Atelier Big Data INRAE : MongoDB dans Phenome

Vincent NEGRE (CEFE CNRS)

Renaud COLIN (MISTEA INRAE)

Atelier Big Data INRAE 10-12 Janv 2023 (Sète)

Sommaire

- Phénome et OpenSILEX
- NoSQL orienté document
- Usages avancés
- Performances ?
- Conclusion

Phenome : Phénotypage haut débit

- **Phénotypage haut débit** : techniques de mesure des caractères observables d'une plante
- Analyser l'**adaptation des plantes au changement climatique** (sécheresse, CO_2, hautes températures, maladies émergentes)
- Développement d'Infrastructures et d'outils de phénotypage
  - **9 sites Français**
  - **capteurs, robots, plantes**
  - **systèmes d'informations** et outils **d'imageries**

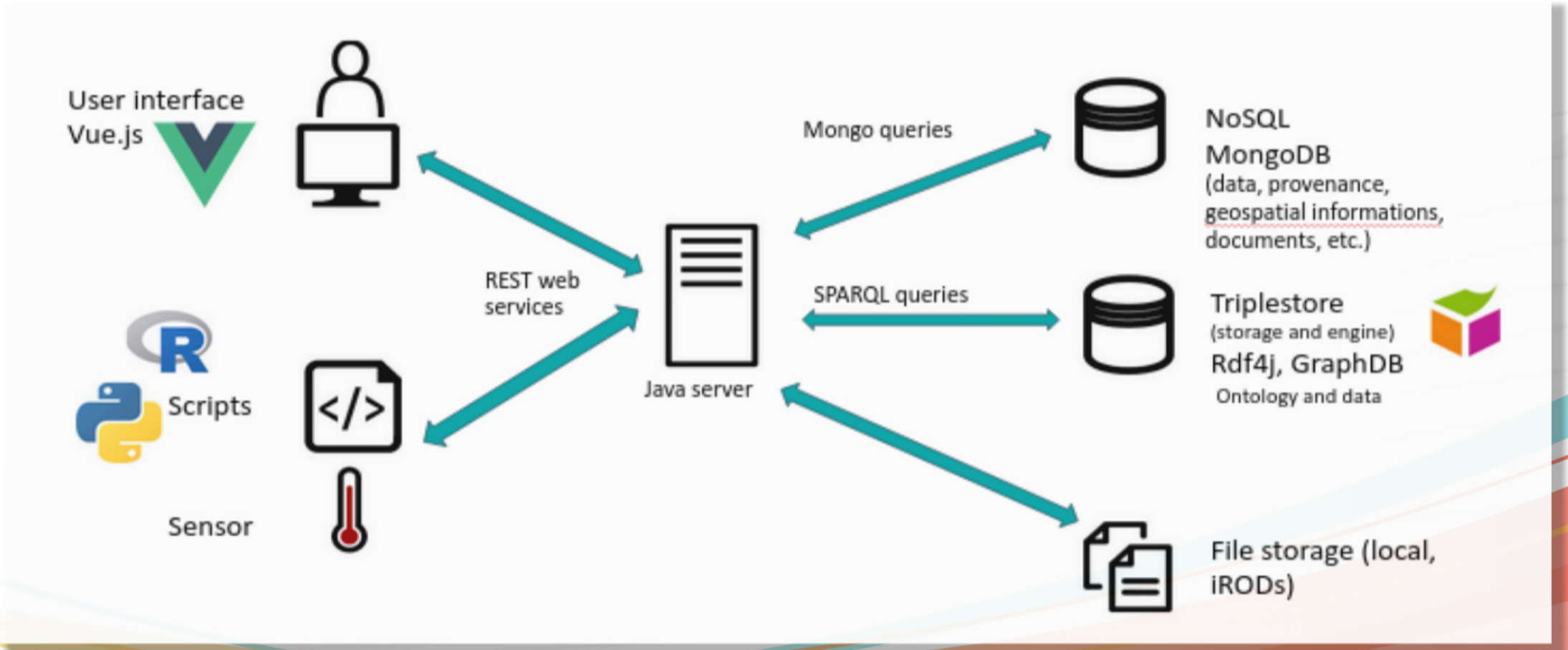
OpenSILEX : présentation

- Système d'information pour la **gestion des données scientifiques**
- Representation des **connaissances à l'aide d'ontologies**
- Gestion et analyse de **données hétérogènes** :
  - **Graphe** : description des objets et concepts pour l'expérimentation
  - **Document** JSON pour le stockage des mesures/acquisition de données
  - Informations **géospatiales** (position gps des objets, forme)
  - **Fichiers** : Document scientifiques, Images, fichier de données brutes

