MONITORAMENTO E OBSERVABILIDADE EM SISTEMAS WEB: UM ESTUDO SOBRE FERRAMENTAS E TÉCNICAS

João Lucas de Sousa Martins
joaolucasmartins@copin.ufcg.edu.br
Universidade Federal de Campina Grande

1 Introdução

Com o avanço das aplicações web distribuídas e baseadas em microsserviços, a capacidade de compreender o comportamento interno desses sistemas tornou-se essencial. Nesse contexto, a observabilidade emerge como um conceito-chave, indo além do monitoramento tradicional ao integrar e correlacionar sinais como métricas, logs e traces para inferir o estado interno de um sistema a partir de sua execução externa (García et al., 2023) [Gar23].

Essa abordagem se torna especialmente crítica em arquiteturas modernas e ambientes em nuvem, onde falhas intermitentes, degradações silenciosas e interdependências complexas entre serviços podem comprometer a estabilidade do sistema. De acordo com Niedermaier et al. (2019) [Nie19], o uso crescente de microsserviços e a heterogeneidade dos sistemas elevam a complexidade do monitoramento e tornam essencial uma abordagem holística de observabilidade.

Embora diversas ferramentas e práticas de observabilidade estejam disponíveis, ainda há lacunas na compreensão sobre como decisões de design afetam a qualidade dos dados observáveis gerados em tempo de execução. Karumuri et al. (2021) [Kar21] destacam que a combinação de múltiplas ferramentas especializadas pode gerar degradação de desempenho, complexidade operacional e dificuldades na gestão de dados em larga escala. Nesse cenário, torna-se essencial não apenas analisar soluções existentes na literatura, mas também avaliar sua eficácia prática por meio de experimentos reprodutíveis e controlados.

Este trabalho propõe uma reprodução experimental com base em um cenário de falha em sistemas distribuídos, utilizando uma aplicação baseada em microsserviços instrumentada com OpenTelemetry. O objetivo é observar e quantificar os efeitos da falha em métricas de desempenho e rastreamento, a fim de analisar a eficácia das ferramentas de observabilidade utilizadas e a confiabilidade da metodologia aplicada. A análise estatística dos dados gerados busca verificar a significância dos impactos e a consistência dos resultados ao longo de diferentes execuções.

Com base nesse contexto, esta pesquisa busca responder à seguinte questão:

Q1. Em que medida as técnicas e ferramentas de observabilidade destacadas na literatura contemporânea se mostram eficazes, do ponto de vista empírico, para identificar e quantificar falhas em sistemas web distribuídos?

2 Trabalhos Relacionados

Os seguintes artigos fundamentam teoricamente este estudo e contextualizam o cenário da observabilidade em sistemas distribuídos:

- **Artigo 1:** *On Observability and Monitoring of Distributed Systems – An Industry Interview Study* (2019);
- **Artigo 2:** *Enhancing Web Applications Observability through Instrumented Automated Browsers* (2023);
- **Artigo 3:** *Towards Observability Data Management at Scale* (2021).

Além disso, o experimento reproduzido neste trabalho está baseado no seguinte estudo:

- **Artigo para Reprodução Experimental:** *Informed and Assessable Observability Design Decisions in Cloud-native Microservice Applications* (2024).

In: M. Meder, A. Rapp, T. Plumbaum, and F. Hopfgartner (eds.): Proceedings of the Data-Driven Gamification Design Workshop, Tampere, Finland, 20-September-2017, published at http://ceur-ws.org

3 Metodologia

A reprodução foi realizada com base no estudo de Borges et al. (2024) [Bor24], que propõe um motor de experimentação em observabilidade chamado Oxn. O experimento selecionado insere uma falha do tipo *pause* no serviço *recommendation-service*, simulando um cenário de degradação em uma aplicação web distribuída.

3.1 Configuração do Ambiente

O ambiente foi composto pelos seguintes elementos:

- **Aplicação de referência:** OpenTelemetry Demo, que simula uma loja virtual com arquitetura de microsserviços.
- **Contêineres:** Executados via Docker Compose, incluindo serviços instrumentados com OpenTelemetry, Prometheus, Jaeger e Locust.
- **Ferramenta de experimentação:** Oxn, instalada localmente em ambiente *virtualenv* com Python 3.12.

