



Ecrire et Lire des métadonnées avec la librairie R geometa

Intervenant: Emmanuel Blondel – Consultant SI
Contact: emmanuel.blondel1@gmail.com

Atelier R métadonnées - Agropolis, Montpellier (France) 8 - 9 Février

Introduction

- Langage R
 - Large communauté d'utilisateurs: gestionnaires de données, scientifiques, statisticiens
 - Pas limité à une audience d'informaticiens et de développeurs
 - Intérêt grandissant au delà du traitement statistique:
 - automatisation de tâches de gestion de données au sens large (conversion, traitement / analyses thématiques)
 - Gestion, traitement et analyse de données spatialisées
- Outils programmatiques pour la gestion de métadonnées
 - Nécessité de se familiariser avec les normes de métadonnées (notamment ISO/OGC)
 - Existants, robustes et certainement recommandés pour la mise en place de systèmes d'information: e.g. Java Apache SIS / GeoToolkit
 - Mais pour la plupart difficiles d'adoption, "réservés" à une petite communautés de spécialistes IT: Développeurs et Architectes Java.
 - Comment mettre à disposition des outils à une large communautés d'utilisateurs avant tout et non de développeurs

Introduction

- Construire progressivement un API dans R pour l'écriture et la lecture de fiches de métadonnées suivant les normes ISO/TC211 et OGC
- Références
 - Normes ISO (certaines approuvées par l'OGC):
 - ISO 19115 (Dataset metadata),
 - ISO 19119 (Service metadata),
 - ISO 19136 (Geographic Markup Language - GML 3.2.1)
 - ISO 19110 (Feature Catalog),
 - ISO 19139 (XML Implementation)
 - Profils ISO 19139 définis pour répondre aux besoins spécifiques d'une communauté
 - Outils existants dans d'autres langages de programmation
 - Java:
 - [GeoAPI](#) / [Apache SIS](#) / [GeoToolKit](#) essentiellement;
 - [GeoTools](#) dans une moindre mesure;
 - Python: [pygeometa](#)

Geometa – le projet

- FOSS (Free and Open Source Software)
- Page web du projet: <https://github.com/eblondel/geometa>
- Statut actuel des développements:
 - Librairie disponible sur CRAN: <https://cran.r-project.org/package=geometa>
 - [Couverture des normes](#)
 - suffisamment exhaustive pour créer des fiches de métadonnées valides selon les normes ISO/OGC, et autres recommandations associées (e.g. INSPIRE)
 - Certaines classes ne sont pas encore gérées, notamment sur le modèle GML
 - Possibilité de gérer des profils communautaires ISO 19139 (en cours)
 - Documentation exhaustive des codes disponible via R, et progressivement sur le [wiki](#)
- Perspectives de développements
 - Ajout de classes manquantes
 - Gestion de nouvelles normes et/ou révisions de normes
 - Ajout d'adaptateurs pour d'autres modèles de métadonnées (e.g. NetCDF, EML)
- Recherche de financements (sponsors et projets techniques)
- Formations techniques

Atelier R métadonnées - Agropolis, Montpellier (France) 8 - 9 Février

Geometa – Comment ça marche?

- Modèle R orienté “objet” (modèle basé sur la librairie “R6”)
- Ensemble de **classes** et de **méthodes** (“fonctions”) associées qui sont accessible via le signe “\$”.
- Principe simple d’utilisation:

1. On crée un objet d’une **classe** donnée avec la méthode **new()**:

```
monObjet <- MaClasse$new()
```

2. Puis on accède aux différentes méthodes via le “\$”. Exemple: méthode pour spécifier le nom d’un objet

```
monObjet$setName("monnom")
```

Geometa – Comment ça marche?

- Modèle basé sur les schémas ISO / OGC des normes suivantes
 - ISO 19115 (Dataset metadata),
 - ISO 19119 (Service metadata),
 - ISO 19136 (Geographic Markup Language - GML 3.2.1)
 - ISO 19110 (Feature Catalog),
 - ISO 19139 (XML Implementation)
- Chaque classe de la norme possède sa classe geometa associée
- Toutes les classes héritent d'une classe `ISOAbstractObject` permettant d'accéder aux fonctions génériques communes à tous les objets
- Principales fonctions:
 - **encode()** → permet d'écrire l'équivalent XML (ISO 19139) de l'objet geometa
 - **decode(xml = xml)** → permet de lire une fiche XML (ISO 19139) en objet geometa
 - **validate()** → permet de tester la conformité du XML produit selon le schéma ISO 19139. Par défaut cette méthode est exécutée dans **encode()**

Geometa – Comment ça marche?