OpenSILEX : cas d'utilisations

- **Gérer, visualiser et analyser** les données au sein d'une expérience scientifique
- **Description, partage** et **réutilisation** de ces données
- Ex : Analyser la resistance au stress hydrique d'une plante
  - **Variables** : température de la plante, taux d'humidité, rendement par hectare, masse, taux d'infection , taux d'ensoleillement
  - **Objets scientifiques** : champ, parcelle, plante, pot, feuille
  - **Propriétés et facteurs des objets** : type de plante, variété, âge, niveau irrigation
  - **Équipements et installations**: capteur de temperature, d'humidité, d'ensoleillement, caméra, serre, cabine d'imagerie
  - **Evénements :** récolte, plantation, pluie, grèle, innondation, attaque de ravageur
  - **Mesure/Data** : température, humidité, photo, analyse d'images

OpenSILEX : architecture



OpenSILEX: description des données

- **Graphe** RDF pour représenter les liens entre concepts et entités
- **Variable, espèce, facteurs expérimentaux**
  - La T° de la plante en °C, le taux d'humidité en %
  - L'èspece "Cupressus"
  - Le taux d'irrigation de la parcelle, l'orientation de la plante au Soleil
- **Objet scientifiques**
  - Ex : l'arbre "cypres_1", d'espèce "cupressus"
- **Capteur, actionneur, installation**
  - Le capteur de T° "RP_TS1", le capteur d'humidité "RP_HS1"
  - La serre "serreA", le champ "champ1"

OpenSILEX : mesures et provenance

- **Provenance** : Contexte d'acquisition de la données (+/- expressif/complexe)

- Ex: Acquisition de donnée faite à l'aide de capteur de temperature "DS18B20" au sein de l'expérience "xp_big_data_temperature"

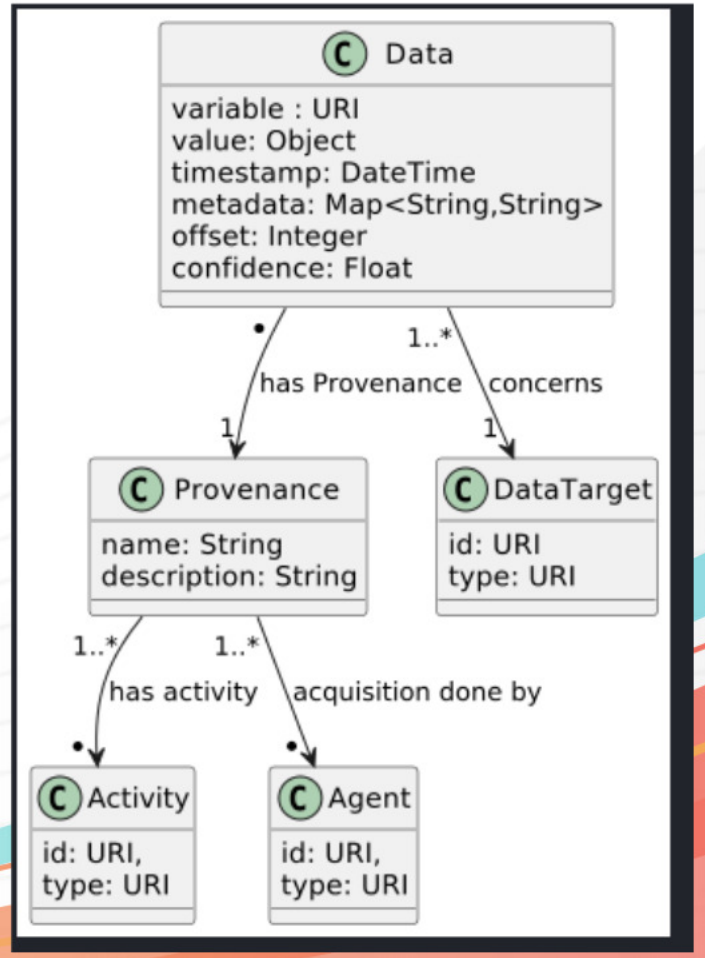
- **Experiment:** xp_big_data_temperature
- **acitivity** : Acquisition de mesure
- **agents** :
  type: capteur de temperature
  id: DS18B20

