CRITEO

GraphQL: La fin de REST ?

Sunny Tech

Henry LAGARDE

Software Engineer

https://github.com/tolares/talks

Sommaire

1. Révisons notre histoire
2. GraphQL... Pourquoi faire ?
3. Implémentation
4. Prenons du Recul

Révisons notre histoire

CRITEO

Un peu d'histoire

Les années 80

IBM

- SNA – Systems network architecture
- LU6 – Application to application

Digital Equipment Corp

- DecNet Digit Equipment Corp.
- Networking Channel – Application to application

Wang Laboratories

- WSN – Wang System Networking
- Remote Port – Application to application

Unix – Everyone had a one

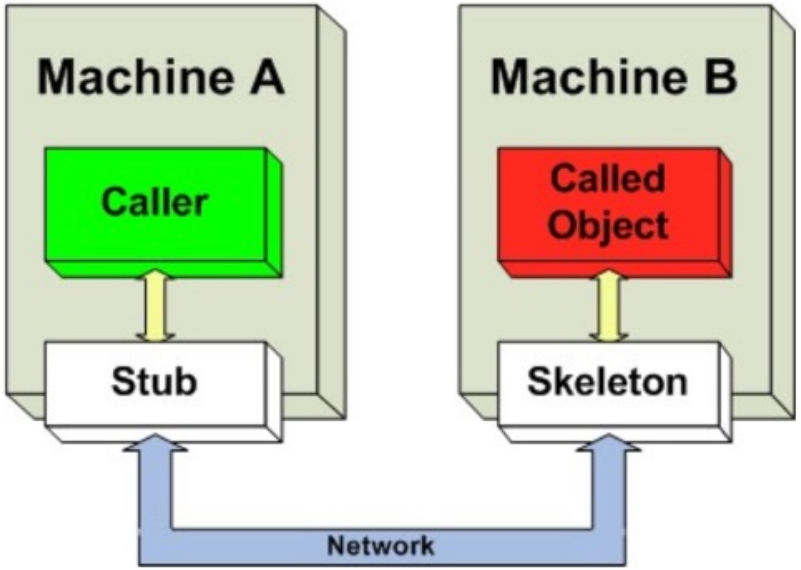
- TCP/IP
- Sockets – Application to application

Communication d'application vers d'autres application (L6/L7)

Un peu d'histoire

RPC (fin 80s)

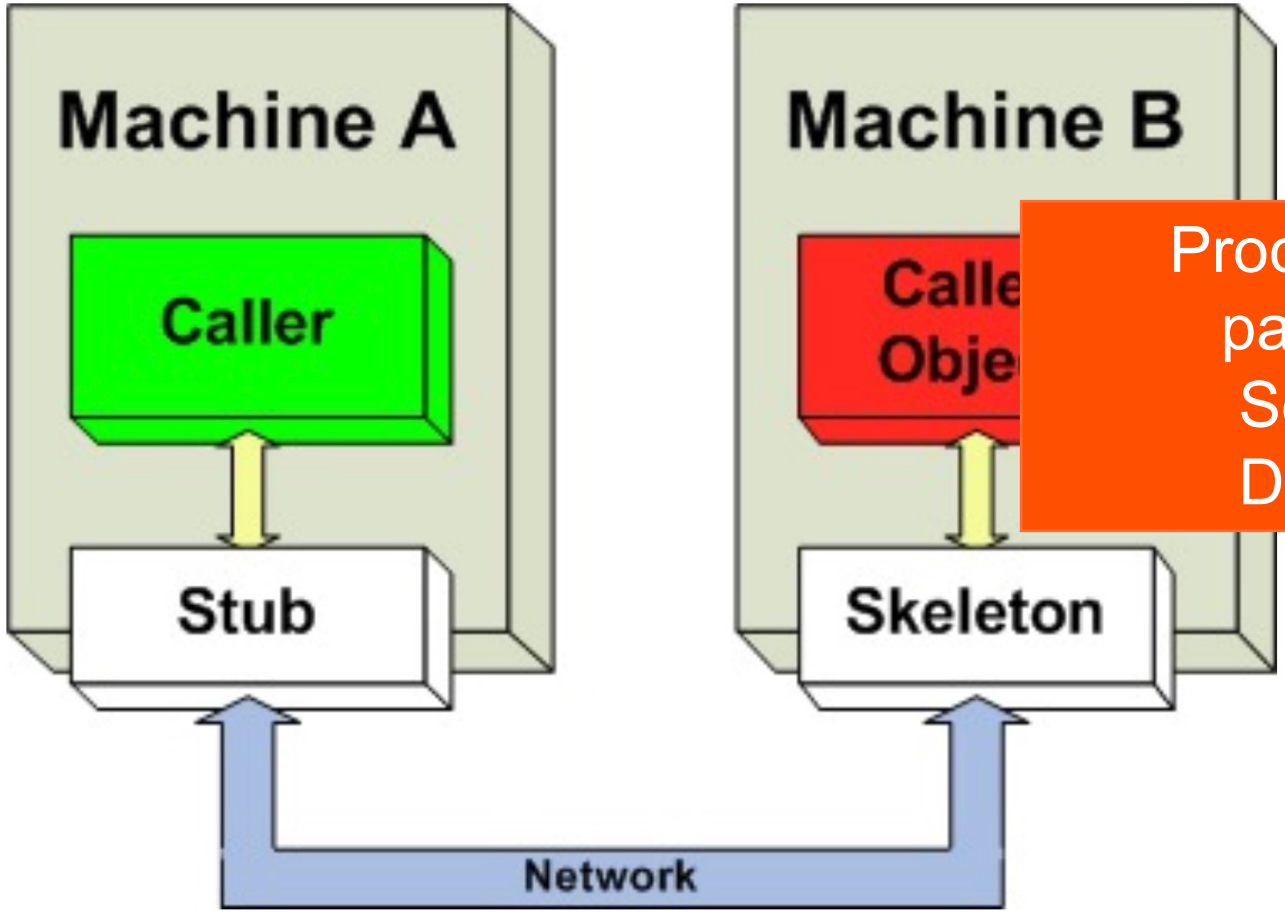
- Remote Procedure Call (RPC)
  - Protocole réseau permettant de faire appels à des procédures sur un ordinateur distant
  - Modèle Client-Serveur
  - Conception des micro-noyaux



CRITEO

Un peu d'histoire

RPC (fin 80s)



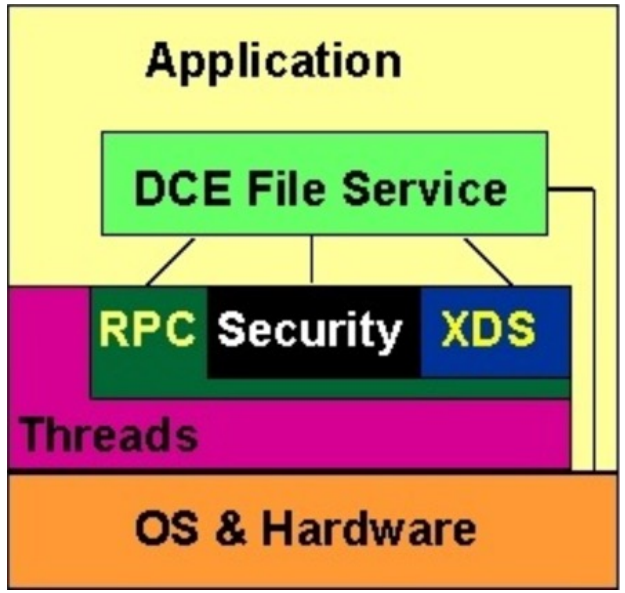
Procedure/Function paradigme avec Sérialisation et Déserialisation

CRITEO

Un peu d'histoire

DCE (début 90s)

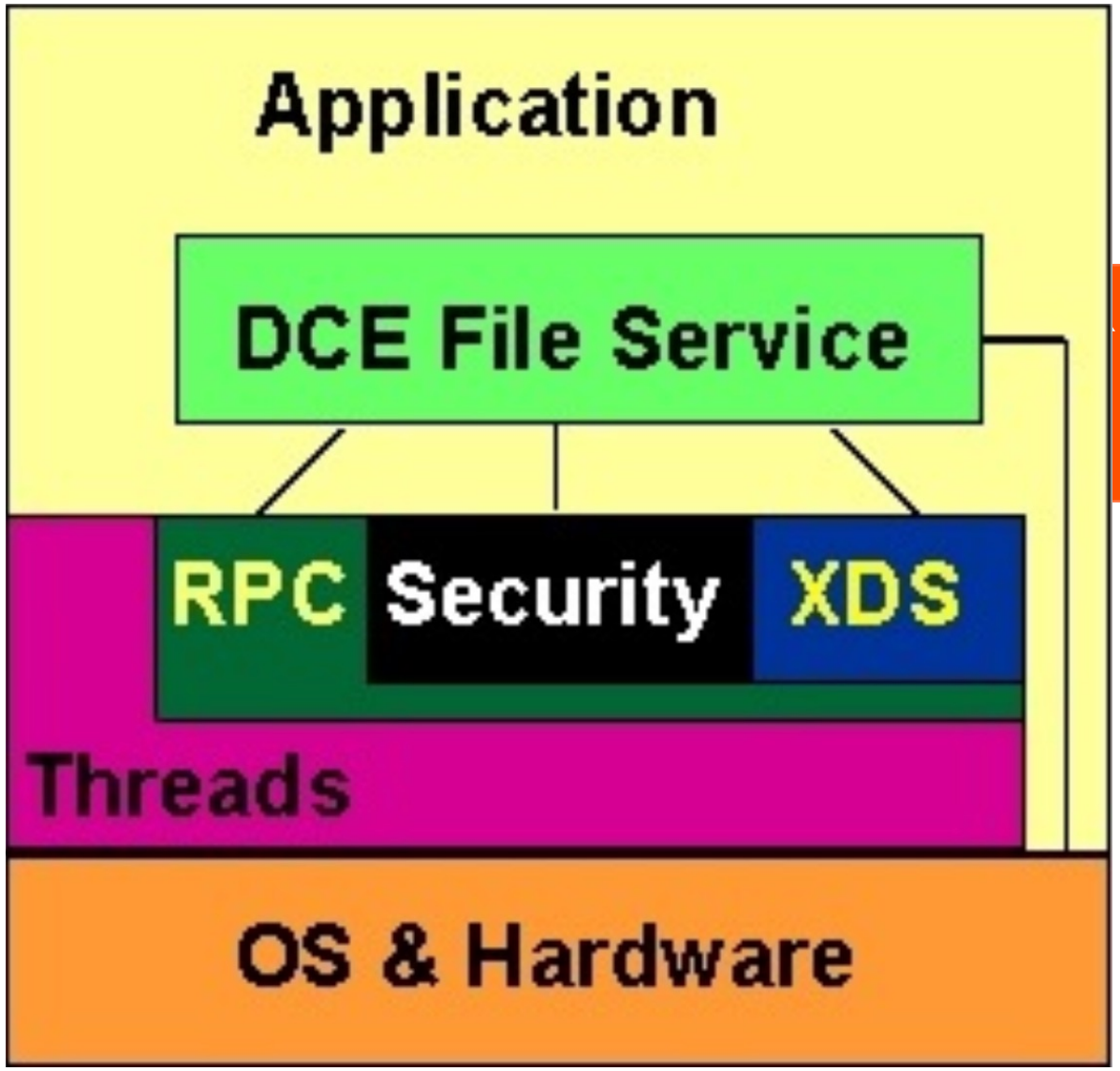
- Distributed Computing Environment (DCE)
  - Middleware développé par un consortium
  - Fournit un cadre et des outils pour développer une application client-serveur
    - RPC (DCE/RPC)
    - Répertoire
    - Service d'authentification
    - Système de fichier distribué (DCE/DFS)



