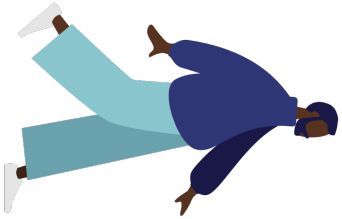


Escute o que a sua aplicação fala!

Rômulo Farias

romulets.github.io

Quem sou eu?

SIEMENS

Programador, 8+ anos

Curitiboca

Esporte, música, viagens

Contabilizei.

bol.com

Onde estou?

bol.com

2,200
Colleagues

11,000,000
Customers

25,000,000
Products

2,500,000
Visitors per day

***8** product domains, **47** products, **1017** applications & **159** teams*

Programando para enviar entre **1 e 4 milhões** de notificações todos os dias no maior e-commerce da Benelux!

Onde estou?

- Codando em **Java e Kotlin**
- Armazenando em **PostgreSQL**
- Rondando no **Kubernetes**
- Observando com **Prometheus, Grafana and Kibana**
- E outros recursos do **Google Cloud**

Heh? “escutar o que a sua aplicação fala”?

O que é “escutar o que a sua aplicação fala”?

Coletar, **processar**, **agregar** e **apresentar** dados em tempo real sobre seu sistema

O que é “escutar o que a sua aplicação fala”?

- **Quantidade** de acessos
- **Quais** acessos
- **Taxa** de erros
- **Uso** de CPU
- **Tempo** de processamento
- E por aí vai!

O que é “escutar o que a sua aplicação fala”?

Monitorar

Tá, e por que quero monitorar? Tá funcionando ué.

E por que eu quero monitorar?

Para confirmar que você está **entregando o melhor** para o consumidor final

E por que eu quero monitorar?

Para **alertar** o time de que a aplicação não está funcionando bem

Ninguém deve ser pago para assistir tela. Coloque tecnologia para trabalhar!

Se não processarmos nenhuma mensagem (0 mensagens) em 10 minutos, e-mail para o time!
10 K
9 K
8 K
7 K
6 K
5 K
4 K
3 K
2 K
1 K
0
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
02:00
03:00
04:00
05:00
06:00
07:00
08:00
09:00
10:00
11:00
12:00
13:00
14:00
15:00

E por que eu quero monitorar?

Para **identificar** sintomas e causas de problemas

Seu sistema está respondendo em 5 segundos (sintoma). Qual a causa?

