Gestion des données ouvertes en Agriculture et Nutrition

Ce cours en ligne est le fruit d'une collaboration entre les partenaires de GODAN Action, y compris Wageningen Environmental Research (WUR), AgroKnow, AidData, l'Organisation des Nations Unies pour l'Alimentation et l'Agriculture (FAO), le Forum Mondial sur la Recherche Agricole (GFAR), l'Institut des Etudes du Développement (IDS), le Land Portal, l'Open Data Institute (ODI) et le Centre Technique de Coopération Agricole et Rurale (CTA).

GODAN Action est un projet de trois ans du Département pour le Développement International du Royaume-Uni pour permettre aux utilisateurs, producteurs et intermédiaires de données de s'engager efficacement avec les données ouvertes et maximiser leur potentiel d'impact dans les secteurs de l'agriculture et de l'alimentation. Nous travaillons en particulier à renforcer les capacités, à promouvoir des normes communes et les meilleures pratiques et à améliorer la manière dont nous mesurons l'impact. [www.godan.info]

MODULE 4 : PARTAGE DES DONNEES OUVERTES

LEÇON 4.3: Interopérabilité structurelle et architecturale

Photo par Mario Antonio Pena Zapatería sous licence CC BY SA 2.0

Objectifs et résultats d'apprentissage

Cette leçon a pour objectif ;

- D'expliquer les bases de l'interopérabilité structurelle : formats de données et structures de données
- D'expliquer les bases de l'interopérabilité architecturale : protocoles, cadres techniques.
- De prodiguer des conseils sur les avantages et les inconvénients des solutions spécifiques.

À la fin de cette leçon, vous devrez être en mesure ;

- De comprendre les bases de l'interopérabilité structurelle et les meilleures pratiques actuelles
- D'évaluer les formats et les protocoles qui répondent le mieux à leurs besoins
- (De guider les développeurs pour) qu'ils adoptent les solutions les plus interopérables.

Sommaire

Liste des illustrations

1. Interopérabilité structurelle

Nous avons dit que c'est à ce niveau que les (méta)données deviennent lisibles par machine. Cependant, la "lisibilité par machine" dans son sens le plus littéral ne suffit pas à l'interopérabilité structurelle : pour que les machines comprennent la structure des (méta)données, c'est-à-dire quels sont les étiquettes/éléments de métadonnées/noms/variables de colonne et quelles sont les valeurs et s'il existe une hiérarchie, les (méta)données doivent être suffisamment structurées pour qu'une machine en extraie les valeurs individuelles.

En d'autres termes, les (méta)données ne doivent pas seulement être lisibles, mais " analysables " par les machines. Puisque " analyser " signifie " diviser un fichier ou une autre entrée en données facilement stockables ou manipulables ", plus un format est régulier et rigoureux, plus il est facile de l'analyser. Idéalement, les formats analysables ont une spécification publiée afin que les développeurs sachent comment la structure est construite et puissent écrire des analyseurs en conséquence. Ce niveau d'interopérabilité est donc atteint grâce au choix du format de données le plus facile à analyser. Les chapitres suivants illustreront les formats les plus courants.

À ce niveau d'interopérabilité, les machines peuvent diviser le fichier et identifier les valeurs et leurs rôles, sans comprendre leur signification. Cependant, bien que les structures de données que nous allons décrire n'aient en elles-mêmes aucune signification et ne permettent pas un traitement significatif, elles peuvent être rendues sémantiques et donc compréhensibles en appliquant les technologies décrites dans la leçon 4.4 sur l'interopérabilité sémantique.

1.1. Sérialisation des (méta)données : structures de données en formats de fichiers

Une structure de données (hiérarchique, relationnelle, tabulaire...) peut être représentée dans un fichier de plusieurs façons. La façon dont les données sont physiquement stockées dans un fichier est appelée la " sérialisation " des données : En informatique, dans le contexte du stockage des données, la sérialisation est le processus consistant à traduire les structures de données ou l'état de l'objet dans un format qui peut être stocké (par exemple, dans un fichier ou un tampon mémoire)'[1]

Il existe de nombreux formats de fichiers qui sont analysables par les machines à proprement parler, n'importe quel format de fichier peut être analysé par certains logiciels, sinon il serait inutile. Ce qui rend un format de fichier plus facilement analysable et donc plus interopérable est :