CRITEO

Un peu d'histoire

DCE (début 90s)



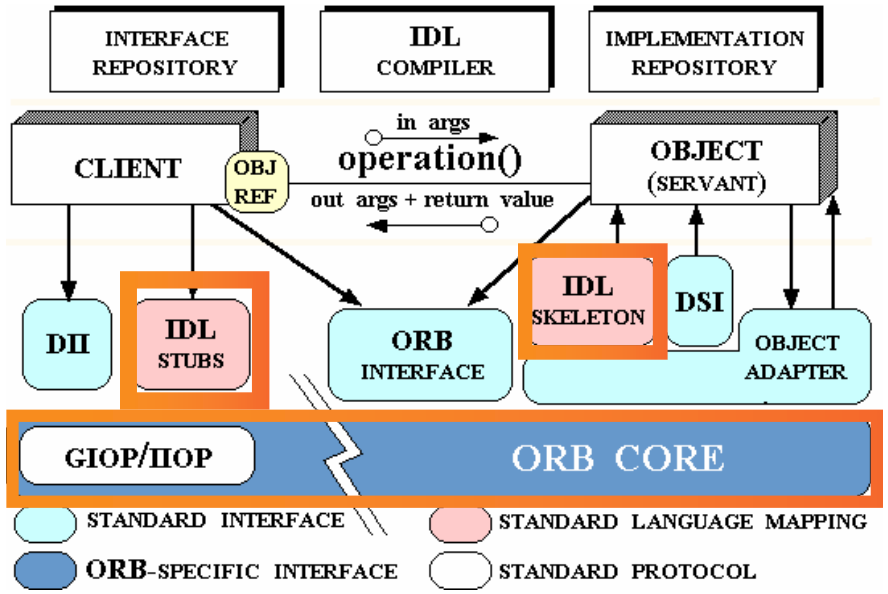
PC standardisé
+
Services

CRITEO

Un peu d'histoire

CORBA (90s)

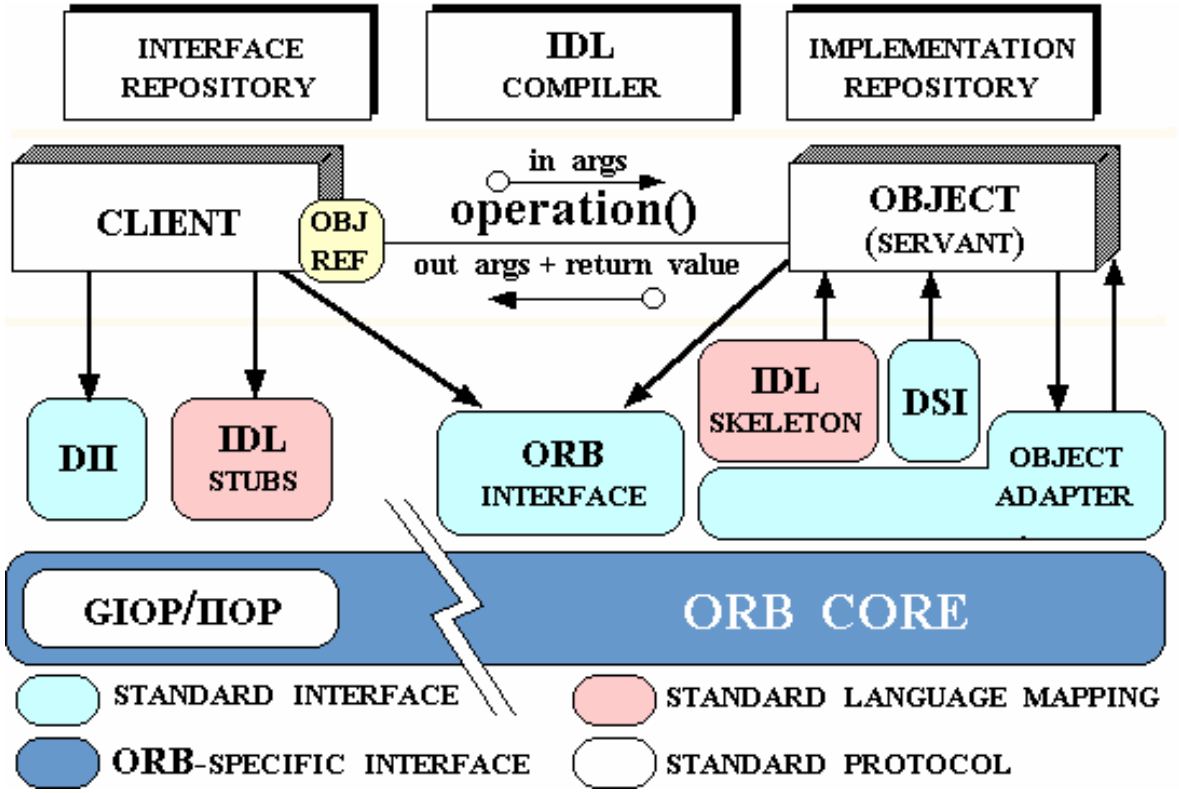
- Common Object Request Broker Architecture (CORBA)
  - Standard défini par Object Management Group
  - Facilite la communication de systèmes déployés sur différente platformes, language, hardware
  - Utilise un model orienté objet même si les systèmes qui l'utilise non pas besoin d'être orienté objet.



CRITEO

Un peu d'histoire

CORBA (90s)



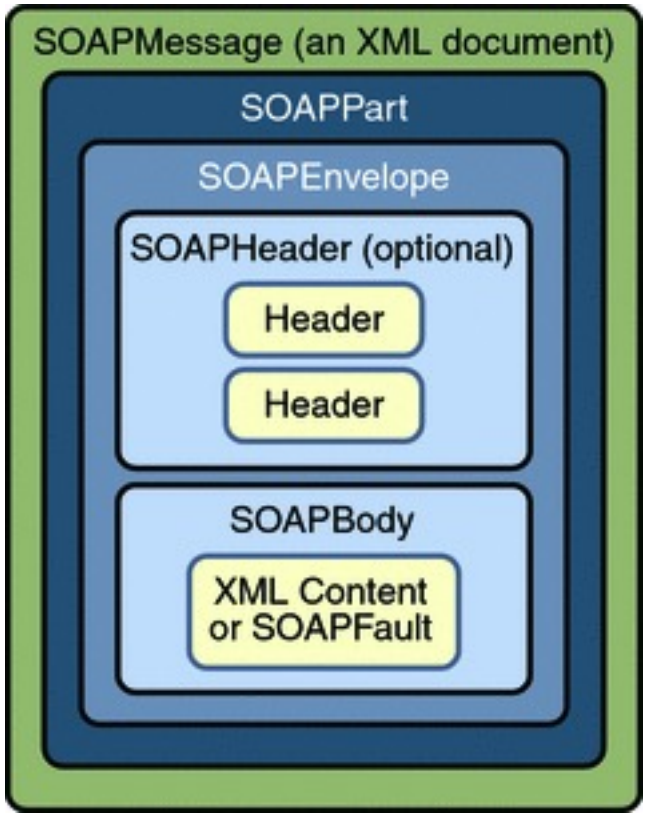
Objets distribués
Services
IDL

CRITEO

Un peu d'histoire

SOAP (00s)

