SATC
EDUCAÇÃO E TECNOLOGIA

Engenharia da Computação

# DevOps

Gledson Scotti

# Monitoramento e Observabilidade

SATC EDUCAÇÃO E TECNOLOGIA | Engenharia da Computação

Engenharia da Computação

# Monitoramento e Observabilidade

**Objetivo da Aula:**

- Compreender os conceitos de monitoramento e observabilidade.
- Diferenciar os dois conceitos e suas aplicações.
- Explorar ferramentas como Jaeger para observabilidade.
- Aplicar os conceitos em exercícios teóricos e práticos.

Engenharia da Computação

# Monitoramento e Observabilidade

## Visão Geral

Monitoramento e observabilidade são fundamentais para gerenciar sistemas distribuídos, assegurando desempenho e disponibilidade. Monitoramento coleta métricas específicas e alerta sobre problemas conhecidos, enquanto observabilidade proporciona uma análise profunda do comportamento interno dos sistemas.

Originalmente voltado para sistemas locais, o monitoramento evoluiu para soluções sofisticadas, capazes de lidar com a complexidade e o grande volume de dados de ambientes distribuídos modernos.

Engenharia da Computação

# Monitoramento - Definição

**Definição:** Monitoramento é o processo de coletar, analisar e alertar sobre métricas e eventos de um sistema para garantir que ele funcione conforme esperado.

**Foco:** Saúde do sistema (uptime, latência, erros).

Engenharia da Computação

# Monitoramento - Exemplos

**Exemplos de métricas:**
Tempo de resposta de uma API.
Uso de CPU e memória.
Taxa de erros (HTTP 500).

**Limitações:**
1. Focado em "o que está errado", não necessariamente no "porquê";
2. Baseado em métricas predefinidas, pode não capturar problemas inesperados.

Engenharia da Computação

# Monitoramento - Ferramentas

| Ferramenta | Descrição |
|---|---|
| Prometheus | Ferramenta de monitoramento open-source focada em métricas de séries temporais, ideal para sistemas dinâmicos como Kubernetes, com coleta via pull e integração com Grafana. |
| Grafana | Plataforma de visualização de dados, usada para criar dashboards interativos, integrando-se a fontes como Prometheus, Zabbix e bases de dados, com suporte a alertas. |
| Zabbix | Solução de monitoramento abrangente, monitora servidores, redes, aplicações e dispositivos via agentes ou protocolos como SNMP, com alertas e relatórios detalhados. |
| Nagios | Ferramenta clássica de monitoramento, verifica hosts e serviços via plugins, com interface web para alertas e relatórios, mas menos escalável em ambientes modernos. |
| Datadog | Plataforma baseada em nuvem que monitora infraestrutura, aplicações e logs, com dashboards personalizáveis, alertas e integração com diversas tecnologias. |
| New Relic | Solução de observabilidade que monitora performance de aplicações, infraestrutura e experiência do usuário, com análise detalhada e suporte a ambientes híbridos. |
| Icinga | Fork do Nagios, open-source, monitora redes e serviços com interface web moderna, suporte a plugins e configuração flexível para alertas. |
| Elastic Stack | Combina Elasticsearch, Logstash e Kibana para monitoramento de logs e métricas, ideal para análise de dados em tempo real e visualizações. |
| Splunk | Plataforma robusta para monitoramento e análise de logs, com recursos de busca avançada, alertas e suporte a grandes volumes de dados corporativos. |

