Web Services RESTFul

2e journée d'échange CATI SICPA/CODEX

anne.tireau@supagro.inra.fr

Web services ?

- architecture logicielle orientée services (*SOA*)

- **communication** entre applications, sur les réseaux
- applications **hétérogènes** (langage, plate-formes, etc.)
- **interopérabilité** des applications
- utilisation de **standards**, normes et protocoles du Web
- **interfaces** de services interprétables par les machines
- format de données et d'**échange** XML, JSON

- technologies et architectures disponibles
  - **WS-*** repose sur les standards SOAP, WSDL, etc.
  - **REST** (*Representational state transfer*)

REST

- ensemble de critères de conception
- architecture mettant en avant les **ressources**
- règles :
  - **interface uniforme** : utilisation des méthodes de HTTP

    POST, GET, PUT, DELETE, OPTIONS, HEAD, etc.

  - utilisation des **URI** (Uniform Ressources Identifier) pour l'**adressage** de ressources

    http://inra.fr/phenomeapi/studies

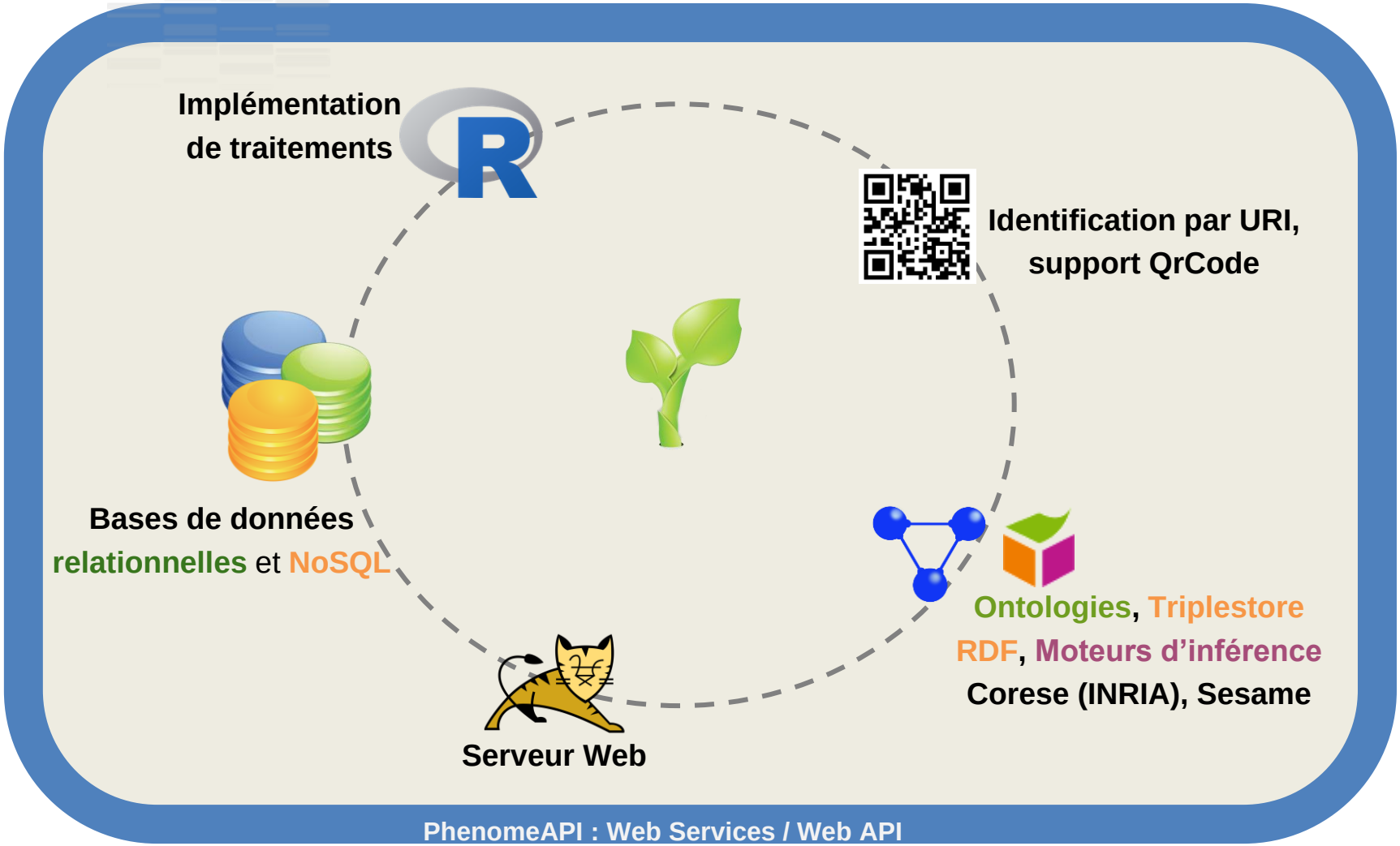
    http://inra.fr/phenomeapi/study/ARCH2015-06-12