O experimento executado foi definido no arquivo *recommendation_pause_no_cpu.yml* e conduzido com o seguinte comando no terminal: oxn experiments/recommendation_pause_no_cpu.yml –times 5 –report pause_multi_report.yml

3.2 Descrição do Experimento

A abordagem de experimentação é inspirada em práticas de engenharia do caos, porém com foco na qualidade dos dados de observabilidade gerados durante falhas. Cada experimento é composto por:

- **Sistema Sob Experimento (SUE):** Subconjunto dos microsserviços da aplicação, com exceção do gerador de carga interno.
- **Carga de trabalho:** Simulada com 50 usuários simultâneos por 10 minutos, com requisições GET e POST para os principais endpoints da aplicação.
- **Tratamento de falha:** Pausa de 120 segundos injetada no contêiner *recommendation-service* em tempo de execução.
- **Variáveis de resposta:** Métricas e traces coletados antes e após o tratamento para avaliar o impacto da falha simulada.

As variáveis monitoradas foram:

- *frontend_traces.duration* – duração dos traces no serviço frontend;
- *recommendation_traces.duration* – duração dos traces no serviço *recommendation-service*;
- *recommendations_total* – métrica Prometheus com o número de recomendações geradas.

A ferramenta Oxn permite definir essas configurações em um arquivo YAML e executá-las de forma automatizada. Sua arquitetura modular compreende um orquestrador, um *runner* para aplicar os tratamentos, um gerador de carga e observadores que capturam os dados de resposta.

3.3 Procedimentos Estatísticos

A cada execução, os dados foram coletados e comparados entre os períodos anterior e posterior à injeção da falha. Para verificar a significância das diferenças observadas, foi utilizado o teste estatístico *t de Welch*, adequado para amostras com variâncias diferentes.

O relatório gerado automaticamente pelo Oxn (*pause_multi_report.yml*) contém os valores de *p-value*, estatísticas do teste e dados agregados de desempenho para cada métrica e tratamento aplicado.

4 Resultados e Discussão

A reprodução do experimento *pause_recommendation* foi realizada cinco vezes, gerando um conjunto de métricas extraídas automaticamente pelo Oxn, consolidado no relatório *pause_multi_report.yml*. O objetivo foi avaliar se a injeção de falha (pausa de 120s no serviço *recommendation-service*) provoca impactos estatisticamente significativos nas variáveis de observabilidade monitoradas.

4.1 Análise Estatística

A Figura 1 apresenta as estatísticas t e os valores p associados às três variáveis analisadas: *frontend_traces.duration*, *recommendation_traces.duration* e *recommendations_total*. O teste aplicado foi o *t de Welch*, adequado para comparar distribuições com variâncias desiguais entre as janelas anterior e posterior ao tratamento.

4.2 Análise Visual dos Resultados

4.2.1 Estatística t por Variável

A Figura 2 apresenta um gráfico de barras com a estatística t por variável e execução. A linha vermelha (t = 2.0) representa um valor de referência comum para indicar significância em testes t bilaterais com amostras moderadas. Nota-se que:

- A variável `frontend_traces.duration` obteve valores negativos de t em 4 das 5 execuções, indicando aumento de duração após a falha.

Execução	Variável	Estatística t	p-value	Teste
05ffec0b	frontend_traces.duration	-4.0013	6.32e-05	Welch t-test
05ffec0b	recommendation_traces.duration	-2.4503	1.43e-02	Welch t-test
05ffec0b	recommendations_total	-1.2397	2.17e-01	Welch t-test
24b83c5b	frontend_traces.duration	-4.7982	1.61e-06	Welch t-test
24b83c5b	recommendation_traces.duration	-3.0016	2.69e-03	Welch t-test
24b83c5b	recommendations_total	31.9266	1.91e-80	Welch t-test
45c142cf	frontend_traces.duration	-5.4813	4.27e-08	Welch t-test
45c142cf	recommendation_traces.duration	-2.8300	4.67e-03	Welch t-test
45c142cf	recommendations_total	18.0811	3.71e-54	Welch t-test
916ba81a	frontend_traces.duration	1.9057	5.78e-02	Welch t-test
916ba81a	recommendation_traces.duration	-3.0015	2.69e-03	Welch t-test
916ba81a	recommendations_total	3.4242	7.47e-04	Welch t-test
a6e516db	frontend_traces.duration	-4.2441	2.20e-05	Welch t-test
a6e516db	recommendation_traces.duration	-2.8296	4.67e-03	Welch t-test
a6e516db	recommendations_total	29.8306	1.45e-75	Welch t-test

