

A NETWORK SCIENCE BASED APPROACH FOR OPTIMAL MICROSERVICE GOVERNANCE

2020-021

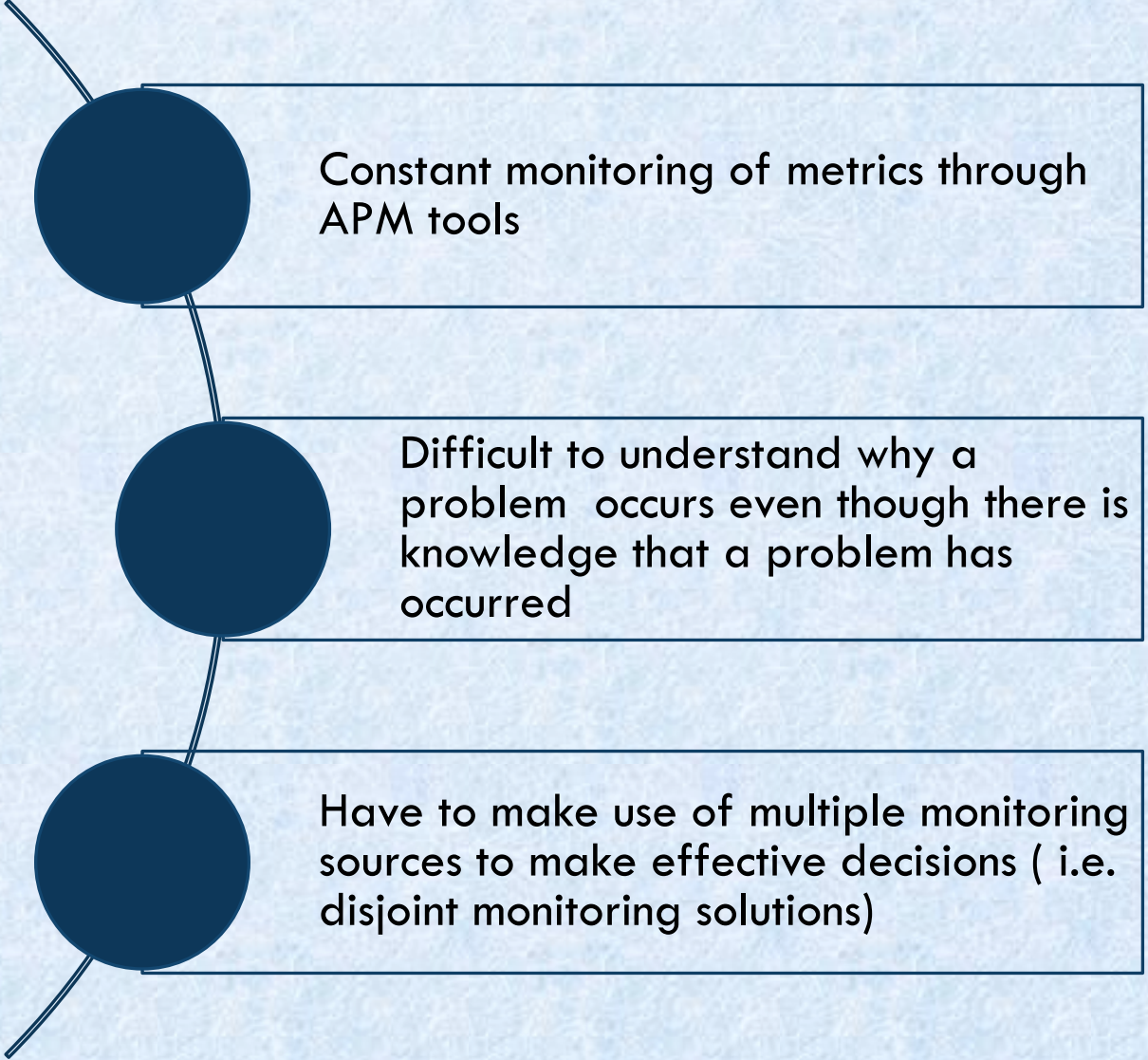
PROGRESS PRESENTATION 1

Student Name	Student ID
Leader: Saranga S.A. G	IT17016230
Member 2: De Silva N.	IT17006880
Member 3: L.S. Jayasinghe	IT17012966
Member 4: M.V. Lakshitha	IT17410250

Supervisor

Dr. Dharshana Kasthurirathna


PROBLEMS




Constant monitoring of metrics through APM tools



Difficult to understand why a problem occurs even though there is knowledge that a problem has occurred

Have to make use of multiple monitoring sources to make effective decisions (i.e. disjoint monitoring solutions)



IN SHORT, THE RESEARCH
PROBLEM THAT OUR
RESEARCH AIMS TO FULFIL
CAN SIMPLY BE DESCRIBED
AS FOLLOWS

- In deploying microservices through Kubernetes, there is no efficient and effective way for developers to evaluate and monitor the effectiveness and viability of a microservice deployment and identify possible performance bottlenecks through the disjoint monitoring solutions that currently available.
 - Hence, developers are not able to optimize their deployments such that they can make the optimal use of their deployed microservices in the cluster.
- 

