Quality assessment in DevOps:

Automated Analysis of a Tax Fraud Detection System

Diego Perez-Palacin, Youssef Ridene, José Merseguer

University of Zaragoza, Netfective Technology

Big Blu

eGov Tax Fraud Detection System
Under Development by Netfective Technology

Big Blu

eGov Tax Fraud Detection System Under Development by Netfective Technology

- Tax fraud represents a huge problem for governments.

https://ec.europa.eu/taxation_customs/fight-against-tax-fraud-tax-evasion/missing-part_en

- EU has estimated tax evasion to be of the order of 1 trillion euros

Big Blu

Big Blu is developed following Agile and DevOps principles

- Follow an iterative process with incremental iterations pursuing
  - Quick design
  - Quick delivery of enhancements
  - Quick feedback

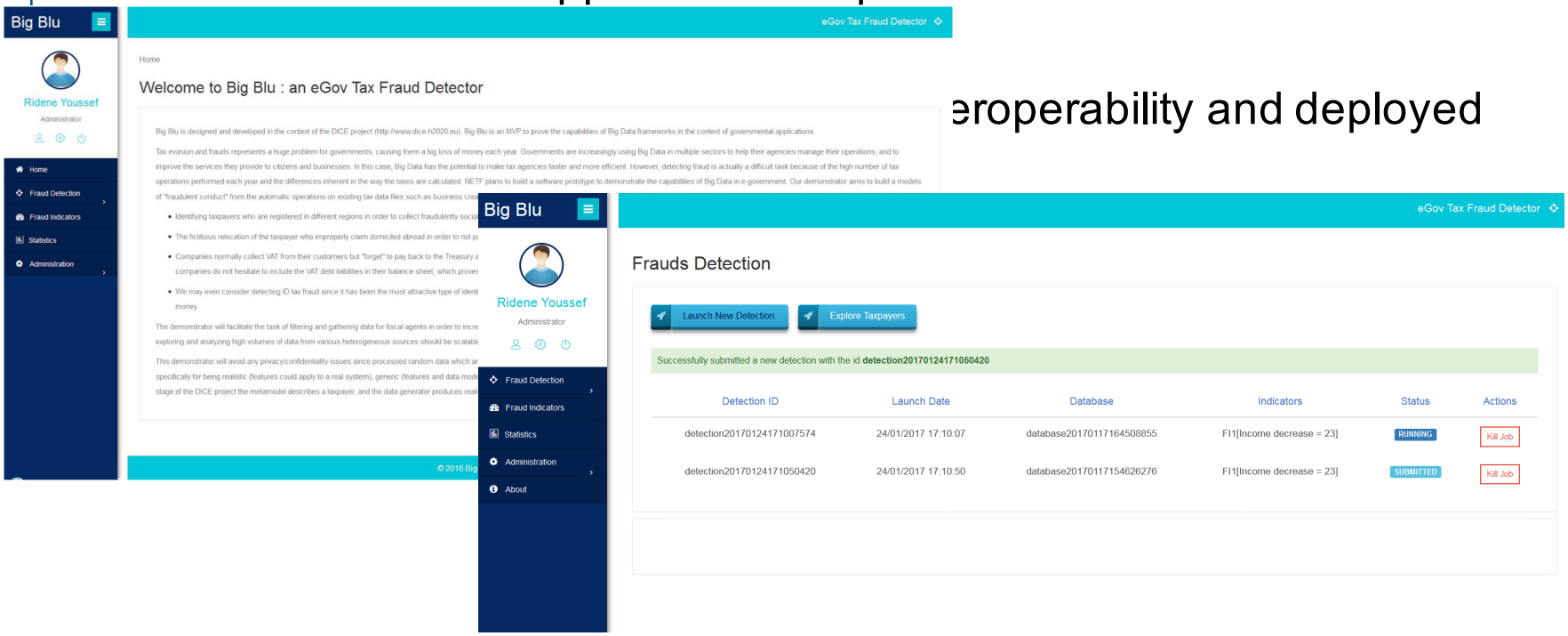
- Bring closer Development and Operations activities to improve the effectiveness of each incremental iteration
  - Achieve faster iterations
  - Achieve higher proportion of iterations with satisfactory results

Big Blu

- Software Architecture composed of 3 main layers:
  - GUI: web based application. Unique interface
  - Web Services: implement RESTful interoperability and deployed on Tomcat
  - Back-end: Data processing elements

Big Blu

- Software Architecture composed of 3 main layers:
  - GUI: web based application. Unique interface



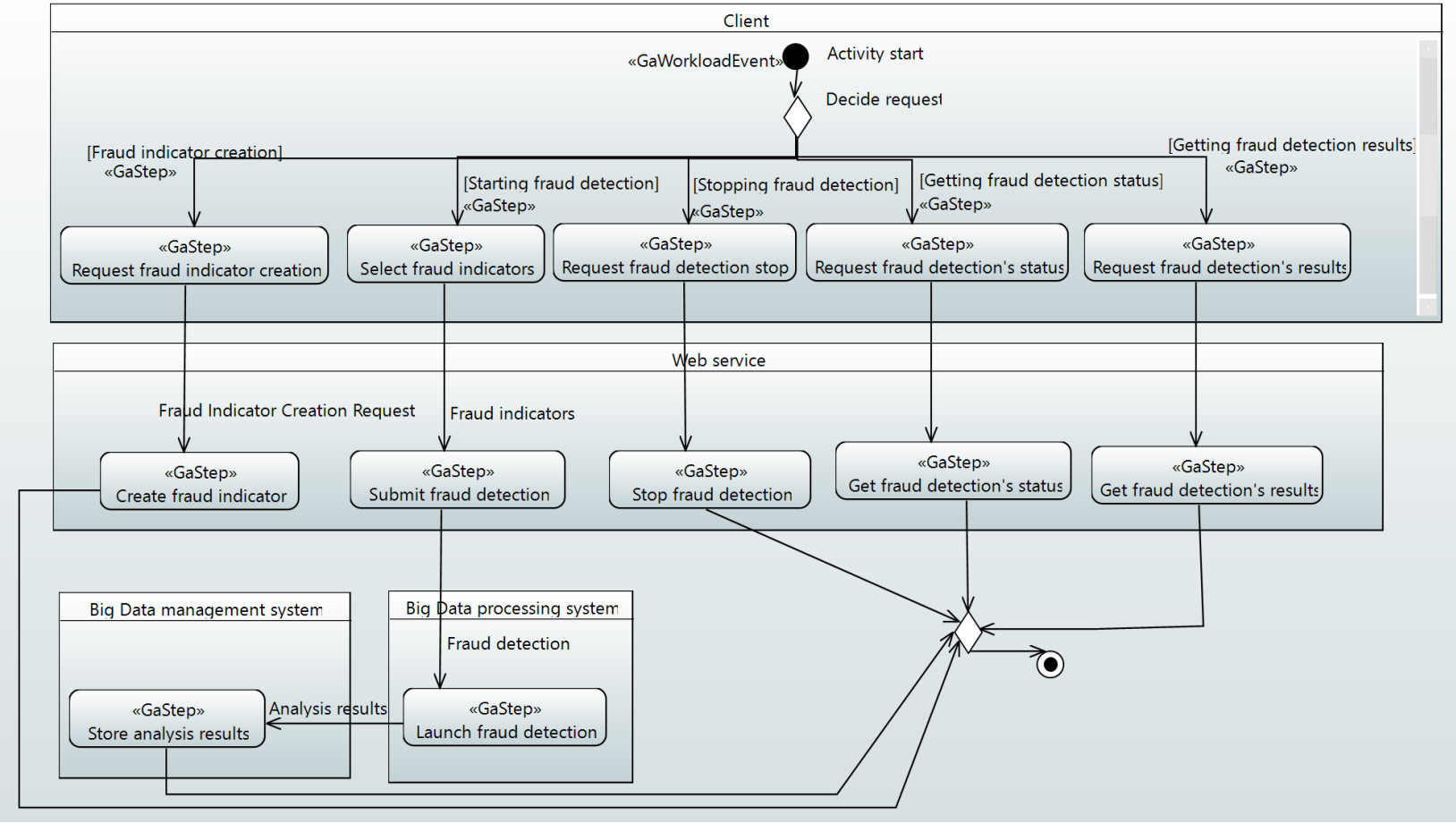
eroperability and deployed

Big Blu

- Software Architecture composed of 3 main layers:
  - GUI: web based application. Unique interface
  - Web Services: implement RESTful interoperability and deployed on Tomcat
  - Back-end: Data processing elements

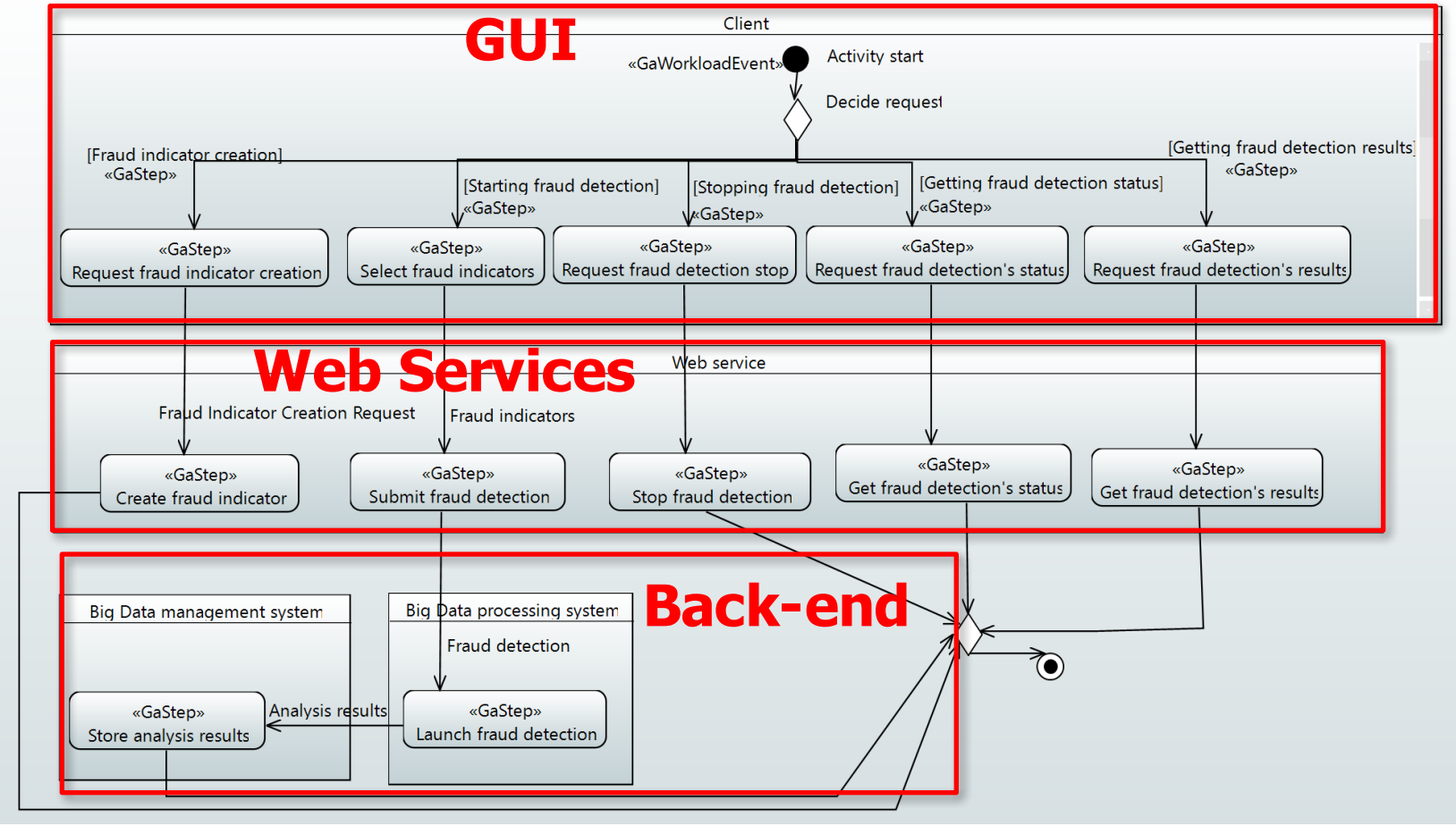
Big Blu

- Software Architecture composed of 3 main layers:



Big Blu

- Software Architecture composed of 3 main layers:



DICE approach

Researches towards building a quality-driven framework for development, deployment, monitoring and continuous improvement of Data-Intensive Cloud Applications.

