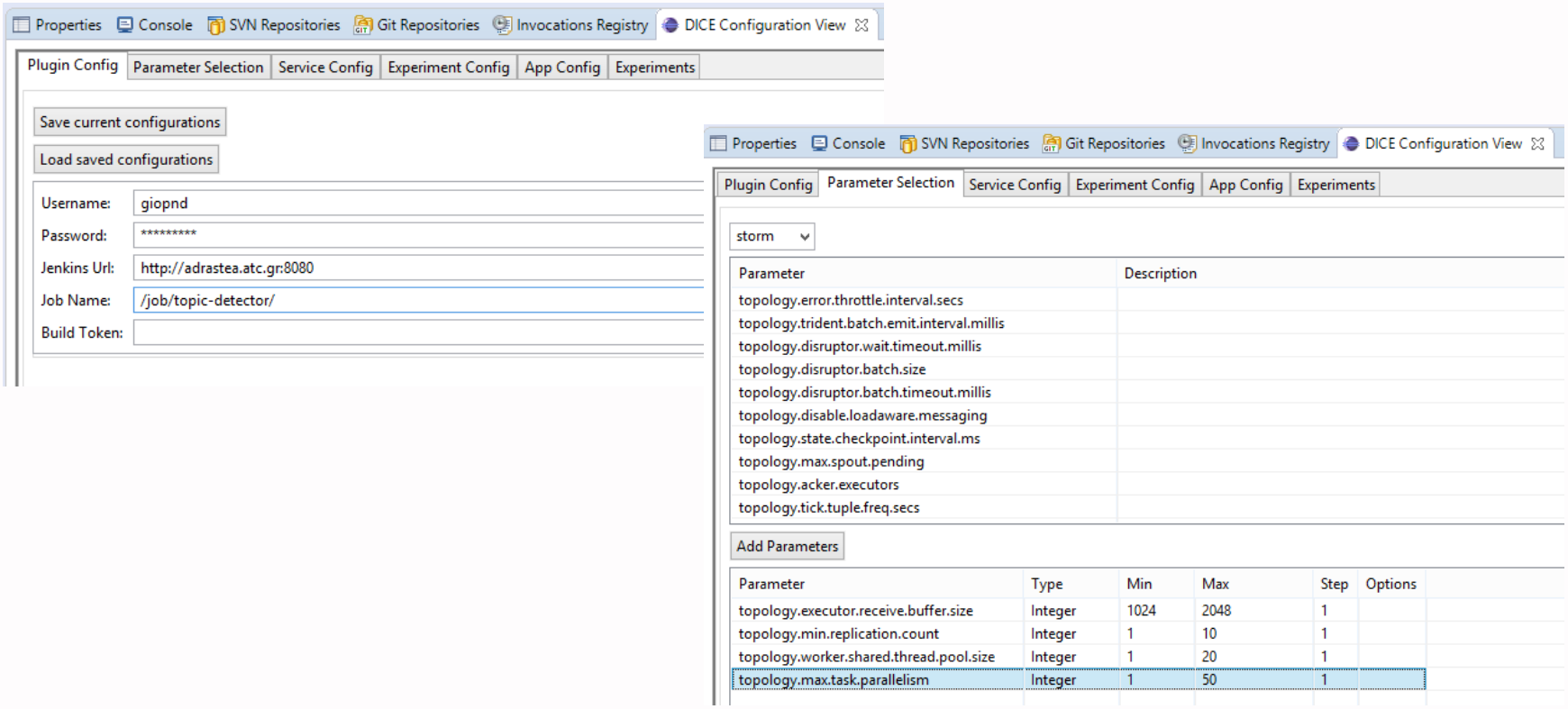
Configuration Optimization (CO)

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	10080
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