Figura 1: Resumo dos testes estatísticos por execução e variável de resposta

- A variável `recommendation_traces.duration` também apresentou resultados consistentes com aumento de duração, com valores de t negativos em todas as execuções.

- A métrica `recommendations_total` apresentou valores de t positivos elevados em 3 execuções, sugerindo aumento de recomendações acumuladas, e resultados não significativos nas demais.

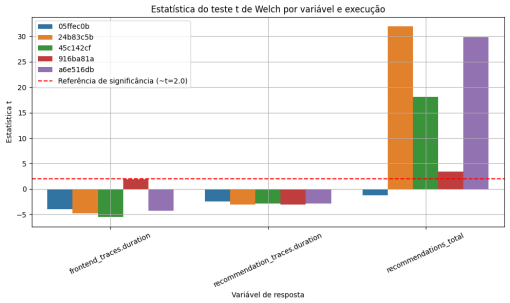

Figura 2: Estatística *t de Welch* por variável e execução

4.2.2 P-values por Execução e Variável

A Figura 3 mostra os valores de p obtidos para cada métrica. A linha pontilhada em vermelho marca o limiar de significância estatística ($p < 0{,}05$). Observa-se que:

- As métricas de `traces` (frontend e recommendation) apresentaram significância estatística na maioria das execuções.

- A variável `recommendations_total` teve *p-values* extremamente baixos em 3 execuções, reforçando evidências de alteração no sistema.

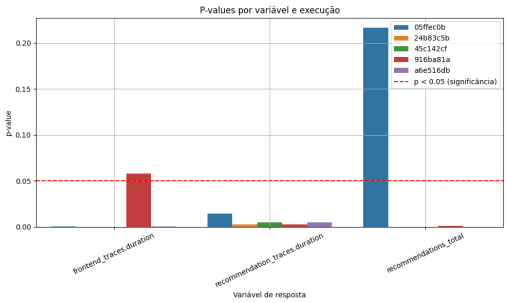

Figura 3: P-values por variável de resposta e execução

4.2.3 Heatmap da Estatística t

A Figura 4 apresenta um heatmap com os valores de t organizados por execução e variável. Esta visualização destaca a consistência dos resultados, com as células mais escuras indicando efeitos estatisticamente mais relevantes.

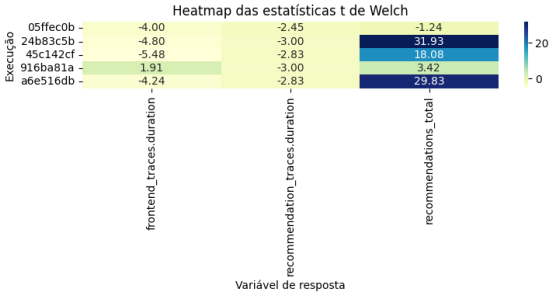

Figura 4: Heatmap das estatísticas t por variável e execução

4.2.4 Evolução da Estatística t

A Figura 5 mostra a variação da estatística t por variável de resposta ao longo das execuções. O comportamento geral evidencia que o tratamento gerou efeitos estatísticos consistentes sobre os `traces` e métricas coletadas, ainda que com alguma variabilidade natural.

4.3 Discussão dos Resultados

Os resultados obtidos demonstram que a ferramenta Oxn foi eficaz em capturar o impacto da falha injetada no sistema. A maioria das execuções revelou mudanças significativas nas métricas observadas, especialmente nas durações dos `traces` e no número de recomendações geradas.

