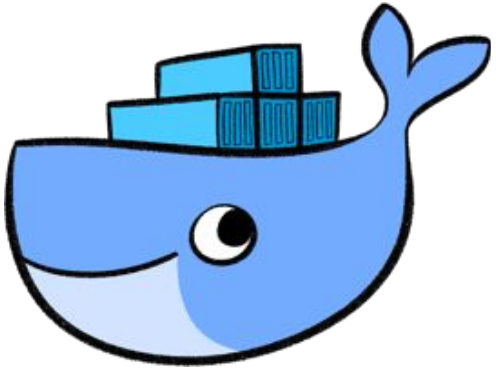


GERENCIAMENTO DE CONTAINERS DOCKER COM AGENTES

VINICIUS PFEIFER

AGENDA

- CONTAINERS
- DOCKER
- PROPOSTA
- DEMO
- RESULTADOS
- TRABALHOS FUTUROS

1.

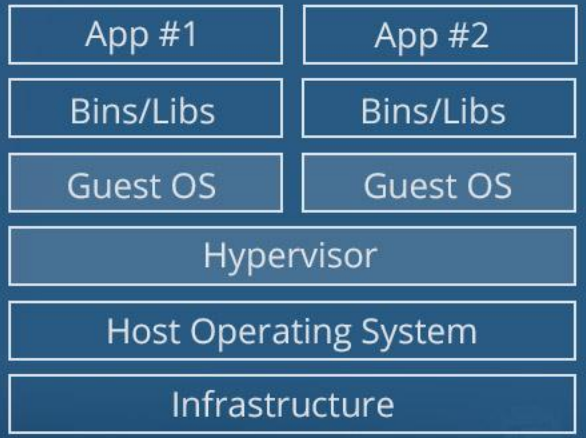
CONTAINERS

“

Método de virtualização de sistema operacional que permite executar um aplicativo e suas dependências em processos com recursos isolados.

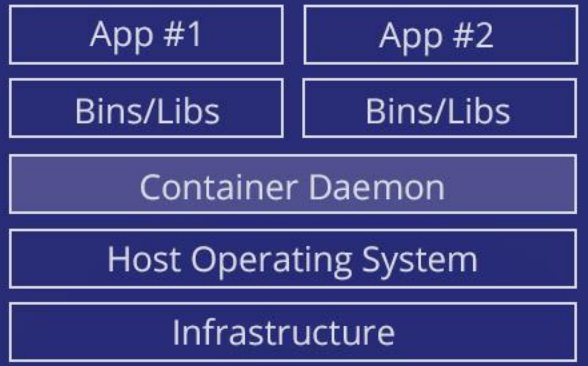
Amazon

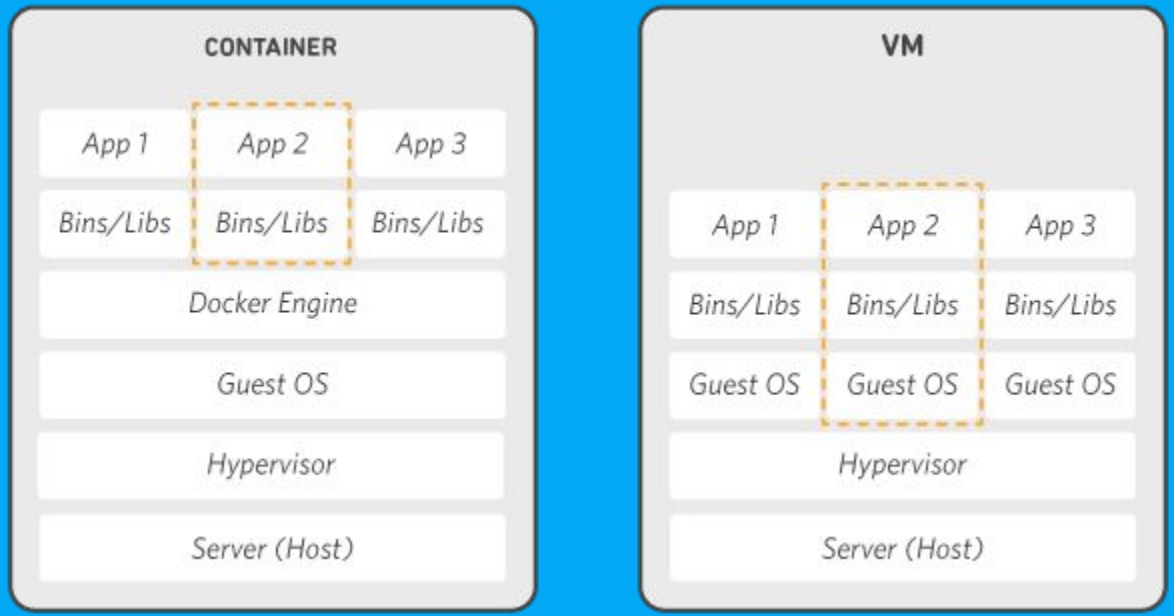
VIRTUAL MACHINES



WHATS *the* DIFF?

CONTAINERS




CONTAINER
App 1
App 2
App 3
Bins/Libs
Bins/Libs
Bins/Libs
Docker Engine
Guest OS
Hypervisor
Server (Host)
VM
App 1
App 2
App 3
Bins/Libs
Bins/Libs
Bins/Libs
Guest OS
Guest OS
Guest OS
Hypervisor
Server (Host)

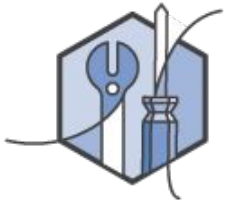
BENEFÍCIOS

CONSISTÊNCIA DE AMBIENTE

Viabiliza portabilidade encapsulando todas as dependências necessárias para que a aplicação seja executada. Tudo o que estiver empacotado no container será executado da mesma forma em qualquer ambiente.

