ARPHA Preprints

PENSOFT

Project Report

Author-formatted document posted on 05/12/2023

Published in a RIO article collection by decision of the collection editors.

DOI: https://doi.org/10.3897/arphapreprints.e116667

DDI-CDI-Workflow description of the EOSC Future WP6 Task 3, Science Project 9 ‘Climate Neutral and Smart Cities’

Benjamin Beuster, Hilde Orten

ARPHA Preprints *Author-formatted document posted on 05/12/2023*. DOI: https://doi.org/10.3897/arphapreprints.e116667

DDI-CDI-Workflow description of the EOSC Future WP6 Task 3, Science Project 9 ‘Climate Neutral and Smart Cities’

Authors: Benjamin Beuster, Hilde Orten

Keywords:
Attitudes; Behaviours; Climate; Air quality; Data integration; ESS; ERA5; EEA; Processes; DDI-CDI

Abstract

The paper focuses on the technical workflow description of EOSC Future’s Science Project 9 'Climate Neutral and Smart Cities', using reputable data sources such as the European Social Survey (ESS), the European Environmental Agency (EEA), and Copernicus ERA5.

A significant contribution of this project to the European Open Science Cloud (EOSC) is its demonstration of cross-domain data integration using the DDI-CDI metadata standard. This serves as a practical example of how the DDI-CDI Process model can offer a standardised methodology for detailing integration processes, thereby ensuring clarity for researchers dealing with integrated data and computed variables.

The paper outlines the key elements of the CDI-Process model selected for this approach, which includes around 10 classes such as 'Activity,' 'Step,' and 'Parameter.' These classes form the structural framework for the data integration steps.

Additionally, a tool developed by Wackerow (2023) under the project visualises the entire workflow as outlined in the CDI workflow description[1].

1) Process description

Process description serves as an essential roadmap for understanding how various tasks are interconnected, the resources required, and the flow of information or data. This is especially crucial in areas like data integration and the computation of variables as outlined by Orten et al. (2023). In the realm of data integration, a well-documented process can eliminate ambiguities about how data flows, what transformations are applied, and how data quality is maintained. This clarity is vital because data integration often involves collaboration among multiple stakeholders, including experts from different research disciplines, data scientists, and data engineers.

Similarly, in the computation of variables, a well-defined process is invaluable. In scientific research or data analytics, where reproducibility is a priority, a clearly outlined process ensures that variables are consistently computed across different runs or datasets. This clarity also facilitates the optimization of calculations, which is particularly important when dealing with large datasets or complex algorithms. Quality checks and validation become more streamlined, thereby ensuring the reliability and accuracy of the computed variables. Additionally, understanding the computational steps involved aids in better allocation of resources, such as CPU time and memory, in cloud computing.

[1] https://eosc-provenance.sikt.no/

Beyond the technical aspects, a well-described process also fosters trust among stakeholders. This is beneficial not just for internal operations, evaluations and also for external reporting and publications.

In summary, process description acts as the backbone for understanding and optimising workflows, making it indispensable for achieving efficiency, accuracy, and compliance, whether in data integration or variable computation.

2) Metadata standards and models for process descriptions

In the field of process description, various models and metadata standards offer diverse approaches for outlining complex workflows and data interactions. Among the most noteworthy is the Business Process Model and Notation (BPMN)[2], which is widely used for its graphical notation that clearly details business processes. The Unified Modeling Language (UML)[3] offers a more general-purpose approach, featuring activity and sequence diagrams that are adaptable across multiple domains, from software engineering to business analysis.

One metadata standard of particular importance is the Provenance Ontology (PROV)[4]. As a The World Wide Web Consortium (W3C) standard[5], PROV provides a comprehensive framework for describing the lineage and transformations that data undergo. This effectively captures the provenance of web-based objects and extends its utility to describe processes that create or modify data. This ontology has become especially crucial in settings that require an accurate and detailed account of data transformations and lineage, thereby ensuring both transparency and reproducibility.

Another metadata standard for describing processes is "Cross Domain Integration" (DDI-CDI), developed by the Data Documentation Initiative (DDI) Alliance[6] (version 1.0, release candidate 1, 2023). DDI-CDI process descriptions can be understood as extensions of the PROV standard.

This standard is designed to facilitate the integration, discovery, and use of data across different domains, sources, and types. The DDI-CDI Detailed Model (2023) aims to provide a comprehensive, flexible framework that can be used for a wide variety of data integration scenarios. It is particularly useful in complex research projects that require the integration of various types of data from multiple sources, ensuring that the data is well-documented, discoverable, and usable for analysis.

DDI-CDI includes sections specifically designed for describing the data integration process. These are essential for understanding how various data sources are combined, how variables are computed, and how data quality is assured.

3) Selected subset of the CDI-Process model

Due to its robust capabilities and the richness and clarity of its elements for describing data integration processes, DDI-CDI was selected as the primary metadata standard for this project. It provides a

[2] https://www.omg.org/spec/BPMN/
[3] https://www.uml.org/
[4] https://www.w3.org/TR/2013/NOTE-prov-overview-20130430/
[5] https://www.w3.org/
[6] https://ddialliance.org/

uniform approach for detailing both data lineage and integration processes, particularly for data from sources like the European Social Survey (ESS)[7], the European Environmental Agency (EEA)[8], and Copernicus ERA5[9]. This comprehensive approach ensures that researchers have a clear understanding when working with integrated data and computed variables.

The process description used in this project focuses on a specific aspect of the DDI-CDI Process model, breaking down the workflow and processes into individual steps, each with its unique purpose and distinct input/output parameters. It's important to note that CDI covers both procedural and declarative processes as described by the DDI Training Group/DDI-CDI Working Group (2021). This approach emphasises the process sequence rather than the entire DDI-CDI process model.

A subset of the CDI model chosen to describe the integration process can be accessed using the following link: https://ddi-alliance.bitbucket.io/DDI-CDI/processSequence_2023-06-04/index.html.