- Terminologie des **classes**

- Approche retenue pour l'instant: Pas de séparations entre les différents *packages* des normes dans un souci de simplification: GMD (Geographic Metadata), SRV (Service Metadata), GFC (Geographic Feature Catalog), etc...
- Séparation simple entre les normes ISO 19115/19119/19110 et le modèle GML (ISO 19136)
- Utilisation de préfixes:
 - **ISOMaClasseAutreQueGML**
 - **GMLMaClasseGML**
- Modèle qui pourra être amené à évoluer dans geometa avec la gestion de nouvelles normes et suivant le retour d'expérience des utilisateurs

Geometa – Comment ça marche?

- Terminologie des **méthodes**
 - Ecriture “Camel”: 1er mot minuscule, les suivants avec la 1ère lettre en majuscule
Exemple: setLeNomDeMonObjet
 - Terminologie qui dépend de la cardinalité des éléments associables à une classe
 - Pour une cardinalité 1-1
 - Exemple: associer le nom d’un objet
 - Utilisation du préfixe “set”: setName
 - Pour une cardinalité 1-N
 - Exemple: ajouter/supprimer des mots-clés
 - Utilisation des préfixes “add” et “del”: addKeyword / delKeyword

Geometa – Chargement de la librairie

- Installer et charger le dernier *release* stable de geometa

```
install.packages("geometa")  
library(geometa)
```

- Installer et charger la version de geometa en cours de développement à partir du dépôt source Github (nécessite la librairie *devtools*)

```
library(devtools)  
install_github("eblondel/geometa", force = TRUE)  
library(geometa)
```

→ Dans le cadre de l'atelier, nous installerons la dernière version de geometa à partir de Github (version 0.3), pour bénéficier des derniers correctifs et améliorations.

Geometa – Pratique - Le début...

- Classe **ISOMetadata**, le début d'une fiche de métadonnées...

```
#création de l'objet ISOMetadata
md = ISOMetadata$new()

#ajout des éléments de base (cardinalité 1-1)
#> identifiant de la fiche
md$setFileIdentifier("my-metadata-identifiant")
#> identifiant de la fiche parente (si existante)
md$setParentIdentifier("my-parent-metadata-identifiant")
#> charset
md$setCharacterSet("utf8")
#> langue de la métadonnée
md$setLanguage("eng")

#> print (résumé de l'objet)
md
```

Geometa – Pratique – Méthode “Print”

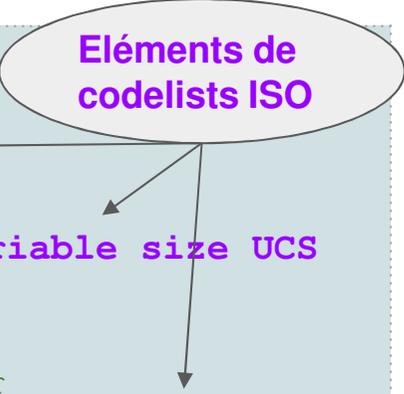
- Résumé d’une fiche (print)
 - Permet de vérifier facilement la métadonnée en cours d’édition

```
<ISOMetadata>
....|-- fileIdentifier: my-metadata-identifiant
....|-- language <ISOLanguage>: eng {English}
.....|-- value: English
....|-- characterSet <ISOCharacterSet>: utf8 {8-bit variable size UCS
Transfer Format, based on ISO/IEC 10646}
.....|-- value: utf8
....|-- parentIdentifier: my-parent-metadata-identifiant
....|-- hierarchyLevel <ISOHierarchyLevel>: dataset {information
applies to the dataset}
.....|-- value: dataset
```

Geometa – Pratique – Méthode “Print”

- Résumé d’une fiche (print)
 - Permet de vérifier facilement la métadonnée en cours d’édition

```
<ISOMetadata>
....|-- fileIdentifier: my-metadata-identifier
....|-- language <ISOLanguage>: eng {English}
.....|-- value: English
....|-- characterSet <ISOCharacterSet>: utf8 {8-bit variable size UCS
Transfer Format, based on ISO/IEC 10646}
.....|-- value: utf8
....|-- parentIdentifier: my-parent-metadata-identifier
....|-- hierarchyLevel <ISOHierarchyLevel>: dataset {information
applies to the dataset}
.....|-- value: dataset
```



The diagram consists of an oval labeled "Éléments de codelists ISO" with three arrows pointing to specific parts of the XML output: one to the language code "eng", one to the character set code "utf8", and one to the hierarchy level code "dataset".