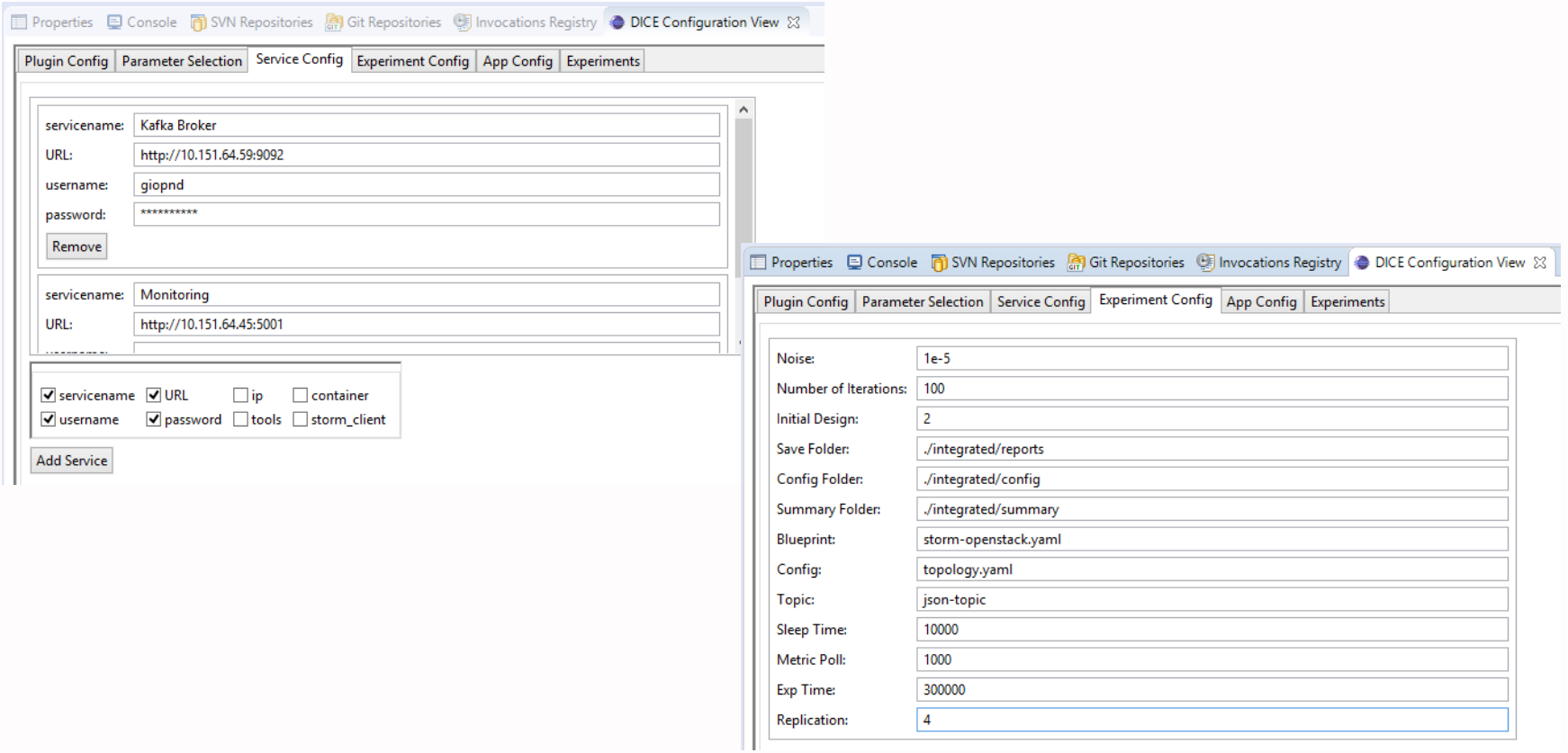
Motive: Find optimal configuration settings wrt computational budget restrictions



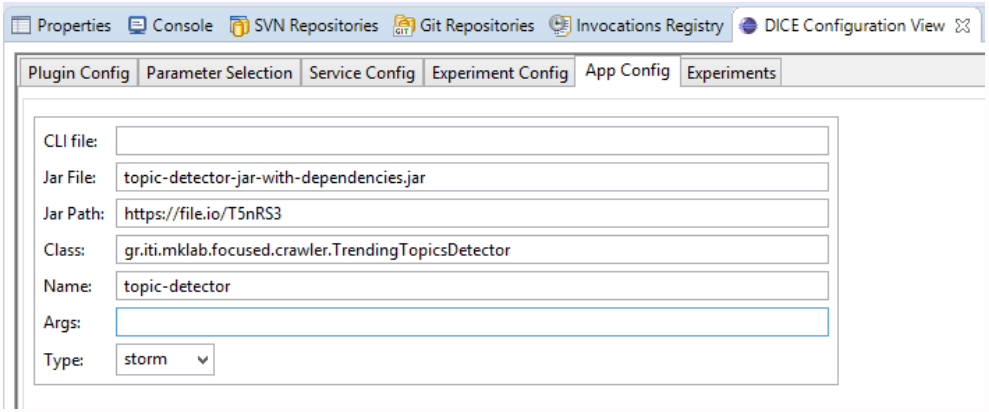
Plugin Configuration, Parameter Selection

