**Data Workflow for Raster Files**

1. Request (input for script)

· Purpose

· Check if script can execute wish or code needs to be added

· **ToDo:** Define nomenclature filename:

o product\_concept\_resolution\_epsg\_YYYY\_version

o workspace\_\_layername\_YYYYMMDDHH

· Categorie/Workspace Name for PGS or folder name (data\_dir/data/im-\*)

· Name of namespace of project to import Resources

· REPEX – wcs:CoverageId (e.g., im-data-\*\_\*)

· Check laptop/working environment (Mac, Windows, Linux) & packages are installed

· Link for data

· Input File folder

· Name of variable, years, resolution, region (shapefile or lat, lon)

· File type

· Data post-processing needed (avg, mean)

· Output file folder path (including info test-production geoserver)

2. Download data

· According to request info

· Post-processing if needed

3. Publish in test geoserver (TGS)

· QGIS validator on PC

· Run tests locally on PC

· Upload data to TGS

· Test – Geoserver data folder (anyone can enter)

· Script exists- Login & import - Static IP address (admin, password)

· Check Memory Issues (related to QGIS), possible file adaptation on TGS

· Run tests for k.LAB (**ToDo:** Modelers test if from Modelers PC everything is accessible and imports into k-LAB env) on TGS

· Check if needed to upload data to PGS

4. Publish in production geoserver (PGS)

· In Docker via ssh into Ura server

· On Ura Folder /data/geoserver\_transfer

· Via scp copy data from PC to Ura geoserver\_transfer folder

· Locate inside Docker container target folder

· Copy data from Ura to PGS docker

· Check if successful

· Delete files Ura

· Delete files PC

· Delete files from TGS

· Publishing on PGS (script needs to be tested)

5. Import data in k.LAB

· Import Resource in defined namespace in Resources (Bulk import resources)

· Adapter type - wcs

· URL corresponding to PGS

· REGEX filter for CoverageId (\_\_ instead of :)

· Change in future: URN containing person name