- Pursues developments with Iterative Quality enhancements
- Delivers a toolchain for:
  - Design
  - Quality analysis
  - Deployment
  - Testing
  - Monitoring (collect data, visualization, anomaly detection, trace checking)
  - Enhancement

UNIFIED MODELING LANGUAGE + PROFILING

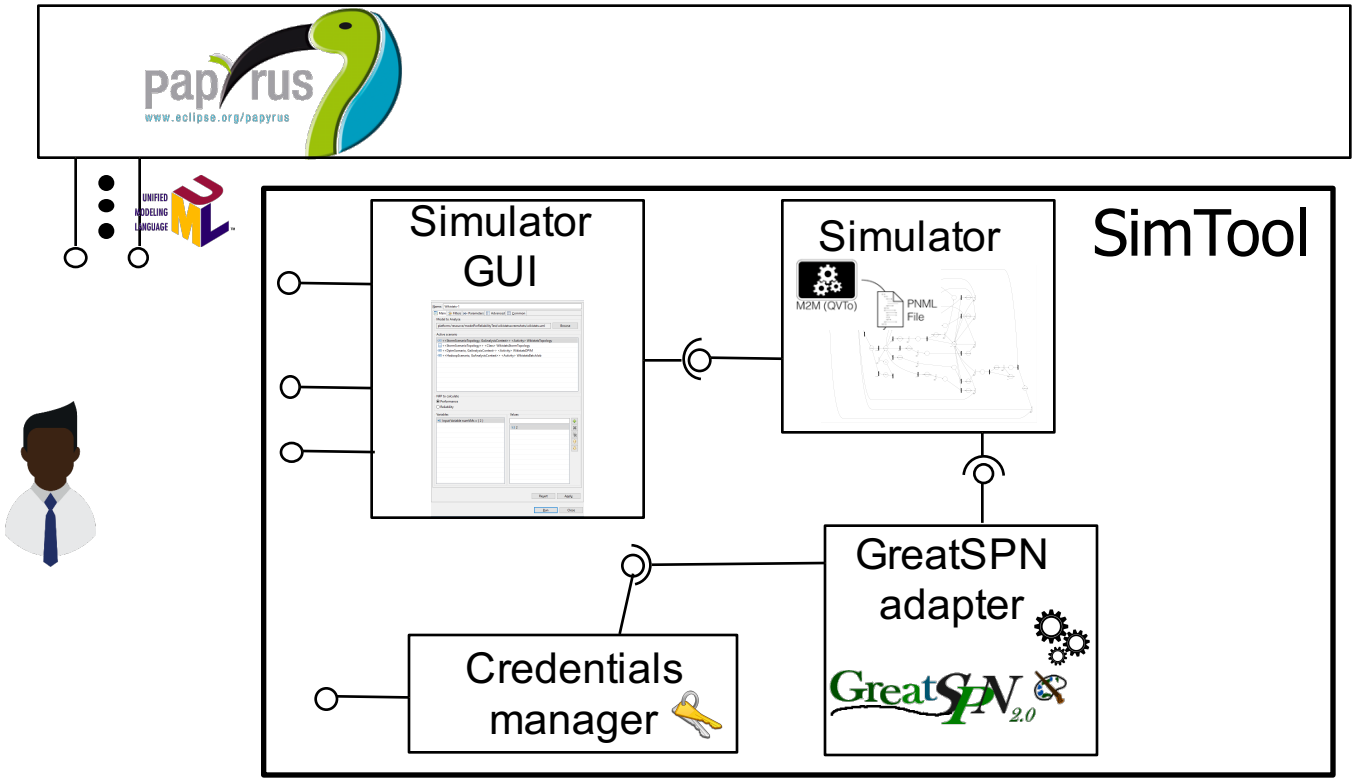
DICE approach

Researches towards building a quality-driven framework for development, deployment, monitoring and continuous improvement of Data-Intensive Cloud Applications.

- Pursues developments with Iterative Quality enhancements
- Delivers a toolchain for:
  - Design
  - **Quality analysis**
  - Deployment
  - Testing
  - Monitoring (collect data, visualization, anomaly detection, trace checking)
  - Enhancement