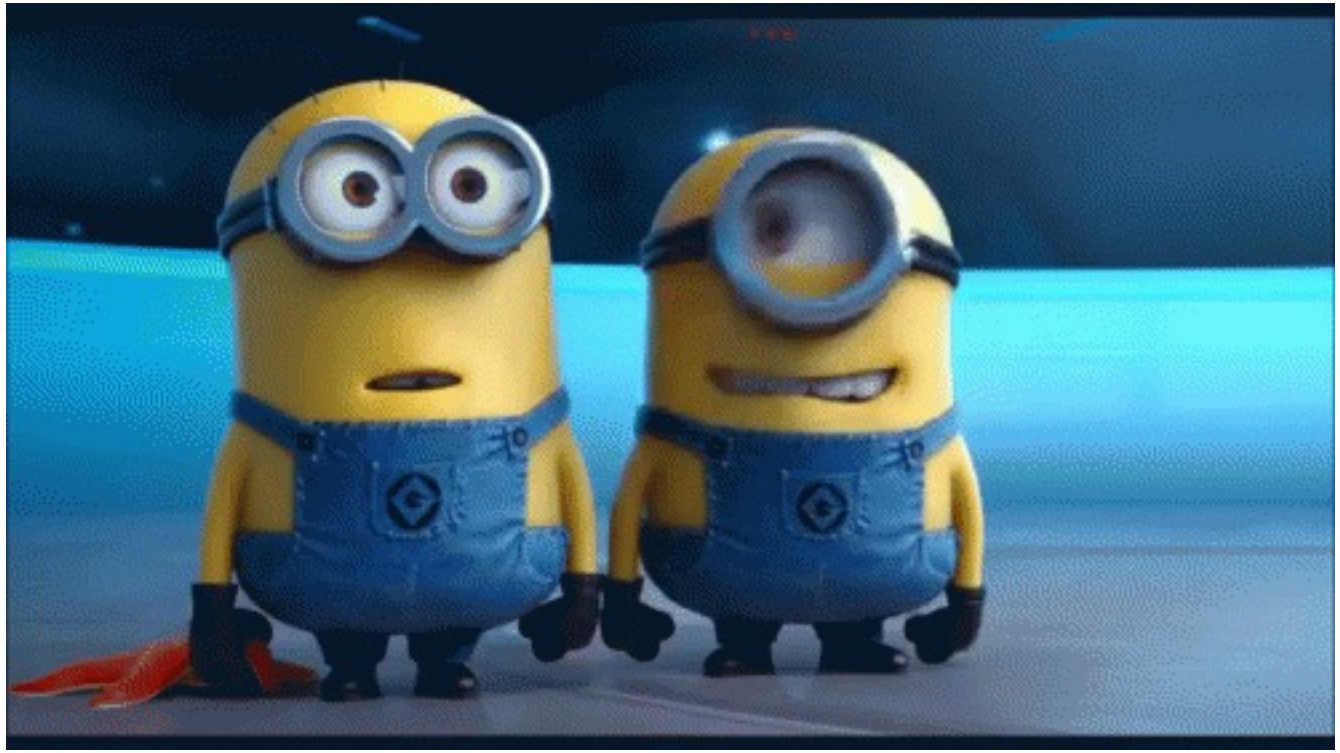
- Simple Object Access Protocol (SOAP)
  - protocole d'échange d'information structurée dans l'implémentation de services web bâti sur XML.
  - SOAP se veut:
    - Flexible
    - Indépendant de la platforme ou du language
    - Extensible
  - SOAP utilise HTTP ou SMTP pour la négotiation et la transmission de message

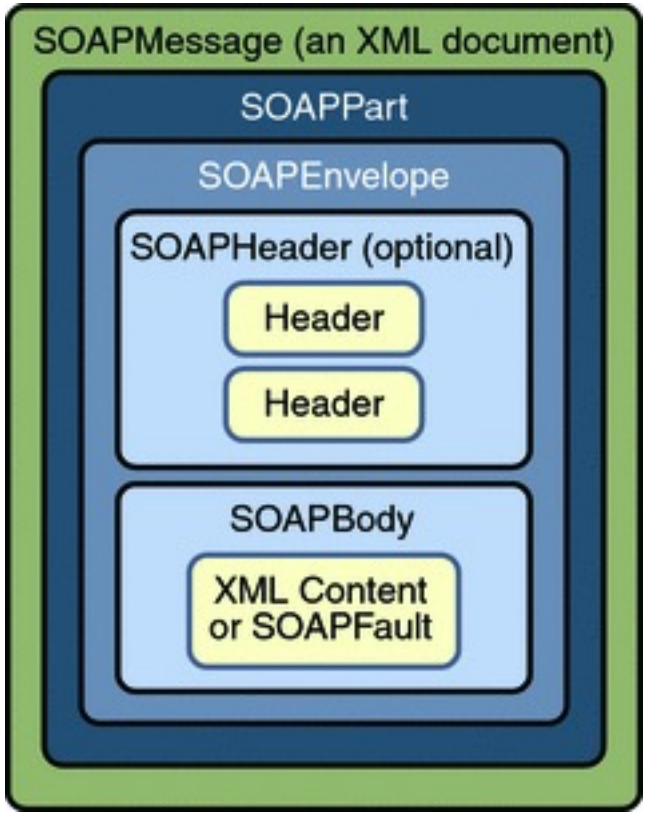


Un peu d'histoire

SOAP (00s)

NOT Simple Object Access Protocol

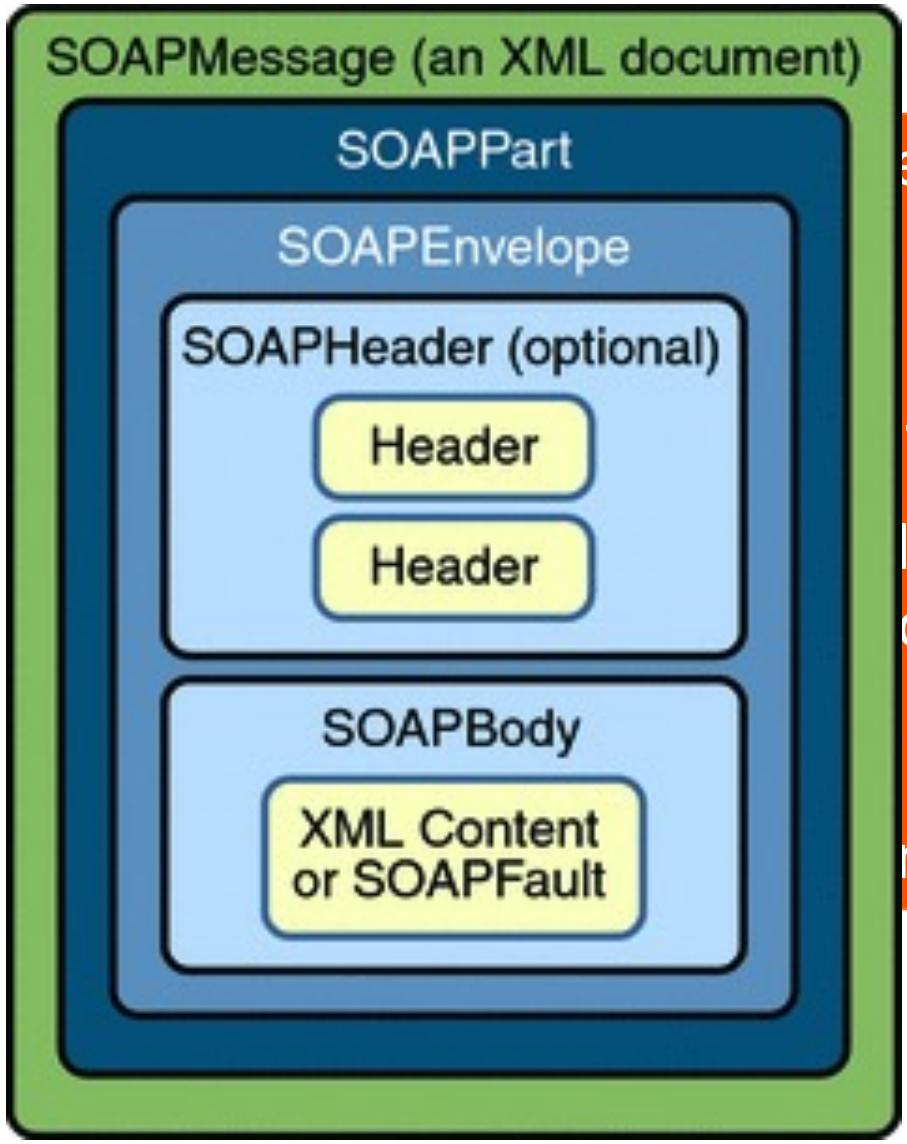




CRITEO

Un peu d'histoire

SOAP (00s)



e problèmes de proxy et
pare-feu

MAIS SURTOUT

lage fort Serveur/Client
ormat Lourd (XML)
Migraines
Nausée
n sur ses choix de carrière

CRITEO

Un peu d'histoire

{ REST } (00s)

- Representational State Transfer (REST)
  - Ensemble de contraintes afin de créer des services web
  - Les services RESTful permet:
    - Interopérabilité entre les ordinateurs
    - Opérations uniforme et prédéfinies sans état
    - Représentation textuelles des ressources

Un peu d'histoire

{ REST } (00s)

Un peu d'histoire

{ REST } (00s)

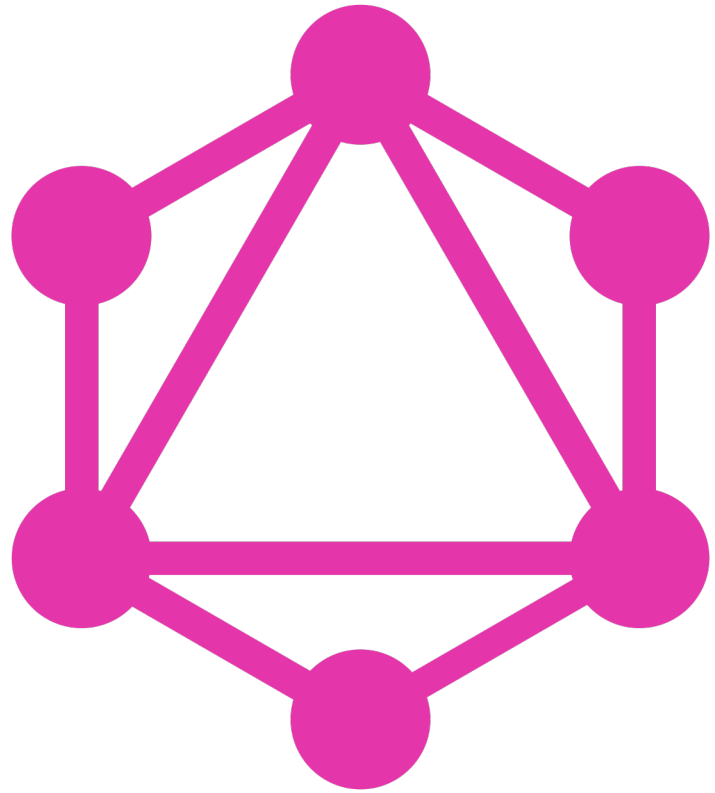
CRUD operations simplifié
MAIS finalement
Du SOAP 🤡

CRITEO

GraphQL... Pourquoi faire ?

GraphQL... Pourquoi faire ?

Histoire de GraphQL



GraphQL protocol

- Création en 2012
- **Open sourced** par Facebook en 2015

GraphQL Foundation

- Created in 2019 hosted by **The Linux Foundation**

Protocole ou Dialect ?

GraphQL... Pourquoi faire ?