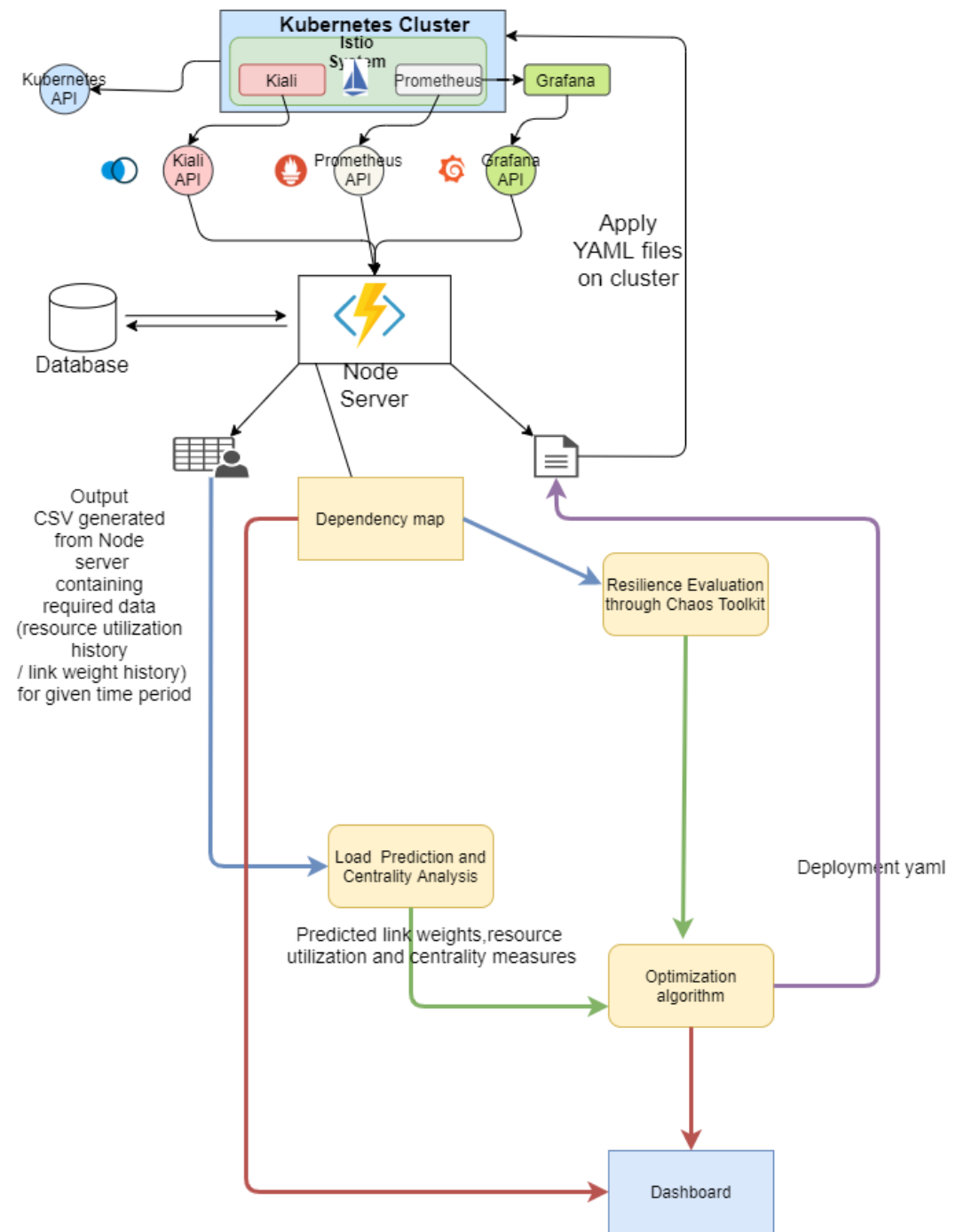


To model a network science-based approach to govern microservice deployments through evaluation and analysis of metrics gathered, and ultimately produce a proposed model which aids to optimize microservice deployments.

**MAIN
OBJECTIVE**



PROPOSED SOLUTION



IT17016230

- Target problem –

- Current marketplace does not contain a way to monitor and query all the metrics regarding the network and the hardware utilization, get an idea about the whole microservice architecture and take the dependency between each and every microservice into consideration.

- Proposed solution –

- To develop a system which can query necessary metrics in regard to network and hardware utilization of a cluster and to obtain an holistic idea about the quantified dependency between microservices

CURRENT PROGRESS - IT17016230 (70%)

- Creation and configuration of a Kubernetes cluster on AKS.
- Deployment of a sample microservice system
- Installation and configuration of Istio in the cluster and enable auto injection.
- Installation of Prometheus and Kiali and configuration to query metrics.
- Port forwarding of services to assign static ports.

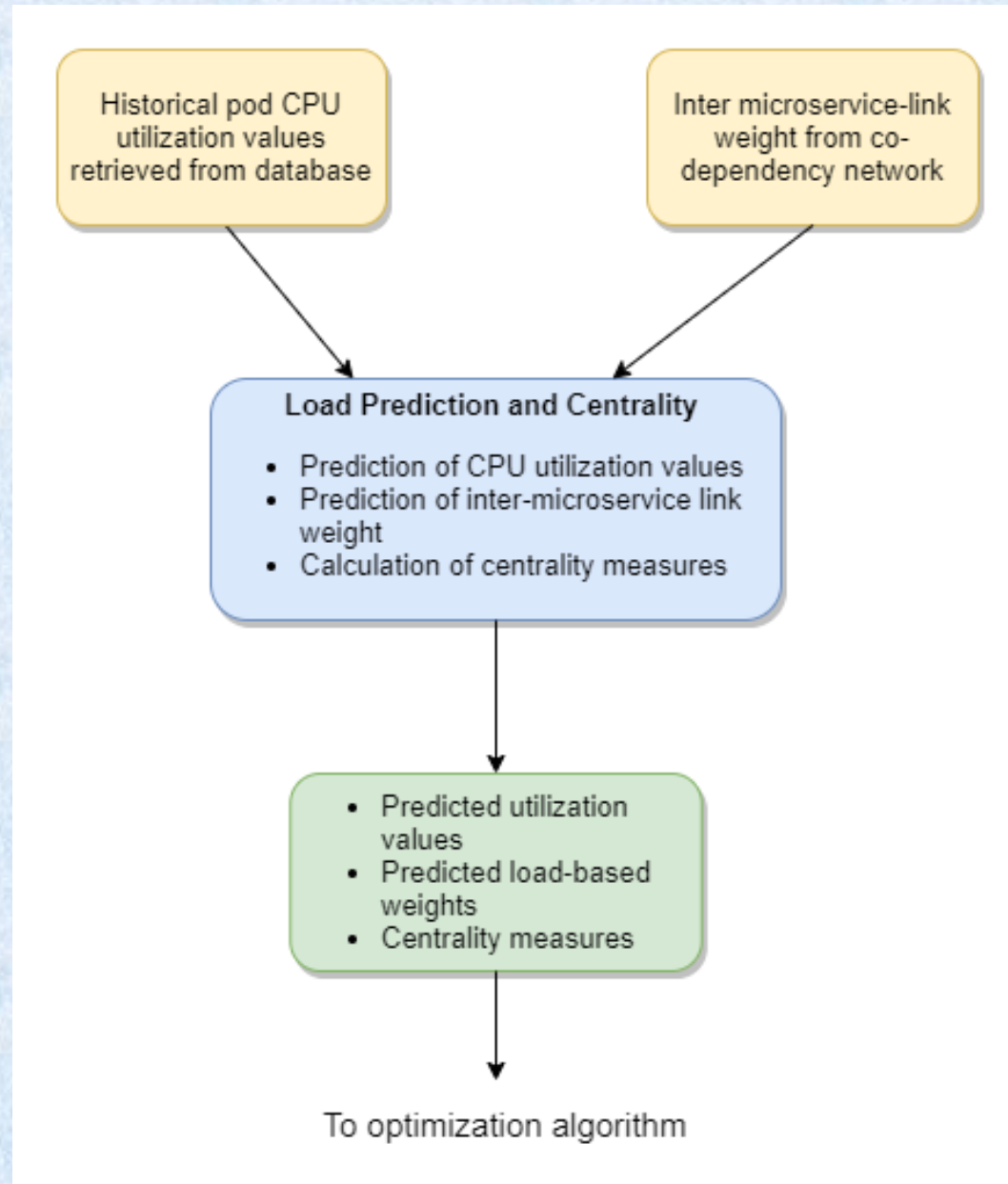
NEXT EXPECTED PROGRESS (100% COMPLETION) - IT17016230