This package includes approximately 10 classes within the DDI-CDI model, such as Activity, Step, and Parameter.

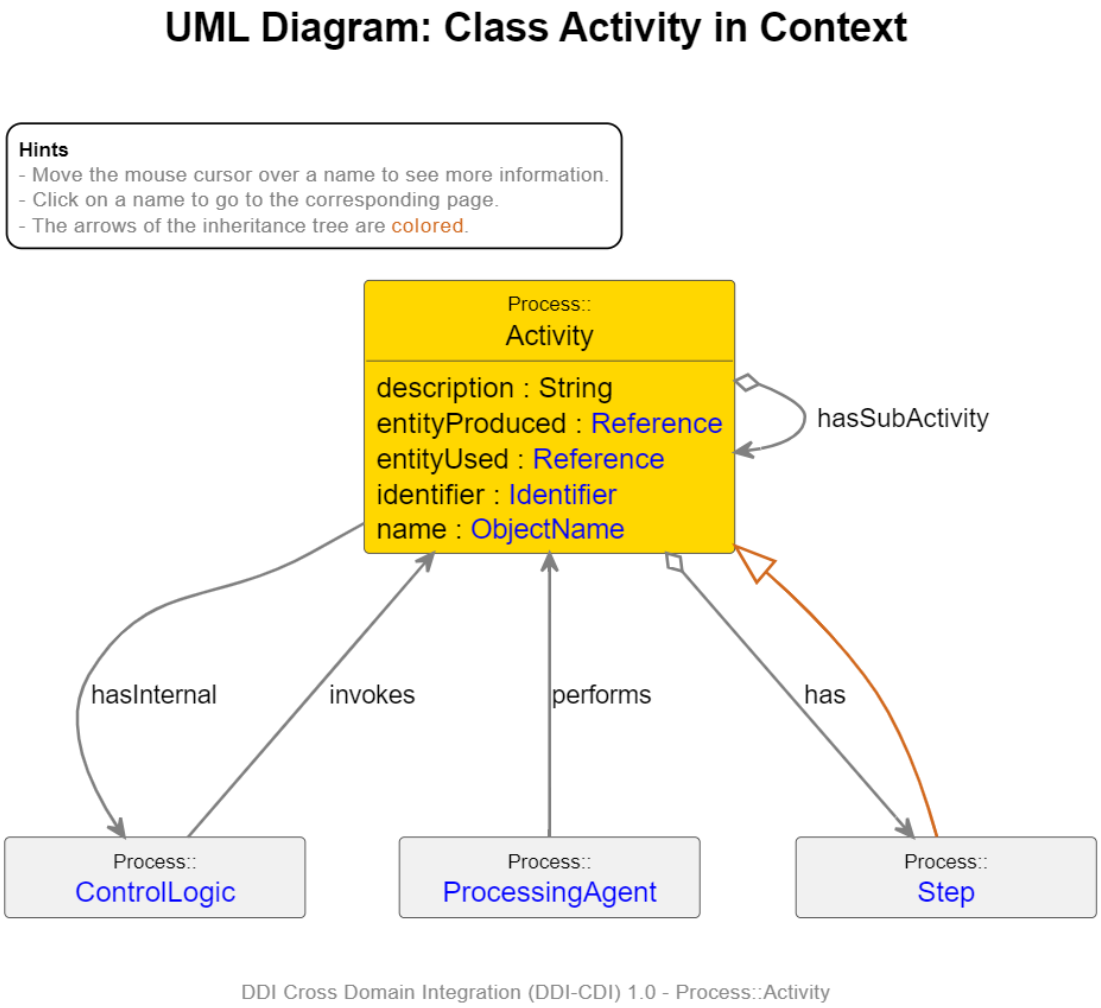
3.1) Activity

Activities serve as high-level overviews of data integration steps and can include sub-activities to establish a hierarchical structure. These activities are not parameterized and operate at a broader scope, such as the data file level. The CDI attributes of the class 'Activity' include 'entityUsed,' which refers to the resources used in the activity, and 'entityProduced,' which refers to the outcomes resulting from the activity. The resources employed can vary and include various types of web input, such as files available via the internet and APIs, as well as web pages and source data descriptions. The outcomes that result from the Activity (entityProduced) are, in our cases, Digital Object Identifiers (DOI) of data files that resolve to the landing pages in the ESS Data Portal as outlined by Bidargaddi et al. (2022).

Activities can be further broken down into steps, which provide a more detailed account of data processing and variable computation.

Below is a UML diagram of the Class Activity[10], available from DDI-CDI UML subset, that was generated as part of the project documentation. The online documentation provides further details on attributes and associations available for this class.

[7] https://www.europeansocialsurvey.org/
[8] https://www.eea.europa.eu/en
[9] https://climate.copernicus.eu/climate-reanalysis
[10] https://ddi-alliance.bitbucket.io/DDI-CDI/processSequence_2023-06-04/DDICDILibrary/Classes/Process/Activity.html#super-class-hierarchy-generalization



Hierarchical structure of the activities involved in the project.

Activities are hierarchically structured using three levels, each with a different level of detail regarding the integration and processing processes.

Level 1: Integrate climate and air quality data with ESS
- Level 2: ERA5 Data (Copernicus)
  - Level 3: ERA5 Get raw input
  - Level 3: ERA5 Marshalling data
  - Level 3: ERA5 Data Processing
- Level 2: EEA Air Quality
  - Level 3: EEA Get raw input
  - Level 3: EEA Marshalling data
  - Level3: EEA Data Processing
- Level 2: Merging of ERA5, EEA and ESS data
  - Level 3: Merging of ERA5, EEA and ESS8 data
  - Level 3: Merging of ERA5, EEA and ESS9 data
  - Level 3: Merging of ERA5, EEA and ESS10 FTF data
  - Level 3: Merging of ERA5, EEA and ESS10 Self-completion data

Detailed breakdown of the activities involved in the project.

