MULTISOURCE

Prototype available for planning platform code

Deliverable 5.4

Deliverable Number and Name	D5.4 - Prototype available for planning platform code
Work Package	WP5 – Planning Platform
Dissemination Level	Public
Author(s)	Maria Chiara Lippera, Jan Friesen
Primary Contact and Email	Jan Friesen, jan.friesen@ufz.de
Date Due	April 30th
Date Submitted	May 6th
File Name	2024_04_30_MULTISOURCE_WP5_D5.4
Status	Final
Reviewed by (if applicable)	Jaime Nivala, Coordinator
Suggested citation	Lippera M.C., Friesen J. (2024) Prototype available for planning platform code. MULTISOURCE Deliverable 5.4, H2020 grant no. 101003527

This document has been prepared in the framework of the European project MULTISOURCE. This project has received funding from the European Union's Horizon 2020 innovation action programme under grant agreement no. 101003527.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

The MULTISOURCE Planning Platform enables the generation and systematic comparison of urban stormwater management scenarios with and without nature based solutions. It provides decisional support for cities in selecting the most economical and sustainable option in the urban environment for stormwater management.

The Planning Platform is composed of a set of stand-alone tools written in python language and organised as independent packages. Each tool is presented in Section 2 in the form of factsheets, as well as the directory structure, and a description of the modules and functions developed within the package.

Two exemplary Planning Platform workflows that utilise different tools are illustrated in Section 3. In this section is reported the code to import the packages and export the results of the tools using the data of the Ecully catchment. When applying the Planning Platform tools to another European city, the workflow can be adapted to target specific objectives in the urban stormwater management and include other constraints.

The python repositories containing the tools of the Planning Platform will be published in GitLab until the end of the project. In a further step, links to other MULTISOURCE tools, the Hydraulic Disconnection Module and the Technology Selection Tool will be automated.

1.0 OVERVIEW

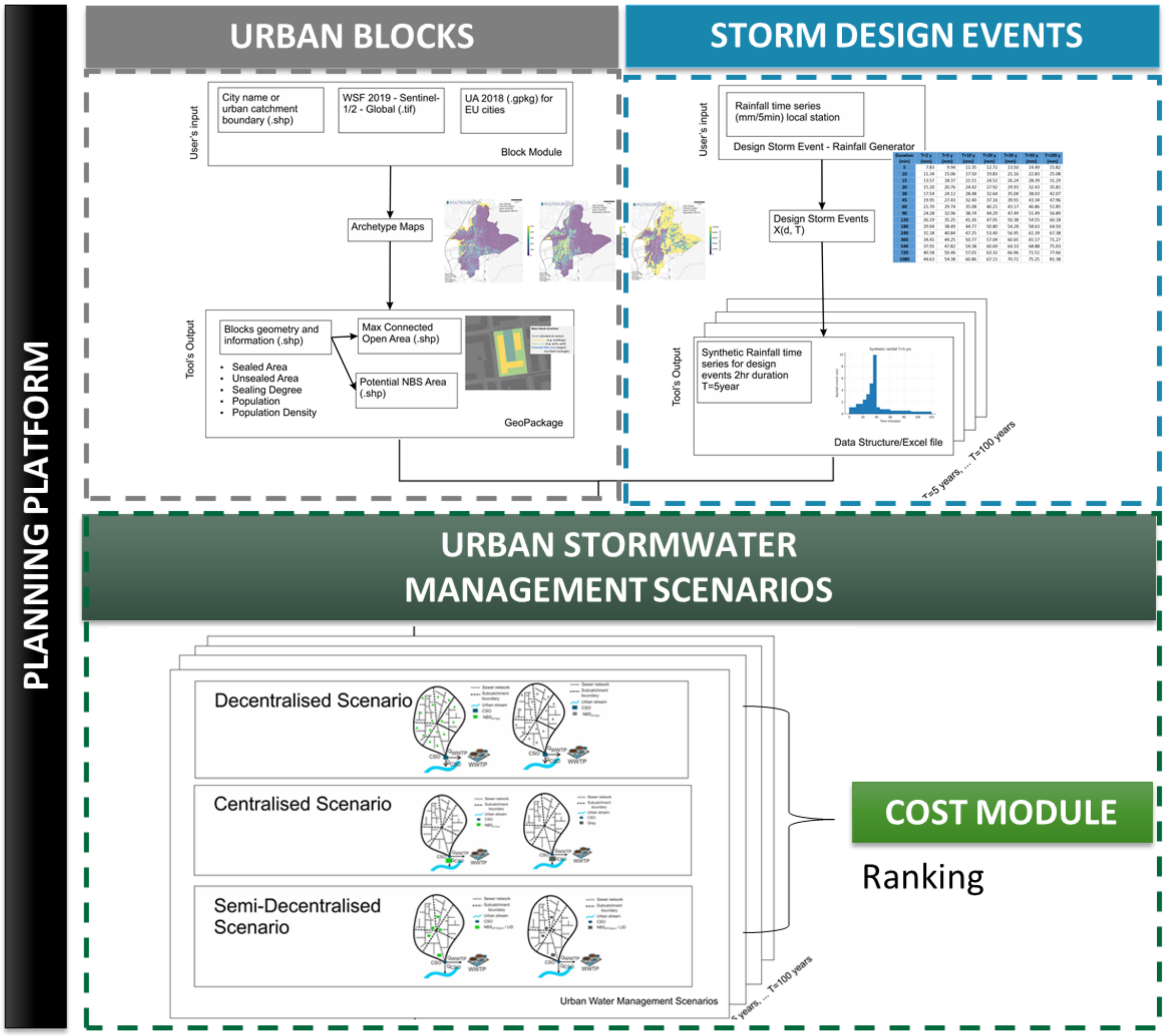
The Planning Platform enables city planners to generate urban stormwater management scenarios and identify suitable locations for the application of nature-based solutions (NBS) for stormwater or treatment. The Planning Platform is a set of modular open source tools, that, combined together provide spatial solutions for the scenarios and cost comparison through a dynamic cost comparison.