Élément de codelist ajouté par défaut (norme ISO 19115)

Geometa – Pratique – Codelists

- Comment spécifier des valeurs de codelists?

```
md$setHierarchyLevel("test")
```

```
[1] TRUE
```

```
Warning message:
```

```
In super$initialize(xml = xml, id = private$xmlElement, value = value,  
:No ISO 'MX_ScopeCode' codelist entry for value 'test'
```

- La méthode renvoie “TRUE”, avec un warning, mais pas d’erreur...
- L’ajout d’une valeur non comprise dans les valeurs d’une codelist est possible
- Une méthode permettant l’ajout d’une valeur de codelist accepte la valeur ou l’élément:

```
#définition d’un élément personnalisé avec sa description
```

```
level <- ISOHierarchyLevel$new(value = "custom", description = "Custom")
```

```
md$setHierarchyLevel(level)
```

Geometa – Pratique - Codelists

- Comment retrouver les valeurs d'une codelists?

```
#les valeurs possibles de la codelist HierarchyLevel  
ISOHierarchyLevel$values()
```

```
[1] "attribute"           "attributeType"       "collectionHardware"  "collectionSession"  
[5] "dataset"            "series"              "nonGeographicDataset" "dimensionGroup"  
[9] "feature"            "featureType"         "propertyType"        "fieldSession"  
[13] "software"           "service"             "model"               "tile"  
[17] "initiative"         "stereomate"          "sensor"              "platformSeries"  
[21] "sensorSeries"       "productionSeries"    "transferAggregate"   "otherAggregate"
```

```
#tableau des valeurs et des libellés  
ISOHierarchyLevel$values(labels = TRUE)
```

Geometa – Pratique - Ecrire la fiche XML (ISO 19139)

```
md$encode()
```

- Par défaut, `encode()` effectue la validation du XML selon les schémas ISO et informe l'état de conformité de la fiche produite. L'état de conformité est également ajouté en commentaire du XML produit.

```
[geometa][WARN] Element '{http://www.isotc211.org/2005/gmd}MD_Metadata': Missing child element(s).  
Expected is one of ( {http://www.isotc211.org/2005/gmd}hierarchyLevel,  
{http://www.isotc211.org/2005/gmd}hierarchyLevelName, {http://www.isotc211.org/2005/gmd}contact )  
at line 1.
```

```
[geometa][WARN] Object 'ISOMetadata' is INVALID according to ISO 19139 XML schemas!
```

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!--
```

```
Creation date/time: 2018-02-05T18:32:12
```

```
ISO 19139 XML generated by 'geometa' R package - Version 0.3-0
```

```
ISO 19139 XML compliance: NO
```

```
-->
```

```
<!--
```

```
geometa R package information
```

```
Contact: Emmanuel Blondel <emmanuel.blondell@gmail.com>
```

```
URL: https://github.com/eblondel/geometa/wiki
```

```
BugReports: https://github.com/eblondel/geometa/issues
```

```
-->
```

```
À suivre...
```

Geometa – Pratique - Ecrire la fiche XML (ISO 19139)

```
<gmd:MD_Metadata xmlns:gco="http://www.isotc211.org/2005/gco"
xmlns:gfc="http://www.isotc211.org/2005/gfc" xmlns:gmd="http://www.isotc211.org/2005/gmd"
xmlns:gmi="http://www.isotc211.org/2005/gmi" xmlns:gmx="http://www.isotc211.org/2005/gmx"
xmlns:gts="http://www.isotc211.org/2005/gts" xmlns:srv="http://www.isotc211.org/2005/srv"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink">
  <gmd:fileIdentifier>
    <gco:CharacterString>my-metadata-identifiser</gco:CharacterString>
  </gmd:fileIdentifier>
  <gmd:language>
    <gmd:LanguageCode codeList="http://www.loc.gov/standards/iso639-2/" codeListValue="eng"
codeSpace="ISO 639-2">English</gmd:LanguageCode>
  </gmd:language>
  <gmd:characterSet>
    <gmd:MD_CharacterSetCode
codeList="http://www.isotc211.org/2005/resources/Codelist/ML_gmxCodelists.xml#MD_CharacterSetCode"
codeListValue="utf8">utf8</gmd:MD_CharacterSetCode>
  </gmd:characterSet>
  <gmd:parentIdentifier>
    <gco:CharacterString>my-parent-metadata-identifiser</gco:CharacterString>
  </gmd:parentIdentifier>
  <gmd:hierarchyLevel>
    <gmd:MD_ScopeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelists.xml#MX_ScopeCode"
codeListValue="custom_level" codeSpace="ISOTC211/19115">Custom level</gmd:MD_ScopeCode>
  </gmd:hierarchyLevel>
</gmd:MD_Metadata>
```

Atelier R métadonnées - Agropolis, Montpellier (France) 8 - 9 Février

Geometa – Pratique - Exporter la fiche XML

- Comment exporter une fiche XML