The “Activity” column contains the activity name and the link to the element in the provenance tool. “Level” indicates the position level in the structure, and “Description” provides the textual description of the activity.

Table 1: Detailed breakdown of the activities involved in the project

Activity	Level	Description
Integrate climate and air quality data with ESS	Level 1	Integrate climate data from Copernicus ERA5 and air quality data from the European Environmental Agency (EEA) with data from the European Social Survey (ESS) for Berlin, Oslo, Stockholm, Brussels, London, Paris, Vienna, Prague, Budapest, and Madrid
ERA5 Data (Copernicus)	Level 2	Ingest and prepare data from ERA5 data (Copernicus)
ERA5 Get raw input	Level 3	The process involves obtaining NUTS - Nomenclature of territorial units for statistics polygons for the relevant regions, followed by calling a public API with GPS coordinates derived from the polygons. A single API call is made per month, resulting in a gridded data response file with a default resolution of 0.1 degree latitude/longitude. Each month and region corresponds to one variable, resulting in over 20.000 files. The performance of this process is relatively slow, taking around a minute per call, and it requires a substantial number of calls to collect the complete dataset. There are over 12.000 raw input files in NetCDF4 format covering the period from 1990 to 2022.
ERA5 Marshalling data	Level 3	The process involves reading NetCDF files into Panda dataframes, obtaining estimated population data for grids from the Global Human Settlements data based on Eurostat, merging the population data with ERA5 data, and writing the merged data to disk in Parquet format. External experts perform quality checks on the merged data, which could be either a one-off or a regular quality assurance check. The process utilizes over 12.000 NetCDF4 files as input as well as data from the GHSL - Global Human Settlement Layer. The output of the process is a single Parquet file named “Interim data for review” with its corresponding URI.

ERA5 Data Processing	Level 3	The process involves creating a date variable from timestamps based on the time zone of each region, considering that the data is recorded hourly. It also addresses unit differences, converting Kelvin to Celsius and meters to millimeters. The data is then grouped by date, variable, and region, and temperature is averaged while also obtaining maximum and minimum values, accumulating precipitation by date, and identifying the maximum wind gust value. Moving averages are calculated for variables using different time windows (7-day, 30-day, 90-day, 365-day). Baseline values for temperature, precipitation, wind gust, and deviations from the baseline (anomalies) are determined based on the period from 1991 to 2020. Data older than 2015 is removed, and a group-by operation is performed, collapsing the data by region using population-weighted averages. It is important to note that the ERA5 data may contain imputed and missing values. In memory, each row corresponds to a region, with mesh-blocks aggregated per day to calculate region-level values by taking the average of all variables weighted by the population of each block. The resulting data is stored to disk in CSV, SAV, or other suitable formats, as the data size remains manageable.
EEA Air Quality	Level 2	Ingest and prepare data from EEA Air Quality
EEA Get raw input	Level 3	The process involves obtaining a list of stations with GPS coordinates from a pan-European metadata CSV file and selecting specific stations based on their GPS coordinates and ID. Only background stations are selected. The NUTS regions, defined by their GPS coordinates for polygons, are taken into account when the data is collected.
EEA Marshalling data	Level 3	All the collected data, including pollutant-by-station-by-hour information, is consolidated into a single file named “eea-stations,” with each row representing a specific pollutant, station, and hour. The merged data is stored in Parquet format, serving as a checkpoint for external reviewers to validate the analysis.
EEA Data Processing	Level 3	The process involves collapsing the data on time to group it by day, using maximum values. It is then collapsed by region, also using the maximum value for each region. Index variables are calculated based on the concentrations provided by the EEA. These index variables help create a classification of air quality (e.g., Good, Fair, Moderate) are based on the European Air Quality Index as of August 2023. Derived variables are computed to determine the worst quality and the number of days with poor quality for different time periods, with specific dates serving as reference points. The resulting data is stored as a file, which can be of a suitable format based on the size requirements.
Merging of ERA5, EEA and ESS data	Level 2	Merging of ERA5, EEA and ESS data

Merging of ERA5, EEA and ESS8 data	Level 3	The process involves merging ERA5, EEA, and ESS8 data, selecting only the observations relevant to the specified regions based on coverage, and saving the integrated data as a single file for ESS Round 8.
Merging of ERA5, EEA and ESS9 data	Level 3	The process involves merging ERA5, EEA, and ESS9 data, selecting only the observations relevant to the specified regions based on coverage, and saving the integrated data as a single file for ESS Round 9.
Merging of ERA5, EEA and ESS10 data	Level 3	The process involves merging ERA5, EEA, and ESS10 face-to-face data, selecting only the observations relevant to the specified regions based on coverage, and saving the integrated data as a single file for ESS Round 10.
Merging of ERA5, EEA and ESS10 Self-completion data	Level 3	The process involves merging ERA5, EEA, and ESS10 Self-completion data, selecting only the observations relevant to the specified regions based on coverage, and saving the integrated data as a single file for ESS Round 10 (Self-completion).