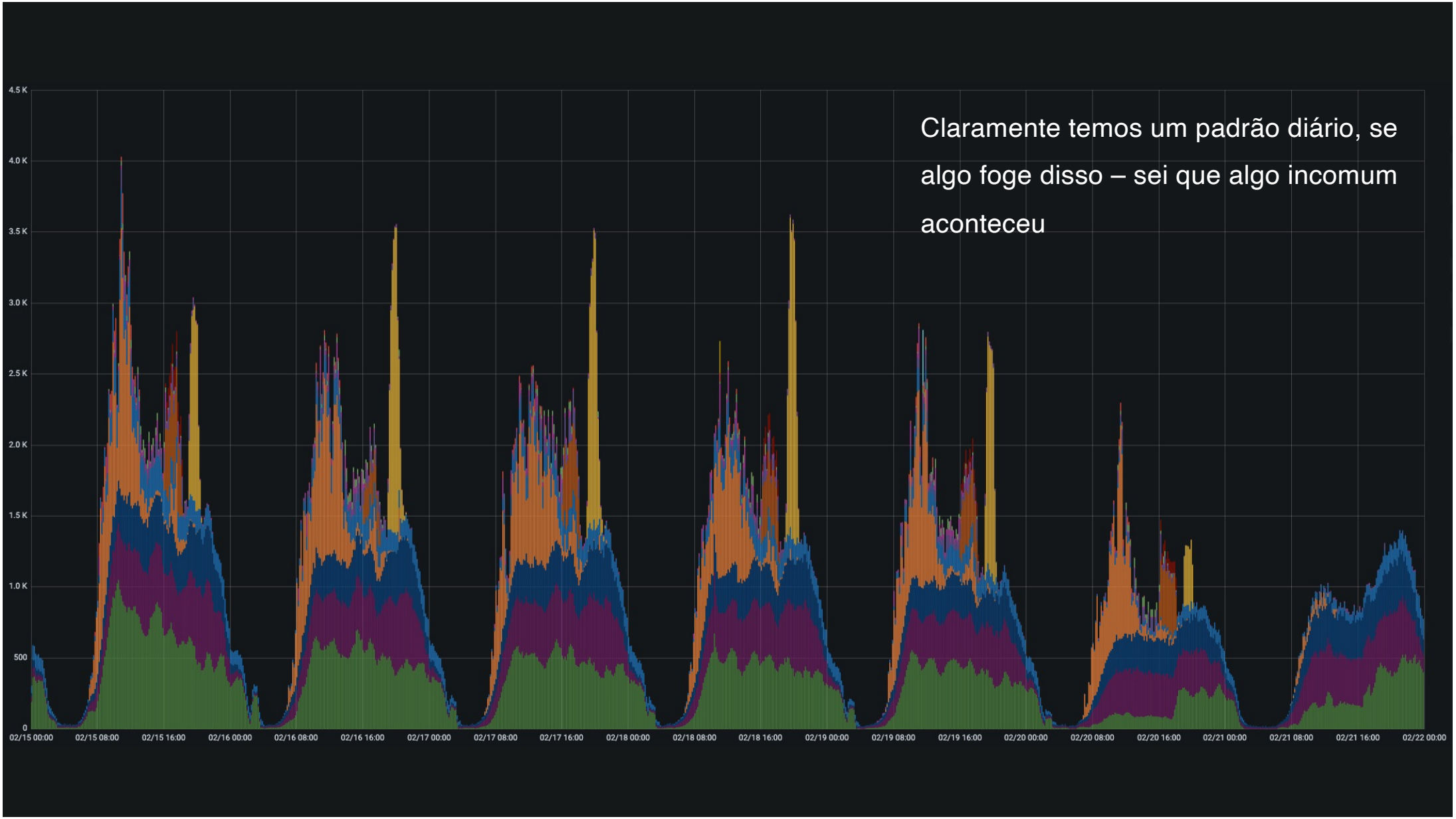
E por que eu quero monitorar?

Para construir **dashboards** que respondem rápidamente questões sobre a saúde da sua aplicação

E é bonito de ver!

E por que eu quero monitorar?

