Étude de Performance

Comparaison des Technologies d'API

REST, SOAP, GraphQL, gRPC

Cas d'Application

Système de Gestion de Réservations Hôtelières

Réalisé par :
Youssef bahaddou
Laksir younes

Date :
20 décembre 2025

Technologies :
Spring Boot 3.2.0, Java 17, MySQL 8.0, Docker, Apache JMeter 5.6.3

Table des matières

1 Métadonnées du Projet

TABLE 1 – Métadonnées techniques du projet

Caractéristique	Valeur
Version du projet	v1.0
Lien du repository	GitHub
License	MIT License
Système de versioning	Git
Langages et frameworks	Java 17, Spring Boot 3.2.0, FastAPI, GraphQL, Apache CXF, gRPC
Base de données	MySQL 8.0 (Docker)
Outils de test	Apache JMeter 5.6.3, SoapUI, BloomRPC, ghz
Environnement	Docker Compose, Java 17+, Maven 3.9+

2 Introduction

2.1 Contexte

Dans un écosystème technologique moderne, le choix du protocole de communication impacte directement l'expérience utilisateur et les coûts d'infrastructure[cite : 27]. Cette étude compare REST, SOAP, GraphQL et gRPC dans un contexte métier de gestion hôtelière[cite : 28].

2.2 Problématique

Comment ces technologies se comportent-elles face à une montée en charge critique (jusqu'à 1000 utilisateurs simultanés) en termes de latence et de débit ? [cite : 30]

3 Résultats des Tests de Performance

3.1 Tableau 1 : Latence Moyenne (ms)

La latence mesure le délai de réponse du serveur[cite : 35]. Les valeurs "N/A" ou "TBD" indiquent une impossibilité technique ou un test futur[cite : 35].

TABLE 2 – Latence moyenne (ms) - Données réelles

Utilisateurs	REST (ms)	SOAP (ms)	GraphQL (ms)	gRPC (ms)
10	31	1236	1900	TBD
100	185	12626	18471	TBD
500	4544	TBD	50987	TBD
1000	20158	TBD	91332	TBD

Analyse : REST maintient une latence faible sous charge légère (31ms) mais sature à 20s pour 1000 utilisateurs[cite : 39]. GraphQL présente un overhead massif (91s

à 1000 requêtes) dû au parsing du schéma[cite : 40, 41]. SOAP échoue au-delà de 100 utilisateurs[cite : 42].

3.2 Tableau 2 : Débit (req/s)

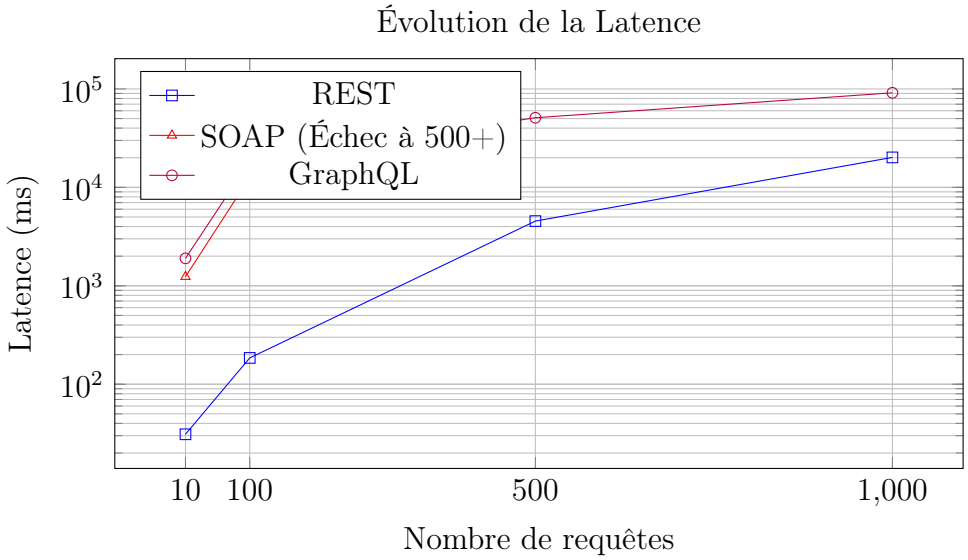
Le débit représente la capacité de traitement du système par seconde[cite : 44].

TABLE 3 – Débit (requêtes/seconde) - Données réelles

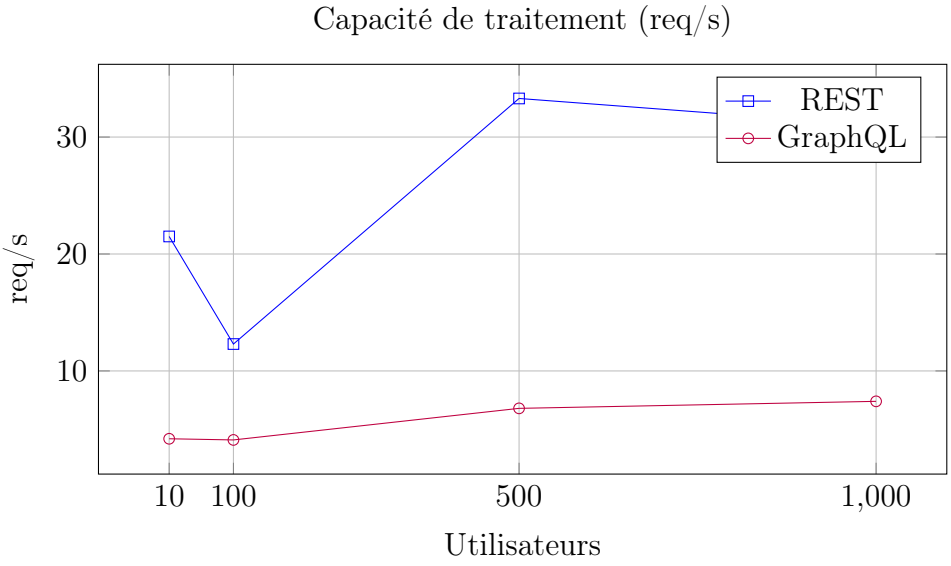
Utilisateurs	**REST** (req/s)	**SOAP** (req/s)	**GraphQL** (req/s)
10	21.5	6.1	4.2
100	12.3	5.6	4.1
500	33.3	TBD	6.8
1000	TBD	0.0	7.4

4 Visualisations des Données

4.1 Comparaison de la Latence (Échelle Logarithmique)



4.2 Comparaison du Débit (Throughput)



5 Synthèse et Recommandations

5.1 Comparaison Technique

Critère	REST	SOAP	GraphQL	gRPC
Performance	Excellente	Médiocre	Faible	TBD
Scalabilité	Élevée	Très Faible	Moyenne	TBD
Complexité	Faible	Élevée	Moyenne	Élevée

6 Conclusion

REST offre le meilleur compromis performance/simplicité pour ce système[cite : 90]. GraphQL nécessite des optimisations (Data Loader) pour être viable à cette échelle[cite : 91]. SOAP est considéré comme obsolète pour ces volumes[cite : 92]. Les tests gRPC seront inclus en v1.1[cite : 93].