**URI Design and Mappings in StatSpace**

Ba Lam Do

June 17th 2016

Contents

[1. URIs patterns for metadata description 2](#_Toc453957547)

[1.1. Patterns 2](#_Toc453957548)

[1.2. Example 2](#_Toc453957549)

[2. List of components used to create metadata 2](#_Toc453957550)

[2.1. Dimensions 2](#_Toc453957551)

[2.2. Measure 3](#_Toc453957552)

[2.3. Attribute 3](#_Toc453957553)

[3. URI patterns of code lists 3](#_Toc453957554)

[3.1. Code list of reference area dimension (cl\_area) 3](#_Toc453957555)

[3.2. Code list of reference period dimension (cl\_period) 3](#_Toc453957556)

[3.3. Code list of age dimension (cl\_age) 4](#_Toc453957557)

[3.4. Code list of education level dimension (cl\_educationLev) 5](#_Toc453957558)

[3.5. Code list of occupation dimension (cl\_occupation) 5](#_Toc453957559)

[3.6. Code list of currency dimension (cl\_currency) 6](#_Toc453957560)

[3.7. Code list of civil status dimension (cl\_civilStatus) 6](#_Toc453957561)

[3.8. Code list of freq dimension (cl\_freq) 6](#_Toc453957562)

[3.9. Code list of sex dimension (cl\_sex) 7](#_Toc453957563)

[3.10. Code list of activity dimension (cl\_activity) 7](#_Toc453957564)

[3.11. Code list of expenditure dimension 8](#_Toc453957565)

[3. 11.1. Code list of COICOP (cl\_coicop) 8](#_Toc453957566)

[3.11.2. Code list of COFOG (cl\_cofog) 8](#_Toc453957567)

[3. 11.3. Code list of COPNI (cl\_copni) 9](#_Toc453957568)

[3. 11.4. Code list of COPP (cl\_copp) 9](#_Toc453957569)

[3.12. Code list of unit of measure (cl\_unitMeasure) 9](#_Toc453957570)

[3.13. Code list of subject (cl\_subject) 10](#_Toc453957571)

[4. Methods for identifying co-reference 10](#_Toc453957572)

[4.1. Identifying co-reference URIs for dimensions 10](#_Toc453957573)

[4.2. Identifying co-reference URIs for values of a dimension 11](#_Toc453957574)

[4. 3. Identifying co-reference URIs for values of area reference dimension 13](#_Toc453957575)

[4.4. Identifying co-reference URIs for unit of measure 17](#_Toc453957576)

# URIs patterns for metadata description

## 1.1. Patterns

Base URI: http://statspace.linkedwidgets.org

|  |  |  |
| --- | --- | --- |
| No | Patterns | Description |
| 1 | /metadata/{datasource-dataset} | URI of a metadata |
| 2 | /dataset/{datasource-dataset} | URI of a dataset |
| 3 | /codelist/cl\_{name} | URI of a code list |
| 4 | /dimension/{name} | URI of a dimension |

## 1.2. Example

|  |  |  |
| --- | --- | --- |
| No | Patterns | Description |
| 1 | <http://statspace.linkedwidgets.org/metadata/ONS-Population-1851-2014> | URI of a metadata |
| 2 | <http://statspace.linkedwidgets.org/dataset/ONS-Population-1851-2014> | URI of a dataset |
| 3 | <http://statspace.linkedwidgets.org/codelist/cl_area> | URI of a code list |
| 4 | <http://statspace.linkedwidgets.org/dimension/activity> | URI of dimension of economic activity |

