

Introdução sobre observabilidade

em sistemas distribuídos

Aluno: Reinan Gabriel Dos Santos Souza
Bacharelado em Sistemas de Informação
Instituto Federal de Sergipe
Campus Lagarto



Reinan Gabriel Dos Santos Souza

Sou um entusiasta de tecnologia apaixonado por aprender e explorar novas áreas.

Desenvolvedor desde 2018
Engenheiro DevOps Pleno – **MOVA**

<https://linktr.ee/reinanhs>



Arquivos para a apresentação

Todos os materiais relacionados à apresentação do minicurso estão disponíveis digitalmente no meu repositório do **Gitlab**. Para acessar esses recursos, basta escanear o **QR Code** na imagem ao lado.



Gitlab: <https://gitlab.com/snct-lagarto-2023>

Github: <https://github.com/ReinanHS>

À medida que os sistemas distribuídos se tornaram mais populares, os desenvolvedores enfrentam um desafio crescente:

- Como solucionar problemas de software, considerando que ele pode ser composto por diversos serviços distintos, cada um executado em diferentes linguagens e plataformas?
- Como detectar mudanças críticas que possam impactar todos os nossos microsserviços?
- Como identificar as raízes reais dos erros?

Não muito tempo atrás, depurar um programa geralmente significava uma coisa: **Navegar nos logs de erros**



SlackLog APP 4:23 PM

Debug Event

Exception: Hello Error Message in

/home/vagrant/Code/test/blog_laravel_55/app/Http/Controllers/SlackController.php:18

Stack trace:

#0 [internal function]: App\Http\Controllers\SlackController->sendSlackMessage()

#1

/home/vagrant/Code/test/blog_laravel_55/vendor/laravel/framework/src/Illuminate/Routing/Controller.php(54): call_user_func_array(Array, Array)

#2

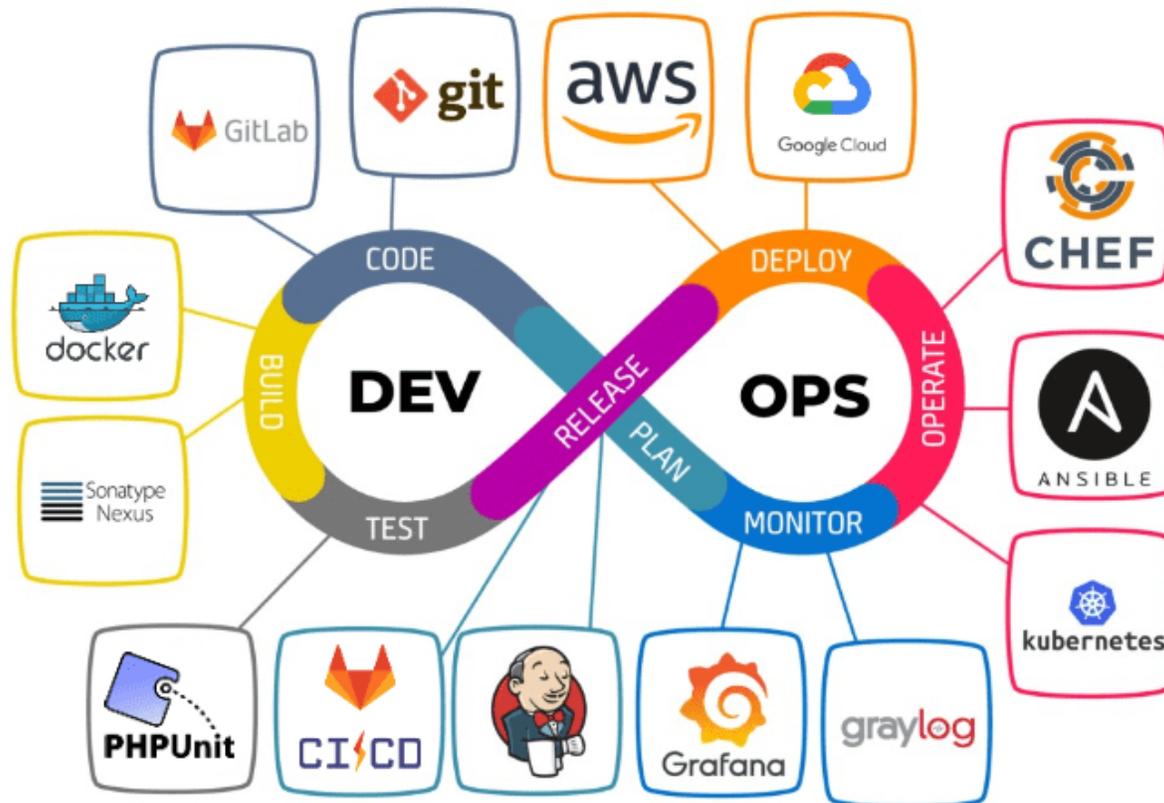
/home/vagrant/Code/test/blog_laravel_55/vendor/laravel/framework/src/Illuminate/Routing/ControllerDispatcher.php(45): Illuminate\Routing\Controller->callAction('sendSlackMessag...', Array)

#3

Mas as coisas em TI estão sempre mudando. Os paradigmas de arquitetura de software evoluíram de monólitos para microsserviços.



Com o surgimento do **DevOps**, os desenvolvedores agora têm novas responsabilidades. Isso significa que eles cuidam dos programas mesmo depois de serem entregues.



Por que a observabilidade é necessária?

A observabilidade é necessária para monitorar, detectar e resolver problemas em sistemas complexos, garantindo sua confiabilidade e desempenho.

Importância da observabilidade para o sucesso empresarial

A **rapidez**, a **confiabilidade** e o **funcionamento** contínuo do software afetam o desempenho da empresa, o que inclui o **lucro**, o trabalho eficiente e a felicidade dos clientes. É essencial garantir que todas as solicitações de aplicativos sejam processadas de forma rápida e precisa.



Por que o mundo precisa de observabilidade?

A observabilidade é ainda mais importante agora para as empresas por algumas razões:

- Sistemas inteligentes em ascensão
- Demanda por acesso aos dados
- Requisitos de conformidade e precaução

A tendência em alta no mercado atual

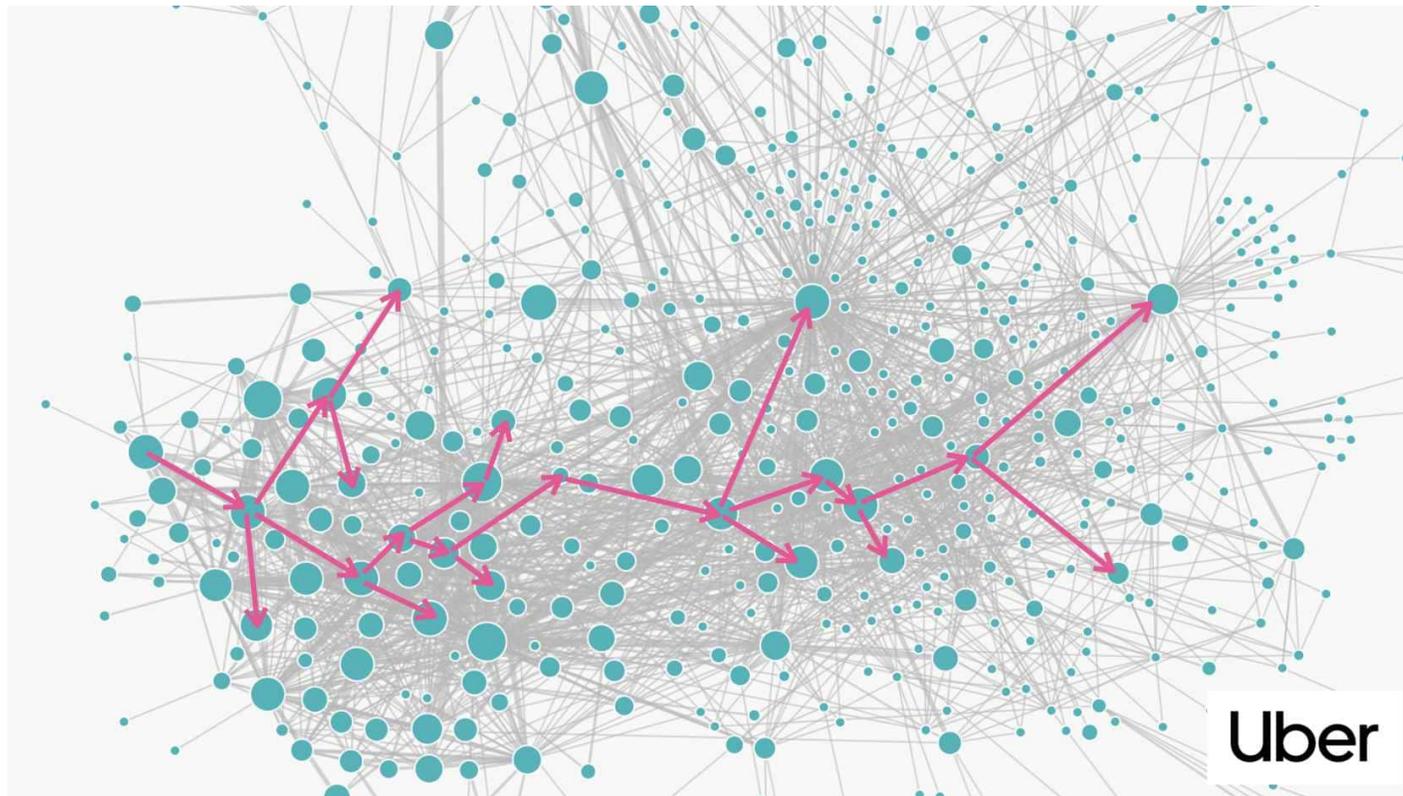
Observabilidade está se destacando como uma abordagem amplamente adotada e valorizada no mercado atual.

