

Panorama : REST, SOAP, GraphQL et les autres

REST n'est pas la seule façon de construire une API. Voici les principales, avec le même besoin à chaque fois : **récupérer le livre n°3** d'une bibliothèque en ligne.

REST

Des **ressources** identifiées par des URL, manipulées avec les verbes HTTP. Le format est libre, JSON dans la grande majorité des cas.

```
GET /v1/livres/3 HTTP/1.1
Accept: application/json
```

```
{"id": 3, "titre": "Le Petit Prince", "auteur": "Antoine de Saint-Exupéry"}
```

Simple, lisible, compatible avec tout ce qui parle HTTP. C'est le sujet de ce cours.

SOAP

Un appel de fonction à distance, emballé dans une enveloppe XML, toujours en **POST**. Le contrat est décrit dans un fichier WSDL.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Body>
    <GetLivre>
      <Id>3</Id>
    </GetLivre>
  </soap:Body>
</soap:Envelope>
```

Encore très présent dans les banques, les assurances et les administrations.

Si un jour on vous confie une API SOAP, ne paniquez pas : c'est verbeux, mais c'est bien documenté par nature.

GraphQL

Une seule URL, et le client décrit **exactement** les champs qu'il veut. Créé par Facebook en 2015.

```
query {  
  livre(id: 3) {  
    titre  
    auteur  
  }  
}
```

Pratique quand les besoins du front varient beaucoup. Plus complexe à mettre en cache et à sécuriser.

gRPC

Appels de fonctions à distance très performants, en binaire (Protocol Buffers) sur HTTP/2. Créé par Google.

```
service Bibliotheque {  
  rpc GetLivre (LivreRequest) returns (Livre);  
}
```

Surtout utilisé entre microservices, rarement exposé au navigateur.

WebSocket

Une connexion qui reste **ouverte** dans les deux sens. Le serveur peut envoyer des messages quand il veut : chat, jeux, cours de bourse, notifications en direct.

Webhooks

L'inverse d'une API classique : « ne nous appelez pas, on vous appellera ». Vous donnez une URL au service, et c'est lui qui vous envoie une requête HTTP quand un événement se

produit (paiement reçu, commit poussé...).

Comparatif

	Format	Transport	Cas d'usage typique
REST	JSON (souvent)	HTTP	API publiques, back de SPA et d'apps mobiles
SOAP	XML	HTTP (POST)	Systèmes d'entreprise historiques
GraphQL	JSON	HTTP (une seule URL)	Fronts complexes, données très liées
gRPC	Binaire (Protobuf)	HTTP/2	Communication entre microservices
WebSocket	Libre	Connexion persistante	Temps réel
Webhooks	JSON (souvent)	HTTP, dans l'autre sens	Notification d'événements

Revision #3
Created 2 October 2026 20:00:40 by Thibaud FRICHET
Updated 2 October 2026 21:00:59 by Thibaud FRICHET