OpenSILEX : mesures et provenance

- **Data:** Acquisition de valeur d'une variable, concernant un objet, à un instant t, selon une provenance

- Ex : Le 12/10/2022 à 12:00, le capteur "DS128B20" a mesuré une temperature de 30°c pour la plante "p1"

- **value** : 30
- **timestamp:** 12 octobre 2022 à 12:00
- **variable** : Température de la plante en °Celsius
- **Target:** la plante "plant_1"
- **provenance** : Acquisition de donnée avec un capteur de température "DS128B20"



Hétérogénéité des données

- **Valeurs simple/litérale**
  - Booléen, entier, réel : true, 8611, 3.14,
  - Texte/Date : "WEST" "20221013:1306:52:+2"
  - URI: "opensilex:id/841654651"
- **Valeurs multiple**
  - Tableau: ["acgt","gtac"], ["opensilex:id/156654654, "opensilex:id/1648698"]
- **Variable multimensionnelle**
  - Map/Dictionnaire: {R: 214, G: 58, B: 21}
  - Objet imbriqué: { taille: 54, couleur{R: 214, G: 58, B: 21}}
- **Fichier**
  - Images
  - Fichiers de données "brute", ex: csv, odt, parqet
  - Format spécifiques: (ex: fasta, 2bit, etc)

NoSQL orienté document

- Représentation à l'aide du format JSON
- Souplesse du modèle de donnée : pas de schéma nécessaire (possibilité de validation ad-hoc via schéma)
- Permet une maj rapide du schéma de donnée coté applicatif
- Permet de représenter l'ensemble des données d'un document sans avoir à faire de jointure (aggregation)

Donnée avec variable simple (JSON)

JSON

```json
{
    "_id": "atelier_bd:data_1",
    "value": 36.5,
    "variable": "atelier_bd:temperature",
    "target": {
        "rdf_type": "vocabulary:Plant",
        "uri" : "atelier_bd:plant_1"
    },
    "timestamp": "0221013:1306:52:+2",
    "provenance": {
        "uri" : "atelier_bd:provenance_1",
        "agents" : [
          {"rdf_type": "vocabulary:TemperatureSensor",
          "uri" : "atelier_bd:tempature_sensor_1"
          }
        ]
    }
}
```

Donnée multi-valuée(JSON)

JSON

```
{
    "_id": "atelier_bd:data_1",
    "value": [45,32,63,58,61,21],
    "variable": "atelier_bd:LeafSize",
    "target": {
        "rdf_type": "vocabulary:Plant",
        "uri" : "atelier_bd:plant_1"
    },
    "timestamp": "0221013:1306:52:+2",
    "provenance": {
        "uri" : "atelier_bd:provenance_2",
        "agents" : [
          {"rdf_type": "vocabulary:Sensor",
          "uri" : "atelier_bd:leaf_size_sensor"
          }
        ]
    }
}
```

Donnée multi-dimensionelle(JSON)

JSON

```
{
  "_id": "atelier_bd:data_1",
  "value": {
    "r": 214,
    "g": 58,
    "b": 21
  },
  "variable": "atelier_bd:color",
  "target": {
    "rdf_type": "vocabulary:Plant",
    "uri": "atelier_bd:plant_1"
  },
  "timestamp": "0221013:1306:52:+2",
  "provenance": {
    "uri": "atelier_bd:provenance_3",
    "agents": [
      {
        "rdf_type": "vocabulary:ColorSensor",
        "uri": "atelier_bd:color_sensor"
      }
    ]
  }
}
```

MongoDB

- Principale base de données Document
- Fonctionnalités classique d'un SGBD : CRUD, requêtage, administration, recherche avancée, déploiement dans le cloud, monitoring
- Grosse communité et beaucoup de drivers (Java, C++,Python, JS, etc)
- Scalabilité horizontale à l'aide de la réplication et du sharding

MongoDB et BigData

- **Volume** :
  - Possibilité de gérer plusieurs dizaines/centaines de millions de documents sur un serveur modéremment puissant
  - Scalabilité horizontable avec sharding et réplication
- **Variété**
  - Souplesse du schéma, permet d'être fléxique sur le type de donnée traité
  - Offre la possibilité de stocker des fichiers (document, image) grâce au système de fichier GridFS)
- **Velocité** :
  - Faible latence pour les opérations usuelles de recherche
  - Bonne performances d'aggrégation
  - Depends du modèle de donnée, de l'indexation et de la configuration