Interesse ao longo do tempo 



Complexidade de entender os serviços

A observabilidade é uma metodologia essencial para lidar com a **complexidade** dos **microsserviços**, que se tornaram uma tendência dominante no desenvolvimento de software.



Para **Leo Cavalcante** (2022): “Observabilidade é a metodologia que o mercado de microsserviços e cloud-native explora para monitorar as aplicações”

Leo Cavalcante. Observabilidade com Hyperf e OpenTelemetry. Medium, 22, mar, 2022. Disponível em: <https://medium.com/inside-picpay/observabilidade-com-hyperf-e-opentelemetry-dd698844eda>. Acesso em: 18 maio. 2023.

A origem da observabilidade

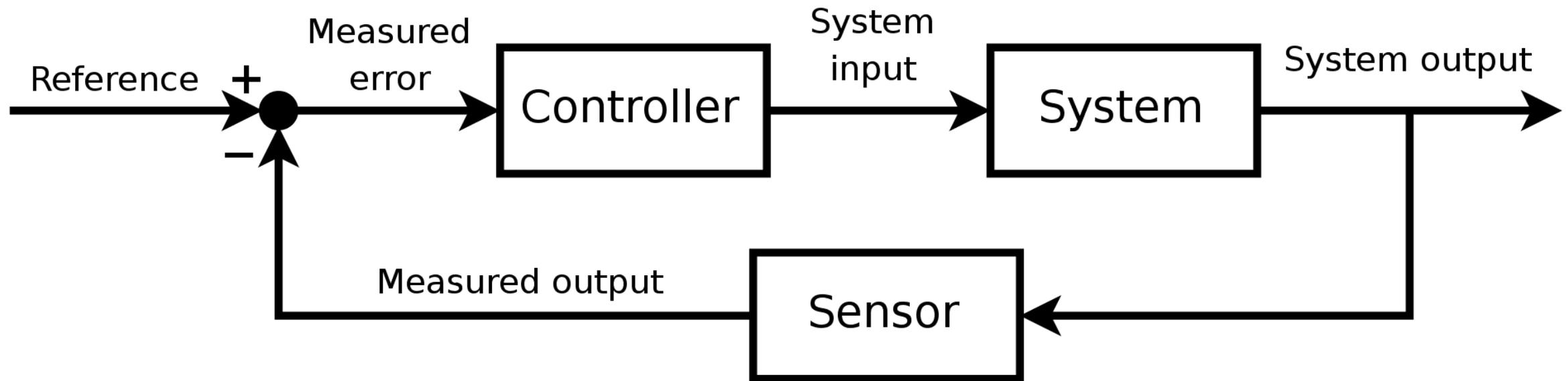
O termo “**observabilidade**” teve sua origem no trabalho do engenheiro americano Rudolf E. Kálmán em agosto de 1960, conforme mencionado em seu livro “**Teoria dos sistemas de controle**” (On the general theory of control systems).



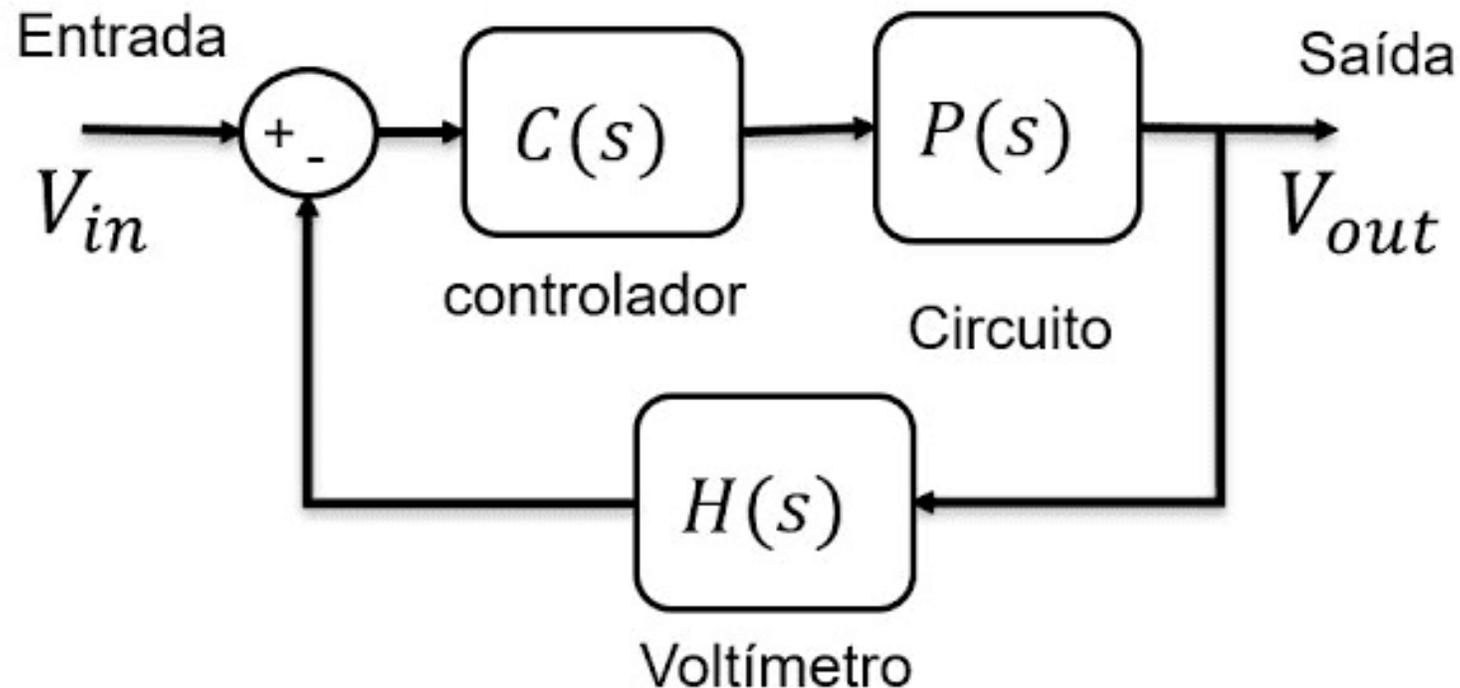
Link para o livro: <https://www.sciencedirect.com/science/article/pii/S1474667017700948>

Fonte para a imagem: https://pt.wikipedia.org/wiki/Rudolf_Kalman

Para **Rudolf E. Kálmán** a observabilidade é uma medida de quão bem os estados internos de um sistema podem ser inferidos a partir do conhecimento de suas saídas externas.



Em suma, **Rudolf E. Kálmán**, lançou as bases do conceito de observabilidade, estabelecendo sua importância na determinação do estado interno dos sistemas.



Surgimento das primeiras equipes de observabilidade em 2013

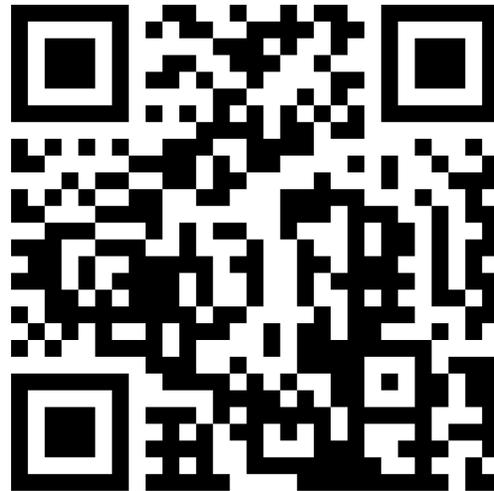
Em 2013, a mudança para sistemas distribuídos já estava acontecendo.

Nesse ano, o X (anteriormente conhecido como 'Twitter') formou uma nova equipe, chamada '*equipe de observabilidade*'.

O principal objetivo dessa equipe é coletar dados de telemetria de centenas de serviços do Twitter de forma centralizada e padronizada.

Publicação sobre observabilidade no Twitter

No final de 2013, a equipe de observabilidade do Twitter compartilhou um post no blog, descrevendo seus primeiros resultados ao usar a observabilidade na empresa. Eles também incluíram uma imagem mostrando o rastreamento de seus sistemas distribuídos.





Fonte da imagem: https://blog.twitter.com/engineering/en_us/a/2013/observability-at-twitter

Distributed Systems Tracing with Zipkin: https://blog.twitter.com/engineering/en_us/a/2012/distributed-systems-tracing-with-zipkin

Publicação sobre abordagem da observabilidade em 2016

O Twitter desempenhou um grande papel em popularizar o conceito de observabilidade na computação.

Em março de 2016, a equipe de engenharia de observabilidade do Twitter compartilhou o post [Observability at Twitter: technical overview](#) no blog explicando sua abordagem à observabilidade.