[1] From Wikipedia: https://en.wikipedia.org/wiki/Serialization

- La simplicité de la structure : moins il y a d'éléments dans la structure et moins les constructions possibles sont nombreuses, plus il est facile pour une machine d'analyser le fichier sans des raisonnements trop complexes. (D'autre part, le format doit également tenir compte des structures de données complexes : le format de fichier le plus approprié est celui qui allie simplicité et flexibilité suffisante pour représenter des structures de données complexes).
- La rigueur et la " régularité " de la structure : moins il y a d'options pour sérialiser la même chose de manière différente, plus il est facile pour une machine d'analyser le fichier et d'identifier le rôle de chaque élément.
- L'existence d'une spécification claire et ouverte pour le format, ce qui facilite l'écriture d'analyseurs pour tout développeur. (De nombreux formats sont liés à un produit logiciel et ne peuvent être correctement et entièrement analysés que par ce produit. Cela rend leur niveau d'interopérabilité très bas.)
- Une aide supplémentaire est l'existence de bibliothèques de logiciels et d'API qui peuvent analyser le format.

Les formats considérés comme les plus interopérables au regard des critères ci-dessus sont CSV, XML et JSON.

Les formats basés sur des tableaux binaires tels que NetCDF et HDF5 conservent une place particulière pour leur utilisation dans la recherche. Nous ne les examinerons pas ici pour quelques raisons : ils sont plus liés à des bibliothèques de logiciels spécifiques (bien que de nombreux outils puissent les lire) ; ils sont encore plus axés sur la compacité et l'efficacité de la transmission des données que sur une interopérabilité plus large, notamment sémantique ; et certains travaux ont déjà été réalisés pour représenter NetCDF dans CSV (format CEDA BADC-CSV[2]) et dans XML (UNIDATA NcML[3]).

Les fichiers CSV (valeurs séparées par des virgules) ont un format tabulaire simple avec un enregistrement d'en-tête avec des noms de colonnes, des virgules pour marquer les limites des champs et des interlignes pour marquer les limites des enregistrements.

CSV est le format de données le plus simple et très utilisé pour les données tabulaires typiques comme les statistiques ou les observations, mais il ne peut pas représenter des structures complexes et n'est pas documenté de manière lisible par machine : " Il n'y a pas de mécanisme dans CSV pour indiquer le type de données dans une colonne particulière, ou si les valeurs dans une colonne particulière doivent être uniques. Il est donc difficile de valider et il est sujet à des erreurs telles que les valeurs manquantes ou les types de données différents

[2] http://help.ceda.ac.uk/article/105-badc-csv

[3] http://www.unidata.ucar.edu/software/thredds/current/netcdf-java/ncml/#NcML22

au sein d'une colonne'[4] Certains travaux visant à améliorer l'interopérabilité des fichiers CSV ont été réalisés par le groupe de travail du W3C sur le " CSV sur le Web ": Cas d'utilisation et exigences"[5].

XML (eXtensible Markup Language[6]) est " un langage de balisage qui définit un ensemble de règles pour coder des documents dans un format à la fois lisible par l'homme et par machine. Les objectifs de conception de XML mettent l'accent sur la simplicité, la généralité et la facilité d'utilisation sur Internet. Bien que la conception de XML soit axée sur les documents, le langage est largement utilisé pour la représentation de structures de données arbitraires..."[7] (voir illustration 1 ci-après).

L'aspect le plus intéressant du XML en termes d'interopérabilité est le support de la définition des'schémas' auxquels un document XML spécifique peut se conformer. Les schémas sont des documents lisibles par machine qui spécifient comment nommer et organiser les éléments dans un document XML et peuvent définir des hiérarchies, des contraintes, etc. Nous verrons dans la leçon 4.4 comment les schémas permettent d'intégrer des vocabulaires sémantiques dans les documents XML.

```
<dataset>
  <metadata>
    <dataset-metadata1>ValueA</dataset-metadata1>
    <dataset-metadata2>ValueB</dataset-metadata2>
    ...
  </metadata>
  <records>
    <record>
      <variable1>Value1</variable1>
      <variable2>Value2</variable2>
      <variable3>Value3</variable3>
      ...
    </record>
    <record>
      <variable1>Value4</variable1>
      <variable2>Value5</variable2>
      <variable3>Value6</variable3>
      ...
    </record>
    ...
  </records>
</dataset>
```

Illustration 1 Exemple d'une structure XML générique pour un ensemble de données

Les métadonnées d'un jeu de données XML peuvent atteindre un niveau de complexité élevé, comme dans cet exemple du service géologique américain: https://sofia.usgs.gov/metadata/sflwww/SOFIA_Cape_Sable.xml, où, outre les métadonnées étendues sur l'ensemble de données (créateurs, conditions, citations), les éléments <attr> décrivent toutes les variables utilisées dans l'ensemble de données (dans ce cas le fichier de métadonnées l'ensemble de données est séparé de celui des données).

