# Labo 1 – Protocoles applicatifs mobiles

## Manipulation 1

***Quel gain, en volume et en temps, peut-on constater en moyenne sur les données échangées (xml, json et aussi protobuf) en utilisant la compression mise en place au point 1.4 ?***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre de données** | **JSON** | | | | | | | |
| **NONE** | | | | **DEFLATE** | | | |
| **Recieved** | **Payload** | **Temps 2G** | **Temps 5G** | **Recieved** | **Payload** | **Temps 2G** | **Temps 5G** |
| 3 | 460 | 460 | 300 | 100 | 200 | 460 | 150 | 90 |
| 10 | 2000 | 2000 | 500 | 250 | 400 | 2000 | 380 | 200 |
|  |  |  |  |  |  |  |  |  |
| **Nombre de données** | **XML** | | | | | | | |
| **NONE** | | | | **DEFLATE** | | | |
| **Recieved** | **Payload** | **Temps 2G** | **Temps 5G** | **Recieved** | **Payload** | **Temps 2G** | **Temps 5G** |
| 3 | 500 | 500 | 350 | 130 | 260 | 500 | 160 | 100 |
| 10 | 1800 | 1800 | 650 | 130 | 500 | 1800 | 200 | 130 |
|  |  |  |  |  |  |  |  |  |
| **Nombre de données** | **Protobuf** | | | | | | | |
| **NONE** | | | | **DEFLATE** | | | |
| **Recieved** | **Payload** | **Temps 2G** | **Temps 5G** | **Recieved** | **Payload** | **Temps 2G** | **Temps 5G** |
| 3 | 90 | 90 | 110 | 100 | 80 | 90 | 110 | 90 |
| 10 | 400 | 400 | 240 | 110 | 300 | 400 | 180 | 100 |

On peut constater que quand les données sont compressées, à la réception il y bien une réduction notable de la taille des données reçues :

* JSON :
  + Avec 3 données une réduction d’environ 50%
  + Avec 10 données une réduction d’environ 80%
* XML :
  + Avec 3 données une réduction d’environ 50%
  + Avec 10 données une réduction d’environ 75%
* Protobuf :
  + Avec 3 données une réduction d’environ 10%
  + Avec 10 données une réduction d’environ 25%

Concernant les temps d’exécution entre 2G et 5G nous pouvons constater que les différences sont surtout notables en 2G qu’en 5G en JSON et XML (environ 50%) alors qu’en 5G les gains de temps ne sont pas très notables. Les gains de temps obtenus avec le protobuf sont aussi plus faibles.

***Est-ce utile de compresser dans tous les cas ?***

Non, ce n’est pas utile de le faire dans les tous les cas. On peut notamment remarquer dans le cas du protobuf où les gains de temps ou de tailles sont très minimes (de 90 à 80 bytes et pratiquement le même temps pour 3 éléments). On peut donc se rendre compte que la compression serait plus judicieuse d’utiliser dans des grands sets de données surtout en JSON et XML. C’est dans ces deux formats de données que nous avons remarqué les plus grandes réductions. Cependant, dans notre cas, même avec des réductions de 50% ou plus les tailles sont minuscules (gain de 260 bytes et 60 ms pour le JSON en 3 éléments) c’est négligeable comme gain absolu.

## 2.1 Question théorique

L’api graphql mise à disposition aurait besoin d’une fonction qui retourne les livres d’un author selon son ID. Malheureusement elle n’est pas disponible et donc nous devons utilisé d’autre fonction pour avoir ce résultat.

J’ai aussi remarqué que le type de données pour les dates était un String ce qui n’est pas optimale mais fonctionne bien dans ce cas de figure.

Il serait intéressant que la pagination soit intégrée au requêtes.

Les fonctions countAuthors et countBooks ne sont pas indispensable mais sont un petit +.

Autrement j’ai bien aimé la simplicité des requêtes.