3.2) Steps and Parameters

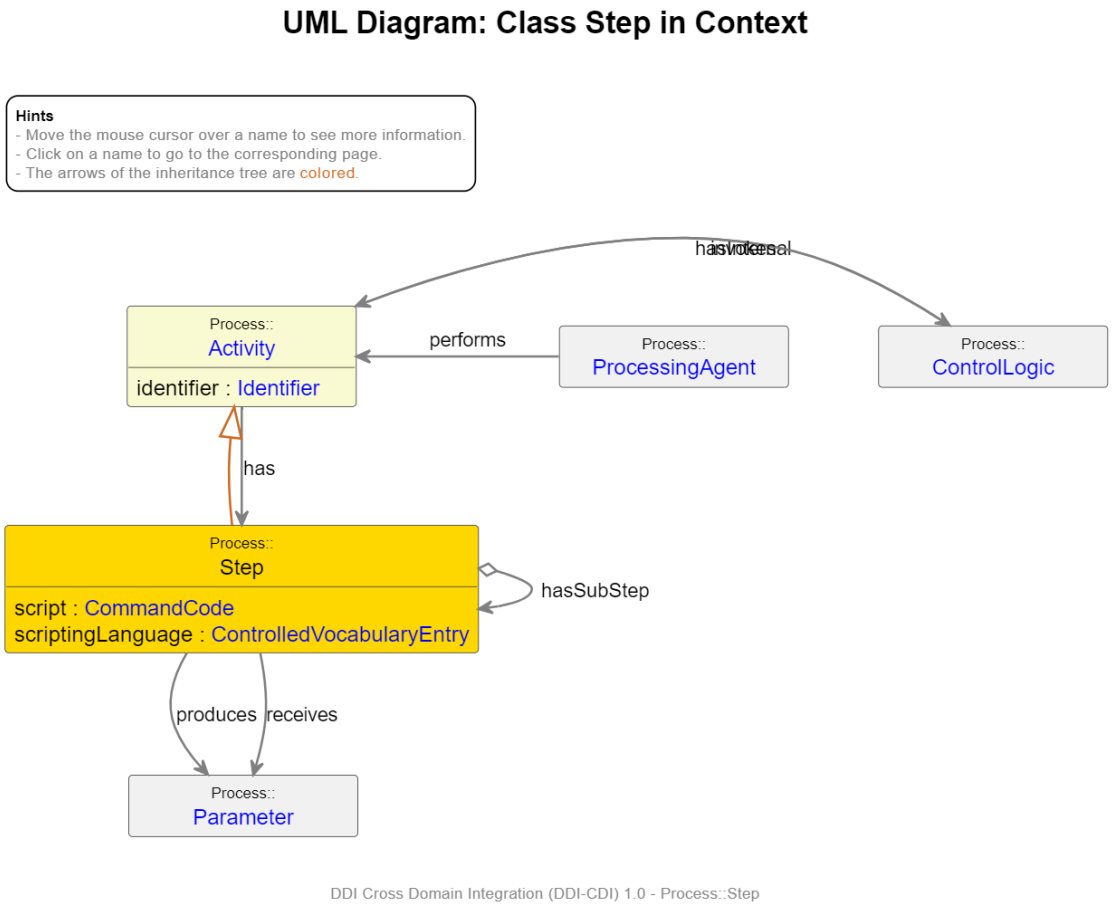
Steps act as modular, parameterized sub-processes that manage the flow of information. Within a single activity, multiple steps can be executed, each processing input parameters and generating output parameters. These parameters can manifest in various forms, such as data files or specific variables.

In our implementation, we've tailored parameters to be instance variables, typically represented as columns within a unit record data file. An attribute named "entityBound" is associated with each parameter. In our context, this attribute contains a URI that directs to the variable's display location in the ESS Data Portal[11]. Notably, the variable's ID is embedded within this URI and aligns with its identifier in the Data Documentation Initiative - Lifecycle documentation (DDI-L)[12].

Another crucial attribute within the step is "Script", which holds details about the command employed for data processing. Within DDI-CDI, "Script" can include a description of the command to help clarify its purpose and process or a URI linking to an external command script. In our approach, the URI leads to a specific Python file housed in a GitHub repository. Moreover, this URI pinpoints the exact line in the script where the output parameter gets computed, ensuring greater clarity for users seeking to grasp the exact variable's computation process.

Below is a UML diagram of the class Step[13], available from DDI-CDI UML subset, that was generated as part of the project documentation. The online documentation provides further details on attributes and associations available for this class.

[11] https://ess-search.nsd.no/en/study/71586b4f-ef66-4b90-aed7-e7e7ad7406ce
[12] https://ddialliance.org/Specification/DDI-Lifecycle/3.3/
[13] https://ddi-alliance.bitbucket.io/DDI-CDI/processSequence_2023-06-04/DDICDILibrary/Classes/Process/Step.html#super-class-hierarchy-generalization



Detailed breakdown of the steps involved in the project

A total of 58 target variables are computed and described in the provenance tool as well as in the DDI-CDI output. Each step refers to one column in the produced data file, detailing its production. It includes both a textual description and the Python programming code, accessible via the script hyperlink. If other variables are employed for the production, and the target variable is utilised for subsequent production (input and output parameters), those elements are incorporated too. Directions are emphasised using arrows. Below is the example for “Compute target variable ‘paccta’ ”, that produces the target variable ‘paccta - Total precipitation, date’.

Process Step

Description Compute target variable 'paccta' based on variable 'pac', 'region_id','date' and 'pop'. Calculate total precipitation by date in mm for each grid cell to be included in the data. The total precipitation over 24 hours is the sum of the individual total precipitation values for each hour. Precipitation measured in m converted to mm. Calculate total precipitation by date for each region as an average of the grid cell value, weighted by variable 'pop' that is based on global human settlements statistics.

This step uses a script written in Python3.