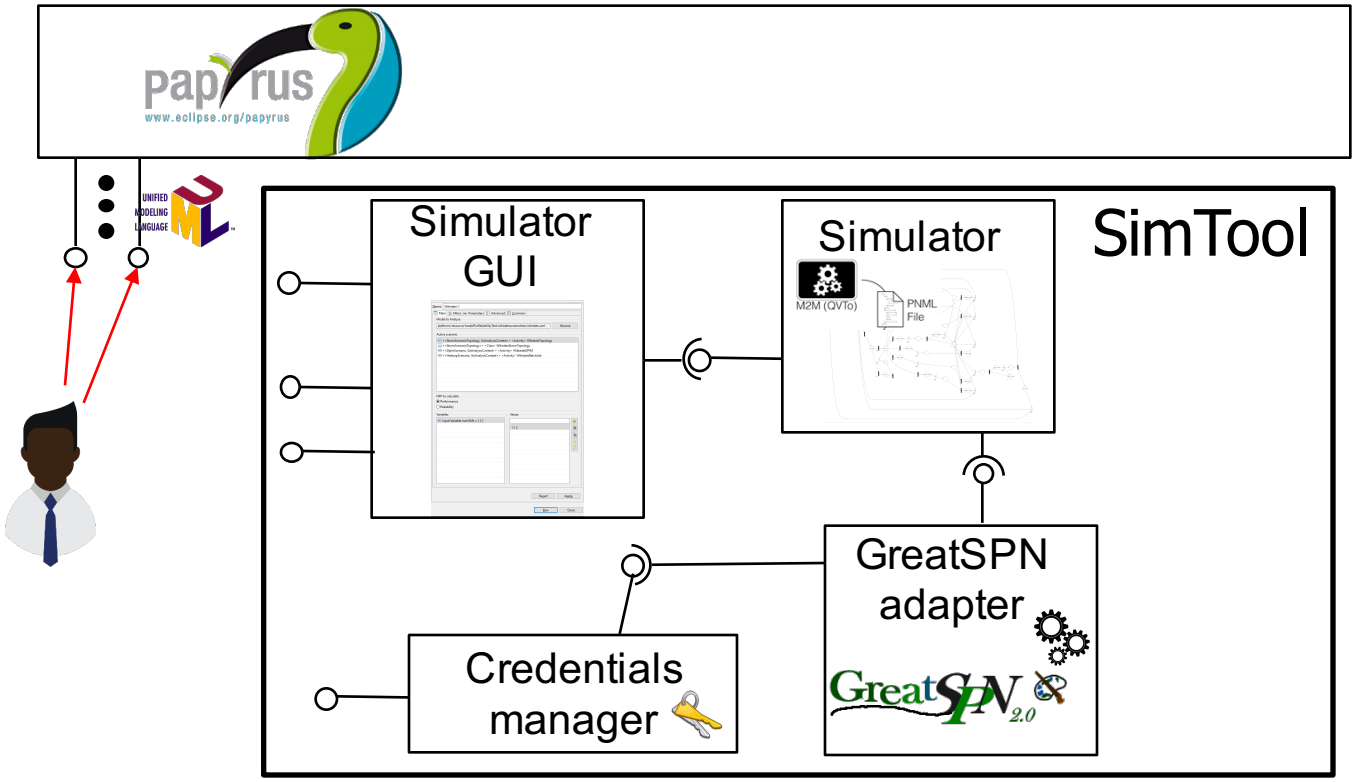
DICE Simulation Tool

- Based on eclipse plugins eclipse Delivered with DICE



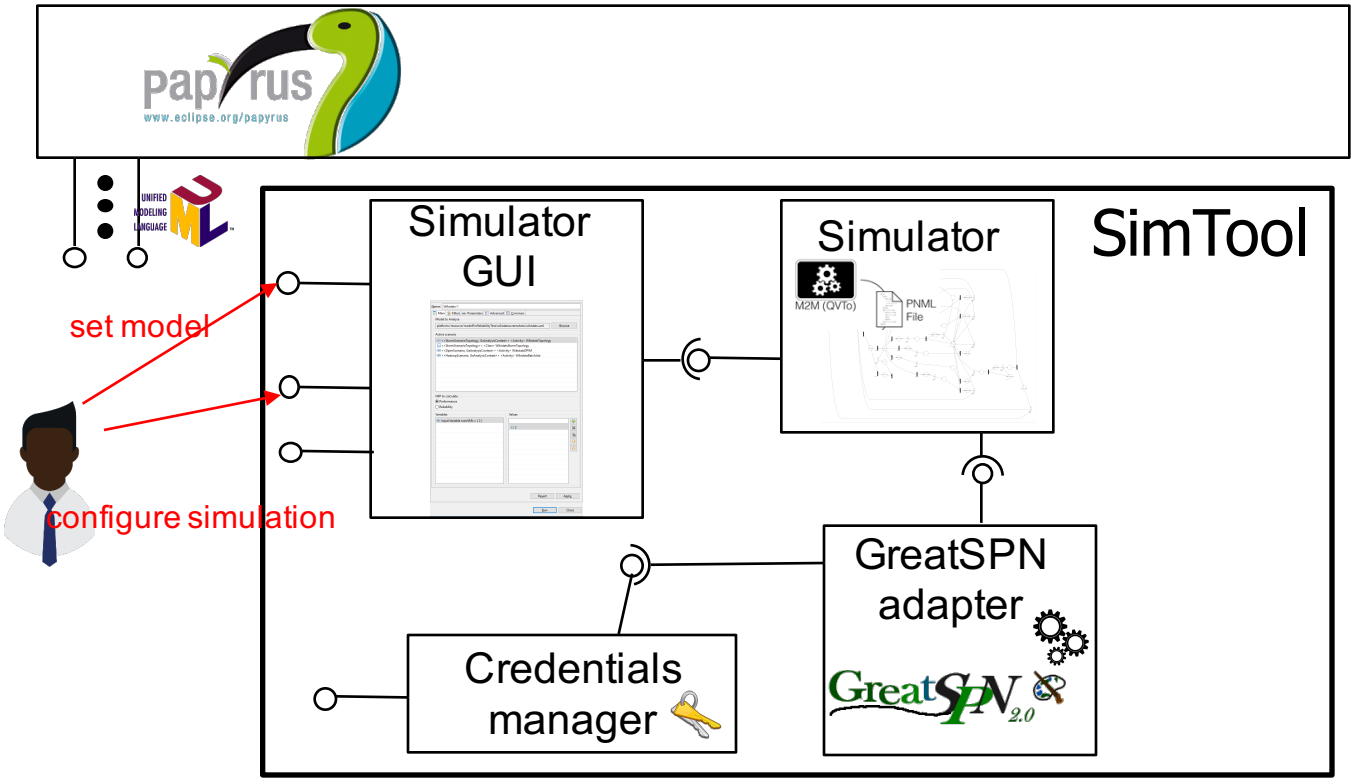
DICE Simulation Tool

- Based on eclipse plugins eclipse Delivered with DICE



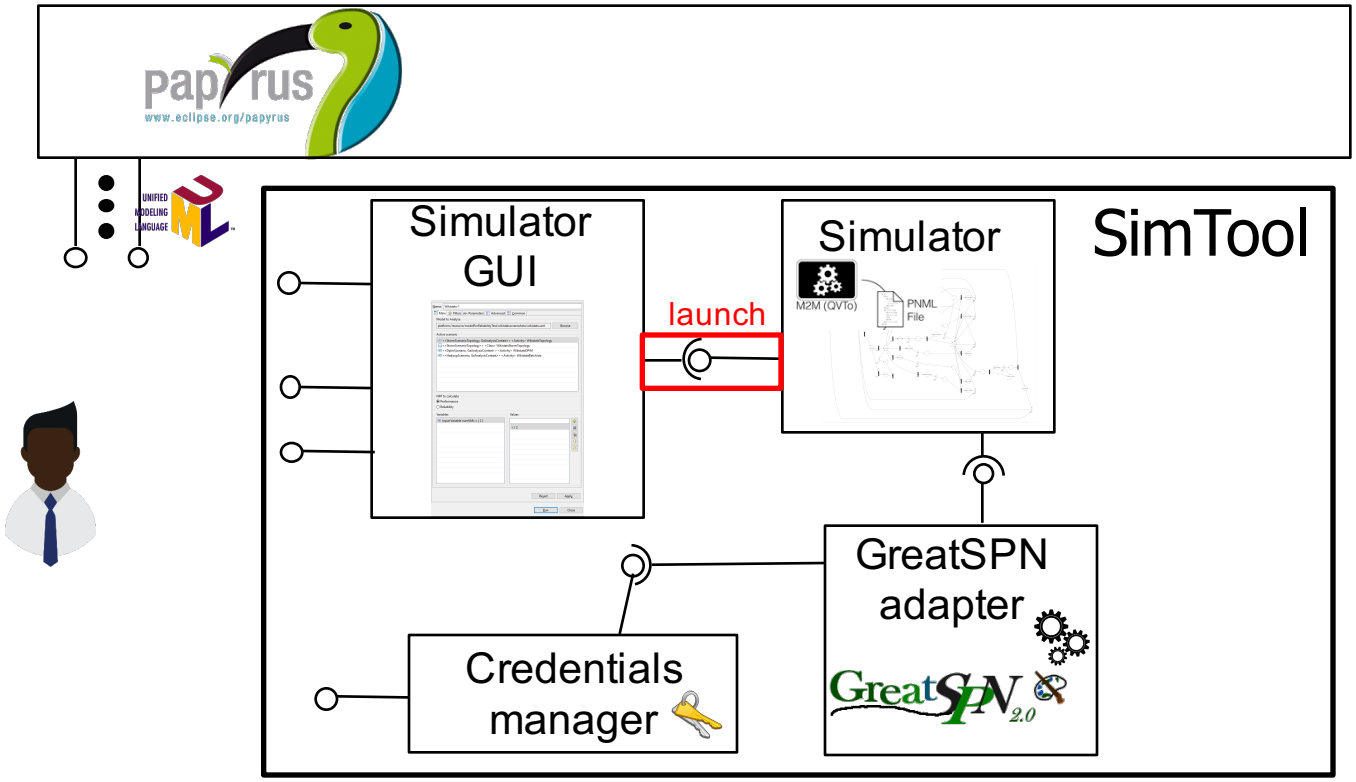
DICE Simulation Tool

- Based on eclipse plugins  Delivered with 



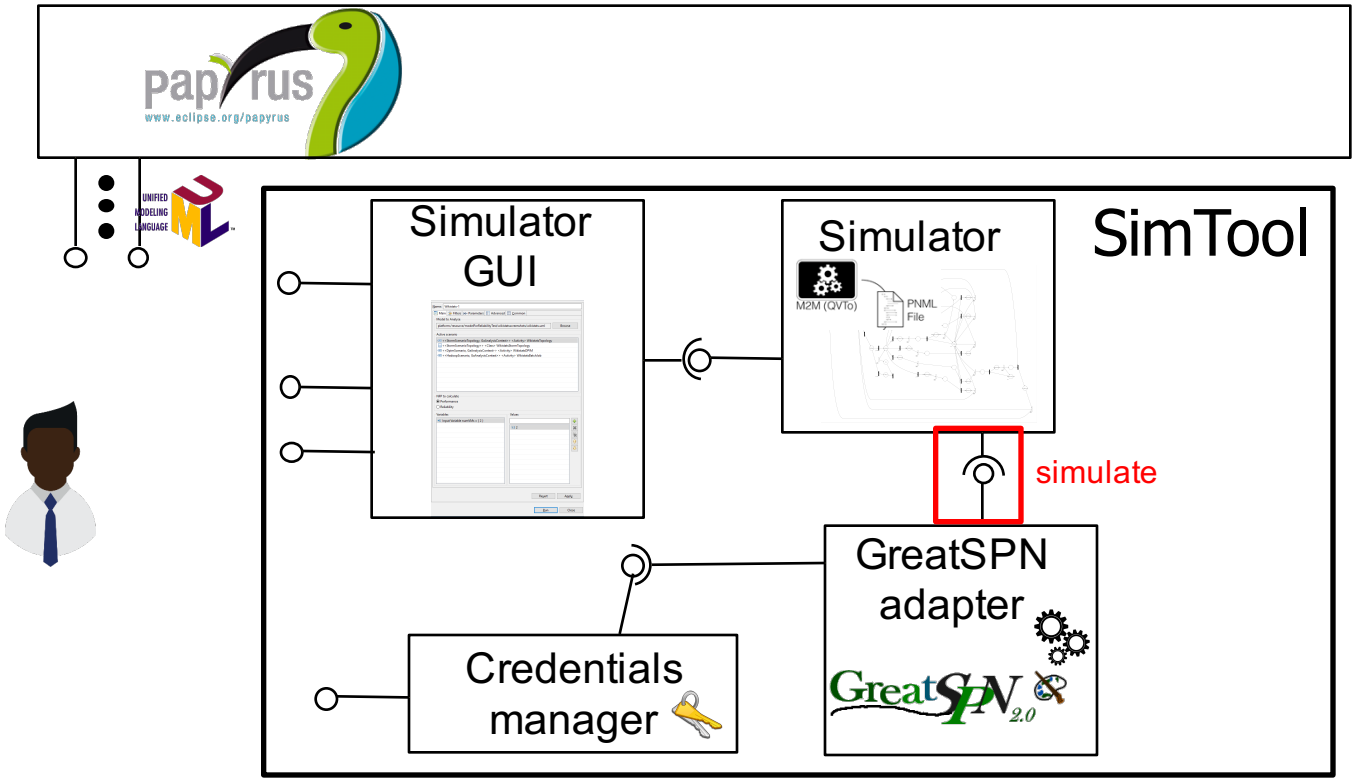
DICE Simulation Tool

- Based on eclipse plugins eclipse Delivered with DICE



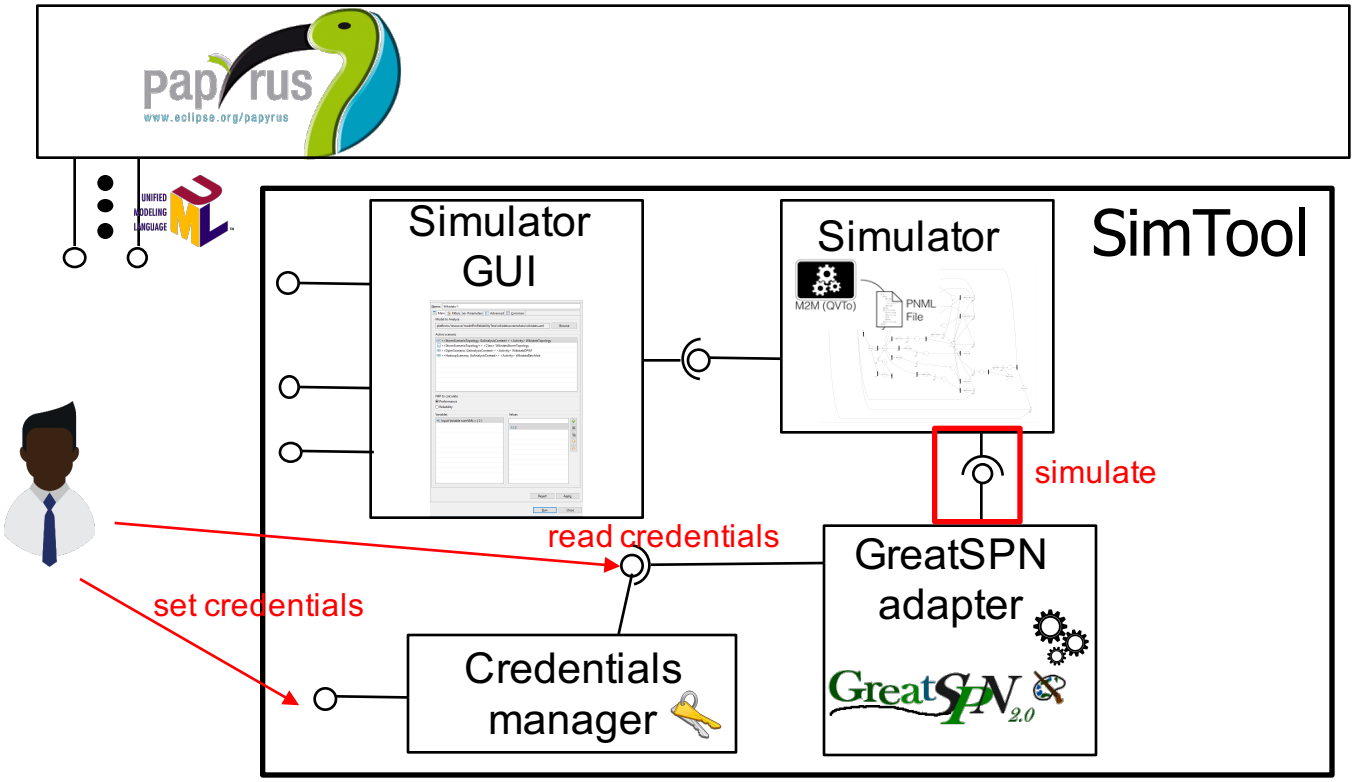
DICE Simulation Tool

- Based on eclipse plugins eclipse Delivered with DICE



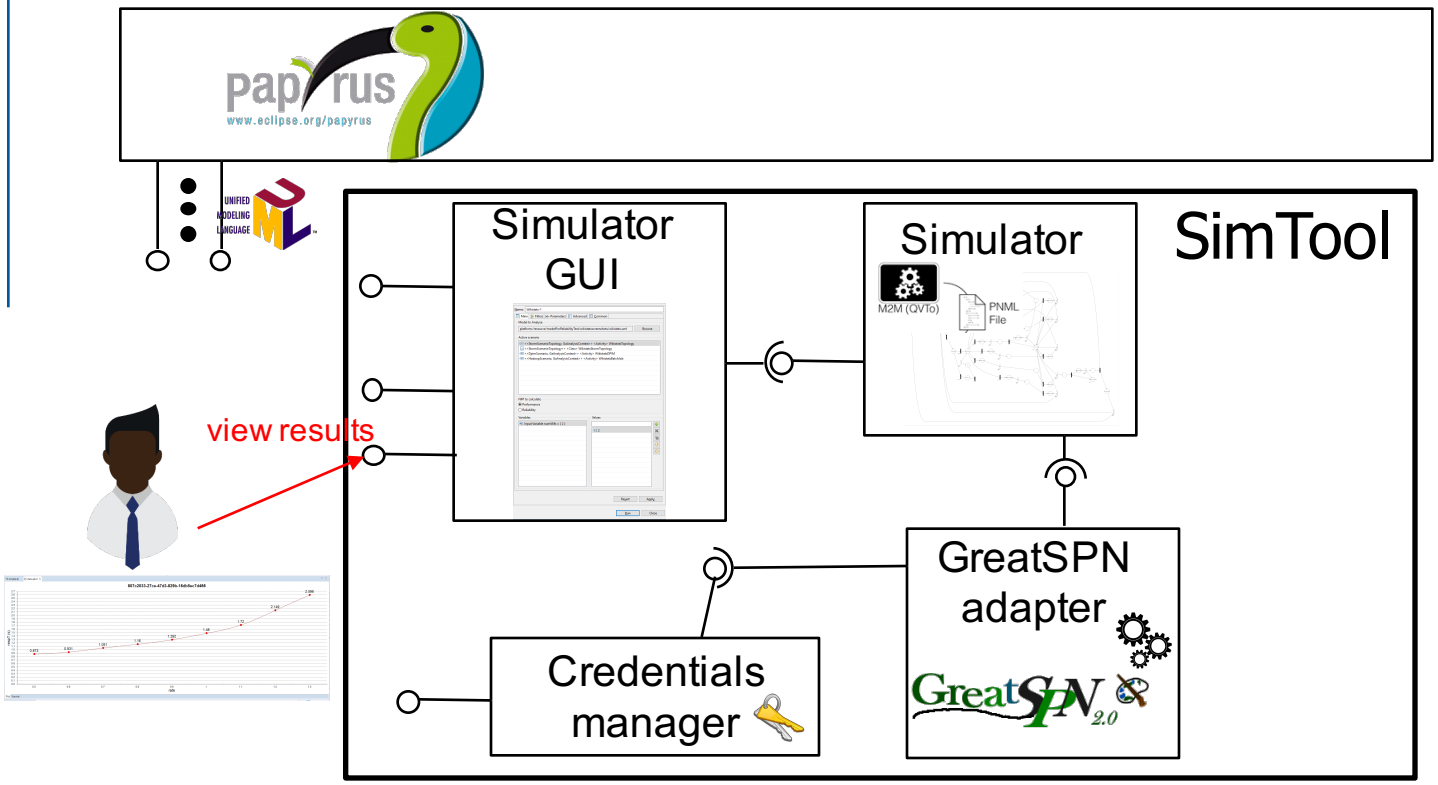
DICE Simulation Tool

- Based on eclipse plugins  Delivered with 



DICE Simulation Tool

- Based on eclipse plugins eclipse Delivered with DICE



DICE Simulation Tool

Usefulness in Agile cycles following DevOps

- Scenario 1: Development of new functionalities

PROBLEM

- In agile cycles, the required quality of the new functionalities may not be clear for developers
  - The quality requirements refer to the overall system quality

CONSEQUENCES

- Obtained quality of the new functionality is not good enough and the cycle has to be repeated

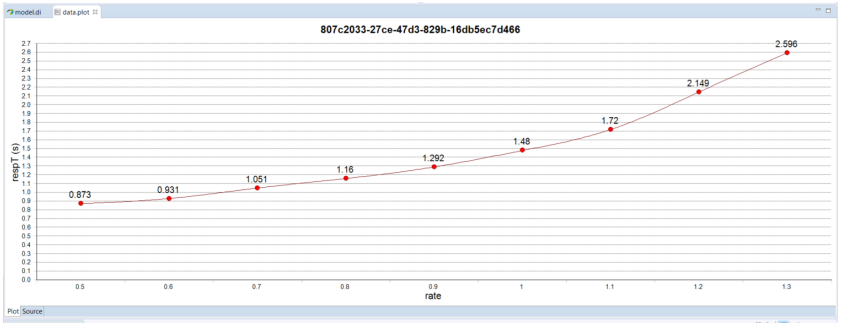
DICE Simulation Tool

Usefulness in Agile cycles following DevOps

- Scenario 1: Development of new functionalities