# List of components used to create metadata

## 2.1. Dimensions

|  |  |  |
| --- | --- | --- |
| No | URI | Label |
| 1 | http://purl.org/linked-data/sdmx/2009/dimension#refArea | Reference Area |
| 2 | http://purl.org/linked-data/sdmx/2009/dimension#refPeriod | Reference Period |
| 3 | http://purl.org/linked-data/sdmx/2009/dimension#age | Age |
| 4 | http://purl.org/linked-data/sdmx/2009/dimension#educationLev | Education Level |
| 5 | http://purl.org/linked-data/sdmx/2009/dimension#occupation | Occupation |
| 6 | http://purl.org/linked-data/sdmx/2009/dimension#currency | Currency |
| 7 | http://purl.org/linked-data/sdmx/2009/dimension#civilStatus | Civil Status |
| 8 | http://purl.org/linked-data/sdmx/2009/dimension#freq | Frequency |
| 9 | http://purl.org/linked-data/sdmx/2009/dimension#sex | Sex |
| 10 | http://statspace.linkedwidgets.org/dimension/activity | Economic Activity |
| 11 | http://statspace.linkedwidgets.org/dimension/expenditure | Expenditure |

## 2.2. Measure

|  |  |  |
| --- | --- | --- |
| No | URI | Label |
| 1 | http://purl.org/linked-data/sdmx/2009/measure#obsValue | Observation |

## 2.3. Attribute

|  |  |  |
| --- | --- | --- |
| No | URI | Label |
| 1 | http://purl.org/linked-data/sdmx/2009/attribute#unitMeasure | Unit of Measure |

# URI patterns of code lists

Base URIs:

* <http://reference.data.gov.uk> for code list of reference period dimension
* <http://statspace.linkedwidgets.org/codelist> for the remaining code lists

## 3.1. Code list of reference area dimension (cl\_area)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_area>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_area.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_area/{Country}/{Area in level 2}/…/{Area in level n} | URI of a geographical area |

* Example

|  |  |
| --- | --- |
| URI | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_area/Austria/Vienna> | Vienna, Austria |

## 3.2. Code list of reference period dimension (cl\_period)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_period>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_period.ttl>
* Patterns

|  |  |
| --- | --- |
| URI Patterns | Description |
| /id/gregorian-year/{year} | URI for a year |
| /id/gregorian-half/{year}-{half} | URI for one-half year |
| /id/gregorian-quarter/{year}-{quarter} | URI for a quarter |
| /id/gregorian-month/{year}-{month} | URI for a month |
| /id/gregorian-day/{year}-{month}-{day} | URI for a day |
| /id/gregorian-hour/{year}-{month}-{day}T{hour} | URI for a hour |
| /id/gregorian-hour/{year}-{month}-{day}T{hour}:{min} | URI for a minute |
| /id/gregorian-hour/{year}-{month}-{day}T{hour}:{min}:{sec} | URI for a second |
| //id/gregorian-week/{year}-{week} | URI for a week |
| /id/gregorian-instant/{dateTime} | URI for an instant |
| /id/gregorian-interval/{dateTime}/{duration} | URI for a duration |

* Example

|  |  |
| --- | --- |
| URIs | Description |
| <http://reference.data.gov.uk/id/gregorian-year/2016> | URI for year 2016 |
| <http://reference.data.gov.uk/id/gregorian-day/2016-01-01> | URI for day 01/01/2016 |

## 3.3. Code list of age dimension (cl\_age)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_age>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_age.ttl>
* Patterns

|  |  |
| --- | --- |
| URI Patterns | Description |
| /cl\_age/Y{n}, n=0, 1, 2,…,105 | URI for an individual age |
| /cl\_age/Y{n}T{n+4}, n=0, 5, 9,…,105 | URI for an age group (5 years) |
| /cl\_age/Y{n}T{n+9}, n=25, 35,…, 95 | URI for an age group (10 years) |
| /cl\_age/Y\_GE\_{n}, n=65, 70,…,90 | URI for an age group (equal or above a specific age) |
| /cl\_age/Y\_LE\_{n}, n=15, 20 | URI for an age group (under a specific age) |
| /cl\_age/TOTAL | URI for the top concept |
| /cl\_age/UNK | URI for an unknown age |

* Example

|  |  |
| --- | --- |
| URIs | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_age/Y80> | URI for age 80 |
| <http://statspace.linkedwidgets.org/codelist/cl_age/Y80T84> | URI for an age group from 80 to 84 |