Diagram of the Process Step

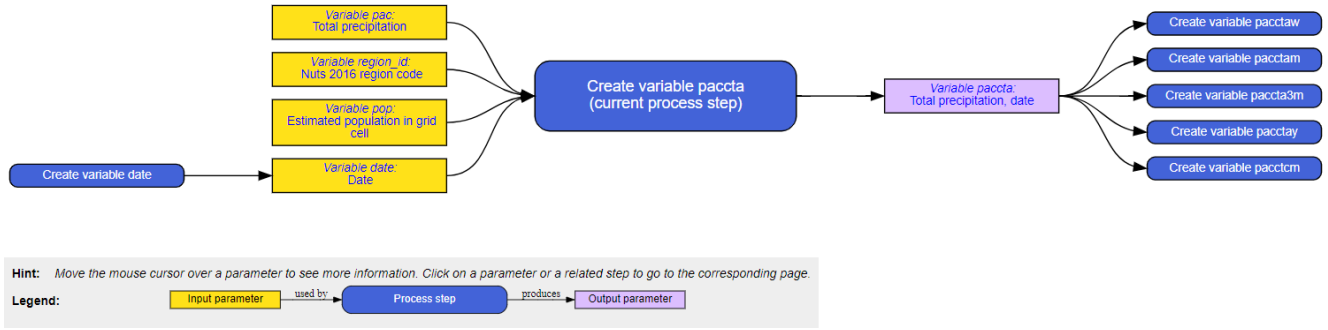


Table 2: Detailed breakdown of the steps involved in the project

Step	Description	Target Variable Label
Create variable tmpdca	Compute target variable 'tmpdca' based on variable 'tmpdc', 'region_id', 'date' and 'pop. Calculate average temperature by date for each grid cell to be included in the data. Temperature measured in Kelvin is converted to degrees Celsius (°C) by subtracting 273.15. Calculate temperature averages for each region, weighted by variable 'pop' that is based on global human settlements statistics.	Temperature in degrees Celcius, date average
Create variable tmpdcmx	Compute target variable 'tmpdcmx' based on variable 'tmpdc', 'region_id', 'date' and 'pop'. Calculate maximum temperature by date for each grid cell to be included in the data. Temperature measured in Kelvin is converted to degrees Celsius (°C) by subtracting 273.15. Calculate average maximums for each region, weighted by variable 'pop' that is based on global human settlements statistics.	Temperature in degrees Celcius, date maximum
Create variable tmpdcmn	Compute target variable 'tmpdcmn' based on variable 'tmpdc', 'region_id', 'date' and 'pop'. Calculate minimum temperature by date for each grid cell to be included in the data. Temperature measured in Kelvin converted to degrees Celsius (°C) by subtracting 273.15. Calculate average minimums for each region, weighted by variable 'pop' that is based on global human settlements statistics.	Temperature in degrees Celcius, date minmum

Create variable tmpdcaw	Calculate average temperature week (seven days) before the date, date included for each region, based on variables 'tmpdca', 'date' and 'region_id'.	Temperature in degrees Celcius, week average to date
Create variable tmpdcam	Calculate average temperature month (thirty days) before the date, date included for each region, date included, based on variables 'tmpdca', 'date' and 'region_id'.	Temperature in degrees Celcius, month average to date
Create variable tmpdca3m	Calculate average temperature three months (ninty days) before the date, date included for each region, date included, based on variables 'tmpdca', 'date' and 'region_id'.	Temperature in degrees Celcius, three months average to date
Create variable tmpdcay	Calculate average temperature year (three hundred and sixty five days) before the date, date included for each region, based on variables 'tmpdca', 'date' and 'region_id'.	Temperature in degrees Celcius, year average to date
Create variable tmpdcacm	Compute target variable 'tmpdcacm' based on variables 'tmpdca', 'region_id' and 'date'. Calculate average temperature for each calendar month for each region.	Temperature in degrees Celcius, calendar month average
Create variable tmpdcamb	Compute target variable 'tmpdcamb' based on variables 'tmpdca', 'region_id' and 'date'. Calculate average temperature for each calendar month for each region, across the baseline period 1991 - 2020.	Temperature in degrees Celcius, multi-year calendar month averages, baseline 1991 - 2020
Create variable tmp95pacmb	Compute target variable 'tmp95pacmb' based on variable 'tmpdc', 'date', 'region_id'.and 'pop' Calculate 95 percentile values for each calendar month for each grid cell as an average of the 1991 - 2020 baseline period values. Temperature measured in Kelvin converted to degrees Celsius (°C) by subtracting 273.15. Calculate calendar month averages for each region as an average of 95 the percentil values for each calendar month for each grid for the baseline period 1991 - 2020, weighted by variable 'pop' that is based on global human settlements statistics.	Temperature in degrees Celcius, multi-year calendar month 95th percentiles, baseline 1991 - 2020

Create variable tmpanod	Compute target variable 'tmpanod' based on variables 'tmpdca', 'tmpdcamb', 'region_id' and 'date'. Calculate 'tmpanod' as day average for each region, minus calendar month average base value 1991 - 2020 (tmpdca - tmpdcamb).	Temperature anomaly, date
Create variable tmpanocm	Compute target variable 'tmpanocm' based on variables 'tmpdcacm', 'tmpdcamb', 'region_id' and 'date'. Calculate 'tmpanocm' as calendar month average for each region, minus calendar month average base value 1991 - 2020 (tmpdcacm - tmpdcamb).	Temperature anomaly, calendar month
Create variable paccta	Compute target variable 'paccta' based on variable 'pac', 'region_id','date' and 'pop'. Calculate total precipitation by date in mm for each grid cell to be included in the data. The total precipitation over 24 hours is the sum of the individual total precipitation values for each hour. Precipitation measured in m converted to mm. Calculate total precipitation by date for each region as an average of the grid cell value, weighted by variable 'pop' that is based on global human settlements statistics.	Total precipitation, date
Create variable pacctaw	Compute target variable 'pacctaw' based on variables 'paccta', 'date' and 'region_id'. Calculate average total precipitation for the week (seven days) before the date, date included, for each region.	Total precipitation, weekly sum to date
Create variable pacctam	Compute target variable 'pacctam' based on variables 'paccta', 'date' and 'region_id'. Calculate average total precipitation for the month (thirty days) before the date, date included, for each region.	Total precipitation, monthly sum to date
Create variable paccta3m	Compute target variable 'paccta3m' based on variables 'paccta', 'date' and 'region_id'. Calculate average total precipitation for the three months (ninty days) before the date, date included, for each region.	Total precipitation, 3-monthly sum to date
Create variable pacctay	Compute target variable 'pacctay' based on variables 'paccta', 'date' and 'region_id'. Calculate average total precipitation for the year (three hundred and sixty five days) before the date, date included, for each region.	Total precipitation, yearly sum to date
Create variable pacctcm	Compute target variable 'pacctcm' based on variables 'paccta', 'date' and 'region_id'. Calculate 'pacctcm' as an average of total precipitation date values for the calendar month.	Total precipitation, calendar month