EFICIÊNCIA OPERACIONAL

Aproveita melhor os recursos de infraestrutura, permitindo executar várias aplicações em uma mesma instância. O uso reduzido de recursos permitem iniciar e parar aplicações rapidamente.



PRODUTIVIDADE

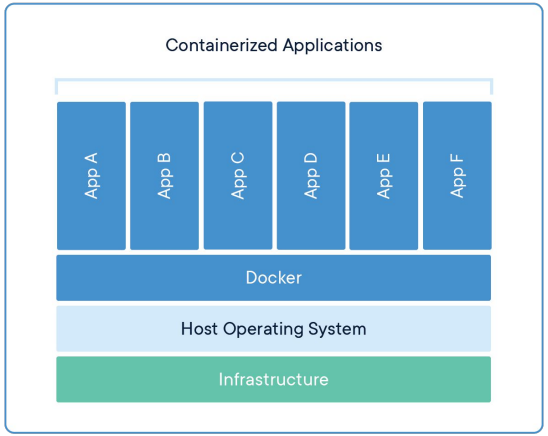
Aumenta a produtividade do desenvolvedor removendo conflitos de dependências entre serviços. Os containers são isolados entre si, eliminando a preocupação com bibliotecas e dependências dos serviços. Implantação mais segura, pois as dependências já estão empacotadas no container.

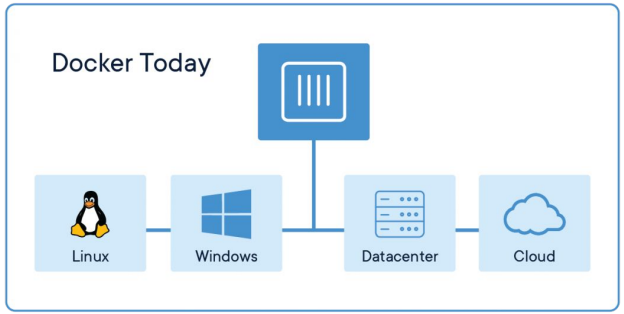
CONTROLE DE VERSÕES

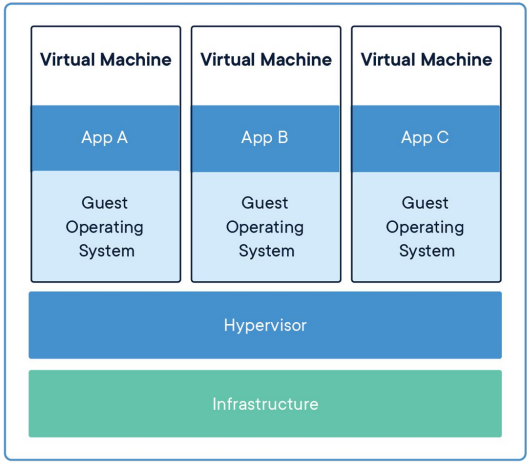
Permite controlar as versões do código do seu aplicativo e de suas dependências.

What is a Container

A standardized unit of software







According to a 2018 IDC survey on containers, 85% of container adopters are using containers for production apps and 83% of container deployers use containers on multiple public clouds (3.7 clouds on average) with 55% of containers running on-premises and 45% run in the public cloud. As organizations continue to scale their container environments in dynamic IT environments they require a solution to address their security, management, and governance needs while maintaining operational efficiency and developer agility.

2.

DOCKER

“

É uma ferramenta desenvolvida para facilitar a criação, implantação e execução de aplicações baseadas em containers.

opensource.com

WHAT CAN DOCKER DO ?

How Docker Works for You

Developers

Tooling that is simple to use, yet powerful and delivers a great user experience so you can focus on what you love — writing great code.