[4] W3C. CSV on the Web: A Primer. https://www.w3.org/TR/tabular-data-primer/

[5] https://www.w3.org/TR/csvw-ucr/

[6] https://www.w3.org/TR/REC-xml/

[7] From Wikipedia: https://en.wikipedia.org/wiki/XML

JSON (JavaScript Object Notation) est " un format de fichier ouvert standard qui utilise du texte lisible par l'homme pour transmettre des objets de données constitués de paires d'attributs et de types de données de tableau (ou toute autre valeur sérialisable) "[8].

JSON a une bonne combinaison de simplicité et de flexibilité en termes de structures de données (il est conçu comme un format pour représenter des objets de données de programmation de toutes sortes) et il a maintenant un support pour la définition de schémas à des fins de validation[9]. Étant donné la facilité d'utilisation des structures JSON dans les langages de programmation (le besoin d'analyse est minimal et la structure reflète les pratiques de programmation orientée vers l'objet), JSON est le format de sérialisation préféré par les programmeurs.

```
{"dataset": {
   "dataset-metadata1": "ValueA",
   "dataset-metadata2": "ValueB",
   "records": {
     "record": [
     {"variable1": "Value1", "Variable2": "Value2", "Variable3": "Value3"},
     "record": [
     {"variable1": "Value4", "Variable2": "Value5", "Variable3": "Value6"},
     ...
     ]
   }
}}
```

Illustration 2 Exemple de structure JSON

CSV, XML et JSON sont les formats les plus interopérables compte tenu également de ce que nous verrons sur l'application des technologies sémantiques : ce sont les formats auxquels ces technologies sont les plus facilement applicables (voir leçon 4.4).

Parmi les trois, XML est traditionnellement considéré comme la meilleure combinaison de (a) simplicité ; (b) flexibilité pour les structures de données complexes ; et (c) support des définitions de schémas qui établissent des contraintes et les rendent plus faciles à comprendre. Cependant, étant donné que JSON supporte également les schémas et que la spécification JSON-LD (JSON for Linking Data[10]) fournit une méthode de codage des données liées, JSON peut être considéré comme aussi approprié que XML.

D'autres formats qui sont extrêmement interopérables sont les principaux formats RDF natifs, Turtle et N-Triples. Puisqu'il s'agit de la sérialisation native de la grammaire RDF, et que la grammaire RDF mérite un traitement séparé, nous allons les décrire dans la section suivante.

---

[8] From Wikipedia: https://en.wikipedia.org/wiki/JSON

[9] http://JSON-schema.org/

[10] 'JSON for Linking Data'. https://json-ld.org/

1.2. Au-delà des formats de fichiers : RDF, une'grammaire' rigoureuse derrière le format

Le cadre de description des ressources (RDF) est " une méthode générale de description conceptuelle ou de modélisation de l'information qui est mise en œuvre dans les ressources Web, en utilisant une variété de notations syntaxiques et de formats de sérialisation des données "[11]. En tant que tel, il n'est pas un format et n'est pas lié à un format spécifique : tout format qui peut représenter la 'grammaire' RDF de base peut implémenter RDF.

La grammaire RDF est basée sur des énoncés faits de sujet - prédicat - objet ; chaque énoncé est un " triple " et l'hypothèse est que des combinaisons de tels triples peuvent tout représenter et décrire. Le " ciment " de ces triples est constitué par les identificateurs uniques de ressources (URI) qui renvoient toujours de manière univoque à la même entité, ce qui permet de diviser des structures de description complexes en triples plus simples où la même ressource est soit le sujet soit l'objet de l'énoncé et des prédicats significatifs lient le sujet à l'objet.