Create variable pacctmb	Compute target variable 'pacctmb' based on variables 'pacctm', 'region_id' and 'date'.Calculate accumulated precipitation averages for each calendar month for each region over the baseline period 1991 to 2020.	Total precipitation, multi-year calendar month averages, baseline 1991 - 2020
Create variable paccdcm	Compute target variable 'paccdcm' based on variables 'pacctcm', 'pacctmb', 'region_id' and 'date'.Compute variable for calendar month precipitation anomaly for each region as (current month/norml)*100. Using variables: (pacctcm/pacctmb)* 100	Total precipitation, calendar month relative to normal
Create variable iwg10mx	Compute target variable 'iwg10mx' based on variables 'iwg10', 'region_id' and 'date'. Calculate maximum wind gust value pr. date for each grid cell. The maximum wind gust value pr. date for a grid cell within the region becomes the value of the target variable.	Instantaneous 10 metre maximum wind gust maximum, date
Create variable iwg10mxaw	Compute target variable 'iwg10mxaw' based on variables 'iwg10mx', 'region_id' and 'date'. Calculate the average maximum wind gust value for the week (seven days) before the date, date included, for each region.	Instantaneous 10 metre maximum wind gust maximum, week average to date
Create variable iwg10mxam	Compute target variable 'iwg10mxam' based on variables 'iwg10mx', 'region_id' and 'date'. Calculate the average maximum wind gust value for the month (thirty days) before the date, date included, for each region.	Instantaneous 10 metre maximum wind gust maximum, month average to date
Create variable iwg10mxa3m	Compute target variable 'iwg10mxa3m' based on variables 'iwg10mx', 'region_id' and 'date'. Calculate the average maximum wind gust value for the three months (ninty days) before the date, date included, for each region.	Instantaneous 10 metre maximum wind gust maximum, 3-month average to date
Create variable iwg10mxay	Compute target variable 'iwg10mxay' based on variables 'iwg10mx', 'region_id' and 'date'. Calculate the average maximum wind gust value for the year (three hundred and sixty five days) before the date, date included, for each region.	Instantaneous 10 metre maximum wind gust maximum, year average to date

Create variable iwg10mxamb	Compute target variable 'iwg10mxamb' based on variables 'iwg10mx', 'region_id' and 'date'. Calculate the average maximum wind gust for each calendar month for each region, across the baseline period 1991 - 2020.	Instantaneous 10 metre maximum wind gust maximum, multi-year calendar month averages, baseline 1991 - 2020
Create variable aqiwdpm2_5	Compute target variable 'aqiwdpm2_5' starting from variables 'Concentration' and 'AirPollutant = PM2.5' for each background station. Compute where 'AirQualityStation' has values for the pollutant. Find 'Concentration' value at the 99th percentile for the pollutant. Create EEA Air Quality Index where PM2.5: 0 to 10 eq 'Good' represented by value '0'; 10 to 20 eq 'Fair' represented by value '1'; 20 to 25 eq 'Moderate' represented by value '2'; 25 to 50 eq 'Poor' represented by value '3'; 50 to 75 eq 'Very Poor' represented by value '4'; 75 to 800 eq 'Extremely poor' represented by value '5'. The worst air quality level measured on a specific day on one of the background stations in the region provides values to the variable.	Worst air quality index level PM2.5, date
Create variable aqiwdso2	Compute target variable 'aqiwdso2' starting from variables 'Concentration' and 'AirPollutant = SO2' for each background station. Compute where 'AirQualityStation' has values for the pollutant. Find 'Concentration' value at the 99th percentile for the pollutant. Create EEA Air Quality Index where SO2: 0 to 100 eq 'Good' represented by value '0'; 100 to 200 eq 'Fair' represented by value '1'; 200 to 350 eq 'Moderate' represented by value '2'; 350 to 500 eq 'Poor' represented by value '3'; 500 to 750 eq 'Very Poor' represented by value '4'; 750 to 1250 eq 'Extremely poor' represented by value '5'. The worst air quality level measured on a specific day on one of the background stations in the region provides values to the variable.	Worst air quality index level SO2, date
Create variable aqiwdno2	Compute target variable 'aqiwdno2' starting from variables 'Concentration' and 'AirPollutant = NO2' for each background station. Compute where 'AirQualityStation' has values for the pollutant. Find 'Concentration' value at the 99th percentile for the pollutant. Create EEA Air Quality Index where NO2: 0 to 40 eq 'Good' represented by value '0'; 40 to 90 eq 'Fair' represented by value '1'; 90 to 120 eq 'Moderate' represented by value '2'; 120 to 230 eq 'Poor' represented by value '3'; 230 to 340 eq 'Very Poor' represented by value '4'; 340 to 1000 eq 'Extremely Poor' represented by value '5'. The worst air quality level measured on a specific day on one of the background stations in the region provides values to the variable.	Worst air quality index level NO2, date

