DESENHO EXPERIMENTAL

Introdução

O experimento tem como propósito verificar se existe diferença mensurável de desempenho entre requisições feitas via API REST e via API GraphQL, considerando especificamente tempo de resposta e tamanho do payload retornado. Todas as etapas a seguir descrevem a estrutura completa do desenho experimental.

A. Hipóteses Nula e Alternativa

H_0 (Hipótese Nula): não há diferença estatisticamente significativa entre REST e GraphQL quanto ao tempo de resposta e ao tamanho das respostas.
H_1 (Hipótese Alternativa): GraphQL apresenta maior eficiência, resultando em tempos menores e/ou payloads mais compactos quando comparado ao REST.

B. Variáveis

1. Variáveis Dependentes (o que será mensurado)

• Tempo de resposta: intervalo total entre o envio da solicitação e a chegada da resposta completa.
• Tamanho do payload: quantidade de bytes retornada no corpo da resposta.

2. Variável Independente (o que será manipulado)

• Tipo de API, composto por duas condições:
– Condição A: chamadas via REST API
– Condição B: chamadas via GraphQL API

C. Tratamentos e Endpoints

Cada tratamento corresponde a uma forma distinta de consulta aplicada aos mesmos repositórios.

Tratamento 1 — REST

Endpoint:
GET https://api.github.com/repos/{owner}/{repo}

A chamada retorna metadados gerais do repositório, incluindo a contagem de issues. Serão registradas as duas métricas principais: tempo de resposta e tamanho do payload.

Tratamento 2 — GraphQL

Endpoint:
POST https://api.github.com/graphql

Body: consulta solicitando o mesmo dado obtido via REST (total de issues). Assim como no tratamento anterior, serão coletados tempo de resposta e tamanho da resposta retornada.

D. Objetos Experimentais

Os objetos do experimento são repositórios GitHub definidos previamente. Serão utilizados pares (owner, repo) representando repositórios diversos. Cada repositório passará pelos dois tratamentos, assegurando um experimento pareado e comparável.

E. Tipo de Projeto Experimental

O experimento segue um desenho pareado (within-subjects).
Cada repositório é avaliado em ambas as condições (REST e GraphQL). Isso elimina interferências oriundas de características internas de cada repositório e reduz variabilidade externa. As medições serão repetidas diversas vezes, alterando a ordem das chamadas para minimizar viés de cache e de sequência.

F. Quantidade de Medições

Para cada repositório e para cada tratamento serão realizadas 30 medições independentes.
O total de coletas é dado por:
número de repositórios × 2 tratamentos × 30 repetições.

O valor 30 foi adotado por ser adequado a análises estatísticas em que se busca distribuição próxima da normal.

G. Ameaças à Validade

1. Validade Interna

• Variações de rede: flutuações de conexão podem afetar os tempos mensurados.
Mitigação: alternar chamadas REST/GraphQL de forma aleatória e descartar outliers.

• Caching: respostas podem ser armazenadas em cache, reduzindo artificialmente o tempo.
Mitigação: configurar cabeçalhos que desativem cache e incluir pausas entre requisições.

2. Validade Externa

• Uso exclusivo de repositórios populares pode não representar cenários menores ou privados.
Mitigação: compor lista heterogênea de repositórios.

3. Validade de Conclusão

• Erro na marcação de timestamps: pode distorcer medições de tempo.
Mitigação: coletar timestamps precisos em milissegundos.

• Variações na codificação do payload: podem alterar tamanho da resposta.
Mitigação: medir apenas o corpo JSON em bytes, de forma consistente.

4. Validade de Construct

• “Eficiência” pode assumir significados diferentes se apenas uma métrica for considerada.
Mitigação: analisar tempo e tamanho separadamente e, se necessário, combinar indicadores.

5. Validade Social/Ecológica

• Condições de teste podem não refletir cenários reais com carga concorrente. Mitigação: registrar essa limitação e sugerir estudos futuros com múltiplos clientes simultâneos.