Protocol

Operations:

- Query
- Mutations
- Subscriptions (Distributed Events)

Se repose sur des requêtes POST avec un body

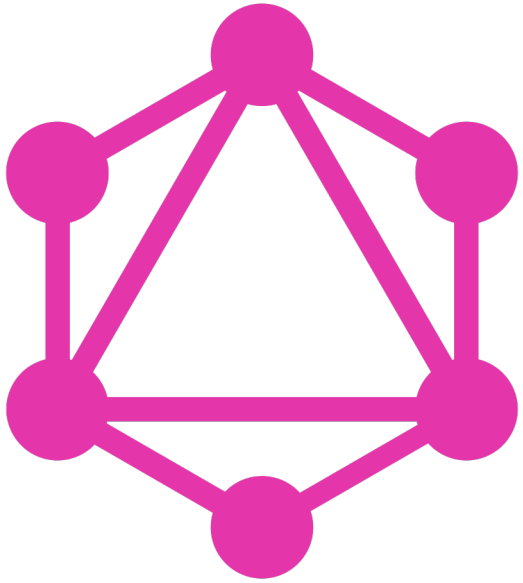
```
{
  "query": "query { allFilms { films { id title episodeID } } }"
}
```

GraphQL... Pourquoi faire ?

Protocol

Requête POST

```
{
  "query": "query { allFilms { films { id title episodeID } } }"
}
```



Response Body


```
{
  "data": {
    "allFilms": {
      "films": [
        {
          "id": "ZmlsbXM6MQ==",
          "title": "A New Hope",
          "episodeID": 4
        },
        {
          "id": "ZmlsbXM6Mg==",
          "title": "The Empire Strikes Back",
          "episodeID": 5
        },
        {
          "id": "ZmlsbXM6Mw==",
          "title": "Return of the Jedi",
          "episodeID": 6
        },
        {
          "id": "ZmlsbXM6NA==",
          "title": "The Phantom Menace",
          "episodeID": 1
        },
```


CRITEO

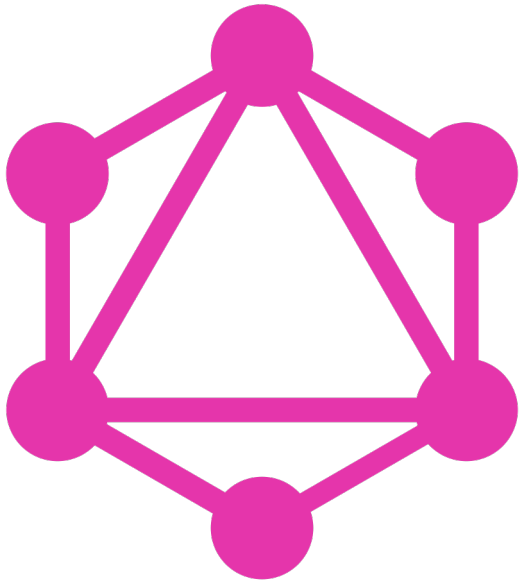
Schema declaration

GraphQL… Pourquoi faire ?

Schema

GraphQL Schema

- Types
- Relations
- Documentation
- Operations



GraphQL Schema Definition Language

GraphQL... Pourquoi faire ?

Schema Types

Type – Read model

Input – Write model

Enum

Scalar – Int, Float, String, ID, etc

Union

Interface – Polymophisme

CRITEO

GraphQL... Pourquoi faire ?

Schema Relationships

Field Types:

- Optional
- Exactly one
- Many

```
type Elephant implements Animal @key(fields: "id") {
  id: ID!
  name: String!
  head: Head
  trunk: Trunk
}
```

```
type Head {
  tail: Tail
  eyes: [Eye]
  ears: [Ear]
  body: Body
}
```

GraphQL... Pourquoi faire ?

Schema Operations

Type d'opérations:

- Query
- Mutation
- Subscription

```
type Query {
    animal(id: ID!): Animal
    animals(animalIds: [ID!]
        offset: Int = 0
        limit: Int = 1000
    ): [Animal]
    elephant(id: ID!): Elephant
    elephants(elephantIds: [ID!]
        offset: Int = 0
        limit: Int = 1000
    ): [Elephant]
}
```

```
type Mutation {
    newElephant(elephant: NewElephant): Elephant!
    updateElephants(
        elephant: [UpdateElephant]!
    ): [ID]
}
```

```
type Subscription {
    elephantCreated(elephant: ID!): Elephant!
}
```

GraphQL… Pourquoi faire ?

Resolvers

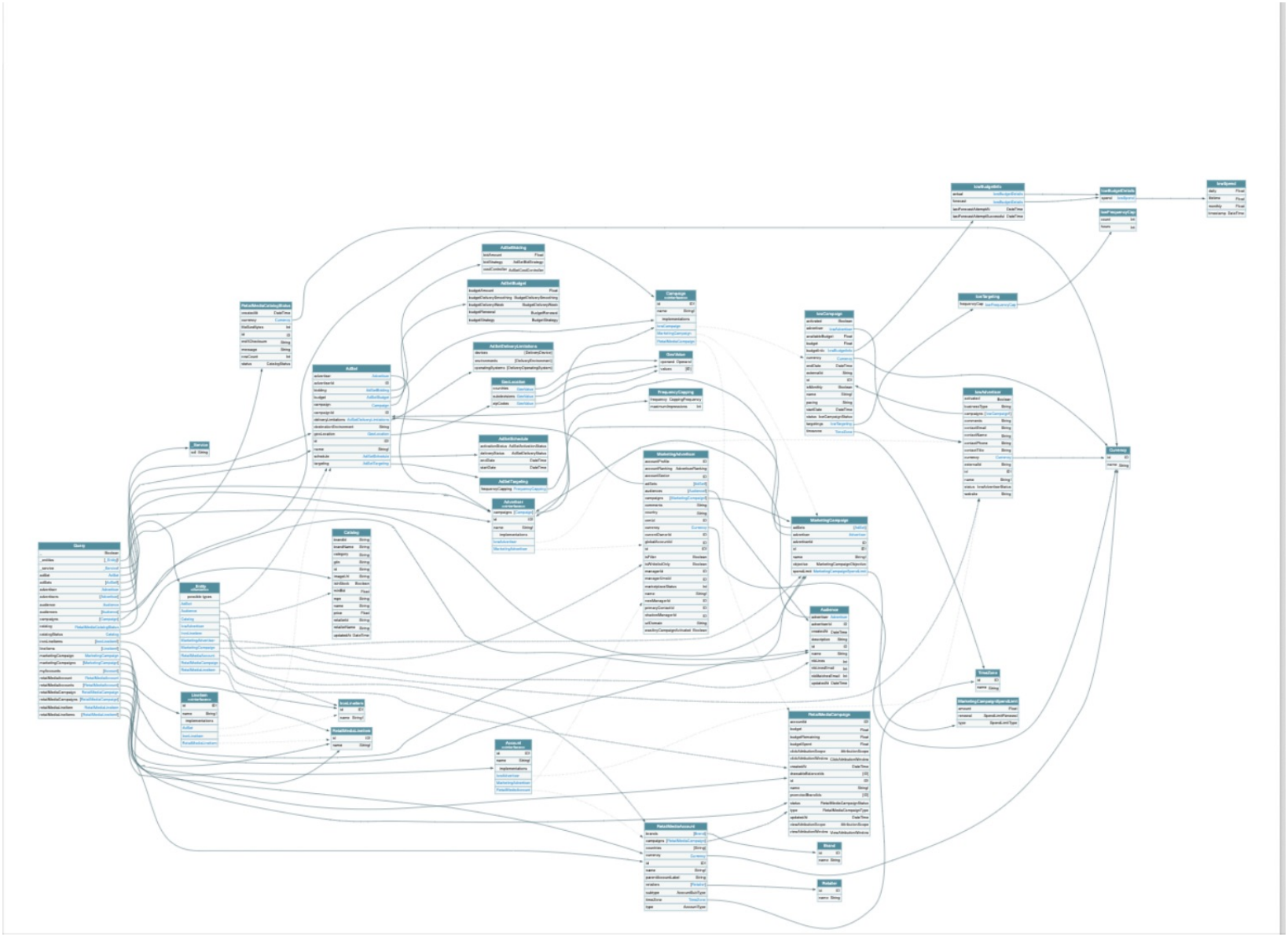
Chaque champs à :

- Resolver par défaut
- Une fonction afin d'agir comme le resolver

```
const animals = async (
  _: any,
  args: QueryAnimalsArgs,
  context: Context,
  info: any
): Promise<Animal[]> => {
  const results = await Promise.all([
    ((await context.dataSources.elephants) as ElephantAdapter).get_all(
      args.ids,
      args.offset,
      args.limit
    ),
    ((await context.dataSources.bear) as BearAdapter).get_all(args.ids),
  ]);
  return [...results[0], ...results[1]];
};
```

Cononical Model

Canonical Model



CRITEO

Quels sont les soucis de REST ?
Rien sauf …

GraphQL… Pourquoi faire ?

Quels sont les soucis de REST ? Rien sauf …

Beaucoup d’endpoints – Internes et Externes

Criteo – 500+

Beaucoup de versions

500 x nombre de versions

Facebook avait entre 90-100 versions par endpoints!

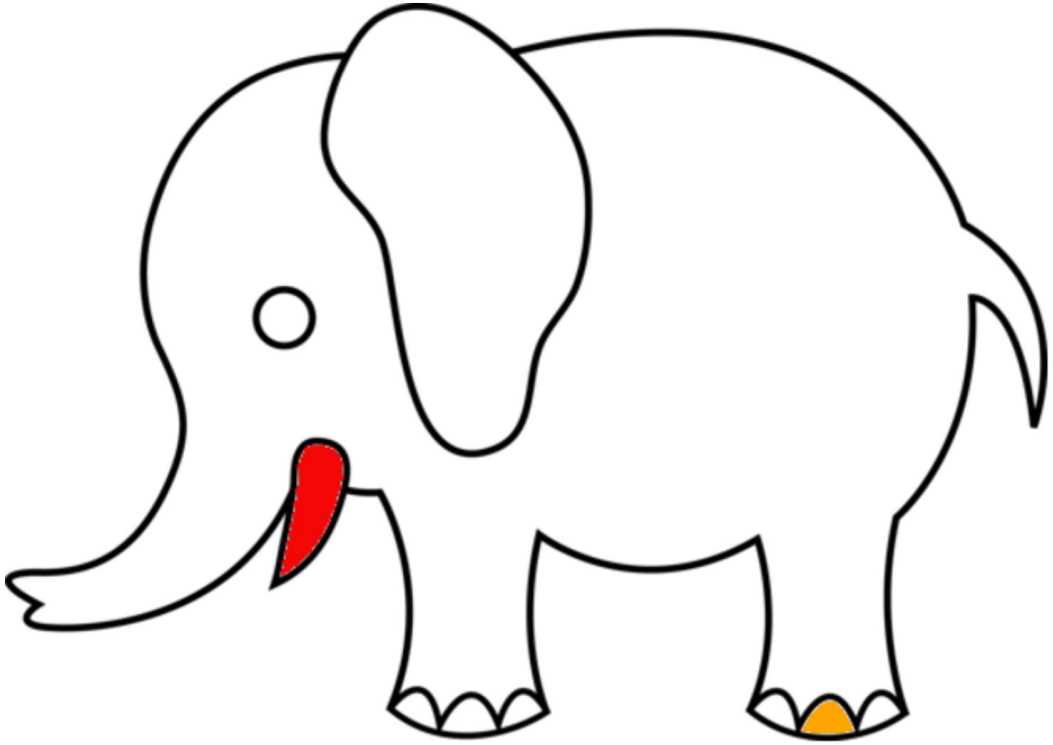
500
x 10

5000

GraphQL… Pourquoi faire ?