Create variable aqiwdo3	Compute target variable 'aqiwdo3' starting from variables 'Concentration' and 'AirPollutant = O3' for each background station. Compute where 'AirQualityStation' has values for the pollutant. Find 'Concentration' value at the 99th percentile for the pollutant. Create EEA Air Quality Index where O3: 0 to 50 eq 'Good' represented by value '0', 50 to 100 eq 'Fair' represented by value '1'; 100 to 130 eq 'Moderate' represented by value '2'; 130 to 240 eq 'Poor' represented by value '3'; 2040 to 380 eq 'Very poor' represented by value '4'; 380 to 800 eq 'Extremely poor' represented by value '5'. The average of the worst air quality level measured on a specific day on each of the background stations in the region provides values to the variable.	Worst air quality index level O3, date
Create variable aqiwd	Compute target variable 'aqiwd' based on 'aqiwdpm10'; 'aqiwdpm2_5'; 'aqiwdso2'; 'aqiwdno2; 'aqiwdo3' by date and region. The maximum value across the source variables becomes the value of the target variable	Worst air quality index level across pollutants, date
Create variable aqiw2dpm10	Compute target variable 'aqiw2dpm10' based on 'aqiwdpm10' 'region' and 'date'. The worst value for the pollutant across a date and the date before becomes the value of the target variable	Worst air quality index level PM10, last two days
Create variable aqiw2dpm2_5	Compute target variable 'aqiw2dpm2_5' based on 'aqiwdpm2_5', 'region' and 'date'. The worst value for the pollutant across a date and the date before becomes the value of the target variable	Worst air quality index level PM2.5, last two days
Create variable aqiw2dso2	Compute target variable 'aqiw2dso2' based on 'aqiwdso2', 'region' and 'date'. The worst value for the pollutant across a date and the date before becomes the value of the target variable	Worst air quality index level SO2, last two days
Create variable aqiw2dno2	Compute target variable 'aqiw2dno2' based on 'aqiwdno2', 'region' and 'date'. The worst value for the pollutant across a date and the date before becomes the value of the target variable	Worst air quality index level NO2, last two days
Create variable aqiw2do3	Compute target variable 'aqiw2do3' based on 'aqiwdo3', 'region' and 'date'. The worst value for the pollutant across a date and the date before becomes the value of the target variable	Worst air quality index level O3, last two days
Create variable aqiw2d	Compute target variable 'aqiw2d ' based on 'aqiw2dpm10'; 'aqiw2dpm2_5'; 'aqiw2dso2'; 'aqiwdno2'; 'aqiw2do3', 'region' and 'date'. The maximum value across the source variables becomes the value of the target variable	Worst air quality index level across pollutants, last two days

Create variable ndyprwpm10	Compute target variable 'ndyprwpm10' based on 'aqiwdpm10', 'region' and 'date'. The number of days during the last seven days from date, date included, where 'aqiwdpm10' = '3' 'Poor' or 'aqiwdpm10' = '4' 'Very poor' or 'aqiwdpm10' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse PM10, week before the date
Create variable ndyprwpm2_5	Compute target variable 'ndyprwpm2_5' based on 'aqiwdpm2_5', 'region' and 'date'. The number of days during the last seven days from date, date included, where 'aqiwdpm2_5' = '3' 'Poor' or 'aqiwdpm2_5' = '4' 'Very poor' or 'aqiwdpm2_5' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse on PM2.5, week before the date
Create variable ndyprwso2	Compute target variable 'ndyprwso2' based on 'aqiwdso2', 'region' and 'date'. The number of days during the last seven days from date, date included, where 'aqiwdso2' = '3' 'Poor' or 'aqiwdso2' = '4' 'Very poor' or 'aqiwdso2' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse on SO2, week before the date
Create variable ndyprwno2	Compute target variable 'ndyprwno2' based on 'aqiwdno2', 'region'and 'date'. The number of days during the last seven days from date, date included, where 'aqiwdno2r' = '3' 'Poor' or 'aqiwdno2r' = '4' 'Very poor' or 'aqiwdno2r' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse on NO2, week before the date
Create variable ndyprwo3	Compute target variable 'ndyprwo3' based on 'aqiwdo3', 'region' and 'date'. The number of days during the last seven days from date, date included, where 'aqiwdo3r' = '3' 'Poor' or 'aqiwdo3r' = '4' 'Very poor' or 'aqiwdo3r' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse on O3, week before the date
Create variable ndyprw	Compute target variable 'ndyprw' based on 'aqiwdpm10', 'aqiwdpm2_5', 'aqiwdso2', 'aqiwdno2', 'aqiwdo3', 'region' and 'date'. The number of days during the last seven days from date, date included, where one or more of variables 'aqiwdpm10', 'aqiwdpm2_5', 'aqiwdso2r', 'aqiwdno2', 'aqiwdo3' has value '3' 'Poor' or '4' 'Very poor' or '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'Poor' air quality level or worse on one or more pollutant indicators, week before the date
Create variable ndyprmpm10	Compute target variable 'ndyprmpm10' based on 'aqiwdpm10', 'region' and 'date'. The number of days during the last thirty days from date, date included, where 'aqiwdpm10' = '3' 'Poor' or 'aqiwdpm10' = '4' 'Very poor' or 'aqiwdpm10' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse PM10, month before the date

Create variable ndyprmpm2_5	Compute target variable 'ndyprmpm2_5' based on ' aqiwdpm2_5 ' , 'region' and 'date'. The number of days during the last thirty days from date, date included, where 'aqiwdpm2_5' = '3' 'Poor' or 'aqiwdpm2_5' = '4' 'Very poor' or 'aqiwdpm2_5' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse PM2.5, month before the date
Create variable ndyprmso2	Compute target variable ' ndyprmso2' based on 'aqiwdso2', 'region' and 'date'. The number of days during the last thirty days from date, date included, where 'aqiwdso2' = '3' 'Poor' or 'aqiwdso2' = '4' 'Very poor' or 'aqiwdso2' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse SO2, month before the date
Create variable ndyprmno2	Compute target variable 'ndyprmno2' based on 'aqiwdno2', 'region' and 'date'. The number of days during the last thirty days from date, date included, where 'aqiwdno2' = '3' 'Poor' or 'aqiwdno2' = '4' 'Very poor' or 'aqiwdno2' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse NO2, month before the date
Create variable ndyprmo3	Compute target variable 'ndyprmo3' based on 'aqiwdo3', 'region' and 'date'. The number of days during the last thirty days from date, date included, where 'aqiwdo3' = '3' 'Poor' or 'aqiwdo3' = '4' 'Very poor' or 'aqiwdo3' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse O3, month before the date
Create variable ndyprm	Compute target variable 'ndyprm' based on 'aqiwdpm10', 'aqiwdpm2_5', 'aqiwdso2', 'aqiwdno2', 'aqiwdo3', 'region' and 'date'. The number of days during the last thirty days from date, date included, where one or more of variables 'aqiwdpm10r', 'aqiwdpm2_5r', 'aqiwdso2r', 'aqiwdno2r', 'aqiwdo3r' has value '3' 'Poor' or '4' 'Very poor' or '5' 'Extremely poor' becomes the value of the target variable	Number of days with days with 'poor' level or worse on one or more pollutant indicators, month before the date
Create variable ndyprypm10	Compute target variable 'ndyprypm10' based on 'aqiwdpm10', 'region' and 'date'. The number of days during the last three hundred and sixty five days from date, date included, where 'aqiwdpm10' = '3' 'Poor' or 'aqiwdpm10' = '4' 'Very poor' or 'aqiwdpm10' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse PM10, year before the date
Create variable ndyprypm2_5	Compute target variable 'ndyprypm2_5' based on 'aqiwdpm2_5', 'region' and 'date'. The number of days during the last three hundred and sixty five days from date, date included, where 'aqiwdpm2_5' = '3' 'Poor' or 'aqiwdpm2_5' = '4' 'Very poor' or 'aqiwdpm2_5' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse PM2.5, year before the date