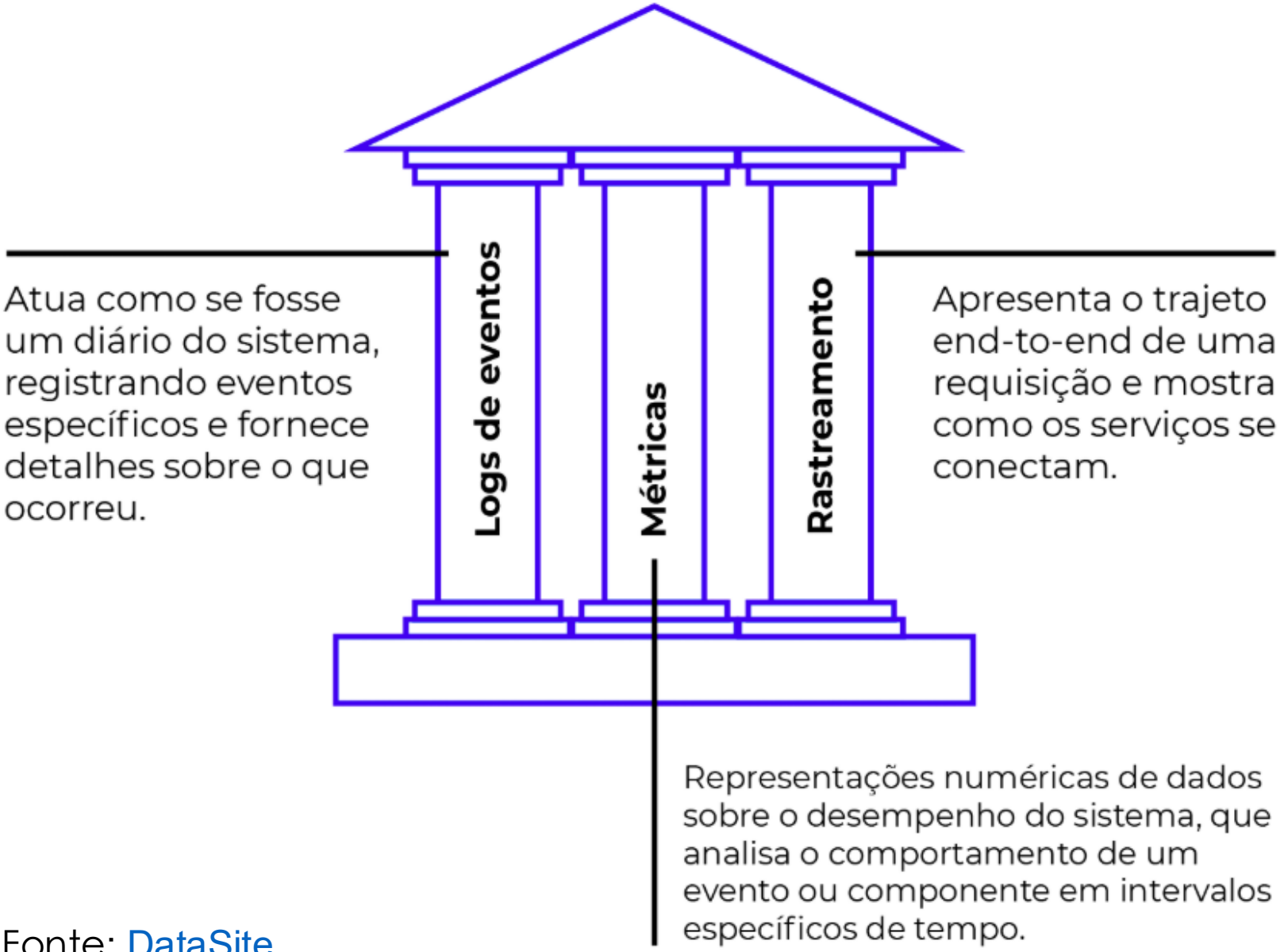
Engenharia da Computação

# Observabilidade - Definição

**Definição**: Observabilidade é a capacidade de entender o estado interno de um sistema a partir de suas saídas externas (logs, métricas e traces).

**Foco:** Diagnosticar o "porquê" de um problema, especialmente em sistemas complexos e distribuídos.

# Observabilidade - Pilares




Fonte: DataSite

Engenharia da Computação

# Observabilidade - Pilares

**Logs:** Registros detalhados de eventos (ex.: erro em uma requisição).

**Métricas:** Dados numéricos agregados (ex.: latência média).

**Traces(APM):** Rastreamento de requisições através de múltiplos serviços. APM (Application Performance Monitoring tool) é uma ferramenta de software que monitora o desempenho de aplicações e sistemas.

**Observabilidade** também nos dá o Diagnóstico profundo de problemas, adaptação a falhas inesperadas e Suporte a sistemas distribuídos (microserviços).