- Development of a server and a database to provide metrics of the cluster to other team members.
- A backend to generate csv files on demand.
- A backend to generate YAML files according to the purposed optimal plan
- Package everything to a docker container to distribute the final product

IT17006880

- Target problem – The use of localized rule-based autoscaling technologies used in microservice deployment which fail to capture a globalized perspective on the effect of autoscaling decisions
- Proposed solution – Usage of a combination of resource utilization prediction and prediction of load based metrics to facilitate an improved autoscaling policy which captures a holistic perspective on the effect of autoscaling.

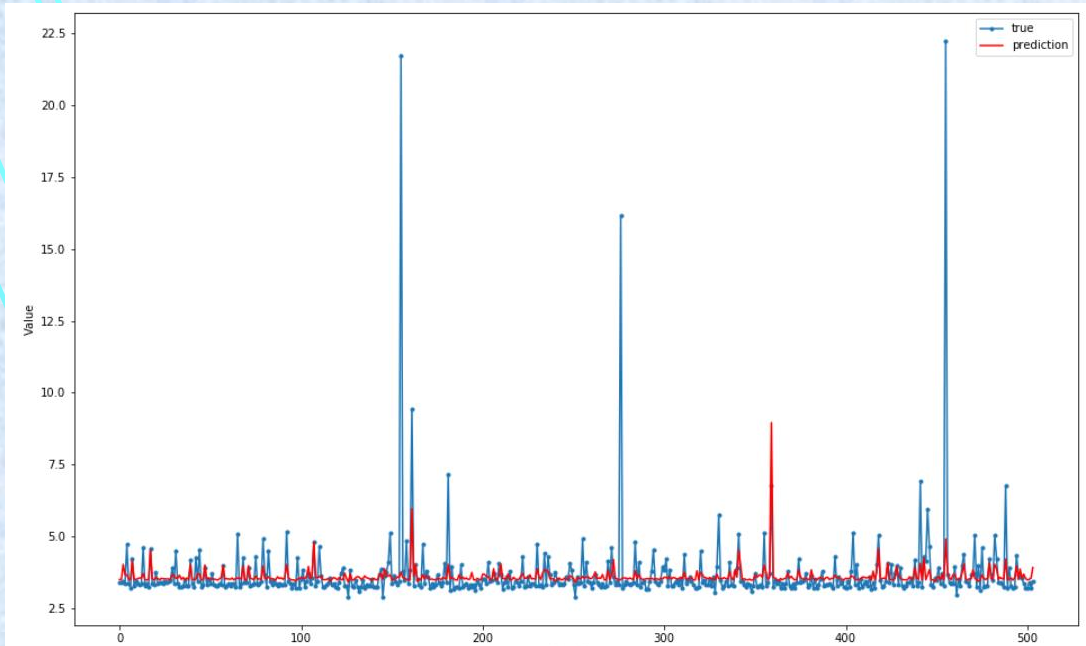
LOAD PREDICTION AND CENTRALITY ANALYSIS COMPONENT - IT17006880



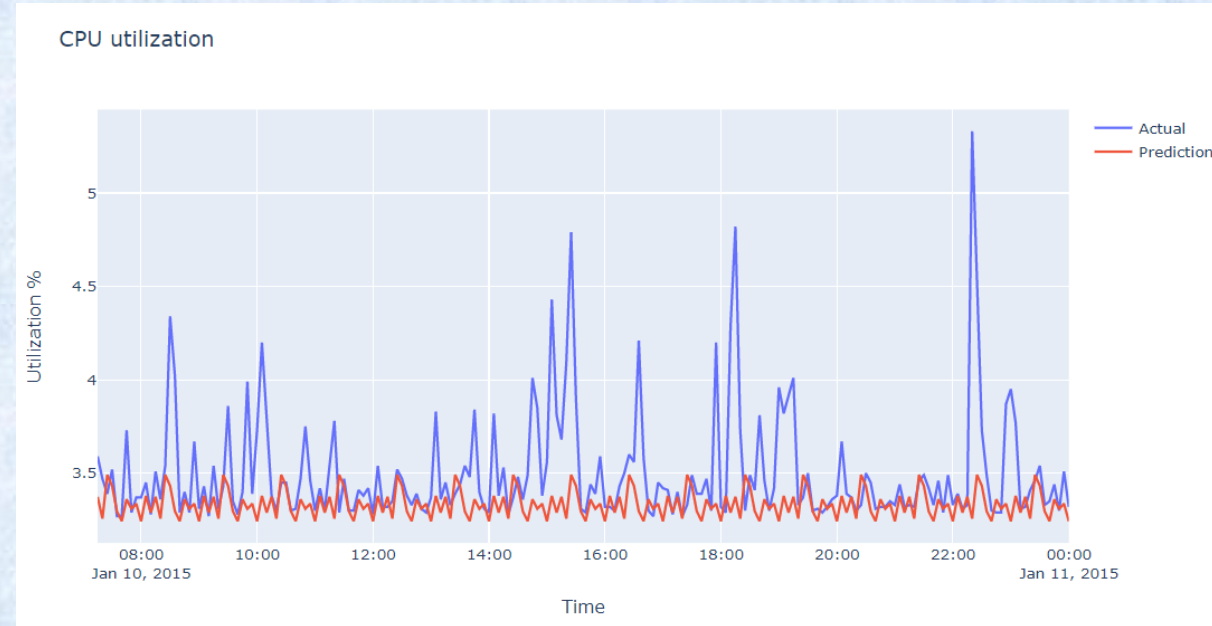
CURRENT PROGRESS - IT17006880 (70%)