## 3.4. Code list of education level dimension (cl\_educationLev)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_educationLev>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_educationLev.ttl>
* URIs of values

|  |  |
| --- | --- |
| URIs | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L0> | Pre-primary education |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L1> | Primary education |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L2> | Lower secondary |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L3> | Upper secondary |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L4> | Post-secondary non-tertiary education |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L5> | Short-cycle tertiary education |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L6> | Bachelor or equivalent |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L7> | Master or equivalent |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L8> | Doctoral or equivalent |
| <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L9> | Not elsewhere classified |

## 3.5. Code list of occupation dimension (cl\_occupation)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_occupation>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_occupation.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_occupation/{code} | URI of an occupation |

* Example

|  |  |
| --- | --- |
| URIs | Description |
| <http://linkedwidgets.org/resource/codelist/cl_occupation/OC1> | URI for Managers |
| <http://statspace.linkedwidgets.org/codelist/cl_occupation/OC11> | URI for Chief executives, senior officials and legislators |

## 3.6. Code list of currency dimension (cl\_currency)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_currency>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_currency.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_currency/{ISO 4217 code} | URI for a currency |

* Example

|  |  |
| --- | --- |
| URI Pattern | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_currency/AED> | URI for United Arab Emirates dirham |
| <http://statspace.linkedwidgets.org/codelist/cl_currency/EUR> | URI for Euro |

## 3.7. Code list of civil status dimension (cl\_civilStatus)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_civilStatus>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_civilStatus.ttl>
* URIs of values

|  |  |
| --- | --- |
| URIs | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/D> | Divorced person |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/E> | Person whose registered partnership was legally dissolved |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/L> | Leggaly separated person |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/M> | Married person |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/P> | Person in Registerd partnership |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/Q> | Person whose registered partnership ended with the death of the partner |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/S> | Single person |
| <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/W> | Widowed person |

## 3.8. Code list of freq dimension (cl\_freq)

* URI of the code list: http://purl.org/linked-data/sdmx/2009/code#freq
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_freq.ttl>
* URIs of values

|  |  |
| --- | --- |
| URIs | Description |
| <http://purl.org/linked-data/sdmx/2009/code#freq-H> | Hourly |
| <http://purl.org/linked-data/sdmx/2009/code#freq-D> | Daily |
| <http://purl.org/linked-data/sdmx/2009/code#freq-N> | Minutely |
| <http://purl.org/linked-data/sdmx/2009/code#freq-S> | Half yearly, semester |
| <http://purl.org/linked-data/sdmx/2009/code#freq-A> | Annual |
| <http://purl.org/linked-data/sdmx/2009/code#freq-Q> | Quarterly |
| <http://purl.org/linked-data/sdmx/2009/code#freq-M> | Monthly |
| <http://purl.org/linked-data/sdmx/2009/code#freq-B> | Daily-business week |
| <http://purl.org/linked-data/sdmx/2009/code#freq-W> | Weekly |

## 3.9. Code list of sex dimension (cl\_sex)

* URI of the code list: http://purl.org/linked-data/sdmx/2009/code#sex
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_sex.ttl>
* URIs of values

|  |  |
| --- | --- |
| URI Pattern | Description |
| <http://purl.org/linked-data/sdmx/2009/code#sex-M> | URI for Male gender |
| <http://purl.org/linked-data/sdmx/2009/code#sex-F> | URI for Female gender |
| <http://purl.org/linked-data/sdmx/2009/code#sex-T> | URI for Total |
| <http://purl.org/linked-data/sdmx/2009/code#sex-U> | URI for Unknown gender |
| <http://purl.org/linked-data/sdmx/2009/code#sex-N> | URI for not applicable gender |

## 3.10. Code list of activity dimension (cl\_activity)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_activity>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_activity.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_activity/{code} | URI of an economic activity |

* Example

|  |  |
| --- | --- |
| URI Pattern | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_activity/A> | URI for activity of Agriculture, forestry, and fishing |
| <http://statspace.linkedwidgets.org/codelist/cl_activity/A01> | URI for activity of Crop and animal production, hunting and related service activities |

## 3.11. Code list of expenditure dimension

* Contains 4 code lists
  + Classification of individual consumption by purpose (COICOP),
  + Classification of the functions of government (COFOG),
  + Classification of the purposes of non-profit institutions serving households (COPNI) and,
  + Classification of outlays of producers by purpose (COPP).

