METADATA STATISTICS

Aims

We have a set of resources (a resource is an object of the class Resource) satisfying certain metadata schema (class MetadataSchema). This set of resources is the statistical sample (class ResourceSample).

• Make a count of the number of valid resources: a resource is valid if it is parseable (for xml, which is well-formed) and has the required metadata in the schema.

• Given the name of a schema metadata (eg title), the program should be able to tell the number of resources of the sample containing a valid value for that attribute.

Clases

Resource: Is a resource representation.

- Attributes
  - MetadataSchema schema: Metadata schema that resource follows
  - .HashMap<String,Object> metadata: resource metadata, Key is a String with the metadata name and an object value with this metadata value.
- Methods:
  - get: it receives an string: the metadata name and returns the metadata value i fis define don the schema, if not, a NoSuchFieldException exception is
  - set: it recieves an string with the metadata name and an object with its value and stores it in the hashmap

MetadataSchema: Clase abstracta que representa un esquema de metadatos, cada nuevo esquema de metadatos deberá extender esta clase.

- Methods:
  - getFields: it returns an ArrayList with all the metadata names (title, description, date, author, etc).

ResourceSource interface: represents a resource. <Resource> Extends Iterable.

AbstractResourceSource: abstract implementation of the previous interface will inherit the two implementations as discussed above.

- Attributes:
  - MetadataParser parser: it parses all the resources of source..

For each call to next () reads a new resource, it parses it and returns an object of type Resource.

FSResourceSourceThis extends from AbstractResourceSource and gets the resources xml from files in the file system.Atributos:

  - String path: the file system path where you will find the directory with the xmls..

JSONAPIResourceSource: extends from AbstractResourceSource and gets the resources of the json API (such as VOA3R) ..

MetadataParser Interface: interface to be implemented by all parsers: **DCMetadataParser**, **QDCMetadataParser**, **Voa3rAPMetadataParser**, etc

- Methods:
  - parse: receives an Object with the resource (either a File to the xml case) returns the metadata Resource refilled.

ResourceSample: represents a statistical sample of resources.

- Attributes
  - name: simple name.
  - schema: MetadataSchema following sample resources.
  - source: MetadataSchema following sample resources.
- Methods
  - size: number of sample resources
  - countValid: Number of valid resources.
  - countHaveField: number of resources that have a certain metadata (eg title) that is passed as a parameter (String).
  - constructor: el constructor is called metadata schema and the source of funds and private method called process.
  - - process: privete, Itera the source (remember that implements Iterable MetadataSource) and each Resource will be saving if it is valid and whether or not a valid value for each schema metadata.

How to extend

To extend the program to other metadata schemes would only have to implement a new class that implements the interface and another that extends MetadataParser MetadataSchema to define new metadata schema.