- Evaluation of various time series prediction models for resource utilization prediction and load-based inter-microservice link weight
- Selection of the optimal prediction model from the evaluated prediction models and implementation using selected prediction model
- Implemented methodology for calculation of centrality measures from co-dependency network
- Implementation of initial code scripts for load-based link weight prediction

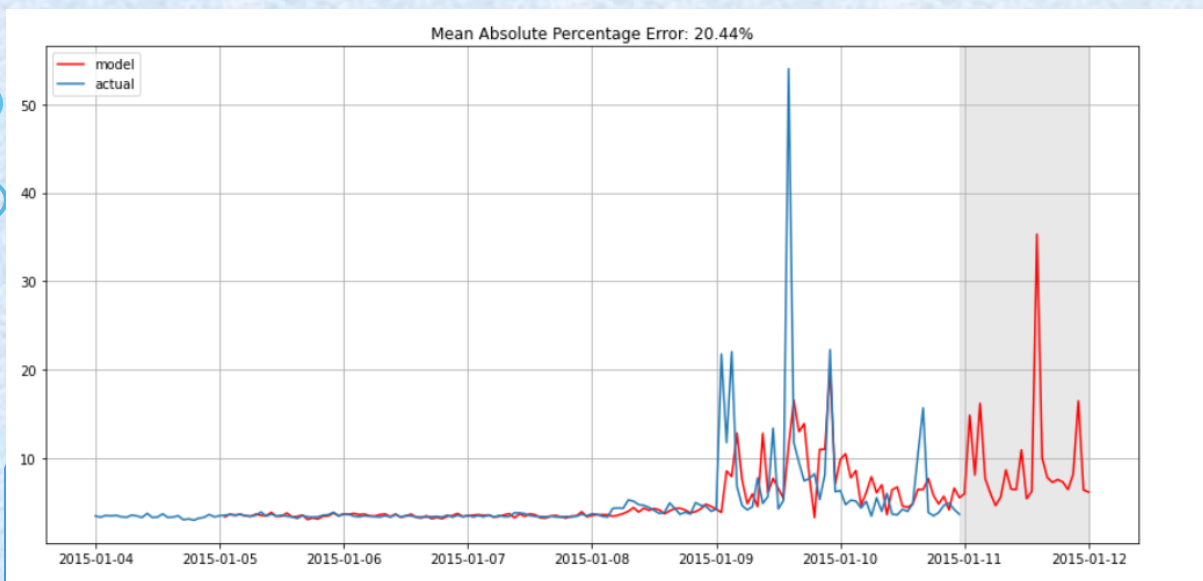
Model	MAPE/RMSE
ARIMA	MAPE 20.44% RMSE 3.40
HOLT Winters	MAPE 5.76% RMSE 0.346
Auto ML (1 step previous)	MAPE 8.13% RMSE 1.334
XGBOOST (1 step previous)	MAPE 6.88 % RMSE 1.194
LSTM Univariate (1 step previous)	MAPE 1.53% RMSE 0.109
LSTM Multivariate (12 steps previous)	MAPE 3.86% RMSE 0.276