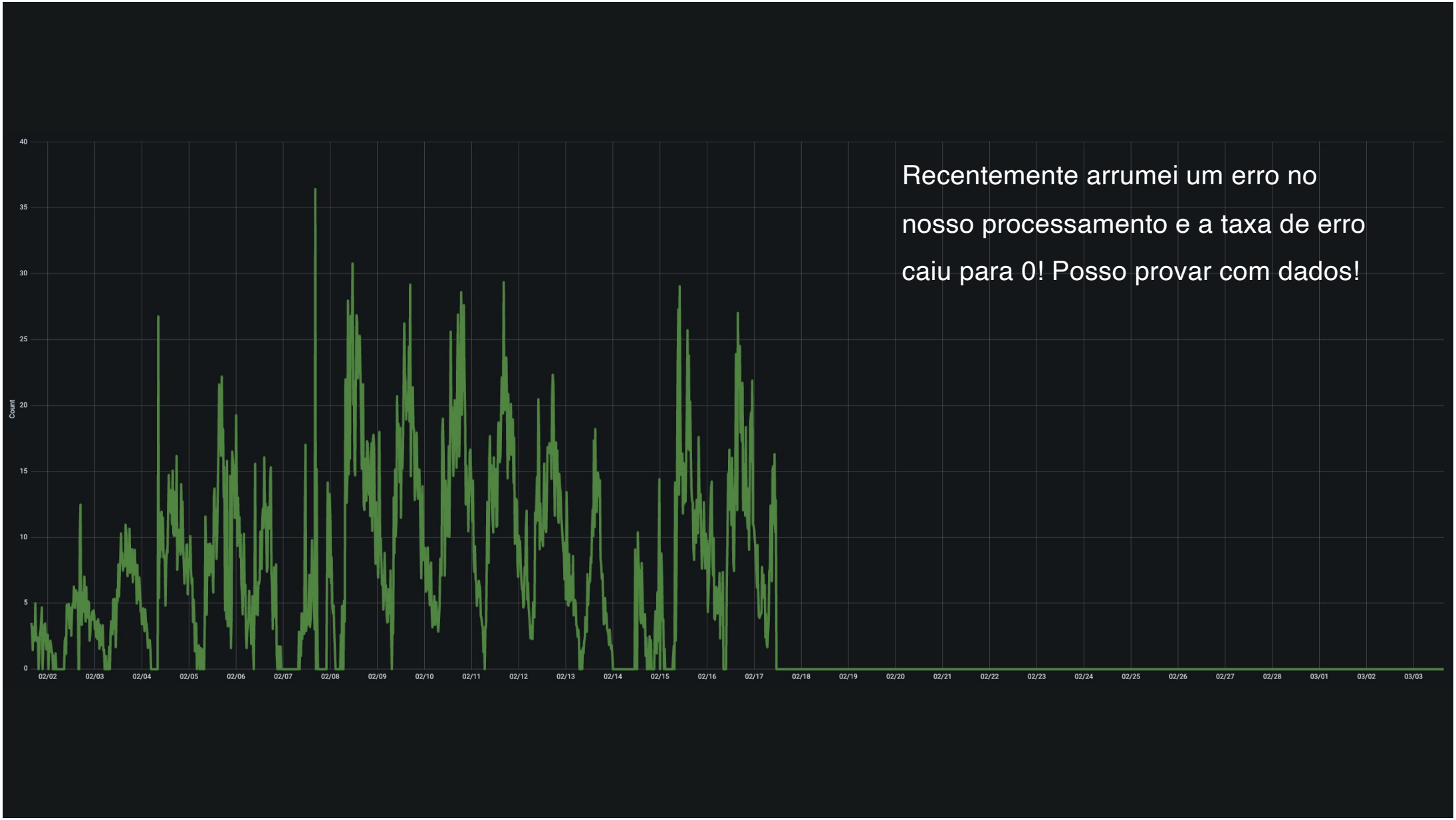
Para analisar **padrões** e **tendências** ao longo do tempo


Claramente temos um padrão diário, se algo foge disso – sei que algo incomum aconteceu
4.5 K
4.0 K
3.5 K
3.0 K
2.5 K
2.0 K
1.5 K
1.0 K
500
0
02/15 00:00
02/15 08:00
02/15 16:00
02/16 00:00
02/16 08:00
02/16 16:00
02/17 00:00
02/17 08:00
02/17 16:00
02/18 00:00
02/18 08:00
02/18 16:00
02/19 00:00
02/19 08:00
02/19 16:00
02/20 00:00
02/20 08:00
02/20 16:00
02/21 00:00
02/21 08:00
02/21 16:00
02/22 00:00

E por que eu quero monitorar?

Para **comparar** períodos e experimentos

Qual foi o impacto no uso do meus sitema depois de entregar a funcionalidade X?


Recentemente arrumei um erro no nosso processamento e a taxa de erro caiu para 0! Posso provar com dados!
Count
40
35
30
25
20
15
10
5
0
02/02
02/03
02/04
02/05
02/06
02/07
02/08
02/09
02/10
02/11
02/12
02/13
02/14
02/15
02/16
02/17
02/18
02/19
02/20
02/21
02/22
02/23
02/24
02/25
02/26
02/27
02/28
03/01
03/02
03/03

E por que eu quero monitorar?

Para **aprender** sobre falhas acontecidas

Ao entender os gráficos plotados durante uma falha é mais fácil de compreender o problema

Que massa! Vou começar a monitorar!

Mas o que monitorar?

O que monitorar?

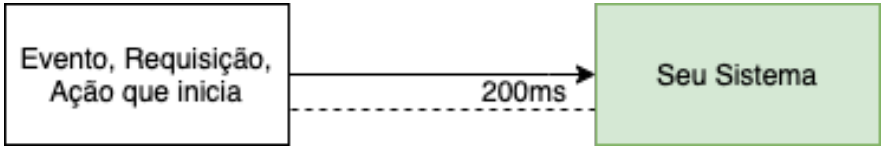
Depende do **contexto**.