→ Get Started

IT Operations

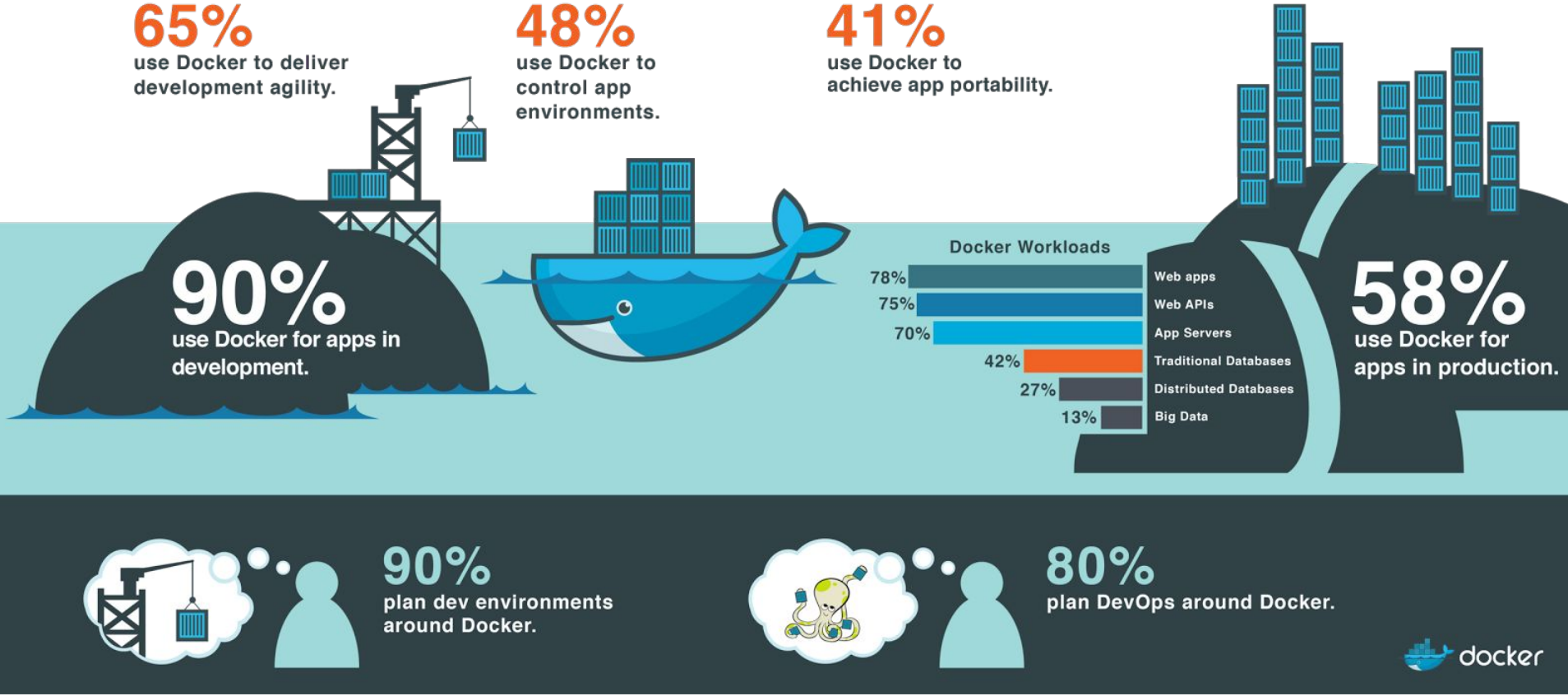
Docker delivers an enterprise-ready container platform to deploy and run applications in a way that makes the best sense for your customers and business.

→ Get Started

Business Leader

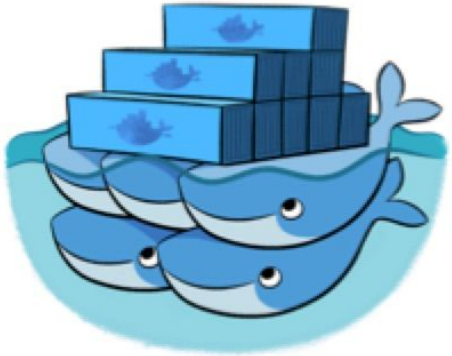
Docker provides a platform to drive your digital transformation by accelerating new innovation while dramatically driving down your existing IT costs.