Engenharia da Computação

# Observabilidade - Ferramentas

| Ferramenta | Descrição |
| --- | --- |
| Jaeger | Ferramenta open-source para rastreamento distribuído, ideal para monitorar latência e dependências em arquiteturas de microsserviços. |
| OpenTelemetry | Framework open-source que padroniza coleta de métricas, logs e traces, integrando-se a várias ferramentas como Jaeger e Prometheus. |
| Honeycomb | Plataforma de observabilidade baseada em nuvem, focada em análise de eventos de alta cardinalidade, com queries rápidas e debugging detalhado. |
| Lightstep | Solução de observabilidade que combina rastreamento, métricas e logs, com foco em performance de sistemas distribuídos e alertas inteligentes. |
| Dynatrace | Plataforma de observabilidade alimentada por IA, monitora aplicações, infraestrutura e experiência do usuário, com automação e análise preditiva. |

Engenharia da Computação

# Monitoramento vs Observabilidade

| Aspecto | Monitoramento | Observabilidade |
|---|---|---|
| Foco | Detectar problemas conhecidos | Entender problemas desconhecidos |
| Abordagem | Reativa (alertas) | Proativa (exploração) |
| Dados | Métricas predefinidas | Logs, métricas, traces |
| Complexidade | Simples a moderada | Alta (sistemas distribuídos) |

# Monitoramento e Observabilidade

**Licenças:**

•**Open-source**: Prometheus, Grafana (OSS), Zabbix, Nagios Core, Icinga, Elastic Stack (componentes principais), Jaeger, **OpenTelemetry**.

•**Pagas**: Datadog, New Relic, Splunk, Honeycomb, Lightstep, Dynatrace, Nagios XI, Grafana Enterprise, Elastic Stack (recursos avançados).

# Monitoramento e Observabilidade

Engenharia da Computação

# Monitoramento e Observabilidade

No contexto de observabilidade e monitoramento, a **instrumentação** adiciona código ou agentes a sistemas para coletar telemetria (métricas, logs, traces), tornando-os observáveis para ferramentas como OpenTelemetry, Prometheus e Grafana. Já a **configuração** ajusta essas ferramentas para coletar, armazenar e processar dados, definindo endpoints (URLs de métricas), regras de coleta, armazenamento e visualizações (ex.: dashboards no Grafana).

Engenharia da Computação

# Monitoramento e Observabilidade

**OpenTelemetry e Instrumentação Automática:**

OpenTelemetry é um framework open-source que padroniza a coleta de telemetria. Ele oferece bibliotecas para linguagens como Java, Python e Go, que podem instrumentar automaticamente frameworks populares (ex.: Spring, Flask, Express).

Por exemplo, ao adicionar a biblioteca OpenTelemetry a uma aplicação Spring, ela automaticamente captura métricas de requisições HTTP (como tempo de resposta) e traces de chamadas entre serviços, sem necessidade de codificação manual extensiva. Isso é chamado de "instrumentação automática". No entanto, para casos específicos, pode ser necessário adicionar instrumentação manual, como registrar eventos customizados (ex.: "usuário fez login").

Engenharia da Computação

# Monitoramento e Observabilidade

**Prometheus e Exporters:**

Prometheus, uma ferramenta de monitoramento de métricas, não instrumenta aplicações diretamente. Em vez disso, ele depende de exporters, que são componentes que traduzem dados de sistemas ou aplicações para o formato de métricas do Prometheus. Por exemplo, para monitorar um banco de dados MySQL, você configura o MySQL Exporter, que expõe métricas como consultas por segundo em um endpoint HTTP (ex.: *http://mysql:9104/metrics*).

A aplicação ou sistema precisa ser configurado para expor essas métricas, o que exige mais esforço manual em comparação com a automação do OpenTelemetry. Para aplicações customizadas, você pode usar bibliotecas do Prometheus (como Client Libraries em Go ou Python) para expor métricas específicas.