Processo **interativo** e **incremental**

O Livro SRE recomenda

“Os 4 Sinais de Ouro”

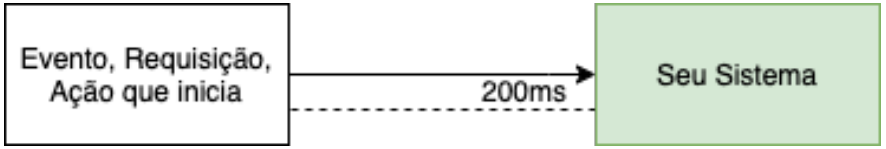
[Site Reliability Engineering: How Google Runs Production Systems, chapter 6]



1. Latência

Quanto tempo seu sistema

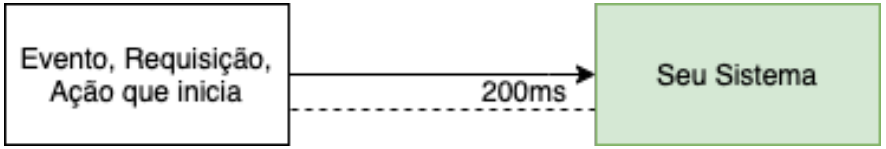
leva para processar algo.



1. Latência

Demonstra como o sistema lida com a carga atual

Possibilidade de comparar o desempenho entre períodos

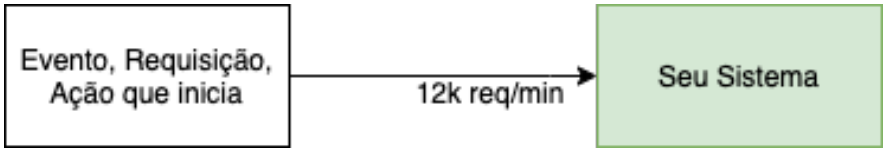


1. Latência

Diferencie a latência por tipos de resposta (sucesso e falhas)

Falhas geralmente possuem tempo de processamento diferentes de sucesso.

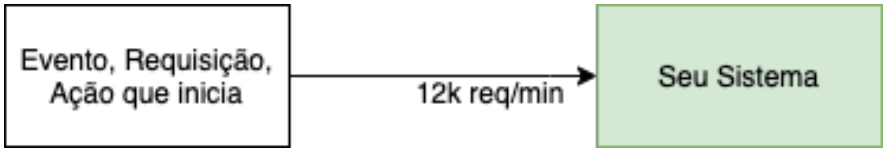
Falhas longas são piores que falhas rápidas



2. Tráfego

A carga que seu sistema

está recebendo



2. Tráfego

Demonstra qual o cenário que seu sistema está sendo submetido

O que monitorar?

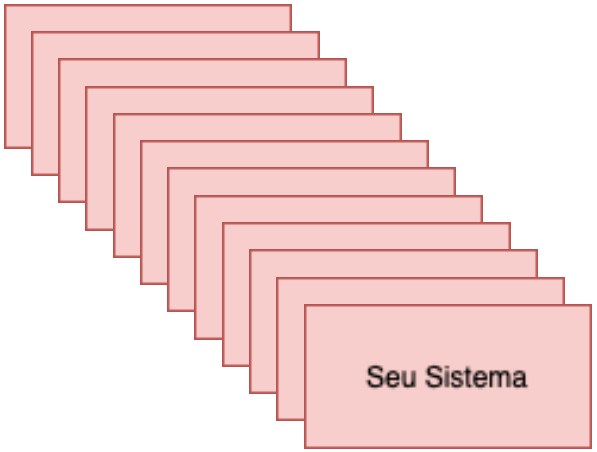
3. Erros

A quantidade de erros

de seu sistema.