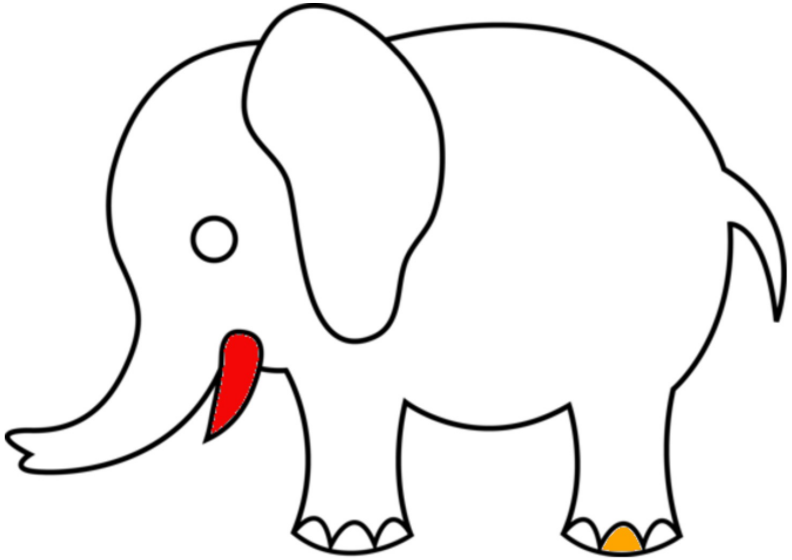
Quels sont les soucis de REST ? Rien sauf …

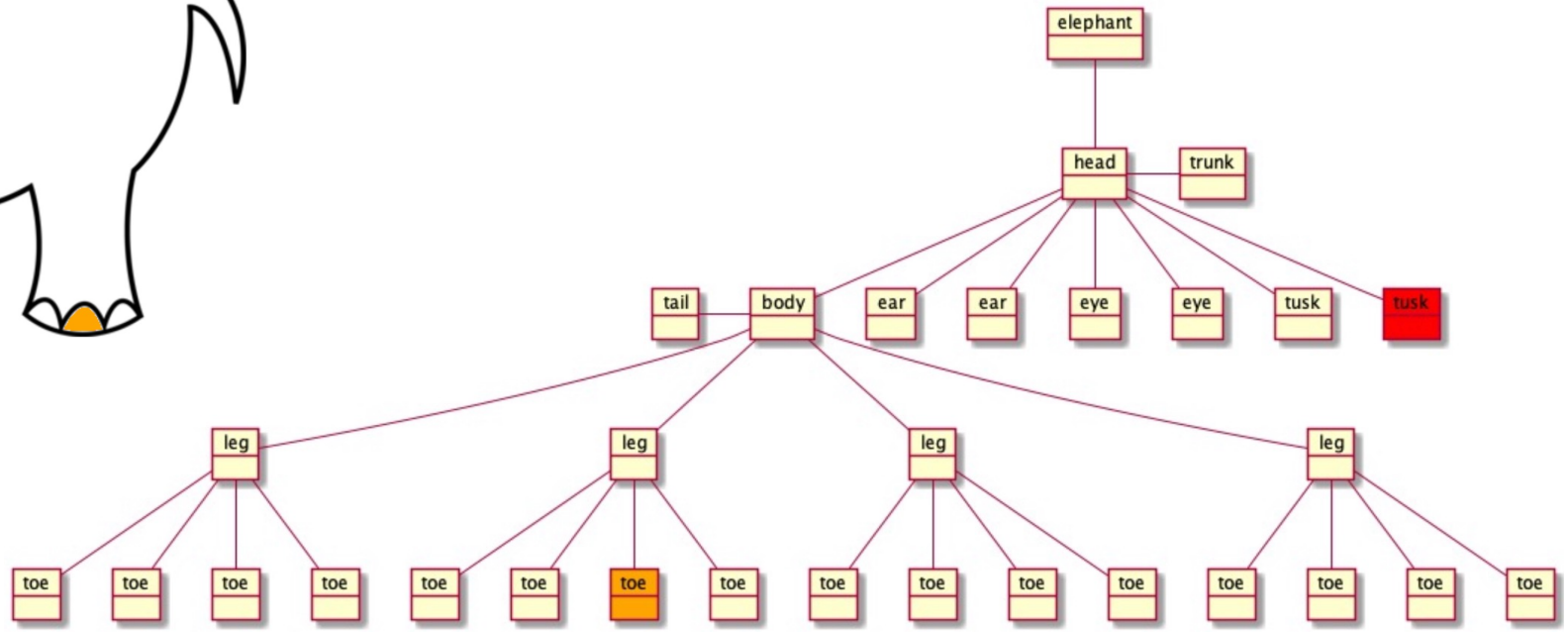
On récupère tout l'objet quand on en as besoin que d'un portion



GraphQL... Pourquoi faire ?

Quels sont les soucis de REST ? Rien sauf ...



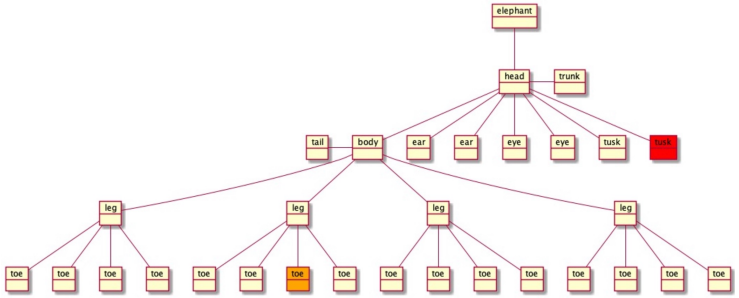


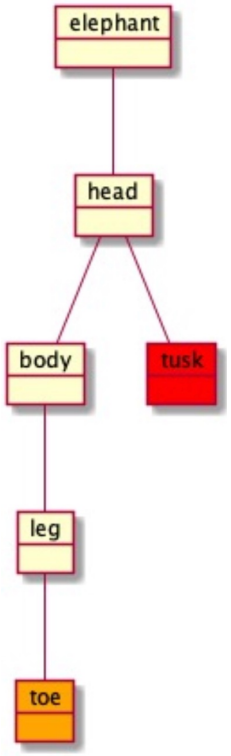
CRITEO

GraphQL... Pourquoi faire ?

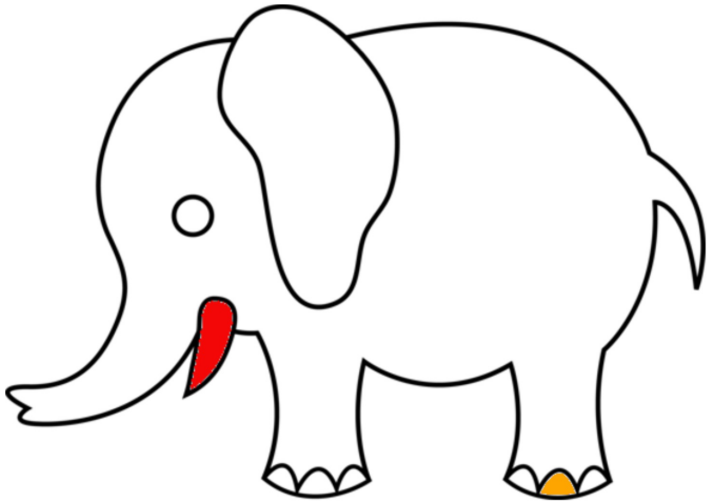
Quels sont les soucis de REST ? Rien sauf ...

Queried object Graph





Only what you want



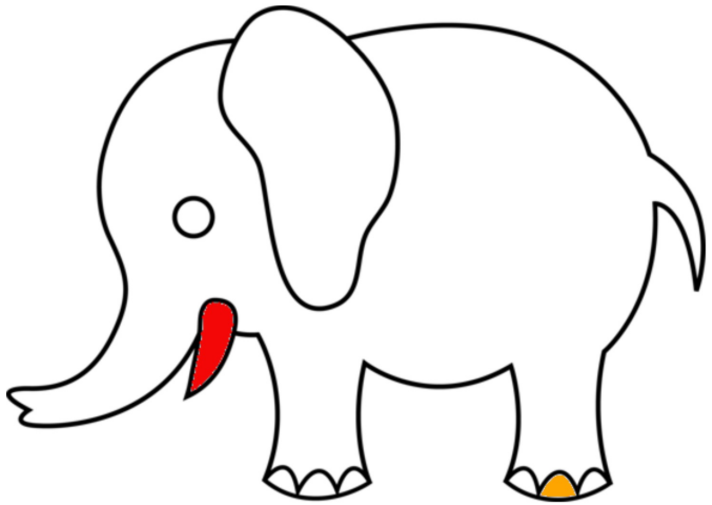
Ask for what you want

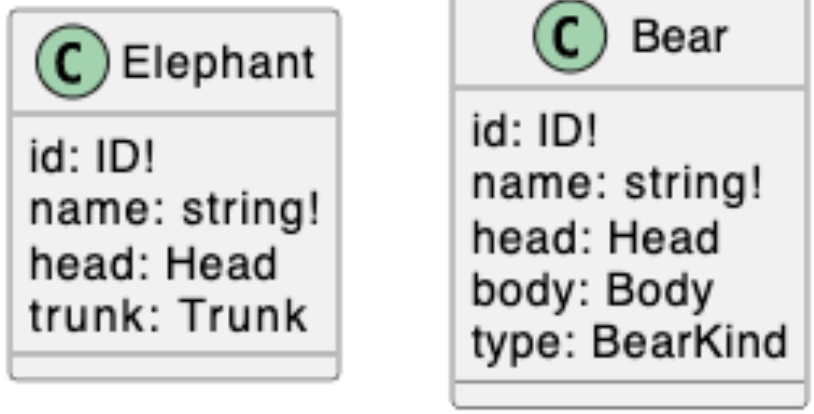
CRITEO

GraphQL... Pourquoi faire ?

