LEAC Methodology Report 2011

Version 1.0

Type of Document:
Methodology Report

Prepared by:
César Martinez
Maria José Ramos
Raquel Ubach

Date:
10.10.2011

Project Manager:
Oscar Gomez

European Environment Agency

Universidad de Malaga
ETCSIA
PTA - Technological Park of Andalusia
c/ Marie Curie, 22 (Edificio Habitec)
Campanillas
29590 - Malaga
Spain

Telephone: +34 952 02 05 48
Fax: +34 952 02 05 59

Contact: etc-sia@uma.es

TABLE OF CONTENTS

Annexes

1 OVERVIEW

LEAC methodology has been constantly updated since it was first defined in 2002. The aim of this document is to compile the various technical procedures followed during the implementation of the last updated LEAC methodology in one single document. With the object to provide a complete and most detailed consulting document for further uses, reviews or improvements.

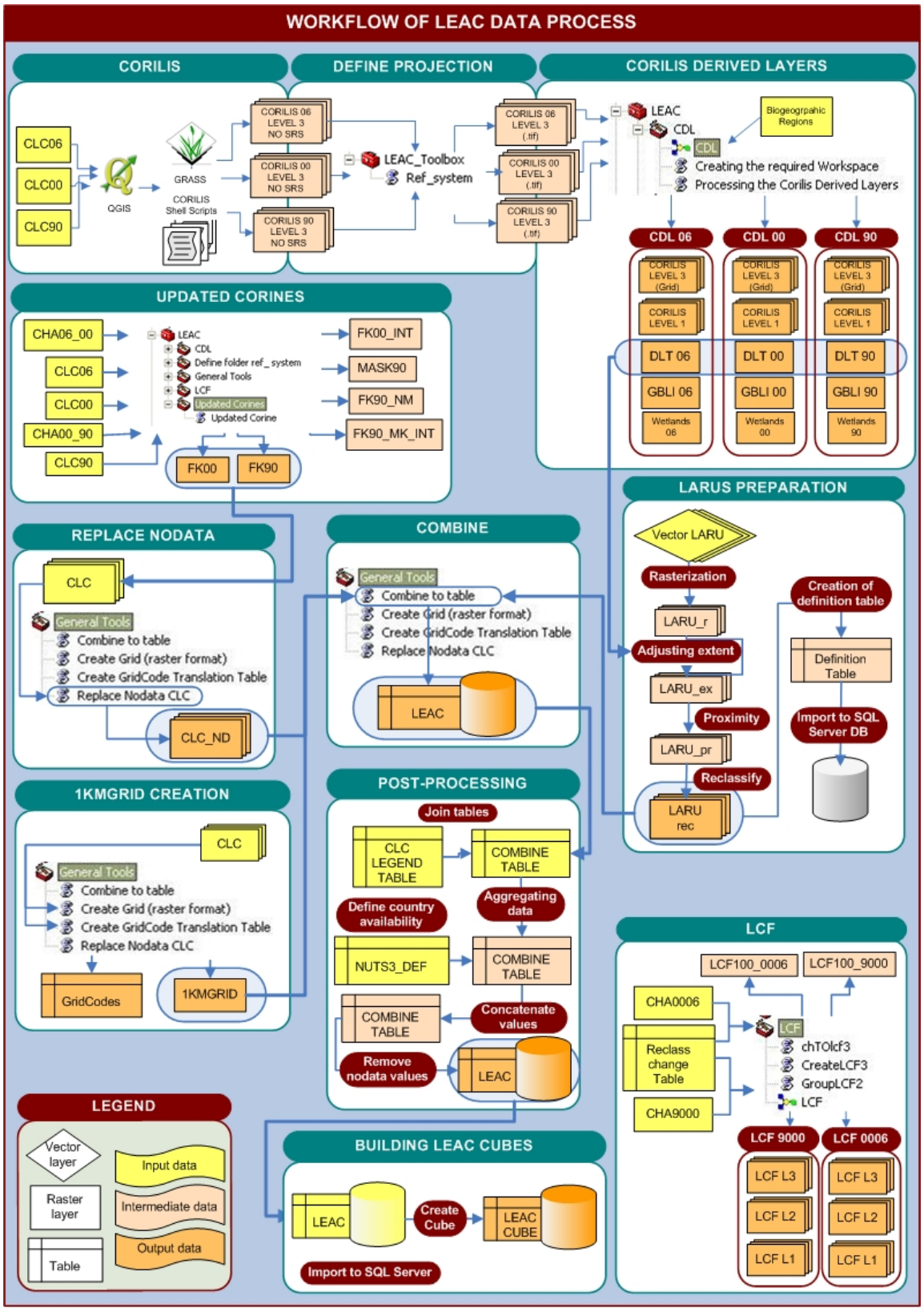
1.1 IN THIS DOCUMENT

The chapters in this document are ordered chronologically following the workflow of the methodology:

- **Processing geographic information (Chapter 2)** integrates all the procedures related with spatial analysis and geographic information management.
- **Building the database (Chapter 3)** gives an overview of the data model used to build the LEAC database and related applications.
- **Annexes (Annex I – Annex VII)** added information for some procedures, description of the tools used and the scripts.

2 PROCESSING GEOGRAPHIC INFORMATION

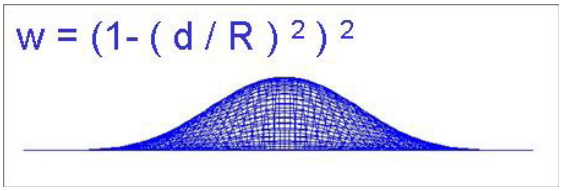
This chapter integrates all the procedures related with spatial analysis and geographic information management.



2.1 CORILIS

CORILIS, from CORIne and LISsage (smoothing in French), is a methodology developed jointly by the French Environment Institute (IFEN), the Hypercarte Research Group and the French National Institute for Statistics and Economic Studies (INSEE) that provides technical specifications for the smoothing of CORINE Land Cover Data.

The purpose of CORILIS is to calculate "intensities" or "potentials" of a given theme in each point of a territory. A Gaussian type statistical function (called BiWeight) is used to weight this information according to the distance from the considered point in kilometers[1].



Where: w is for weight, d for distance in Km and R for radius in kilometers.

The neighbourhood matrix is calculated according to the radius following the next function: N = (R * 2) – 1. Where: N sets the dimensions of the matrix N x N. For example, for a radius of 5km ((5*2)-1 = 9) a 9x9 neighbourhood matrix is created.

That means that the value of a cell is calculated by adding the values of the cells inside the matrix NxN that has been previously weighted. Weighting factors decrease to 0 as the distance increases.

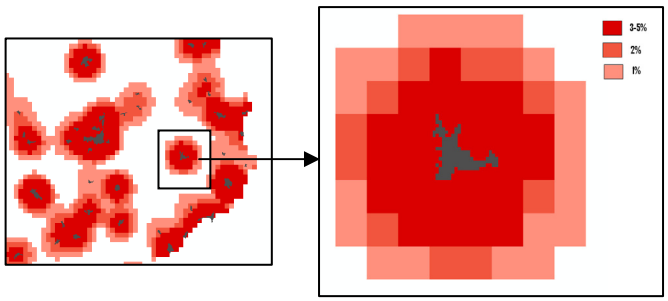
For a radius of 5km (9x9 Matrix), the weighted values are as follows:

0	0	0,04	0,1024	0,1296	0,1024	0,04	0	0
0	0,0784	0,2304	0,36	0,4096	0,36	0,2304	0,0784	0
0,04	0,2304	0,4624	0,64	0,7056	0,64	0,4624	0,2304	0,04
0,1024	0,36	0,64	0,8464	0,9216	0,8464	0,64	0,36	0,1024
0,1296	0,4096	0,7056	0,9216	1	0,9216	0,7056	0,4096	0,1296
0,1024	0,36	0,64	0,8464	0,9216	0,8464	0,64	0,36	0,1024
0,04	0,2304	0,4624	0,64	0,7056	0,64	0,4624	0,2304	0,04
0	0,0784	0,2304	0,36	0,4096	0,36	0,2304	0,0784	0
0	0	0,04	0,1024	0,1296	0,1024	0,04	0	0

CORILIS results into probability surfaces (varying from 0 to 100) for the presence of a certain CLC class within a smoothing radius (5, 10 or 20 km).

The next figure shows in grey colour an extraction of Continuous Urban Fabric areas (Class 1) from the Corine Land Cover 2000 (resolution 100 ha), and in red colours the resulted Corilis layer applying a radius of 5km and with a final resolution of 1km.

[1] To have more information about Corilis methodology visit: http://www.eea.europa.eu/data-and-maps/data/corilis-2000-2/dataservice-sharedfiles-downloads-rad93e83-english_v2-download-corilis_methodology.pdf



The results show the "intensity" or "potential" of continuous urban fabric areas in relation to the total "intensity" or "potential" of the area. (100* (CI intensity / total intensity))

The total "intensity" or "potential" is calculated by smoothing together all the CLC classes presents in an area.

A schematic description of the Corilis Methodology and the weighted process can be found in the **AnnexI- Corilis Methodology and Steps.**

2.1.1 Corilis Process

Corilis data is processed on a GNU/Linux system using Quantum GIS and Grass software. Although almost all the processes have been automated to be run in Grass, Quantum GIS is used as intermediate software to create the structure of folders (location/mapsets) inside Grass and to import the input data (Corine Land Cover).

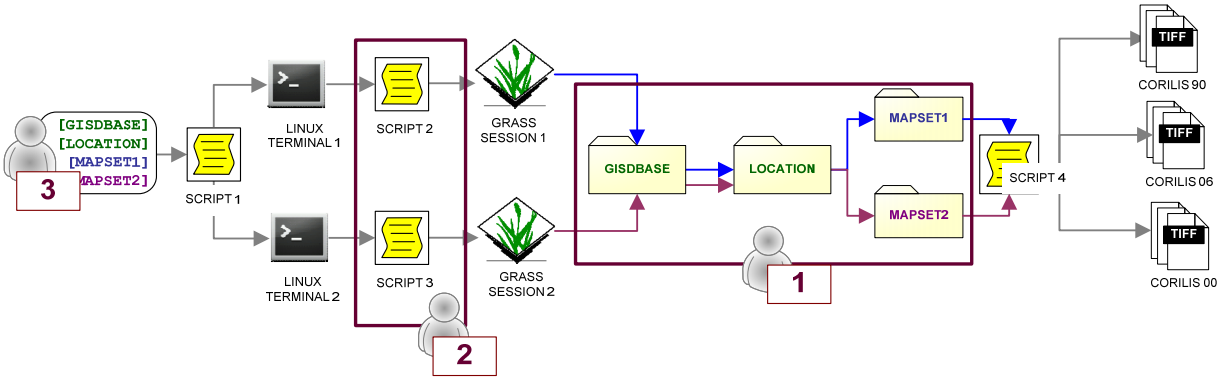
A plot of 4 scripts has been created that enables to process corilis in a continuous way without stopping for one year to another, and furthermore, to process corilis for two years at the same time, for example 2000 and 2006 by opening two grass sessions.

Before launching the plot of 4 scripts, the user has to follow the next steps:

1) The user has to create, using QGIS, a structure of two mapsets and one location in Grass and to import the input data (clc90, clc00 or clc06) into each Mapset.
2) Inside each Mapset will take place different processes. The script2 will launch the processes for the mapset1 and the script3 for the mapset2. The user has to open each script and add at the end of the script a new line of code specifying the corilis processes to be run by the next parameters: [*input data], [start class], [end class], [radius], region ([top], [bottom], [left], [right]), [resolution], [no data value] (input data),[the ouput folder].*

Once the grass structure has been created and the script2 and script3 have been modified, the user can launch the script1.

3) The user runs the script1 by opening the linux terminal and writing the path to the script and the next parameters: [gisdbase], [location], [mapset1] and [mapset2].



The general process can be summarized in the next points:

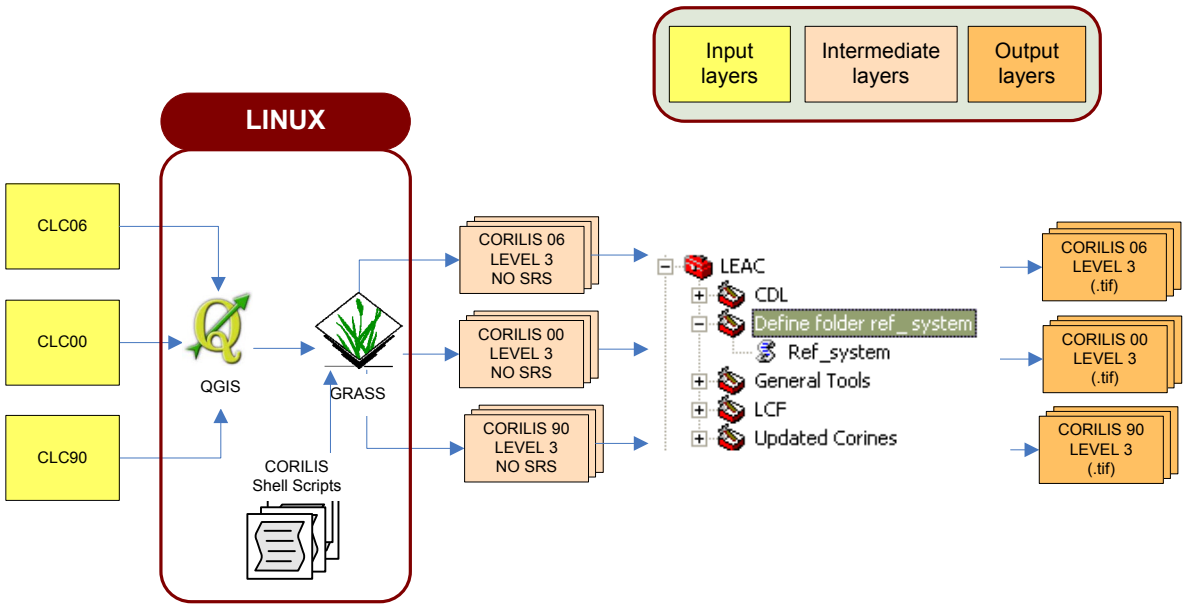
- The script1 opens two Unix terminals and runs one script in each terminal (script2 and script3).
- The script2 and script3 opens each one a new grass session in a specific mapset.
- The script2 runs the corilis processes in mapset1 following the user conditions (input data, start class, end class,…..)
- The script3 runs the corilis processes in mapset2 following the user conditions (input data, start class, end class,…)

The approximate time processing for the calculation of Corilis for the 44 classes for three Corine Land Cover data (1990, 2000 and 2006) is one day (24 h).

The detailed steps to calculate Corilis can be found in **AnnexI- Corilis Methodology and Steps**.

The 44 Corilis level 3 layers (.tiff) resulted doesn't have a Reference System Definition compatible with ESRI, for that reason after the process in Ubuntu is finished, the data is submitted to a process of SRS definition using a toolbox created in ArcGIS (9.3) for that purpose (See the SRS analysis and the tool in **Annex II – Define projection tool**),.

2.1.2 Corilis workflow



2.2 CORILIS DERIVED LAYERS

CORILIS methodology defines a procedure for smoothing Corine Land Cover data using a variant of the Gaussian function called BiWeight. For each CLC class a probability surface is obtained, which shows the percentage of that class within a specific radius. That is, each grid corresponds to a CORINE Land Cover Level 3 class smoothed. These grids can be summed up or thematically grouped. In this way, it is produced Corilis Level 1, as well as other customized aggregations such as Green Background Landscape Index (BLI), Wetlands and Dominant Land Types (DLT).

In summary, the resulting Corilis derived layers are:

- Corilis Level 1
- Green Background Landscape Index (BLI)
- Wetlands
- Dominant Land Types (DLT).

2.2.1 Corilis Level 1

Corilis Level 1 is a group of 10 raster datasets coming from the **aggregation of several classes of Corilis Level 3**, the aggregation of classes is as follows:

- c1 Artificial surfaces = ∑ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and 11.
- c2a Arable and permanent crops = ∑ 12, 13, 14, 15, 16, 17 and 19
- c2b1 Pastures = ∑ 18
- c2b2 Mosaic farmland = ∑ 20, 21 and 22
- c3a1 Standing forests = ∑ 23, 24 and 25
- c3a2 Transitional woodland and shrub =∑ 29
- c3b Natural grassland, heathland, sclerophylous vegetation = ∑ 26, 27 and 28
- c3c Open space with little or no vegetation = ∑ 30, 31, 32, 33 and 34
- c4 Wetlands = ∑ 35, 36, 37, 38 and 39
- c5 Water bodies = ∑ 40, 41, 42 and 43

2.2.2 Green background landscape index (GBLI)

Green Background Landscape Index is a conceptual **grouping of CORILIS Level 1** layers to approach as European green areas, which expresses the vegetation potential of the territory according to land use intensity. It is based on the spatial distribution of pasture, agriculture mosaics, forests and other semi-natural or natural land. This layer groups the "green" classes in order to provide a "green background" for overlaying land cover changes. The CORILIS Level 1 classes grouped in this layer are:

- C2B Pastures & mixed farmland (c2b1,c2b2)
- C3A Forests and transitional woodland shrub (c3a1,c3a2)
- C3B Natural grassland, heathland, sclerophylous vegetation
- C3C Open space with little or no vegetation
- C4 Wetlands
- C5 Water bodies

2.2.3 Wetlands

The relevance of wetlands on the global biodiversity, the environmental pressures from land use and pollution that these ecosystems are receiving, and their characteristic dynamism in terms of processes being on the interface of different systems (terrestrial/maritime); justify the need to provide a more holistic approach for a better understanding on the impacts and their evolution. For all these, it has been integrated in the LEAC assessments in order to facilitate analysis and give support to policy development and management.

Wetlands layer **groups some Corilis Level 3** classes based on Ramsar definition of wetlands, which include:

- ∑ (rice fields (14), wetlands (35,36,37,38,39), coastal lagoons (42) and estuaries (43)).

2.2.4 Dominant Land Cover Types (DLT)

Dominant Land Cover Types are defined by classification of **the resulting values of CORILIS Level 3** into dominant classes in a given Biogeographic Region[2].

The criterion Vn > mean + standard deviation is used to assign dominant character to smoothed features. Vn is the smoothed value of class n in a given cell of the map.[3].

The final data published on the EEA dataservice is a raster file in tiff format with the following legend:

0. No Data.
1. Artificial dominance (Urban dense areas)
2. Dispersed urban areas
3. Broad pattern intensive agriculture
4. Rural mosaic and pasture landscape
5. Forest landscape
6. Open semi-natural or natural landscape
7. Composite landscape

2.2.5 CDL Process

The process has been automated throw the creation of the tool CDL included in the LEAC toolbox. The process can be divided in two different parts:

- Preparing workspace with the required input data (Corilis level 3) and folder structure.
- Processing data.

The input data for the Corilis Derived Layers is the Corilis Level 3 layers.

[2] The bio-geographic regions dataset contains the official delineations used in the Habitats Directive (92/43/EEC) and for the EMERALD Network set up under the Convention on the Conservation of European Wildlife and Natural Habitats (Bern Convention)(http://www.eea.europa.eu/data-and-maps/data/biogeographical-regions-europe-2008)

[3] For more information visit: http://www.eea.europa.eu/data-and-maps/data/dominant-land-cover-types-2000-1

The output of the Corilis process (see **2.2.1 Corilis Process**) is 44 new raster datasets (in the case that the sea class, CLC 44, was included), in .tiff format and with a specific nomenclature (*nameinput*_R*radius*_C*class.*tif, for example lceugr00_R5_C1.tif).

- Preparing workspace with the required input data (Corilis level 3) and folder structure.

However, the Corilis Derived Layers process needs a folder called level3 with the Corilis Level 3 layers in grid format and with a specific nomenclature (*rradius*sm*class,* for example the name for the Corilis level 3 layer for the class 1 with a radius of 5 km has to be r5sm1).

For that reason, the first part of the script:

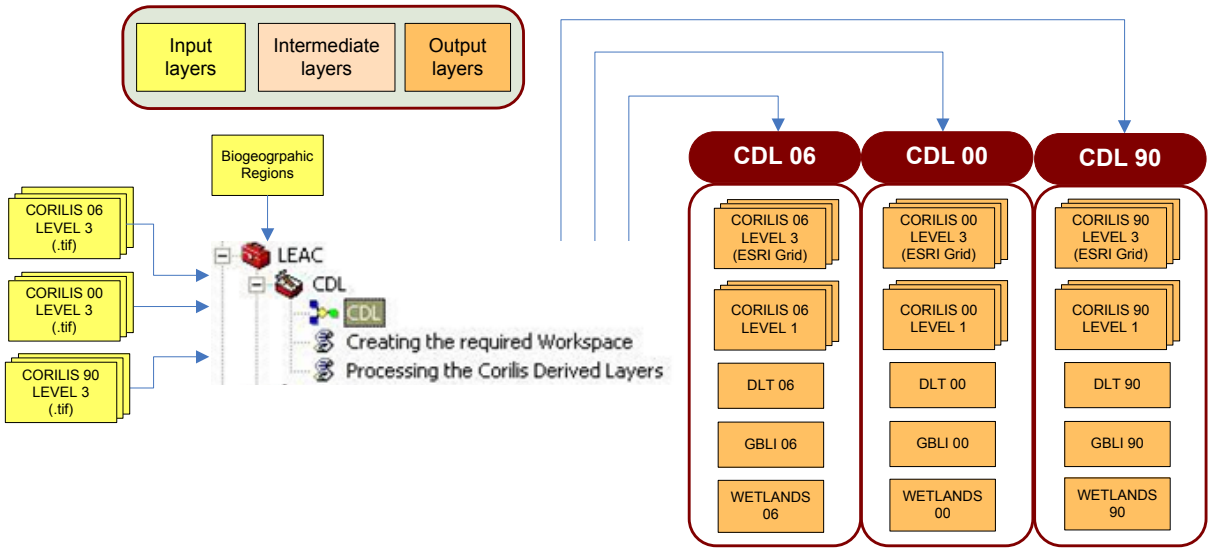
- Creates a folder called level3
- Imports the Corilis Level 3 layers in grid format
- Renames each Corilis Level 3 layer.
- Creates the folder structure where the output layers will be stored

- Processing data.

The second part of the process is based in 4 aml scripts, which generate the different aggregations for Corilis level 1, GBLI, Wetlands and DLT. These scripts are run sequentially in the following order:

1. The creation of the Corilis Level 1 (g_level1.aml)
2. The creation of the Green background landscape index (g_gbli.aml)
3. The creation of the Wetlands layer (g_wetlands.aml)
4. The creation of the Dominant Land Cover Types layer (CALCDLT.aml)

2.2.6 CDL tool workflow



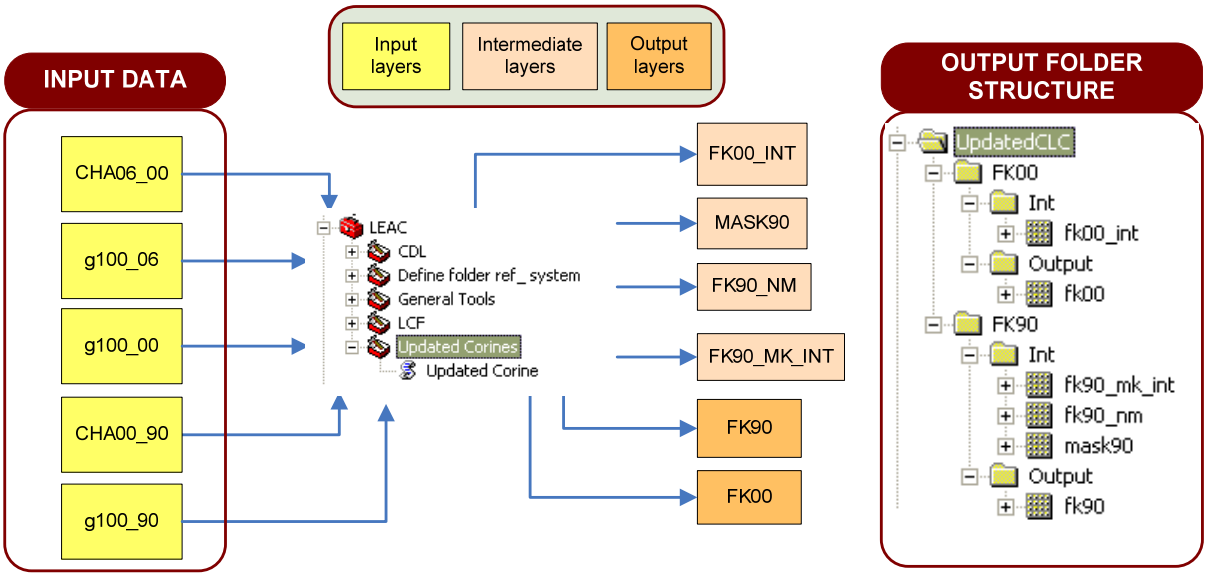
For further details on CDL tools, scripts and processes see **Annex III - CDL steps.**

2.3 CREATING UPDATED CORINE LAND COVER LAYERS

In the delivery of Corine Land Cover data, the most updated information appears in the layers of changes. In order to work with the most updated data, it is needed to build up an updated Corine Land Cover based on changes layers where information is more accurate.

A tool included in the LEAC toolbox has been developed to automate the process of Updated CLC creation for year 00 and 90, as well as the required folder structure where the intermediate and output layers will be stored.