La syntaxe des triples est très simple :

Ressource A URI - a un titre -'Guerre et Paix'.
Ressource A URI - a un auteur - Personne A URI
Personne A URI - a un nom -'Lev Tolstoj'

Les triples énoncés sont normalement représentés sous forme de graphiques avec des nœuds et des arcs, où le sujet et l'objet sont des nœuds, tandis que les prédicats sont des arcs (aussi appelés bords : les flèches, les relations). Chaque combinaison de nœud-arc-nœud est un triple distinct. Toutes les composantes du triple sont idéalement identifiées par les URI, même les prédicats et les arcs (voir comment ces URI permettent d'intégrer des vocabulaires sémantiques dans les énoncés RDF de la leçon 4.4).

Dans l'illustration suivante, le prédicat/arc est l'URI de la propriété telle que définie dans un vocabulaire de métadonnées existant qui a à son tour été formalisé en RDF, ce qui signifie que toutes les classes et propriétés ont des URI).

[11] From Wikipedia: https://en.wikipedia.org/wiki/Resource_Description_Framework

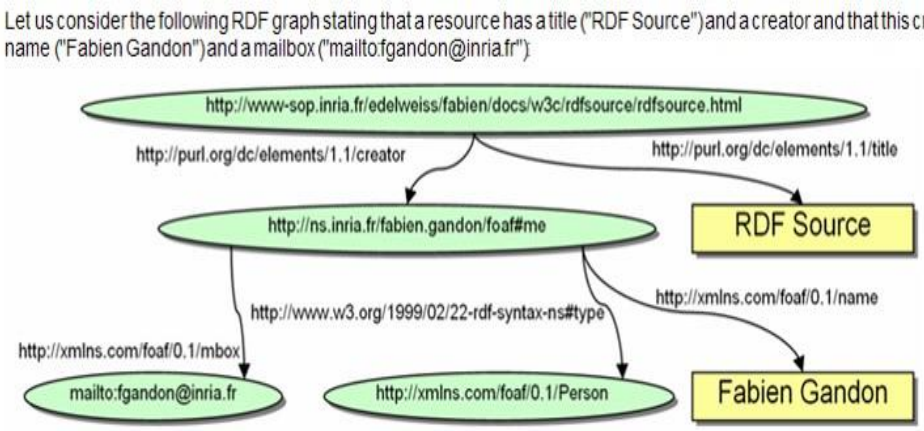

Illustration 3 RDF représenté sous forme de graphique, à partir du W3C[12]

Comme nous l'avons dit, n'importe quel format qui permet la triple construction peut implémenter RDF. Cependant, de nouveaux formats natifs, créés uniquement pour sérialiser les triples avec une syntaxe rigoureuse qui ne permet que la construction des triples, sontTurtle[13] et N-Triples[14].

```
<http://www-sop.inria.fr/edelweiss/fabien/docs/w3c/rdfsource/rdfsource.html> dc:title    "RDF Source"
<http://www-sop.inria.fr/edelweiss/fabien/docs/w3c/rdfsource/rdfsource.html> dc:creator <http://ns.inria.fr/fabien.gandon/foaf#me>
<http://ns.inria.fr/fabien.gandon/foaf#me>     rdf:type   foaf:Person
<http://ns.inria.fr/fabien.gandon/foaf#me>     foaf:name  "Fabien Gandon"
<http://ns.inria.fr/fabien.gandon/foaf#me>     foaf:mbox  <mailto:fgandon@inria.fr>f:mbox rdf:resource="mailto:fgandon@inria.fr"/>
```

Illustration 4 Exemple de sérialisation RDF en N-Triples du graphique ci-dessus, à partir de la même page W3C

La grammaire RDF a été appliquée avec succès au XML : XML/RDF n'est pas vraiment un nouveau 'format' (c'est encore formellement XML), mais plutôt un XML qui utilise des contraintes spécifiques qui renforcent la triple logique. Ces 'contraintes' sont définies dans la spécification 'RDF 1.1 XML Syntax' W3C.[15].

```
<rdf:RDF xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
 <rdf:Description rdf:about="http://www-sop.inria.fr/edelweiss/fabien/docs/w3c/rdfsource/rdfsource.html">
  <dc:title>RDF Source</dc:title>
  <dc:creator>
    <foaf:Person rdf:about="http://ns.inria.fr/fabien.gandon/foaf#me">
     <foaf:name>Fabien Gandon</foaf:name>
     <foaf:mbox rdf:resource="mailto:fgandon@inria.fr"/>
   </foaf:Person>
  </dc:creator>
 </rdf:Description>
</rdf:RDF>
```

Illustration 5 Exemple de sérialisation RDF en XML (XMLRDF) du graphique ci-dessus, à partir de la même page

Une comparaison rapide entre les fichiers XML suivant un schéma non-RDF et les fichiers RDF/XML montre que l'utilisation de la grammaire RDF rend les fichiers XML plus simples et plus rigoureux.