→ Get Started



Docker Inc., 2016

FERRAMENTAS DE GERENCIAMENTO



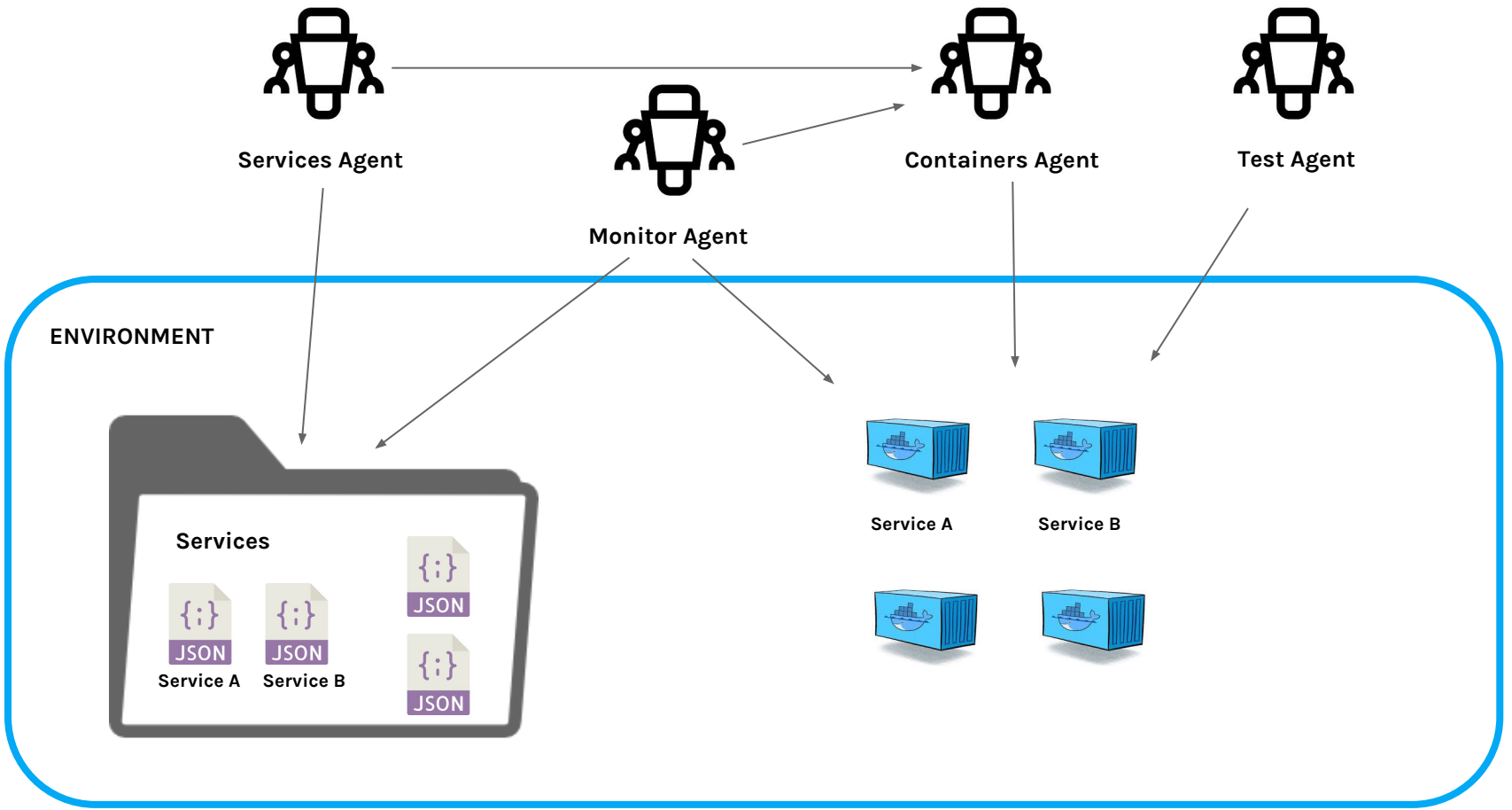
Docker Swarm

Kubernetes
(from Google)

Mesosphere DCOS
(based on Apache Mesos)

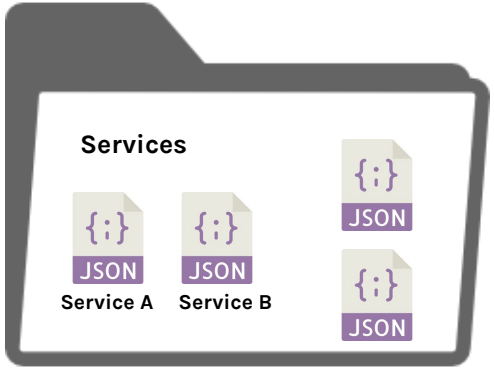
3.

PROPOSTA


Services Agent
Monitor Agent
Containers Agent
Test Agent
ENVIRONMENT
Services
JSON
Service A
JSON
Service B
JSON
JSON
Service A
Service B

SERVICES

Pasta com os arquivos modelo dos serviços.




```
{
 "name" : "aspnetcore_sample",
 "image" : "microsoft/dotnet-samples:aspnetapp",
 "replicas" : 1,
 "hostPort" : "8000",
 "containerPort" : "80"
}
```

- **name** - Nome do serviço.
- **Image** - Imagem do Docker
- **hostPort** - Porta do host
- **containerPort** - Porta do container

SERVICES AGENT

Services Agent

- Monitora a pasta Services
- Quando um novo serviço é criado adiciona ele na base de crenças.
- Quando um arquivo de serviço é apagado envia um sinal de achieve para o agente de container remover os containers daquele serviço

CONTAINERS AGENT

Containers Agent

- Monitora os containers do host
- Remove containers quando o agente de serviços manda sinais
- Cria e reinicia um container quando recebe sinal do agente de monitoramento

MONITOR AGENT

Monitor Agent

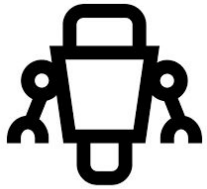
Existem 2 implementações do agente de monitoramento.

Interval Monitor Agent

AI Monitor Agent

INTERVAL MONITOR AGENT

Monitora os serviços a cada 500 ms verificando se existe algum serviço sem container correspondente.



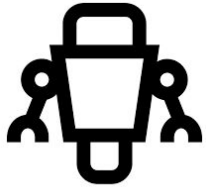
Interval Monitor Agent

AI MONITOR AGENT

AI Monitor Agent

Executa o algoritmo proposto no artigo *Detection of transmissible service failure in distributed service-based systems* dos autores Dayong Ye, Qiang He, Yanchun Wang, Yun Yang

SUPERVISORY GAME



AI Monitor Agent

Table 1. Payoff matrix of a supervisor and a worker

A supervisor \ A worker	*laze*	*work*
inspect	$r - c, -(p - b)$	$-c, 0$
not inspect	$-r, b$	$0, 0$

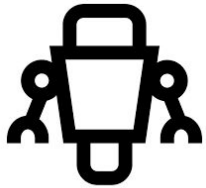
Table 2. Payoff matrix of a task agent and a task

A task agent \ A task	*fail*	*not fail*
monitor	x_{11}, y_{11}	x_{12}, y_{12}
not monitor	x_{21}, y_{21}	x_{22}, y_{22}

Q-LEARNING ALGORITHM

1 Let $k = 1$ be the time slot, γ be the discount factor, and ξ be the learning rate;
2 For each action, a, initialise value function $Q(a)$ to 0 and initialise the probability for selecting the action, $\pi(a)$, to $\frac{1}{n}$, where n is the number of available actions;
3 **repeat**
4 select an action, a, from available actions based on the probability distribution π over these actions;
5 take the selected action, observe payoff, pay, and update the Q-value of the selected action:
$Q(a) \leftarrow (1 - \xi)Q(a) + \xi(pay + \gamma max_{a'} Q(a'))$;
6 approximate the failure probability of the monitored task based on the Q-value, ;
7 evaluate the importance of the task;
8 for each action a, update the probability for selecting the action, $\pi(a)$, based on the approximation and evaluation;
9 $\pi \leftarrow Normalise(\pi)$;
10 $\xi \leftarrow \frac{k}{k+1} \cdot \xi$;
11 $k \leftarrow k + 1$;
12 **until** *the monitored task fails and is detected*;