Create variable ndypryso2	Compute target variable 'ndypryso2' based on 'aqiwdso2', 'region' and 'date'. The number of days during the last three hundred and sixty five days from date, date included, where 'aqiwdso2' = '3' 'Poor' or 'aqiwdso2' = '4' 'Very poor' or 'aqiwdso2' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse SO2, year before the date
Create variable ndypryno2	Compute target variable 'ndypryno2' based on 'aqiwdno2', 'region' and 'date'. The number of days during the last three hundred and sixty five days from date, date included, where 'aqiwdno2' = '3' 'Poor' or 'aqiwdno2' = '4' 'Very poor' or 'aqiwdno2' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse NO2, year before the date
Create variable ndypryo3	Compute target variable ' ndypryo3' based on 'aqiwdo3', 'region' and 'date'. The number of days during the last three hundred and sixty five days from date, date included, where 'aqiwdo3' = '3' 'Poor' or 'aqiwdo3' = '4' 'Very poor' or 'aqiwdo3' = '5' 'Extremely poor' becomes the value of the target variable	Number of days with 'poor' air quality level or worse O3, year before the date
Create variable ndypry	Compute target variable 'ndypry ' based on 'aqiwdpm10', 'aqiwdpm2_5', 'aqiwdso2', 'aqiwdno2', 'aqiwdo3', 'region' and 'date'. The number of days during the last three hundred and sixty five days from date, date included, where one or more of variables 'aqiwdpm10', 'aqiwdpm2_5', 'aqiwdso2', 'aqiwdno2', 'aqiwdo3' has value '3' 'Poor' or '4' 'Very poor' or '5' 'Extremely poor' becomes the value of the target variable	Number of days with days with 'poor' level or worse on one or more pollutant indicators, year before the date

4) Syntax Representation of Metadata

The process model is implemented using the XML format for its syntax representation. Although the current version of the CDI metadata standard supports both XML and JSON-LD, DDI-CDI is designed to be expressed in multiple syntaxes. This adaptability is rooted in its design as a UML model and its goal to integrate seamlessly with other standards.

The choice of XML representation was influenced by two primary factors: firstly, its established support in the current CDI release; and secondly, the implementers' expertise with DDI-Lifecycle XML. Opting for XML is consistent, especially given that other segments of the project's documentation, such as study-level and variable metadata, also employ DDI-L XML. These standards are applied simultaneously throughout the project's documentation.

5) Generation of DDI-CDI XML

Given the absence of a database or tool capable of generating DDI-XML, a novel workflow had to be designed. Once the essential classes and attributes for the subset of the process model were defined, individual spreadsheets were generated for each class.

Within these spreadsheets, the attributes and associations of each class are represented as columns. Typically, attributes comprise names and descriptions, while associations are references either to other classes within the CDI model or to external objects outside of the DDI. A Python script is then employed to interpret the content of these spreadsheets, converting the data into DDI-CDI-XML. The primary libraries utilised in this Python script are ElementTree and Pandas.

For validation purposes, the Oxygen application[14] was used, ensuring the XML aligns with the most recently published CDI schema.

6) Funding

The EOSC Future Science Project *Climate Neutral and Smart Cities* is funded under the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017536.

7) Acknowledgments

Many thanks to people working on the *Climate Neutral and Smart Cities* for input and support: Arofan Gregory, Joachim Wackerow

8) References

DDI Cross Domain Integration (DDI-CDI), version 1.0, release candidate 1 (2023).. https://bitbucket.org/ddi-alliance/ddi-cdi/, https://ddialliance.org/Specification/ddi-cdi

DDI-Cross Domain Integration: Detailed Model (2023)
https://ddi-alliance.atlassian.net/wiki/download/attachments/860815393/Part_2_DDI-CDI_Detailed_Model_PR_1.pdf?version=3&modificationDate=1586887411228&cacheVersion=1&api=v2

Wackerow, J. (2023). ddi-cdi_process2web.
https://eosc-provenance.sikt.no

DDI Training Group/DDI-CDI Working Group (2021). Data Integration: Using DDI-CDI with Other Standards, 33-37
chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://codata.org/wp-content/uploads/2021/09/DDI-CDI_Other_Standards_Webinar.pdf

Orten, H., Rayner, D., Stavestrand, E., Alfredsson I., Lace, I., Vipavc Brvar, I., Dolinar, M., Wackerow, J.,Clark, H. (2023) Climate and Air Quality Indices for the European Social Survey , 17
https://preprints.arphahub.com/article/114675/

Bidargaddi, A., Agasøster, B., Skjåk, K., K., Risnes, Ø. (2022)
Report on Preparing the ESS for Services in the EOSC (ESS as a Service)
https://zenodo.org/records/6779526#.Y8ewm0GZO1s

[14] https://www.oxygenxml.com/