2.3.1 Updated Corine Land Cover Layers workflow



For further details on Updated Corine tool, scripts and processes see **Annex IV – Updated Corine steps.**

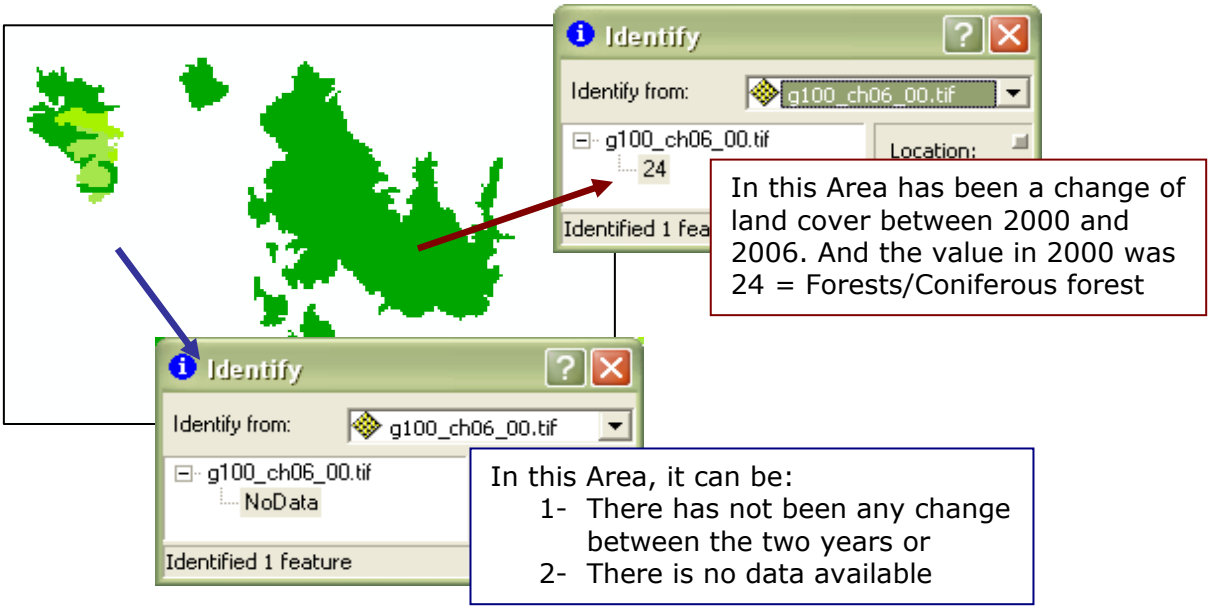
2.3.2 Updated CLC 00

To build the updated CLC00, the main layer is the **clc changes 00-06 with data of year 00 (cha06_00).** This raster dataset shows where has been a change of land cover between 2000 and 2006 and which was the land cover value in 2000.

In the changes layers, there is only data where a change of land cover has happened between the two years and the rest of cells are Null. The No Data value can be due to two cases:

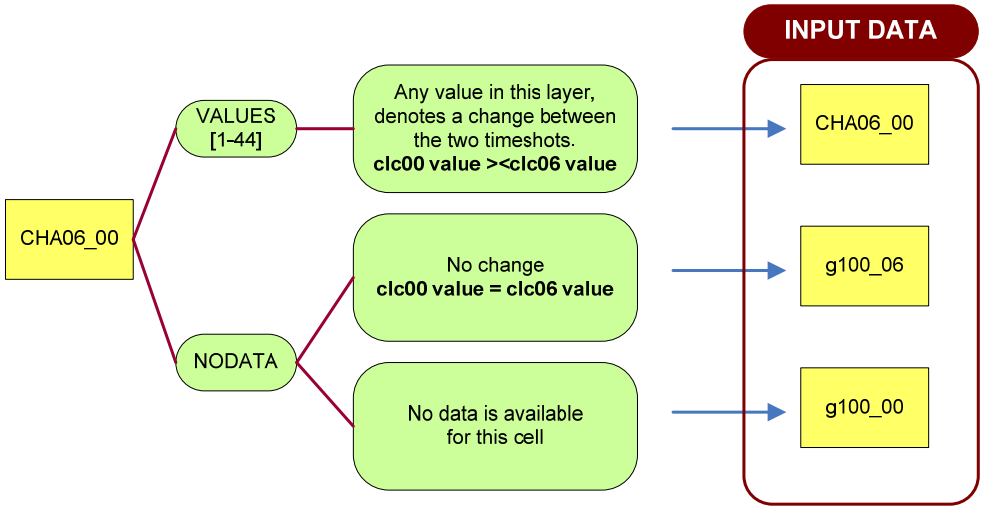
- There has not been any change between the two years, so the land cover for year 00 is the same that for the year 06
- There is no data available for a certain country.

The next figure shows an extraction of the clc changes 00-06 with data of year 00.



If **no changes have occurred,** that means that data from 00 and 06 has the same values, then **the new value is taken from last year available**, in this case CLC 2006 (g100_06), so more recent data is considered.

If **there are changes**, so the changes layer has a value, **the new value is taken from the changes layer 2000-2006 with values of 2000** because the CLC changes layers are the most updated data. The first process obtains an intermediate layer (Fk00_int).

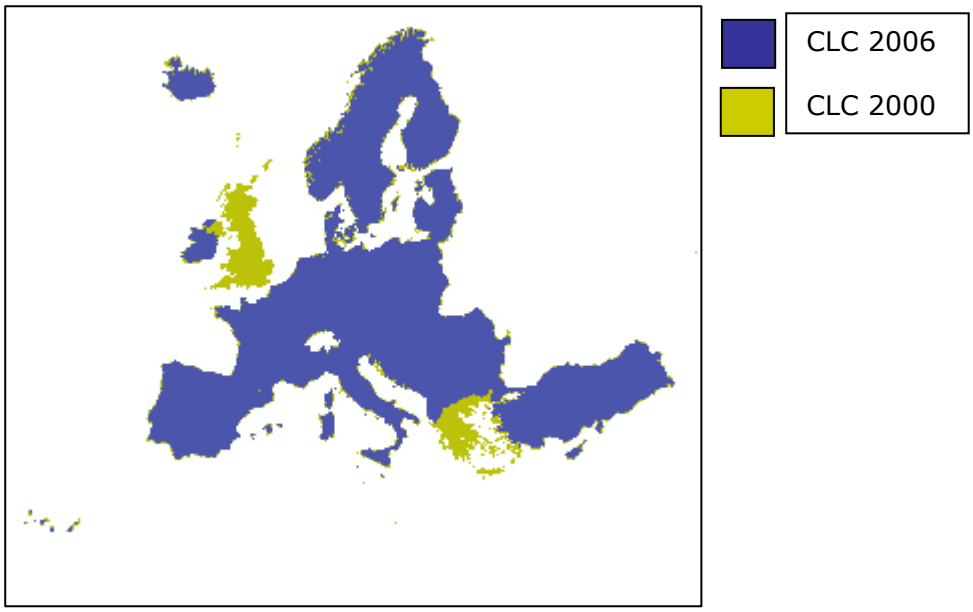


Map Algebra:

```
Fk00_int = con(IsNull([ch06_00]), [g100_06], [ch06_00])
```

In the previous Map Algebra expression is important to verify that the no data values in the CLC changes layers is null instead of 49 or 255.

The extension of territory covered by CLC can vary between years, for example in the CLC version 13, United Kingdom (UK) and Greece (GR) was covered by CLC2000 but not by CLC2006.



From the last Map Algebra expression, the zones not covered by CLC06 will set to no data even if these zones are covered by CLC2000. For that reason, there is a second step in the Updated CLC2000 that takes the value for these zones from the CLC2000 original.

```
Fk00 = con([Fk00_int] <>49, [Fk00_int], [lceugr100_00.tif])
```

2.3.3 Updated CLC 90

The Updated Corine for the year 1990 follows the same conceptual approach, with some slight adaptations. A mask for clc90 needs to be created, at least from the last version data (CLC_2006_v13), to have the extent and the information from 90 data layer.

Map Algebra:

```
mask90 = setnull([g100_90] == 49, 1)
```

Again we use the layer of changes, now changes between clc 90-00 with data of year 90 (cha00_90). **Where there are changes**, we take values corresponding to **clc changes layer 90_00 with values of 90 (that is cha00_90)**. Where no changes have occurred, data is taken from the fake corine (fk_clc00)

Map Algebra:

```
fk90_nm = con(IsNull([g100_ch00_90]), [fk00], [g100_ch00_90])
```

The final step is to multiply the output raster by the mask, in this way null value is given to land with no data for that year.

Map Algebra:

```
fk90_mk_int = fk90_nm * [mask90]
```

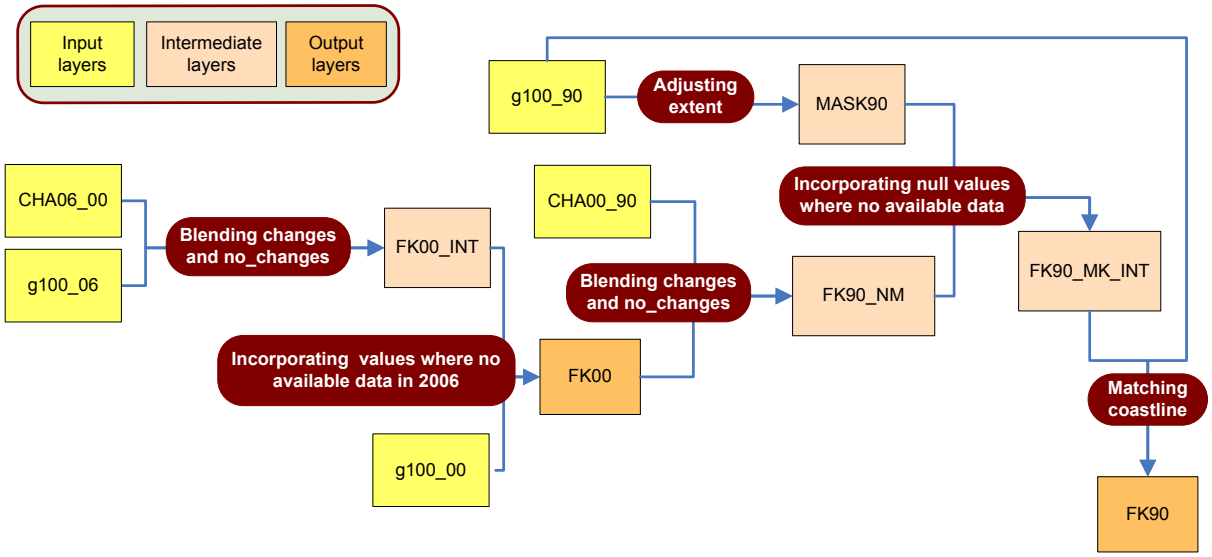
In some places there is a difference between the coast line values in 90 and 00, that is in 00 a pixel is land and in 90 it is sea. This discrepancy in coastline is reflected in some areas in NO, SE, FI, AL and TR.

To match coastline to the extent of clc90, another step is needed:

- When clc90 is sea then sea value is given in fake clc90
- all other values then take the values from fk90_mk_int

```
fk90 = con([g100_90] == 44, 44, [fk90_mk_int])
```

2.3.4 Update Corine Land Cover conceptual workflow



2.4 LAND COVER FLOWS

Land Cover Accounts summarize and interpret the possible one-to-one changes between the 44 CORINE land cover classes. The changes are grouped in land cover flows and are classified according to major land use processes.

Corine land cover changes are classified into land cover flows based on Land cover accounts (LEAC) methodology, and generalised using the 1 Km reference grid size. Land cover flows are grouped into three hierarchical levels (level1, level2 and level3)

The next table presents the nine LCF at level 1:

Code	Description
LCF1	**Urban land management**: Internal transformation of urban areas
LCF2	**Urban residential sprawl**: Land uptake by residential buildings altogether with associated services and urban infrastructure (classified in CLC111 and 112) from non-urban land (extension over sea may happen)
LCF3	**Sprawl of economic sites and infrastructures**: Land uptake by new economic sites and infrastructures (including sport and leisure facilities) from non-urban land (extension over sea may happen)
LCF4	**Agriculture internal conversions:** Conversion between farming types. Rotation between annual crops is not monitored by CLC
LCF5	**Conversion from forested and natural land to agriculture**: Extension of agriculture land use
LCF6	**Withdrawal of farming:** Farmland abandonment and other conversions from agriculture activity in favour of forests or natural land
LCF7	**Forests creation and management**: Creation of forests and management of the forest territory by felling and replanting. Due to the CLC cycle of 10 years, only one part of the shrubs are tall enough to be identified as trees. In order to taking stock of all recent plantations, conversions of semi-natural land to CLC324 are conventionally recorded as afforestation (although some natural colonisation may take place). In the case of conversion from farmland, see LCF61
LCF8	**Water bodies creation and management**: Creation of dams and reservoirs and possible consequences of the management of the water resource on the water surface area
LCF9	**Changes of land cover due to natural and multiple causes**: Changes in land cover resulting from natural phenomena with or without any human influence

The process has been automated in a tool included in the LEAC toolbox. However, the process comprises three different steps that have been grouped in three different scripts.

For further details on CDL tools, scripts and processes see **Annex V - LCF.**
Input data:

- CLC Changes 00-06 : To calculate the LCF for the period 2000-2006
- CLC Changes 90-00 : To calculate the LCF for the period 1990-2000
- Reclassification Table: This table can be created by adding a new integer field of three digits into the change legend table (see next point 2.4.3.1 Reclass)

2.4.1 Process

The process can be divided in three parts:

1- Reclassification of the CLC Changes XXYY layer by replacing its values by the Land Cover Flow level 3 in numeric format.

2- Calculation of the Land Cover Flows at level 3

3- Calculation of the Land Cover Flows at level 2 and level 1

2.4.1.1 Reclass

The change legend Table that is delivered with the Change Layers data will be used as reclassification table, but previously it is needed to add a new field.

This table ha a total of 1937 elements that represent all the possible combinations (change and not change) between two years (44 classes * 44 classes = 1936 combinations + 1 No data combination = 1937 combinations).

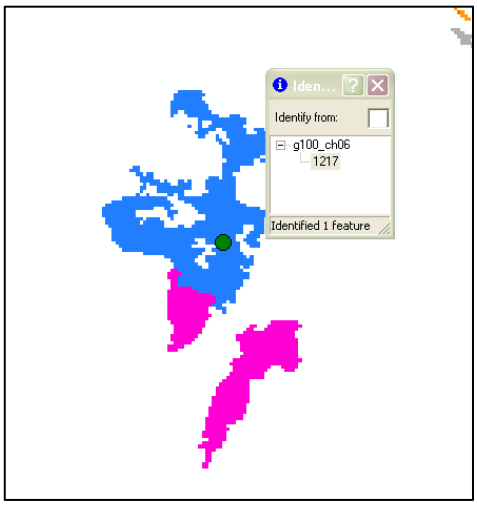
This table sets the relation between each combination of change and the Land Cover Flow description.

The next table is an extraction of the change legend Table:

GridCode	Change	LCF1	LCF2	LCF3	LCF1-DES	LCF2-DES	LCF3-DES
1	111-111	NC	NC	NC	NC No Change	NC No Change	NC No Change
45	112-111	Lcf1	Lcf11	Lcf111	Lcf1 Urban land management	Lcf11 Urban development/ infilling	Lcf111 Urban development/ infilling

- **GridCode** is the value of the Layer of Changes XX-YY raster dataset (ex: g100_ch06). This field will be the "*From value field*" and "*To value field*" in the reclassification process.
- Change is the combination of CLC classes between the two years
- LCF1, LCF2, LCF3 are the codes of the Land Cover Flows
- LCF1-DES, LCF2-DES, LCF3-DES are the description of the Land Cover Flows

The layer of Changes XX-YY gives information about changes occurred in the period (XX-YY) . The next figure shows an example of the Layer of Changes 00_06.



The value of the pixel is 1217 (going to the change legend table):

- GridCode: 1217
- Change: 323-423
- LCF1-DES: lcf9 Changes of Land Cover due to natural and multiple causes.
- LCF2-DES: lcf93 Coastal erosion
- LCF3-DES: lcf930 Coastal erosion

In order to process the Land Cover Flows, the layer of Changes XX-YY has to be reclassified according to the Land Cover Flow code at level 3.

The field Land Cover Flow code at level 3 is a string field (ex:lcf111) so it is not possible to make a direct reclassification. Before that, it is necessary to create a **new field integer** with the numeric part of the string (lcf111 – 111). It can happen that some codes do not have three numeric digits (ex: lcf12), that means that the land cover flow at level 3 is lcf120.

The next table shows an extraction of the change legend Table with the New field lcf3 created:

GridCode	Change	LCF1	LCF2	LCF3	New lcf3	LCF1-DES	LCF2-DES	LCF3-DES
1	111-111	NC	NC	NC	0	NC No Change	NC No Change	NC No Change
33	111-334	Lcf9	Lcf92	Lcf92	920	Lcf9 Changes of Land Cover due to natural and multiple causes	Lcf92 Forests and shrubs fires	Lcf92 Forests and shrubs fires
45	112-111	Lcf1	Lcf11	Lcf111	111	Lcf1 Urban land management	Lcf11 Urban development/ infilling	Lcf111 Urban development/ infilling

Once the new field lcf3 is created then it is possible to make the reclassification that is base in the next parameters.

- *Input*: g100_ch06
- "*From value*": GridCode
- "*To value*": GridCode
- "*New value*": New lcf3
- *Output raster*: LCF100

The output raster, LCF100 has a resolution of 100m; however the output layers (lcf at level 3, level 2 and level 1) will be aggregated to 1km.

2.4.1.2 Creating Land Cover Flows at level 3

The Land Cover Flows at level 3 are 19 raster datasets at 1km resolution presenting the lcf at the minimum level.

The AML script:"CreateLCF1.aml" is used to create the Land Cover Flows at level 3.

The input data for this process is the LCF100 raster at 100m, obtained in the previous step. The script aggregates data to 1000m and creates a file for each LCF. The script command should be as the example below for "grid 110":

AGGREGATE(<grid>, <cell_factor>, {aggregation_type}, {EXPAND | TRUNCATE}, {DATA | NODATA})

Aggregate(lcf3 == 110, 10, SUM, TRUNCATE, DATA)

The <cell_factor> should be 10 which is the factor used to multiply the cell size (10x10) of LCF at level 3 input grids to obtain the desired resolution for the output grid (1000x1000).

The aggregation type should be the SUM of the input cells values that the coarser output cell will encompass.

The TRUNCATE option will trim the remaining cells on the bottom and/or right boundaries of the input grid making the number of rows and/or columns in the input grid a multiple of the <cell_factor>.

DATA should be the selected option since we want to specify that if a NODATA value exists for any of the cells that fall within the spatial extent of a larger cell on the output grid, in this way the NODATA value will be ignored when determining the value for that output cell location.

2.4.1.3 Grouping Land Cover Flows (Creation of LCF level 1 and 2)

Once the Land Cover Flows at level 3 are ready then we group them by summing the different grids to create LCF at levels 1 and 2.The AML script is called: "GROUPLCF2.aml"

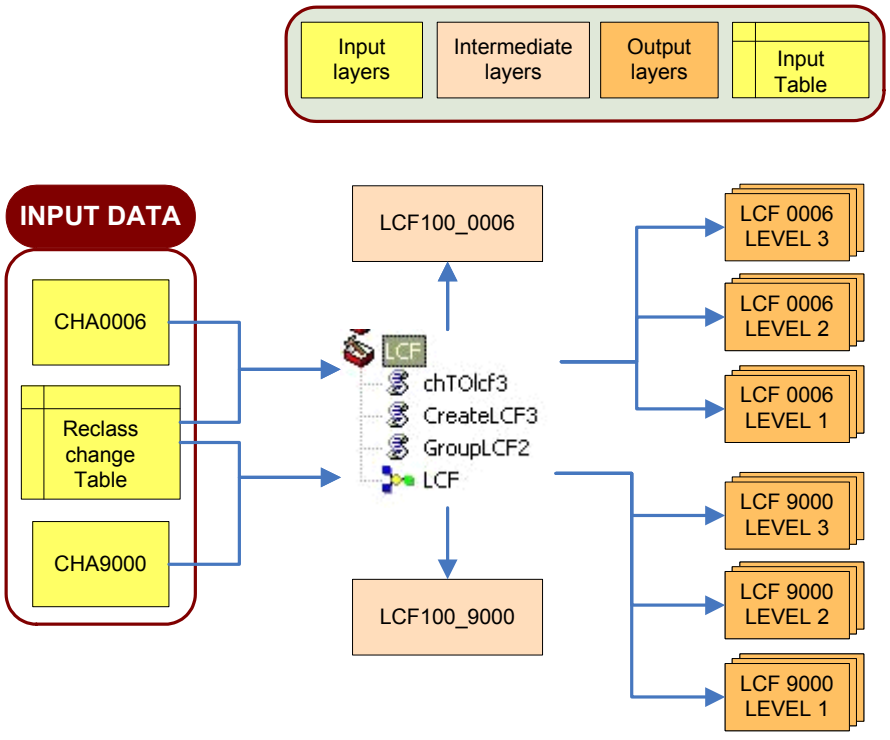
For further details, see the complete script code in **Annex V – LAND COVER FLOWS**.

2.4.2 Output data

Output data is three datasets containing:

- LCF Level 3
- LCF Level 2
- LCF Level 1

2.4.3 LCF tool workflow



2.5 CREATING THE 1 KM REFERENCE GRID

Equal area grids are suitable for generalising data, statistical mapping and analytical work where an equal area of cells is the basis of these analyses allowing comparison between grids. A geographical grid system is a harmonised multi-resolution grid with a common point of origin, standardised location and size of grid cells (as defined by the INSPIRE Directive[4] 2007/2/EC).

The EEA Reference grid[5] is based on the recommendations at the 1st European Workshop on Reference Grids[6] in 2003 and later INSPIRE geographical grid systems[7]. For Europe and each country three vector polygon grid shape files, 1, 10 and 100 km, are available. For further details on EEA Reference grid, check the Guide for EEA map production[8].

The LEAC database is structured in the basis of a 1Km Reference grid in order to allow the characterization of land patterns and processes, the comparison between different portions of territory, and the later analyses of land and land changes of use.

2.5.1 Process

Anticipating to a later step to combine different analytical information, now it is necessary to create a 1Km Reference grid in a raster format for the whole coverage of available data. To do so, the process has been automated in a general toolset included in the LEAC toolbox. For the grid creation there is a tool, named Create Grid (raster format), which creates a European reference GRID1KM in raster format. The cells are encoded using the new LEAC numeric cell encoding. This tool is only available for ArcGIS 10.0, as NumPyArrayToRaster is not available on ArcGIS 9.3.

Note: several temporal rasters are created in the output directory, and they are not deleted afterwards (to help monitoring the tool behaviour). Ensure you delete them after successfully executing the tool.

A translation table to reference string codes ("1KME2342N3324") can be easily created using the "Create GridCode translation table" tool. This tool creates a table containing the equivalence between the reference code (ex: "1kmE3264N4702") and the LEAC numeric code (32644702). The table will be named "gridCodes" and it will contain the fields "NUMCODE" and "REFCODE".

For further details on Updated Corine tool, scripts and processes see **Annex VI – General Tools toolset.**

[4] Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE).

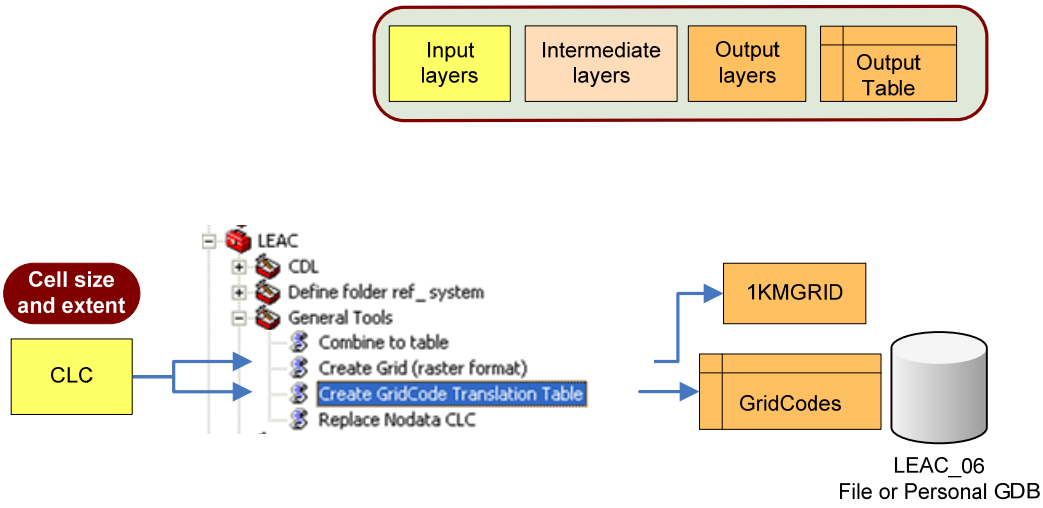
[5] The EEA Reference grid is available via this link: http://www.eea.europa.eu/data-and-maps/data/eea-reference-grids-1

[6] http://eusoils.jrc.ec.europa.eu/projects/alpsis/Docs/ref_grid_sh_proc_draft.pdf

[7] D2.8.I.2 INSPIRE Specification on *Geographical Grid Systems* – Guidelines (available at http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/INSPIRE_Specification_GGS_v3.0.1.pdf)

[8] See sections 2.10. What is the EEA Reference grid?, 2.12. How to select or define a grid?; and 2.13. How to create an ETRS-LAEA grid using ESRI ArcGIS? available at: http://www.eionet.europa.eu/gis/docs/Data_requirement_for_map_production_GISguide_v4.pdf

2.5.2 GridCode creation workflow



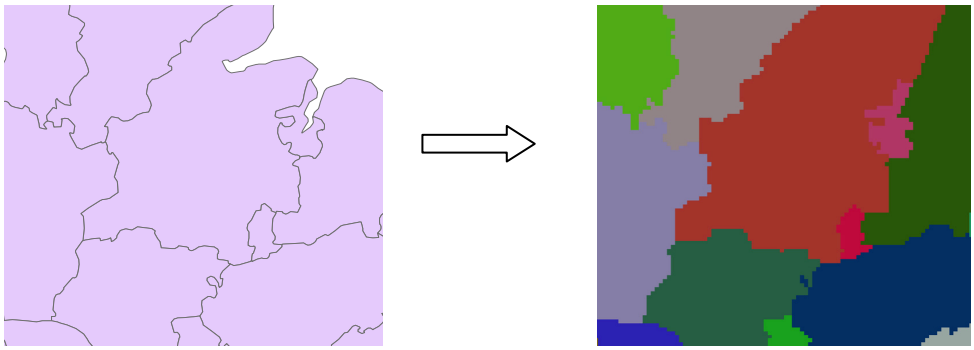
2.6 PROCESSING ANALYTICAL AND REPORTING UNITS

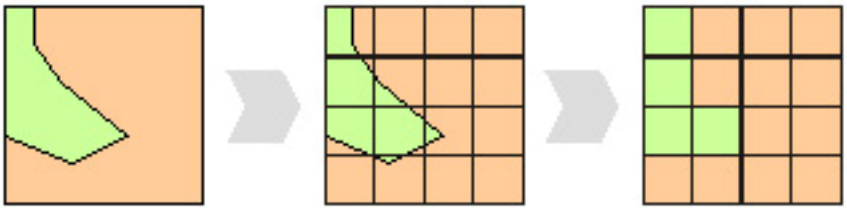
There are a number of LARUS that have been included in the latest version of the LEAC System. The list of datasets is as follows:

- Biogeographic regions (rbio)
- Administrative regions NUTS 2003 (nuts03)
- Administrative regions NUTS 2006 (nuts06)
- Sea Catchments (rsea)
- Elevation Breakdown (ebk)
- 16 Massifs (massifs)
- Dominant Land Cover Types 2006 (dlt06)
- River Basin Districts (rbd)
- Large Urban Zones (luz)
- Urban Audit Cities (city)

2.6.1 Rasterize the vector source

LARUS are normally provided in a vector format with some exceptions such as Dominant Land Cover Types (DLT) and the Elevation Breakdown (EBK). Therefore all these datasets need to be rasterised in order to be included into the System. The rasterization process is performed in ArcView or ArcMap. The "Maximum area" criterion is used, as it is one of the most standard methods for rasterization processes.