### 3. 11.1. Code list of COICOP (cl\_coicop)

* URI of the code list COICOP: <http://statspace.linkedwidgets.org/codelist/cl_coicop>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_coicop.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_coicop/{code} | URI for expenditure of an individual |

* Example

|  |  |
| --- | --- |
| URI | Description |
| [http://statspace.linkedwidgets.org/codelist/cl\_coicop/CP01](http://linkedwidgets.org/resource/codelist/cl_coicop/CP01) | URI for expenditure of Food and non-alcoholic beverages |

### 3.11.2. Code list of COFOG (cl\_cofog)

* URI of the code list COFOG: <http://statspace.linkedwidgets.org/codelist/cl_cofog>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_cofog.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_cofog/{code} | URI for expenditure of government |

* Example

|  |  |
| --- | --- |
| URI | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_cofog/GF01> | URI for expenditure of government for General public services |

### 3. 11.3. Code list of COPNI (cl\_copni)

* URI of the code list COPNI: <http://statspace.linkedwidgets.org/codelist/cl_copni>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_copni.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_copni/{code} | URI for expenditure of non-profit organization |

* Example

|  |  |
| --- | --- |
| URI | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_copni/PN1> | URI for expenditure of non-profit organization for Housing |

### 3. 11.4. Code list of COPP (cl\_copp)

* URI of the code list COPP: <http://statspace.linkedwidgets.org/codelist/cl_copp>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_copp.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_copp/{code} | URI for expenditure of producer |

* Example

|  |  |
| --- | --- |
| URI | Description |
| <http://statspace.linkedwidgets.org/codelist/cl_copp/PP1> | URI for expenditure of producer for Outlays on infrastructure |

## 3.12. Code list of unit of measure (cl\_unitMeasure)

* URI of the code list: [http://statspace.linkedwidgets.org/codelist/cl\_unitMeasure](http://linkedwidgets.org/resource/codelist/cl_unitMeasure)
* RDF Data:  [http://statspace.linkedwidgets.org/code/cl\_unitMeasure.ttl](http://statspace.linkedwidgets.org/code/cl_activity.ttl)
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_unitMeasure/{unit} | URI of a unit |

* Example

|  |  |
| --- | --- |
| URI Pattern | Description |
| <http://linkedwidgets.org/resource/codelist/cl_unitMeasure/P1> | URI for unit “People” |
| [http://statspace.linkedwidgets.org/codelist/cl\_unitMeasure/TU](http://linkedwidgets.org/resource/codelist/cl_unitMeasure/TU) | URI for unit “Twenty-Foot Equivalent” |

## 3.13. Code list of subject (cl\_subject)

* URI of the code list: <http://statspace.linkedwidgets.org/codelist/cl_subject>
* RDF Data: <http://statspace.linkedwidgets.org/code/cl_subject.ttl>
* Pattern

|  |  |
| --- | --- |
| URI Pattern | Description |
| /cl\_subject/{code} | URI of a subject |

* Example

|  |  |
| --- | --- |
| URI Pattern | Description |
| [http://statspace.linkedwidgets.org/codelist/subject/AG.SRF.TOTL.K2](http://linkedwidgets.org/resource/codelist/subject/AG.SRF.TOTL.K2) | URI for subject “Surface area (sq. km)” |
| [http://statspace.linkedwidgets.org/codelist/subject//SP.POP.TOTL](http://linkedwidgets.org/resource/page/codelist/subject/SP.POP.TOTL) | URI for subject “Population, total” |

# Methods for identifying co-reference

## 4.1. Identifying co-reference URIs for dimensions

Input:

* + a URI and its label
  + e.g., ex:ref-Area, Reference area

Output:

* + co-reference URI of this URI
  + e.g., <http://purl.org/linked-data/sdmx/2009/dimension#refArea>

|  |  |  |
| --- | --- | --- |
| No | Dimension | Methods |
| 1 | Reference Area | * Define a set of possible names for representing this dimension. * Check the existence of one name in the input URI or label * {"ref-area", "refarea", "country", "refdistrict", "refstate", "place", "geocode", "region", “reference area”} |
| 2 | Reference Period | * {"ref-period", "ref-date","ref-year","refperiod", "timeperiod", "date", "year", "time-period", “time period”} |
| 3 | Age | * {“/age","\_age", "#age","refage”} |
| 4 | Education Level | * {"educationlev", "edulev", “education level”}; |
| 5 | Occupation | * {“occupation”} |
| 6 | Currency | * {“currency”} |
| 7 | Civil Status | * {“civil”, “status”} |
| 8 | Frequency | * {"freq”} |
| 9 | Sex | * {"sex", "gender"} |
| 10 | Economic Activity | * {"activity" ,”economy”} |
| 11 | Expenditure | * cofog = {"funcofgov", "function of government", "functions of government"}; * coicop = {"indvcons", "individual consumption"}; * copp = {"outlayofproducer", "outlay of producer", "outlays of producer"}; * copni= {"purposeofnpi", "purpose of non-profit institution", "purposes of non-profit institution" }; |

## 4.2. Identifying co-reference URIs for values of a dimension

Input:

* + a URI and its label
  + e.g., ex:AT, Austria

Output:

* + co-reference URI of this URI
  + e.g., <http://statspace.linkedwidgets.org/codelist/cl_area/Austria>

|  |  |  |
| --- | --- | --- |
| No | Value of dimension | Methods |
| 1 | Reference Area | * Detect hierarchical relation among areas in the input dataset * Algorithm: presented in section 4.3 |
| 2 | Reference Period | * Use Patterns e.g.,   Interval: [1-9][0-9]{3}-[1-9][0-9]{3}  Year: [1-9][0-9]{3}  Month:[1-9][0-9]{3}-[0-1][0-9]  Quarter: [1-9][0-9]{3}-Q[1-4]  Date: [1-9][0-9]{3}-[0-1][0-9]-[0-3][0-9] |
| 3 | Age | * Literal values e.g., {Value}^^http://www.w3.org/2001/XMLSchema#long => identify value, then builing corresponding URI * URIs e.g., ex:Y{Value1}-Y{Value2}, ex:{Value}%2B => identify age group, then building correpsonding URI |
| 4 | Education Level | * Compare label and code of the URI with values in the corresponding code list. * For example: if the URI ends with code “L0” or its label contains “Pre-primary education” => map to <http://statspace.linkedwidgets.org/codelist/cl_educationLev/L0> |
| 5 | Occupation | * Compare label and code of the URI with values in the corresponding code list. * For example: if the URI ends with code “OC11” or its label contains “Chief executives, senior officials and legislators => map to <http://statspace.linkedwidgets.org/codelist/cl_occupation/OC11> |
| 6 | Currency | * Compare label and code of the URI with values in the corresponding code list. * For example: if the URI ends with code “EUR” or its label contains “EURO” => map to <http://statspace.linkedwidgets.org/codelist/cl_currency/EUR> |
| 7 | Civil Status | * Compare label and code of the URI with values in the corresponding code list. * For example: if the URI ends with code “D” or its label contains “Divorced person” => map to <http://statspace.linkedwidgets.org/codelist/cl_civilStatus/D> |
| 8 | Frequency | * Compare label and code of the URI with values in the corresponding code list. * For example: if the URI ends with code “H” or its label contains “Hourly” => map to * <http://purl.org/linked-data/sdmx/2009/code#freq-H> |
| 9 | Sex | * Compare label and code of each URI with values in the corresponding code list. * For example: if the URI ends with code “M” or its label contains “Male” => map to <http://purl.org/linked-data/sdmx/2009/code#sex-M> |
| 10 | Economic Activity | * Compare label and code of each URI with values in the corresponding code list. * For example: if the URI ends with code “A” or its label contains “Agriculture, forestry, and fishing” => map to <http://statspace.linkedwidgets.org/codelist/cl_activity/A> |
| 11 | Expenditure | * Compare label and code of each URI with values in the corresponding code list. * For example: if the URI ends with code “GF01” or its label contains “General public services” => map to <http://statspace.linkedwidgets.org/codelist/cl_cofog/GF01> |