DRAFT

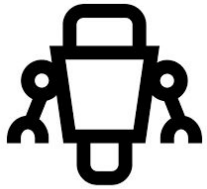


AI Monitor Agent

FAILURE PROBABILITY (LINE 6)

$$P_1 = \lim_{\alpha_1 \to 1} \sum_{1 \le i \le 2} \left(\sum_{1 \le j \le 2} x_{ij} \alpha_i \beta_j \right) = \sum_{1 \le j \le 2} x_{1j} \beta_j \qquad (1)$$

$$P_2 = \lim_{\alpha_2 \to 1} \sum_{1 \le i \le 2} \left(\sum_{1 \le j \le 2} x_{ij} \alpha_i \beta_j \right) = \sum_{1 \le j \le 2} x_{2j} \beta_j, \qquad (2)$$



AI Monitor Agent

$$Q(1) = P_1 = \sum_{1 \le j \le 2} x_{1j} \beta_j \qquad (3)$$

$$Q(2) = P_2 = \sum_{1 \le j \le 2} x_{2j} \beta_j \qquad (4)$$

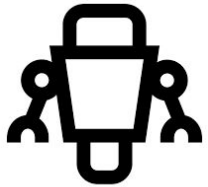
IMPORTANCE (LINE 7)

$$I_i = \frac{u_i - u_{min}}{u_{max} - u_{min}} * (max - min) + min, \qquad (5)$$

O artigo sugere uma abordagem de importância baseada no número de usuários, onde :

u_i é a quantidade de usuários do serviço i,
u_{min} é a quantidade mínima de usuários do serviço
u_{max} é a quantidade máxima
max é a importância máxima
min é a importância mínima

Os testes do trabalho foram realizados em um serviço, portanto essa função retorna um valor fixo, ou seja, o serviço tem sempre a mesma importância.



AI Monitor Agent

ACTION PROBABILITIES (LINE 8)

$$\alpha_1^{(k+1)} = I_i * (\alpha_1^{(k)} + \eta \frac{\partial P}{\partial \alpha_1^{(k)}}) \quad (8)$$

and

$$\alpha_2^{(k+1)} = (max - I_i) * (\alpha_2^{(k)} + \eta \frac{\partial P}{\partial \alpha_2^{(k)}}), \quad (9)$$

Os valores de Q podem ser utilizados para aproximar o valor da derivada parcial.

AI Monitor Agent

NORMALISE (LINE 9)

1 Suppose that there are m actions, i.e., $a_1, a_2, ..., a_m$;
2 Let $d = min_{1 \le k \le m} \pi(a_k)$, mapping center $c_0 = 0.5$, and mapping lower bound $L = 0.001$;
3 **if** $d < L$ **then**
4 $\quad \rho \leftarrow \frac{c_0 - L}{c_0 - d}$;
5 $\quad$ **for** $k = 1$ *to* m **do**
6 $\quad\quad \pi(a_k) \leftarrow c_0 - \rho \cdot (c_0 - \pi(a_k))$;
7 **for** $k = 1$ *to* m **do**
8 $\quad r \leftarrow \sum_{1 \le k \le m} \pi(a_k)$;
9 $\quad \pi(a_k) \leftarrow \frac{\pi(a_k)}{r}$;
10 **return** π;

AI Monitor Agent

Normaliza os valores de probabilidades das ações para que a soma seja igual a 1.

VALORES UTILIZADOS

Table 3. Parameters Setting

Parameters	Values	Explanations
r, c, p, b	5, 1, 6, 2	reward parameters
ξ	0.8	Learning rate
γ	0.65	Discount factor
η	0.0001	Gradient step size
max	1	Upper bound of importance
min	0	Lower bound of importance

AI Monitor Agent

Mesmos valores utilizados pelos autores do artigo.

TEST AGENT

Para um container aleatório a cada 5s até atingir a condição de parada, que é atingir um determinado valor na crença step(X).