  - programmation **sans état**

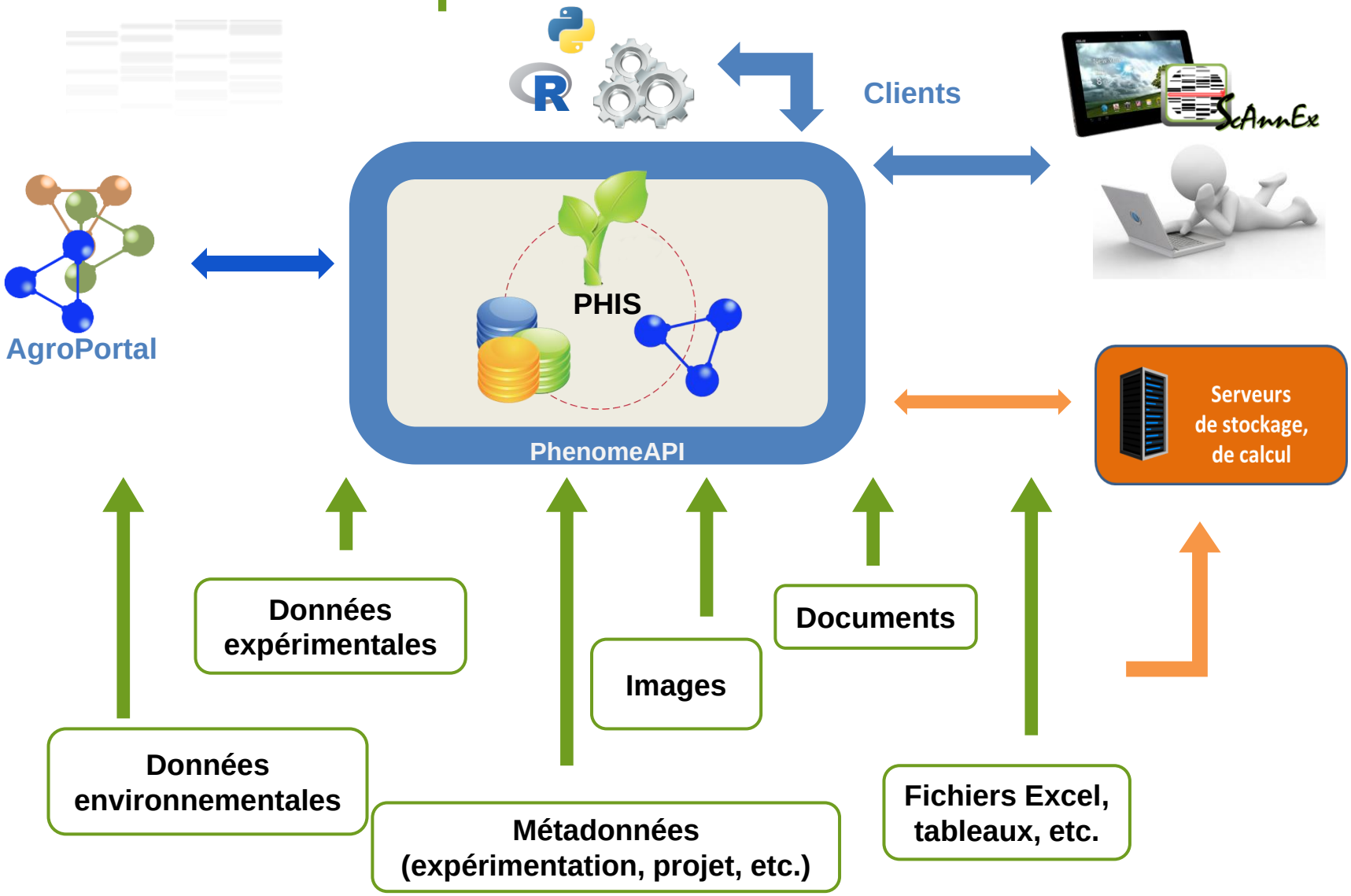
  - liens entre les ressources doivent êtres fournis : **connectivité**

    si des projets contiennent des études, les liens vers les ressources liées sont fournis