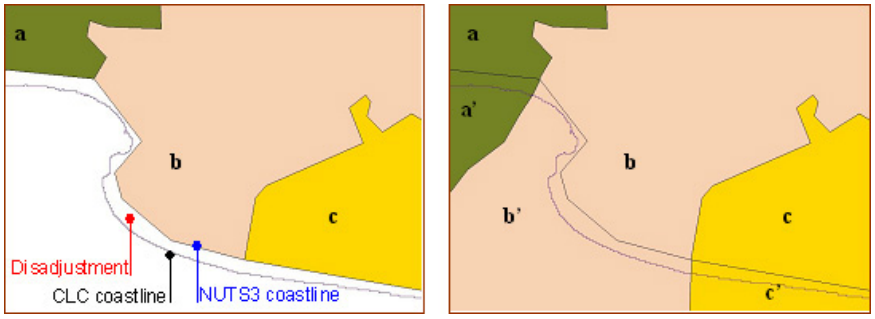


In the rasterization process is very important to set up the analysis parameters: extent and resolution. The extension should cover the maximum estimated extent for the CLC project.

It is important to make sure that extent and resolution are correct.

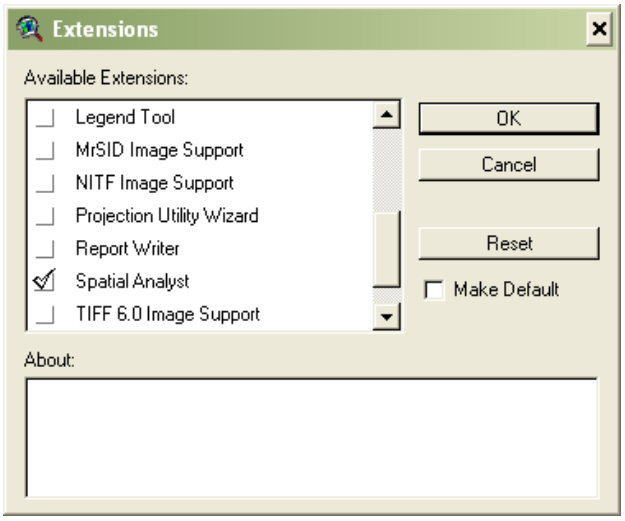
2.6.2 Create proximity surface

Due to geometric shifts between Land Cover layers and thematic layers (LARU) some of the coastal cells fall in "No Data" areas. To minimize this error we create proximity surfaces of the LARU attributes before assigning these values to the grid cells. The proximity surface is a kind of interpolation that assumes that a point with unknown value will get the new value from the closest point with known value.

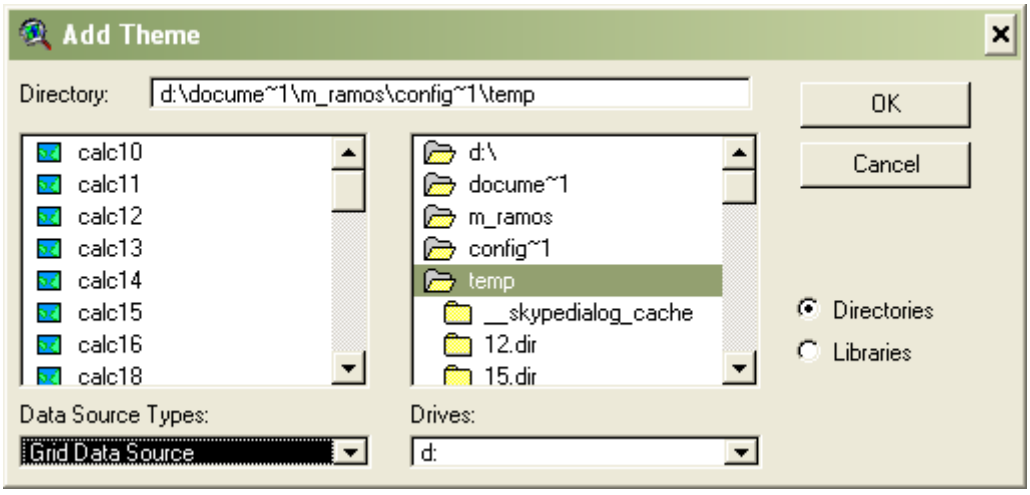


To perform the proximity surface the Spatial Analyst extension of ArcView 3.x is used. First of all, it is important to verify if the Spatial Analyst extension is enabled:

- Open an ArcView 3.3 session and creates a new View
- Go to File>Extensions and click on Spatial Analyst



Now the Spatial Analyst tools will appear in the view. At the moment to introduce the data you must to add them as Grid Data Source, otherwise the Spatial Analyst tools will not be enabled.



The Assign Proximity tool from Analysis Menu in ArcView 3.x allows us to create the grid theme needed that store for each cell the id, or value, of the nearest feature. In this step thematic data is rasterized and interpolated to empty areas (Sea, lakes...)

2.6.3 Rasters reclassification

All LARU rasters should be then reclassified using the Spatial Analyst Tools in ArcToolbox (ArcGIS9.3). All no data values should be changed to "0". As it will be seen in the combining data point, this step is essential to capture information for all the extent.

2.6.4 Output data

2.6.4.1 Creation of a definition Table

It is needed to create and store a definition table for each LARU that means a table where to find the description for each element of the LARU. The field value from the proximity raster will be used in the final processes, for that reason this field will be the id (primary key) of the dictionary tables.

DEFS_DLT : Table

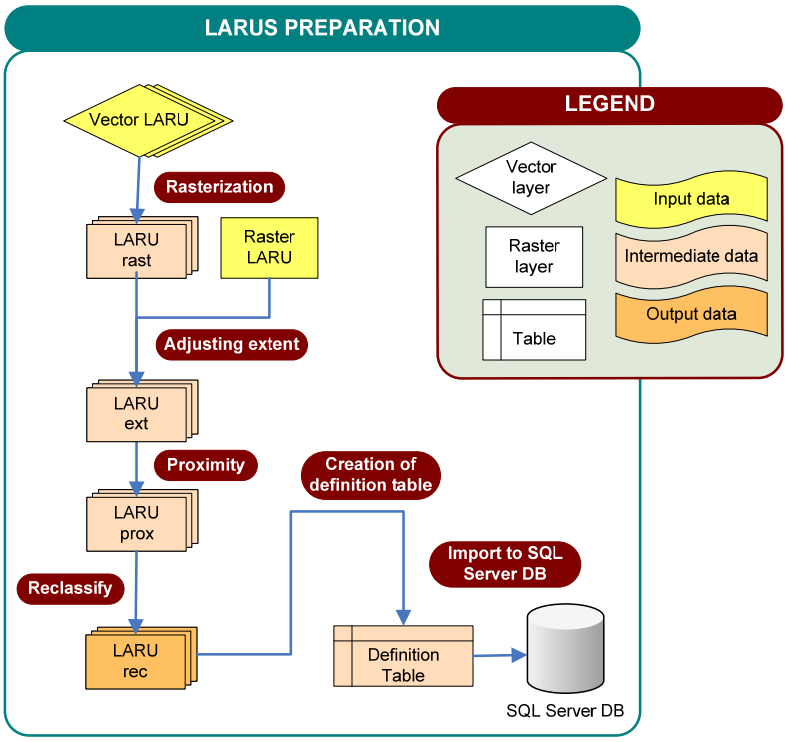
DLT_ID	DLT_CD	DLT_NM	DLT_CDNM
1	A1	Urban dense areas	A1 - Urban dense areas
2	A2	Dispersed urban areas	A2 - Dispersed urban areas
3	B1	Broad pattern intensive agriculture	B1 - Broad pattern intensive agriculture
4	B2	Rural mosaic and pasture landscape	B2 - Rural mosaic and pasture landscape
5	C1	Forested landscape	C1 - Forested landscape
6	C2	Open semi-natural or natural landscape	C2 - Open semi-natural or natural landscape
7	D1	Composite landscape	D1 - Composite landscape

Record: 1 of 7

2.6.4.2 Import definition tables into SQL Server Database

Once the LARU definition tables are ready, then they are imported to the SQL Server Database.

2.6.5 Process workflow

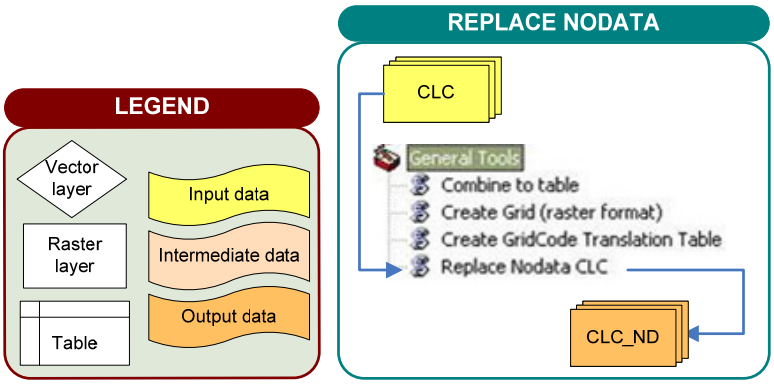


2.7 PROCESSING CORINE LAND COVER DATA

For some countries, CLC may have diverse coverage for different time shots. For instance, a certain country can have information for CLC 90 and for CLC 00, while no data for CLC 06. If a combine is processed just like this, information for this country would be lost as nodata because it will compute as if multiplying by 0. For this reason, a previous step to the combination process is required, a replacement of nodata values is applied with the tool created for this purpose, **Replace Nodata CLC**, which is stored in General Tools in the LEAC toolbox.

The tool replaces the noData values with value 49 in each of the CLC input raster layers. The output is saved in a new layer, appending the provided suffix to the input layer name. The format of the output layer will always be GRID, as Map Algebra always generates uncompressed TIFF files, which creates layers bigger than 10GB in the case of CLC.

2.7.1 Process workflow

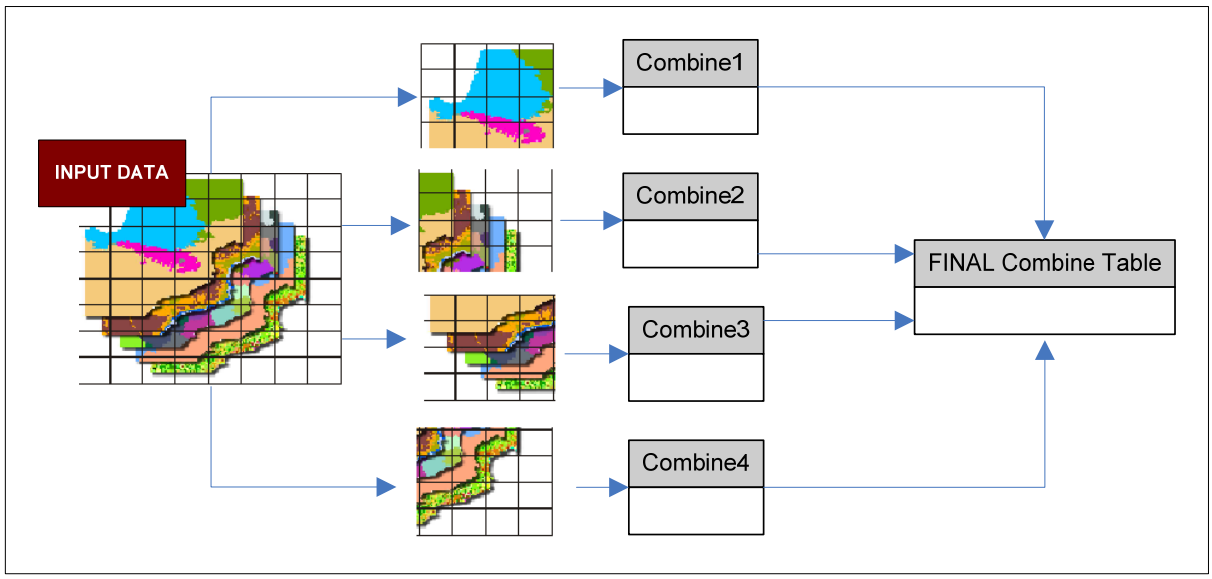


2.8 COMBINING DATA

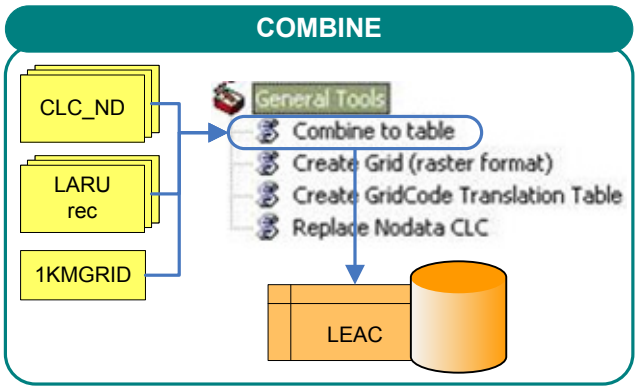
The combine functionality from ArcGIS has been identified as one of the main processes in the new methodology as it substitutes the tabulation process used in previous cubes. The tool Combine to table from LEAC toolbox, is used to combine all LARU datasets (Chapter 2.7) together with Corine Land Cover data (Chapter 2.8) and the 1 Km reference grid (Chapter 2.6).

The **Combine to table** tool combines a set of input rasters (using ArcGIS "Combine" process) and exports the results to a table named "LEAC" in the provided database (file or personal geodatabase).The database should not exist previously, as it will be created in the process.

Input rasters are splitted in sub-zones when combine is applied, so that big raster can be correctly processed. In addition, the default Combine_Cell_Sixe parameter is set to 100m.



2.8.1 Process workflow

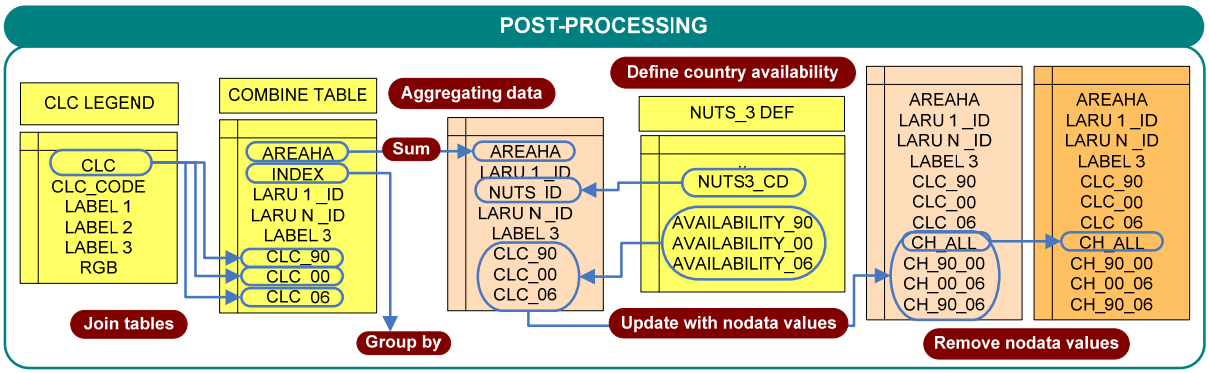


For further details on General tools, scripts and processes see **Annex VI – General Tools toolset.**

2.9 POST-PROCESSING DATA

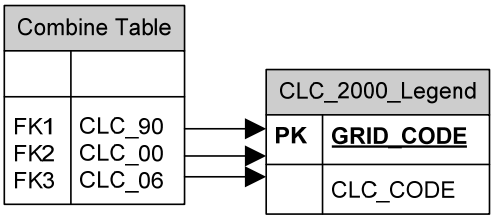
After the combination of data is complete, some last processing is needed previously to build the LEAC cube. This post-processing can be executed in MS Access.

2.9.1 Post-processing workflow

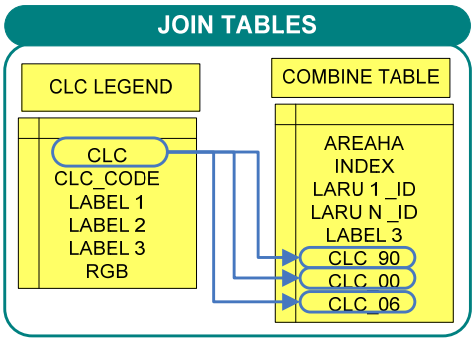


2.9.2 Join tables

The field CLC_90, CLC_00 and CLC_06 has been processed using the Grid-Code value (1-44). At this step, the "clc_2000_legend" table is imported and joined with the main table by the GRID_Code field.

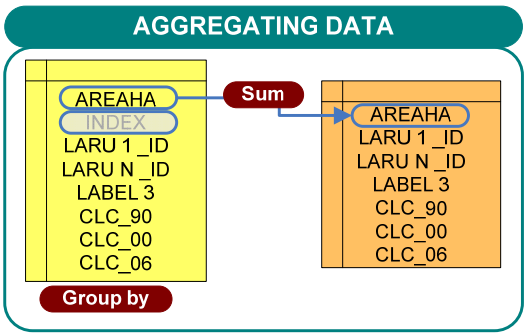


The field CLC_90, CLC_00 and CLC_06 are updated by replacing the Grid-Code Value by the 3 digits CLC class code from the "clc_2000_legend" table. Ex: The value 1 will be replaced by 111.



2.9.3 Aggregating data

In order to reduce the size of the table, the data is aggregated by grouping all fields without taking into account the INDEX FIELD and summing the field "AREAHA".



Example of SQL Sentence (agrega1):

SELECT Sum(COMBINE000690.AREAHA) AS SumOfAREAHA, COMBINE000690.DLT06_ID, COMBINE000690.EBK_ID, COMBINE000690.CITY_ID, COMBINE000690.LUZ_ID, COMBINE000690.MASSIFS_ID, COMBINE000690.NUTS03_ID, COMBINE000690.NUTS06_ID, COMBINE000690.RBD_ID, COMBINE000690.RBIO_ID, COMBINE000690.RSEA_ID, COMBINE000690.CLC_00, COMBINE000690.CLC_06, COMBINE000690.CLC_90 INTO combine_ag1

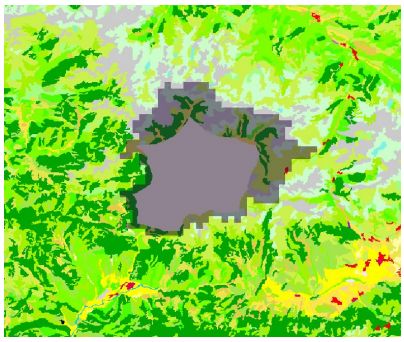
FROM COMBINE000690

GROUP BY COMBINE000690.DLT06_ID, COMBINE000690.EBK_ID, COMBINE000690.CITY_ID, COMBINE000690.LUZ_ID, COMBINE000690.MASSIFS_ID, COMBINE000690.NUTS03_ID, COMBINE000690.NUTS06_ID, COMBINE000690.RBD_ID, COMBINE000690.RBIO_ID, COMBINE000690.RSEA_ID, COMBINE000690.CLC_00, COMBINE000690.CLC_06, COMBINE000690.CLC_90;

2.9.4 Define country availability

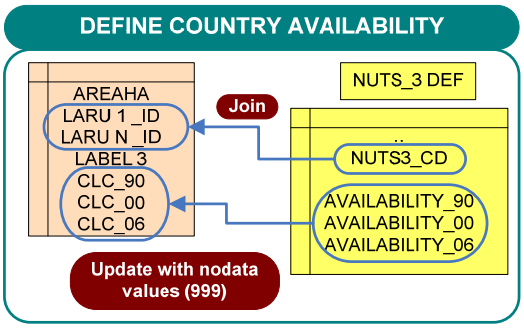
In some cases it may appear some countries without clc data that will have associated some land cover information in the cube. This is due to adjustment problems between administrative borders in raster format and Corine Land Cover data.

As an example, in the image bellow it can be seen the overlapping of Andorra NUTS raster (in transparent dark colour) and Corine land cover 2006.



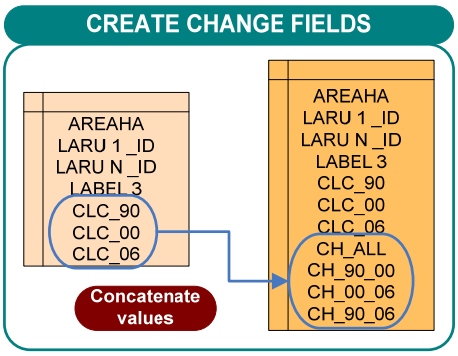
For all that, the resulting table from aggregating data process (see previous step) should be filtered by countries according to data availability, and afterwards those values for countries where there is no Corine available should be removed. To do so, first it is needed to add the availability field for each time shot (as availability can vary depending on the year) in the definition table of NUTS_3 (considering the last version of it) and give

it the corresponding values (1 or 0). Next, it is needed to join NUTS_3 table with the resulting one from aggregating process, and updating CLC values with the corresponding NODATA (=999).



2.9.5 Concatenate values

Changes fields are created for each year combination (1990-2000, 2000-2006 and 1990-2006) by concatenating the CLC data for each year. These fields will be joined with Land Cover Flows information in order to be able to query about LCF in the cube.

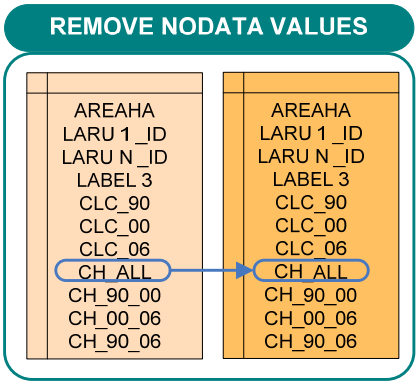


SELECT combines.AREAHA, combines.DLT06_ID, combines.EBK_ID, combines.CITY_ID, combines.LUZ_ID, combines.MASSIFS_ID, combines.NUTS03_ID, combines.NUTS06_ID, combines.RBD_ID, combines.RBIO_ID, combines.RSEA_ID, combines.CLC_00, combines.CLC_06, combines.CLC_90, combines.CLC_90 & [CLC_00] & [CLC_06] AS CHANGES_ALL, combines.CLC_90 & [CLC_00] AS CHG90_00, combines.CLC_00 & [CLC_06] AS CHG00_06, combines.CLC_90 & [CLC_06] AS CHG90_06 INTO LEAC_DATA_3_dates_v2

FROM combines;

2.9.6 Remove no data values

Then we remove all the records with no data for the 3 dates considered. We filter the table where CHANGES_ALL = 999999999 and delete the selected records.