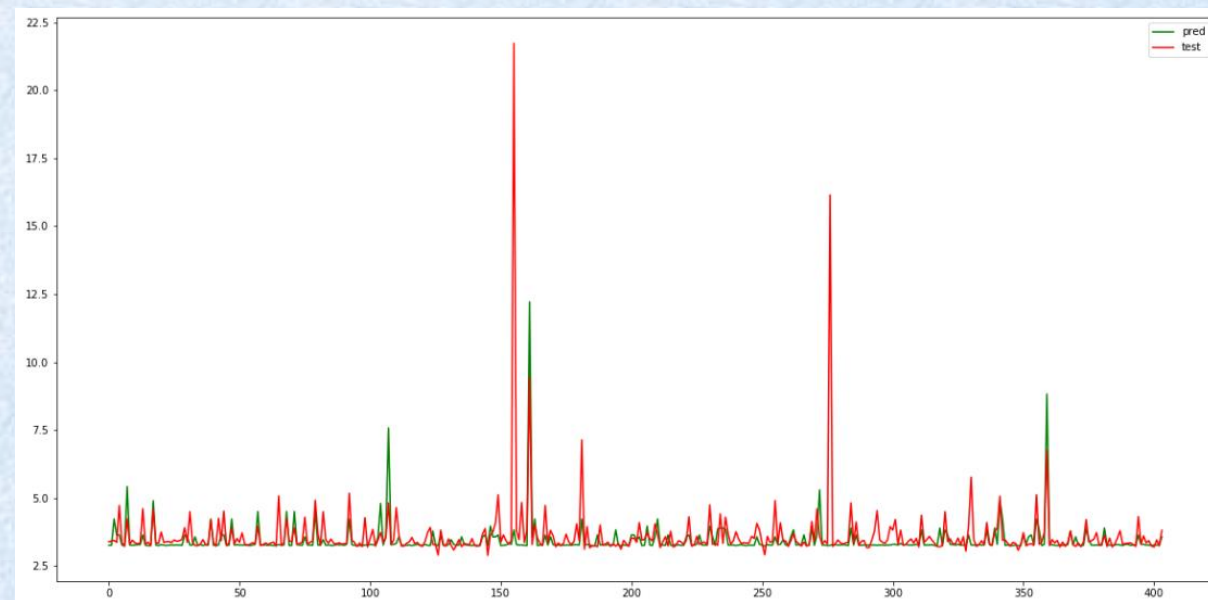
Auto ML - TPOT



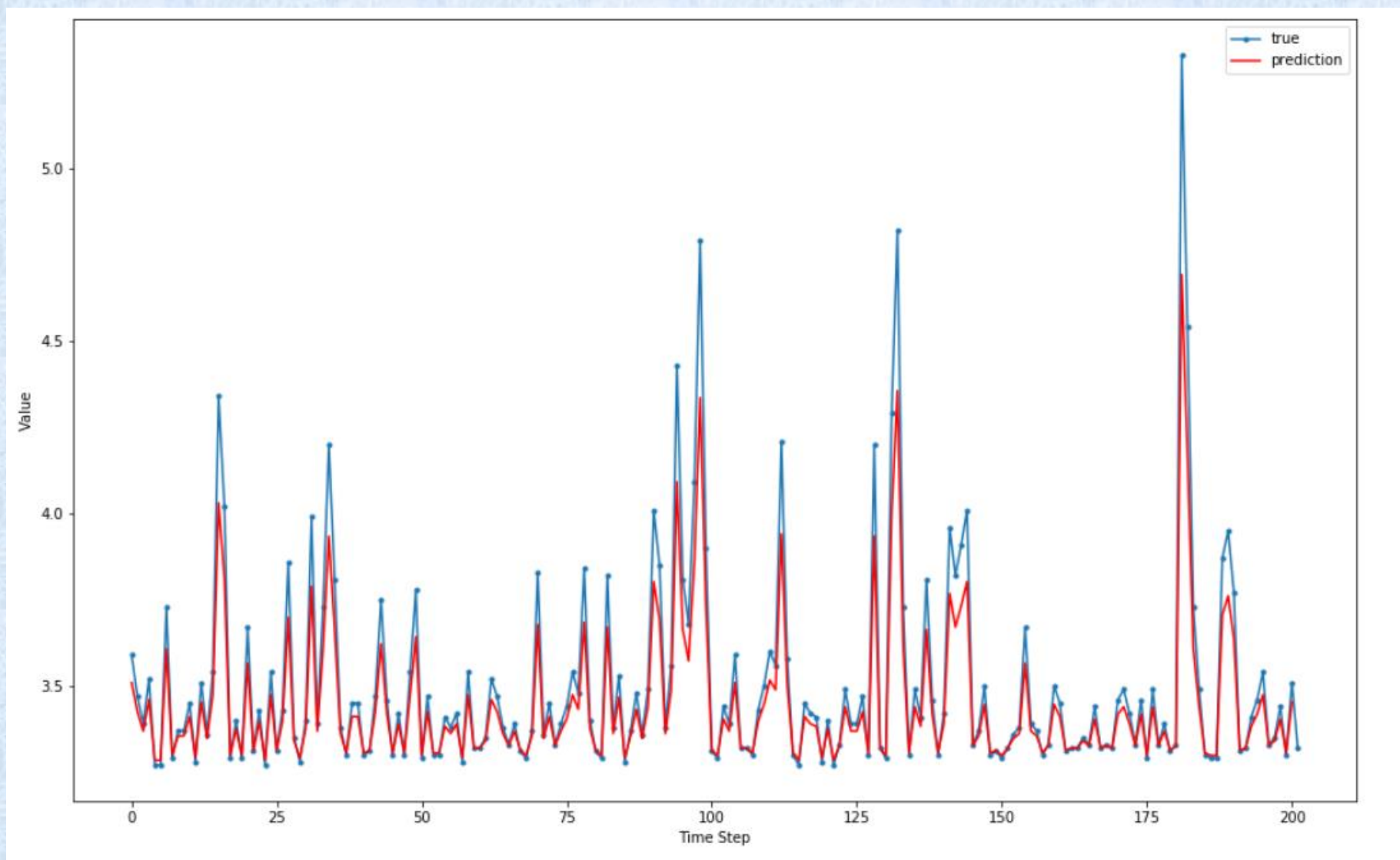
Holt Winters



ARIMA



XG Boost



LSTM Univariate Model

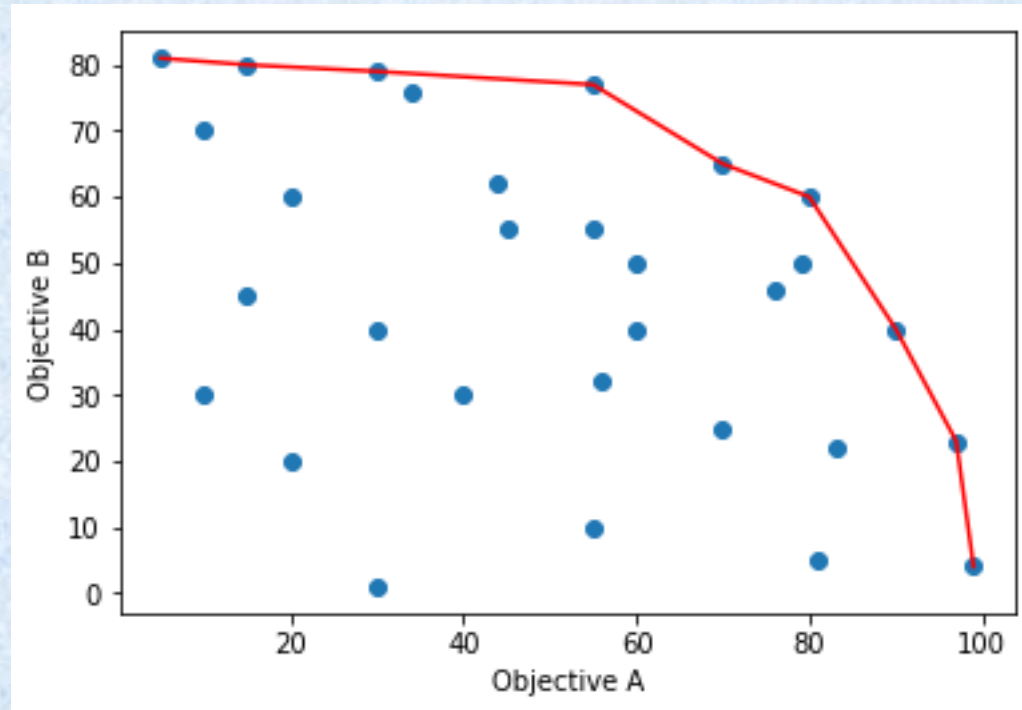
NEXT EXPECTED PROGRESS (100% COMPLETION) - IT17006880

- Integrate with co-dependency map and the related components
 - Add necessary conversion functions to convert data from metric APIs to a csv format to be stored in the database solutions
 - Integrate with database solution to retrieve historical data for time series prediction
- Usage of actual values utilization and load-based weight from the developed APIs

