XML import manual for the European Pollutant Release and Transfer Register

Documentation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **JOB NUMBER:** | | | **DOCUMENT REF:** | | | |
| **Version** | **Purpose of issue** | **Originated** | **Checked** | **Reviewed** | **Authorised** | **Date** |
| 1.0 | Initial Draft | Martin Holleufer-Sørensen |  |  |  | 25.05 2009 |
| 1.1 | Access import modified | Wolfgang Starzer |  |  |  | 03.05.2011 |
| 1.2 | Kml process implemented | Morten Hjelmsmark |  |  |  | 03.05.2011 |
| 1.3 | Detailed description of batch file and XML-upload | Wolfgang Starzer |  |  |  | 28.03.2012 |
| 1.4 | New XML import procedure | Helle Vollmer |  |  |  | 19.10.2010 |
| 1.5 | Scripted import procedure | Morten Hjelmsmark |  |  |  | 23.04.2013 |
| **ATKINS DANMARK** | | | | | | |

Table of Contents

[1 Summary 1](#_Toc355077255)

[2 RUN the xml import 2](#_Toc355077256)

[3 Do an after import check 5](#_Toc355077257)

[4 Upload Log files 6](#_Toc355077258)

[5 Postprocessing imported data 7](#_Toc355077259)

# Summary

This document explains all steps needed to import xml files uploaded by the member states to the E-PRTR system. The xml files containing E-PRTR data are uploaded to the Central Data Register (CDR) at EEA by the member states. The data will first be imported and validated in a temporary database (EPRTRxml) and then the data will be moved to the E-PRTR master database making the data ready for review at the Review site. Atkins will handle the import and validation and will supply the member states with a feedback document in the member states envelope at CDR. When data is imported into EPRTRmaster, some Sql files has to be executed (repairing and exporting), two access files has to be and some RDF files has to be created and a KMZ file generated, before the import process is finalized.

### Prerequisites:

* SQL Server 2008 R2 Native Client
* Microsoft .NET Framework 3.5 SP1
* XML\_Import program
* Python 2.7
* \_mssql (pymssql) from <http://www.lfd.uci.edu/~gohlke/pythonlibs/#pymssql>

Prerequisites for running RDFExport:

* JDK (make sure %JAVA\_HOME% is pointing to it)
* msbuild
* gnuwin from <http://gnuwin32.sourceforge.net/>

### Parameters:

Global parameters:

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Value** |
| {SVN} | Location of import scripts | <https://svn.eionet.europa.eu/repositories/EPRTR/trunk/DataImport> |
| {EIONET} | Location of recently released XML files | <http://cdr.eionet.europa.eu/recent_eprtr> |

Currently active Environments:

|  |  |  |
| --- | --- | --- |
| **Name** | **Build (Atkins)** | **Test/Prod (EEA)** |
| {DB-SERVER} | Sdkcga6332 | Tetrasql |
| {DB-SERVER-PHYSICAL} | Sdkcga6332 | Tetrasql |
| {DB-USER} | gis | Sa |
|  |  |  |

Environment specific parameters

|  |  |  |
| --- | --- | --- |
| **Name** | **Sdkcga6332** | **EEA** |
| {DataImport\_dir} | D:\EPRTRimport\EIONET\DataImport | EEAPC8662 E:\\projects\EPRTR\DataImporte |
| doValidate | True | False |

# the xml import

**How to install:**

Download all files in the EPRTRimport folder from SVN (<https://svn.eionet.europa.eu/repositories/EPRTR/trunk/DataImport/EPRTRimport>)

This new setup is based on Python 2.7 and uses the \_mssql library (<http://www.lfd.uci.edu/~gohlke/pythonlibs/#pymssql>).

The EPRTRimport folder includes two python files; config.py and LoadSubmissions.py, a batch file Validate\_and\_Import\_XML\_File.bat and two folders; SQLscripts and MAPFORCE.

The dataimport is executed by running the LoadSubmissions .py python script.