Making use of the set of modular tools, the Planning Platform workflow can be adapted to cities with different objectives or constraints for the planning of NBS solutions. Furthermore, more detailed data can be integrated to increase the detail of planning.

Each tool of the Planning Platform is independent and structured as a python project and will be published on GitLab, compliant with open-source standards. The python project structure of each tool ensures organisation, collaboration and distribution among developers. Thus, in the future, the tools of the Planning Platform will be enhanced through the implementation of new projects.

The modular tools in this prototype version of the Planning Platform are designated as “urban_blocks”, “storm_design_events”, “stormwater_managem_scenarios” and “cost_module” and will be presented in the following paragraphs. These are provisional designations for the packages, which may be subject to a change in name in the event that other distributed packages with identical names are present in the PyPI repository, which is a repository for software developed and shared by the python community.

Figure 1 Exemplary scenario development workflow of the Planning Platform

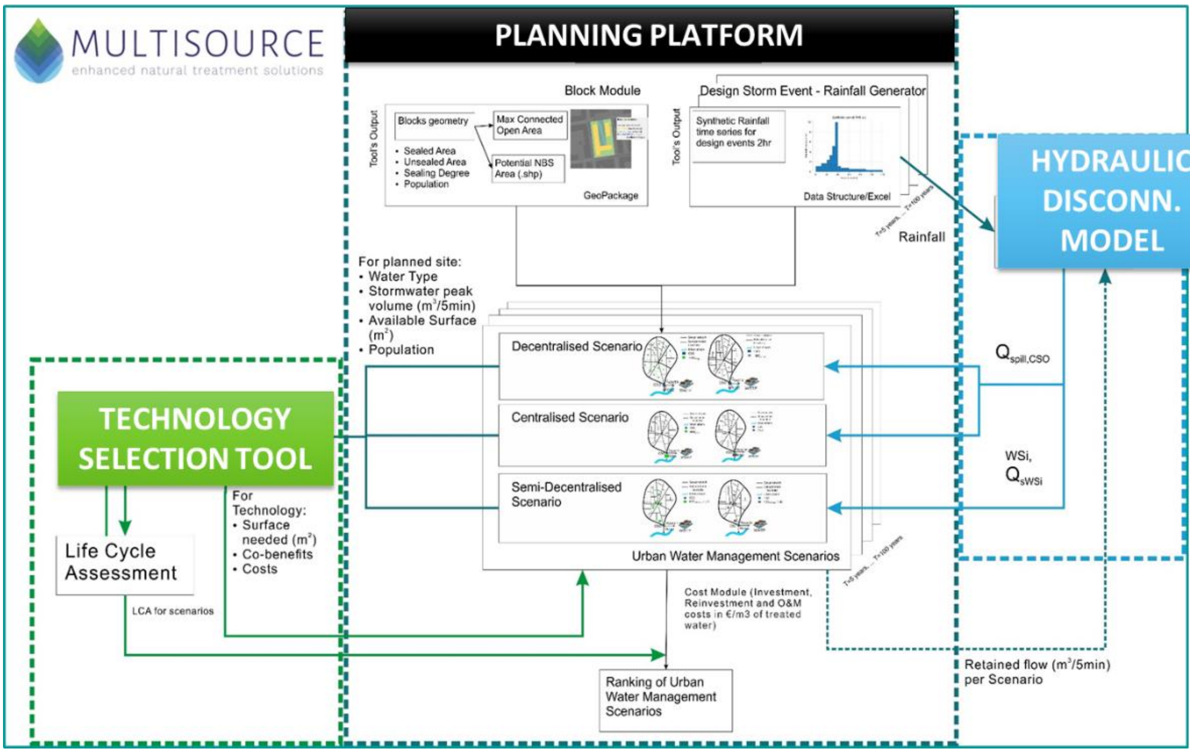


The codes to implement the Planning Platform will be presented in Section 3.0 *Planning Platform Workflows*. It is presented the workflow to compute the decentral stormwater potential, and the workflow to compute the urban stormwater management scenarios for the Ecully catchment. The decentral stormwater potential shows the potential maximum runoff that can be contained within the blocks. These results are obtained from the "urban_blocks" and "design_storm_events" tools, and can be applied to cities from public geospatial data, to assess the potential of decentralised stormwater retention. In the workflow for the computation of urban stormwater management scenarios, it is achieved the objective of reducing the Combined Sewer Overflow (CSO) within the Ecully catchment, through the simulations of the CSO spills from the Hydraulic Disconnection Module for the storm design events. In this workflow, decentralised and centralised solutions for urban stormwater retention/pre-treatment are compared to target at the same CSO spill volume reduction. As already specified, the workflow of the Planning Platform can be adapted to apply the tools to a different urban context, depending on the level of data availability and the specific target objectives for the planning of NBS.

Figure 1 illustrates the workflow of the Planning Platform to compare the urban stormwater management scenarios, including the tools to compute the urban blocks, the design storm events, and the generation of urban stormwater management scenarios.