---

[12] https://www.w3.org/Submission/rdfsource/

[13] https://en.wikipedia.org/wiki/Turtle_(syntax)

[14] https://www.w3.org/TR/n-triples

[15] https://www.w3.org/TR/rdf-syntax-grammar/

L'explication technique exigerait trop de temps, mais il suffit de dire que certaines des sources possibles d'ambiguïté dans l'analyse du XML (les éléments imbriqués peuvent représenter des sous-propriétés ou des entités liées, les attributs peuvent représenter des propriétés ou des méta-métadonnées) sont surmontées par une " syntaxe à bandes " (les éléments doivent représenter alternativement des noeuds et des bords) et il y a beaucoup moins d'ambiguïté dans l'utilisation des attributs et des éléments imbriqués pour la même fonction.

La simplicité de la grammaire, la rigueur des contraintes et la flexibilité simultanée du mécanisme des trois énoncés qui peuvent être combinés pour créer des structures de données complexes font de RDF probablement la meilleure combinaison d'interopérabilité et de flexibilité par rapport aux critères énumérés dans la section 1.

2. Architectural interoperability

L'interopérabilité architecturale est plus proche de ce que HIMSS définit comme l'interopérabilité " fondamentale ", mais elle n'est pas liée aux protocoles de transmission généraux de base (TCP/IP, HTTP, FTP) autant qu'aux protocoles d'échange de données de niveau supérieur conçus pour le partage de (méta)données.

Outre les formats de données et la sémantique partagée, le partage de données à un niveau plus avancé peut également impliquer le respect de certaines exigences architecturales, notamment s'il est prévu de participer à des initiatives et des partenariats qui adoptent une architecture spécifique. Dans de tels cas, en plus d'être mises à disposition sous la forme d'un simple fichier qui peut être analysé directement, les (méta)données doivent souvent être servies par des services.

L'avantage d'exposer des (méta)données par le biais de services est : (a) les (méta)données peuvent être interrogées et des sous-ensembles sélectionnés peuvent être extraits ; (b) des métadonnées supplémentaires sur une ressource liée peuvent être extraites ; et (c) les (méta)données peuvent être paginées.

Le nom générique de cette tendance à l'exposition des données par le biais des services est " Data-asa-Service " (Daas), qui va de la simple exposition des données par un fournisseur de données aux grandes architectures de fournisseurs de données et de consommateurs.

Nous nous limiterons ici à ce que signifie 'exposer des données à travers un service' et nous utiliserons les exemples des deux protocoles les plus utilisés pour ce faire, mais il est important d'être conscient du fait qu'il existe de nombreuses façons de mettre Daas en œuvre.

De manière similaire à ce que nous avons dit à propos des formats de données et de la sémantique, plus un protocole est utilisé (en général ou dans la communauté où vous voulez partager vos données), plus vos données seront interopérables si vous le mettez en œuvre.

2.1. Protocoles les plus utilisés pour le partage de données

Les protocoles les plus populaires pour exposer les données en tant que service sont le protocole d'initiative d'archives ouvertes pour la collecte des métadonnées (OAI-PMH) et SPARQL, mais de plus en plus d'API RESTful normalisées [16] sont également utilisées. OAIPMH et SPARQL sont des API techniquement sans état et comme toutes les API, elles exposent des 'méthodes' qui acceptent un certain nombre de paramètres et ces méthodes peuvent être appelées via une requête HTTP et renvoient une réponse lisible par machine.

OAI-PMH est né dans l'environnement des bibliothèques et a été conçu principalement pour les métadonnées, mais il peut être utilisé pour exposer tout type de 'documents'. Tant que le format de réponse est XML et que le conteneur XML suit le schéma OAIPMH, l'élément <métadonnées> sous chaque <enregistrement> peut contenir XML en utilisant n'importe quel schéma, même XML/RDF, il peut donc convenir, par exemple pour exposer les enregistrements de métadonnées, des ensembles de données en utilisant un vocabulaire de métadonnées d'ensemble de données (comme le W3C Data Catalogue Vocabulary, DCAT).