Test Agent

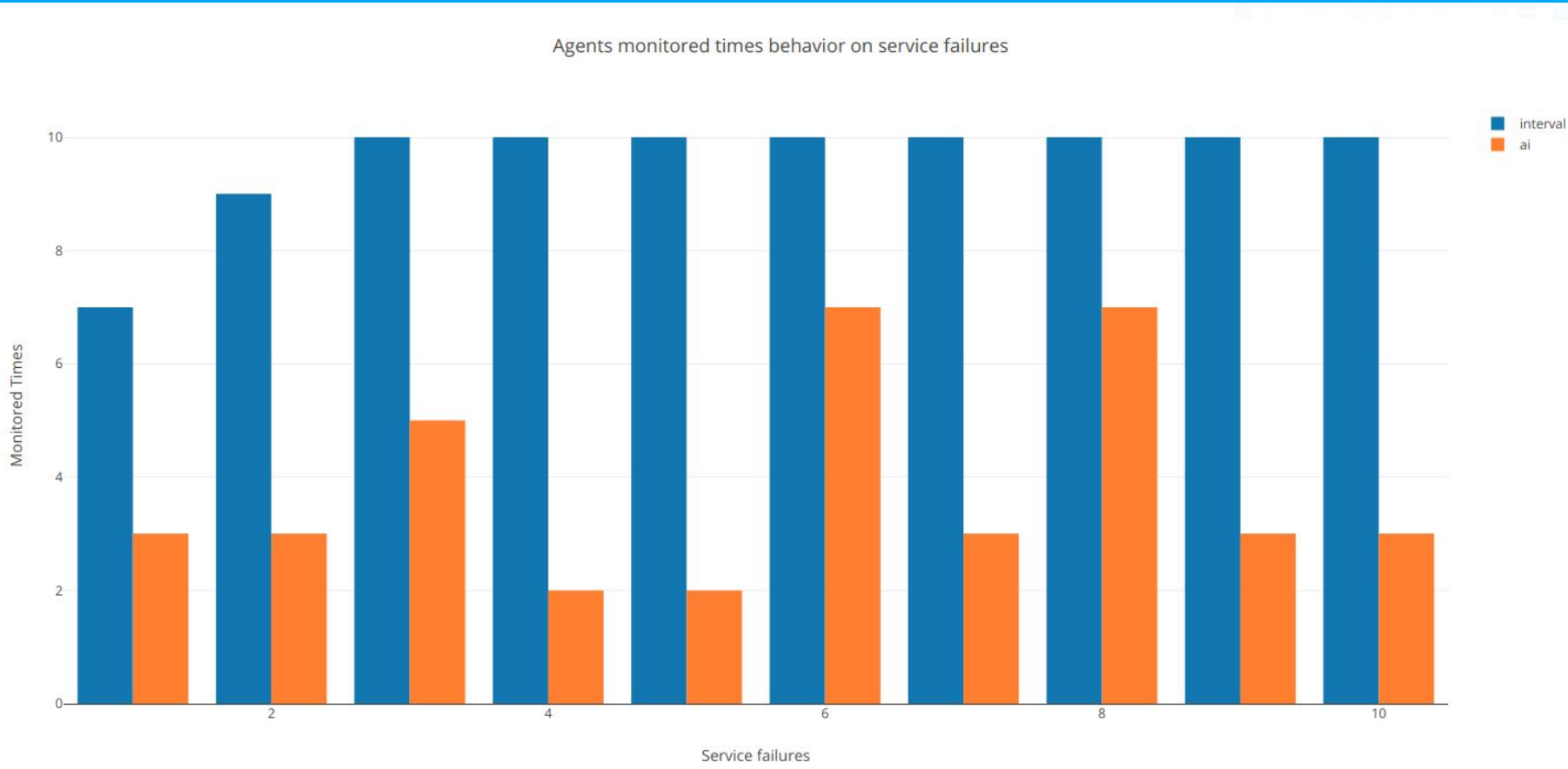
4.