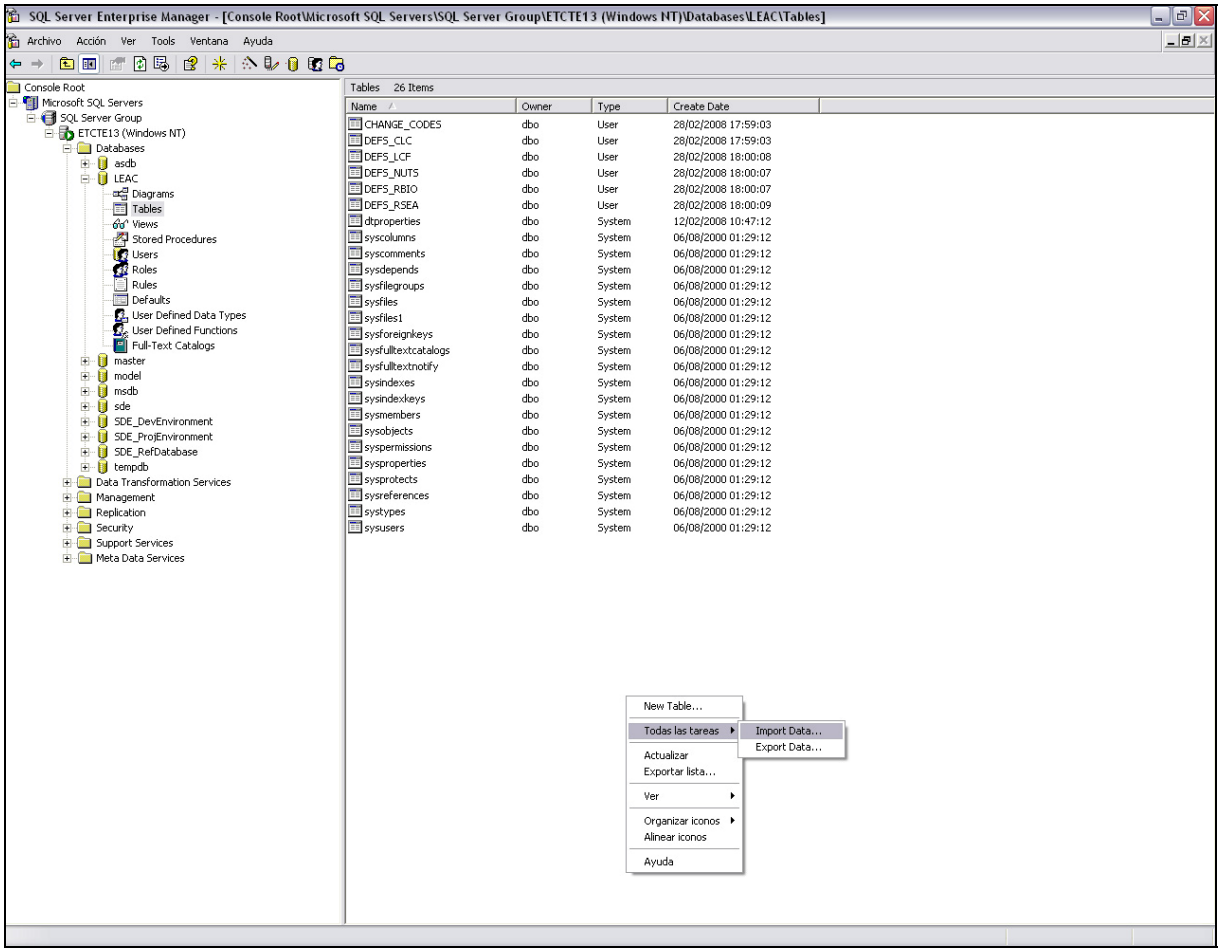
3 BUILDING THE LEAC CUBES

3.1 IMPORTING TABLES INTO MS SQL SERVER

3.1.1 Process:

The current version of SQL Server installed at the ETC-LUSI in ETCTE13 Server is 2000 or 8.00.382. The SQL Enterprise Manager console is used to import tables into the Database. The procedure is as follows:

Click with the right button of the mouse on the "tables" window site. And select "*All tasks*" option and then "*Import Data*" to open the wizard tool.

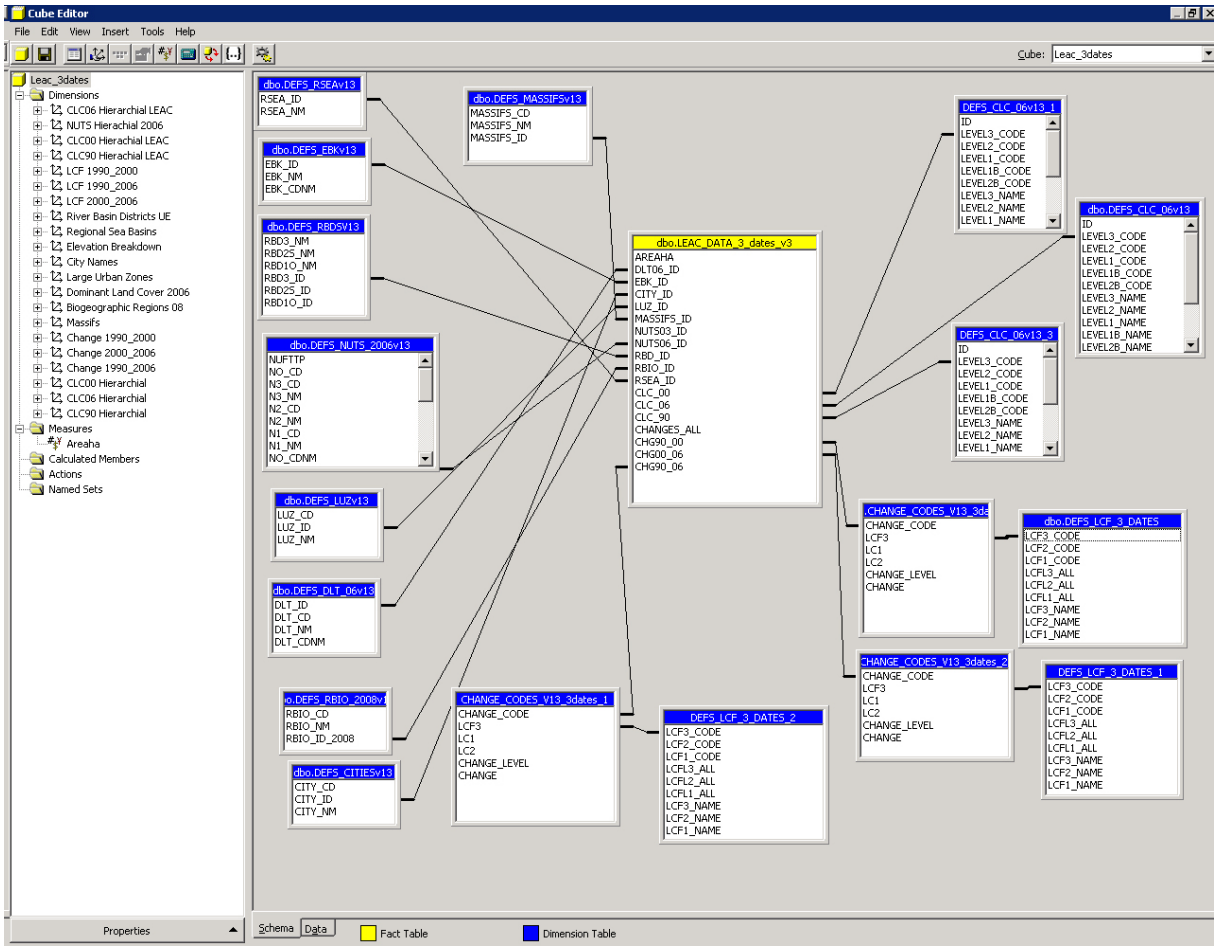


When the wizard is opened there is an option to select the Database from where the data will be imported. The data is stored in an Access database called "Leac.mdb". Choose the SQL Database destination "LEAC" and then select the tables needed to build the cube.

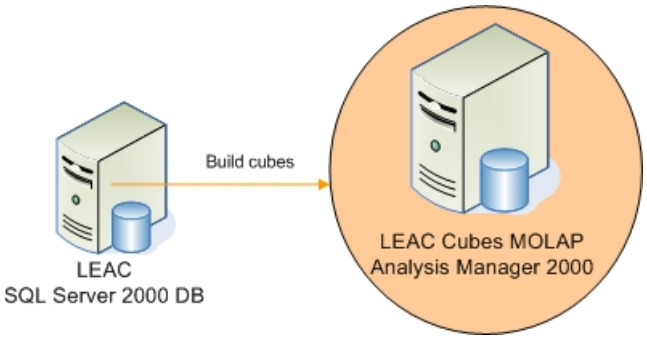
LARU fields should be aggregated and tables should be already included in the Leac Access Database before importing the tables to the SQL database. It is highly recommended to have the same ID for table definitions as in the fact table "CLC90_00" because then SQL will generate automatically the relationships among the tables.

3.2 PREPARING THE LEAC DATA MODEL

Relationships between tables in the LEAC Database:



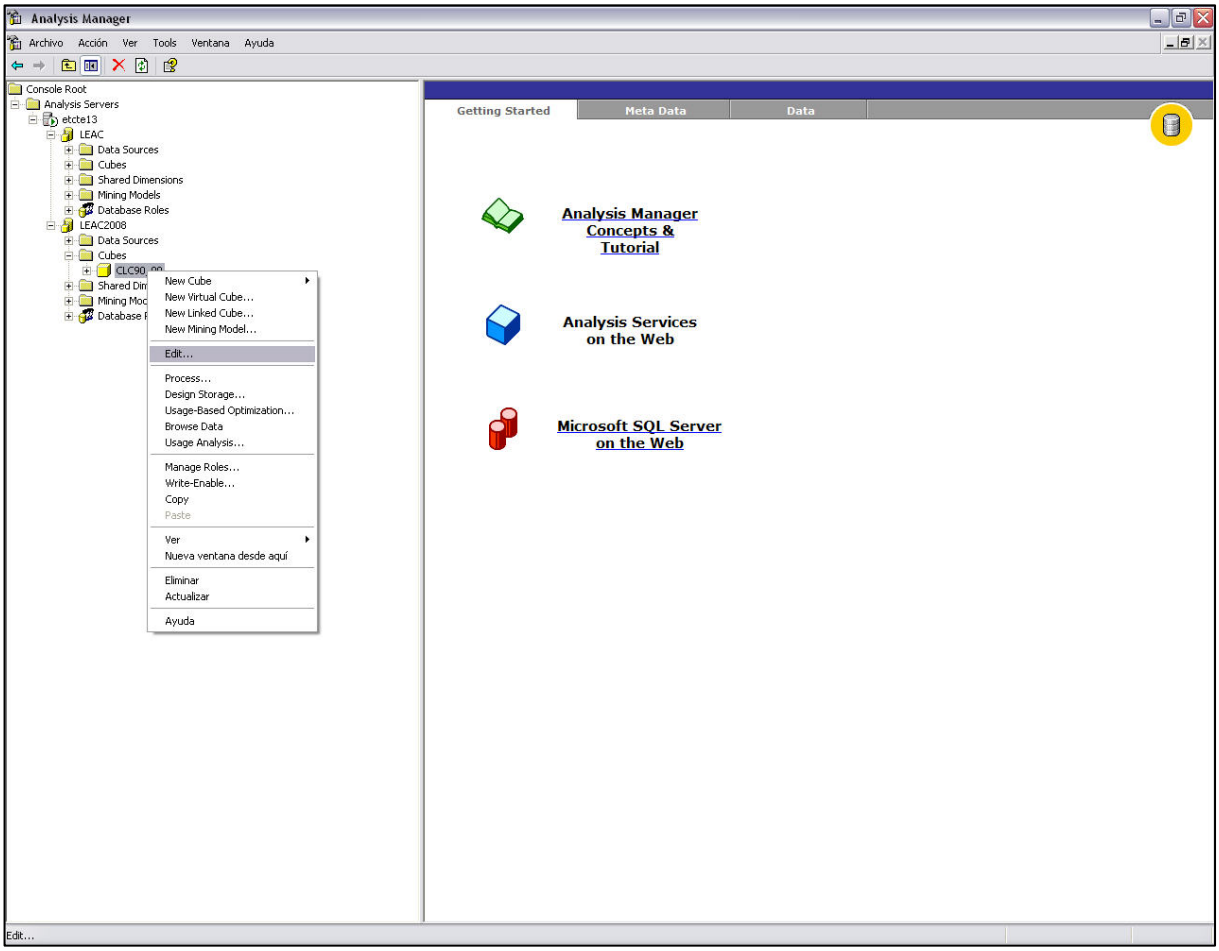
3.3 CREATING OLAP CUBES



Once the data is imported into the SQL Database and the Data Model is well defined we can start building the cube using the Analysis Manager.

Open the Analysis Manager. Register the server "Etcte13" if needed.

Select the Leac cube you want to process and click with the right button of the mouse to edit the cube. If you need to create a new cube you can select the option new cube and choose your fact table, then insert the tables that you previously imported.

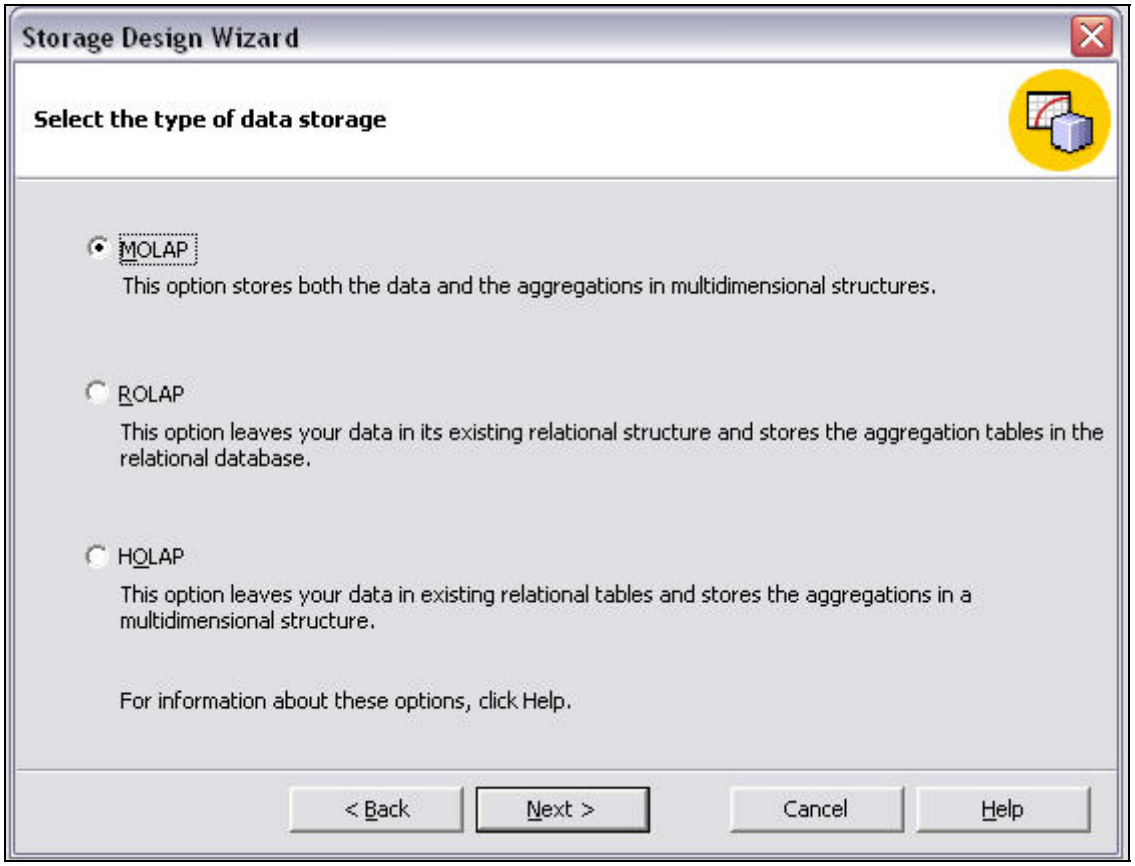


Make sure that all the tables are related. To generate the dimensions for the cube drag the fields or attributes from the tables and drop them into the Dimensions folder in the left hand side window. Rename the dimensions with a comprehensive name.

Measures can be added as the same way that dimensions are generated just by dragging and dropping the attributes to the Measures folder. For Calculated Members click with the right button of the mouse and select edit in case that value expression needs to be changed.

The last step is to process the cube. Click yes when you are asked to save the cube before proceeding. Then you will be asked if you want to design aggregations before processing the cube, click yes to start the Storage Design Wizard.

Within the wizard select the type of Storage as MOLAP and click next. Leave the default set aggregation options and click start.



Note: When the cube is processed an offline OLAP cube can be created from Excel based on the data from OLAP server. This will create a separate file that can be queried without accessing the ETC-LUSI server. To create the file establish a connection through Excel and from the Pivot table menu select the offline OLAP option and choose the dimensions and levels that you want to include in your personal cube file.

3.3.1 Example of list of fields used for Cube Dimensions

- **NUTS Hierarchial 2006** : Dimension table: dbo.DEFS_NUTS_2006v13

  Level 1: Nuts 0 Field table "N0_CDNEM"

  Level 2: Nuts 1 Field table "N1_CDNM"

  Level 3: Nuts 2 Field table "N2_CDNM"

  Level 4: Nuts 2_3 2 Field table "NX_CDNM"

  Level 5: Nuts 3 Field table "N3_CDNM"
- **Biogeographic Regions 08**: Dimension table: dbo.DEFS_RBIO_2008v13

  Level 1: Biogeographic Regions Field table "RBIO_NM"
- **Change 1990_2000 :** Dimension table: dbo.DEFS_CHANGE_CODES_v13_3dates

  Level 1: Change 1990_2000 Field table "CHANGE"
- **Change 1990_2006 :** Dimension table: dbo.DEFS_CHANGE_CODES_v13_3dates_1

  Level 1: Change 1990_2006 Field table "CHANGE"
- **Change 2000_2006:** Dimension table: dbo.DEFS_CHANGE_CODES_v13_3dates_2

Level 1: Change 2000_2006 Field table "CHANGE"

- **City Names** Dimension table: dbo.DEFS_CITIESv13

  Level 1: Field table "CITY_NM"
- **Massifs** Dimension table: dbo.DEFS_MASSIFSv13

  Level 1: Massifs Field table "MASSIFS_NM"
- **CLC 00 Hierarchial LEAC** Dimension table: dbo.DEFS_CLC_06v13_1

  Level 1: Field table "L1B_ALL"

  Level 2: Field table "L2B_ALL"

  Level 3: Field table "L3_ALL"
- **CLC 06 Hierarchial LEAC** Dimension table: dbo.DEFS_CLC_06v13

  Level 1: Field table "L1B_ALL"

  Level 2: Field table "L2B_ALL"

  Level 3: Field table "L3_ALL"
- **CLC 90 Hierarchial LEAC** Dimension table: dbo.DEFS_CLC_06v13_3

  Level 1: Field table "L1B_ALL"

  Level 2: Field table "L2B_ALL"

  Level 3: Field table "L3_ALL"
- **CLC 00 Hierarchial** Dimension table: dbo.DEFS_CLC_06v13_1

  Level 1: Field table "L1_ALL"

  Level 2: Field table "L2_ALL"

  Level 3: Field table "L3_ALL"
- **CLC 06 Hierarchial** Dimension table: dbo.DEFS_CLC_06v13

  Level 1: Field table "L1_ALL"

  Level 2: Field table "L2_ALL"

  Level 3: Field table "L3_ALL"
- **CLC 90 Hierarchial** Dimension table: dbo.DEFS_CLC_06v13_3

  Level 1: Field table "L1_ALL"

  Level 2: Field table "L2_ALL"

  Level 3: Field table "L3_ALL"
- **LCF 1990_2000** Dimension table: dbo.DEFS_LCF_3_DATES

  Level 1: Field table "LCFL1_ALL"

  Level 2: Field table "LCFL2_ALL"

  Level 3: Field table "LCFL3_ALL"
- **LCF 2000_2006** Dimension table: dbo.DEFS_LCF_3_DATES_1

  Level 1: Field table "LCFL1_ALL"

  Level 2: Field table "LCFL2_ALL"

  Level 3: Field table "LCFL3_ALL"
- **LCF 1990_2006** Dimension table: dbo.DEFS_LCF_3_DATES_2

Level 1: Field table "LCFL1_ALL"

Level 2: Field table "LCFL2_ALL"

Level 3: Field table "LCFL3_ALL"

- **Dominant Land Cover 2006** Dimension table: dbo.DEFS_DLT_06v13 Level 1: Dominant Land Cover 2006 Field table "DLT_CDNM"
- **Elevation Breakdown**. Dimension table: dbo.DEFS_EBKv13

  Level 1: Elevation Breakdown Field table "EBK_CDNM"
- **Large Urban Zones** Dimension table: dbo.DEFS_LUZv13

  Level 1: Large Urban Zones Field table "LUZ_NM"
- **Regional Sea Basins** Dimension table: dbo.DEFS_RSEAv13

  Level 1: Regional Sea Basins Field table "RSEA_NM"
- **River Basin Districts UE** Dimension table: dbo.DEFS_RBDSv13

  Level 1: Hydrosystem Ocean Field table "RBD1O_NM"

  Level 2: Hydrosystem Sea Field table "RBD2S_NM"

  Level 3: River Basin Name Field table "RBD3_NM"

4 INTEGRATION OF NEW DATA

The processes described till now make possible the creation of the basic structure of the data to be included in the LEAC System; however it is possible to integrate new data once the basic structure has been created.

4.1.1 Integration of new Land and Analytical Reporting Units

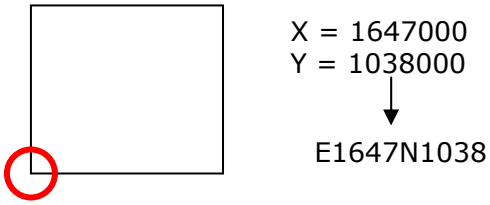
The new data has to follow the steps described in the point 2.6 (Processing Analytical and Reporting Units):rasterize, create proximity surface, replace no-data, create definition table.

The rasterization process will determinate the integration methodology. Whenever is possible the output resolution will be 1000 m; in those cases that a resolution of 1000 m means a big lost of input data, the resolution can be smaller. In this last case, the final resolution of the data model will vary depending on the input data; however the suitable minimum limit is fixed to 100 m (such as CLC resolution) in order to ensure a thematic and technical coherence.

If the output resolution is 1km:

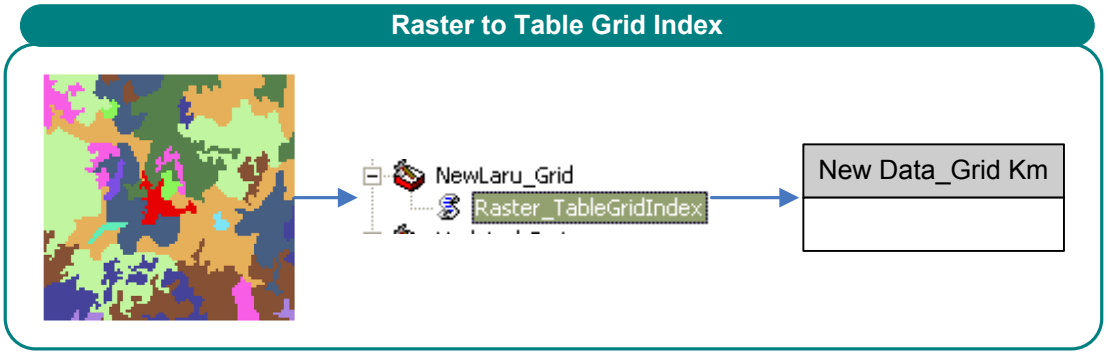
- First, the data has to be aligned to 1km Reference Grid
- Secondly, the data is processed in order to generate a new table with the value of the raster and the value of the grid index.

The grid index is based on the coordinates of the left-down vertex of the pixel:

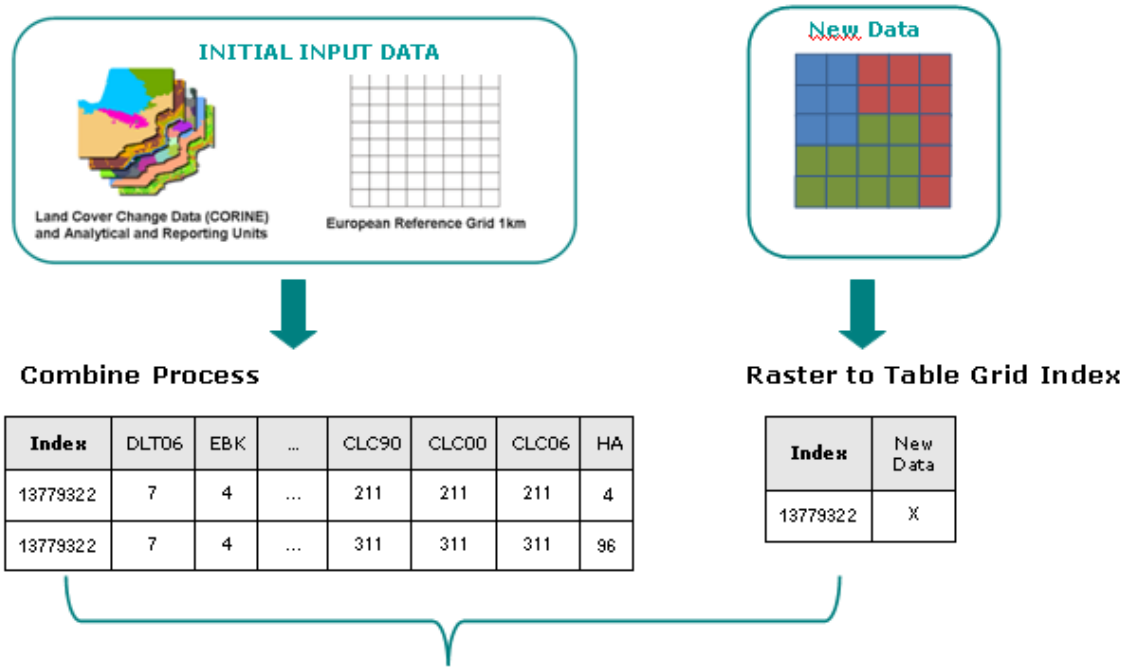


This process has been automated by creating the tool Raster_TableGridIndex in the LEAC toolbox/ NewLaru_Grid, which generates the grid index for each cell in numeric format: 16471038 instead of E1647N1038.