Les méthodes de l'API OAI-PMH sont conçues pour permettre une série d'appels qui permettent à l'appelant (une application) de vérifier préalablement certaines métadonnées du référentiel (le nom, quels schémas de métadonnées sont supportés, quels sous-ensembles peuvent être récupérés...) et de récupérer les enregistrements filtrés selon leur format, leur période, leur sous-ensemble. Les enregistrements peuvent être récupérés en plus petits lots à l'aide d'un " jeton de reprise ".

[16] 'Le transfert représentationnel d'état (REST) ou les services web RESTful est un moyen d'assurer l'interopérabilité entre les systèmes informatiques sur l'Internet. Les services Web conformes à REST permettent aux systèmes demandeurs d'accéder aux représentations textuelles des ressources Web et de les manipuler en utilisant un ensemble uniforme et prédéfini d'opérations sans état. Tiré de Wikipédia: https://en.wikipedia.org/wiki/Representational_state_transfer

Illustration 6 Le workflow OAI-PMH [17]

Bien que les (méta)données exposées par le biais d'OAI-PMH puissent être rendues hautement interopérables, y compris sur le plan sémantique et même à l'aide de RDF et d'URI, le protocole en soi a quelques limites: a) la possibilité d'interroger les données est limitée à quelques paramètres acceptés et b) les enregistrements de la section <ListRecords> sont en général censés être du même type et avoir la même structure (bien que cela ne soit techniquement pas appliqué), de sorte que des appels différents doivent être effectués pour différents types d'entités.

SPARQL (SPARQL Query Language for RDF) est un langage d'interrogation des données RDF, mais c'est aussi le nom du protocole qui permet d'envoyer la requête via une requête HTTP et d'obtenir la réponse dans plusieurs formats compatibles RDF (RDF/XML, Turtle, JSON...).

Par rapport à OAI-PMH, SPARQL permet des requêtes beaucoup plus complexes construites à l'aide de la puissante triple grammaire et peut exposer des triples avec n'importe quel type de sujet (un ensemble de données, une organisation, un lieu, un sujet ...) et le tout dans une interface identique. Cela permet de filtrer les données de manière très granulaire et de rechercher d'autres propriétés (par exemple les étiquettes) des ressources référencées par les URI. Les clients SPARQL peuvent également envoyer la même requête à plusieurs moteurs SPARQL et combiner les réponses : l'utilisation des URI permet au client de fusionner les doublons et de consolider les propriétés d'une même entité provenant de systèmes différents.

SPARQL en tant que protocole est utilisable en soi, mais c'est aussi l'une des composantes de l'architecture plus large des données liées.

[17] https://www.oaforum.org/tutorial/index.php

Le terme Données liées fait référence à un ensemble de meilleures pratiques pour la publication et l'interconnexion de données structurées accessibles à la fois aux humains et aux machines grâce à l'utilisation de la famille de normes RDF (Resource Description Framework) pour l'échange de données et SPARQL pour les requêtes ".[18]

Ces meilleures pratiques ont pour but d'aider à mettre en œuvre les principes de données liées proposés par Tim Berners-Lee :

1.Utilisez les URI pour nommer les choses ;
2.Utilisez les URI HTTP pour que les personnes et les agents utilisateurs puissent se référer aux objets et les consulter (" déréférencer ");
3.Quand quelqu'un cherche une URI, fournissez des informations utiles, en utilisant les normes Web ouvertes telles que RDF, SPARQL ;
4.Incluez des liens vers d'autres choses connexes en utilisant leurs URI lors de la publication sur le Web.