Quels sont les soucis de REST ? Rien sauf ...

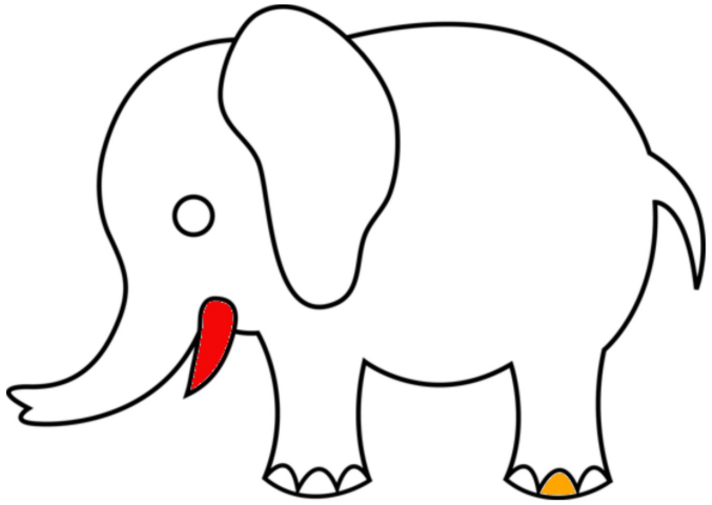
Plusieurs endpoints utilisent le même type d'object avec des potentielles inconsistences



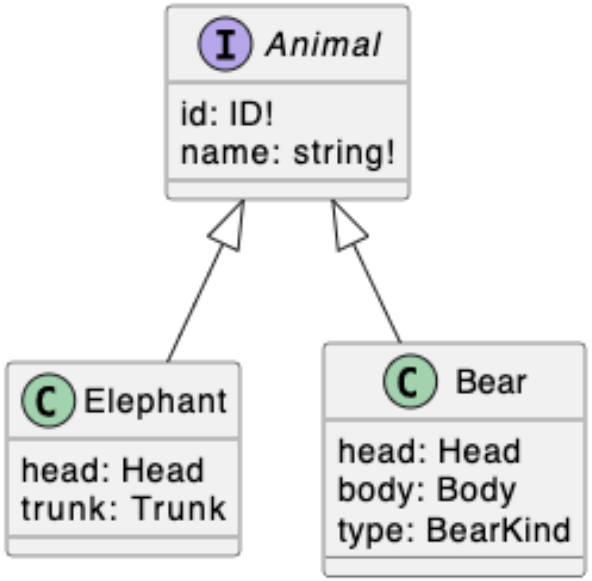


GraphQL... Pourquoi faire ?

Quels sont les soucis de REST ? Rien sauf ...



Interface polymorphism

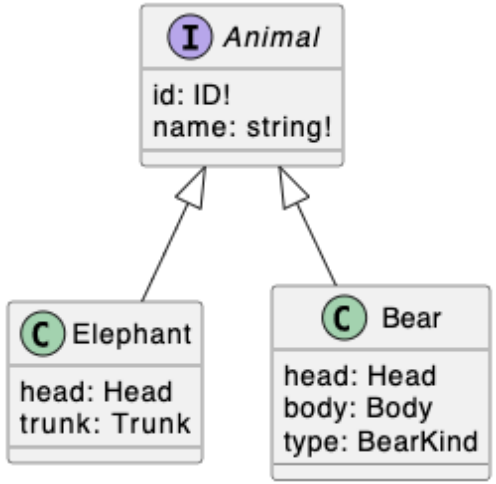


CRITEO

GraphQL… Pourquoi faire ?

Quels sont les soucis de REST ? Rien sauf …

```
query Animals {
  animals {
    id
    name
    __typename
  }
}
```

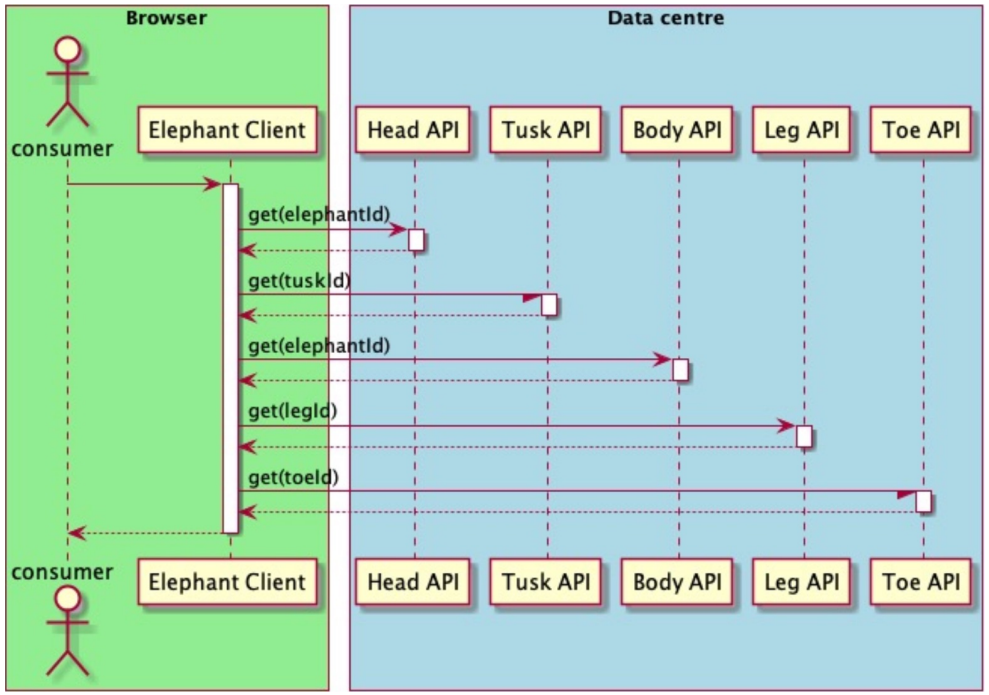



```
{
  "data": {
    "animals": [
      {
        "id": 12,
        "name": "Super Elephant",
        "__typename": "Elephant"
      },
      {
        "id": 14,
        "name": "Super second Elephant",
        "__typename": "Elephant"
      },
      {
        "id": 15,
        "name": "Super Elephant 3",
        "__typename": "Elephant"
      },
      {
        "id": 20,
        "name": "Grizzly",
        "__typename": "Bear"
      },
      {
        "id": 21,
        "name": "Brown Bear",
        "__typename": "Bear"
      }
    ]
  }
}
```

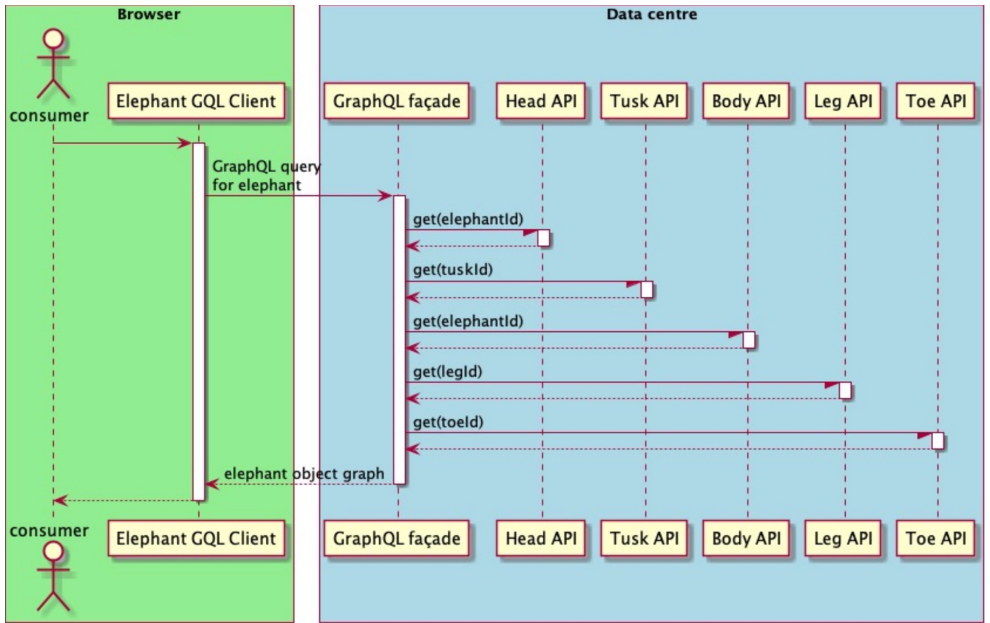
Utilisation du réseau

5 appels REST effectué par le Client

Payload de retour contenant l'ensemble de la data



1 appel GraphQL
5 appels interne au sein du Datacenter
Payload de retour contenant uniquement l'élément souhaité



CRITEO

Ask what you want
And
Only what you want

CRITEO

GraphQL... Pourquoi faire ?