IT17012966

- Target problem – Increase performance and availability in microservice application.
- Proposed solution – Optimal placement for microservices

NSGA II



CHROMOSOME DESIGN

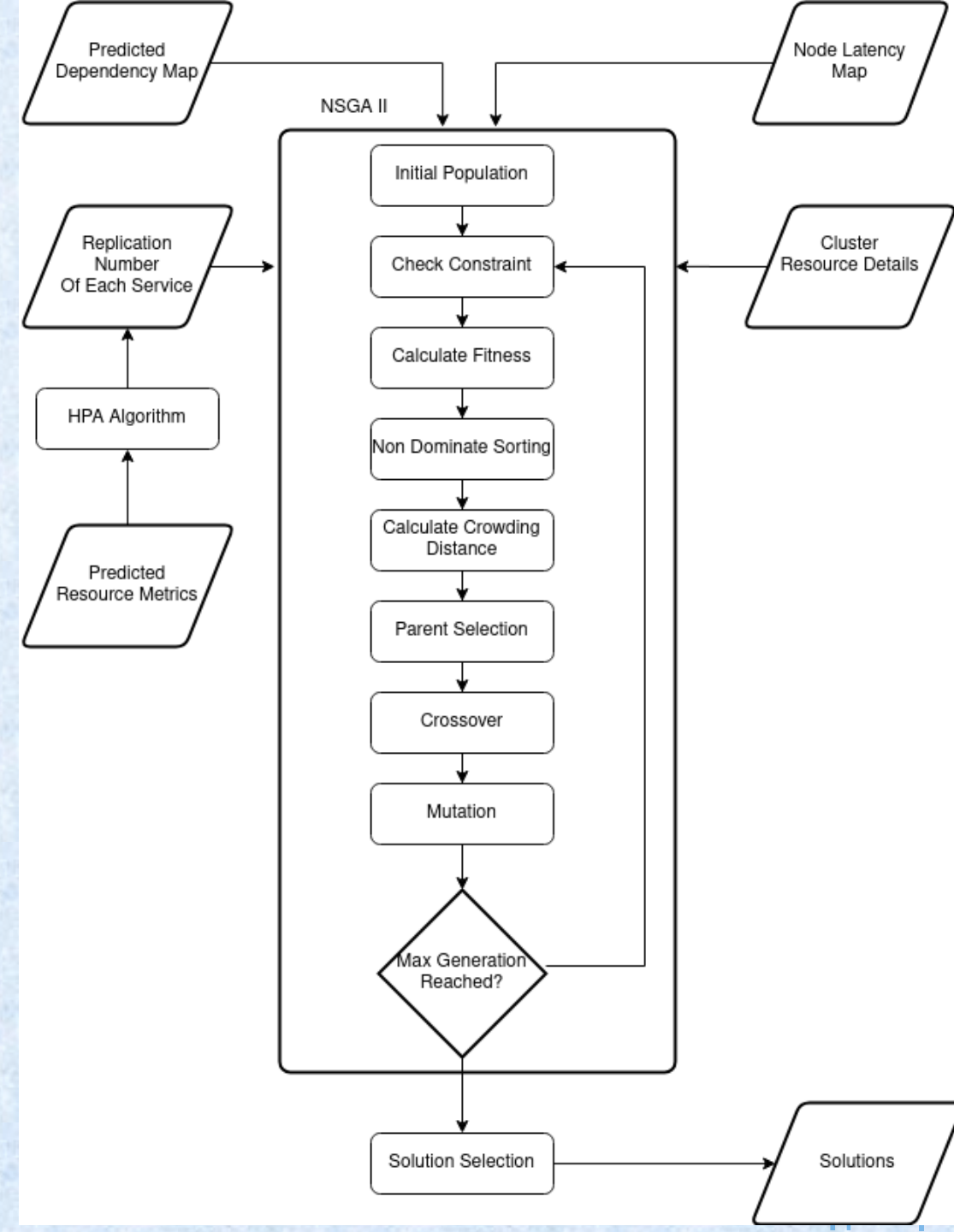
Service\Node	Node 1	Node 2	Node 3
Service A	0	2	2
Service B	3	0	1
Service C	1	3	4
Service D	1	1	0



Deployment matrix

[0,2,2,3,0,1,1,3,4,1,1,0]

METHODOLOGY



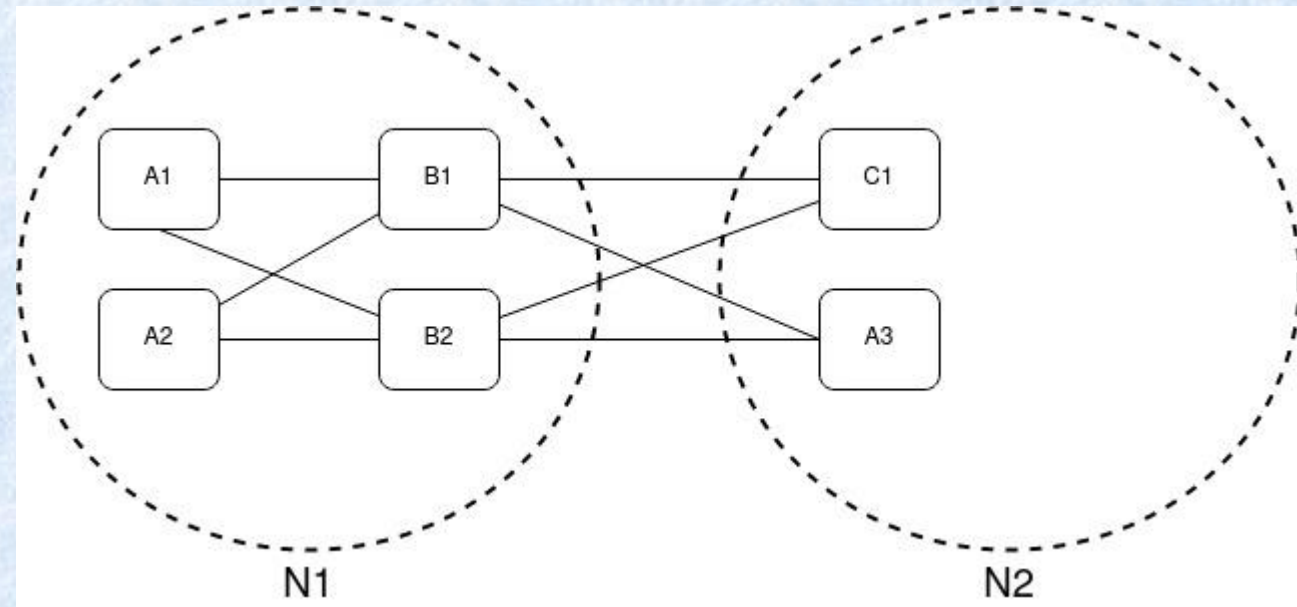
PERFORMANCE FITNESS

- i =each link in pod level
- n =number of link
- m = number of link in app level
- W =request weight of a link in app level
- L =latency of a link
- A_j =Each link average latency in app level