Erro, problema, exceção, HTTP 500,

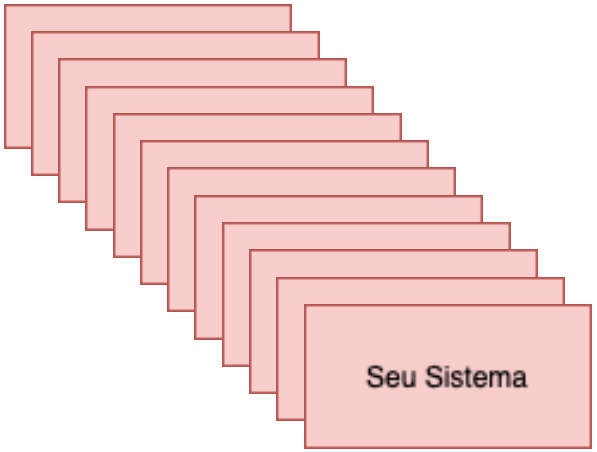
HTTP 200 { erro: true }

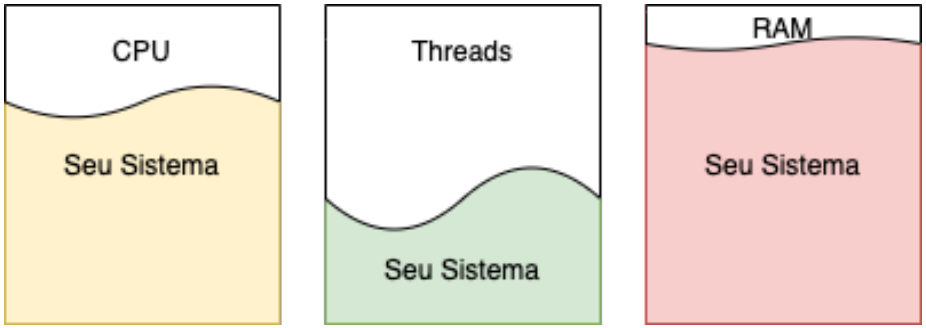


3. Erros

Erros não deveriam acontecer.

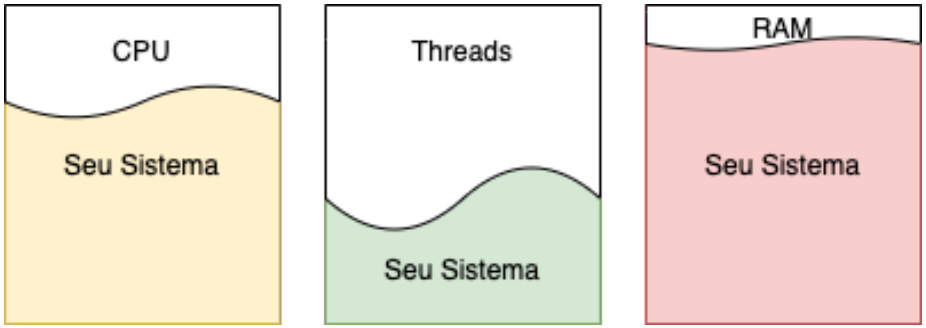
Erro? Ação necessária!





4. Saturação

O quão “cheio” o seu sistema está. CPU, RAM, I/O, Thread etc.



4. Saturação

Ajuda a entender fatores

“externos” à aplicação, e.g.

Garbage Colector

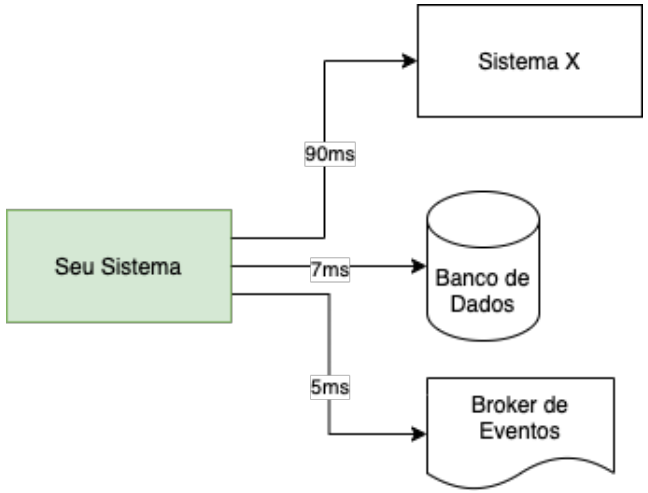
E eu tenho um

5º Sinal de Ouro

pessoal!

