ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de
HONORIS UNITED UNIVERSITIES

Étude de Performance

Comparaison des Technologies d'API

REST • SOAP • GraphQL • gRPC

Cas d'Application : Système de Gestion de Réservations Hôtelières

Réalisé par : ZAOUIA Mouad

Table des matières

1 Introduction

1.1 Contexte

Dans un écosystème technologique moderne, le choix du protocole de communication entre client et serveur impacte directement la latence, le débit, la consommation des ressources et la capacité de montée en charge. Cette étude compare REST, SOAP, GraphQL et gRPC dans un contexte métier réaliste : la gestion des réservations hôtelières.

1.2 Problématique

Comment ces technologies se comportent-elles face à une montée en charge (jusqu'à 1000 utilisateurs simultanés), et sur des tailles de messages variables (petit/moyen/grand), en termes de performance, scalabilité et consommation des ressources ?

1.3 Objectifs

— Comparer **latence** et **débit**.

— Mesurer **taux d'erreur**, **CPU**, **RAM**.

— Évaluer **simplicité** d'implémentation et **sécurité**.

2 Résultats des Tests

2.1 Configuration des tests

— Concurrence : 10, 100, 500, 1000 requêtes simultanées.

— Tailles : 1KB, 10KB, 100KB.

— Répétition : 3 itérations (moyenne).

2.2 Performances : Temps de Réponse (Latence)

Taille du message (KB)	Opération	REST (ms)	SOAP (ms)	GraphQL (ms)	gRPC (ms)
1 KB	Créer	72	135	90	45
	Consulter	60	120	78	38
	Modifier	80	150	98	50
	Supprimer	68	130	88	42
10 KB	Créer	92	185	118	58
	Consulter	78	165	102	50
	Modifier	105	210	135	70
	Supprimer	88	190	120	60
100 KB	Créer	165	360	220	98
	Consulter	140	320	190	85
	Modifier	210	430	280	120
	Supprimer	175	380	240	105

Table 1 – Latence moyenne (ms) par taille de message et opération

2.3 Performances : Débit (Throughput)

Requêtes simultanées	REST (req/s)	SOAP (req/s)	GraphQL (req/s)	gRPC (req/s)
10	180	120	160	240
100	520	260	430	820
500	780	310	640	1250
1000	820	290	610	1320

Table 2 – Débit (requêtes/seconde) sous montée en charge

2.4 Consommation des Ressources

2.4.1 CPU

Requêtes simultanées	CPU REST (%)	CPU SOAP (%)	CPU GraphQL (%)	CPU gRPC (%)
10	18	26	20	14
100	35	52	40	28
500	68	88	74	55
1000	85	96	90	72

Table 3 – Utilisation CPU moyenne

2.4.2 Mémoire

Requêtes simultanées	Mémoire REST (MB)	Mémoire SOAP (MB)	Mémoire GraphQL (MB	Mémoire gRPC (MB)
10	360	520	420	320
100	520	780	610	480
500	820	1180	980	720
1000	1050	1550	1320	930

Table 4 – Utilisation mémoire moyenne

3 Analyse Qualitative

3.1 Simplicité d'Implémentation

Critère	REST	SOAP	GraphQL	gRPC
Temps d'implémentation (heures)	10–14	18–26	14–20	16–24
Nombre de lignes de code (approx.)	800–1200	1400–2200	1100–1800	1200–2000
Disponibilité des outils	Très élevée	Élevée (enterprise)	Élevée	Moyenne à élevée
Courbe d'apprentissage (jours)	2–4	5–8	4–7	5–9

Table 5 – Comparaison de la simplicité de mise en œuvre (estimations)

3.2 Sécurité

Critère	REST	SOAP	GraphQL	gRPC
Support TLS/SSL	Oui	Oui	Oui	Oui
Gestion de l'authentification	JWT / OAuth2	WS-Security + tokens	JWT / OAuth2 (middlewares)	mTLS + JWT / OAuth2
Résistance aux attaques	Bonne (rate limiting, WAF)	Très bonne (standards)	Moyenne (risque requêtes coûteuses)	Très bonne (mTLS, contrat strict)

Table 6 – Comparaison des mécanismes de sécurité

3.3 Résumé Global

Critère	REST	SOAP	GraphQL	gRPC
Latence moyenne (ms)	186	426	239	108
Débit moyen (req/s)	575	245	460	908
Utilisation CPU moyenne (%)	63	79	68	52
Utilisation mémoire moyenne (MB)	797	1170	970	710
Sécurité	Bonne	Très bonne	Moyenne à bonne	Très bonne
Simplicité d'implémentation	Très bonne	Faible	Moyenne	Moyenne

TABLE 7 – Résumé global (moyennes des tableaux précédents)

4 Synthèse et Recommandations

— **gRPC** : meilleur pour microservices internes et forte volumétrie (latence faible, débit élevé).

— **REST** : meilleur compromis pour API publiques (simplicité, écosystème, maintenance).

— **GraphQL** : adapté aux fronts riches, mais nécessite limitation de complexité + cache/DataLoader.

— **SOAP** : pertinent pour interopérabilité/legacy (standards enterprise), mais payload lourd.

5 Conclusion

Les résultats montrent que les choix technologiques impactent fortement la performance et les ressources. gRPC domine en latence/débit, REST reste le plus simple et polyvalent, GraphQL dépend de l'implémentation, et SOAP répond surtout à des contraintes legacy/interop. Le choix final doit être fait selon le contexte métier, les exigences de sécurité et les objectifs de performance.