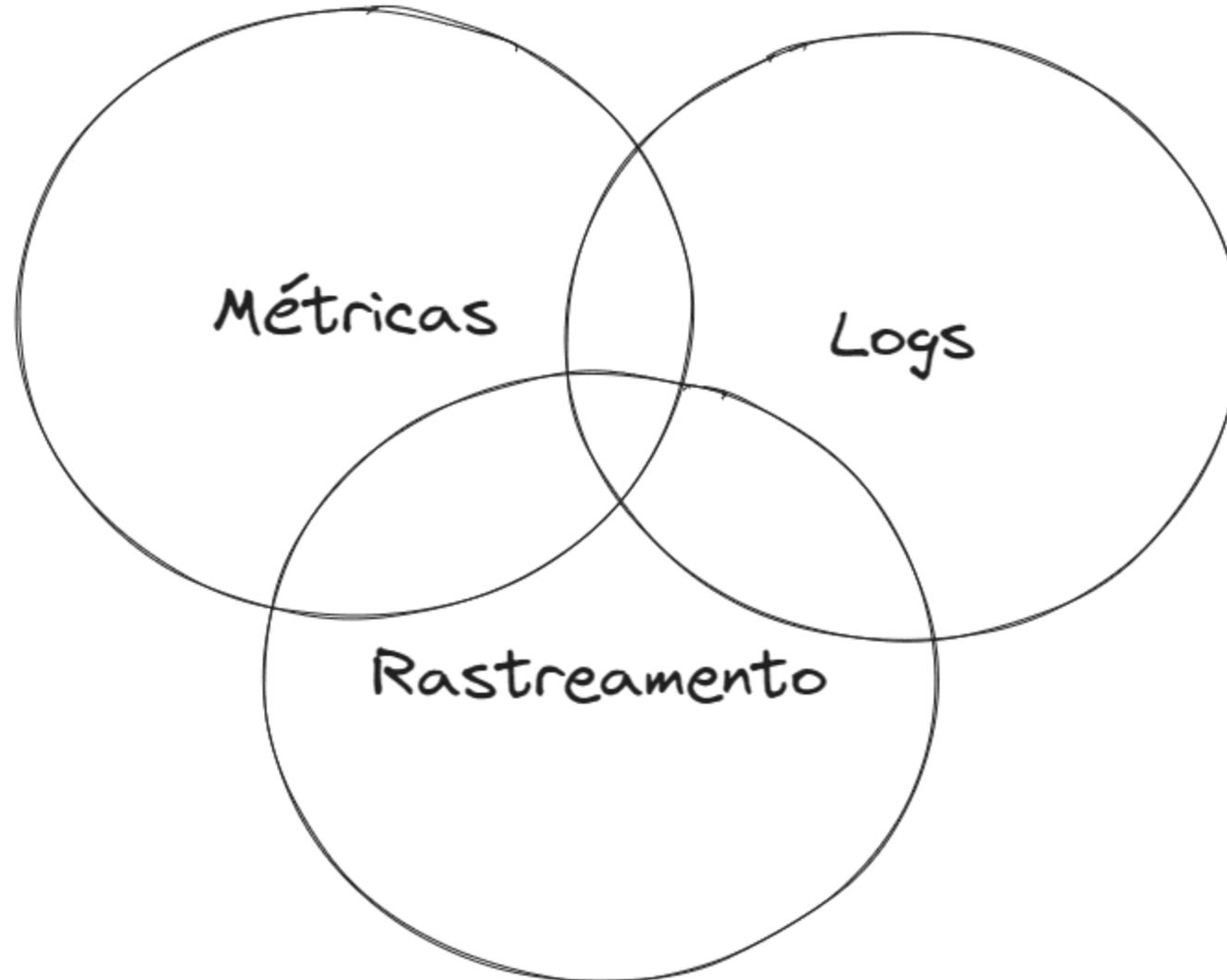


Observabilidade hoje

Hoje, a observabilidade é essencial para lidar com aplicativos complexos. Desenvolvedores, engenheiros de TI e equipes de DevOps não podem viver sem ela.

Princípios da observabilidade

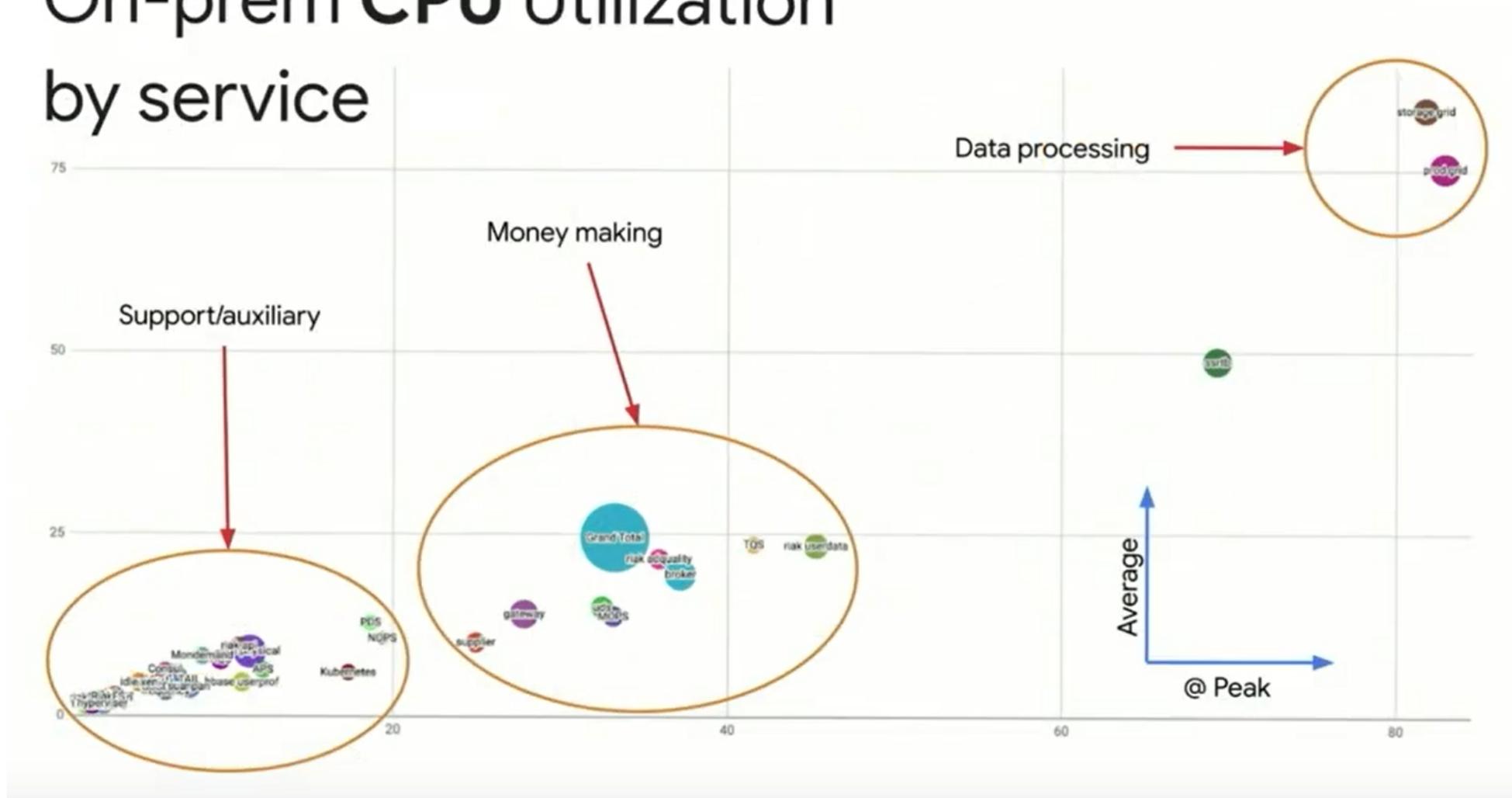
A observabilidade se fundamenta em quatro pilares-chave:



As métricas fornecem medidas quantitativas do desempenho e do comportamento dos sistemas.

- Monitoramento de tendências
- Identificação de padrões
- Base para tomada de decisões
- Otimização e escalonamento

On-prem **CPU** Utilization by service



RED Metrics fornece aos desenvolvedores um modelo para instrumentar seus serviços e construir painéis de controle de forma consistente e repetível.

- **Rate:** O número de solicitações que o serviço está processando por segundo.
- **Error:** O número de solicitações com falha por segundo.
- **Duration:** A quantidade de tempo que cada solicitação leve.

Os logs são registros detalhados das atividades e eventos que ocorrem em um sistema.

- Ferramenta de depuração
- Registro de eventos relevantes
- Sequenciamento de eventos
- Contexto e causa de problemas

Ilustração de logs com uma organização deficiente

```
18-08-23 12:34:56 - INFO - Dados ausentes  
18-08-23 12:34:56 - Evento: INFO - Ocorreu algo inesperado no cadastro do produto.  
18-08-23 12:34:56 - ERRO: 500 - Problema com o sistema. Algo deu errado.
```

- Não identifica quais dados ou a causa da ausência.
- A falta de clareza na descrição do evento torna difícil entender o que realmente aconteceu.
- A mensagem de erro é vaga e não fornece informações úteis sobre o problema.