```
library(XML)
cat(saveXML(md$encode()), file = "metadata.xml")
```

A partir de geometa ≥ 0.3 (installable depuis Github), une nouvelle fonction **save(file)** simplifie cet export:

```
md$save("metadata.xml")
```

Geometa – Pratique - Lire/éditer une fiche XML

- Comment lire une fiche XML?

```
library(XML)
xml <- xmlParse("metadata.xml")
md2 <- ISOMetadata$new()
md2$decode(xml = xml)
```

Je peux lire une fiche... donc je peux l'éditer, la mettre à jour... et

l'exp

```
md2$setDateStamp(Sys.time())
```

Geometa – Pratique - Comparer des fiches

- Je souhaite comparer des fiches de métadonnées pour voir si elles sont identiques. Exemple précédent:

```
ISOAbstractObject$compare(md, md2)
```

```
[1] TRUE
```

Geometa – Pratique - Créer un objet contact

```
rp <- ISOResponsibleParty$new()
rp$setIndividualName("someone")
rp$setOrganisationName("organization")
rp$setPositionName("position")
rp$setRole("owner")
contact <- ISOContact$new()
phone <- ISOTelephone$new()
phone$setVoice("myphonenummer")
phone$setFacsimile("myfacsimile")
contact$setPhone(phone)
address <- ISOAddress$new()
address$setDeliveryPoint("theaddress")
address$setCity("thecity")
address$setPostalCode("111")
address$setCountry("France")
address$setEmail("someone@theorg.org")
contact$setAddress(address)
res <- ISOOnlineResource$new()
res$setLinkage("http://www.somewhereovertheweb.org")
res$setName("somenam")
contact$setOnlineResource(res)
rp$setContactInfo(contact)
```

Geometa – Pratique - Créer un objet de mots-clés

```
#citer un thésaurus
th <- ISOCitation$new()
th$setTitle("General")
th$setAlternateTitle("General")
th_date <- ISODate$new()
th_date$setDate(Sys.time())
th_date$setDateType("revision")
th$addDate(th_date)
```

#Exemple 1 - ajout de simples mots-clés

```
kwds1 <- ISOKeywords$new()
kwds1$addKeyword("keyword1")
kwds1$addKeyword("keyword2")
kwds1$setKeywordType("theme")
kwds1$setThesaurusName(th)
```

#Exemple 2 - ajout de mots-clés avec références URIs

```
kwds2 <- ISOKeywords$new()
kwd1 <- ISOAnchor$new(name = "keyword1", href = "http://myvocabulary.geometa/keyword1")
kwds2$addKeyword(kwd1)
kwd2 <- ISOAnchor$new(name = "keyword2", href = "http://myvocabulary.geometa/keyword2")
kwds2$addKeyword(kwd2)
kwds2$setKeywordType("theme")
kwds2$setThesaurusName(th)
```

Geometa – Pratique - Créer une généalogie

```
dq <- ISODataQuality$new()
scope <- ISOScope$new()
scope$setLevel("dataset")
dq$setScope(scope)
#créer un objet 'lineage'
lineage <- ISOLineage$new()
lineage$setStatement("Descriptif des étapes réalisées")

#ajouter des étapes
ps1 <- ISOProcessStep$new()
ps1$setDescription("Etape 1 - Acquisition de données")
ps1$setDateTime(Sys.time())
#ps1$addProcessor()
lineage$addProcessStep(ps1)

ps2 <- ISOProcessStep$new()
ps2$setDescription("Etape 2 - Qualification de données")
ps2$setDateTime(Sys.time())
#ps2$addProcessor()
lineage$addProcessStep(ps2)

dq$setLineage(lineage)
```

Geometra – Pratique – Autre exemples?

?ISOMaClasse

Merci pour votre attention

Atelier R métadonnées - Agropolis, Montpellier (France) 8 - 9 Février