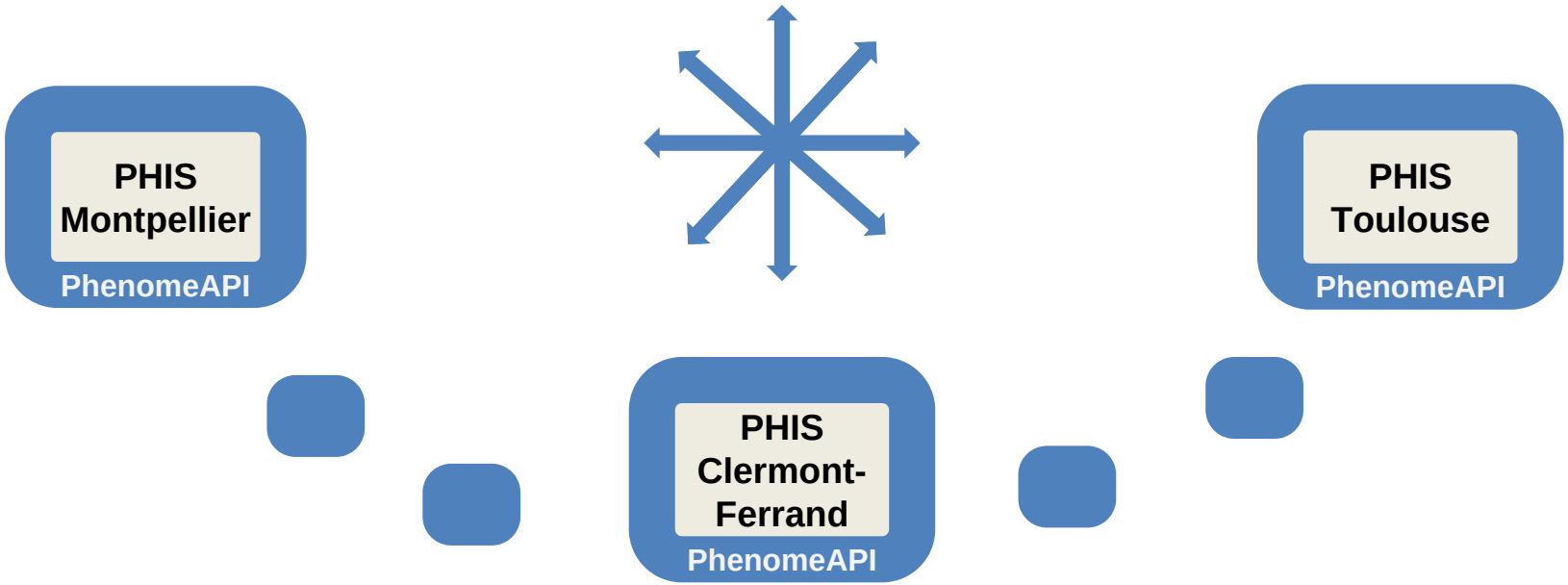
PHIS : Abstraction



PHIS : Interopérabilité

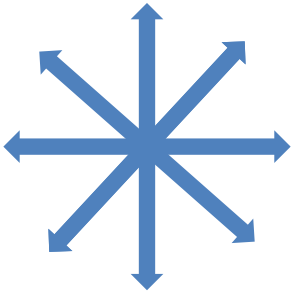


PHIS : Communauté



PHIS : Communauté

Ephesis / GnpIS (URGI)



PHIS Montpellier
PhenomeAPI

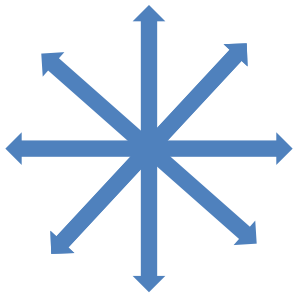
PHIS Clermont-Ferrand
PhenomeAPI

PHIS Toulouse
PhenomeAPI

PHIS : Communauté

Plant Breeding API
brapi • brapi

Ephesis / GnpIS (URGI)