APPROACH TO SOLUTION

- Obtain values for ``appropriate quality'' of the new functionality that can be already asserted during the unit tests



- Developers deliver a functionality that passes these unit tests a go to next phases of the cycle with some confidence about the quality

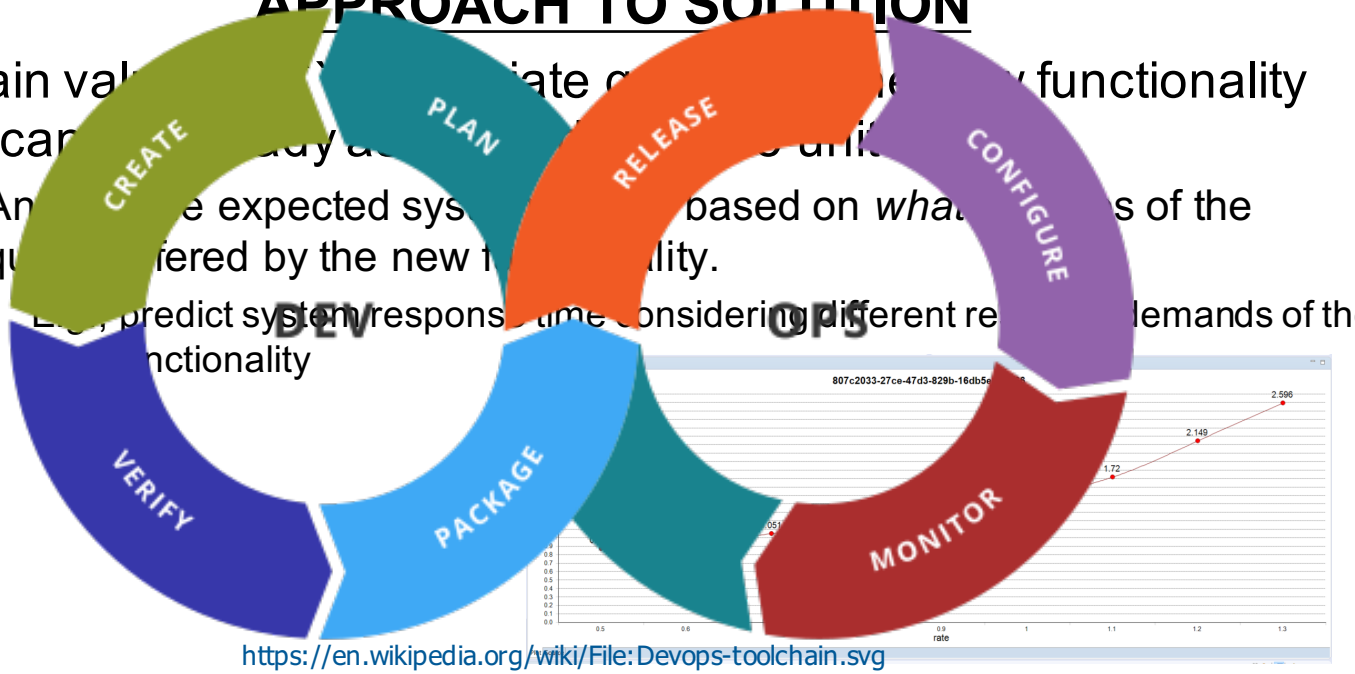
DICE Simulation Tool

Usefulness in Agile cycles following DevOps

- Scenario 1: Development of new functionalities

APPROACH TO SOLUTION

- Obtain val… …iate q… …e… …y functionality that can… …dy a… …s unit…
  - An… …e expected sys… …based on *what*… …s of the qu… …fered by the new f… …lity.
    - E.g., predict system respons… time considering different re… demands of the …nctionality
- Deliver a functionality that passes these unit tests



https://en.wikipedia.org/wiki/File:Devops-toolchain.svg

DICE Simulation Tool

Usefulness in Agile cycles following DevOps

- Scenario 2: Maintenance of functionalities

PROBLEM

- Quality of a functionality has to be improved…
  - Due to changes in the utilization of the application
  - Due to new quality restrictions and improvable designs
- …and can be improved in different phases of DevOps toolchain

CONSEQUENCES

- Maintenance may not achieve the expected quality
- Modifications result more expensive than necessary