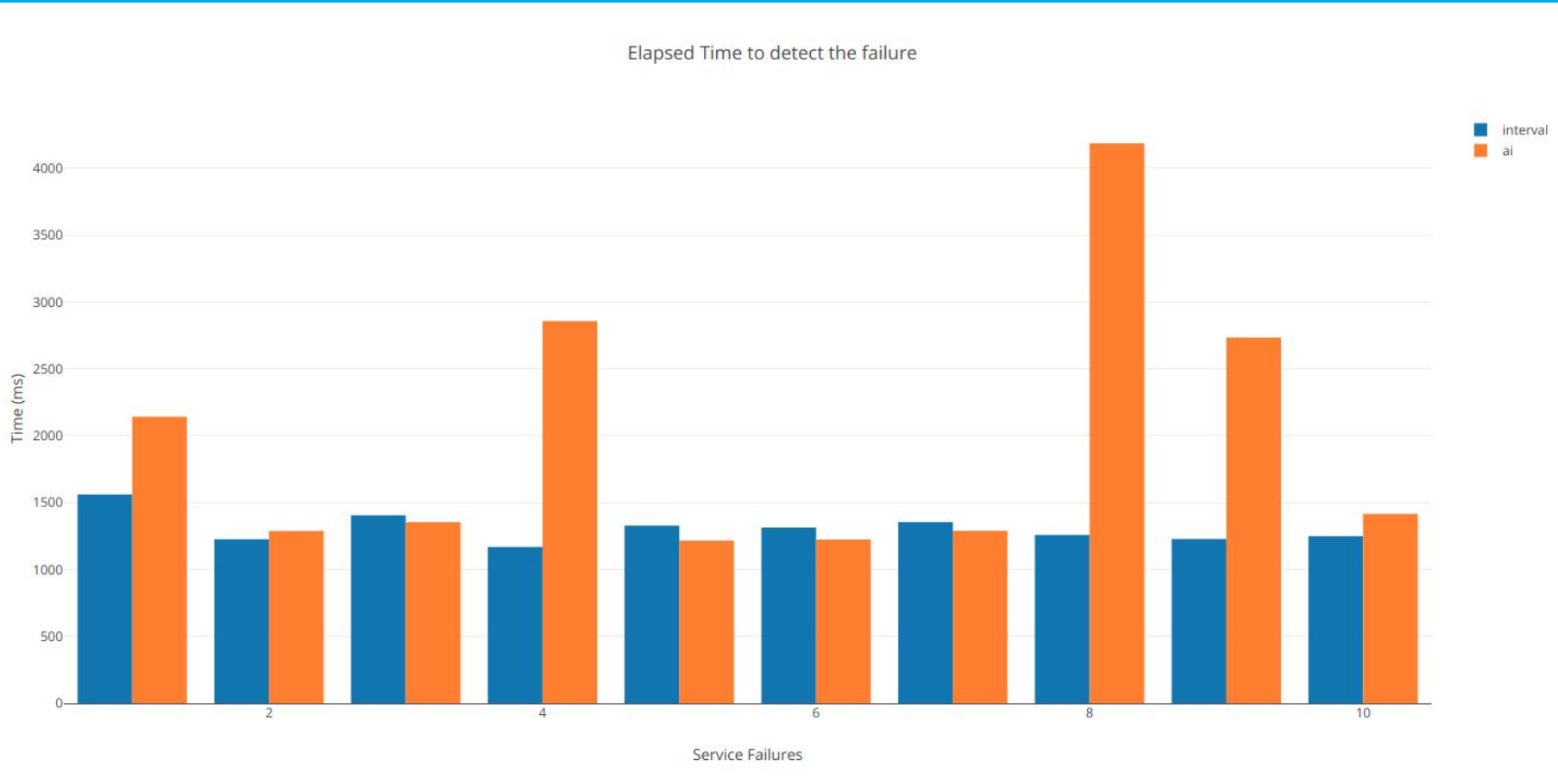
DEMO

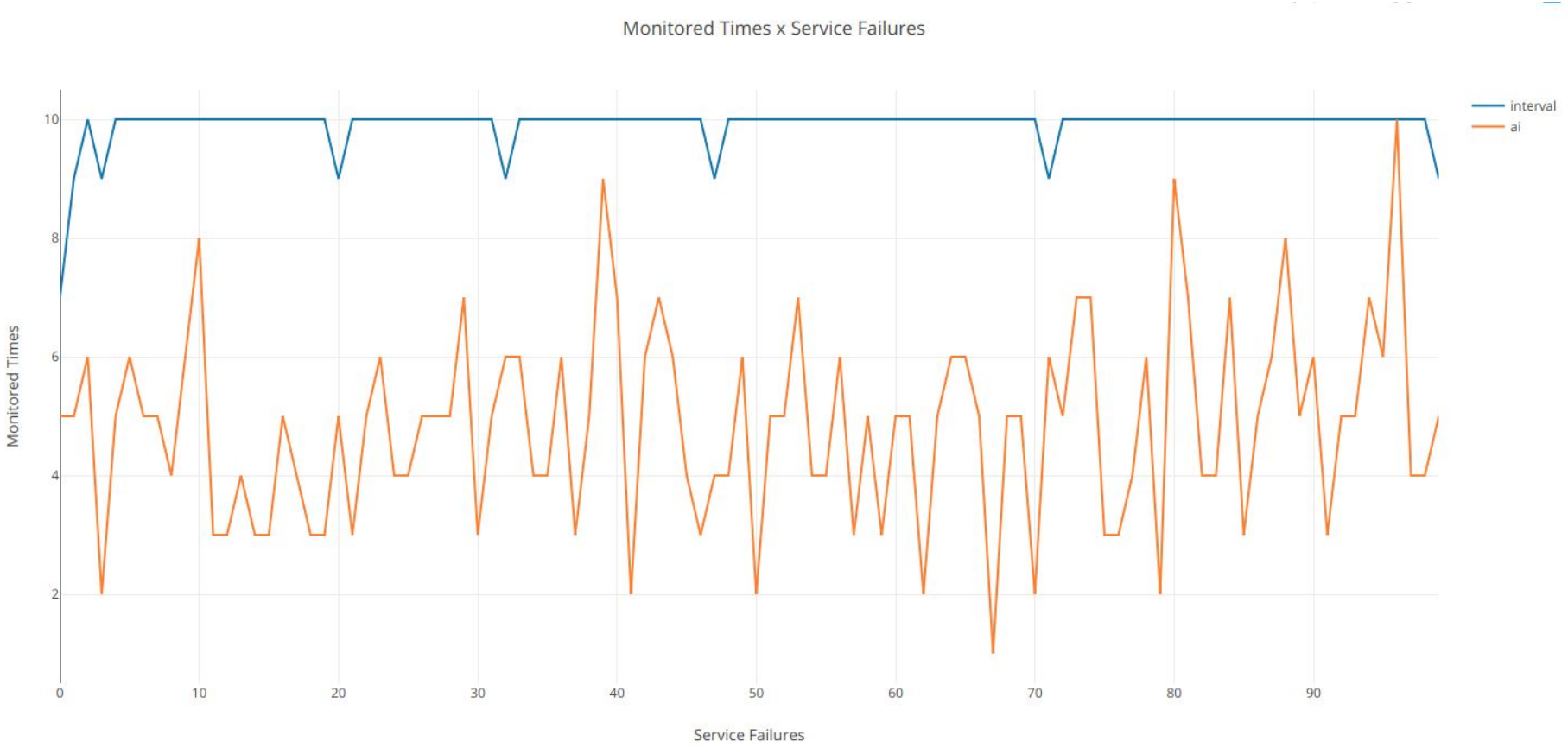

mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
elif operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True
#selection at the end -add back the deselected mirror modifier object
ror_ob.select= 1
ifier_ob.select=1
context.scene.objects.active = modifier_ob
("Selected" + str(modifier_ob)) # modifier ob is the active ob
mirror_ob.select = 0