**What this python script does:**

* Creates the log folder within the folder set as working directory
* Requests the CDR site for envelopes using the xmlrpc\_search\_envelopes\_feedback function. We use the E-PRTR obligation as a parameter ([*http://rod.eionet.europa.eu/obligations/538*](http://rod.eionet.europa.eu/obligations/538)) and Released = 1 as the other. We filter the result by released date ensuring we only address the envelopes released within the period of interest. We then filter the envelopes by the feedback title of the last feedback, ensuring that the title is '*ready for import*'.
* For each envelope:
  + Rename (name includes country code and submission year) and copies the uploaded XML into our working directory
  + Recreates the EPRTRxml database which is a temporary database used for validation
  + Executes the batch file Validate\_and\_Import\_XML\_File. The reason for calling the batch file and not executing the SQL in python is to create the log file. The log contains the print statements from the executed stored procedure, messages that we cannot access from python. The batch executes the MapForce based EPRTR\_Import\_CMD.exe which imports the XML into the EPRTRxml database. Hereafter the data is validated by calling the validate\_xml\_data stored procedure. At last the data is copied into EPRTRmaster.
* As the last thing a status csv file is written summarizing which envelopes that has been imported

**How to configure:**

The parameters are set in the config.py file to make things more accessible.

* Working Directory

This is the folder where the XML files are uploaded to and it's within this folder the log folder is created

#=======================================================================

# IMPORTANT Submission folder path (Working directory)

#=======================================================================

*self*.path = *'D:\EPRTR\DataImport\XML\_Import\Submission\_test'* #Where xml and log files are stored

* Validate:

We normally run the import on a test environment before data is imported into EPRTRmaster at the EEA. Validate creates the log output which we upload as feedback, so validate has to be 'True' until data is accepted. When the import is run at the EEA validate can be set to 'False' to reduce processing time.

#=======================================================================

# IMPORTANT VALIDATE?

# Set to True if the import has to create validations logs

#=======================================================================

*self*.validate = True

* Limited:

When testing the import we do not necessarily want to import all data in that case set Limited to the number of iterations you want to run. If set to *None* all envelopes are processed.

#=======================================================================

# IMPORTANT LIMITED?

# Set to a number if you want this import limited

#=======================================================================

*self*.limited = None

* Date filter:

If you are only interested in processing envelopes which are released within a given period then it is possible to set a minReportingDate and a maxReportingDate.

#=======================================================================

# IMPORTANT - Date filter

# minReportingDate - only Submissions reported after this date (yyyy-mm-dd)

# maxReportingDate - only Submissions reported before this date (yyyy-mm-dd)

# If None filter will be ignored

#=======================================================================

*self*.minReportingDate = *'2013-01-01'*

*self*.maxReportingDate = None

* CDR path and credentials:

If the CDR path is changed or the obligation URL then you need to correct these parameters

You also need to set your cdr username and password (Since we have no GUI we need to set them here)

#=======================================================================

# IMPORTANT - CDR path and credentials

#=======================================================================

*self*.cdrserver = *"http://cdr.eionet.europa.eu/"* #CDR path

*self*.obligation = *"http://rod.eionet.europa.eu/obligations/538"* # E-PRTR obligation url

*self*.released = 1 #We only want released

*self*.cdr\_path = *'http://cdr.eionet.europa.eu/loggedin'* #Used when accessing the restricted cdr envelopes

*self*.cdr\_user = *'mycdrusername'* #CDR eionet username

*self*.cdr\_pass = *'\*\*\*\*\*\*\*\*\*\*\*'* #CDR eionet password

* Database connection parameters:

Used for creating the temporary EPRTRxml database and for accessing the EPRTRmaster database

#=======================================================================

# IMPORTANT SQL Connection params

# Server name and credentials for creating EPRTRxml and accessing the EPRTRmaster db

#=======================================================================

*self*.sp = {}

*self*.sp[*'server'*] = *"myservername"*

*self*.sp[*'user'*] = *"mydbuser"*

*self*.sp[*'passw'*] = *"mydbpassword"*

**How to execute:**

If python 2.7 is installed on the machine and the python.exe path is added to environment path, the LoadSubmissions.py can be executed from a command prompt.