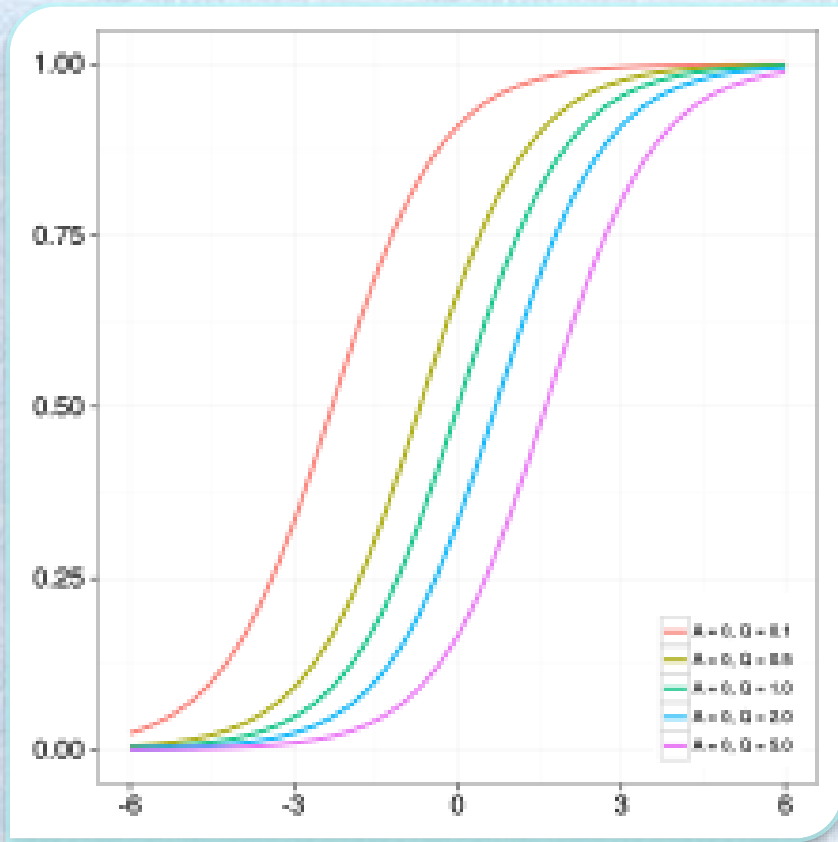
- $$A_j = \frac{\sum_{i=1}^{i=n} L_i}{n}$$

- $$TL = \sum_{j=1}^{j=m} W_j * A_j$$

- $P \sim 1/TL$



AVAILABILITY FITNESS



- When $x=0, y \sim 0$
- When $x=1, y \sim 1$
- y never exceed 1.
- $A_i = R_i * \text{generalisedLogisticFunction}(S_i/R_i)$
- $TA = \sum_{i=1}^n A_i - \text{generalisedLogisticFunction}(1) * \sum_{i=1}^n R_i$

CURRENT PROGRESS - IT17012966 (60%)

- NSGA II
- Performance fitness
- Availability fitness
- Separate the solution
- API server

DEMO

https://mysliit-my.sharepoint.com/:v:/g/personal/it17016230_my_sliit_lk/EYY-6nHpLwxPty46J42tOu0BVkndS7UIJBRRS52TVNHEjQ?e=xDJ25B

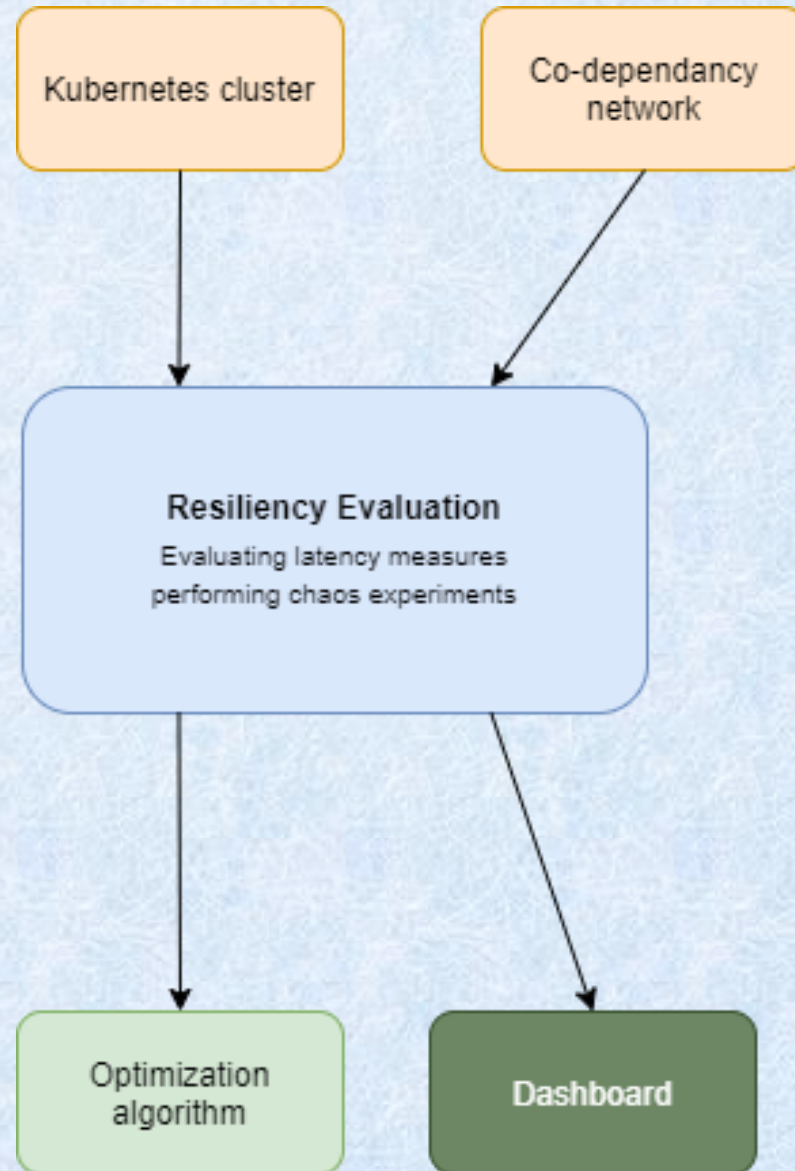
NEXT EXPECTED PROGRESS (100% COMPLETION) - IT17012966