PHIS Montpellier
PhenomeAPI

PHIS Toulouse
PhenomeAPI

PHIS Clermont-Ferrand
PhenomeAPI

PhenomeAPI

- nombreuses technologies existent :
  JAXB / JAX-RS / Jersey (Apache CXF ou Jboss RESTEasy)

- correspondance entre des ressources et des instances de classes Java

- utilisation d'annotation pour créer simplement des services

PhenomeAPI : exemple

```java
@Path("/brapi/v1/studies")
public class Studies {
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Response getStudies(@QueryParam("pageSize") @DefaultValue("100") int limit,
            @QueryParam("page") @DefaultValue("1") int page,
            @QueryParam("sessionId") String id,
            @QueryParam("projectId") String projectId) throws URISyntaxException {
        /* code */
        LstStudies myLst = findStudies();
        /* return / serialisation */
        return Response.status(Response.Status.OK).entity(myLst.convertToJson(finalObj)).build();
    }
```

PhenomeAPI : exemple - studies

Appel dans un navigateur :
http://147.99.7.151:8080/phenomeapi/brapi/v1/studies?sessionId=7d6a1a7946131d9d168a68e26651fa37


```
{
  metadata: {
    pagination: {
        pageSize: 100,
        currentPage: 1,
        totalCount: 100,
        totalPages: 1
    },
    status: [ ]
  },
  result: {
    data: [
      {
          studyDbId: "ARCH2015-03-05",
          name: "ARCH2015-03-05",
          studyObjective: "",
          type: "EXPERIMENT",
          location: http://www.phenome-fppn.fr/m3p/phenoarch,
          startDate: "2015-03-05",
          endDate: "2015-04-10",
          programName: "",
          designType: "",
        seasons: [
            "2015"
          ],
        projectName: [
            "Test"
          ],
        keyContact: [
            "welcker@supagro.inra.fr",
            "cabrera@supagro.inra.fr"
          ]
      },
```

PhenomeAPI : exemple d'authentification

- authentification
  - plusieurs possibilités suivant les frameworks utilisés
  - Jersey: gestion des autorisations par annotations au niveau des méthodes et groupes d'utilisateurs / rôles (BD)

- PhenomeAPI
  - méthode du jeton (grain de sable)

http://147.99.7.151:8080/phenomeapi/brapi/v1/token?username=anne.tireau@supagro.inra.fr&password=monpsw


```
{
  ▾ metadata: {
        pagination: { },
        status: [ ]
    },
    session_token: "11dce8a5477a0c04bfef0337922bbd62"
}
```

PhenomeAPI : exemple appel depuis R (1/2)

```r
url="http://147.99.7.151:8080/phenomeapi/brapi/v1/token?username=anne.tireau@supagro.inra.fr&password=monpsw"
con <- curl::curl(url) # connection Get
getResults = readLines(con)
cleanedGet <- gsub("\"","",getResults) #suppression guillemets
token <- gsub("session_token = ","",cleanedGet) #recuperation du token
close(con)
```

```
> getResults
[1] "{"
[2] "  \"metadata\": {"
[3] "    \"pagination\": {},"
[4] "    \"status\": []"
[5] "  },"
[6] "  \"session_token\": \"f2612bb6db9e985e75ef6cb00ceae5cd\""
[7] "}"
```

PhenomeAPI : exemple appel depuis R (2/2)

```
library(jsonlite)
key <- "&apikey=39c83d5a4acc42be993ee637e2e4ba3d"
drone_bills <- fromJSON(paste("http://openstates.org/api/v1/bills/?q=drone", key))
drone_bills$title <- substring(drone_bills$title, 1, 40)
print(drone_bills[1:5, c("title", "created_at","state", "chamber", "type", "id")])
```

```
> print(drone_bills[1:5, c("title", "created_at","state", "chamber", "type", "id")])
                                    title          created_at state chamber type          id
1 Relates to prohibiting civilian drone us 2015-11-21 15:32:34    ny   upper bill NYB00080375
2                                  Drones 2015-10-27 04:51:04    fl   upper bill FLB00010977
3 Relating to: interfering with hunting, f 2015-10-23 05:37:04    wi   lower bill WIB00006477
4 Relating to: interfering with hunting, f 2015-10-17 05:35:04    wi   upper bill WIB00006454
5                          Sexual Predators 2015-10-13 04:42:45    fl   upper bill FLB00010856
```

Merci
Des questions ?