Pourquoi utiliser GraphQL

- Plus d'un seul client (ex: Web + iOS)
- Microservices Architecture
- API REST sont devenue complexe limitant les ajouts de features.
- Séparation frontend et backend

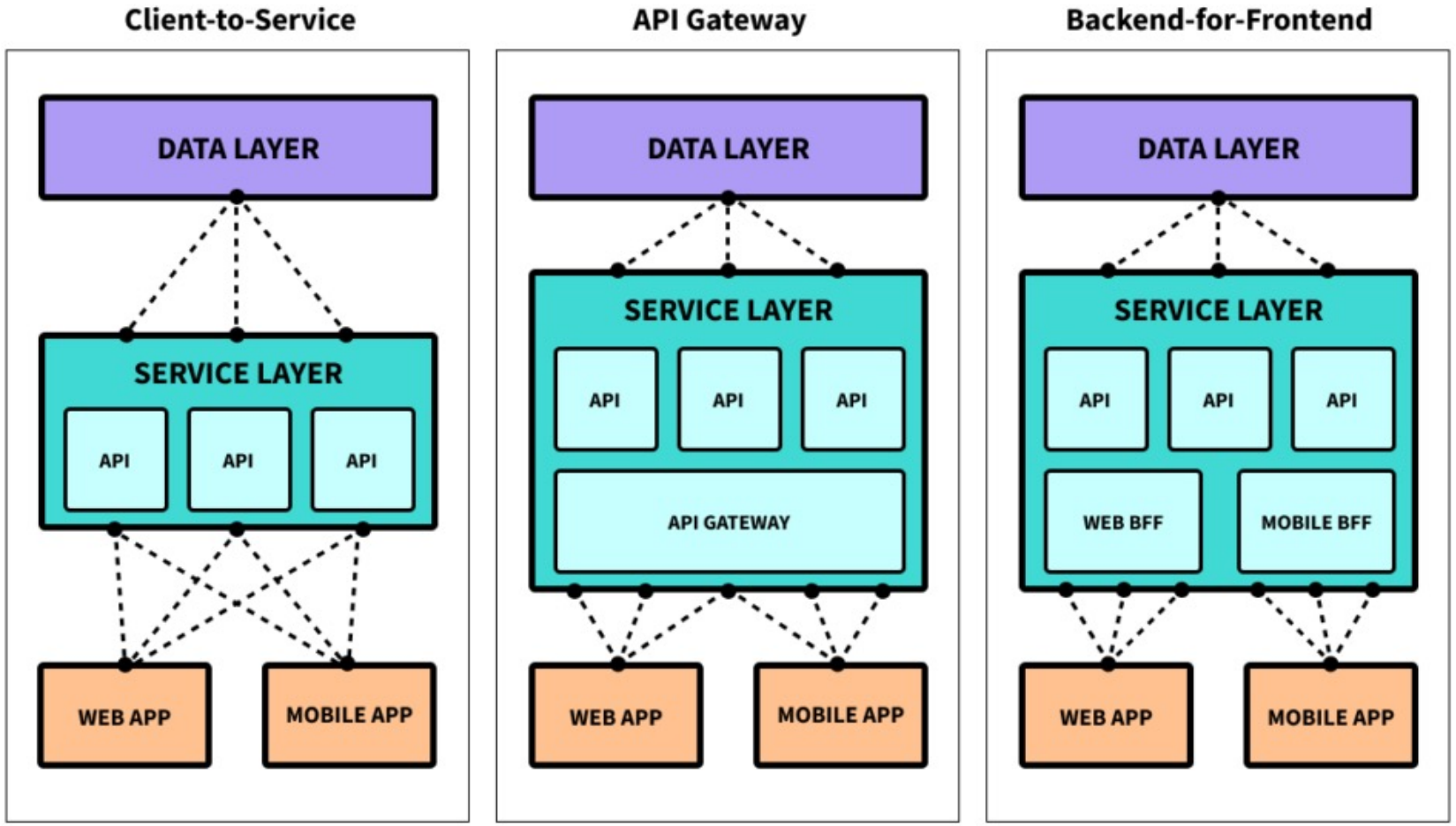
- Un endpoint unique
- Flexibilité: Récupération uniquement de la data souhaitée
- Aucun soucis de versionning
- Les endpoints restent consistant entre les vues
- Deprecation au niveau des schemas