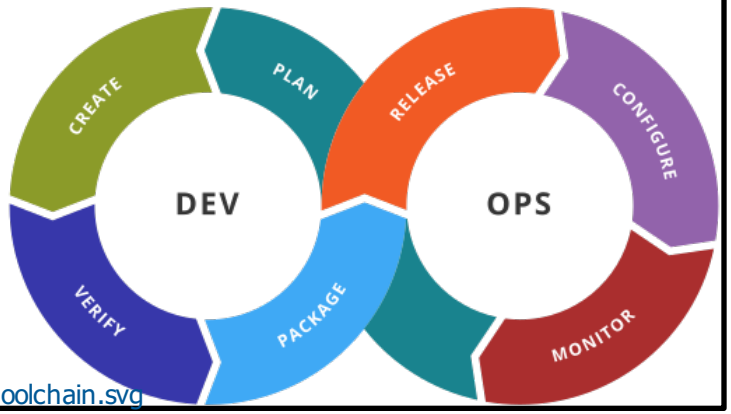
DICE Simulation Tool

Usefulness in Agile cycles following DevOps

- Scenario 2: Maintenance of functionalities

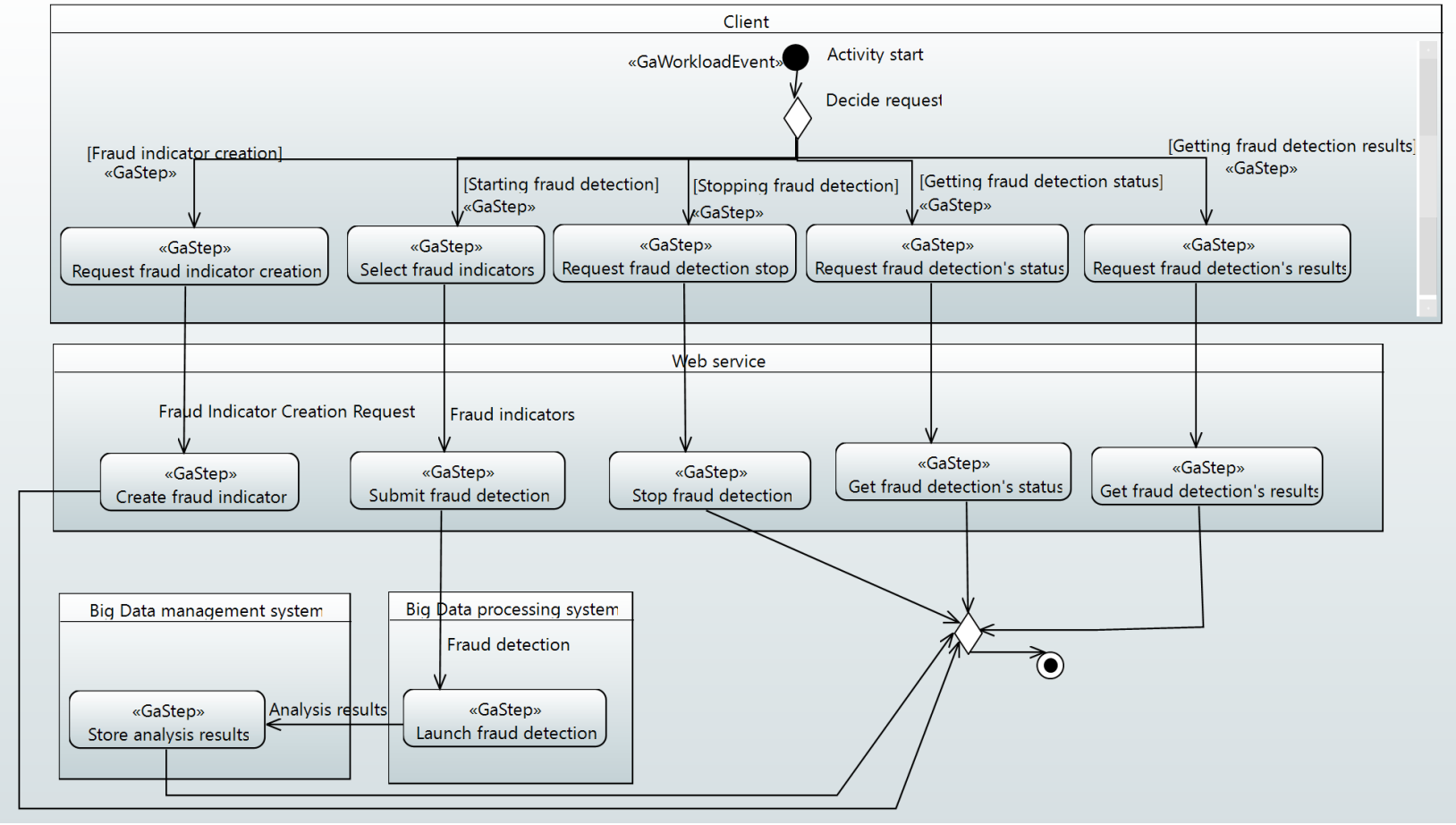
APPROACH TO SOLUTION

- Update the models with recent monitored data
- Identify quality issues
- Evaluate the alternatives to solve the issues
- Decide for the maintenance that seems the ”smartest” action