Dicas para melhorar a organização dos logs com pequenas

```
'default' => [  
    'handler' => [  
        'class' => Monolog\Handler\StreamHandler::class,  
        'constructor' => [  
            'stream' => 'php://stdout',  
            'messageType' => Monolog\Handler\ErrorLogHandler::OPERATING\_SYSTEM,  
            'level' => Monolog\Level::Info,  
        ],  
    ],  
    'formatter' => [  
        'class' => Monolog\Formatter\JsonFormatter::class,  
        'constructor' => [  
            'includeStacktraces' => true,  
        ],  
    ],  
    'PsrLogMessageProcessor' => [  
        'class' => Monolog\Processor\PsrLogMessageProcessor::class,  
    ],  
    'processors' => [  
        new Monolog\Processor\GitProcessor(),  
        new Monolog\Processor\MemoryUsageProcessor(),  
        new Monolog\Processor\ProcessIdProcessor(),  
        new Monolog\Processor\WebProcessor(),  
    ],  
]
```

Demonstração da organização de log para requisições

Campos de registro Histograma [Criar métrica](#)

Campos de registro <>

Campos e valores de pesquisa

RESOURCE TYPE

- Revisão do Cloud Run Limpar

SEVERITY

- Padrão 8
- Informações 7
- Aviso 1

LOG NAME

- run.googleapis.com/stdout 10
- cloudaudit.googleapis.com/system_event 2
- run.googleapis.com/requests 2
- cloudaudit.googleapis.com/activity 1
- run.googleapis.com/varlog/system 1

PROJECT ID

- directed-asset-402321 16

SERVICE NAME

- hyperf-example-app-p... Limpar

REVISION NAME

- hyperf-example-app-poc-00003-6qb 14
- Valor ausente 2

Histograma

Resultados da consulta 16 entradas de registro Encontrar nos

GRAVIDADE	HORA	ART	RESUMO	Editar	Campos de resumo	Unir as linhas
i	2023-10-21 12:17:47.989	POST	200	690 B	530 ms	PostmanRuntime... https://hyperf-example-app-poc-b2fbr4ztia-uc.a.run.app/

Copiar Entradas semelhantes Expandir os campos aninhados Ocultar resumo de registros

```
{
  httpRequest: {
    latency: "0.530024964s"
    protocol: "HTTP/1.1"
    remoteIp: "191.241.129.11"
    requestMethod: "POST"
    requestSize: "965"
    requestUrl: "https://hyperf-example-app-poc-b2fbr4ztia-uc.a.run.app/"
    responseSize: "690"
    serverIp: "216.239.38.53"
    status: 200
    userAgent: "PostmanRuntime/7.34.0"
  }
  insertId: "6533eb9c00082af13305545d"
  labels: {}
  logName: "projects/directed-asset-402321/logs/run.googleapis.com%2Frequests"
  receiveTimestamp: "2023-10-21T15:17:48.543845889Z"
  resource: {}
  severity: "INFO"
  spanId: "12595761529754900420"
  timestamp: "2023-10-21T15:17:47.989652Z"
  trace: "projects/directed-asset-402321/traces/43b8f69f5971643616f4f41583564ecc"
```

Explorando a organização de logs em casos de erros de validação

Resultados da consulta 16 entradas de registro

GRAVIDADE HORA ART ↑ RESUMO [Editar](#) Campos de resumo Unir as linhas

2023-10-21 12:28:26.847 Dados ausentes durante o processo de cadastrar um novo produto

[Copiar](#) [Entradas semelhantes](#) [Expandir os campos aninhados](#) [Ocultar resumo de registros](#)

```
{
  insertId: "6533ee1a000cef0f521af832"
  jsonPayload: {
    channel: "skeleton"
    context: {
      payload: {
        code: "1658869"
        descricao: "Esse é um produto inovador que está sendo disponibilizado"
        preco: "19,50"
      }
      reason_error: "O campo nome é obrigatório"
      stack_traces: [13]
      validation_messages: {
        nome: [
          0: "O campo nome é obrigatório"
        ]
      }
    }
    extra: {
      git: {
        branch: "main"
        commit: "3d06e385dc2c356c35450426c0c8eedff2b03002"
      }
      memory_usage: "8 MB"
      process_id: 5
    }
    message: "Dados ausentes durante o processo de cadastrar um novo produto"
  }
}
```

Análise de logs com Stack Traces: Exemplos e Insights

Consulta Recente (8) Salvo (0) Sugeridos (1) Biblioteca Limpar consulta Salvar

Últimos 30 minutos Revisão do Cloud Run +1 Nome do regist

Campos de registro Histograma Criar métrica Criar alerta

Resultados da consulta 16 entradas de registro Encontrar nos resultados

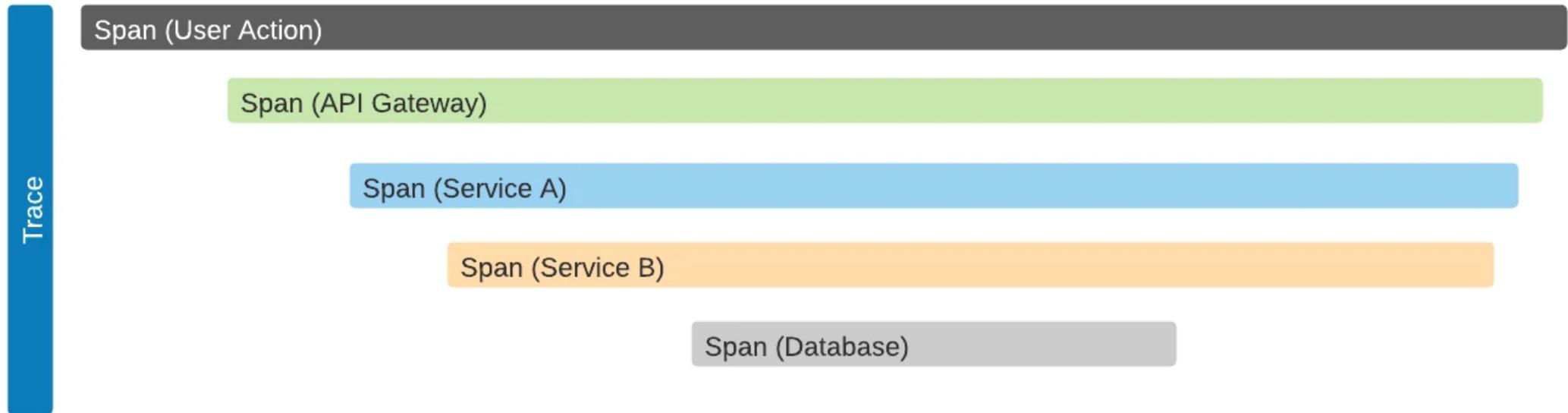
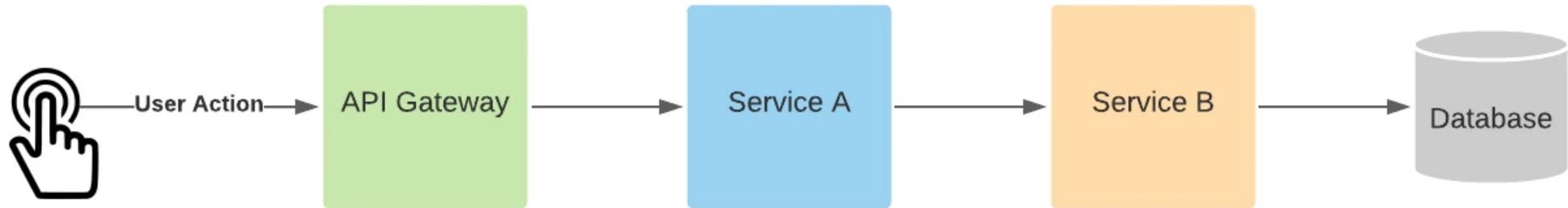
GRAVIDADE HORA ART RESUMO Editar Campos de resumo Unir as linhas

```
{
  insertId: "6533ee1a00cef0f521af832"
  jsonPayload: {
    channel: "skeleton"
    context: {
      payload: {3}
      reason_error: "O campo nome é obrigatório"
      stack_traces: [
        0: {
          class: "App\Controller\IndexController"
          file: "/opt/www/vendor/hyperf/http-server/src/CoreMiddleware.php"
          function: "index"
          line: 142
          type: "->"
        }
        1: {
          class: "Hyperf\HttpServer\CoreMiddleware"
          file: "/opt/www/vendor/hyperf/http-server/src/CoreMiddleware.php"
          function: "handleFound"
          line: 92
          type: "->"
        }
        2: {
          class: "Hyperf\HttpServer\CoreMiddleware"
          file: "/opt/www/vendor/hyperf/dispatcher/src/AbstractRequestHandler.php"
          function: "process"
          line: 44
          type: "->"
        }
        3: {
          class: "Hyperf\Dispatcher\AbstractRequestHandler"
```