A reprodutibilidade dos experimentos e a clareza dos dados gerados indicam que a abordagem proposta no artigo original é viável e útil para avaliar observabilidade em sistemas distribuídos. Além disso, os testes estatísticos automatizados fortalecem a análise

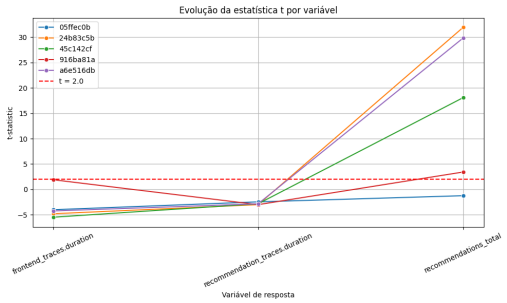

Figura 5: Evolução da estatística t por variável nas execuções

empírica dos dados e podem subsidiar decisões de engenharia em ambientes produtivos.

Todos os arquivos utilizados na reprodução, incluindo o relatório gerado (*pause_multi_report.yml*), os gráficos estatísticos e os scripts de configuração do experimento (*recommendation_pause_no_cpu.yml*), estão disponíveis em repositório público no GitHub: `https://github.com/lucasops96/Mestrado/tree/master/FPCC%202/trabalho-reproducao-fpcc2`

5 Conclusão

Este trabalho apresentou a reprodução experimental de um estudo sobre observabilidade em sistemas distribuídos, com foco na avaliação de métricas e traces diante da injeção de falhas controladas. A partir da utilização da ferramenta Oxn, foi possível automatizar a aplicação do tratamento, coletar dados observáveis e realizar análises estatísticas robustas.

Os resultados obtidos confirmaram que a abordagem de observabilidade investigada é reprodutível e eficaz para identificar impactos relevantes em variáveis como traces e contadores de recomendações. A consistência dos valores de estatística t e a presença de *p-values* significativos demonstram que a falha injetada produziu efeitos mensuráveis, especialmente nos serviços diretamente afetados.

Do ponto de vista metodológico, a reprodutibilidade do experimento fortalece a confiança na abordagem adotada, destacando o valor de experimentações controladas para validar decisões de engenharia em ambientes modernos e complexos. Além disso, o uso combinado de OpenTelemetry, Locust e Oxn mostrou-se adequado para simular cenários realistas de falha com instrumentação eficiente.

Como trabalho futuro, sugere-se a aplicação da mesma metodologia a outros tipos de falhas (como perda de pacotes ou aumento de latência) e em diferentes arquiteturas de microsserviços, além da exploração de ferramentas alternativas de observabilidade. Os achados aqui apresentados também poderão compor uma revisão sistemática mais ampla sobre o estado da arte da observabilidade em sistemas web.

Referências

[Bor24] M. C. Borges, J. Bauer, S. Werner, M. Gebauer, and S. Tai. Informed and assessable observability design decisions in cloud-native microservice applications. In *Proceedings of the 2024 IEEE 21st International Conference on Software Architecture (ICSA)*, pages 69–78. IEEE, 2024. DOI: `https://doi.org/10.1109/ICSA59870.2024.00015`.

[Gar23] B. García, F. Ricca, J. M. del Alamo, and M. Leotta. Enhancing web applications observability through instrumented automated browsers. *Journal of Systems and Software*, 203:111723, 2023. DOI: `https://doi.org/10.1016/j.jss.2023.111723`.

[Kar21] S. Karumuri, F. Solleza, S. Zdonik, and N. Tatbul. Towards observability data management at scale. *SIGMOD Record*, 49(4):18–23, 2021. DOI: `https://doi.org/10.1145/3456859.3456863`.

[Nie19] S. Niedermaier, F. Koetter, A. Freymann, and S. Wagner. On observability and monitoring of distributed systems – an industry interview study. In *Service-Oriented Computing: 17th International Conference, ICSOC 2019, Toulouse, France, October 28–31, 2019, Proceedings*, pages 36–52. Springer, 2019. DOI: `https://doi.org/10.1007/978-3-030-33702-5_3`.