

## **Choix Technique - WEB SERVICE**

### ➤ A quoi sert un WEB SERVICE ?

Pour rester simple, c'est une interface de distribution de données, qui peut répondre à plusieurs demandes HTTP et exposer différentes données venant de différentes sources (plusieurs bases de données, lire des fichiers ... ).

Il existe 2 grandes familles :

- type SOAP (protocole)
- type REST (style d'architecture orientée ressources)

### ➤ Le REST, le SOAP, c'est quoi ?

**REST** (Representational State Transfer) est un style d'architecture permettant de construire des applications (Web, Intranet, Web Service). L'architecture **REST** utilise les spécifications originelles du protocole **HTTP**.

Les différentes opérations possible (**CRUD**) sont :

- x Create : **POST**
- x Read : **GET**
- x Update : **PUT**
- x Delete : **DELETE**

Le serveur renverra les opérations acceptées sur une ressource via l'entête **HTTP Allow**.

**SOAP** est un protocole de transmission de messages.

Lien : <http://www.soapuser.com/fr/basics1.html>

La différence majeure entre ces 2 éléments est le degré de liaison entre le client et le serveur. Un client développé avec le protocole SOAP ressemble à un logiciel d'ordinateur, car il est étroitement lié au serveur. Si une modification est effectuée d'un côté ou de l'autre, l'ensemble peut ne plus fonctionner. Il faut effectuer des mises à jour du client s'il y a des changements sur le serveur et vice-versa.

Un client de type **REST** sait utiliser un protocole et des méthodes standardisées. Son application doit rentrer dans ce modèle. On ne crée pas de méthodes supplémentaires, on utilise les méthodes standardisées que l'on développe pour le type de media dont on a besoin. Il y a en conséquence beaucoup moins de couplage entre le client et le serveur : un client peut utiliser un service de type **REST** sans aucune connaissance de l'**API**.

A l'inverse, un client **SOAP** doit tout savoir des éléments qu'il va utiliser pendant son interaction avec le serveur, sinon cela ne fonctionnera pas.

Pour notre projet, **SOAP** ne fera pas l'objet d'une étude détaillée étant donné que nous opterons vers une solution **RESTFull** qui correspond à notre besoin et qui est facile d'apprentissage.

➤ Le format d'échange

**REST** n'impose pas de format d'échange entre client et serveur. Les plus connus sont **XML** et **JSON** bien qu'il en existe d'autres. Certains membres de l'équipe ont déjà représenté les données en **JSON** et pour cela, nous opterons pour celui-ci. Le **JSON** reste le plus adapté à l'heure actuelle pour représenter des données comme une **BDD**.

Lien 1 : [https://msdn.microsoft.com/fr-fr/library/bb412179\(v=vs.110\).aspx](https://msdn.microsoft.com/fr-fr/library/bb412179(v=vs.110).aspx)

Ce lien permet de voir en détails comment sérialiser et désérialiser une instance d'un type défini à **JSON**.

Lien 2 : [https://msdn.microsoft.com/fr-fr/library/system.runtime.serialization.json.datacontractjsonserializer\(v=vs.110\).aspx](https://msdn.microsoft.com/fr-fr/library/system.runtime.serialization.json.datacontractjsonserializer(v=vs.110).aspx)

Détails sur la librairie **DataContractJsonSerializer** permettant de sérialiser des objets au format **JSON** et désérialiser les données **JSON** vers des objets.

Lien 3 : <http://www.json.org/json-fr.html>

Ce lien détaille le langage d'échange de données.

➤ Quels seront les différentes opérations de synchronisation ?

Un diagramme de séquence va être réalisé pour afficher les différentes opérations.