For further details on Raster to Table Grid Index tool, scripts and processes see **Annex VII – Raster to table Grid Index tool.**



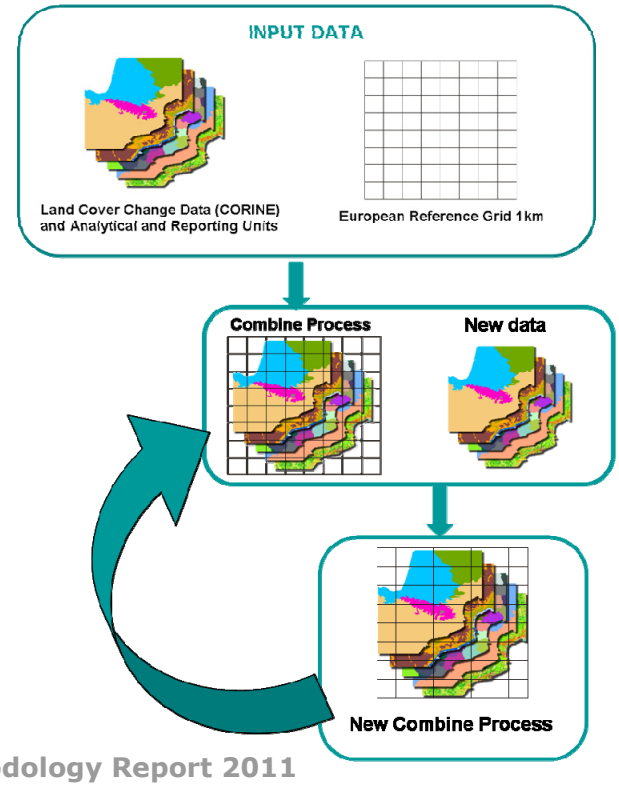
The output table can be directly joined with the output table coming from the combining data process used to build the Basic LEAC data structure (see point 2.8).



The output table has to be submitted to post-processing operations (point 2.9), and the cube has to be rebuilt and processed (point 3).

If the output resolution is 100m:

- The new data has to be submitted to a new combine process with the output raster (rasters) coming from the combining data process used to build the Basic LEAC data structure (point 2.8)



The output data has to be submitted to Post-processing data operations and the cube has to be rebuilt.

4.1.2 Integration of new measures

The addition of new measures, calculated values such as the population, will be based on the distribution of the new data by the 1km GRID. In this sense, for each grid cell the value of the indicator or the new data will be assigned

LEAC Methodology Report 2011

ANNEXES

Type of Document:
ANNEXES

Prepared by:
César Martinez
Maria José Ramos
Raquel Ubach

Date:
10.10.2011

Project Manager:
Oscar Gomez

European Environment Agency

UNIVERSIDAD DE MÁLAGA

Universidad de Malaga
ETCSIA
PTA - Technological Park of Andalusia
c/ Marie Curie, 22 (Edificio Habitec)
Campanillas
29590 - Malaga
Spain

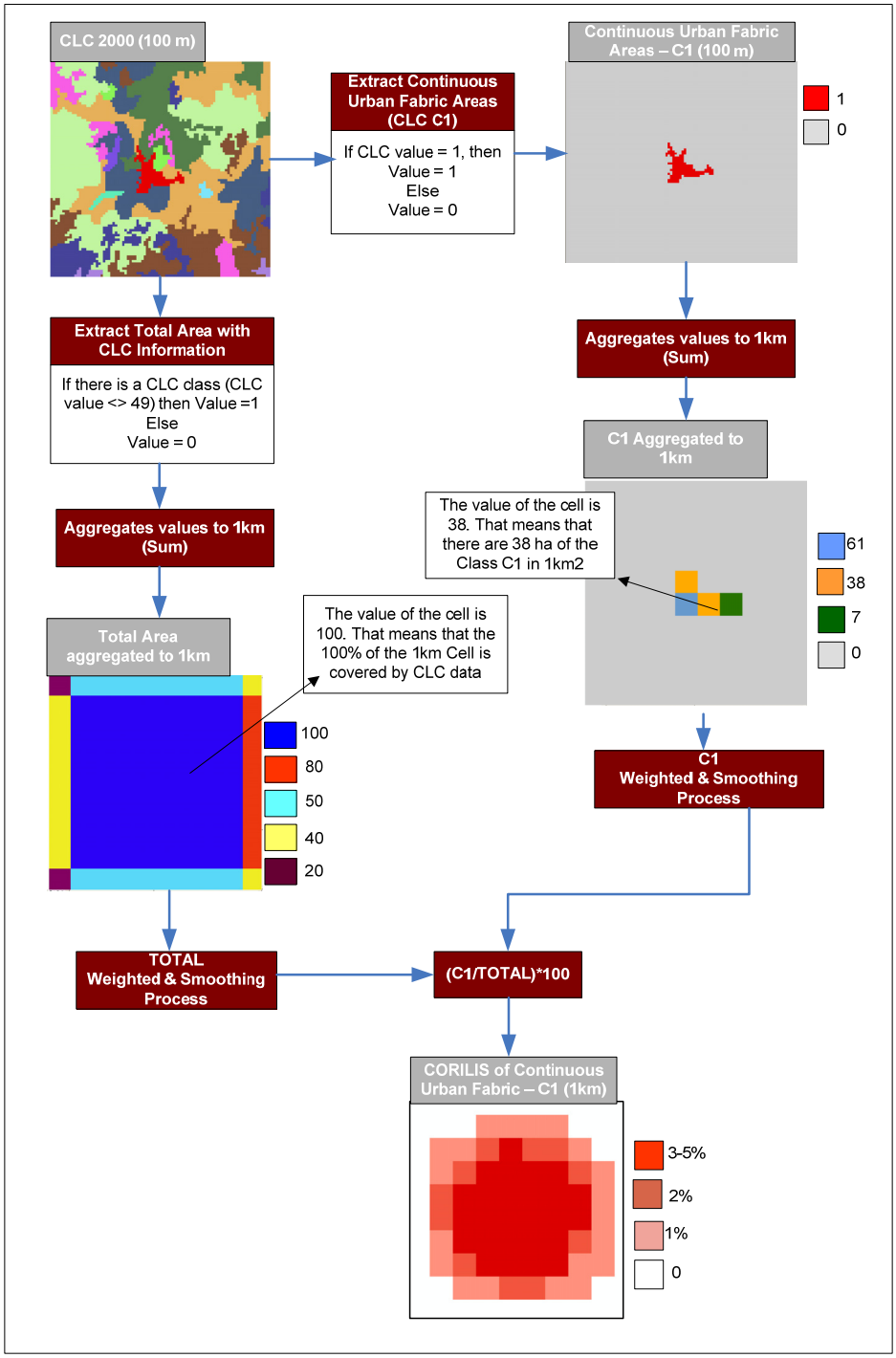
Telephone: +34 952 02 05 48
Fax: +34 952 02 05 59

Contact: etc-sia@uma.es

1 ANNEX I –CORILIS METHODOLOGY AND STEPS

1.1 CORILIS METHODOLOGY

The next figure shows the general schema of Corilis for the C1 class (Continuous Urban Fabric) in a specific region.



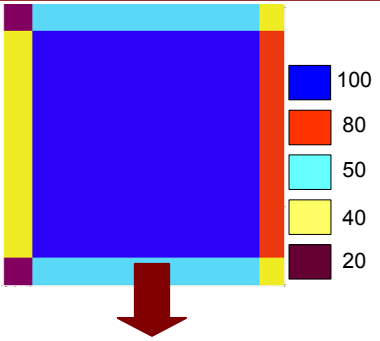
1- On the one hand, a new Boolean raster (1,0) presenting the total area covered by Corine Land Cover is generated. This raster is aggregated to 1km by summing the values. The value of each cell represents the area covered by all the CLC classes

inside the 1km2. Ex: if the cell value is 80 that means that there are 80 ha. Covered by CLC data inside the 1km2 cell.

2- The Total aggregated raster is submitted to a weighted and smoothing process.

Example of the weighed process for a specific pixel of the Total aggregated raster.

Total presence of CLC aggregated to 1km .



Marked with red colour the cell to be calculated

9 X 9

100	100	100	100	100	100	100	80
100	100	100	100	100	100	100	80
100	100	100	100	100	100	100	80
100	100	100	100	100	100	100	80
100	100	100	100	100	100	100	80
100	100	100	100	100	100	100	80
100	100	100	100	100	100	100	80
50	50	50	50	50	50	50	40

×

The values of the cells inside the 9x9 neighbourhood matrix are weighted. The weighted value for the cell to be calculated is 1.

0	0	0,04	0,1024	0,1296	0,1024	0,04	0	0
0	0,0784	0,2304	0,36	0,4096	0,36	0,2304	0,0784	0
0,04	0,2304	0,4624	0,64	0,7056	0,64	0,4624	0,2304	0,04
0,1024	0,36	0,64	0,8464	0,9216	0,8464	0,64	0,36	0,1024
0,1296	0,4096	0,7056	0,9216	1	0,9216	0,7056	0,4096	0,1296
0,1024	0,36	0,64	0,8464	0,9216	0,8464	0,64	0,36	0,1024
0,04	0,2304	0,4624	0,64	0,7056	0,64	0,4624	0,2304	0,04
0	0,0784	0,2304	0,36	0,4096	0,36	0,2304	0,0784	0
0	0	0,04	0,1024	0,1296	0,1024	0,04	0	0

Resulted values from the weighting process.

0	0	4	10,24	12,96	10,24	4	0	0
0	7,84	23,04	36	40,96	36	23,04	7,84	0
4	23,04	46,24	64	70,56	64	46,24	23,04	3,2
10,24	36	64	84,64	92,16	84,64	64	36	8,192
12,96	40,96	70,56	92,16	100	92,16	70,56	40,96	10,368
10,24	36	64	84,64	92,16	84,64	64	36	8,192
4	23,04	46,24	64	70,56	64	46,24	23,04	3,2
0	7,84	23,04	36	40,96	36	23,04	7,84	0
0	0	2	5,12	6,48	5,12	2	0	0

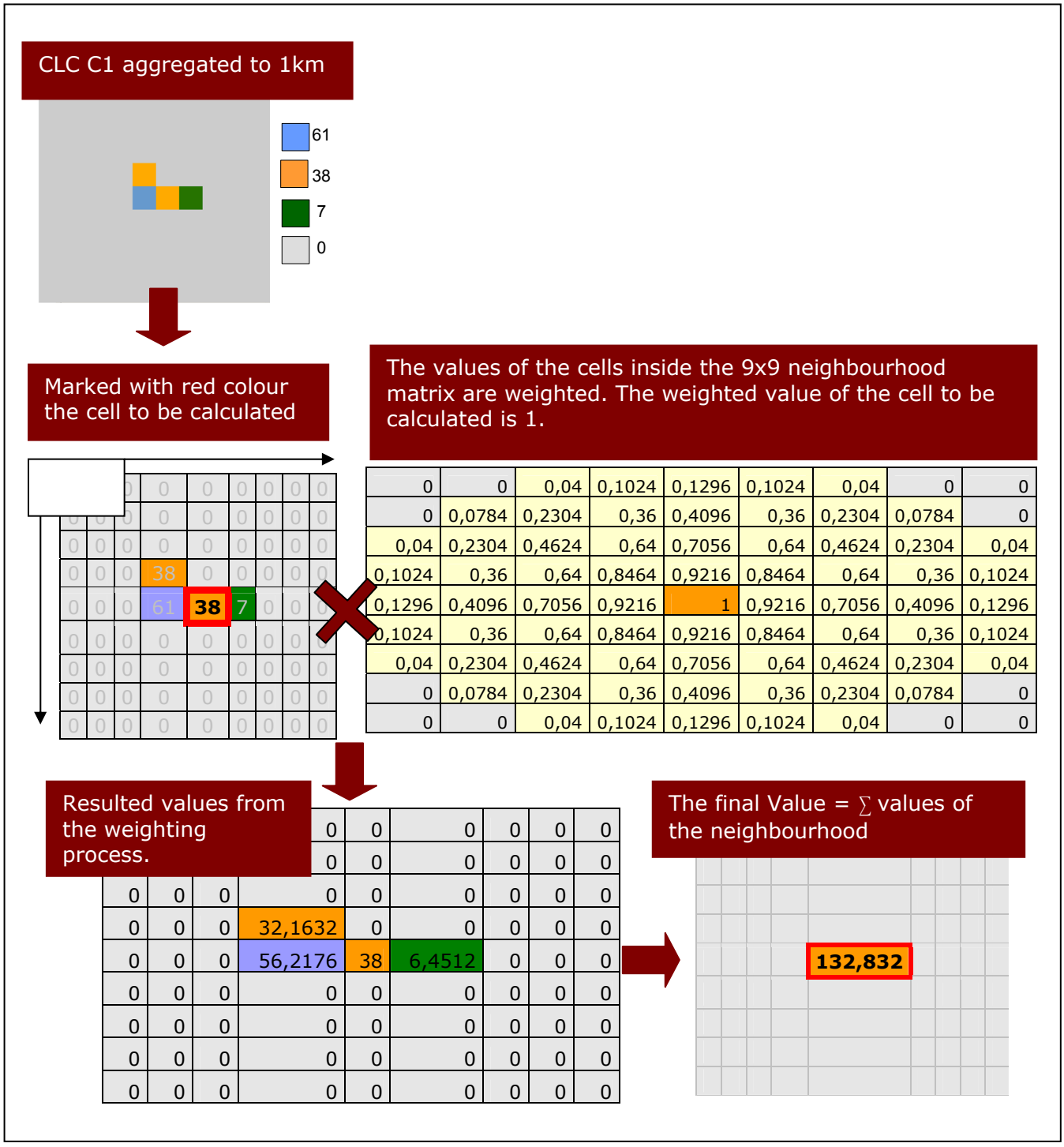
The final Value = ∑ values of the neighbourhood (9x9 Matrix)

2590,672

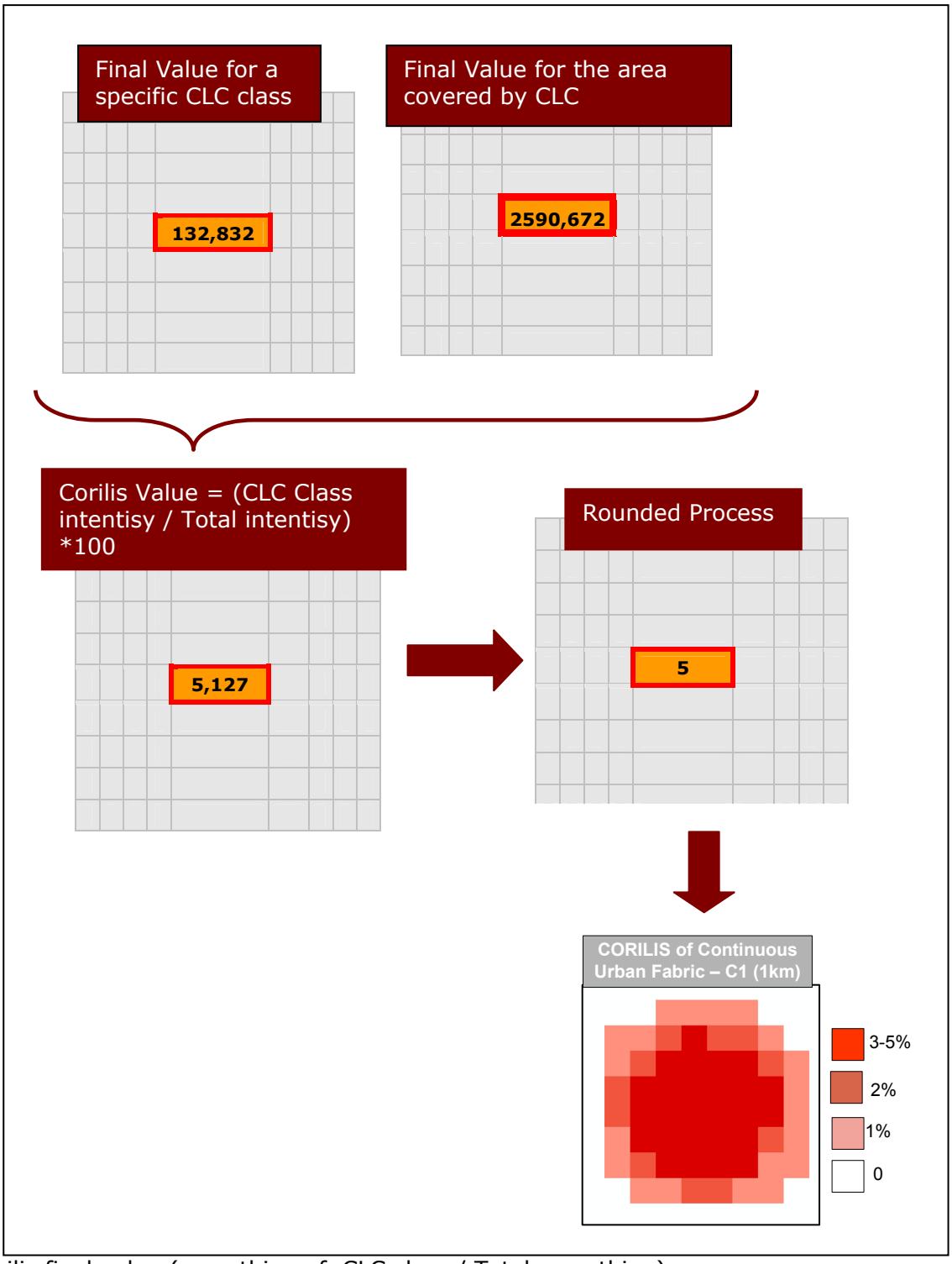
3- On the other hand, the area covered by a specific CLC class is extracted to a new Boolean raster (1,0). This raster is also aggregated to 1km by summing the values. The value of each cell represents the area of the CLC class in the 1km. Ex: Ex: if the Class to be processed is C1 and the value of the cell is 38, that means that there are 38 ha of the Class C1 inside the 1km2 cell.

4- The CLC class aggregated raster is submitted to a weighted and smoothing process.

Example of the weighed process for a specific pixel of the CLC class aggregated raster (in this case, it is an extraction of the CLC C1 class)



5- Finally Corilis output raster is calculated as the percentage of C1 "intensity" in relation to the total "intensity"



Corilis final value (smoothing of CLC class / Total smoothing)

1.2 CORILIS STEPS

1.2.1 Connections parameters of the Linux machine

1- Username: **etclusi**

2- Password: **etclusi2010**

1.2.2 Writing the outputs directly to a network folder (LEACWK)

1- The computer must to be connected to the windows network (Ethernet cable).

2- It is needed to activate the link between a local folder from the linux computer and the folder in the server, ex: /mnt/peterpan (computer) with /PeterPan/leacwk.

3- To activate the link you must to run the mountPeterpan.sh script with the variable: username. The script will ask two password: the one of the computer (user etclusi) and the one of the specified variable:

Script:

```
sudo mount -t cifs //peterpan.uab.es/leacwk /mnt/peterpan -o noauto,rw,users,username=$1,workgroup=ETCLUSI,uid=1000
```

To run the script:

4- Open the unix terminal

5- Write the path to the script + variable : path/mountPeterpan.sh usuari

   a. Ex: Desktop/mountPeterpan.sh m_ramos

6- Introduce the password of the computer:

   b. [sudo] password for etclusi: etclusi2010

7- Introduce the password of the user:

   c. Password: password of m_ramos

One you are connected it is possible to create folders and to work directly from the laptop opening /mnt/peterpan.

1.2.3 Adapt the h1.sh and h2.sh script.

a) First of all, it is high important to know the inputs and its properties, for example:

Corine Land Cover 00:
- Format: grid
- No Data value: 49
- Extent:
  - Top: 6500000
  - Left: 800000
  - Right: 7500000
  - Bottom: 700000
- Class 44: yes

b) Secondly, for each input describes the expected output by the next parameters that will have to be introduced inside the h1.sh or h2.sh script.

The classes from Corine Land Cover that you want to process	**Start Class**	The initial class you want to process: ex-1
	End Class	The final class you want to process: ex- 44
The radius to process Corilis	**Radius**	In general a radius of 5 or 10 kilometers is used. This value has to be greater than 0.
The extent you want process. Maybe you only want to process a specific region	Top	6500000
	Bottom	700000
	Left	800000
	Right	7500000
The Final Resolution	**Res**	The final resolution you want. In general the resolution is set to 1000 (1km)

Ex:

Input CLC00: I would like to process corilis in the next scenarios:

Scenario 1: For each class (star:1; end:44) using a radius of 5 for all the input extent (Top:6500000;Bottom:700000; Left:800000; Right:7500000) and with a final cell size of 1km (res:1000)

Scenario 2: Only for certain classes (star:4; end:8) using a radius of 15 for a certain region (Top: 2220180; Bottom: 2042712; Left: 3324770; right: 3555479) and with a final cell size of 100 (res:100).

Scenario 3: Only for certain classes (star:16 end:21) using a radius of 5 for a certain region (Top: 2220180; Bottom: 2042712; Left: 3324770; right: 3555479) and with a final cell size of 100 (res:100).

Input CLC90: I would like to process corilis in the next scenarios:

Scenario 4: For each class (star:1; end:44) using a radius of 5 for all the input extent (Top:6500000;Bottom:700000; Left:800000; Right:7500000) and with a final cell size of 1km (res:1000)

Once the scenarios are clear, in order **to divide the work in two grass sessions**, you have to decide which processes **will take place in mapset1 (h1.sh)** and which one in **mapset2 (h2.sh).**

Ex:

Scenario 1 and 2 in Mapset1 (h1.sh): the input data is CLC00

Scenario 3 and 4 in Maset2 (h2.sh): the input data is CLC90 and CLC00

c) The h1.sh and h2.sh facilitate the creation of different corilis in a sequentially order without having to run the script each time when one corilis is finished.

these scripts call another script (corilis.sh) passing it the needed variables to make the process possible.

Modify each script adapting it to the new scenarios. Open h1.sh script and go the end of the document and introduce the new code lines. In blue are the parameters that the user must to modify.

Bash $4/corilis.sh [data1]@$MAPSET [start_class] [end_class] [Radius] [top] [bottom] [left] [right] [resolution] [no_data value] [output_folder]

- [data1]: the name of the input raster map but without the extension of the mapset.
- [start_class], [end_class] and [radius]
- The region so it is necessary to pass the next variables[top] [bottom] [left] [right] [resolution]
- The [nodata] value the input raster (we have found that sometimes is 49 and another times is 255)
- The output location:
    - If we are connected to the network(see point 1.a)
        - /mnt/peterpan/*folder*

Write as many new lines as processes you want for the MAPSET1. Following with the previous examples, in h1.sh for the scenario 1 and 2, the user should introduce two new lines in the code:

 # Mapset1- h1.sh

#For the scenario 1

Bash $4/corilis.sh clc00@$MAPSET 1 44 5 6500000 700000 800000 7500000 1000 49 /output_1

#For the scenario 2

Bash $4/corilis.sh clc00@$MAPSET 4 8 15 2220180 2042712 3324770 3555479 100 49 /output_2

Go to the h2.sh script and write also as many new lines as processes you want to be taken place in MAPSET2.

 # Mapset2- h2.sh

#For the scenario 3

Bash $4/corilis.sh clc00@$MAPSET 16 21 5 2220180 2042712 3324770 3555479 100 49 /output_3

#For the scenario 4

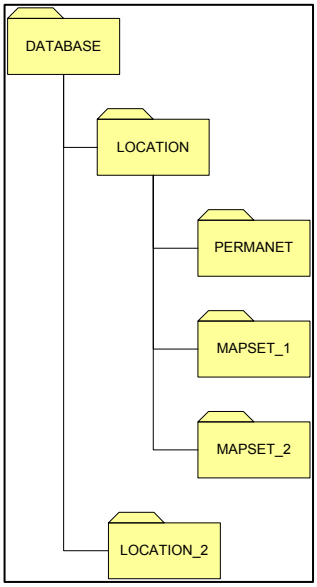
Bash $4/corilis.sh clc90@$MAPSET 1 44 5 6500000 700000 800000 7500000 1000 49 /output_4

1.2.4 Grass Structure

Grass is a Geographic Information System (GIS) software used for geospatial data management and analysis, image processing, graphics/maps production, spatial modelling, and visualization.

Grass works based on the LOCATION/MAPSET concept:

- DATABASE is a folder that will store both location and mapsets
- LOCATION is a permanent directory that has a specific geographic extent and contains data that (should) all be in the same coordinate system.
- MAPSET is a subdirectory of the location where it is able to organize GIS maps thematically or geographically for example.
- PERMANET mapset usually contain read-only base data but also contains some information about the location itself that is not found in other mapsets (projection info etc.), thus it must exist in every location.



To work with Grass it is necessary to have:

- a location and a mapset created with a determinate extent
- input data that will be used in the processes.

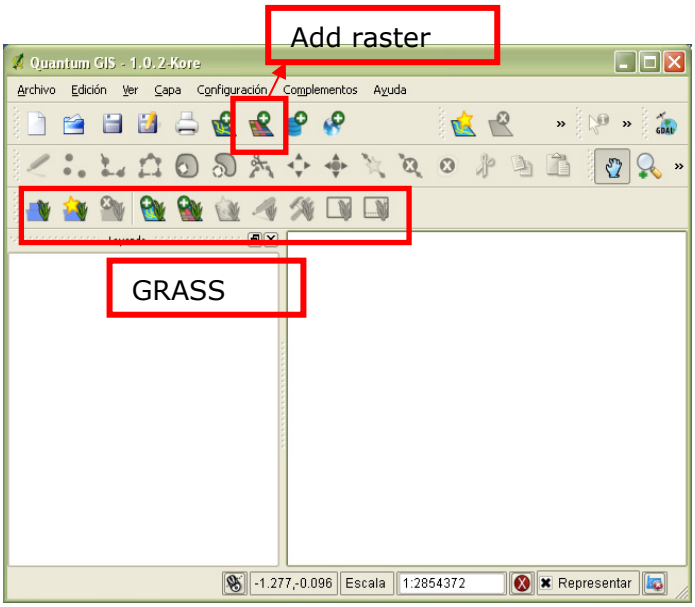
The easier way is to use Quantum GIS to create the Grass structure and import the data

Once the location and the mapsets are created then it is possible to run the shells scripts.

Once you have decided which processes will take place in each mapset (mapset1 and mapset2) you have to create the Grass structure (gisdbase, location, mapset1 and mapset2).

A) Open QGIS: To open Quantum GIS: Applications> Science>Quantum GIS

Note » these steps take into account that the plugin for grass is already enabled if not you must go to Plugins>QGIS Plugin Manager- enabled Grass.



B) Open the raster that will be processed, it can be open in grid format (Arc/Info Binary Grid Format):

b.1) Add Raster Layer

1. Browse to the folder (click Places)
2. Select all other files
3. Select hdr.adf (this is the header, and contains information on the tile sizes, and number of tiles in the dataset. It also contains assorted other information).

b.2) Specify the coordinate system and units of the Quantum GIS project.