5. Latência das Dependências

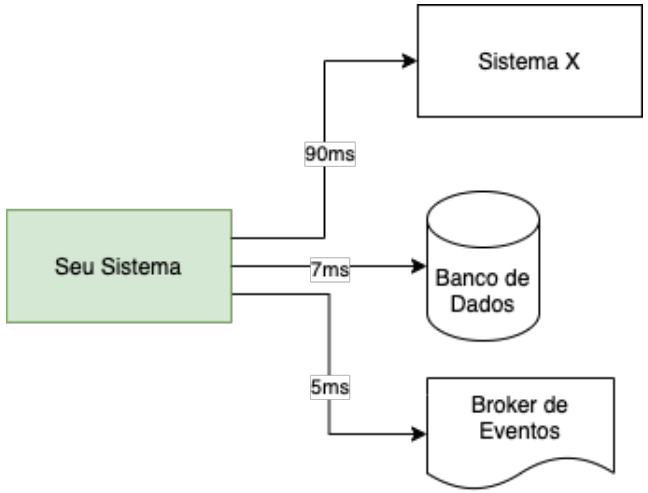
Tempo que depências levam para responder, por sucesso e falhas



O que monitorar?

5. Latência das Dependências

Ajuda a entender onde está problema



E **seja crítico**!

Leve seu contexto em consideração.

Tá e na vida real, como que monitoro?

Tá e na vida real, como que monitoro?

Eu, como desenvolvedor, não sou reponsável pela infrastrutura, mas pelas **métricas** que exponho e os **alertas** que crio

Tá e na vida real, como que monitoro?

Eu utilizo **Prometheus** para métricas.

Grafana para criar dashboards

Tá e na vida real, como que monitoro?

Tento utilizar o **máximo de métricas disponibilizadas** pelas ferramentas que uso

kube_daemonset_status_number_available
kube_daemonset_status_number_misscheduled
kube_daemonset_status_number_ready
kube_daemonset_status_number_unavailable
kube_daemonset_updated_number_scheduled
kube_deployment_created
kube_deployment_labels
kube_deployment_metadata_generation
kube_deployment_spec_paused
kube_deployment_spec_replicas
kube_deployment_spec_strategy_rollingupdate_max_surge
kube_deployment_spec_strategy_rollingupdate_max_unavailabl
kube_deployment_status_condition
kube_deployment_status_observed_generation
kube_deployment_status_replicas
kube_deployment_status_replicas_available
kube_deployment_status_replicas_unavailable
kube_deployment_status_replicas_updated
kube_endpoint_address_available
kube_endpoint_address_not_ready
kube_endpoint_created
kube_endpoint_info
kube_endpoint_labels
kube_hpa_labels
kube_hpa_metadata_generation
kube_hpa_spec_max_replicas
kube_hpa_spec_min_replicas
kube_hpa_spec_target_metric
kube_hpa_status_condition
kube_hpa_status_current_replicas
kube_hpa_status_desired_replicas
kube_ingress_created
kube_ingress_info

Tá e na vida real, como que monitoro?

Exponho em nível de aplicação

utilizando a biblioteca **Micrometer**

(*micrometer.io*)

```java
@GetMapping("/monitoramento/exemplo")
@Timed(value =  "app.contexto.nomeDaMetrica",
        percentiles = { 0.9, 0.99, 0.999 },
        histogram = true)
public ResponseEntity<String> exemploDeMonitoramento() {
    return ResponseEntity.ok("Tudo o que aconteceu nesse endpoint foi cronometrado!");
}
```

```java
Counter.builder( name: "app.contexto.nomeDaMetrica")
        .tag("labelDaMetrica", "SUCESSO")
        .register(meterRegistry)
        .increment();
```

Tá e na vida real, como que monitoro?