In combination with the other MULTISOURCE tools, the urban stormwater management scenarios can be ranked, including life cycle analysis and socio-cultural ecosystem services. This combined workflow is shown in Figure 2. Thus, employing the full set of MULTISOURCE tools, the Planning Platform enables a systematic comparison of stormwater management strategies, including co-benefits, between traditional grey infrastructure and NBS potential development to increase the resilience of European cities.

Figure 2 Workflow of the Planning Platform in combination with the other MULTISOURCE tools.



2.0 MODULE FACTSHEETS

The tools that are part of the Planning Platform are illustrated in the form of fact sheets below. A detailed explanation is provided of the tools used to identify suitable locations for applying nature-based solutions

(NBS) in the urban environment. Furthermore, additional tools for urban stormwater management that have been developed during the MULTISOURCE project are presented.

The Planning Platform Tools description will differentiate between **packages**, **modules** and **functions** through the use of colour, thus facilitating the identification of each piece of code within the remainder of text. It is essential to note the following:

- **Packages** in python are directories containing **modules** and an __init__.py file. The __init__.py file may be either empty or contain the package's initialisation code. Packages facilitate the organisation of code into logical units, thus making it more straightforward to manage and maintain larger projects. The directory structure of each Planning Platform tool is reported below the factsheet. The package is designated by the same name that is used to describe the tool (e.g. "**urban_blocks**")

- **Modules** are python files containing a set of **functions**. Within these files with the extension ".py", there can be defined functions, variables, and classes that constitute the specific application (e.g. "**urban_NBS_area.py**"). Modules facilitate the organisation of code into reusable units, similar to a code library, that can be imported into other modules or scripts using the import statement.

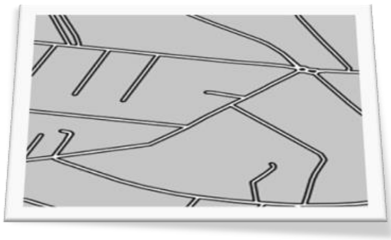
- A **function** in python is a block of code that performs a specific task. It takes inputs (parameters), processes them, and returns an output. Functions allow for the organisation of reusable pieces of code, thus, the code is reused when new input parameters are passed and the same operations are performed (e.g. "**rotational_search**").