4. File
5. Project properties
6. General > Layer Units (Meters)
7. Coordinate Reference System (CRS) > Projected Coordinate Systems> Lambert Azimuthal Equal Area> ETRS89/ ETRS-LAEA

C) Create a new location and Mapset for Grass

1. New mapset
2. Select a database (where you want to store the location and mapset where Grass will work). You have to select an existing folder, so if it has to be created previously. [e.g. Gisdbase_Corilis]
3. Create a new location.
   a. Write a name [e.g. location_Corilis]
   b. Define the coordinate System : Projected Coordinate Systems> Lambert Azimuthal Equal Area> ETRS89/ ETRS-LAEA

D) Default Grass Region: Set current QGIS extent

The Default Grass Region is the region of the location and it is the default, the mapset inside a location can have different regions but always the extent must be inside the localiton extent, that is the default.

One mapset can have more than one region but always one at a time. For that reason, the region is a parameter to be specified inside the scrip2 and script3 because it will be changed dynamically depending on the data to be proceeded.

E) Mapset; write a name [e.g. Map1_corilis]

"New mapset successfully created and set as current working mapset".

F) Import the data into grass location and mapset

c. Grass tools > File > Import > Import raster

d. r.in.gdal.qgis- Import loaded raster or r.in.gdal- import GDAL supported raster

[GDAL raster layer: hdr]

e. Give a proper name

[name for output raster map: clc90]

f. Run

You will have to repeat this step for every data you want to introduce inside the MAPSET.

Note that the STEP E and STEP F has to be repeated TWICE. One for the MAPSET1 and other MAPSET2. At the end the structure would be like this:

- Database: Gisdbase_Corilis
- Location: location_Corilis
- Mapset Map1_corilis: (with data from 90,00,06)
- Mapset Map2_corilis: (with data from 90,00,06)
- Mapset: PERMANENT

The next link goes to a web where it is able to find all the command available to be used in grass with its explanations and details:

http://grass.itc.it/grass65/manuals/html65_user/general.html

C) CLOSE Quantum GIS. it is not necessary to save it, because it is only an intermediate software

1.2.5 Run the ho.sh script

1- Open the unix terminal

2- Write the path to the script + variables : path/ho.sh *[gisdbase], [location], [mapset1] and [mapset2]*

a. Note: gisdbase string has to contain the path to the folder: ex: /home/gisdbase
b. Note: location string is only the name of the location folder without the path

Ex: Corilis_scripts/ho.sh Gisdbase_Corilis location_Corilis Map1_corilis Map2_corilis

Points to be taken into account about the scripts:

- All the scripts must be in the same directory.
- The scripts can be opened with any text editor, but can appear a problem if you copy and paste the file and open it with the default editor of Ubuntu, in particular with the lines in white in the text. Solution: open with Kate, select all >Tools>End of line>unix>save
- Important enabled the scripts to be executed:
  - Right button > properties > Permissions > Allow executing file as program
- To run the script inside grass write in the grass manager:
  bash /home/...../file.sh
- The time processing depends a lot on the data extent.

Ex: `g.region n=4200000 s=900000 w=1500000 e=6100000 res=1000 (6 min/ classe)`
a g.region n=6500000 s=700000 w=800000 e=7500000 res=1000 (15 min/classe)

Although inside the script the nodata parameter is set to 255 in the r.out.gdal function. The corilis.sh script can show a message at the end of the process referring to the nodata parameter *"Input raster map contains cells with NULL-value (no-data). The value 255 was used to represent no-data values in the input map. You can specify nodata value by nodata parameter."* This message is not important.

1.3 CORILIS SCRIPTS

1.3.1 H0.sh

```
#!/bin/bash
#Initial script that launch two new terminals and executes one script on each terminal.
# Usage: bash h0.sh [GISDBASE] [LOCATION] [MAPSET1] [MAPSET2]
# MRJ-2010
# Source bash_profile (LD_LIBRARY_PATH)
#. /home/peifer/.bash_profile
# All the scripts must be in the same directory
dire=${0%/*}
echo $dire
xterm -e $dire/h1.sh $1 $2 $3 $dire&
xterm -e $dire/h2.sh $1 $2 $4 $dire&
```

1.3.2 H1.sh

```
#!/bin/bash
#This scripts initiate a Grass session by specifiying its variables and launch the Corilis
#script in a batch way to process all the specified layers.
#This script must be modify each time a new CORILIS version is needed. It is necessary
#to add, delete or modify lines at the end depending on the case. See line 39.
# Usage: bash h1.sh [GISDBASE] [LOCATION] [MAPSET]
# MRJ-2010
# Source bash_profile (LD_LIBRARY_PATH)
#. /home/peifer/.bash_profile


#Verify the arguments:
if [ -n"$1" ] && [ -n"$2" ] && [ -n"$3" ] ; then
        echo "$1 $2 $3"
else
        echo "you must to define the needed parameters: GISDBASE, LOCATION AND
MAPSET"
exit 1
fi
#Established grass variables
export GISDBASE=$1
export LOCATION=$2
export MAPSET=$3
USUARIO=$GISDBASE
tmpfile=.grass6_rc_$MAPSET

echo "GISDBASE: $GISDBASE
LOCATION_NAME: $LOCATION
```

```
MAPSET: $MAPSET
" > $USUARIO/$tmpfile
export GISRC=$USUARIO/$tmpfile

export GISBASE=/usr/lib/grass64
export PATH=$PATH:$GISBASE/bin:$GISBASE/scripts
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$GISBASE/lib
#Now it is able to run grass commands
g.gisenv
g.region -p
#Call to corilis script
#Add a line, delete it or modify it depending on the RASTERS INSIDE THE SAME MAPSET to be processed (clc2000, clc1990, clc20006), the region to be obtained, the resolution to be obtained or the classes to be processed.
#Usage:[src_dataset] [start_class] [end_class] [radius] [north_region] [south_region] [west_region] [east_region] [res_region] [nodata_landmask] [output_location]

#All the scripts must be in the same folder
#EX:
#bash $4/corilis.sh test1@$MAPSET 1 2 5 6500000 700000 800000 7500000 1000 49 /home/etclusi/raquel/test1
```

1.3.3 H2.sh

```
#!/bin/bash
#This scripts initiate a Grass session by specifiying its variables and launch the Corilis script in a batch way to process all the specified layers.
#This script must be modify each time a new CORILIS version is needed. It is necessary to add, delete or modify lines at the end depending on the case. See line 39.
# Usage: bash h2.sh [GISDBASE] [LOCATION] [MAPSET]
# MRJ-2010
# Source bash_profile (LD_LIBRARY_PATH)
#. /home/peifer/.bash_profile


#Verify the arguments:
if [ -n"$1" ] && [ -n"$2" ] && [ -n"$3" ] ; then
        echo "$1 $2 $3"
else
  echo "you must to define the needed parameters: GISDBASE, LOCATION AND MAPSET"
  exit 1
fi
#Established grass variables
export GISDBASE=$1
export LOCATION=$2
```

```
export MAPSET=$3
USUARIO=$GISDBASE
tmpfile=.grass6_rc_$MAPSET

echo "GISDBASE: $GISDBASE
LOCATION_NAME: $LOCATION
MAPSET: $MAPSET
" > $USUARIO/$tmpfile
export GISRC=$USUARIO/$tmpfile

export GISBASE=/usr/lib/grass64
export PATH=$PATH:$GISBASE/bin:$GISBASE/scripts
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$GISBASE/lib
#Now it is able to run grass commands
g.gisenv
g.region -p
#Call to corilis script
#Add a line, delete it or modify it depending on the RASTERS INSIDE THE SAME MAPSET to be processed (clc2000, clc1990, clc20006), the region to be obtained, the resolution to be obtained or the classes to be processed.
#Usage: [src_dataset] [start_class] [end_class] [radius] [north_region] [south_region] [west_region] [east_region] [res_region] [nodata_landmask] [output_location]
#All the scripts must be in the same folder
#EX:
#bash $4/corilis.sh test1@$MAPSET 1 2 5 6500000 700000 800000 7500000 1000 49 /home/etclusi/raquel/test2
```

1.3.4 Corilis.sh

```
#!/bin/bash
# Generate CORILIS raster layers by CLC class, following the CORILIS methodology
# see http://dataservice.eea.europa.eu/download.asp?id=18758&filetype=.pdf
# Usage: bash corilis.sh [src_dataset] [start_class] [end_class] [radius] [north_region] [south_region] [west_region] [east_region] [res_region] [nodata_landmask] [output_location]
# Hermann, October 2009

# Source bash_profile (LD_LIBRARY_PATH)
#. /home/peifer/.bash_profile

#Script modify to implement more variables in order to allow to run it in a batch way: MRJ 2010

# evaluate the environment
eval $(g.gisenv)

# define LOCATION
LOCATION="$GISDBASE/$LOCATION_NAME/$MAPSET"
```

```
# Input raster data, start, end class and radius
#-n"$1" && -n"$2" && -n"$3" && -n"$4" && -n"$5" && -n"$6" && -n"$7" && -
n"$8" && -n"$9" && -n"$10" && -n"$11" ] ; then
# If we have 4 command line arguments
if [ -n"$1" ] && [ -n"$2" ] && [ -n"$3" ] && [ -n"$4" ] && [ -n"$5" ] && [
-n"$6" ] && [ -n"$7" ] && [ -n"$8" ] && [ -n"$9" ] && [ -n"${10}" ] && [ -
n"${11}" ] ; then
       SRC_DATA=$1
       START=$2
       END=$3
       R=$4
       n=$5
       s=$6
       w=$7
       e=$8
       res=$9
       nodata=${10}
       out_loc=${11}
    echo "$SRC_DATA $START $END $R $n $s $w $e $res $nodata $out_loc"
# Print error message
else
       echo "Wrong command line arguments"
       echo "Usage: sh corilis.sh [src_dataset] [start_class] [end_class]
[radius]   [north_region]   [south_region]   [west_region]   [east_region]
[res_region] [nodata_landmask] [output_location]"
       exit 1
fi


g.region n=$n s=$s  w=$w e=$e res=$res


# Create a weights file
N=$(( $R * 2 - 1 ))


awk -vR=$R -vN=$N '
BEGIN {
       for (i = 1; i <= N; i++) {
               for (j = 1; j <= N; j++) {
                       d = sqrt((i - R) ^ 2 + (j - R) ^ 2)
                       printf "%.7f%s", d <= R ? (1 - (d / R) ^ 2) ^ 2 : 0,
j == N ? "" : " "
               }
               print ""
       }
}' > $out_loc/weights.file

# Define base name

SRC="${SRC_DATA%@*}"

# Create a rules file for r.colors
printf "0 black\n100 white\n" > $out_loc/grey100.rule


# Create a land mask and class0 (total land area), if not existing
echo "valorating the existence of $LOCATION/cellhd/${SRC}_LAND"

if [ ! -f $LOCATION"/cellhd/"${SRC}_LAND ] ; then
```

```
        echo
        echo "Starting time: $(date)"
        echo "Creating a LAND mask for ${SRC}.."

        # Create a rules file for r.reclass
        printf  "${nodata}= 0\n* = 1\n" > $out_loc/reclass.rule

        # reclass the original CLC 100m raster layer
        r.reclass "$SRC_DATA" out=${SRC}_LAND rules=$out_loc/reclass.rule

        echo "Calculating total land area for 1 km grid cells.."
        r.resamp.stats ${SRC}_LAND out=${SRC}_c0 method=sum --overwrite
        r.neighbors        ${SRC}_c0        out=${SRC}_c0_smth        size=$N
weight=$out_loc/weights.file method=sum --overwrite
fi



# Create CORILIS raster layers by CLC GRID_CODE
for (( c = $START ; c <= $END ; c++ ))
do
        # print some info to stdout
        echo
        echo "$(date) - $SRC - Class: $c"

        # Create a rules file for r.reclass
        printf "%s = 1\n* = 0\n" $c > $out_loc/reclass.rule

        # reclass the original CLC 100m raster layer
        r.reclass   "$SRC_DATA"   out=tmpmap   rules=$out_loc/reclass.rule   --
overwrite

        # resample tmpmap to 1 km, summing the pixels (= hectares)
        echo "Resampling CLC data.."
        r.resamp.stats tmpmap out=${SRC}_c$c method=sum --overwrite

        # smoothen in the given radius, use weighting factors
        echo "Smoothening in a ${N}x${N} neighbourhood.."
        r.neighbors       ${SRC}_c$c       out=${SRC}_c${c}_smth       size=$N
weight=$out_loc/weights.file method=sum --overwrite

        # calculate relative weight (density indicator), apply LAND mask
        echo "Calculating density indicator, in percent.."

        # Apply LAND mask according to on-the-fly NN resampling
        r.mapcalc "${SRC}_c${c}_smth_pct = ${SRC}_LAND == 0 ? null() : 100 *
${SRC}_c${c}_smth / ${SRC}_c0_smth"

        # Apply land mask according to 50% limit
        # r.mapcalc "${SRC}_c${c}_smth_pct = ${SRC}_c0 < 50 ? null() : 100 *
${SRC}_c${c}_smth / ${SRC}_c0_smth"

        # How the ETC seems to mask (checked with CORILIS based on CLC v9
(corilis00_r5l3_c2.tif)
        # r.mapcalc "${SRC}_c${c}_smth_pct = ${SRC}_c0 == 0 ? null() : 100 *
${SRC}_c${c}_smth / ${SRC}_c0_smth"

        r.support   "${SRC}_c${c}_smth_pct"   title="Density   indicator,   in
percent"

        # export into GeoTIFF format
```

```
       echo  "Exporting density indicator, in GeoTIFF format.."
       r.mapcalc                        "${SRC}_c${c}_smth_pct_rnd           =
round(${SRC}_c${c}_smth_pct)"
       r.support     "${SRC}_c${c}_smth_pct_rnd"  title="Density  indicator,
rounded percent"
       r.colors   "${SRC}_c${c}_smth_pct_rnd" rules=$out_loc/grey100.rule
       r.out.gdal                                    "${SRC}_c${c}_smth_pct_rnd"
out="$out_loc/${SRC}_R${R}_C${c}.tif"    type=Byte    createopt=compress=lzw
nodata=255.0

       echo "Cleaning up.."
       echo "Final time: $(date)"
       g.remove
tmpmap,${SRC}_c$c,${SRC}_c${c}_smth,${SRC}_c${c}_smth_pct_rnd,${SRC}_c${c}_
smth_pct
done

exit 0
```

2 ANNEX II – DEFINE PROJECTION TOOL

2.1 DEFINE PROJECTION BACKGROUND

QGIS is used as intermediate software to generate the location and the mapset in Grass.

In QGIS, the definition of the ETRS89/ETRS-LAEA (EPSG: 3035) is defined in proj4 format:

+proj=laea +lat_0=52 +lon_0=10 +x_0=4321000 +y_0=3210000 +ellps=GRS80 +units=m +no_defs

In Grass, the reference system of the location is stored in the PERMANENT Mapset, inside the file PROJ_INFO (location/PERMANENT).

At the moment to create a new Grass location, QGIS ask you to select the SRS (ETRS89/ETRS-LAEA (EPSG: 3035), if afterwards the file (*Location/*PERMANENT/PROJ_INFO) is opened, this will be the information that will be shown:

name: Lambert Azimuthal Equal Area
proj: laea
ellps: grs80
lat_0: 52
lon_0: 10
x_0: 4321000
y_0: 3210000
no_defs: defined

The command to query about the region in grass is: g.region –p (note the extent is not real).

projection: 99 (Lambert Azimuthal Equal Area)

```
zone:       0
datum:      ** unknown (default: WGS84) **
```

ellipsoid: grs80

```
north:      100000
south:      -100000
west:       -100000
east:       100000
nsres:      200
ewres:      200
rows:       1000
cols:       1000
cells:      1000000
```

It is possible to modify the datum: "unknown (default: WGS84)" by datum: etrs89 by using an ESRI prj file.
It is needed to modify it inside the PERMANENT mapset using the command g.proj.

Note: to change quickly of mapset you can use the command g.gisenv set=MAPSET = PERMANENT

Once you are in the PERMANENT MAPSET it is possible to change the SRS of the location using the wkt option: "ASCCII file that contains a description of the WKT projection".

- c : Create new projection files (modifies current location unless 'location' option specified)

g.proj -c wkt=d:/ETRS_1989_LAEA.prj

- If now we see the projection, the datum is etrs89.

```
PROJ_INFO----------------------------------------------
name       : Lambert Azimuthal Equal Area
proj       : laea
datum      : etrs89
ellps      : grs80
lat_0      : 52
lon_0      : 10
x_0        : 4321000
y_0        : 3210000
no_defs    : defined
```

- PROJ_UNITS----------

```
unit       : Meter
units      : Meters
meters     : 1
```

Now this is the SRS Definition for all the mapsets of the location.

If you want that at the moment to import a raster in the MAPSET to set its projection as the location projection, it is needed to set the parameter –o in the import command:

- r.in.gdal –o

Now the input data and the projection are in the suitable srs.

Export data

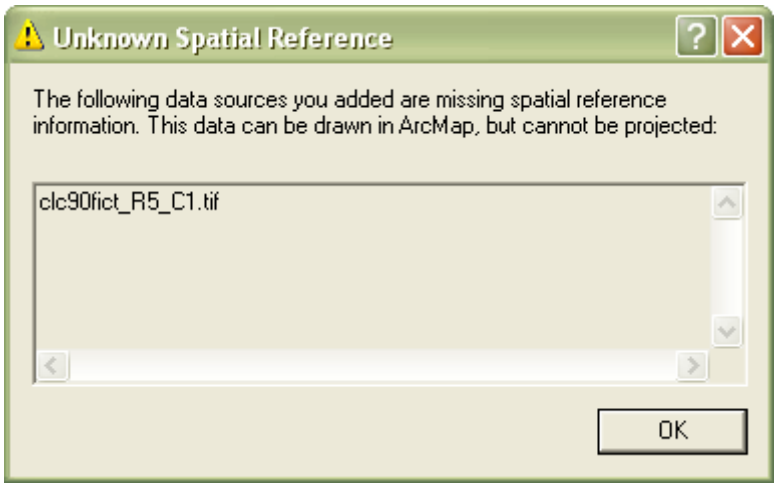
To export raster data to GTiff the command r.out.gdal is used, if a gdalinfo command on the output raster is used afterwards:

ex: gdalinfo d:/test.tif

The definition of the srs is correct:

PROJCS[”Lambert Azimuthal Equal Area”,
GEOGCS[”grs80”,
DATUM[”European_Terrestrial_Reference_System_1989”,
SPHEROID[”GRS 1980”,6378137,298.2572221010002,
AUTHORITY[”EPSG”,”7019”]],
AUTHORITY[”EPSG”,”6258”]],
PRIMEM[”Greenwich”,0],
UNIT[”degree”,0.0174532925199433]],
PROJECTION[”Lambert_Azimuthal_Equal_Area”],
PARAMETER[”latitude_of_center”,52],
PARAMETER[”longitude_of_center”,10],
PARAMETER[”false_easting”,4321000],
PARAMETER[”false_northing”,3210000],
UNIT[”metre”,1,
AUTHORITY[”EPSG”,”9001”]]]

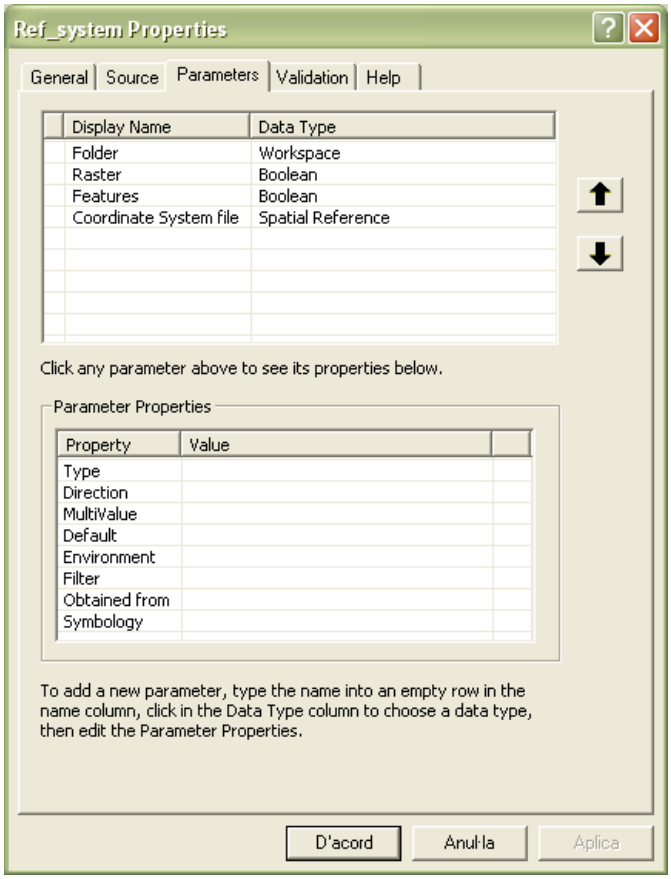
However, if we open the exported raster with ArGIS there is no definition of reference system (nor ArcCatalog neither ArcMap).



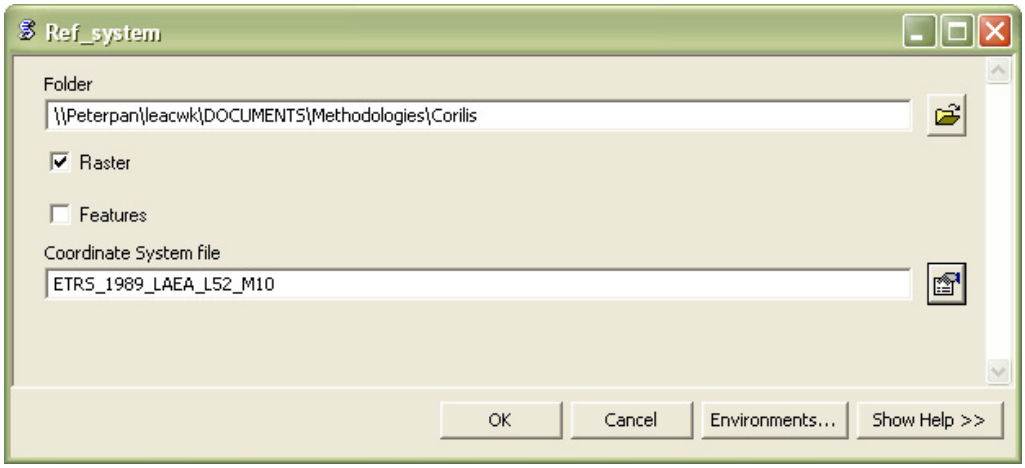
For that reason the best option is to create a script in ArGIS to define the SRS.

2.1.1 Define projection tool

- Open the Ref_system tool in LEAC toolbox (LEAC/Define Folder ref_system/Ref_system)
- The Ref_system tool has 4 parameters, as it can be seen in the next figure.



- **Folder:** the folder where the corilis level 3 rasters (.tif) are stored.
- **Raster or Features:** the tool has been thought to define the same projection for all the rasters or feature classes inside a folder
- **Coordinate System**: The coordinate system to be applied.