# Monitoramento e Observabilidade

**Prometheus - Configuração de Endpoints, Scrape Intervals e Alertas**

Prometheus usa um modelo de coleta **pull**, onde ele periodicamente faz requisições HTTP (scraping) a endpoints de métricas (ex.: /metrics). Você configura o arquivo prometheus.yml para definir:

- **Endpoints**: Quais sistemas ou aplicações serão monitorados (ex.: http://app:8080/metrics).
- **Scrape Intervals**: A frequência de coleta (ex.: a cada 15 segundos, ajustado via scrape_interval: 15s). Intervalos curtos aumentam a granularidade, mas consomem mais recursos.
- **Regras de Alerta**: Regras baseadas em métricas, como "se o uso de CPU ultrapassar 80% por 5 minutos, envie um alerta". Essas regras são definidas em arquivos de regras e integradas com ferramentas como Alertmanager para notificações (ex.: via Slack ou PagerDuty).
- **Armazenamento**: Configurar onde as métricas serão armazenadas, como um disco local ou soluções remotas (ex.: Thanos para escalabilidade).

# Monitoramento e Observabilidade

**Grafana - Configuração de Dashboards**:

Grafana é uma ferramenta de visualização que se conecta a fontes de dados como Prometheus, Zabbix ou OpenTelemetry (via backends como Jaeger para traces). A configuração envolve:

- **Conectar Fontes de Dados**: Adicionar o Prometheus como fonte (ex.: apontando para http://prometheus:9090).
- **Criar Dashboards**: Configurar painéis com gráficos, tabelas ou gauges, usando queries no formato da fonte (ex.: PromQL para Prometheus, como rate(http_requests_total[5m]) para taxa de requisições).
- **Configurar Alertas**: Definir alertas baseados em thresholds (ex.: "notificar se latência > 500ms"), integrando com canais como e-mail ou Slack.
- **Personalização**: Ajustar layouts, variáveis e filtros para facilitar a análise (ex.: dropdown para selecionar ambientes como "produção" ou "teste").

Exemplo: Um dashboard no Grafana pode mostrar a latência média de uma aplicação, puxando métricas do Prometheus e exibindo um gráfico de séries temporais.

# Monitoramento e Observabilidade

Engenharia da Computação

**Desafios**

•**Volume de Dados**: Gerenciar grandes quantidades de métricas, logs e traces exige infraestrutura escalável e eficiente.

•**Complexidade**: Sistemas distribuídos com microserviços aumentam a dificuldade de rastreamento e analise.

•**Segurança**: CProteger dados sensíveis coletados é vital para manter conformidade e privacidade.

•**Cultura DevOps:** Fomentar colaboração entre equipes de desenvolvimento e operações potencializam o sucesso das práticas.

**Boas Práticas**

•**Instrumentação**: Use instrumentação automática quando possível (ex.: OpenTelemetry), mas adicione métricas customizadas para KPIs específicos. Evite coletar dados desnecessários para reduzir overhead.

•**Configuração**: Defina scrape intervals adequados (ex.: 15s a 1min) e retenção de dados (ex.: 15 dias no Prometheus) com base na escala do sistema. Teste regras de alerta para evitar falsos positivos.

•**Integração**: Combine ferramentas (ex.: OpenTelemetry para traces, Prometheus para métricas, Grafana para visualização) para uma observabilidade completa.

Engenharia da Computação

# Monitoramento e Observabilidade

## Benefícios do Monitoramento e Observabilidade

1

### Detecção Rápida

Problemas são identificados precocemente, minimizando impactos e tempo de inatividade.

2

### Otimização de Performance

Maior eficiência no uso de recursos aumenta a capacidade e reduz custos operacionais.

3

### Diagnóstico Preciso

Identificação rápida e assertiva da causa raiz dos incidentes.

4

### Decisões Baseadas em Dados

Insights gerados orientam estratégias e melhorias contínuas.

## Conclusão: Rumo a um Sistema Mais Resiliente

Monitoramento e observabilidade são pilares estratégicos para garantir a confiabilidade e desempenho de sistemas modernos. Investir em ferramentas adequadas e em práticas maduras é fundamental para acelerar a detecção e resolução de incidentes.

Uma cultura de observabilidade contínua promove sistemas mais resilientes, capazes de se adaptar rapidamente a mudanças e desafios, garantindo assim alta disponibilidade e satisfação dos usuários.