The script is written using Aptana Studio 3 which is based upon Eclipse and imbeds the PyDev tools. In Aptana Studio you should install and open the SVN repository perspective. From here you can download the code as a project, do edits and run the script in debug mode.

**What is copied to EPRTRmaster**

Coping data from database EPRTRxml to EPRTRmaster the following attributes are set automatically during the import-procedure based on the information entered in the batch files:

* POLLUTANTRELEASEANDTRANSFERREPORT.CDRURL 🡪 URL of envelope in CDR
* POLLUTANTRELEASEANDTRANSFERREPORT.CDRUPLOADED 🡪 Date and time when the xml file was uploaded
* POLLUTANTRELEASEANDTRANSFERREPORT.CDRRELEASED 🡪 Date and time when the envelope in CDR was released
* POLLUTANTRELEASEANDTRANSFERREPORT.RESUBMITREASON 🡪 Voluntary description from the envelope in CDR
* FACILITYREPORT.LOV\_RIVERBASINDISTRICTID 🡪 RiverBasinDistrictID of RiverBasinDistrict based on spatial analysis
* FACILITYREPORT.LOV\_NUTSRegionID 🡪 NUTSRegionID of NUTSRegion based on spatial analysis
* FACILITYREPORT.LOV\_STATUS 🡪 Valid, Outside or Missing with regards to the coordinates of the facility

Besides the above attributes a couple of default values that are not required in the xml file are set during xml import:

* POLLUTANTRELEASE.LOV\_TOTALQUANTITYUNITID is set to 2 which refers to KGM
* POLLUTANTRELEASE.LOV\_ACCIDENTALQUANTITYUNITID is set to 2 which refers to KGM POLLUTANTTRANSFER.LOV\_QUANTITYUNITID is set to 2 which refers to KGM
* WASTETRANSFER.LOV\_QUANTITYUNITID is set to 3 which refers to TNE

# Do an post-import check

This part is manual work.

The first part is to open all the log files (placed in the working directory\log folder) and ensure that there are no SQL errors and that the errors found is expected ones.

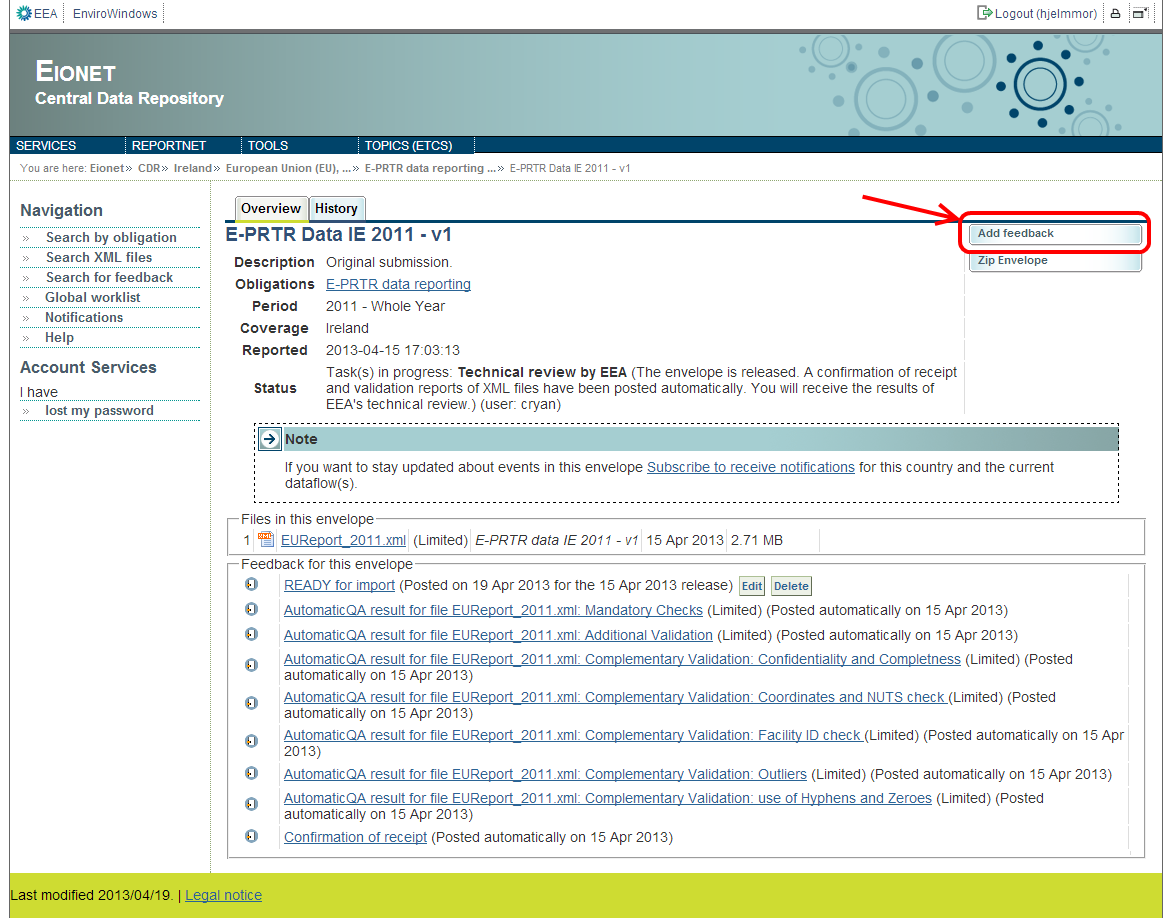
In the log file you'll find:

* Count: Total number of facilities in report: 4894
* Check of National-ID's : The National-ID's that wasn't recognized for a given year will be listed under *Previous nationalids that do not exist for the given previous reporting year.* This is not a big problem and will be fixed by running *RepairFacilWithWrongPrevReportYear.sql.*
* Check for multiple references: Validating if a facility in a previous report is referenced more than once in the new report
* Check for multiple records: Validating if previous nationalid return multiple records
* Geometry check: Validating if reported coordinates are within country polygons (this is where the old NUTS codes creates errors). All facilities which are not within a riverbasindistrict polygon or where the NUTS code are not identified are listed
* Geometry check 2: List of facilities with coordinates set to zero or not reported
* At last the status for copying to EPRTRmaster.

If there is no SQL error messages in the log files, the files are ready for upload.

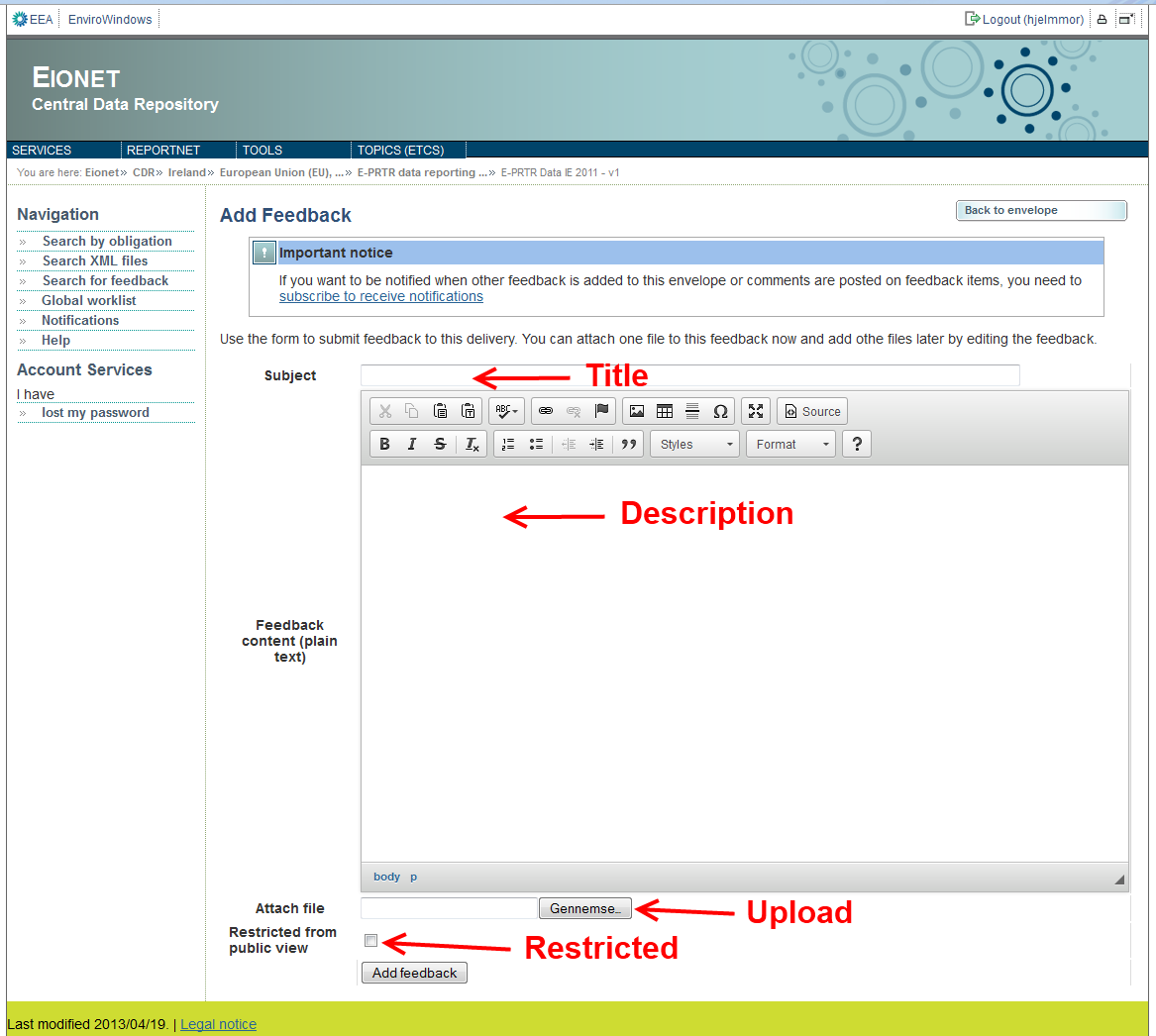
# Upload Log files

This is a manual process. For each submission / envelope you have to add a feedback. To be able to add a feedback you have to log in, and then you'll hopefully see the *Add feedback* button.

Besides uploading the log file, you have to add a specific title, a comment and mark whether the file is restricted or not. If the submission xml is marked as restricted the log file should also be restricted. In the import status csv file you can see if the log should be restricted or by opening the eprtr\_recent website without being logged in.

The feedback title is important because it is used by the scripts in the cdr site to define the status of the envelope. The titles and descriptions you can use are:

* Import went ok:   
  Title: EPRTR import  
  Description:  All data have been successfully imported
* Import failed:

Title: EPRTR import - fail  
Description: Data could not be imported.

# Postprocessing imported data

## Repair broken NationalID links

In the folder {SVN}/DataImport/PostImportRepoairScripts you will find the SQL file ListOfFacilWithWrongPrevReportYear.sql which you open in SQL Server Management Studio and executes connected to the correct SQL server. Export the result to an xml or csv file and upload that to the Submission ticket (taskman). This is a complete list of facilities which will be corrected by running this next sql script. In the same folder you will find the SQL file

RepairFacilWithWrongPrevReportYear.sql and again – open – execute.

## Copy data to Review Site

In order to copy the imported data to the Review Site download the file copyEPRTRmasterViews2EPRTRwebTables.sql from {SVN}/DataImport/EprtrImport/SQLScripts. Please replace every instance of the word EPRTRweb with EPRTRreview in the script. Also make sure that the "**set Published date**" part of the script is commented out and the "**set ForReview date**" part of the script will be executed (last part of file). When everything is ready execute the script in a sql-editor.