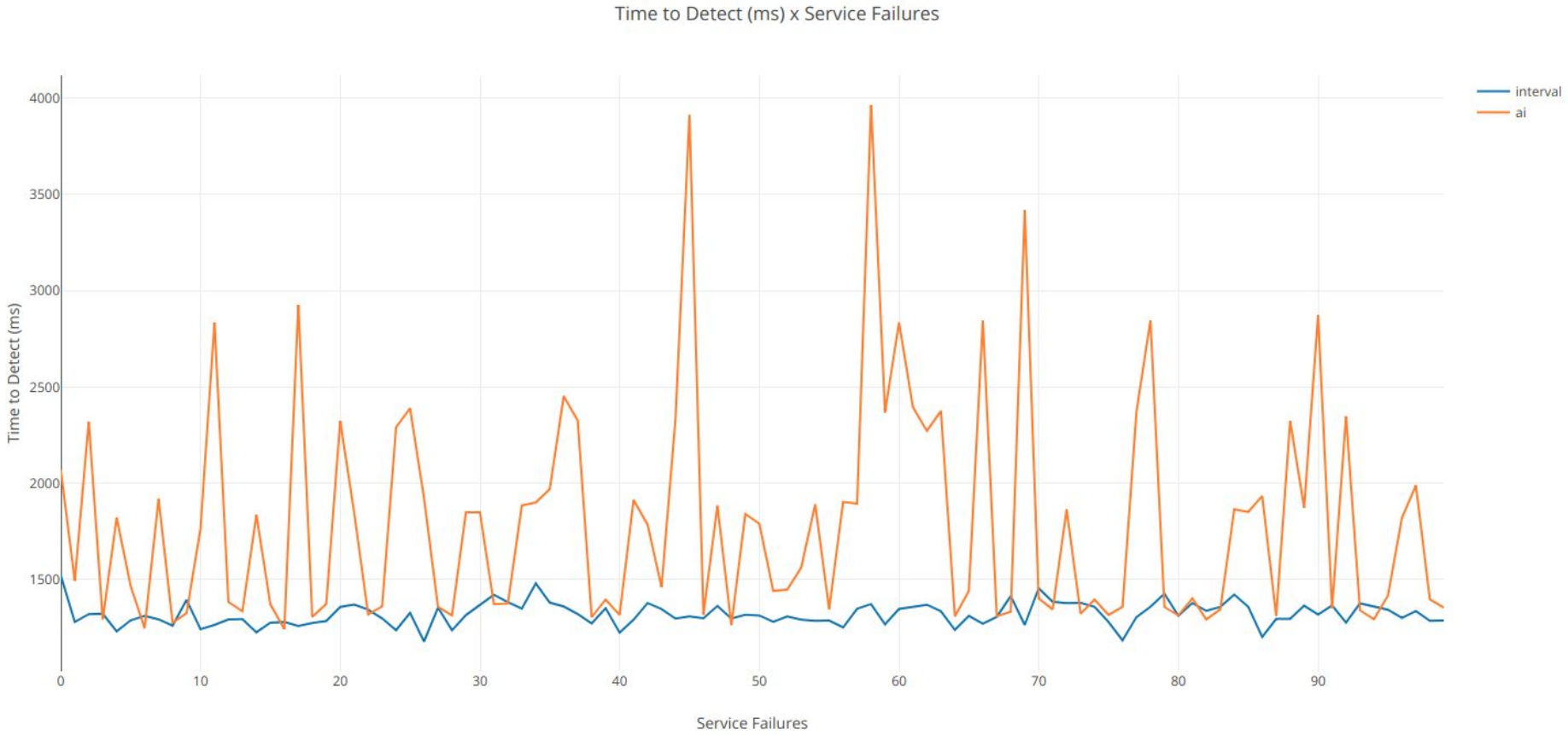
5.

RESULTADOS


Agents monitored times behavior on service failures
interval
ai
Monitored Times
0
2
4
6
8
10
2
4
6
8
10
Service failures


Elapsed Time to detect the failure
interval
ai
Time (ms)
4000
3500
3000
2500
2000
1500
1000
500
0
2
4
6
8
10
Service Failures


Monitored Times x Service Failures
interval
ai
Monitored Times
10
8
6
4
2
0
10
20
30
40
50
60
70
80
90
Service Failures


Time to Detect (ms) x Service Failures
Time to Detect (ms)
4000
3500
3000
2500
2000
1500
0
10
20
30
40
50
60
70
80
90
Service Failures
interval
ai

9,9x

Monitoramento médio do interval agent

4,84x

Monitoramento médio do ai agent

50%

De monitoramentos reduzidos aproximadamente.

1320,13ms

Tempo médio para detectar a falha do interval agent

1798,23 ms

Tempo médio para detectar a falha do ai agent

36%

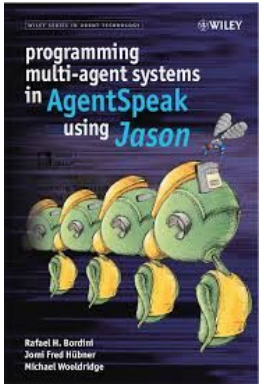
a mais de tempo para detectar aproximadamente.

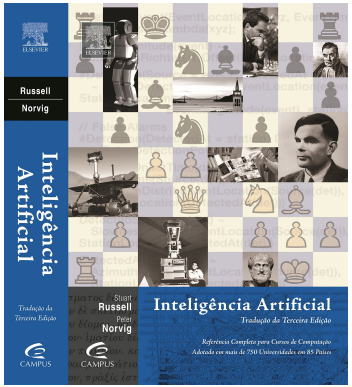
TRABALHOS FUTUROS

- AI agent suportar múltiplos serviços
- Definir uma função de importância para os serviços
- Monitorar múltiplos hosts
- Suportar múltiplos containers de um mesmo serviço

REFERÊNCIAS

- Amazon - What are containers ?
- Opensource.com - What is Docker ?
- Docker - Why Docker ?
- Dayong Ye, Qiang He, Yanchun Wang, Yun Yang - Detection of transmissible service failure in distributed service-based systems
- BORDINI, R. H., HUBNER, J. F., and WOOLDRIDGE, M. Programming Multi-Agent Systems in AgentSpeak Using Jason. John Wiley and Sons Ltd, October 2007.
- Cartago by Examples
- Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Norvig
- Spotify Docker Client
- Template da apresentação SlidesCarnival





container-monitor-agent

OBRIGADO!

Perguntas?