## 4. 3. Identifying co-reference URIs for values of area reference dimension

Input:

* + A set contains URIs and their lables
  + L ={l1, l2,…,ln}, li ={uri\_li, label\_li}

Output:

* + Mapping L to G
  + G = {g1, g2,…,gn}, gi={uri\_gi, label\_gi, lat\_gi, lng\_gi, type\_gi}

Methods

1. **Procedure** **sortInAscendingOrder(L)**

// sort areas in L in ascending order of uri

1. **Procedure isBroaderArea(uri\_li, uri\_lj)**

//return true if uri\_li is a broader area of uri\_lj

if (uri\_lj startsWith(uri\_li + “/”) &&

length(uri\_lj) > length(uri\_li) + 1 && uri\_lj.substring(length(uri\_li)+1).indexOf(“/”)==-1)

return true;

if(uri\_lj startsWith(uri\_li)&&length(uri\_lj) == length(uri\_li) + 1

return true;

return false;

1. **Procedure indexOfBroaderArea(L, li)**

//remain index of the area which is a broader area of li in list L (use *isBroaderArea* procedure)

1. **Procedure filterByDistance(G, gi)**

//remain only one result in gi, that is, the one which has the minimal distance to adjacent areas gi-2, gi-1, gi+1, gi+2

1. **Procedure getLevel(uri)**

//split the input based on “/” character, and return the length/size of the returned list

1. **Procedure isGoogleBroaderArea(uri\_gi, uri\_gj)**

//return true if uri\_gi is a broader area of uri\_gj

if (uri\_gj startsWith(uri\_gi + “/”){

if(length(uri\_gj) > length(uri\_gi) + 1 && uri\_gj.substring(length(uri\_gi)+1).indexOf(“/”)==-1)

return true;

if(uri\_gj.contains("/") && uri\_gi.contains("/")){

String[] area1 = uri\_gj.split("/"); //Austria/Vienna

String[] area2 = uri\_gi.split("/"); //Austria/Vienna/Vienna

remove duplicate elements in area1, area2

if(arrUri2.length==arrUri1.length+1)

return true;

}

}

return false;

1. **Procedure identifyMapping(L)**

// L ={l1, l2,…,ln}, li ={uri\_li, label\_li}

// G = {g1, g2,…,gn}, gi={uri\_gi, label\_gi, lat\_gi, lng\_gi, type\_gi}

// query Google’s geocoding API

boolean bUseBroaderArea

string sLabel, sQuery

int level, googleLevel

sortInAscendingOrder(L)

//step 1. query labels with Google’s geocoding API

for each area lj in L do

//construct a query for this area

i = indexOfBroaderArea(L, lj)

if(i!=-1) then

if(label\_li != label\_lj) then

sLabel = label\_lj + “ “ + label\_li

bUseBroaderArea = true

else

k = indexOfBroaderArea(L, li)

if(k!=-1) then

if(label\_li != label\_lj) then

sLabel = label\_lj + “ “ + label\_li

bUseBroaderArea = true

else

sLabel = label\_lj

bUseBroaderArea = false

end if

else

sLabel = label\_lj

bUseBroaderArea = false

end if

end if

else

sLabel = label\_lj

bUseBroaderArea = false

end if

//query Google

sQuery = “https://maps.googleapis.com/maps/api/geocode/xml ?address”= + sQuery

responseCode = URL(*sQuery) //query this URL*

if(responseCode==200) then

gj <- results from the query

else

if(bUseBroaderArea==true) then

sQuery = “https://maps.googleapis.com/maps/api/

geocode/xml?address=” + label\_lj

responseCode = URL(*sQuery) //query this URL*

if(responseCode==200) then

gj <- results from the query

end if

end if

end if

end for

//step 2. identify mappings

//step 2.1. identify mappings for areas which do not have broader areas

for each area lj in L do

//step 2.1.1. check if this is a country

if(label\_lj is name\_of\_a\_country or uri\_lj.endswith(name\_of\_a\_country) or