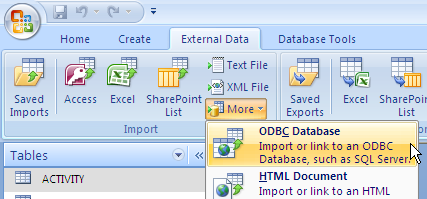
## Copy data to Public Site

**Notice!** This is not done before data has been accepted on the review site!

In order to copy the imported data to the Public Site download the file copyEPRTRmasterViews2EPRTRwebTables.sql from {SVN}/DataImport/EprtrImport/SQLScripts. Please replace every instance of the word EPRTRweb with EPRTRpublic in the script. Also make sure that the "**set ForReview date**" part of the script is commented out and the "**set Published date**" part of the script will be executed (last part of file). When everything is ready execute the script in a sql-editor.

## Export data to Full and Public Access databases

In order to export E-PRTR data to Access databases used by ETC and EEA dataservice download the two Access databases found in {SVN}/DataImport/Export2Access and follow the procedure given below. It must be mentioned, that the imported date will not be correct if this is done before data is copied to the public site.

1. Open up the Access database
2. Open up a ODBC connection to the EPRTRmaster database
3. Select “Import the source data into a new table.....” and click OK
4. Click New (Data Source), choose SQL Server Native Client 10.0 or SQL Server and click Next. The following steps 5-8 only has to be done once. Next time just select the Data Source already set up.
5. Enter a name for the new Data Source (e.g. tetrasql) and click Next and click Finish
6. Enter the appropriate Server (e.g. TETRASQL) and click Next
7. Choose SQL Server authentication and type in login ID (e.g. gis) and password (e.g XXXXXX) and click Next
8. Change the default database to EPRTRmaster and click Next, Finish, Test Data Source, and OK
9. Now select the Data Source that was just created and click OK. Type in password and click OK
10. If the Access database opened is the Full\_Database.mdb select all views prefixed with FULL (e.g. dbo.FULL\_ACTIVITY) (12 views in all) and click OK. If the Access database opened is the Public\_Database.mdb select all views prefixed with PUBLISH (e.g. dbo.PUBLISH\_ACTIVITY).
11. Run Macro FINISH\_DATABASE (calls macro RENAME\_TABLES, Add\_Primary\_Keys and Add\_Foreign\_Keys)
12. Change manually name of table dbo\_FULL\_UPLOADEDREPORTS to UPLOADEDREPORTS.
13. Save the database and exit Access
14. Change the name of the database to Full\_Database\_YYY-MM-DD, respective Public\_Database\_YYYY-MM-DD.
15. Zip the databases and put them on EEA ftp server (<ftp://ftp.eea.europa.eu/Ose3%20GIS/EPRTR/>) username: eeaftp

An outdated version of the documentation of the two databases can be found in {SVN}/documents/TechnicalSpecification/, a newer version is published by EEA on: <http://www.eea.europa.eu/data-and-maps/data/member-states-reporting-art-7-under-the-european-pollutant-release-and-transfer-register-e-prtr-regulation-3>

## Creating KMZ file

This is the procedure to follow when a kml file has to be recreated.

The scripts etc. can be found in {SVN}/DataImport/*/*Eprtr\_kml\_solution, or in the [\\SEASTAR\arcgisserver\SOURCES\Air\EPRTR\_KML](file:///\\SEASTAR\arcgisserver\SOURCES\Air\EPRTR_KML) folder*.*

### If the submission covers a new year

If the data is covering a new submission year (since last kmz was generated), you need to update the base layer file, if it is a resubmission go to the next step.

Locate the Layer file EPRTR\_style\_basic\_reportid.lyr ([\\SEASTAR\.. \EPRTR\_KML](file:///\\SEASTAR\..%20\EPRTR_KML)\Data) and open it in ArcMap.