Bien que l'exposition de (méta)données en tant que RDF utilisant des URI pour identifier des objets et fournissant une interface SPARQL satisfasse la plupart des principes de données liées, la mise en œuvre du principe au point 2 ci-dessus nécessite une étape supplémentaire. Linked Data prescrit que les URI sont utilisées pour " nommer " des objets mais aussi pour en faire la recherche : les meilleures pratiques recommandent donc d'utiliser des URL pour les URI, de sorte qu'un URI soit toujours " déréférencé " vers une adresse URL qui peut être " cherchée " via HTTP. En outre, cependant, il prescrit également que l'URL doit servir de réponse à la fois pour les " personnes et les agents utilisateurs ", ce qui signifie que, selon la demande (soit à partir d'un navigateur Web pour les spectateurs humains ou d'un client RDF comme agent utilisateur) l'URL doit servir une réponse HTML ou une réponse RDF (répondant également avec le code 303 tout en redirigeant vers l'autre ressource). Ceci est normalement réalisé par une technique complexe appelée 'négociation de contenu', qui exploite les métadonnées d'en-tête 'type de contenu' dans les requêtes HTTP pour déclencher différentes réponses[19].

Ces protocoles et architectures sont très génériques et l'un (OAI-PMH) a été créé à l'origine dans et pour la communauté des bibliothèques (il est maintenant très utilisé pour le patrimoine culturel en général) et l'autre (SPARQL) est né dans la communauté RDF plus orientée vers l'informatique et autour des idées de Tim Berners-Lee.

D'autres protocoles ont été mis au point dans les communautés scientifiques, en s'appuyant sur des pratiques communes d'échange de données scientifiques, en mettant l'accent sur l'efficacité du transfert des données

[18] W3C. Best Practices for Publishing Linked Data. https://www.w3.org/TR/ld-bp/

[19] http://linkeddata.org/conneg-303-redirect-code-samples

(traitement de grandes quantités de données) et en tirant parti des formats d'échange traditionnels comme NetCDF et HDF5. Un exemple est OPeNDAP.

OPeNDAP ('Projet open-source pour un protocole d'accès aux données du réseau') est " une architecture et un protocole de transport de données largement utilisés par les géo-scientifiques. Le protocole est basé sur HTTP et [...] inclut des normes pour encapsuler des données structurées, annoter les données avec des attributs et ajouter des sémantiques qui décrivent les données. Un client OPeNDAP envoie des requêtes à un serveur OPeNDAP et reçoit divers types de documents ou de données binaires en réponse. Les données sur le serveur sont souvent au format HDF ou NetCDF, mais peuvent être dans n'importe quel format, y compris un format défini par l'utilisateur. Par rapport aux protocoles de transfert de fichiers ordinaires (par exemple FTP), l'un des principaux avantages d'OPeNDAP est la possibilité de récupérer des sous-ensembles de fichiers et d'agréger les données de plusieurs fichiers en une seule opération de transfert''[20]

Le choix des protocoles à mettre en œuvre pour partager les données devrait être influencé principalement par la communauté avec laquelle les données sont censées être partagées et ensuite par des considérations techniques telles que la facilité de mise en œuvre, le soutien pour des formats spécifiques et le soutien général pour les principes de partage des données décrits dans la leçon 4.1.

Par exemple, OPeNDAP est largement utilisé par des organismes gouvernementaux comme la NASA et la NOAA pour la diffusion de données satellitaires, météorologiques et autres données d'observation des sciences de la Terre, ce qui en fait un bon choix pour les institutions qui partagent ces données ou des types de données similaires. D'autre part, si l'on souhaite un partage plus large entre les communautés, OAIPMH pourrait être le choix le plus simple, ou SPARQL et un environnement Linked Data-enabled si faire partie du Web sémantique et de l'Internet des objets est une priorité.

En guise de conclusion de cet excursus sur les différents types d'interopérabilité des données, il est important de noter que la mise en œuvre de toutes les exigences techniques peut être très difficile : à moins que les données ne soient traitées manuellement, converties manuellement en formats interopérables et annotées manuellement avec une sémantique (ce qui ne peut arriver que pour un très petit référentiel), l'exposition des (méta)données se fait via un outil logiciel. Dans la plupart des cas, plutôt que de développer des logiciels ad hoc, il peut être très pratique d'utiliser des outils existants. Puisqu'il peut n'y avoir aucun outil qui implémente toutes les exigences (surtout en termes d'interopérabilité sémantique), il est important d'évaluer les outils existants par rapport à tous les critères expliqués dans cette leçon et de prioriser les critères en fonction des besoins spécifiques du

[20] From Wikipedia: https://en.wikipedia.org/wiki/OPeNDAP

partage de données. Voir la leçon 3.3 pour plus d'informations sur les outils du référentiel de données.