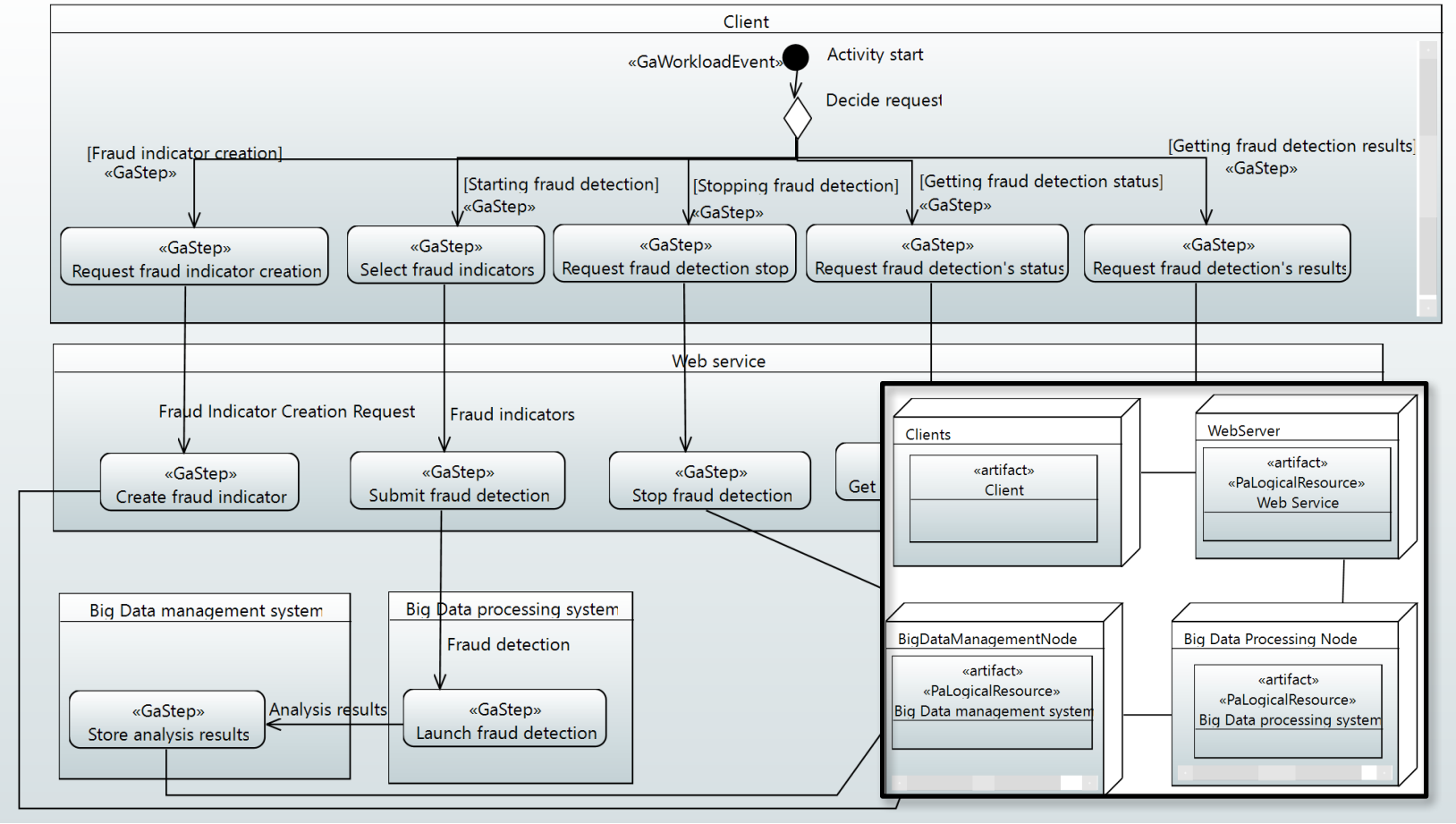


https://en.wikipedia.org/wiki/File:Devops-toolchain.svg

Big Blu Quality assessment

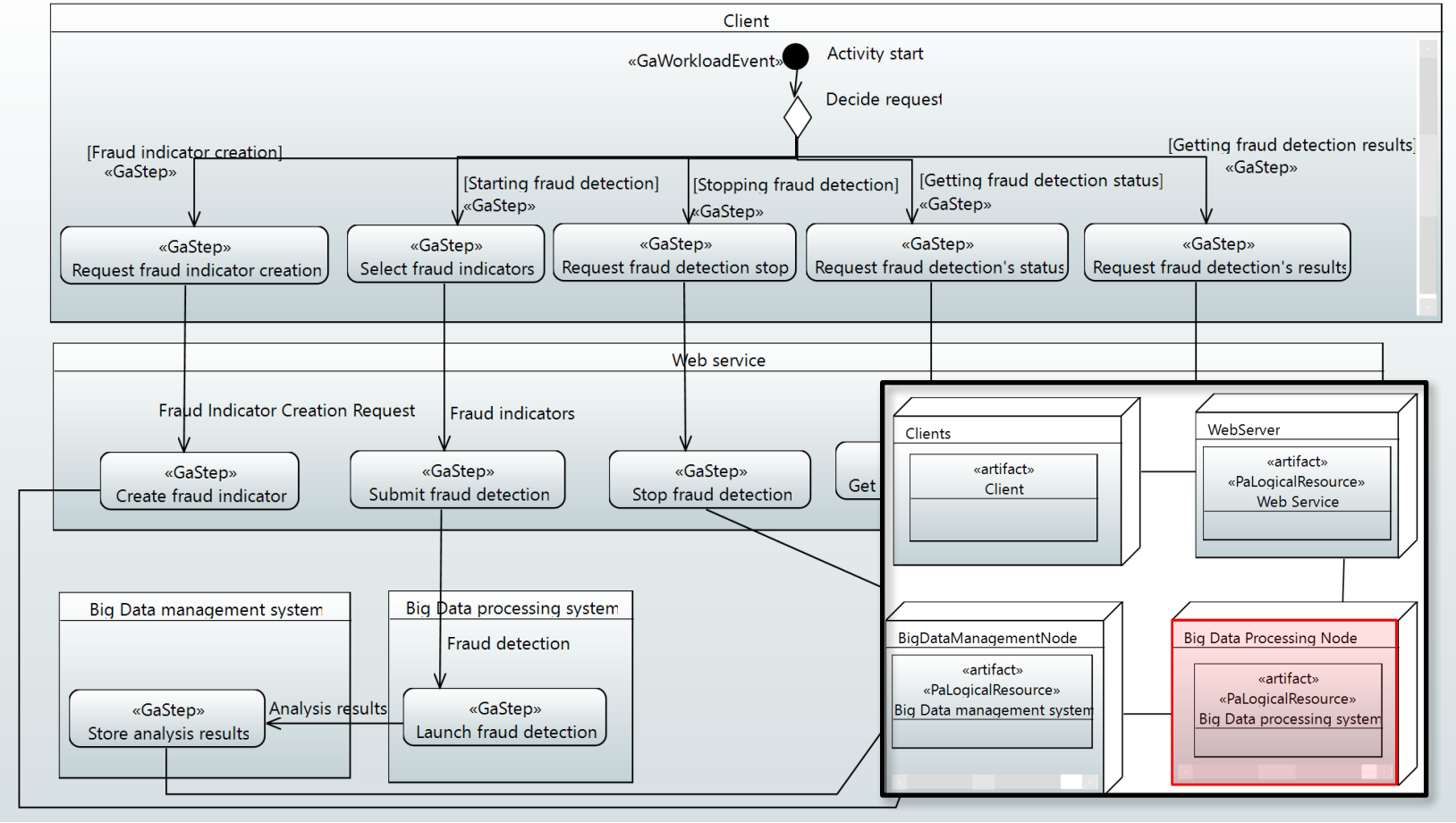


Big Blu Quality assessment



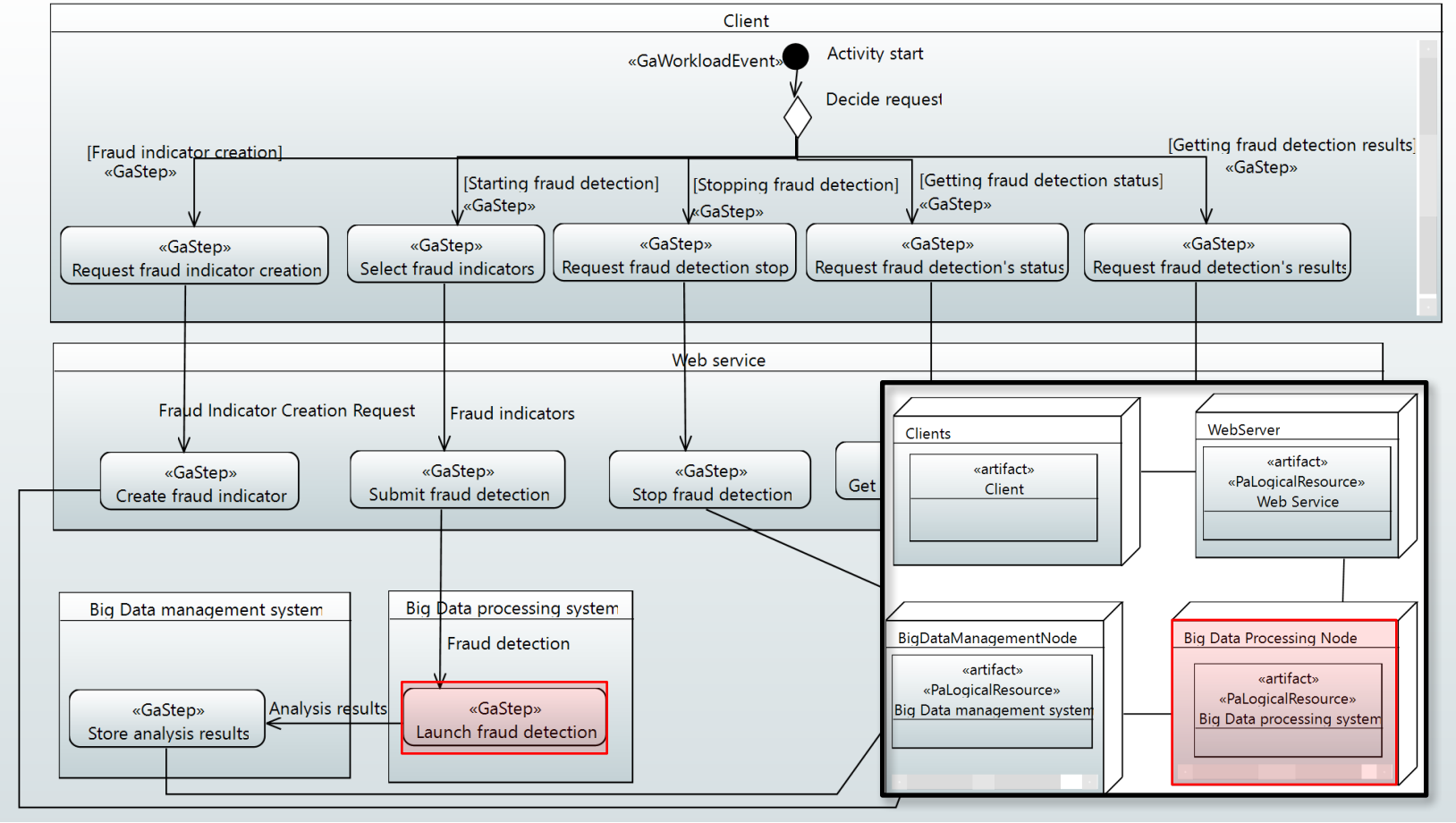
Big Blu Quality assessment

- Quality malfunction reported → maintenance



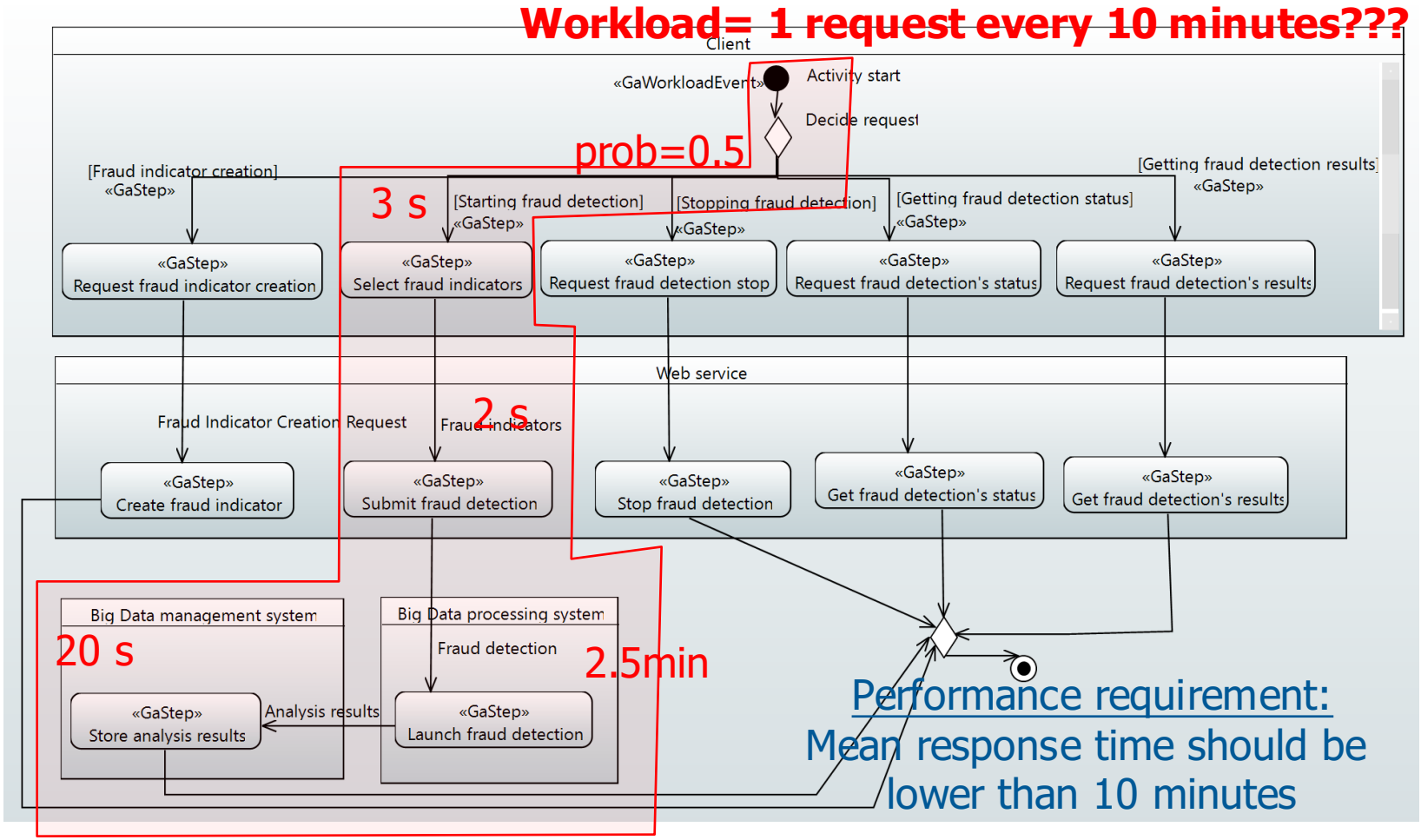
Big Blu Quality assessment

- Quality malfunction reported → maintenance



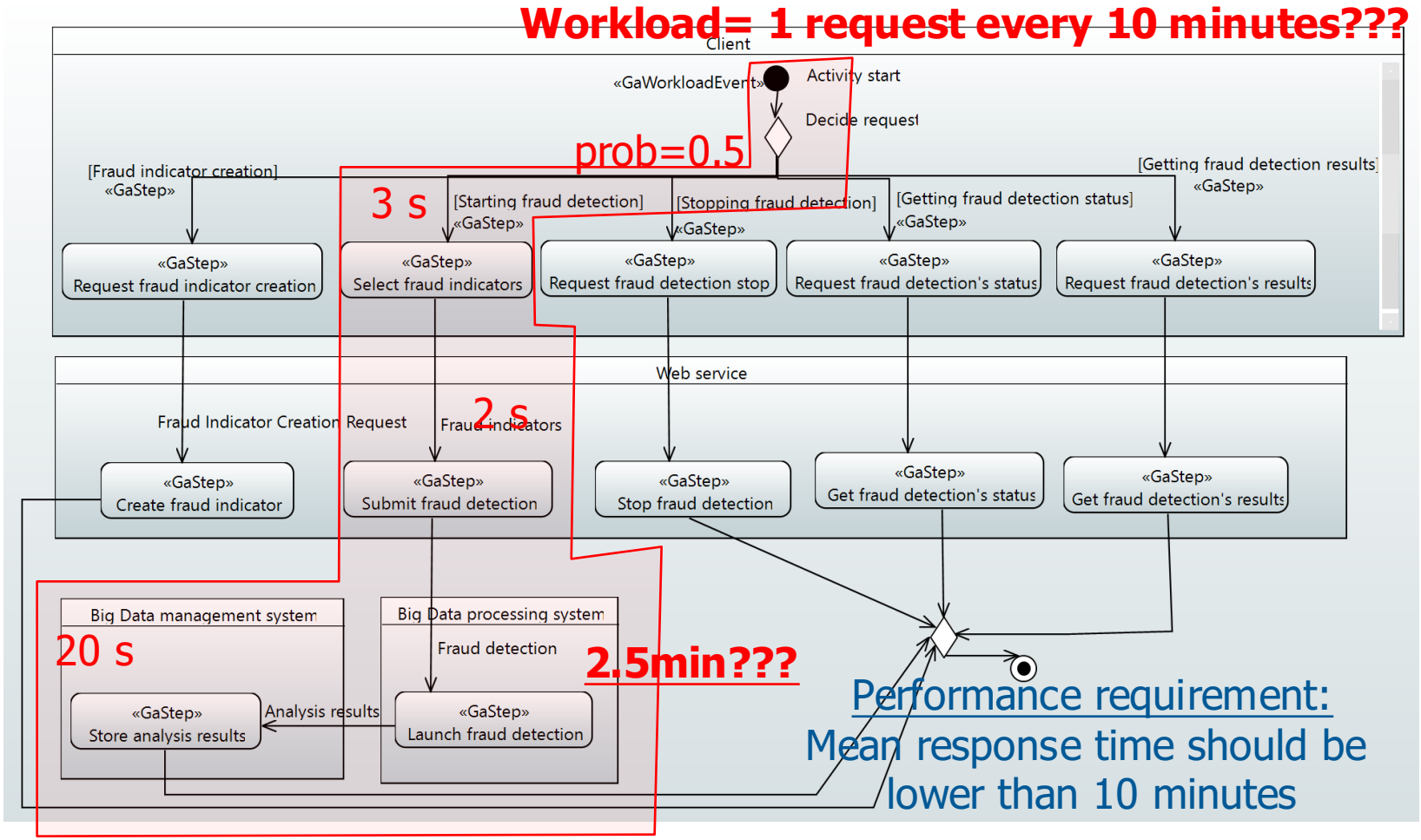
Big Blu Quality assessment

- Quality malfunction reported → maintenance