uri\_lj.endswith(iso-alpha-2) or

uri\_lj.endswith(iso-alpha-3)) then

set uri\_gj based on name\_of\_this\_country

set type\_gj = administrative-area

continue;

end if

if(indexOfBroaderArea(L, lj)==-1) then

if(size(gj)==0) then

if(label\_gj!=””) then

set uri\_gj to “/undefined/”+label\_gj

else

set uri\_gj to “/undefined/”+ending-part-of uri\_lj

end if

type\_gj = non-administrative-area

else

if(size(gj)==1) then

type\_gj = administrative-area

else

//filter by level of area

level = getLevel(*uri\_lj*)

googleLevel = -1

for each area lk in L do

if(getLevel(uri\_lk)==level && size(uri\_gk)==1)) then

googleLevel = getLevel(uri\_gk)

end if

end for

//assume *gj* = *{g*j1*,..,gjm}*, g*jk* = (*uri\_gjk, label...*)

if(googleLevel!=-1) then

for each gjk in gj

if(googleLevel!=getLevel(uri\_gjk)) then

remove gjk

end if

end for

end if

if(size(gj)==0) then

type\_gj = non-administrative-area

else

if(size(gj)==1) then

type\_gj = administrative-area

else

//filter by narrower areas, assume *gj*=*{g*j1*,..,gjm}*,g*jk*=(*uri\_gjk, label...*)

in L, identify narrower areas of uri\_lj

=>set of indexes{j1,..,jt}

in G, identify uri\_gjk that has the largest apperance in {gj1, gj2,.., gjt)

keep gjk in gj and remove other results

type\_gj = administrative-area

end if

end if

end if

end if

end if

end for

//step 2.2. identify mappings for areas which have broader areas

for each area lj in L do

if(size(gj)==1 && type\_gj!=null) then

continue;

end if

i = indexOfBroaderArea(L, lj)

if(size(gj)==1 && type\_gj==non-administrative-area) then

i = indexOfBroaderArea(L, li)

end if

if(size(gi)==1 && type\_gi==administrative-area) then

assume *gj* = *{g*j1*,..,gjm}*, g*jk* = (*uri\_gjk, label...*)

//filter by boarder area  
 for each result *gjk in* *gj* do

if ( !isBroaderAreaInGoogleGeo(uri\_*gi*, *uri\_gjk*)) then   
 remove *gjk* from *gj*  
 end if

end for

//filter by distance

if(size(gj)>1) then

filterByDistance(G, gj)

end if

//identify mapping

if(size(gj)==1) then

type\_gj = administrative-area

else

if(label\_gj!=””) then

set uri\_gj to “/undefined/label\_gj

else

set uri\_gj to “/undefined/ending-part-of uri\_lj

end if

type\_gj = non-administrative-area

end if

end if

end for

//step 2.3. identify mappings for areas which haven’t defined yet

for each area lj in L do

if(size(gj)==1 && type\_gj!=null) then

continue;

end if

//filter by distance

if(size(gj)>1) then

filterByDistance(G, gj)

end if

//identify mapping

if(size(gj)==1) then

type\_gj = administrative-area

else

if(label\_gj!=””) then

set uri\_gj to “/undefined/label\_gj

else

set uri\_gj to “/undefined/ending-part-of uri\_lj

end if

type\_gj = non-administrative-area

end if

end for

end procedure

## 4.4. Identifying co-reference URIs for unit of measure

Input:

* + a URI and its label
  + e.g., <http://dd.eionet.europa.eu/vocabulary/eurostat/unit/1000PERS>, 1000 persons

Output:

* + co-reference URI of this URI
  + e.g., <http://statspace.linkedwidgets.org/codelist/cl_unitMeasure/P1.3>

Method:

* + Step 1. Identify unit (i.e., P1)

Use step of possible names for a unit to identify the co-reference URI for the input URI. For example, to detect unit “People”, our set is as follows:

{"people", “person", "worker", "population", "migration", "migrant", "labor", "births", "adults"}

* + Step 2. Indentify scale/power of unit (i.e., 3)

Detect value in the input label, then identifying scale of this unit. For example, if a label contains “1000” or "1,000" or "1.000" or "thousand", returns value 3.