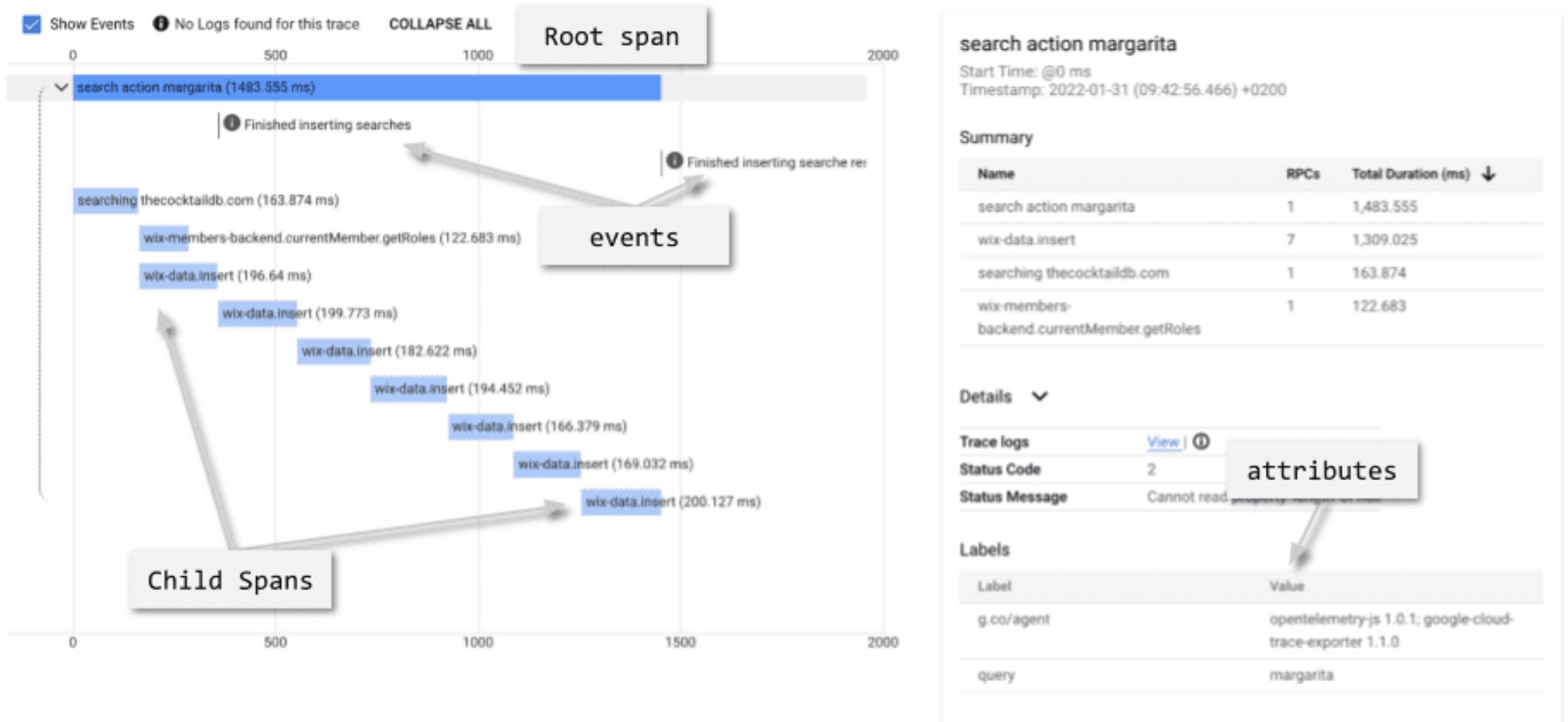
O rastreamento, também conhecido como tracing, é a capacidade de acompanhar o fluxo de uma transação ou uma solicitação através de vários componentes de um sistema distribuído.

- Visibilidade em sistemas distribuídos
- Identificação de gargalos e problemas
- Suporte à resolução de problemas

Rastreamento distribuído



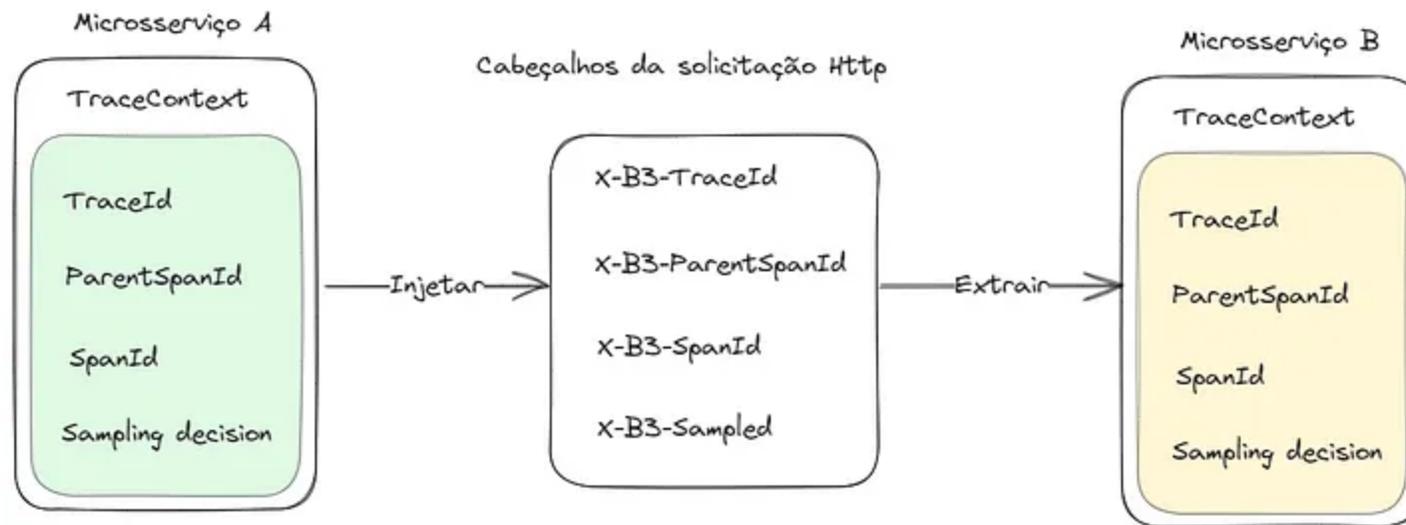
GCP Cloud Trace



Fonte da imagem: <https://www.wix.com/velo/reference/velo-package-readmes/gcp-cloud-trace-integration>

Contexto de rastreamento do B3 propagation

O B3 é uma especificação para o cabeçalho das requisições em nossa aplicação. Essa especificação define identificadores usados para agrupar uma operação em uma árvore de rastreamento.



Documentação oficial: <https://github.com/openzipkin/b3-propagation>

Exemplificação dos cabeçalhos do B3 — Fonte do autor

Exemplo em uma requisição HTTP usando o B3 propagation

```
* Connected to httpbin.org (52.201.100.107) port 443 (#0)
* ALPN, offering http/1.1
* CAfile: /etc/ssl/certs/ca-certificates.crt
* CApath: none
* SSL re-using session ID
* SSL connection using TLSv1.2 / ECDHE-RSA-AES128-GCM-SHA256
* ALPN, server accepted to use http/1.1
* Server certificate:
* subject: CN=httpbin.org
* start date: Mar  1 00:00:00 2023 GMT
* expire date: Nov 19 23:59:59 2023 GMT
* subjectAltName: host "httpbin.org" matched cert's "httpbin.org"
* issuer: C=US; O=Amazon; CN=Amazon RSA 2048 M02
* SSL certificate verify ok.
> GET /get HTTP/1.1
Host: httpbin.org
User-Agent: GuzzleHttp/7
x-b3-traceid: 436ac96e81e42bdb
x-b3-spanid: b3436f39eac6473f
x-b3-parentspanid: 436ac96e81e42bdb
x-b3-sampled: 1
```

Contexto de rastreamento do X-Cloud-Trace-Context

- Ele é usado pelo Cloud Trace na GCP para correlacionar chamadas entre serviços e fornecer visibilidade da latência e do desempenho de um aplicativo.

```
"X-Cloud-Trace-Context: TRACE_ID/SPAN_ID;o=TRACE_TRUE"
```

```
curl "http://www.example.com" \  
--header "X-Cloud-Trace-Context:105445aa7843bc8bf206b12000100000/1;o=1"
```

Observabilidade versus monitoramento

O monitoramento tradicional não funciona bem em sistemas complexos, como microsserviços e sistemas distribuídos. Ele só pode rastrear o que já é conhecido. Por exemplo:

- Qual a taxa de requisições por segundo do meu aplicativo?
- Quanta memória o Redis está usando?
- Qual é a latência média de uma solicitação?

No entanto, no decorrer da manutenção de aplicações, muitas vezes precisamos ser capazes de responder a perguntas mais profundas como estas:

- Onde está o gargalo no sistema causando latência para um endpoint específico?
- O que mais mudou em meu sistema que pode estar causando esse erro?
- Qual serviço preciso atualizar para corrigir esse erro?

Monitoramento

muitos aspectos importantes
não são visíveis apenas
com monitoramento.



Observabilidade é sobre entender
o que não é facilmente visível



Observabilidade

Selecionei algumas ferramentas para mostrar como a observabilidade é usada. Lembre-se de que existem outras opções disponíveis, mas escolhi as mais comuns. Veja abaixo os principais tópicos que vão ser falados:

- Principais ferramentas para o monitoramento e análise de métricas.
- Principais ferramentas para o rastreamento distribuído.
- Principais ferramentas para o gerenciamento e centralização de logs.