2.1 Urban Blocks

URBAN BLOCKS

Description

The aim of the Urban Blocks tool is to generate the smallest functional unit for the urban planning of NBS solutions and provide blocks support information.



Input data

-City name
-Urban Atlas file (.gpkg)
-World Footprint Settlement file (.tiff)
-[opt] Boundary file urban catchment (.shp)

Output data

-Urban blocks (.shp)
-Potential area (.shp)
-NBS rectangle area (.shp)

Products

- Geometries of the blocks with: sealed and unsealed area, sealing degree, population, population density
- Archetype maps
- Max connected open area
- Potential NBS area (largest inscribed rectangle)

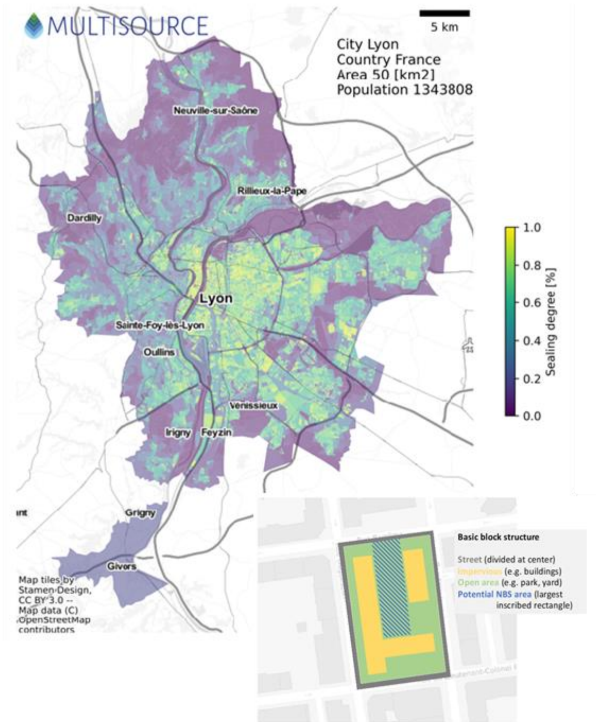


Figure 3 Directory structure for the Urban Block tool

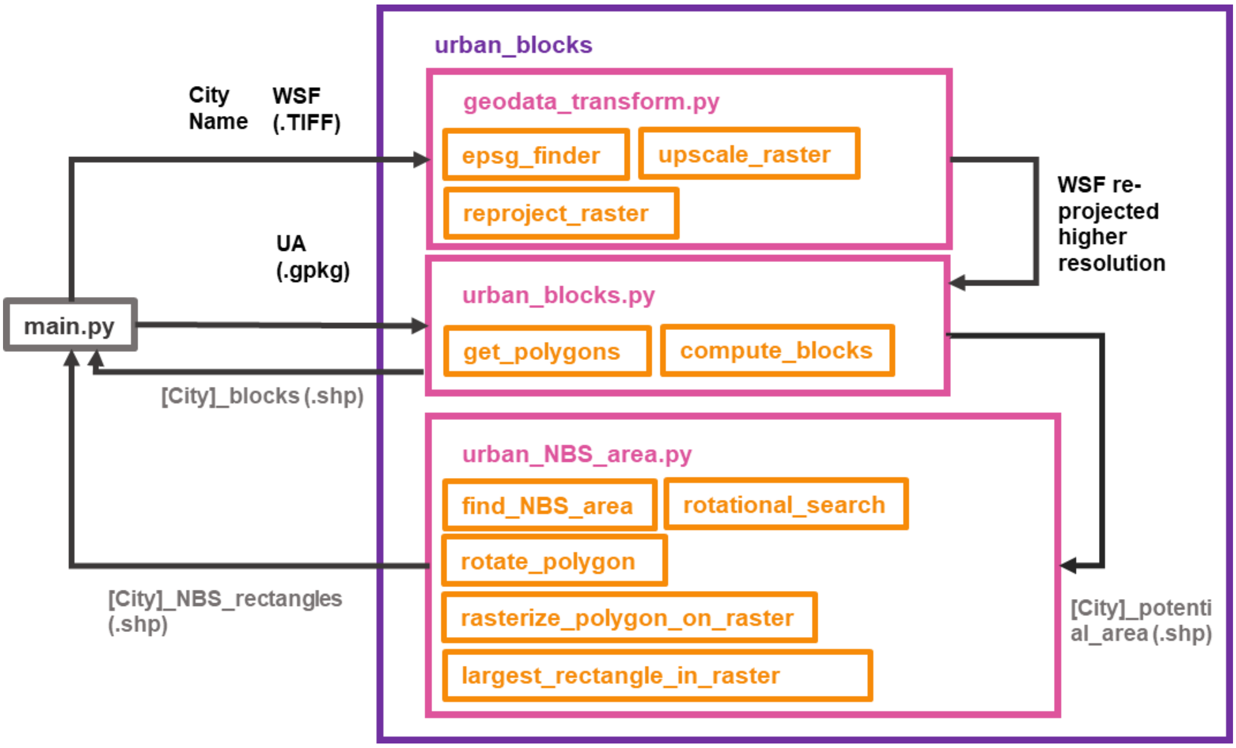
```
Planning Tools

+--- urban_blocks
|       __init__.py
|       geodata_transform.py
|       urban_blocks.py
|       urban_NBS_area.py
|       __init__.py
|
+---data
|   +---export
|   |   +---urban_environment
|   |           [City]_blocks.cpg
|   |           [City]_blocks.dbf
|   |           [City]_blocks.prj
|   |           [City]_blocks.shp
|   |           [City]_blocks.shx

|   |           [City]_NBS_rectangles.cpg
|   |           [City]_NBS_rectangles.dbf
|   |           [City]_NBS_rectangles.prj
|   |           [City]_NBS_rectangles.shp
|   |           [City]_NBS_rectangles.shx

|   |           [City]_potential_area.cpg
|   |           [City]_potential_area.dbf
|   |           [City]_potential_area.prj
|   |           [City]_potential_area.shp
|   |           [City]_potential_area.shx
|   |
|   +---processed
|   |       reprojected_higher_resWSF2019_.tif
|   |       reprojected_WSF2019_.tif
|   |
|   +---raw
|       +---urban_environment
|           |   [UA_2018_file_name].gpkg
|           |   [WSF2019_file_name].tif
|           |
|           +---boundaries
|                   [boundaries_file_name].cpg
|                   [boundaries_file_name].dbf
|                   [boundaries_file_name].prj
|                   [boundaries_file_name].qix
|                   [boundaries_file_name].shp
|                   [boundaries_file_name].shx
```