Utilisation dans OpenSILEX

- Souplesse pour la représentation des acquisitions et provenance
- Sérialisation d'objets effectuée par l'API Java MongoDB
- Bonne performances de recherche et d'écriture
- Utilisation du système GridFS pour le stockage de fichier
- Utilisation de plusieurs indexes pour accélerer les recherches
- Peu de configuration pour quelques millions/dizaines de millions de documents

CRUD : création

JavaScript

```
db.data.insertOne({
  "variable": "opensilex:id/variable/temperature",
  "provenance": "opensilex:id/provenance/prov_raspeberry_pi_temperature",
  "target": {
    "uri": "opensilex:id/plant_1",
    "rdfType": "vocabulary:Plant"
  },
  "date": "2023-01-07T00:00:00+01:00",
  "value": 12
})
```

CRUD : lecture (par identifiant)

- Recherche par le champ **uri** (unique)

```JavaScript
db.getCollection('data').find( {"uri": "http://www.phenome-fppn.fr/id/data/2020-11-17-173904484167/2017-06-23-125751165" })
```

- Recherche par le champ **_id** (MongoDB)

```JavaScript
db.data.findOne({ _id : ObjectId ("127x74937107dkcde3beb986") })
```

CRUD : recherches basiques

- Recherche par **variable**

```JavaScript
db.data.find(
    {
        "variable: { $eq: "opensilex:id/provenance/prov_test1" }
    }
```

- Recherche des données concernant une plante

```JavaScript
db.data.find(
    {
        "target.uri": { $eq: "opensilex:id/plant_1" }
    }
)
```

CRUD : recherches basiques

- Recherche des données sur les plantes

```JavaScript
db.data.find(
    {
        "target.rdfType": { $eq: "vocabulary:Plant" }
    }
)
```

- Recherche de toutes les mesures de temperature > 30°

```JavaScript
db.data.find(
    {
        "variable": {$eq:"opensilex:id/variable#variable.air_temperature" },
        "value" : {$gte: 30}
    }
)
```

CRUD : ordre, pagination et projection

- Ordre et pagination

```javascript
db.data.find({ }).limit(1000).sort({"date": -1})
```

- Projection (Limiter les champs recupérés)

```javascript
db.data.find(
    {
        "variable": {$eq:"opensilex:id/variable#variable.air_temperature" },
        "value" : {$gte: "30"}
    },
    { value: 1, date: 1}  // projection part : fetch value and date fields only
)
```

Bonnes pratiques (performances)

- Operation en **Batch** lors d'insertion et modifications massives
- **Modélisation et recherche**
  - Jointure/Aggregation avec d'autres collections
  - Maj d'un grand nombre de documents suite à la maj d'une "clé étrangère" ?
- **Indexation**
  - Ajouter des indexes sur les champs filtrant et utilisé régulierement
  - Les indexes doivent correspondrent aux requêtes/besoins metiers
  - Trop d'index tue l'index
- **Réplication et cohérence**
  - Réplica -> duplication des écritures sur chaque réplica si cohérence stricte
  - Cohérence eventuelle selon les besoins applicatifs
- **Monitoring**
  - MongoDB Compass : visualisation des collections/requêtes "hot" et monitoring
  - Analyse des plans d'éxecutions des requêtes
  - Analyse de l'usage RAM et des performances I/O (disque et réseau)

Liens

- https://www.mongodb.com/docs/manual/core/schema-validation/specify-json-schema/#std-label-schema-validation-json
- https://www.mongodb.com/docs/spark-connector/current/
- https://www.mongodb.com/docs/kafka-connector/current/
- https://www.mongodb.com/docs/manual/core/replica-set-write-concern/
- https://www.mongodb.com/docs/manual/crud/
- https://www.mongodb.com/docs/manual/core/data-modeling-introduction/

Bonnes pratique (modélisation)

- Utilisation des JSON Schémas pour améliorer la qualité de ses données
- L'efficacité de l'interrogation depends toujours du modèle de donnée et des besoins. Un modèle trop complexe ou inadapté induiras toujours des difficultés de développements
- La souplesse du schéma n'implique pas non plus de ne pas documenter propement son modèle de donnée

MongoDB et BigData

- Efficace dans un cadre transactionnel
- Bonne possibilité pour de l'analyse (batch, aggregation et MapReduce) mais limité par rapport à d'autre outils (Hadoop, Spark, Column stores) sur de très gros besoin en volume ou latence
- Bonne intégration avec des outils d'analyse et d'integration de données : Kafka, Spark