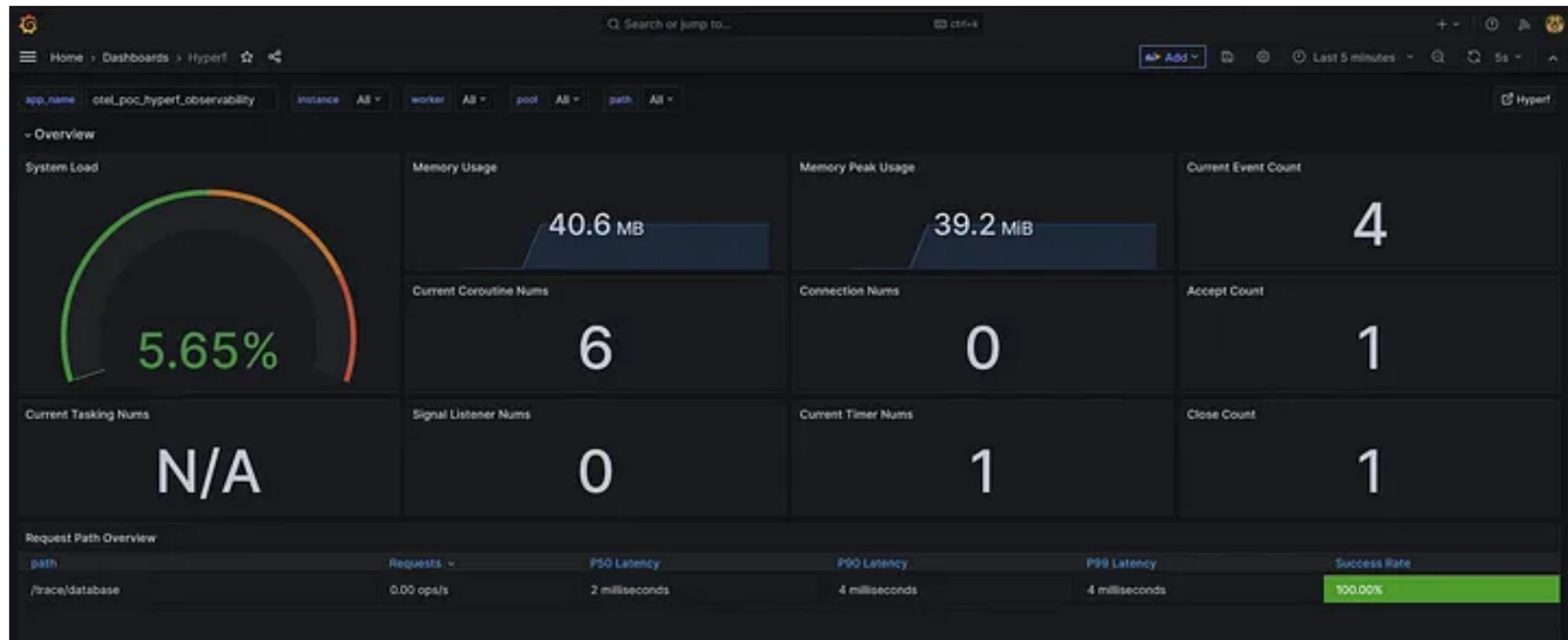
O Prometheus é um sistema de monitoramento de **código aberto** amplamente utilizado que coleta e armazena métricas sobre sistemas e aplicativos. Ele fornece uma estrutura flexível para a coleta de dados de telemetria.



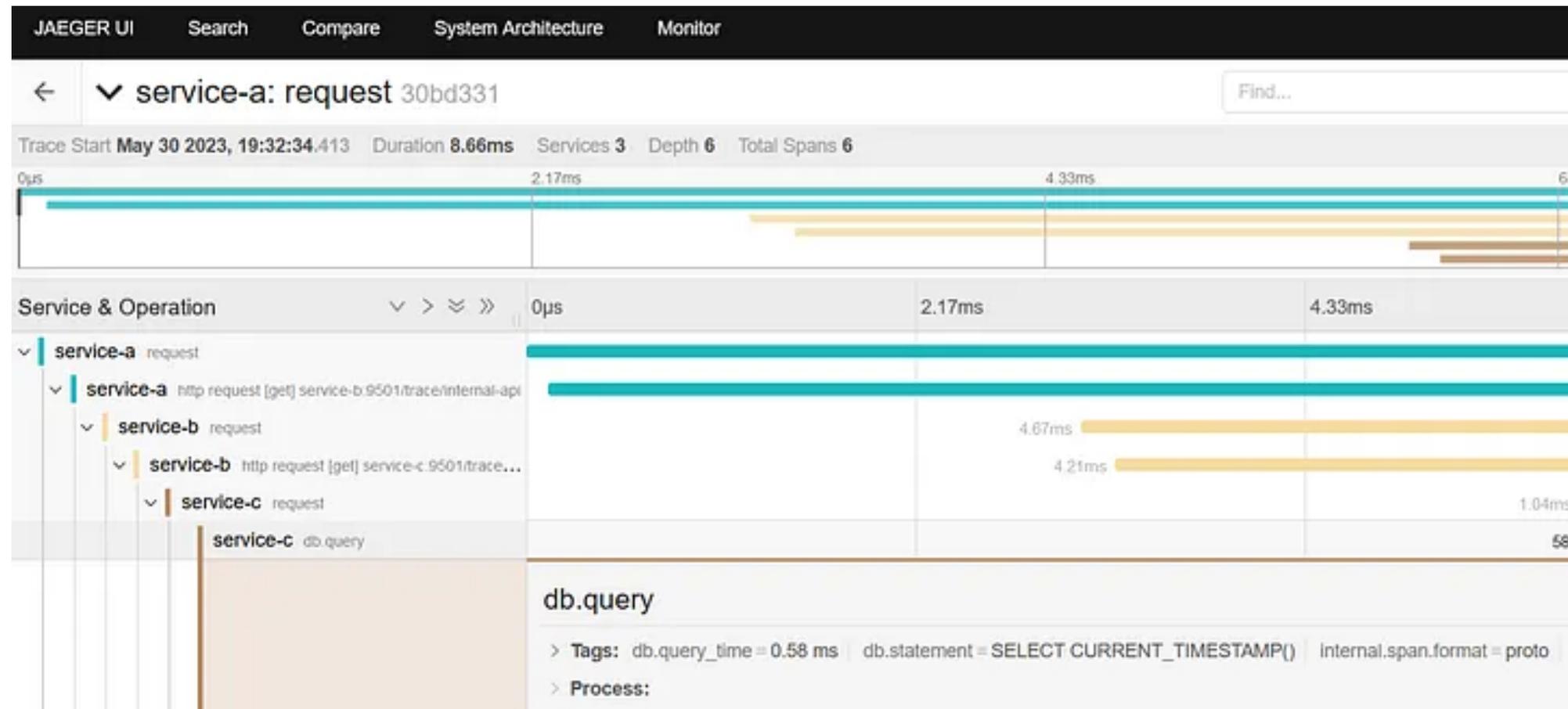
Prometheus

Grafana

O Grafana é uma plataforma de **código aberto** amplamente usada para visualização e monitoramento de dados em tempo real. Ele permite que os usuários criem painéis interativos e gráficos.



O Jaeger é uma plataforma de **código aberto** para rastreamento distribuído, projetada para monitorar e solucionar problemas em sistemas complexos, como microsserviços.



Outra ferramenta importante é o Zipkin. O Zipkin é outra plataforma de rastreamento distribuído de **código aberto** que oferece recursos semelhantes ao Jaeger.

The screenshot displays the Zipkin web interface. At the top, there is a navigation bar with the Zipkin logo, a search bar labeled 'Find a trace', a 'Dependencies' link, a language selector set to 'ENGLISH', and a 'Search by trace ID' button. Below the navigation bar, the main content area shows the trace details for 'poc-hyperf-observability: request' with a trace ID of 'baed15e92f6383fb'. The interface includes a 'SPAN TABLE' button and a summary bar indicating a duration of 26.744ms, 1 service, and 2 total spans. A timeline visualization shows the span 'poc-hyperf-observability: request' from 0ms to 26.744ms, and a sub-span 'poc-hyperf-observability: db.query' from 17.829ms to 26.744ms. A table on the right provides metadata for the selected span, including service name, span name, span ID, and parent ID. An 'Annotation' section is also visible at the bottom right.