Within the package **urban_blocks**, the modules **geodata_transform.py**, **urban_blocks.py** and **urban_NBS_area.py** contain different functions.
The module **geodata_transform.py** transforms raster data (TIFF files) to perform geospatial operations. This module implements the function **epsg_finder** to find the EPSG code for the relative WGS84 / UTM zone given the declared city. The function **reproject_raster** performs raster reprojections and saves a destination raster. The function **upscale_raster** can modify the raster resolution of a declared factor (e.g. 2), and there is the option to resample the data for grid-based raster values.
The **module urban_blocks.py** contains the function **get_polygons** to derive polygon features from raster data, while the **compute_blocks** processes blocks from street graphs from the package OSMnx (Boeing 2017)The geometries of the blocks are computed, and the attributes of the blocks, such as the population and sealing degree, are derived from the geospatial operations involving the Urban Atlas (copernicus_v_3035_10_m_ua-2018_p_2017-2019_v01_r03, DOI: https://doi.org/10.2909/fb4dffa1-

6ceb-4cc0-8372-1ed354c285e6) and World Settlement Footprint (WSF) maps (Data: WSF data ©DLR 2019 All rights reserved., DOI: https://doi.org/10.15489/twg5xsnquw84) .

The module **urban_NBS_area.py** is composed of several functions to identify the largest rectangle that can be inscribed within the open area of the block having an irregular shape. In the function **rotate_polygon**, the geometry of the open area within the block is rotated around its centroid for various angles. The interval in degree of the rotation angle can be defined by the user (e.g. 1°). The smaller the defined rotation angle, the more accurate the results, at the expense of computation time. In **rasterize_polygon_on_raster** the rotated polygon is rasterised on the original raster grid, having the extent of the WSF grid to create a binary image. In **largest_rectangle_in_raster,** from the binary image, the largest rectangle x-axis orientated is found. For the set of rectangles, the best rectangle for the rotational search is rotated back to its original orientation and saved as the largest inscribed NBS rectangle area.