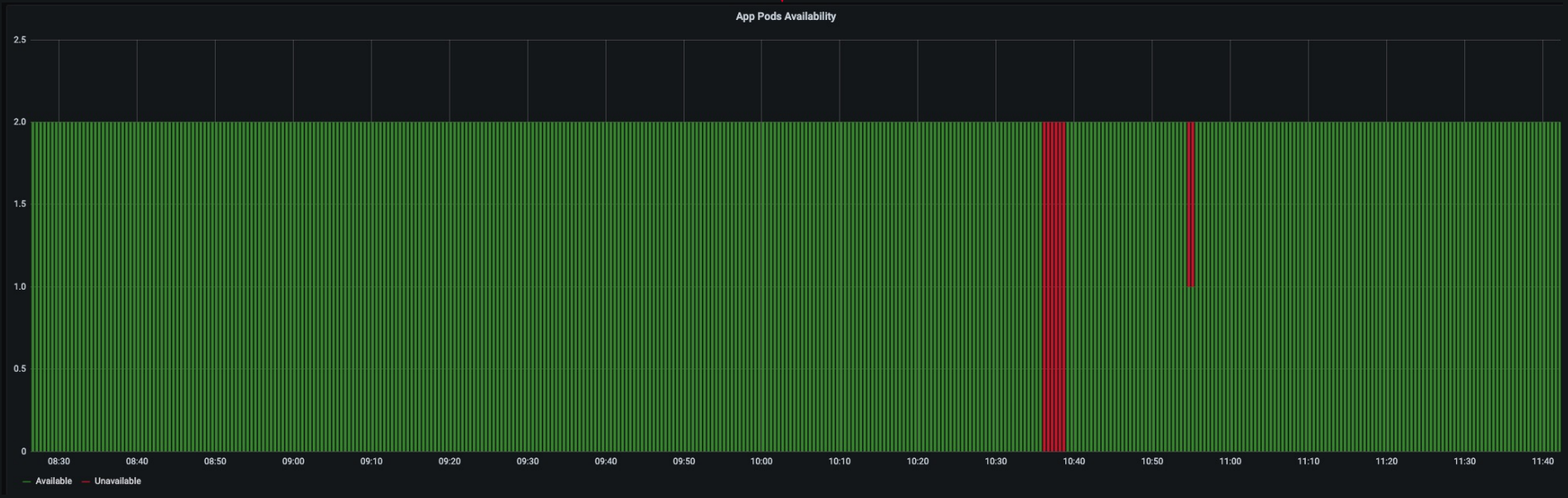
Utilizo muito **histogramas**, evito gráficos que mostram apenas o estado atual

---

Algo estranho pode ter acontecido 1 hora atrás e o seu dashboard não irá te mostrar.

App Pods Availability

2.0



Tá e na vida real, como que monitoro?

Procuro monitorar o “tail” da métrica.