Service name	Span name
poc-hyperf-observability	request

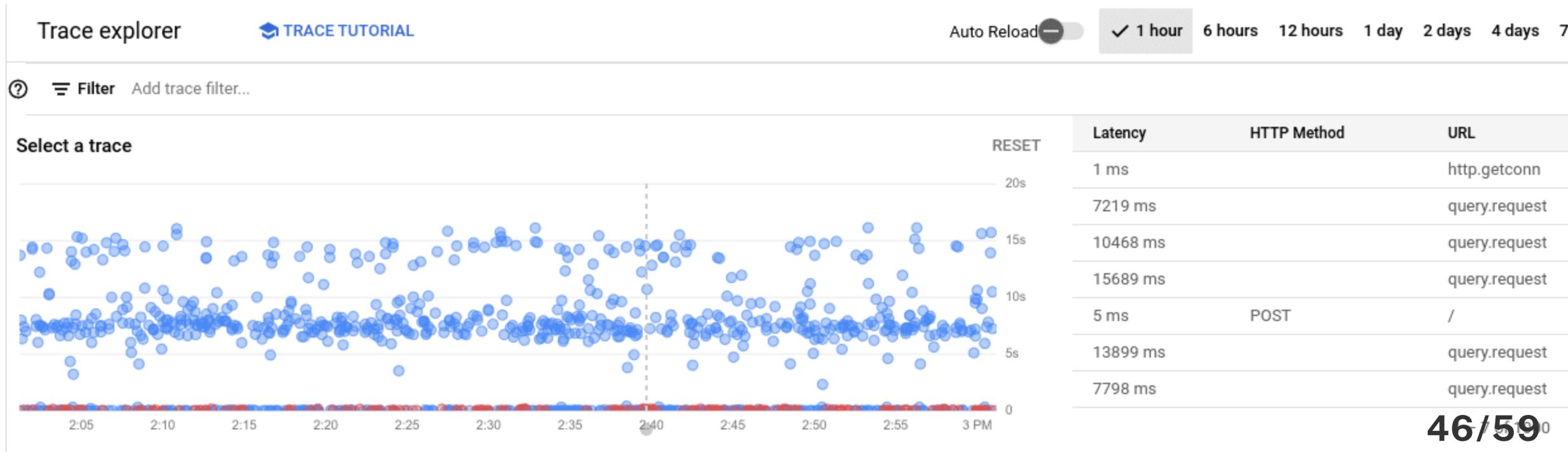
Span ID	Parent ID
baed15e92f6383fb	none

Start Time	Value
05/28 16:20:46.405	Server Start

Cloud Trace

O Cloud Trace é um sistema de rastreamento distribuído que coleta dados de latência dos aplicativos e os exibe no Console do Google Cloud.

- US\$ 0,20/milhão por trace processado



O Graylog é uma plataforma de gerenciamento de registros e análise de dados de **código aberto** que oferece uma solução centralizada para coletar, armazenar, pesquisar e analisar registros de sistemas e aplicativos.

The screenshot displays the Graylog web interface. At the top, there is a navigation bar with the Graylog logo and menu items: Search, Streams, Alerts, Dashboards, Sources, System (with a dropdown arrow and a red notification badge with the number 1), In 0 / Out 0 msg/s, Help (with a dropdown arrow), and Administrator (with a dropdown arrow). Below the navigation bar, there is a search bar with a dropdown menu set to "Search in the last 5 minutes", a play button, a "Not updating" dropdown, and a "Saved searches" dropdown. The search query is "g12_source_input:58be0be8f0e1fc0e94f80450".

The main content area is divided into two panels. The left panel, titled "Search result", shows "Found 12,529 messages in 26 ms, searched in 1 index." and "Results retrieved at 2017-04-11 00:00:31." It includes buttons for "Add count to dashboard", "Save search criteria", and "More actions". Below this, there are tabs for "Fields" and "Decorators", and a "Filter fields" section with options for "Default", "All", "None", and "Filter fields".

The right panel, titled "Histogram", shows a bar chart with the y-axis ranging from 0 to 3K and the x-axis showing time intervals: 23:55, 23:56, 23:57, 23:58, 23:59, and Tue 11. The bars show an increasing trend in message counts over time. A button "Add to dashboard" is visible in the top right corner of the histogram panel.

At the bottom of the interface, there is a "Messages" section with a pagination control showing "Previous", "1" (highlighted), "2", "3", "4", "5", "6", "7", "8", "9", "10", and "Next".

Cloud Logging

Gerenciamento de registros em tempo real e totalmente gerenciado com armazenamento, pesquisa, análise e alertas em escala de exabytes.

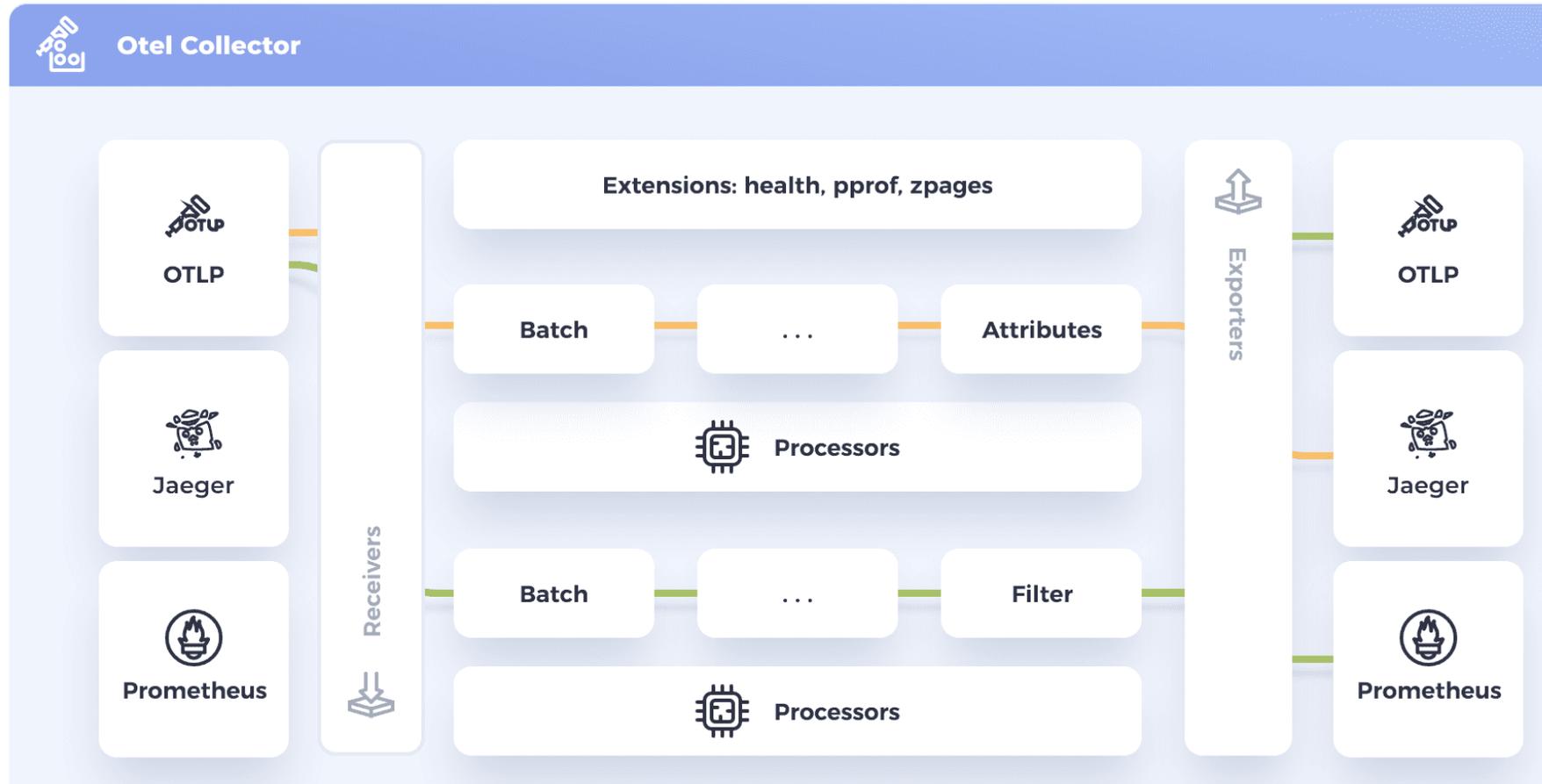
RECURSO	PREÇO
Armazenamento do Logging	US\$ 0,50/GiB
Retenção de registros	US\$ 0,01/GiB



