PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
ENGENHARIA DE SOFTWARE - CAMPUS LOURDES
LABORATÓRIO DE EXPERIMENTAÇÃO DE SOFTWARE

Ana Carolina Corrêa, Caio Elias, Henrique Diniz e Maria Clara Santos

GraphQL vs REST : Um Experimento Controlado

1. Introdução

Este trabalho teve como objetivo investigar as diferenças de desempenho entre as APIs GraphQL e REST, com foco em dois aspectos principais: tempo de resposta e tamanho das respostas. Para isso, foram formuladas as seguintes hipóteses:

1. **RQ1:** Respostas a consultas GraphQL são mais rápidas que respostas a consultas REST?

   Hipótese Nula (H_0): O tempo de resposta médio das consultas GraphQL não é significativamente diferente do tempo de resposta médio das consultas REST.

   Hipótese Alternativa (H_1): O tempo de resposta médio das consultas GraphQL é significativamente menor que o tempo de resposta médio das consultas REST.

2. **RQ2:** Respostas a consultas GraphQL têm tamanho menor que respostas a consultas REST?

   Hipótese Nula (H_0): O tamanho médio das respostas de consultas GraphQL não é significativamente diferente do tamanho médio das respostas de consultas REST.

   Hipótese Alternativa (H_1): O tamanho médio das respostas de consultas GraphQL é significativamente menor que o tamanho médio das respostas de consultas REST.

2. Metodologia

A. Variáveis Dependentes

- Tempo de resposta (em milissegundos).
- Tamanho da resposta (em bytes).

B. Variáveis Independentes

- Tipo de API: REST ou GraphQL.
- Complexidade da consulta (simples ou complexa).
- Quantidade de dados retornados pela consulta.

C. Tratamentos

Foram implementadas consultas em GraphQL e REST que retornassem o mesmo conjunto de informações, variando a complexidade das consultas.

D. Objetos Experimentais

Os experimentos utilizaram endpoints de uma aplicação hipotética com suporte a GraphQL e REST, conectados a um banco de dados compartilhado para garantir consistência nos resultados.

E. Tipo de Projeto Experimental

Foi utilizado um **desenho cruzado (crossing)**, no qual cada tipo de consulta foi avaliado em ambas as APIs, assegurando a comparabilidade dos resultados.

F. Quantidade de Medições

- **Primeiro Script:** 30 medições de consultas simples em cada API.
- **Segundo Script:** 10 medições de consultas simples e complexas em cada API, totalizando 40 medições adicionais. Consultas complexas incluíram paginação e coletas complementares no caso do REST.

G. Ameaças à Validade

1. **Ameaças Internas:**

   Variações no desempenho do servidor e banco de dados.

   Diferenças na configuração dos endpoints GraphQL e REST.

2. **Ameaças Externas:**

   Limitação na generalização dos resultados para outros contextos ou APIs.

   Tamanho reduzido das consultas utilizadas no experimento.

3. **Ameaças de Construção:**

   Possíveis inconsistências na implementação das consultas.

3. Resultados

A. Tempo de Resposta

- **Consultas Simples:**

  GraphQL apresentou tempo médio de 1.005 ms.

  REST apresentou tempo médio de 950 ms.

  Conclusão: REST foi ligeiramente mais rápido em consultas simples.

- **Consultas Complexas:**

  GraphQL apresentou tempo médio de 1.393 ms.

  REST apresentou tempo médio de 12.256 ms.

  Conclusão: GraphQL foi 8,8 vezes mais rápido em consultas complexas.

B. Tamanho da Resposta

- **Consultas Simples:**

  GraphQL apresentou tamanho médio de 1.303 bytes.

  REST apresentou tamanho médio de 57.926 bytes.

  Conclusão: GraphQL gerou respostas 44 vezes menores.

- **Consultas Complexas:**

  GraphQL apresentou tamanho médio de 7.668 bytes.

  REST apresentou tamanho médio de 265.953 bytes.

  Conclusão: GraphQL gerou respostas 34 vezes menores.

C. Validação Estatística

Os testes de significância (como t-test) realizados confirmaram que, em consultas complexas, as diferenças de desempenho entre GraphQL e REST, tanto no tempo de resposta quanto no tamanho das respostas, são estatisticamente significativas.

4. Discussão Final

Os resultados obtidos demonstraram a superioridade do GraphQL em cenários de maior complexidade, apresentando tanto menores tempos de resposta quanto tamanhos de dados significativamente menores. Essa eficiência reflete a capacidade do GraphQL de evitar dados desnecessários e dados insuficientes, otimizando a experiência.

Por outro lado, o REST apresentou desempenho ligeiramente superior em consultas simples, devido à sua simplicidade arquitetural e menor overhead em operações básicas.

Implicações Práticas:

- **GraphQL** é ideal para aplicações que exigem consultas personalizadas e envolvem alta complexidade de dados.
- **REST** continua sendo uma solução viável para integrações mais simples e sistemas legados.

Limitações do Experimento:

- Os resultados foram obtidos em um ambiente controlado, com número limitado de medições.
- Seria recomendável ampliar os cenários testados e replicar o experimento em ambientes de produção.

5. Conclusão

O trabalho confirmou que o GraphQL oferece desempenho superior em consultas complexas, tornando-se uma solução eficiente em sistemas modernos e dinâmicos. Por outro lado, o REST mantém sua relevância em casos de menor complexidade, sendo ainda uma escolha válida dependendo do contexto do projeto.

Portanto, a escolha entre REST e GraphQL deve considerar as necessidades específicas do sistema, avaliando o trade-off entre flexibilidade e simplicidade.