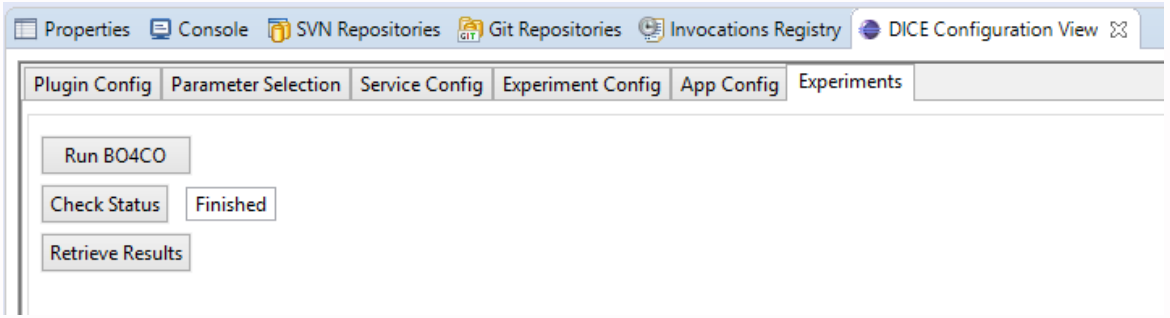


Service & Experiment Configuration



Application Configuration, Experiments





CO Benefit

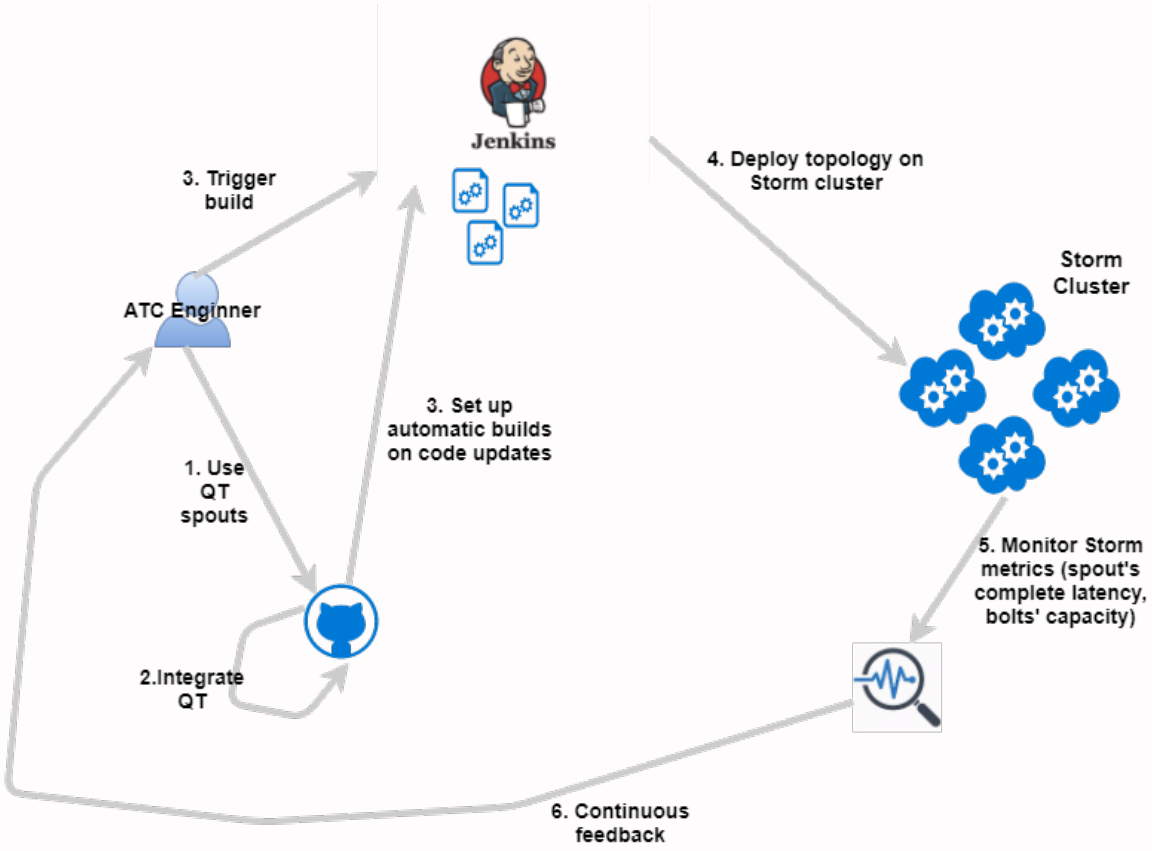
- **Throughput improvement:**
  - more than twice compared to the default configuration
  - achieved after only 100 iterations
  - took 100 * 10 min = 16h to run
- **No expert needed!!**