Google
Cloud Logging

OpenTelemetry

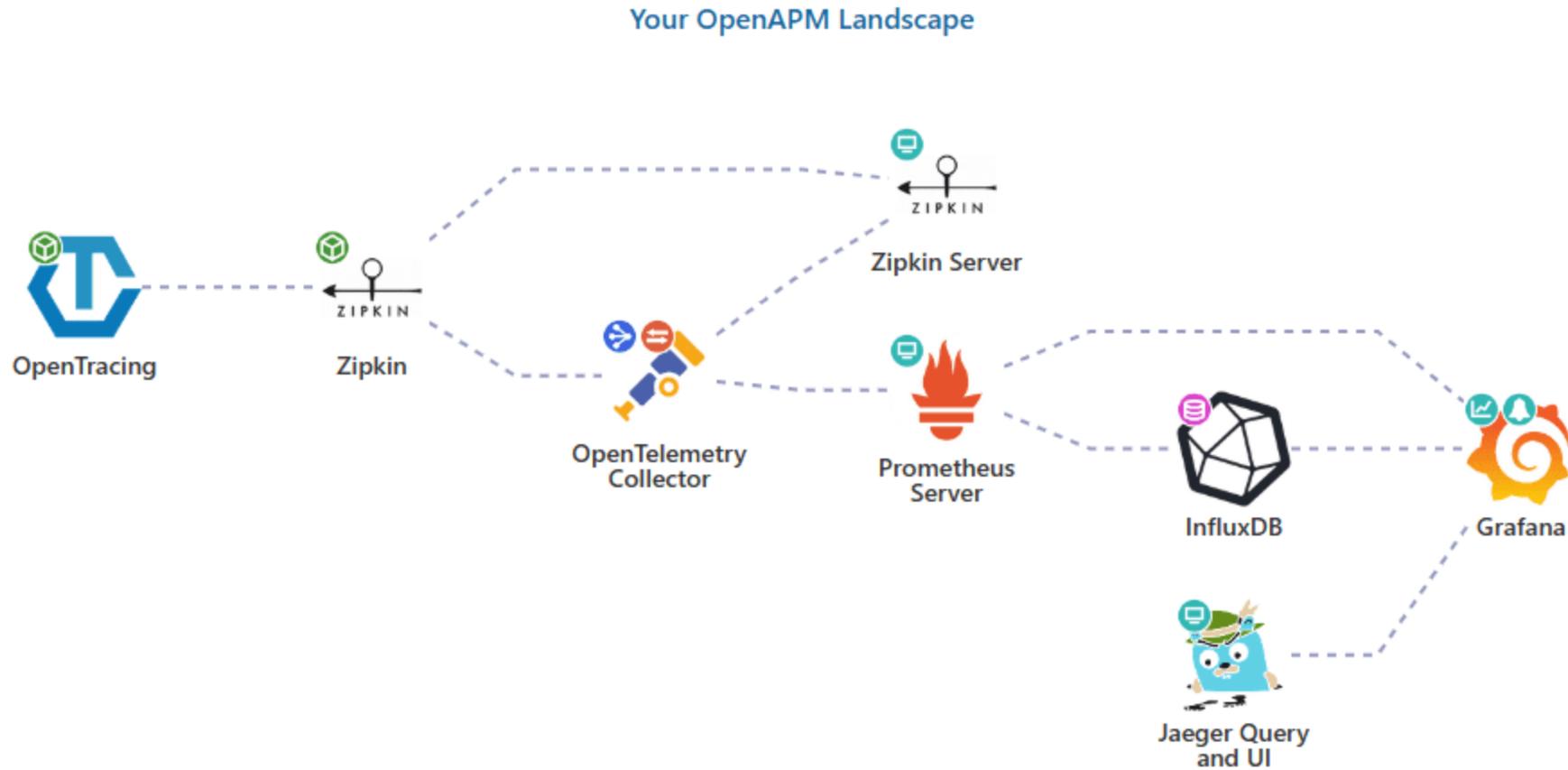
OpenTelemetry é um projeto de **código aberto** que fornece um conjunto de APIs e SDKs para coletar, processar e exportar dados de telemetria de aplicativos e infraestrutura.



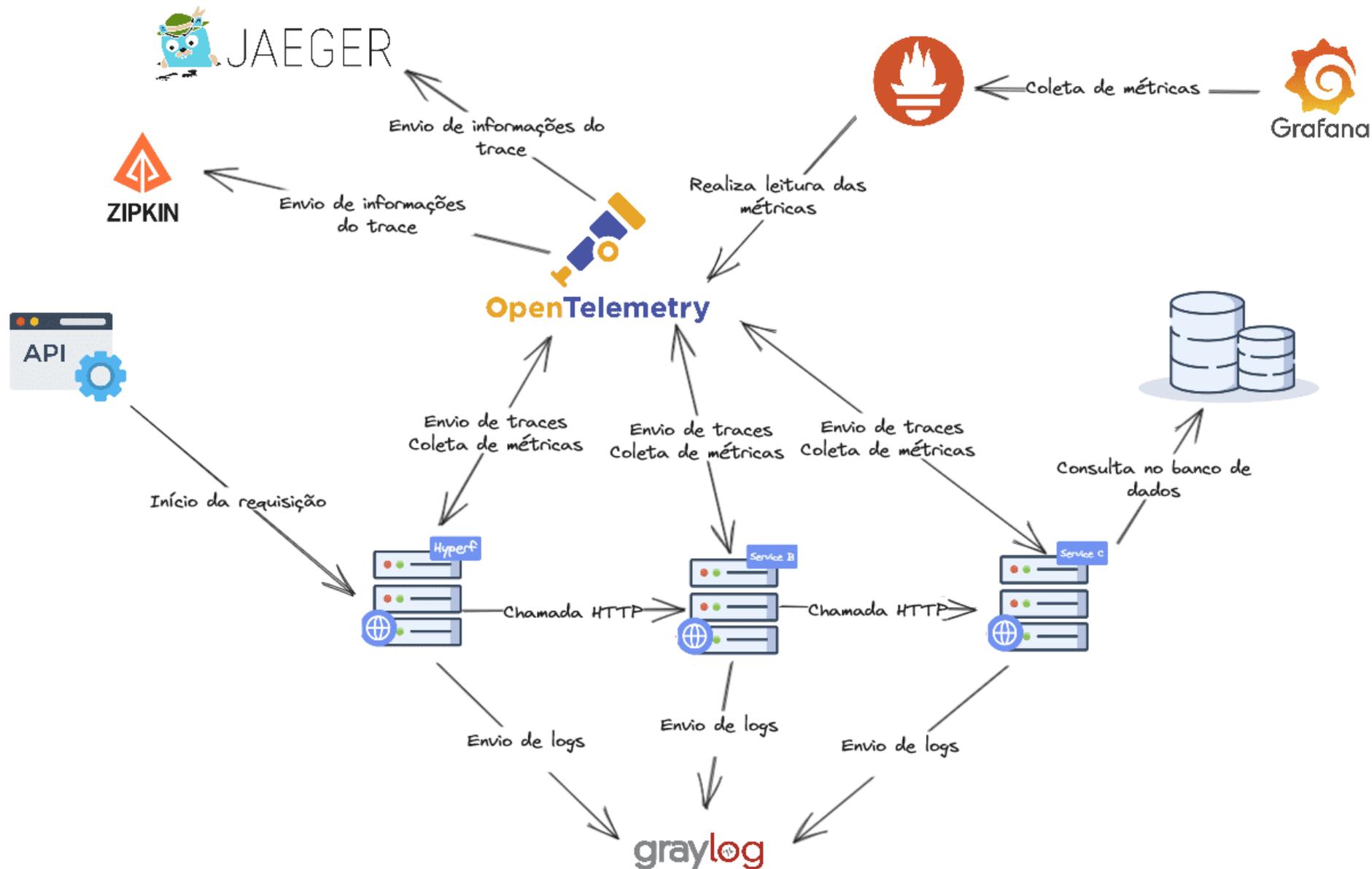
Destaque de outras ferramentas que merecem menção

Ferramenta	Classificação	Avaliações
Dynatrace	4.5	1398
New Relic	4.5	1358
Datadog	4.4	660
IBM Instana Observability	4.2	273
Amazon CloudWatch	4.4	217
Elastic Observability	4.5	159

Dica de uma excelente ferramenta para desenhar uma stack



Explorando o uso dessas ferramentas em um cenário de exemplo



Agora, vamos explorar como essas ferramentas estão conectadas e como os dados estão fluindo entre elas em um nível de código. O primeiro passo para obter essas informações é acessar o repositório no **GitHub** listado abaixo:



O que é service mesh?

Uma service mesh é uma camada de infraestrutura que fornece funcionalidades para a comunicação entre serviços em uma arquitetura de microsserviços. Ela é responsável por gerenciar o tráfego entre os serviços, fornecendo:

- Gerenciamento de tráfego
- Segurança
- Observabilidade
- Resiliência

O que é Istio?

Istio é uma service mesh open source criada pelo **Google** e pela **IBM**. É uma plataforma que permite aos desenvolvedores e equipes de operações gerenciar e proteger aplicativos baseados em microsserviços de forma eficiente.

- Alcance uma rede de serviços consistente.
- Proteja seus serviços com os benefícios do Istio.
- Melhore o desempenho do aplicativo.
- Alcance a observabilidade dos seus serviços.

Observabilidade com o Istio

Namespace: default | Traffic | Versioned app graph

Display | Find... | Hide... | 1

Oct 18, 10:10:42 PM ... 10:11:42 PM

Current Graph

- NS default ✓
- 12 apps (12 versions)
- 12 services
- 29 edges

	Inbound	Outbound	Total
HTTP (requests per second):			
Total	%Success	%Error	
5.51	100.00	0.00	

0 25 50 75 100

■ OK ■ 3xx ■ 4xx ■ 5xx ■ NR

Hide

Apresentação prática de observabilidade com o Istio

Para entender melhor como nossos microsserviços estão se comportando, vamos usar o Istio, uma ferramenta que fornece informações de observabilidade. Para começar, vamos acessar o repositório no **Gitlab** listado abaixo.



Espero que esta apresentação tenha fornecido uma visão inicial sobre os conceitos de observabilidade.

- As tecnologias mencionadas ao longo desta publicação são importantes e amplamente utilizadas no mercado.
- Selecionei algumas tecnologias que considerarei relevantes e que podem ser utilizadas localmente por meio do Docker.
- Lembre-se de que as ferramentas observabilidade são bastante poderosas para monitorar, depurar e otimizar seu projeto.
- A observabilidade é um tema amplo e em constante evolução.

Agradeço a atenção de todos. Agora, vou separar um pequeno tempo para responder perguntas e dúvidas que possam ter surgido durante a minha explicação na apresentação.