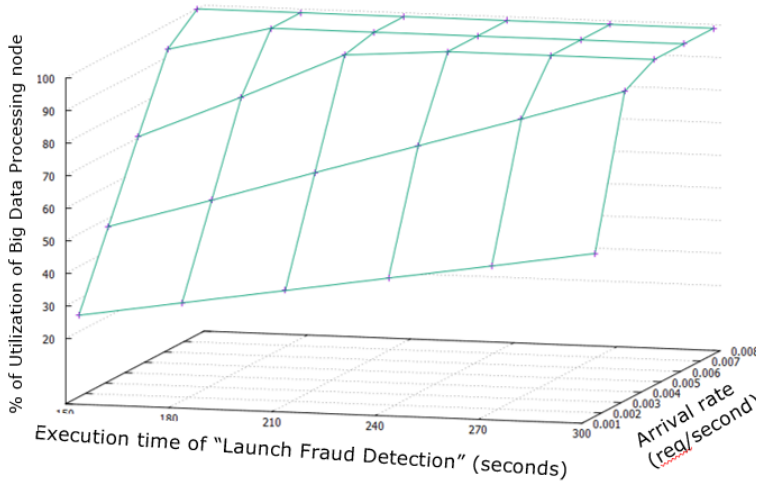
Big Blu Quality assessment

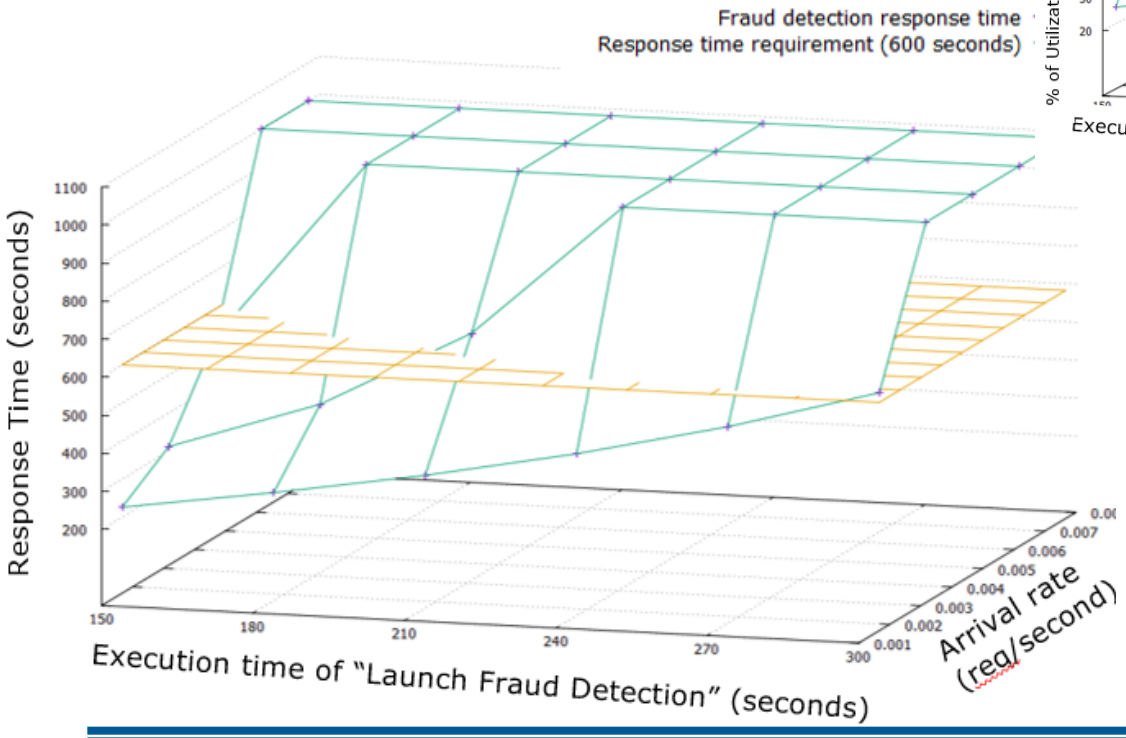
- Quality malfunction reported → maintenance



Big Blu Quality assessment

- Quality malfunction reported → mair
- Using the SimTool we obtain



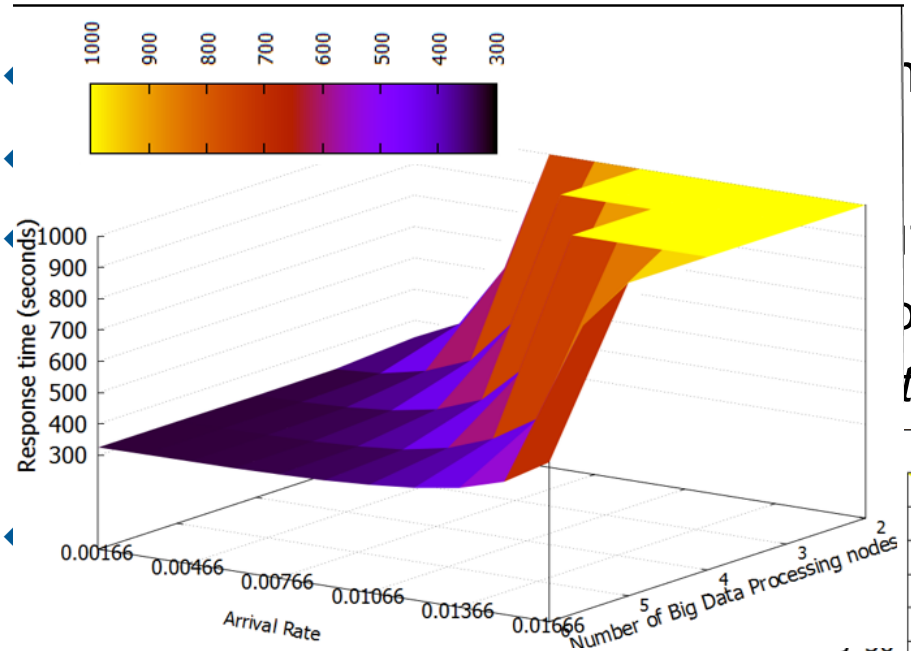


Big Blu Quality assessment

- Quality malfunction reported → maintenance
- Using the SimTool we obtain
- Developers see two possible solutions
  - Acquire more computing nodes to parallelise requests
  - Reengineer *Launch Fraud Detection* activity to make it faster