* Update the data source
* Make sure the joins still works
* In the definition query tab, update query to the correct reportingyear
* In the fields tab reset the list of implemeted fields – remember to **Apply**!

|  |  |
| --- | --- |
| **Layer name** | **Alias** |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.FacilityReportID | FacilityReportID |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.FacilityName | FacilityName |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.ReportingYear | ReportingYear |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.Address | Address |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.City | City |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.PostalCode | PostalCode |
| EPRTRPUBLIC.SDE.FACILITYSEARCH\_MAINACTIVITY\_GEOGRAPHICALCOORDINATE.IASectorCode | IASectorCode |
| EPRTRPublic.dbo.LOV\_COUNTRY.Name | Country |
| EPRTRPublic.dbo.LOV\_ANNEXIACTIVITY.Code | Sector code |
| EPRTRPublic.dbo.LOV\_ANNEXIACTIVITY.Name | Sector |

* When done, save the layer by using ’Save As Layer file..’ and save the lyr file in the data folder

After the layer file is updated, make sure the ..\scripts\ EPER\_fill\_mxd.py points at it.

#Path to the base lyr file

inpPth = solPth + "\Data\EPRTR\_style\_basic\_reportid.lyr"

### Create KMZ

Now it should be ok to create the kmz file. The script can be run by double clicking the **Create\_EPRTR\_KML.bat** file located in the Script folder (on SEASTAR in [\\SEASTAR\arcgisserver\SOURCES\Air\EPRTR\_KML\Scripts](file:///\\SEASTAR\arcgisserver\SOURCES\Air\EPRTR_KML\Scripts)).

**Notice!** Because of a bug in the arcpy tool MapToKml\_conversion you need to open ArcCatalog or ArcMAP, open the tool MapToKML (Toolboxes – System Toolboxes – Conversion tools – to KML – Map to KML), Set *Map Document* to be the ...EPRTR\_KML\Backup\EPRTR\_filled\_(current date).mxd, accept proposed DataFrame, set output File to be ...EPRTR\_KML\Backup\EPRTR\_facilities\_(current date)\_2.kmz and execute. Overwrite the ...EPRTR\_KML\EPRTR\_facilities.kmz with the new one.

Just as with the Access files you need to upload the kmz file to EEA ftp server (<ftp://ftp.eea.europa.eu/Ose3%20GIS/EPRTR/>) username: eeaftp

## Generating RDF files

The RDFExport utility exports the EPRTR database to RDF format. The utility uses GenerateRDF from {SVN}/DataImport/RDFExport/E-PRTR. GenerateRDF requires 3 configuration files: database.properties, rdfexport.properties, and a makefile to run it. The property files and makefile are generated given database parameters by MakeConfiguration solution.

The workflow of RDFExport is following:

1. Build MakeConfiguration using msbuild
2. Launch MakeConfiguration and generate property files
3. Launch generated makefile, which compiles GenerateRDF and runs the export
4. Cleanup and copy rdf files (found in rdf\_export.zip) to the [\\GOOSE\eprtrrdf](file:///\\GOOSE\eprtrrdf) folder.

Prerequisites for running RDFExport:

* Install JDK, make sure %JAVA\_HOME% is pointing to it.
* Install gnuwin from http://gnuwin32.sourceforge.net/.
* Ensure msbuild directory is correct in RDFExport.bat.
* Ensure EPRTR database parameters are correct in RDFExport.bat.
* Use flag -z on MakeConfiguration to gzip each item in the output.

On eeapc8662 I had some issues with java; java version showed fine, but javac was not recognized. After a little fine tuning of PATH and JAVA\_HOME and a restart it now worked.

Executing:

If all prerequisites are set you just need to execute the {SVN}/DataImport/RDFExport/RDFExport.bat file. For getting the batch file to work correctly you need to execute if through the gnuwin32 command prompt. On eeapc8662 there is a shortcut on the desktop.

At last copy the rdf files (found in rdf\_export.zip) to the [\\GOOSE\eprtrrdf](file:///\\GOOSE\eprtrrdf) folder.

## Update the ticket

It is of course mandatory to update the ticket (taskman) regarding the submission with a status of the import.