Implémentation

Serveur

Implementation

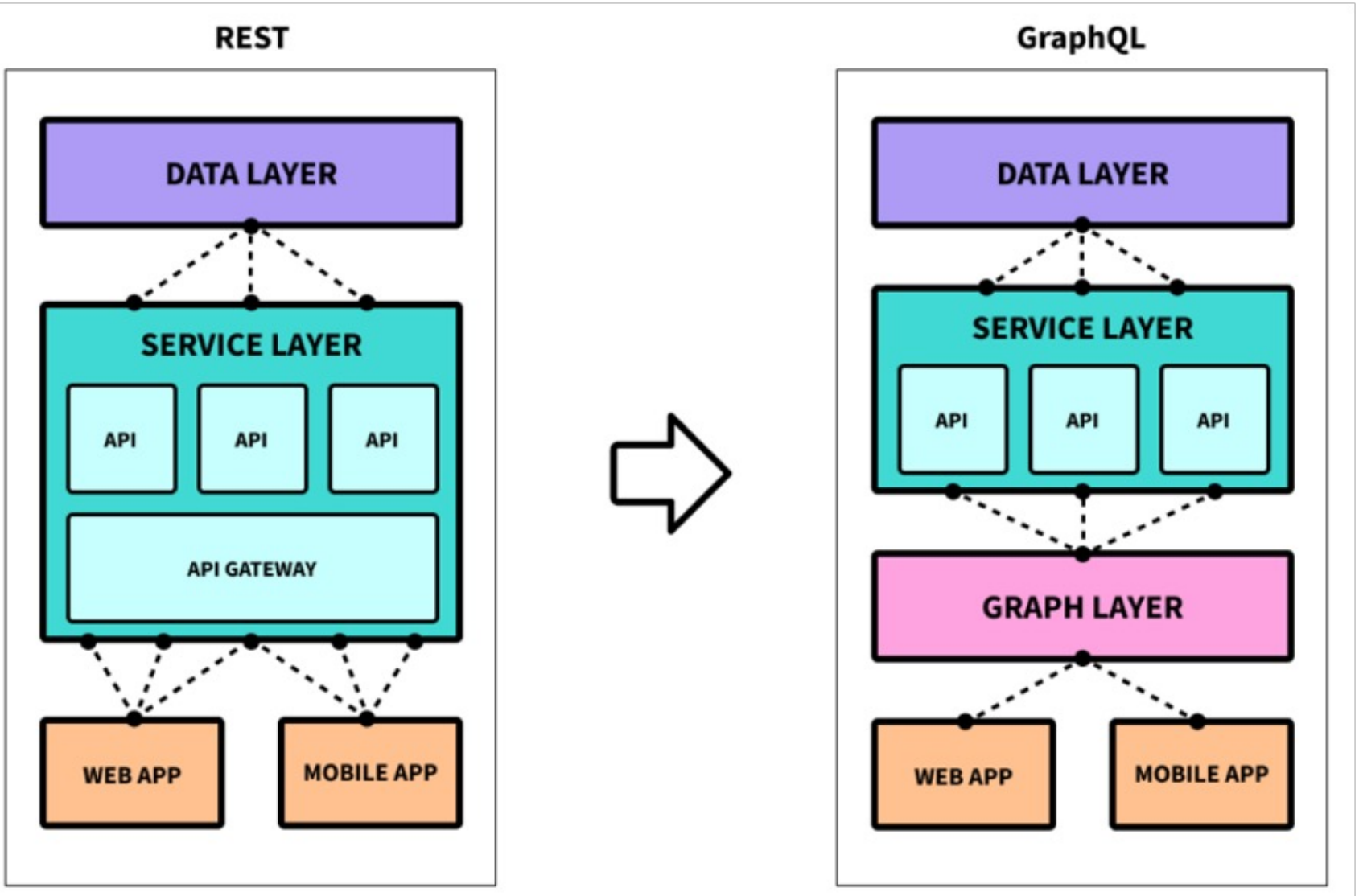
Serveur



CRITEO

Implementation

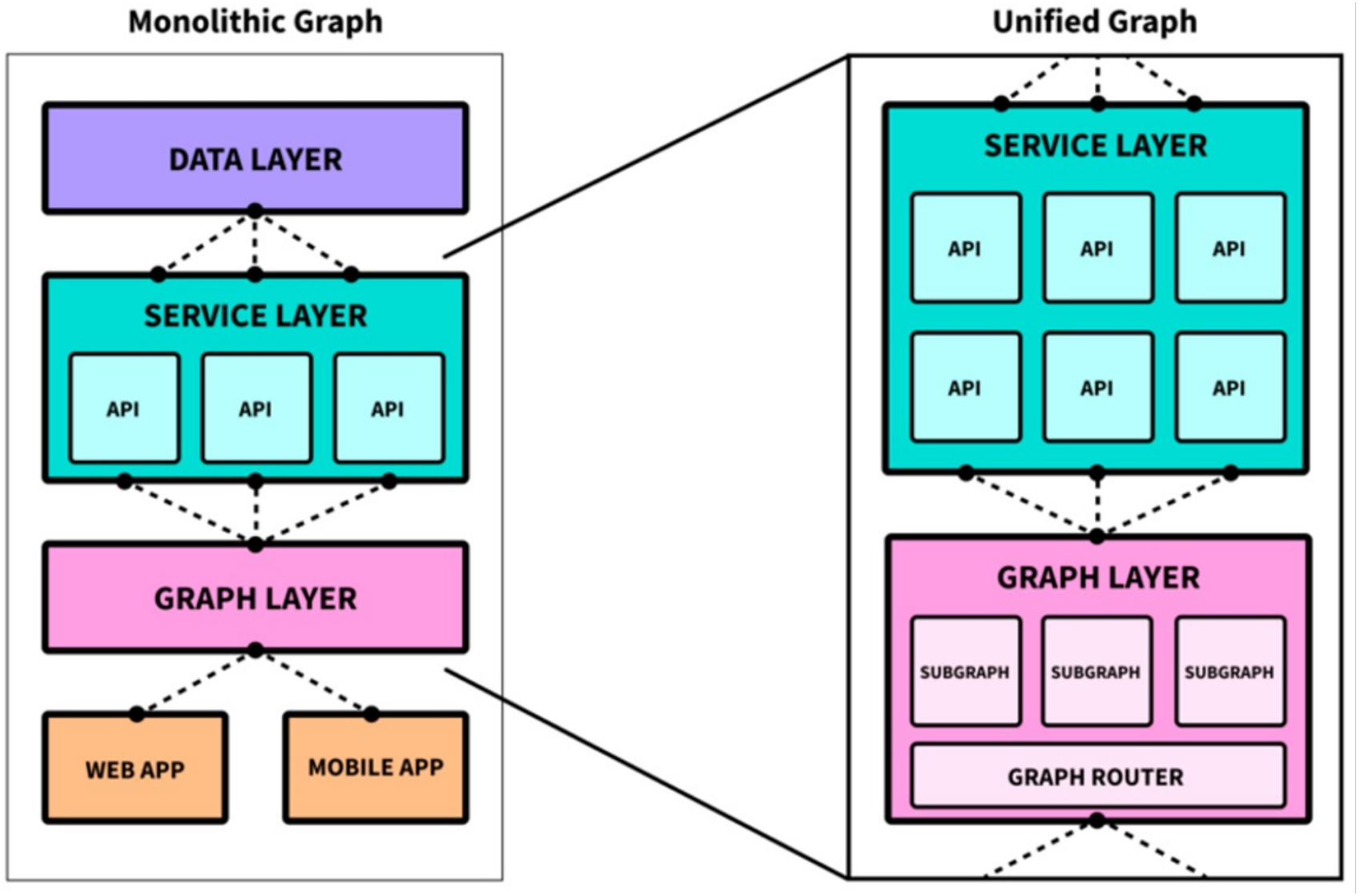
Serveur



CRITEO

Implementation

Façade

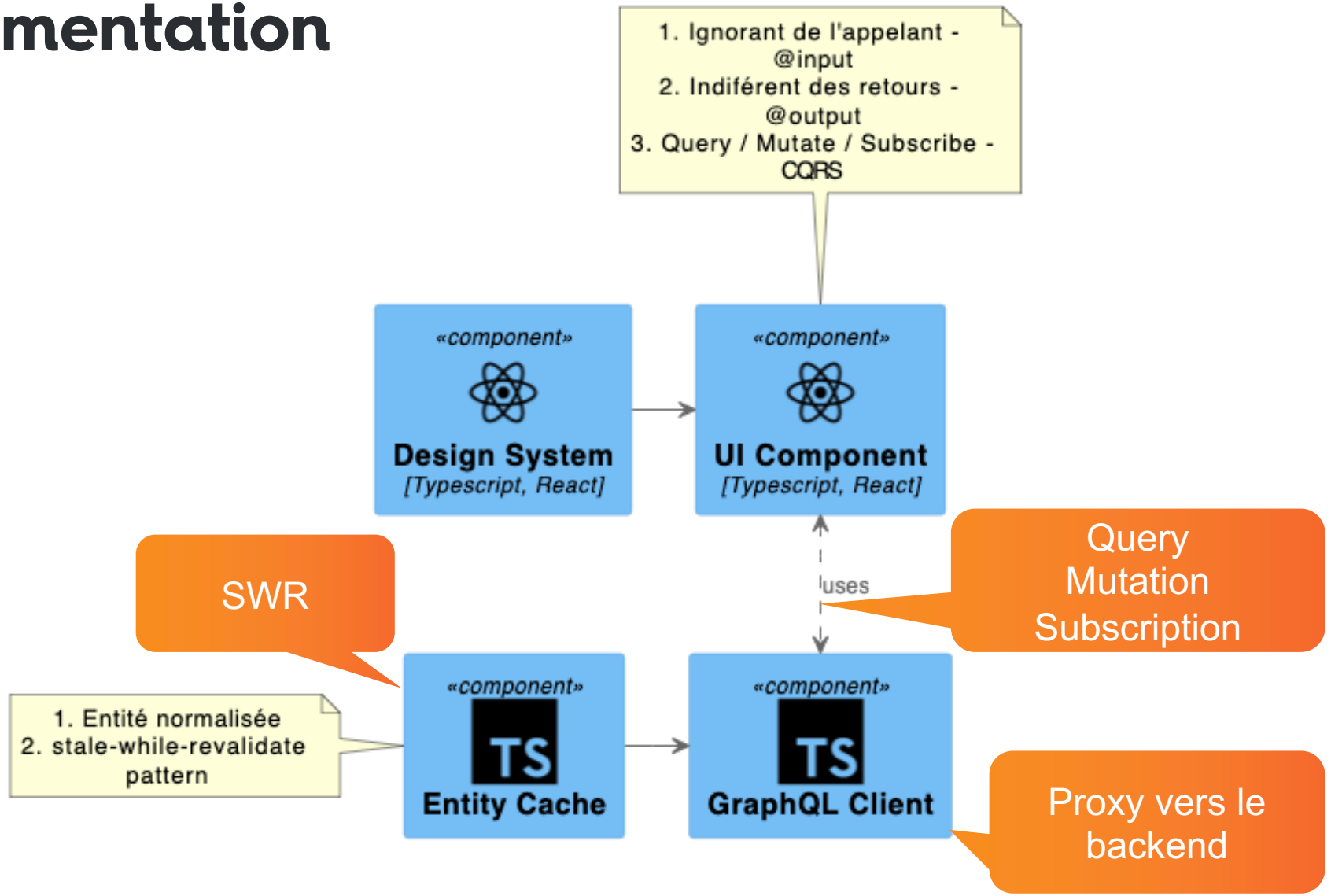


CRITEO

Client

Implémentation

Client

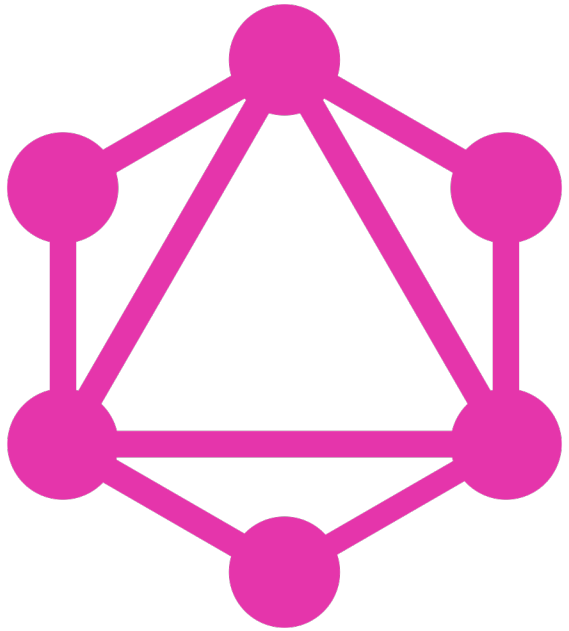


CRITEO

Implémentation

Client

- Développement FrontEnd simplifié
  - Dialecte Unique pour la communication Client-Serveur
  - Endpoint Unique
  - Separation of concern



Réutilise, régule et agrèges les APIS

Implémentation

Outils disponible

Qui l'utilise ?

GraphQL, la solution magique ?

GraphQL, la solution magique ?

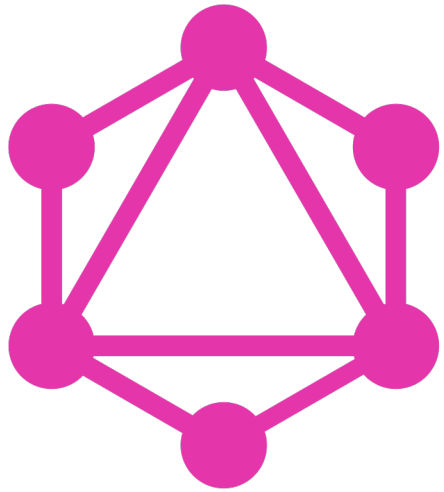
Schema First design

Canonical Model – Contrat et source de vérité

Optimisation des appels réseaux

Résolution des soucis d'over/under fetching

Cache normalisé



GraphQL, la solution magique ?

Si vous étiez mauvais à :

REST, vous le serez avec GraphQL

SOA, vous le serez avec GraphQL

Si votre API ou design est mauvais GraphQL ne fixera pas tout.

GraphQL, la solution magique ?

GraphQL est à REST ce que REST était à SOAP

REST est à SOAP ce que SOAP était à CORBA

SOAP est à CORBA ce que CORBA était à DCE

CORBA est à DCE ce que DCE était aux Sockets

Les langages sont incrémentaux

GraphQL est au calcul distribué
ce que SQL a été au Data Storage

CRITEO

GraphQL, la solution magique ?

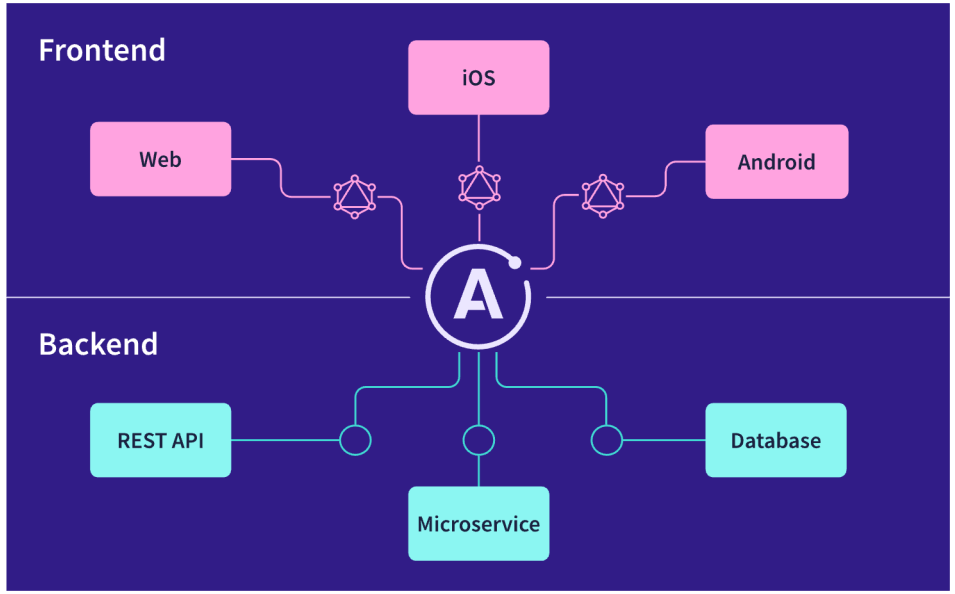
- Avant SQL, on devait gérer:
  - File Location
  - Index & Data Block
  - Compaction
  - Backup
  - Transactions
  - Joins

GraphQL, la solution magique ?

- SQL:
  - Schema: DDL
  - Queries: DML
  - Compaction, index, joins selective, alias etc...

GraphQL, la solution magique ?

- GraphQL:
  - Schema: Schema Definition Language
  - Queries: Queries, mutations, Subscriptions
  - Relationships, selective columns, aliasing etc...



CRITEO


Learn yet another framework
Lose my job

CRITEO

Questions?

CRITEO