- Algorithm to retrieve required instances for each microservice
- Testing functionalities
- Put algorithm in to the scheduler.
- Connect algorithm with other component
- Outputs warnings
- UI Creation

IT17410250

- Target problem – Lack of focus on resiliency measures when considering the microservice deployment and governance
- Proposed solution – perform resiliency evaluations and exploit the results on creating a deployment plan for an optimal microservice governance

RESILIENCY EVALUATION COMPONENT



CURRENT PROGRESS - IT17410250 (50%)

- Identifying the impact of resiliency measures on microservice governance.
- Performing chaos experiments on Kubernetes cluster.
- Initial implementation of dashboard

SAMPLE SCRIPT OF TERMINATING A POD

```
1  {
2    "title": "Terminate-pods to examine the behaviour",
3    "description": "|",
4    "tags": [
5      "tutorial",
6      "filesystem"
7    ],
8    "steady-state-hypothesis": {
9      "title": "all-our-microservices-should-be-healthy",
10     "probes": [
11       {
12         "type": "probe",
13         "name": "service-is-unavailable",
14         "tolerance": [
15           200,
16           503
17         ],
18         "provider": {
19           "type": "http",
20           "url": "http://52.188.218.236:80/productpage"
21         }
22       }
23     ],
24   },
25   "method": [
26     {
27       "name": "terminate-pod",
28       "type": "action",
29       "provider": {
30         "type": "python",
31         "module": "chaosk8s.pod.actions",
32         "func": "terminate_pods",
33         "arguments": {
34           "rand": "true"
35         }
36       }
37     }
38   ]
39 }
```

SAMPLE EXPERIMENT FOR SIMPLE PYTHON SERVICE

```
1 [{"title": "Does our service tolerate the loss of its exchange file?",
2  "description": "Our service reads data from an exchange file, can it support that file disappearing?",
3  "tags": [
4    "tutorial",
5    "filesystem"
6  ],
7  "steady-state-hypothesis": {
8    "title": "The exchange file must exist",
9    "probes": [
10     {
11       "type": "probe",
12       "name": "service-is-unavailable",
13       "tolerance": [
14         200,
15         503
16       ],
17       "provider": {
18         "type": "http",
19         "url": "http://52.188.218.236:80/productpage"
20       }
21     }
22   ],
23 },
24 "method": [
25   {
26     "name": "move-exchange-file",
27     "type": "action",
28     "provider": {
29       "type": "python",
30       "module": "os",
31       "func": "rename",
32       "arguments": {
33         "src": "./exchange.dat",
34         "dst": "./exchange.dat.old"
35       }
36     }
37   }
38 ]
```

```
(chaostk) lucky@DESKTOP-29JARHE:/mnt/c/personal$ chaos run newexperiment.json
[2020-07-12 14:02:51 INFO] Validating the experiment's syntax
[2020-07-12 14:02:52 INFO] Experiment looks valid
[2020-07-12 14:02:52 INFO] Running experiment: Does our service tolerate the loss of its exchange file?
[2020-07-12 14:02:52 INFO] Steady state hypothesis: all-our-microservices-should-be-healthy
[2020-07-12 14:02:52 INFO] Probe: service-is-unavailable
[2020-07-12 14:02:52 INFO] Steady state hypothesis is met!
[2020-07-12 14:02:52 INFO] Action: terminate-pod
[2020-07-12 14:02:55 INFO] Steady state hypothesis: all-our-microservices-should-be-healthy
[2020-07-12 14:02:55 INFO] Probe: service-is-unavailable
[2020-07-12 14:02:55 INFO] Steady state hypothesis is met!
[2020-07-12 14:02:55 INFO] Let's rollback...
[2020-07-12 14:02:55 INFO] No declared rollbacks, let's move on.
[2020-07-12 14:02:55 INFO] Experiment ended with status: completed
```

SAMPLE REPORT

Summary

Does our service tolerate the loss of its exchange file?

Our service reads data from an exchange file, can it support that file disappearing?

Status failed
Tagged tutorial, filesystem
Executed From b2d6b69a2f14
Platform Linux-4.4.0-66-generic-x86_64-with-Ubuntu-16.04-xenial
Started Wed, 24 Jan 2018 13:25:55 GMT
Completed Wed, 24 Jan 2018 13:25:55 GMT
Duration 0 seconds

Experiment

The experiment was made of 1 actions, to vary conditions in your system, and 0 probes, to collect objective data from your system during the experiment.

Steady State Hypothesis

The steady state hypothesis this experiment tried was **"The exchange file must exist"**.

It was verified with the following probes:

Probe	Tolerance
service-is-unavailable[200, 503]	

Result

The experiment was conducted on Wed, 24 Jan 2018 13:25:55 GMT and lasted roughly 0 seconds.

Action - move-exchange-file

Status succeeded
Background False
Started Wed, 24 Jan 2018 13:25:55 GMT
Ended Wed, 24 Jan 2018 13:25:55 GMT
Duration 0 seconds

The action provider that was executed:

Type python
Module os
Function rename
Arguments {'dst': './exchange.dat.old', 'src': './exchange.dat'}

NEXT EXPECTED PROGRESS (100% COMPLETION) - IT17410250

- Writing a set of chaos experiments on kubernetes cluster
- Generating necessary data from resiliency evaluation to use in optimization algorithm
- Completing implementation of the dashboard