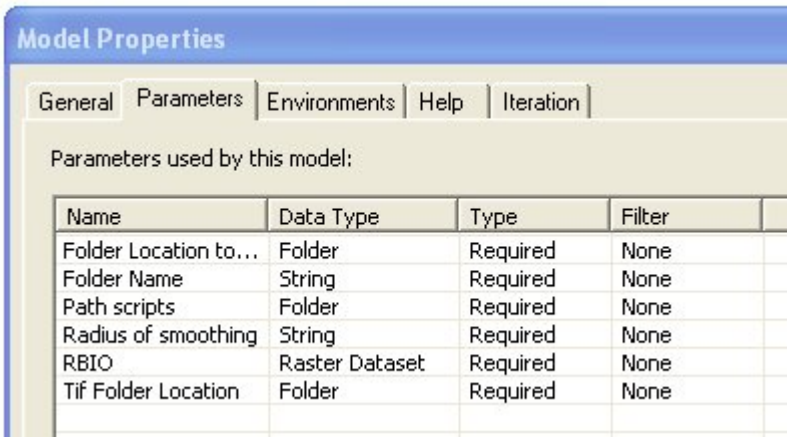


2.1.2 Define projection script

```
#  ---------------------------------------------------------------------------
--
# d_proi.py
# Created on: mar ago 31 2010 11:08:40
#   (generated by ArcGIS/ModelBuilder)
#  ---------------------------------------------------------------------------
--

# Import system modules
import sys, string, os, arcgisscripting

# Create the Geoprocessor object
gp = arcgisscripting.create(9.3)

# Load required toolboxes...
gp.workspace=sys.argv[1]
gp.AddMessage(gp.workspace)

raster= sys.argv[2]
gp.AddMessage(raster)
features = sys.argv[3]
gp.AddMessage(features)
Coordinate_System=sys.argv[4]

if raster=="true":
    rcs = gp.ListRasters()
    for rc in rcs:
# Local variables...
        gp.AddMessage(rc)
        gp.DefineProjection_management(rc, Coordinate_System)

if features=="true":
    fcs = gp.ListFeatureClasses()
    for fc in fcs:
# Local variables...
        gp.AddMessage(fc)
        gp.DefineProjection_management(fc, Coordinate_System)

gp.workspace=""
```

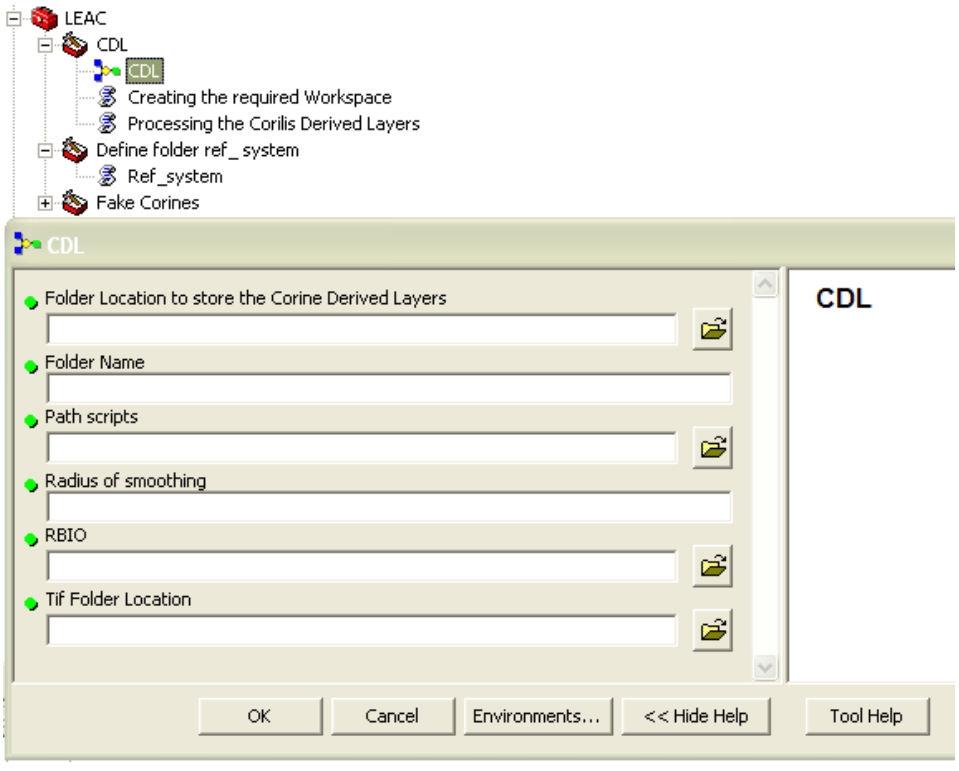
3 ANNEX III – CORILIS DERIVED LAYERS STEPS

1. Open the CDL tool (it is a Model) in LEAC toolbox: (LEAC/CDL/CDL)



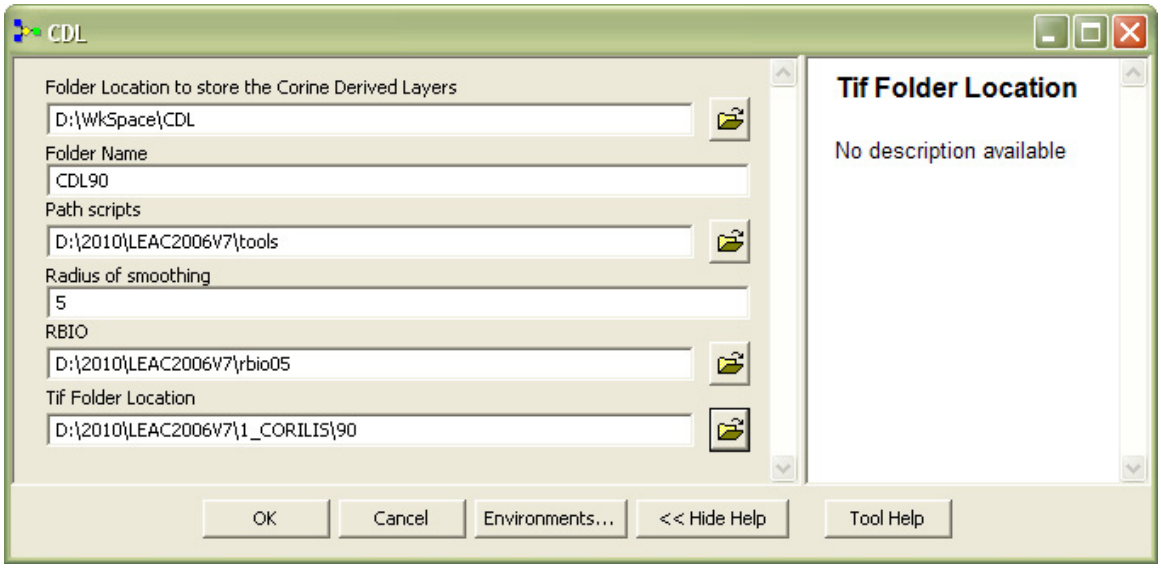
Input paramenters:

- ✓ Folder location to store the CDL layers
- ✓ Folder name (where the CDL files will be stored)
- ✓ Path to the scripts
- ✓ Radius of smoothing
- ✓ RBIO (Biogeographic Regions dataset – proximity)
- ✓ Tif folder location: the path to the corilis layers.



2. Fill in the parameters details:

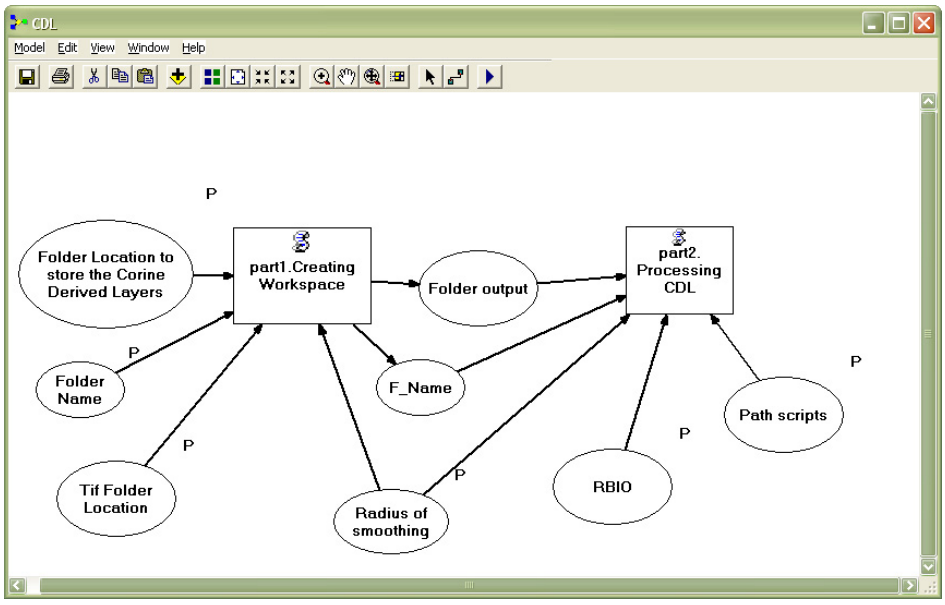
Screenshot example for CDL 90



Output data:
(each folder contains 5 datasets dlt, gbli, level1, level3 and wetlands)
90
00
06

```
Executed (part1.Creating Workspace) successfully.
End Time: Mon Sep 12 18:20:33 2011 (Elapsed Time: 5 minutes 36 seconds)

Completed script PART2...
Executed (part2. Processing CDL) successfully.
End Time: Tue Sep 13 09:32:57 2011 (Elapsed Time: 15 hours 12 minutes 24
seconds)

Executed (Model) successfully.
End Time: Tue Sep 13 09:32:57 2011 (Elapsed Time: 15 hours 18 minutes 0
seconds)
```

CDL model diagram:



3.1.1 CDL Scripts

3.1.1.1 PART1.PY

```
# ---------------------------------------------------------------------------
# part1.py
# Created on: lun jul 05 2010 12:58:42
#   (generated by ArcGIS/ModelBuilder)
# ---------------------------------------------------------------------------

# Import system modules
import sys, string, os, arcgisscripting

# Create the Geoprocessor object
gp = arcgisscripting.create(9.3)

Folder_Location0 = sys.argv[1]
Folder_Name0 = sys.argv[2]
gp.workspace = sys.argv[3]
radi = sys.argv[4]

#Create folder structure
gp.CreateFolder_management(Folder_Location0, Folder_Name0)
Folder_Location= Folder_Location0 + "\\" + Folder_Name0 + "\\"
Folder_Name= "level3"
Path_Folder = Folder_Location + Folder_Name
gp.CreateFolder_management(Folder_Location, Folder_Name)

# Get a list of grids in the workspace.
rasters = gp.ListRasters("", "TIF")
stringcerca = "_R" + radi + "_C"
gp.AddMessage(sys.argv[1])
gp.AddMessage(sys.argv[2])
gp.AddMessage(sys.argv[3])
gp.AddMessage(sys.argv[4])

for raster in rasters:
    num = raster.find(stringcerca)+ 5
    num2 = raster.find(„.")
    clase = raster[num:num2]
    gp.AddMessage(clase)
    new_name = "r" + radi + "sm" + clase
    gp.AddMessage(new_name)
    raster_input = gp.workspace + "\\" + raster
    gp.AddMessage(raster_input)
    raster_output= Path_Folder + "\\" + new_name
    gp.AddMessage(raster_output)
    gp.Copy_management(raster_input, raster_output, "RasterDataset")
    gp.RasterToOtherFormat_conversion(raster_output + ".tif", Path_Folder, "GRID")
    Delete_succeeded = raster_output
    gp.Delete_management(raster_output + ".tif" , "RasterDataset")
 gp.AddMessage(Folder_Location)
```

3.1.1.2 g_level1.aml

```
&args p0 s R z p2
w d:/
&s p1 = %p0%\%s%\

w %p1%level3
createworkspace ..\level1
createworkspace ..\gbli
createworkspace ..\wetlands
createworkspace ..\dlt
grid
..\level1\c1 =
SUM(r%R%sm1,r%R%sm2,r%R%sm3,r%R%sm4,r%R%sm5,r%R%sm6,r%R%sm7,r
%R%sm8,r%R%sm9,r%R%sm10,r%R%sm11)
..\level1\c2a =
SUM(r%R%sm12,r%R%sm13,r%R%sm14,r%R%sm15,r%R%sm16,r%R%sm17,r%R
%sm19)
..\level1\c2b1 = SUM(r%R%sm18)
..\level1\c2b2 = SUM(r%R%sm20,r%R%sm21,r%R%sm22)
..\level1\c3a1 = SUM(r%R%sm23,r%R%sm24,r%R%sm25)
..\level1\c3a2 = SUM(r%R%sm29)
..\level1\c3b = SUM(r%R%sm26,r%R%sm27,r%R%sm28)
..\level1\c3c = SUM(r%R%sm30,r%R%sm31,r%R%sm32,r%R%sm33,r%R%sm34)
..\level1\c4 = SUM(r%R%sm35,r%R%sm36,r%R%sm37,r%R%sm38,r%R%sm39)
..\level1\c5 = SUM(r%R%sm40,r%R%sm41,r%R%sm42,r%R%sm43)
q
&run %p2%\g_gbli.aml %p1% %R% %z% %p2%
```

3.1.1.3 g_gbli.aml

```
&args p1 R z p2

W %p1%level1

grid
..\gbli\ebg05km = SUM(c2b1,c2b2,c3a1,c3a2,c3b,c3c,c4,c5)
q
&run %p2%\g_wetlands.aml %p1% %R% %z% %p2%
```

3.1.1.4 g_wetlands.aml

```
&args p1 R z p2

w %p1%level3

GRID
..\wetlands\wet =
SUM(r5sm14,r5sm35,r5sm36,r5sm37,r5sm38,r5sm39,r5sm42,r5sm43)
Q
&run %p2%\CALCDLT.aml %p1% %R% %z% %p2%
```

3.1.1.5.CALCDLT.aml

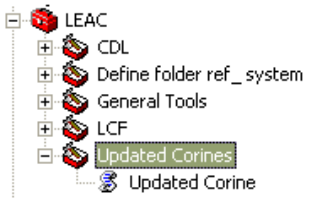
```
&args p1 R z p2

grid
/* Grouping into main LC classes
..\dlt\dlt1 =
SUM(r%R%sm1,r%R%sm2,r%R%sm3,r%R%sm4,r%R%sm5,r%R%sm6,r%R%sm7,r
%R%sm8,r%R%sm9,r%R%sm10,r%R%sm11)
..\dlt\dlt2 =
SUM(r%R%sm12,r%R%sm13,r%R%sm14,r%R%sm15,r%R%sm16,r%R%sm17,r%R
%sm19)
..\dlt\dlt3 = SUM(r%R%sm18,r%R%sm20,r%R%sm21,r%R%sm22)
..\dlt\dlt4 = SUM(r%R%sm23,r%R%sm24,r%R%sm25,r%R%sm29)
..\dlt\dlt5 =
SUM(r%R%sm26,r%R%sm27,r%R%sm28,r%R%sm30,r%R%sm31,r%R%sm32,r%R
%sm33,r%R%sm34,r%R%sm35,r%R%sm36,r%R%sm37,r%R%sm38,r%R%sm39,r
%R%sm40,r%R%sm41,r%R%sm42,r%R%sm43)
q
w ..\dlt
grid
/* Creating boolean layers where SMOOTHED VALUE > MEAN + STDV in a given zone
&do i := 1 &to 5
        cond%i% = con(dlt%i% > sum(zonalstd(%z%, dlt%i%), zonalmean(%z%,
dlt%i%)), 1, 0)
&end
/* Computing the TOTAL layer used in the next step
total = sum(cond1,cond2,cond3,cond4,cond5)

/* Computing DLT based on conditional queries
r%R%dlt = con(total == 1 and cond1 == 1, 1,con(total > 1 and cond1 == 1,
2,con(total == 1 and cond2 == 1, 3,con(cond1 == 0 and cond3 == 1, 4,con(cond1
== 0 and cond3 == 0 and cond4 == 1, 5,con(cond1 == 0 and cond3 == 0 and cond4
== 0 and cond5 == 1, 6,con(total == 0, 7, 9999)))))))

/* Deleting temporary files
&do i := 1 &to 5
        kill cond%i% all
        kill dlt%i% all
&end
kill total all
q
```

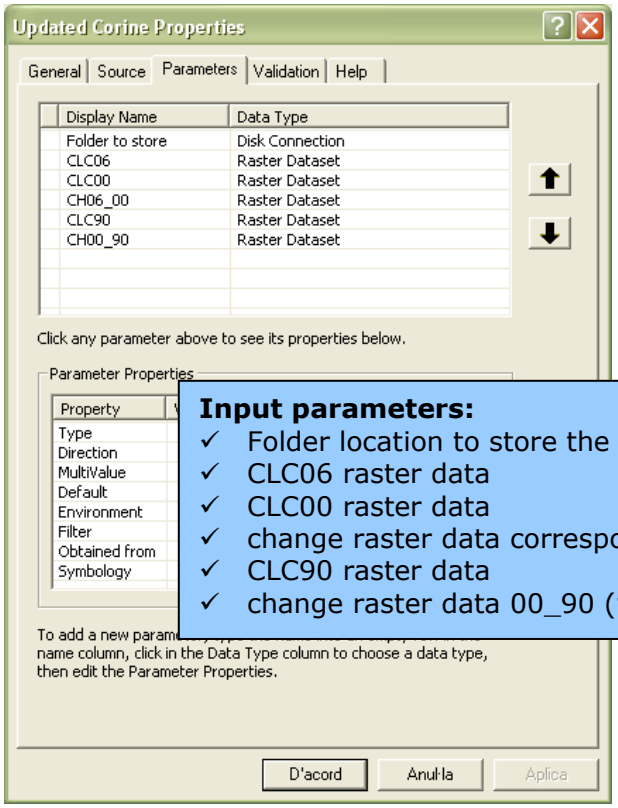
4 ANNEX IV – UPDATED CORINE STEPS

4.1 UPDATED CORINE TOOL

Very important: *Check nodata values.*
Note that in cases g100_00 and g100_06 nodata value is 49

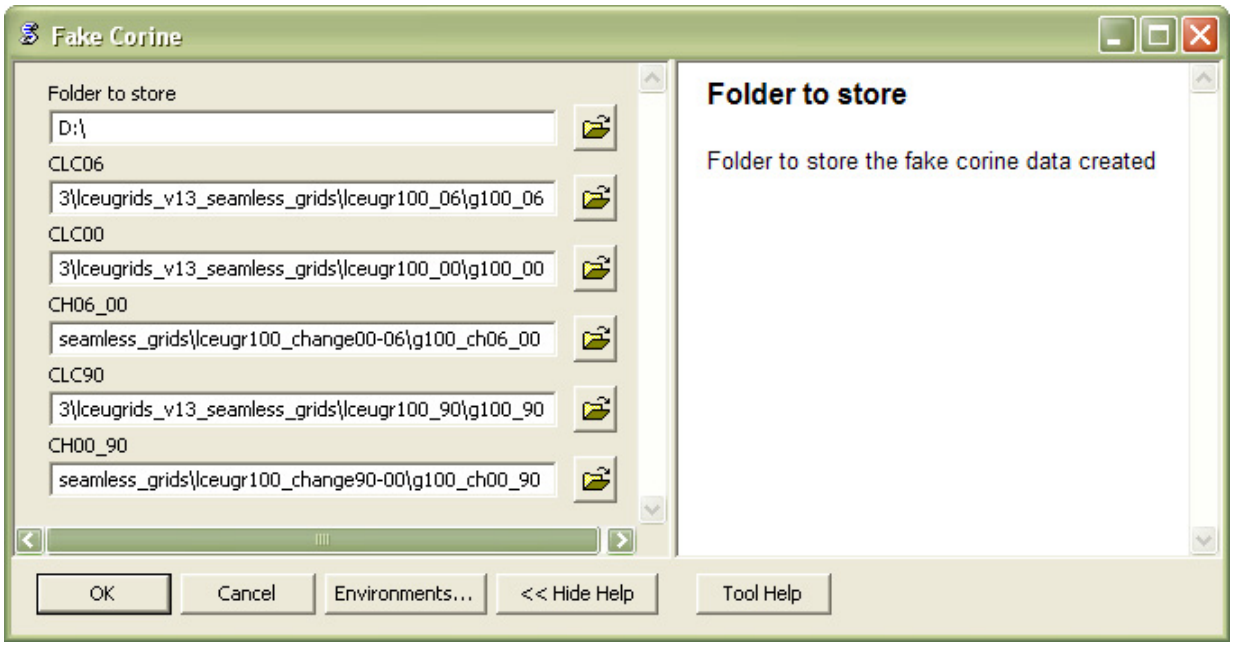
- Open the Fake Corine tool in LEAC toolbox





Input parameters:
- ✓ Folder location to store the Updated Corine data created
- ✓ CLC06 raster data
- ✓ CLC00 raster data
- ✓ change raster data corresponding to 06_00 (values of 00)
- ✓ CLC90 raster data
- ✓ change raster data 00_90 (with values corresponding to 90)

- Fill in the required parameters:



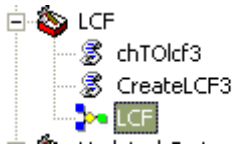
4.2 UPDATED CORINE LAND COVER SCRIPTS

```
# ---------------------------------------------------------------------------
# fk_clc00a.py
# Created on: lun sep 20 2010 05:08:43
#   (generated by ArcGIS/ModelBuilder)
# ---------------------------------------------------------------------------

# Import system modules
import sys, string, os, arcgisscripting

# Create the Geoprocessor object
gp = arcgisscripting.create(9.3)

# Check out any necessary licenses
gp.CheckOutExtension("spatial")

# Process: creating folders
Folder_Location0 = sys.argv[1]
Folder_Name0 = "UpdateCLC"
gp.CreateFolder_management(Folder_Location0, Folder_Name0)

Folder_Location1= Folder_Location0 + "\\" + Folder_Name0 + "\\"
Folder_Name1= "FK00"
Folder_Name2= "FK90"

gp.CreateFolder_management(Folder_Location1, Folder_Name1)
gp.CreateFolder_management(Folder_Location1, Folder_Name2)

Folder_Location2= Folder_Location1 + "\\" + Folder_Name1 + "\\"
Folder_Name3= "Int"
Folder_Name4= "Output"
```

```
gp.CreateFolder_management(Folder_Location2, Folder_Name3)
gp.CreateFolder_management(Folder_Location2, Folder_Name4)

Folder_Location3= Folder_Location1 + "\\" + Folder_Name2 + "\\"
gp.CreateFolder_management(Folder_Location3, Folder_Name3)
gp.CreateFolder_management(Folder_Location3, Folder_Name4)

# Process: fake 00
clc06 = sys.argv[2]
clc00 = sys.argv[3]
ch06_00 = sys.argv[4]
fk00_int = Folder_Location2 + "\\" + Folder_Name3 + "\\fk00_int"
Map_expression_1 = "con(IsNull(" + ch06_00 + "),"" + clc06 + "," + ch06_00 + ")"
Map_expression_2 = "con(IsNull(" + fk00_int + "),"" + clc00 + "," + fk00_int + ")"
fk00 = Folder_Location2 + "\\" + Folder_Name4 + "\\fk00"

gp.AddMessage("fake 00 intermedi")
gp.SingleOutputMapAlgebra_sa(Map_expression_1, fk00_int, "")
gp.AddMessage("fake 00 final")
gp.SingleOutputMapAlgebra_sa(Map_expression_2, fk00, "")

# Process: mask creation
clc90 = sys.argv[5]
ch00_90 = sys.argv[6]
mask90 = Folder_Location3 + "\\" + Folder_Name3 + "\\mask90"
Map_expression_3 = "\"VALUE\" = 49"
gp.AddMessage("mask creation")
gp.AddMessage(clc90 +" "+ mask90 )
gp.SetNull_sa(clc90, "1", mask90, Map_expression_3)

# Process: fake 90 without mask
gp.AddMessage("fake 90 without mask")
fk90_nm = Folder_Location3 + "\\" + Folder_Name3 + "\\fk90_nm"
Map_expression_4 = "con(IsNull(" + ch00_90 + "),"" + fk00 + "," + ch00_90 + ")"
gp.SingleOutputMapAlgebra_sa(Map_expression_4, fk90_nm, "")

# Process: fake 90 with mask
gp.AddMessage("fake 90 with mask")
gp.AddMessage("fake 90 with mask intermediate")
fk90_mk_int = Folder_Location3 + "\\" + Folder_Name3 + "\\fk90_mk_int"
gp.Times_sa(fk90_nm, mask90, fk90_mk_int)

gp.AddMessage("fake 90 with mask final")
fk90 = Folder_Location3 + "\\" + Folder_Name4 + "\\fk90"
Map_expression_6 =  "\"VALUE\" = 44"
gp.Con_sa(clc90, « 44 », fk90, fk90_mk_int, Map_expression_6)
```

5 ANNEX V – LAND COVER FLOWS

1- Open the Tool LCF (Model) from the LEAC Toolbox: LEAC/LCF



The model joins to scripts:

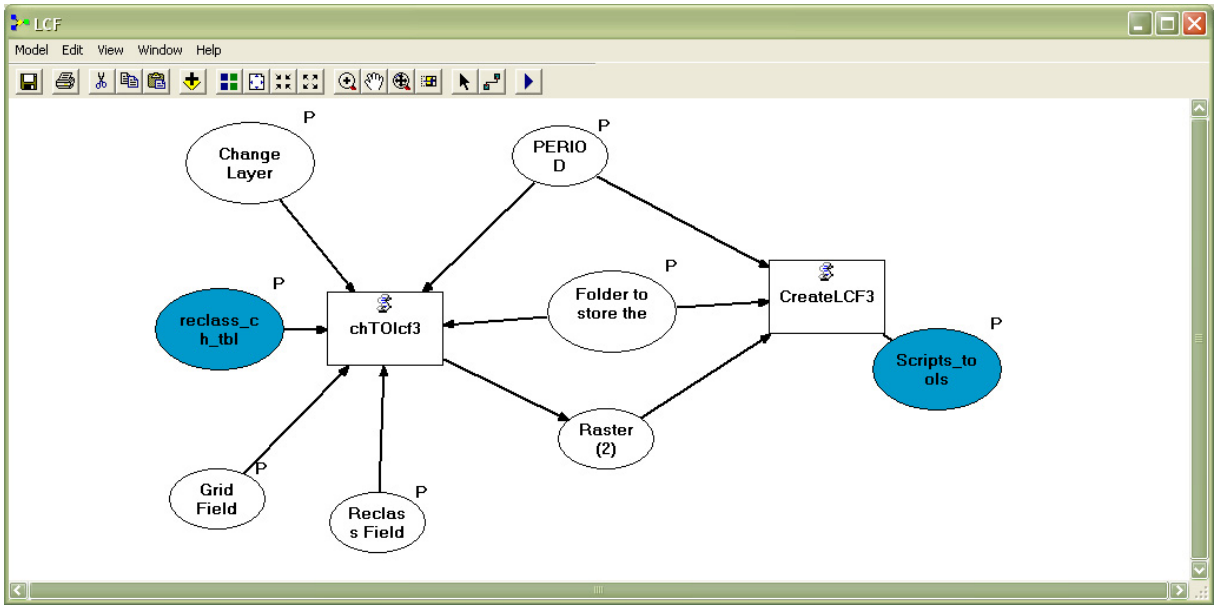
a) chTOLcf3: script in python that reclassifies the Corine Land Cover Changes XXYY layer by replacing the gridcode (1-1937) by the LCF code 3.

b) CreateLCF3:

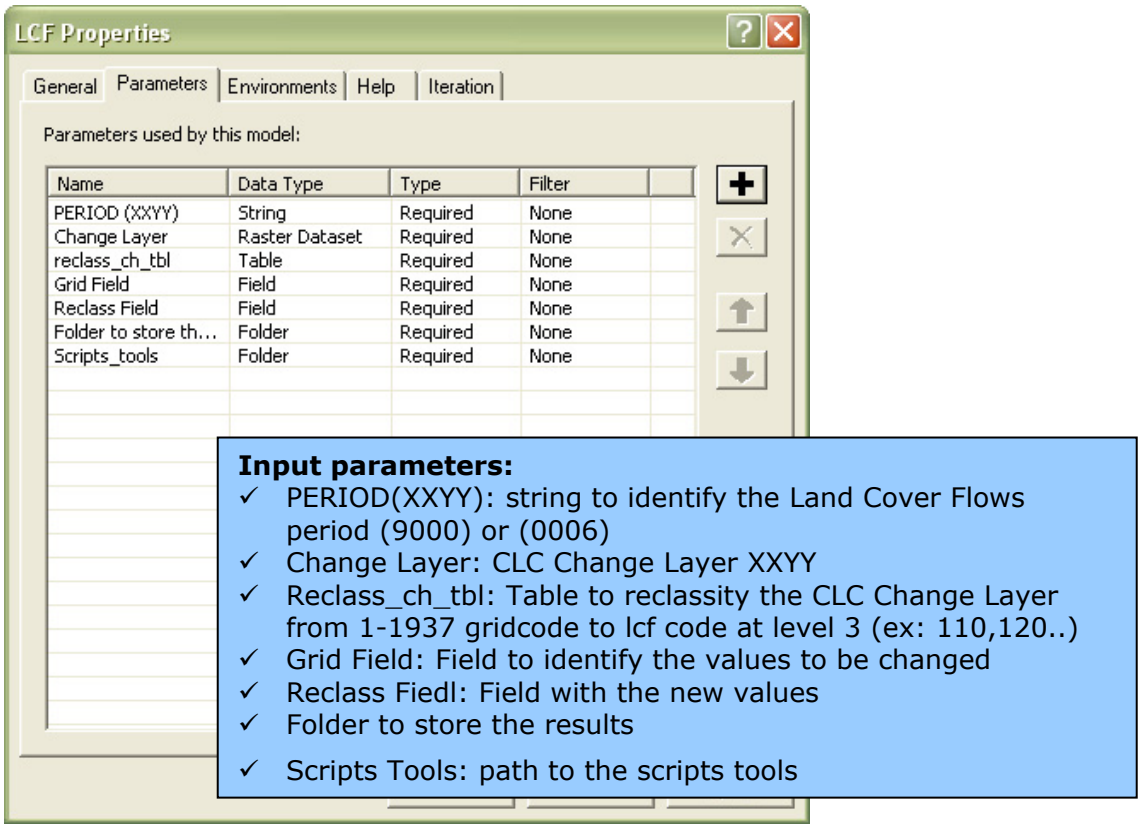
   a. CreateLCF1.aml script that creates the level3 folder with the lcf at level 3

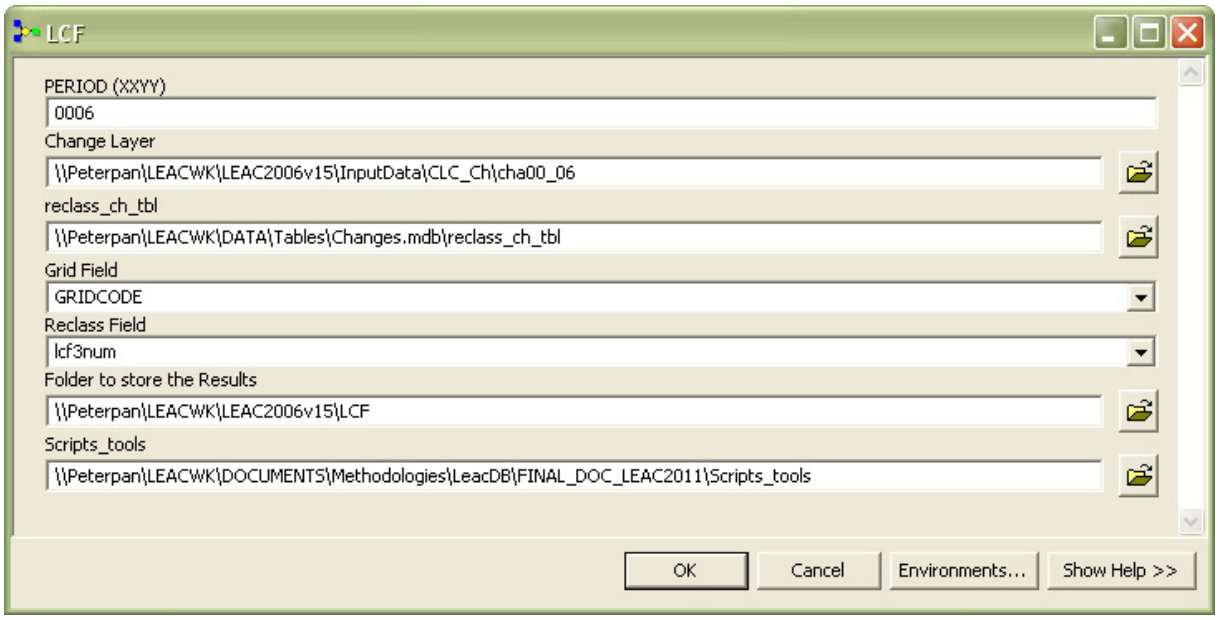
   b. Call the second aml script GROUP_LCF2.aml that creates and generates the folders and lcf at level 2 and level1.

The next figure shows the model diagram:



The parameters to be introduced are:





5.1 LAND COVER FLOWS SCRIPTS

5.1.1 Ch2lcf3.py

```
# ---------------------------------------------------------------------------
# ch2lcf3.py
# Created on: lun sep 19 2011 10:16:35
#   (generated by ArcGIS/ModelBuilder)
# ---------------------------------------------------------------------------
# Import system modules
import sys, string, os, arcgisscripting
# Create the Geoprocessor object
gp = arcgisscripting.create(9.3)
# System arguments...
PERIOD = sys.argv[1] #LCF period XXYY (9000 or 0006)
CH_LAYER = sys.argv[2] #change raster for the period XXYY
reclass_ch_tbl = sys.argv[3] #change reclass table (default:\\Peterpan\LEACWK\DATA\Tables\Changes.mdb\reclass_ch_tbl)
Grid_Field = sys.argv[4]
Reclass_Field = sys.argv[5]
folder_results = sys.argv[6] #folder to store the created LCF
gp.AddMessage(sys.argv[1])
gp.AddMessage(sys.argv[2])
gp.AddMessage(sys.argv[3])
gp.AddMessage(sys.argv[4])
gp.AddMessage(sys.argv[5])
gp.AddMessage(sys.argv[6])


# Local variables...
lcf100_XXYY = folder_results + "\\" + "lcf100_" + PERIOD #lcf 100m resolution for XXYY period


# Process: Reclass by Table...
Results = gp.ReclassByTable_sa(CH_LAYER , reclass_ch_tbl, Grid_Field, Grid_Field, Reclass_Field, lcf100_XXYY, "DATA")


# Set parameters...
gp.AddMessage(Results.GetOutput(0))
gp.SetParameter(6, Results.GetOutput(0))
```

5.1.2 CreateLCF3 (CreateLCF1.aml)

```
&args LCF_XXYY lcf_folder lcf100_XXYY pathscripts
w %lcf_folder%
echo "%LCF_XXYY%"
echo "%lcf_folder%"
echo "%lcf100_XXYY%"
echo "%pathscripts%"

createworkspace .\%LCF_XXYY%
createworkspace .\%LCF_XXYY%\level3
grid
.\%LCF_XXYY%\level3\lcf110 = aggregate(%lcf100_XXYY% == 110, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf120 = aggregate(%lcf100_XXYY% == 120, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf130 = aggregate(%lcf100_XXYY% == 130, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf210 = aggregate(%lcf100_XXYY% == 210, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf220 = aggregate(%lcf100_XXYY% == 220, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf310 = aggregate(%lcf100_XXYY% == 310, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf320 = aggregate(%lcf100_XXYY% == 320, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf330 = aggregate(%lcf100_XXYY% == 330, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf340 = aggregate(%lcf100_XXYY% == 340, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf350 = aggregate(%lcf100_XXYY% == 350, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf360 = aggregate(%lcf100_XXYY% == 360, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf370 = aggregate(%lcf100_XXYY% == 370, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf380 = aggregate(%lcf100_XXYY% == 380, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf411 = aggregate(%lcf100_XXYY% == 411, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf412 = aggregate(%lcf100_XXYY% == 412, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf421 = aggregate(%lcf100_XXYY% == 421, 10, SUM, TRUNCATE, DATA)
```

```
.\%LCF_XXYY%\level3\lcf422 = aggregate(%lcf100_XXYY% == 422, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf431 = aggregate(%lcf100_XXYY% == 431, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf432 = aggregate(%lcf100_XXYY% == 432, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf433 = aggregate(%lcf100_XXYY% == 433, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf441 = aggregate(%lcf100_XXYY% == 441, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf442 = aggregate(%lcf100_XXYY% == 442, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf443 = aggregate(%lcf100_XXYY% == 443, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf444 = aggregate(%lcf100_XXYY% == 444, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf451 = aggregate(%lcf100_XXYY% == 451, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf452 = aggregate(%lcf100_XXYY% == 452, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf453 = aggregate(%lcf100_XXYY% == 453, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf461 = aggregate(%lcf100_XXYY% == 461, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf462 = aggregate(%lcf100_XXYY% == 462, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf463 = aggregate(%lcf100_XXYY% == 463, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf470 = aggregate(%lcf100_XXYY% == 470, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf511 = aggregate(%lcf100_XXYY% == 511, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf512 = aggregate(%lcf100_XXYY% == 512, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf521 = aggregate(%lcf100_XXYY% == 521, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf522 = aggregate(%lcf100_XXYY% == 522, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf523 = aggregate(%lcf100_XXYY% == 523, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf530 = aggregate(%lcf100_XXYY% == 530, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf540 = aggregate(%lcf100_XXYY% == 540, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf610 = aggregate(%lcf100_XXYY% == 610, 10, SUM, TRUNCATE, DATA)
```

```
.\%LCF_XXYY%\level3\lcf620 = aggregate(%lcf100_XXYY% == 620, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf710 = aggregate(%lcf100_XXYY% == 710, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf720 = aggregate(%lcf100_XXYY% == 720, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf730 = aggregate(%lcf100_XXYY% == 730, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf740 = aggregate(%lcf100_XXYY% == 740, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf810 = aggregate(%lcf100_XXYY% == 810, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf820 = aggregate(%lcf100_XXYY% == 820, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf911 = aggregate(%lcf100_XXYY% == 911, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf912 = aggregate(%lcf100_XXYY% == 912, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf913 = aggregate(%lcf100_XXYY% == 913, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf920 = aggregate(%lcf100_XXYY% == 920, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf930 = aggregate(%lcf100_XXYY% == 930, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf940 = aggregate(%lcf100_XXYY% == 940, 10, SUM, TRUNCATE, DATA)
.\%LCF_XXYY%\level3\lcf990 = aggregate(%lcf100_XXYY% == 990, 10, SUM, TRUNCATE, DATA)
q
&run %pathscripts%\GROUP_LCF2.aml %lcf_folder%\%LCF_XXYY%
```

5.1.3 GROUP_LCF2.aml

```
&args wksp
echo "inside lcf2"
echo "%wksp%\level3"

w %wksp%\level3
createworkspace ..\level2
copy lcf110 ..\level2\lcf11
copy lcf120 ..\level2\lcf12
copy lcf130 ..\level2\lcf13
copy lcf210 ..\level2\lcf21
copy lcf220 ..\level2\lcf22
copy lcf310 ..\level2\lcf31
copy lcf320 ..\level2\lcf32
copy lcf330 ..\level2\lcf33
copy lcf340 ..\level2\lcf34
copy lcf350 ..\level2\lcf35
copy lcf360 ..\level2\lcf36
copy lcf370 ..\level2\lcf37
copy lcf380 ..\level2\lcf38
copy lcf470 ..\level2\lcf47
copy lcf530 ..\level2\lcf53
copy lcf540 ..\level2\lcf54
copy lcf610 ..\level2\lcf61
copy lcf620 ..\level2\lcf62
copy lcf710 ..\level2\lcf71
copy lcf720 ..\level2\lcf72
copy lcf730 ..\level2\lcf73
copy lcf740 ..\level2\lcf74
copy lcf810 ..\level2\lcf81
copy lcf820 ..\level2\lcf82
copy lcf920 ..\level2\lcf92
copy lcf930 ..\level2\lcf93
copy lcf940 ..\level2\lcf94
copy lcf990 ..\level2\lcf99
grid
..\level2\lcf41 = sum(lcf411,lcf412)
..\level2\lcf42 = sum(lcf421,lcf422)
..\level2\lcf43 = sum(lcf431,lcf432,lcf433)
..\level2\lcf44 = sum(lcf441,lcf442,lcf443, lcf444)
..\level2\lcf45 = sum(lcf451,lcf452,lcf453)
..\level2\lcf46 = sum(lcf461,lcf462,lcf463)
..\level2\lcf51 = sum(lcf511,lcf512)
..\level2\lcf52 = sum(lcf521,lcf522)
..\level2\lcf91 = sum(lcf911,lcf912,lcf913)
q
w ..\level2
createworkspace ..\level1
grid
..\level1\lcf1 = sum(lcf11,lcf12,lcf13)
..\level1\lcf2 = sum(lcf21,lcf22)
..\level1\lcf3 = sum(lcf31,lcf32,lcf33,lcf34,lcf35,lcf36,lcf37,lcf38)
..\level1\lcf4 = sum(lcf41,lcf42,lcf43,lcf44,lcf45,lcf46,lcf47)
..\level1\lcf5 = sum(lcf51,lcf52,lcf53,lcf54)
..\level1\lcf6 = sum(lcf61,lcf62)
..\level1\lcf7 = sum(lcf71,lcf72,lcf73,lcf74)
..\level1\lcf8 = sum(lcf81,lcf82)
..\level1\lcf9 = sum(lcf91,lcf92,lcf93,lcf94,lcf99)
q
```

6 ANNEX VI – GENERAL TOOLS TOOLSET

It is available an executable installer for the General toolset toolbox, which contains the following tools:

- Combine To Table: combines several rasters and export the attribute table
- Create Grid1km (raster format): creates a GRID1KM in raster format, using the LEAC numeric cellcode
- Create GridCode Translation table: creates a translation table from LEAC numeric cellcode to European reference cell codes.

See the detailed description bellow, or check the ArcGIS integrated help.

TITLE Combine to table

Summary

Combines a set of input rasters (using ArcGIS "Combine"process) and exports the results to a table named "LEAC" in the provided database(file or personal geodatabase).The database should not exist previously, as it will be created in the process.
Input rasters are splitted in sub-zones when combine is applied, so that big raster can be correctly processed.

Syntax

CombineToTable (Input_Rasters, Output_Database, Combine_Cell_Size)

Parameter	Explanation	Data Type
Input_Rasters	**Dialog Reference** A set of input rasters to be combined. There is no python reference for this parameter.	Multiple Value
Output_Database	**Dialog Reference** The name of the geodatabase (file or personal) to hold the exported table. The database will be created in the proces, so it should not exist previously. A table named "LEAC" will be created in the database to store the results of the combine operation. There is no python reference for this parameter.	File
Combine_Cell_Size	There is no explanation for this parameter.	Cell Size

TITLE Create Grid1km (raster format)

Summary

Creates a European reference GRID1KM in raster format. The cells are encoded using the new LEAC numeric cell encoding. A translation table to reference string codes ("1KME2342N3324") can be easily created using the "Create GRID1KM translation table" tool.
This tool is only available for ArcGIS 10.0, as NumPyArrayToRaster is not available on ArcGIS 9.3.
Note: several temporal rasters are created in the output directory, and they are not deleted afterwards (to help monitoring the tool behaviour).Ensure you delete them after successfully executing the tool.

Syntax

CreateGrid1kmRaster (Output_Raster, Cell_Size, Extent)

Parameter	Explanation	Data Type
Output_Raster	There is no explanation for this parameter.	Raster Dataset
Cell_Size	**Dialog Reference** The size (in metres) of the cells to be created (100m, 1km, 10km, etc). There is no python reference for this parameter.	Cell Size
Extent	**Dialog Reference** The extent of the GRID to be created. By default: [1500000, 900000, 7400000, 5500000]. There is no python reference for this parameter.	Extent

TITLE Create GridCode Translation Table

Summary

Creates a table containing the equivalence between the reference code (ex: "1kmE3264N4702") and the LEAC numeric code (32644702). The table will be named "gridCodes" and it will contain the fields "NUMCODE" and "REFCODE".

Syntax

CreateGridCodeTransTable (Output_Database, Cell_Size, Extent)

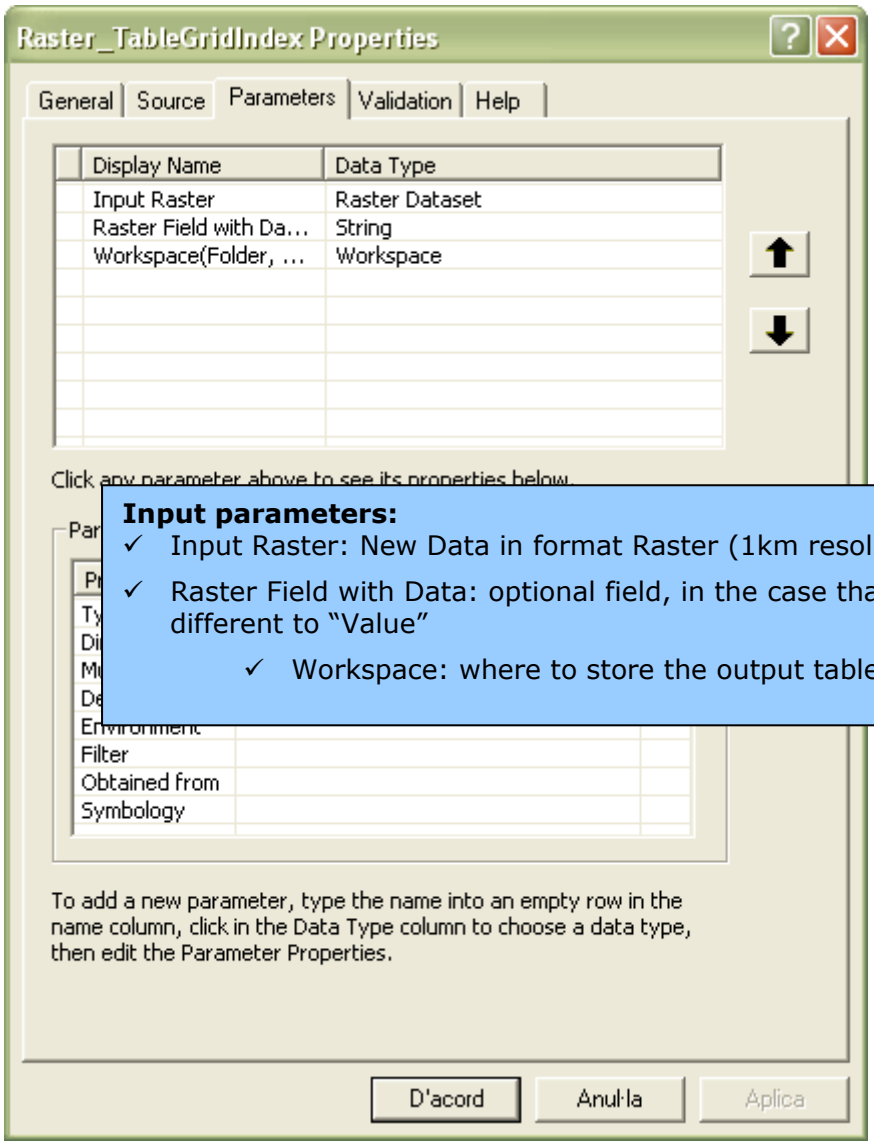
Parameter	Explanation	Data Type
Output_Database	**Dialog Reference** The name of the (file or personal) geodatabase to store the translation table. The database will be created in the proces, so it should not exist previously. The translation table will be named "gridCodes". There is no python reference for this parameter.	File
Cell_Size	**Dialog Reference** The size (in metres) of the cells of the GRID to be translated (100m, 1km, 10km, etc). The size is reflected in the lenght of the gridcodes. There is no python reference for this parameter.	Cell Size
Extent	**Dialog Reference** If provided, the translation table will only include cells that are located inside this extent.Otherwise, a default extent including all the EEA member countries [1500000, 900000, 7400000, 5500000] would be used. There is no python reference for this parameter.	Extent

7 ANNEX VII – RASTER TO TABLE GRID INDEX TOOL

7.1 RASTER TO TABLE GRID INDEX TOOL

1- Open the Raster_TableGridIndex tool (LEAC/NewLaru_Grid).

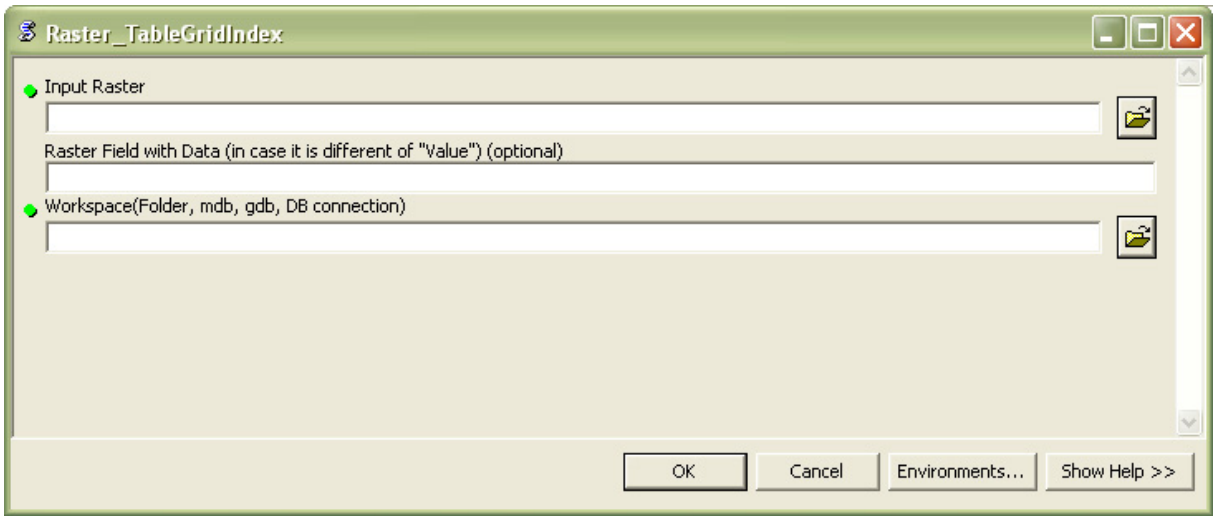
Each parameter of the extent of the input raster (top, left, right and Bottom) has to be a multiple of 1000 in order to ensure the alignment with the European Reference Grid



Input parameters:

- ✓ Input Raster: New Data in format Raster (1km resolution)
- ✓ Raster Field with Data: optional field, in the case that it is different to "Value"
- ✓ Workspace: where to store the output table

2- Fill in the required parameters



7.2 RASTER TO TABLE GRID INDEX TOOL SCRIPT (RASTER_TO_TABLEGRIDINDEX.PY)

```
# ---------------------------------------------------------------------------
# uno.py
# Created on: lun may 09 2011 11:14:49
#   (generated by ArcGIS/ModelBuilder)
# ---------------------------------------------------------------------------
# Import system modules
import sys, string, os, arcgisscripting
# Create the Geoprocessor object
gp = arcgisscripting.create(9.3)

raster = sys.argv[1]
rasterF = sys.argv[2]
work = sys.argv[3]
if (rasterF ==""):
   rasterF == "Value"

try:
   if gp.CheckExtension("spatial") == "Available":
         gp.CheckOutExtension("spatial")
   else:
         raise "LicenseError"
```

```
desc = gp.Describe(raster)
name = desc.Name
extent = desc.Extent

#YMAX = str(extent.YMAX)
YMIN = str(extent.YMIN)[(str(extent.YMIN).rfind(".")-3):str(extent.YMIN).rfind(".")]
YMAX = str(extent.YMAX)[(str(extent.YMAX).rfind(".")-3):str(extent.YMAX).rfind(".")]
XMIN = str(extent.XMIN)[(str(extent.XMIN).rfind(".")-3):str(extent.XMIN).rfind(".")]
XMAX = str(extent.XMAX)[(str(extent.XMAX).rfind(".")-3):str(extent.XMAX).rfind(".")]

CH = desc.MeanCellHeight
CW = desc.MeanCellWidth

if (YMIN!= "000") or (YMAX !="000")or (XMIN != "000") or (XMAX != "000")or (CH!= 1000) or (CW != 1000):
    if (CH != 1000) or (CW != 1000):
        gp.AddError("Cell size is different to 1000, it is: " + CH )
    else:
        gp.AddError("The extent is not multiple of 1000")
    sys.exit()

desc2 = gp.Describe(work)
if (desc2.WorkspaceType == "FileSystem"):
    gp.CreateFileGDB(work, name + ".gdb")
    FGDB = sys.argv[3]+ "/"+ name + ".gdb"
else:
    FGDB = sys.argv[3]

points = FGDB + "/" + name + "_P"
gp.AddMessage("converting raster to points")
gp.RasterToPoint_conversion(raster, points, rasterF)
gp.AddMessage("adding xy to points")
gp.AddXY_management(points)
#table will be FGDB + "/" + name + "_P_1#
gp.AddMessage("table to geodatabase")
gp.TableToGeodatabase_conversion(points, FGDB)
```

```
    gp.AddMessage("delete points")
    gp.Delete_management(points, "")
    Table = points + "_1"
    gp.AddMessage("adding field gridindex")
    gp.AddField_management(Table, "GRIDINDEX", "LONG", "", "", "", "",
"NULLABLE", "NON_REQUIRED", "")
    rows = gp.updatecursor(Table)

    # Get the first feature in the searchcursor
    row = rows.next()

    # Iterate through the rows in the cursor
    gp.AddMessage("Rows")
    while row:
        px = int((row.POINT_X - 500)/1000)
        py = int((row.POINT_Y - 500)/1000)
        index = int("%d%04d" % (px, py))
        row.SetValue("GRIDINDEX",index)
        rows.UpdateRow(row)
        row = rows.next()

    gp.DeleteField_management(Table, "POINT_X")
    gp.DeleteField_management(Table, "POINT_Y")
except "LicenseError":
    print "Spatial Analyst license is unavailable"
```