Quality Testing (QT)

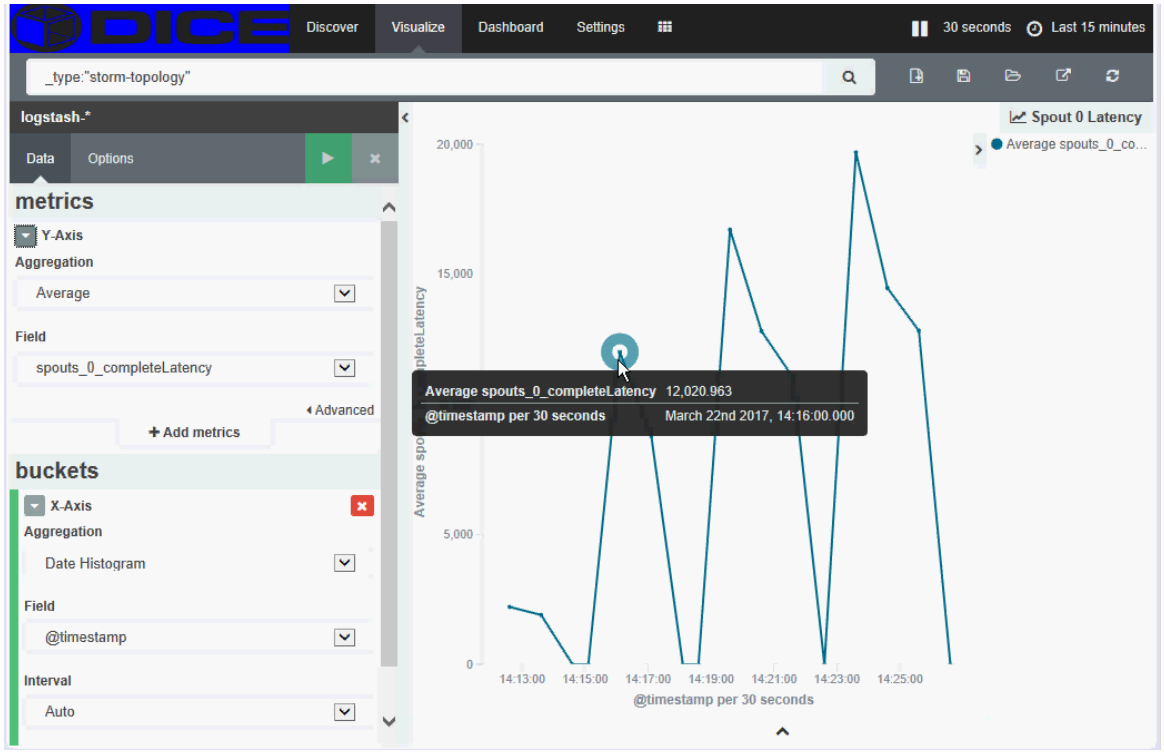
- **Motive:** Stress test the capacity of the 'trending topic detection' topology
- Run multiple iterations/experiments
  - Inject constantly increasing stream load
    - With respect to 'complete latency' metric threshold
- Evaluate the maximum input rate
  - Twitter Streaming API has limitations
 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



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!

Funded by the Horizon 2020 Framework Programme of the European Union

- Web page: http://www.dice-h2020.eu/
- Code repository: https://github.com/dice-project
- Twitter: @diceh2020 Facebook: DiceH2020

- Matej Artač: matej.artac@xlab.si @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

Acknowledgement

DICE

Horizon 2020 Research & Innovation Action

Grant Agreement no. 644869

http://www.dice-h2020.eu

DICE

Funded by the Horizon 2020
Framework Programme of the European Union