- Using the SimTool we obtain

Big Blu Quality assessment

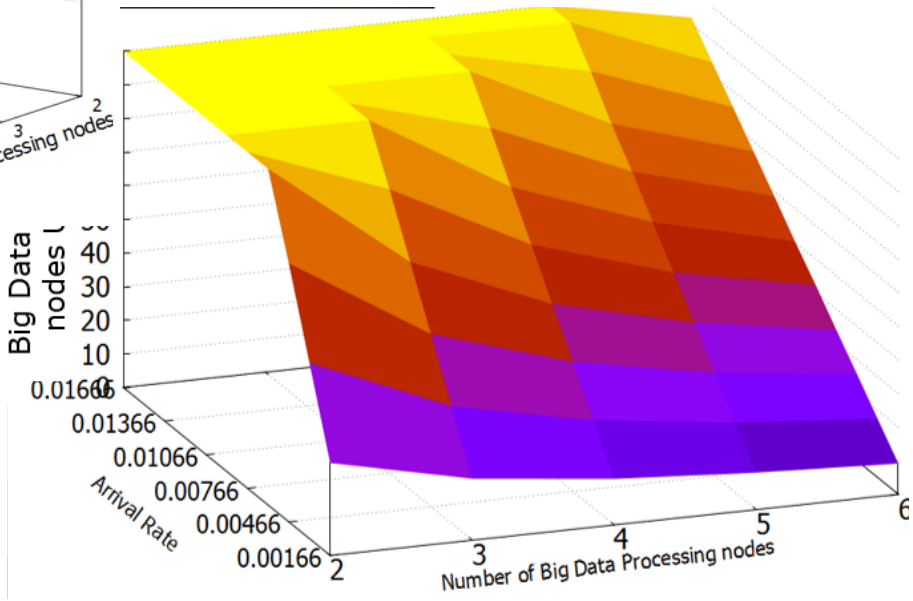


haintenance

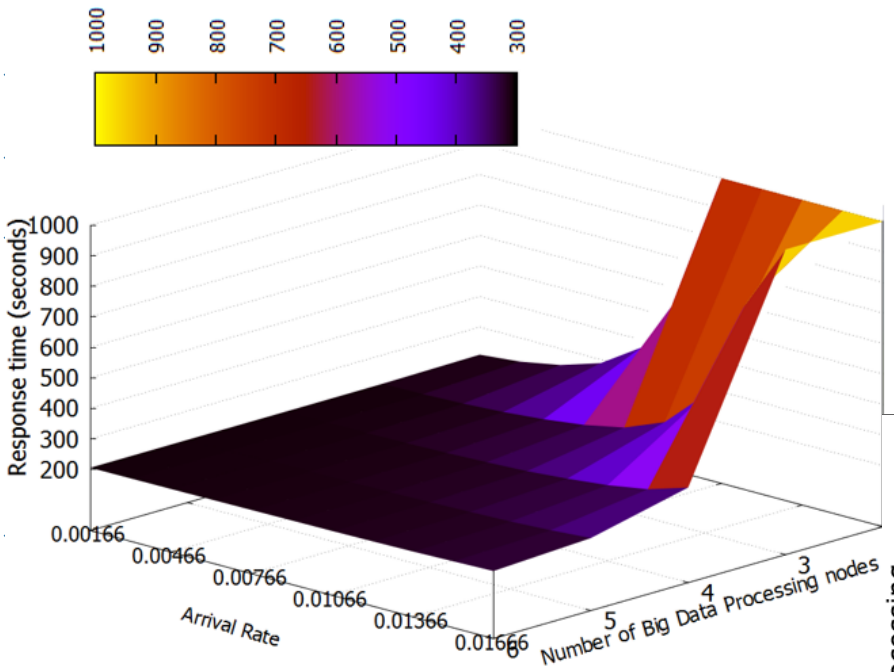
tions

r to parallelise requests

tion activity to make it faster



Big Blu Quality assessment

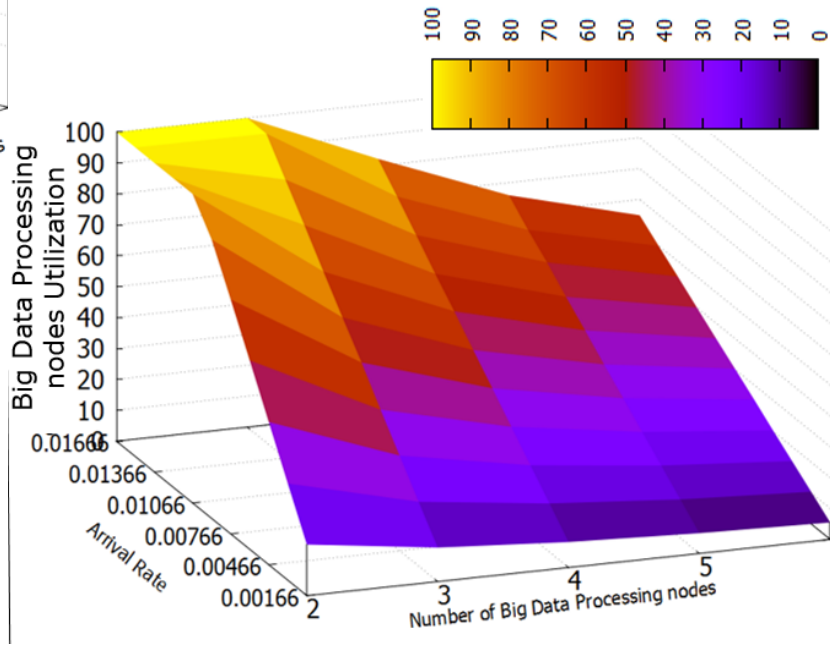


aintenance

ions

- to parallelise requests

on activity to make it faster

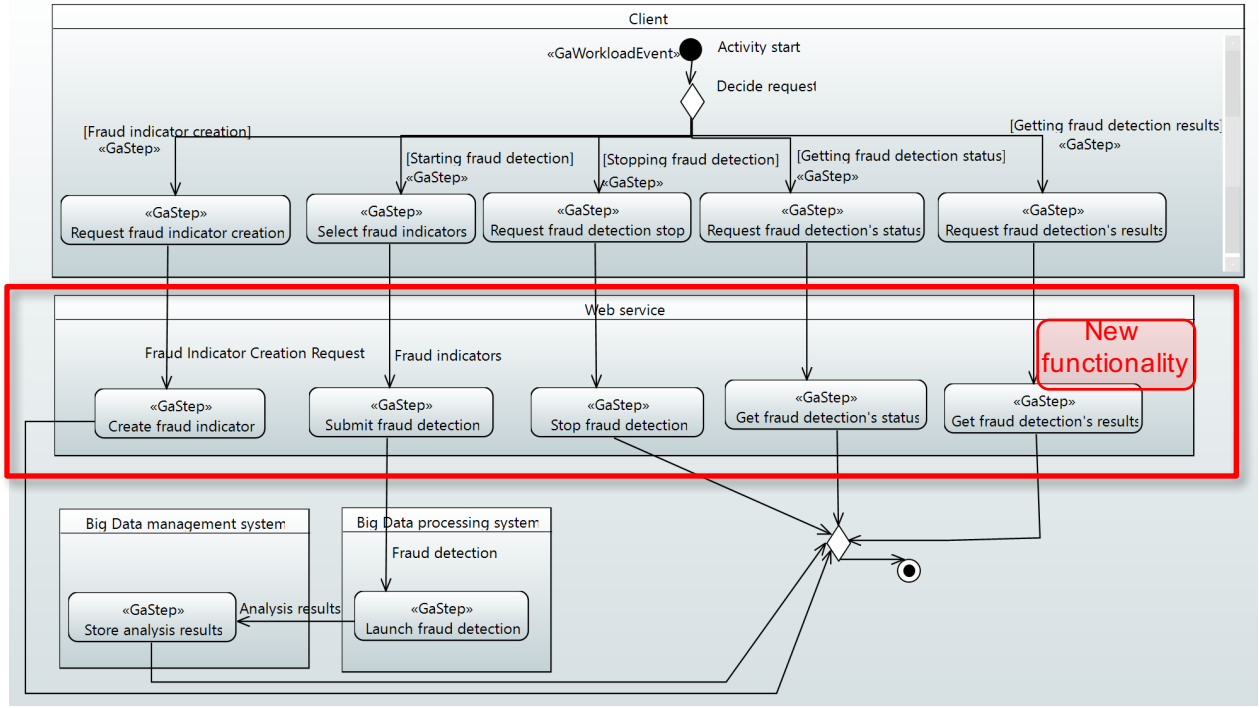


Big Blu Quality assessment

- Adding a new functionality
  - API that is invoked frequently
  - Provides volatile information to all clients

Big Blu Quality assessment

- Adding a new functionality
  - API that is invoked frequently
  - Provides volatile information to all clients
  - It executes in the Web Services layer



Big Blu Quality assessment

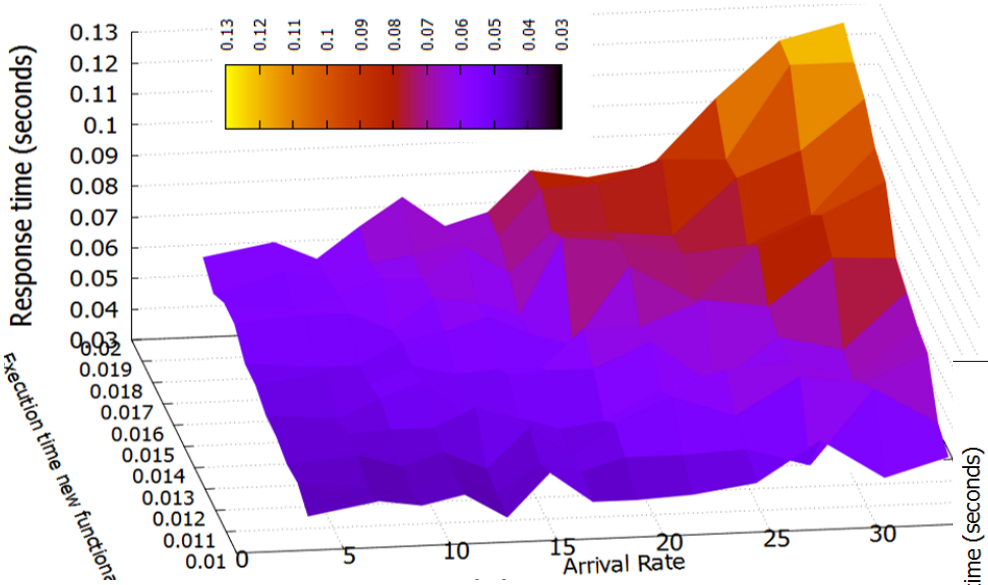
- Adding a new functionality
  - API that is invoked frequently
  - Provides volatile information to all clients
  - It executes in the Web Services layer
  - Each of the current 200 clients will make 1 request per minute
  - Developers believe that they can make the demand of the API to be less than 20 milliseconds for execution, but there are no new performance requirements

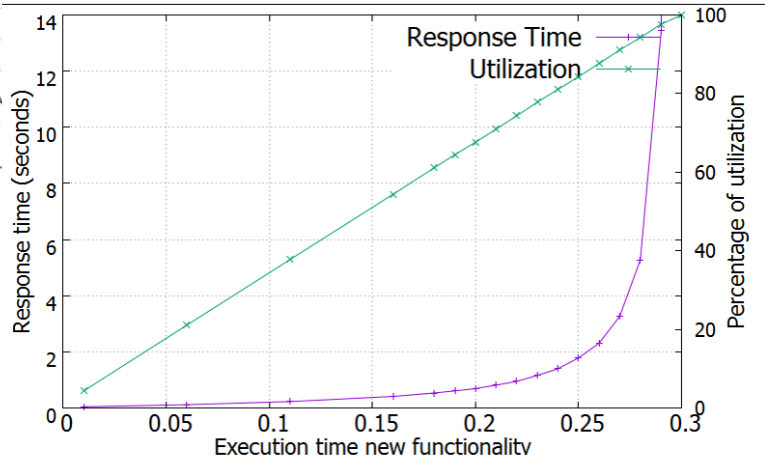
Big Blu Quality assessment

- Adding a new functionality
  - API that is invoked frequently
  - Provides volatile information to all clients
  - It executes in the Web Services layer
  - Each of the current 200 clients will make 1 request per minute
  - Developers believe that they can make the demand of the API to be less than 20 milliseconds for execution, but there are no new performance requirements
  - Developers do not know if their development will be good enough until integration or operation

Big Blu Quality assessment

- Adding a new functionality
  - Using the SimTool we obtain

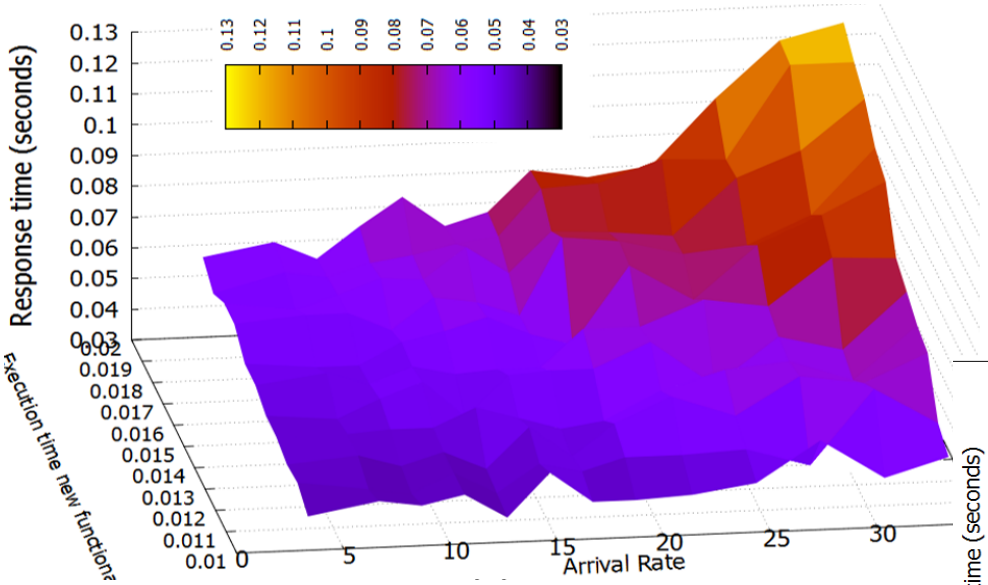


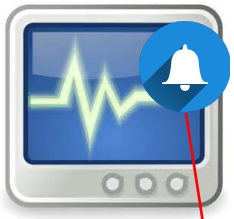


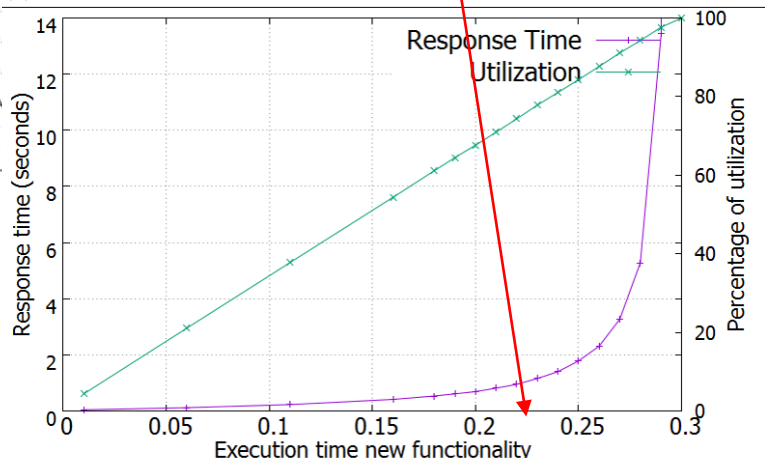
"The mean mean response time of all paths should be less than 10 seconds, except for the 10 minutes allowed for Launch Fraud Detection path"

Big Blu Quality assessment

- Adding a new functionality
  - Using the SimTool we obtain







Conclusions

- Report an experience on the utilization of a quality evaluation tool during DevOps-oriented software development
- Reduce the number of development cycles until reaching a satisfactory modification of the system
- Reported two common scenarios in development cycles
  - Maintenance activity
  - Development of new functionality

FUTURE:

- SimTool will incorporate characteristics of Big Data technologies (Big Blu uses Apache Spark)
- Complete the full integration of the different quality analysis tools used within the DICE methodology

Quality assessment in DevOps:

Automated Analysis of a Tax Fraud Detection System

THANK YOU FOR YOUR ATTENTION!