Gráficos baseados em **média**

enganam

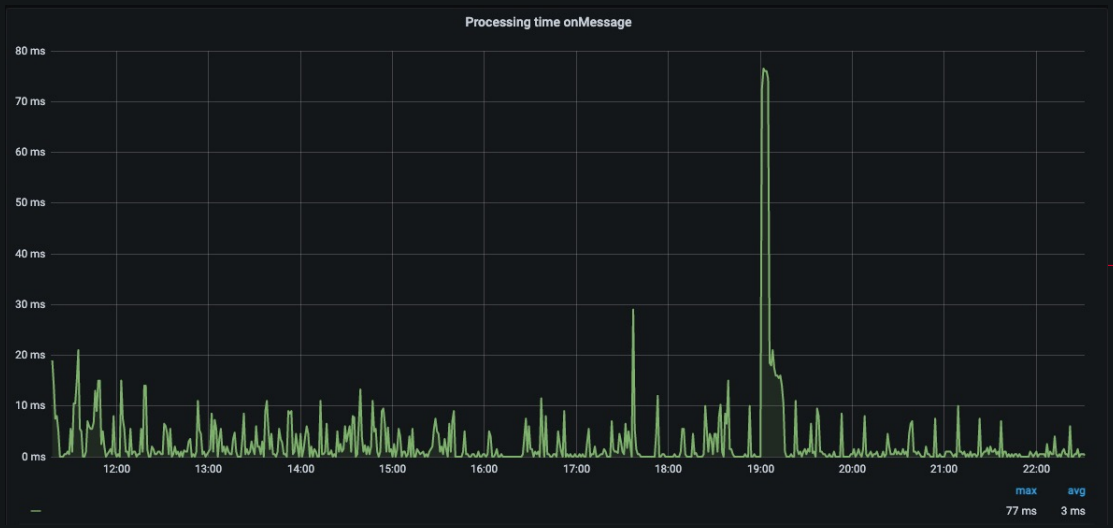
---

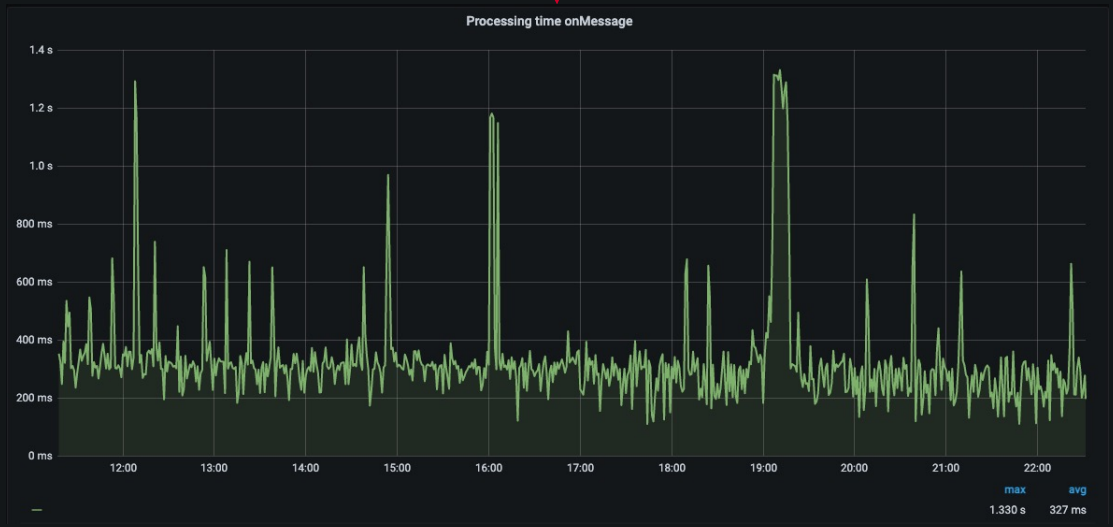
1 minuto, 1k requisições.

900req - 100ms

100req - 5000ms

$$\frac{(900req \ * 100ms) + (\ 100req \ * 5000ms)}{1000req} = 590ms$$


Processing time onMessage
80 ms
70 ms
60 ms
50 ms
40 ms
30 ms
20 ms
10 ms
0 ms
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
max
avg
77 ms
3 ms


Processing time onMessage
1.4 s
1.2 s
1.0 s
800 ms
600 ms
400 ms
200 ms
0 ms
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
max
avg
1.330 s
327 ms

Tá e na vida real, como que monitoro?

Monitoramento faz **parte do processo de desenvolvimento (DoD)**.

Tá e na vida real, como que monitoro?

Meus alertas são **versionados**

```yaml
- alert: replicas-nao-disponiveis
  # Se meu app possuir mais de 0 (zero) réplicas não disponíveis
  expr: sum(kube_deployment_status_replicas_unvailable{deployment="meu-app"}) > 0
  # Por dois minutos
  for: 2m
  # Alerte no canal
  labels:
    channel: "email_do_time@empresa.com"
  # Com o essa descrição
  annotations:
    description: "Aplicação tem réplicas não disponíveis"
```

Tá e na vida real, como que monitoro?

Mantenho meus dashboards e regras de alerta **simples**.

Um dashboard tem que responder perguntas, e não criar

Cuido de meus alertas de forma crítica.

Falsos Alertas podem ser tão prejudiciais quanto a falta de monitoramento e alertas

Isso é o que eu faço. Adapte para sua realidade!

Mais sobre o assunto?

Mais sobre o assuto?

Foco: **colocar aplicações de qualidade em produção**

---

sre.google/books

Mais sobre o assuto?

Busque tecnologias do mercado. Como **Prometheus**, **DataDog**, **New Relic** entre outros

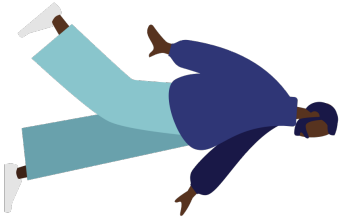
Mais sobre o assuto?

Discuta com seu time!

Curtiu minha stack e o que estou mostrando?

Chega pra ver mais de perto!

careers.bol.com



Muito obrigado, e espero poder ter contribuido de alguma forma!

romulets.github.io

Twitter: @romulets