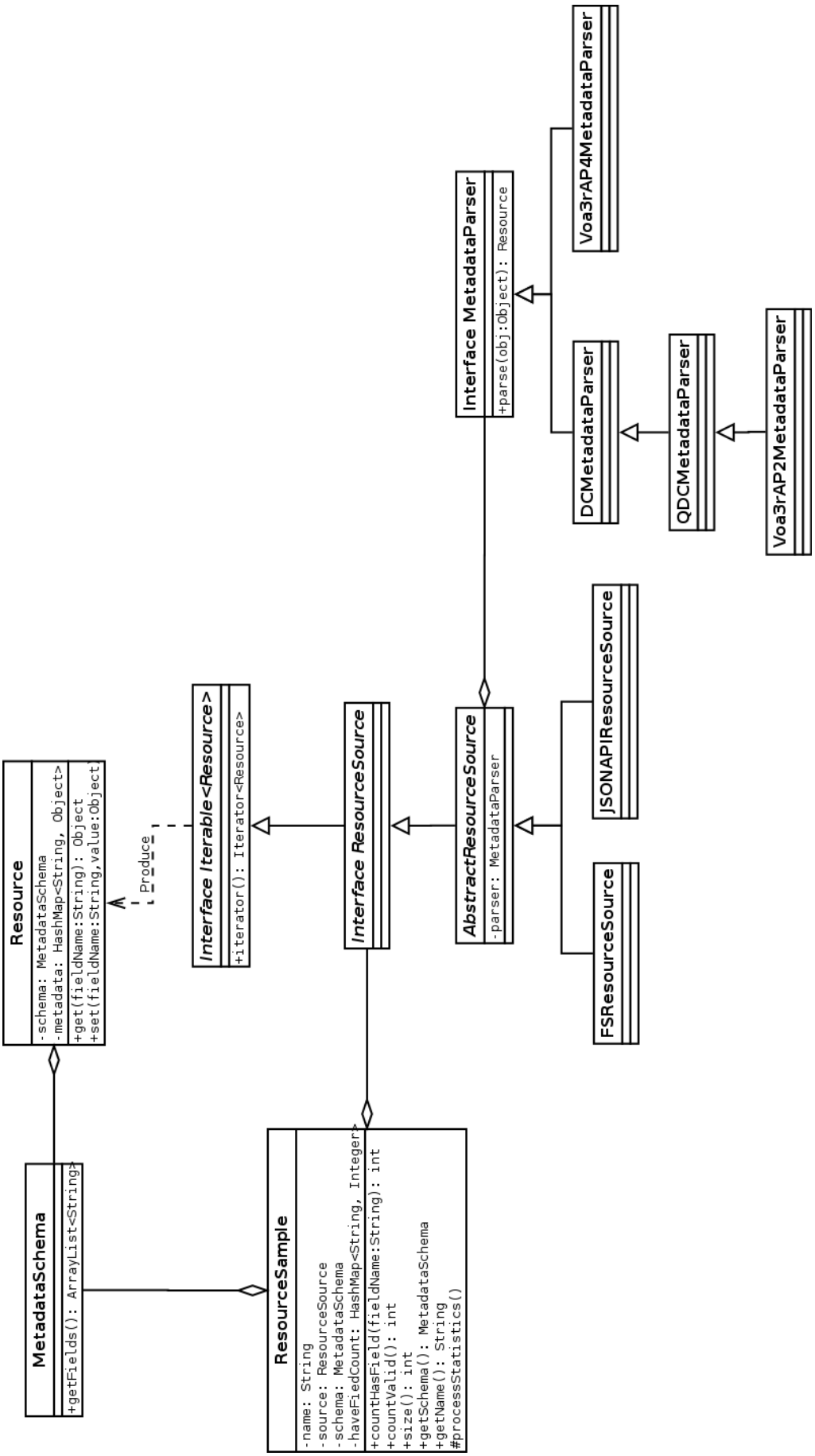
If we extend it to other resources would only have to implement a new class that extends or implements the interface AbstractResourceSource resourceSource.

How to run

Java –jar **metadata-statistics.jar** <<**XML_PATH**>><< **METADATA SCHEMA** >>

- **XML_PATH**= Full, path of the xml repository. As this script creates a resource representation, the folder(s) – This will work correctly depending on the memory amount allocated to the JVM and the number of resources included in the folder (including subfolders)
- **METADATA SCHEMA** = the xml metadata Schema:
  voa3rAP2 | voa3rAP4 | dublinCore | Agris

CLASS DIAGRAM


Resource
-schema: MetadataSchema
-metadata: HashMap<String, Object>
+get(fieldName:String): Object
+set(fieldName:String,value:Object)
Produce
Interface Iterable<Resource>
+iterator(): Iterator<Resource>
Interface ResourceSource
AbstractResourceSource
-parser: MetadataParser
Interface MetadataParser
+parse(obj:Object): Resource
Voa3rAP4MetadataParser
DCMetadataParser
QDCMetadataParser
Voa3rAP2MetadataParser
JSONAPIResourceSource
FSResourceSource
MetadataSchema
+getFields(): ArrayList<String>
ResourceSample
-name: String
-source: ResourceSource
-schema: MetadataSchema
-haveFiedCount: HashMap<String, Integer>
+countHasField(fieldName:String): int
+countValid(): int
+size(): int
+getSchema(): MetadataSchema
+getName(): String
#processStatistics()