Figure 4 Modules and functions of the Urban Block tool

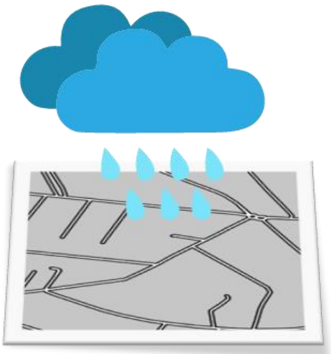


2.2 Storm Design Events

STORM DESIGN EVENTS

Description

The aim of the Storm Design Events tool is to perform an extreme events frequency analysis and generate extreme synthetic rainfall events with selected return periods.



Input data

-Sub-hourly rainfall data from the local weather station (.xlsx)
-[opt] Pre-formatted design events table when available

Output data

-Design events table (.xlsx)
-Synthetic rainfall events for specified duration and return period (.xlsx)

Products

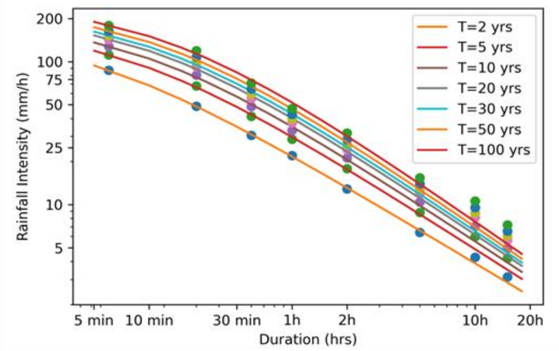
- Parameters of the EVI (Gumbel) probability distribution function

-Plotting positions of extreme events

- Intensity-Frequency-Duration curves

- Design Events for selected return periods and duration

- Plot of synthetic rainfall time series with 5 min resolution



Duration (min)	T=2 y (mm)	T=5 y (mm)	T=10 y (mm)	T=20 y (mm)	T=30 y (mm)	T=50 y (mm)	100 y (mm)
5	7.83	9.94	11.35	12.72	13.50	14.49	15.82
10	11.34	15.06	17.50	19.83	21.16	22.83	25.08
15	13.57	18.37	21.51	24.52	26.24	28.39	31.29
20	15.20	20.76	24.42	27.92	29.93	32.43	35.81
30	17.54	24.12	28.48	32.64	35.04	38.03	42.07
45	19.95	27.43	3				
60	21.70	29.74	3				
90	24.28	32.96	3				
120	26.19	35.25	4				
180	29.04	38.49	4				
240	31.18	40.84	4				
360	34.41	44.25	5				
540	37.91	47.82	5				
720	40.58	50.46	5				
1080	44.63	54.38	6				

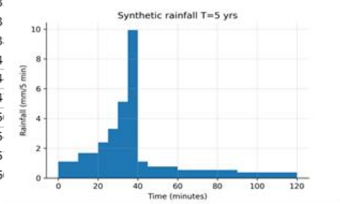


Figure 4 Directory Structure of the Design Storm Events

```
Planning Tools

+---storm_design_events
|       generate_design_event_from_time_series.py
|       generate_synthetic_rainfall.py
|       __init__.py
+---data
|   +---export
|   |      Synthetic_rainfall_events.xlsx
|   |
|   |
|   +---processed
|   |     Design_events_5min_res.xlsx
|   |
|   \---raw
|        30 years Rain 6min.xlsx
|
+---Graphs
|       EulerIItype_rainfall_T=10.png
|       EulerIItype_rainfall_T=100.png
|       EulerIItype_rainfall_T=2.png
|       EulerIItype_rainfall_T=20.png
|       EulerIItype_rainfall_T=30.png
|       EulerIItype_rainfall_T=5.png
|       EulerIItype_rainfall_T=50.png
|       plottingPostion1h.pdf
|       plottingPostion1h.png
|       plottingPostion2h.pdf
|       plottingPostion2h.png
|       return_periods.pdf
|       return_periods.png
```

In the Storm Design Events tool, the module **generate_design_events_from_time_series.py** has implemented several functions to compute the extreme events frequency analysis. The user can define the return periods of interest and the level of data availability.

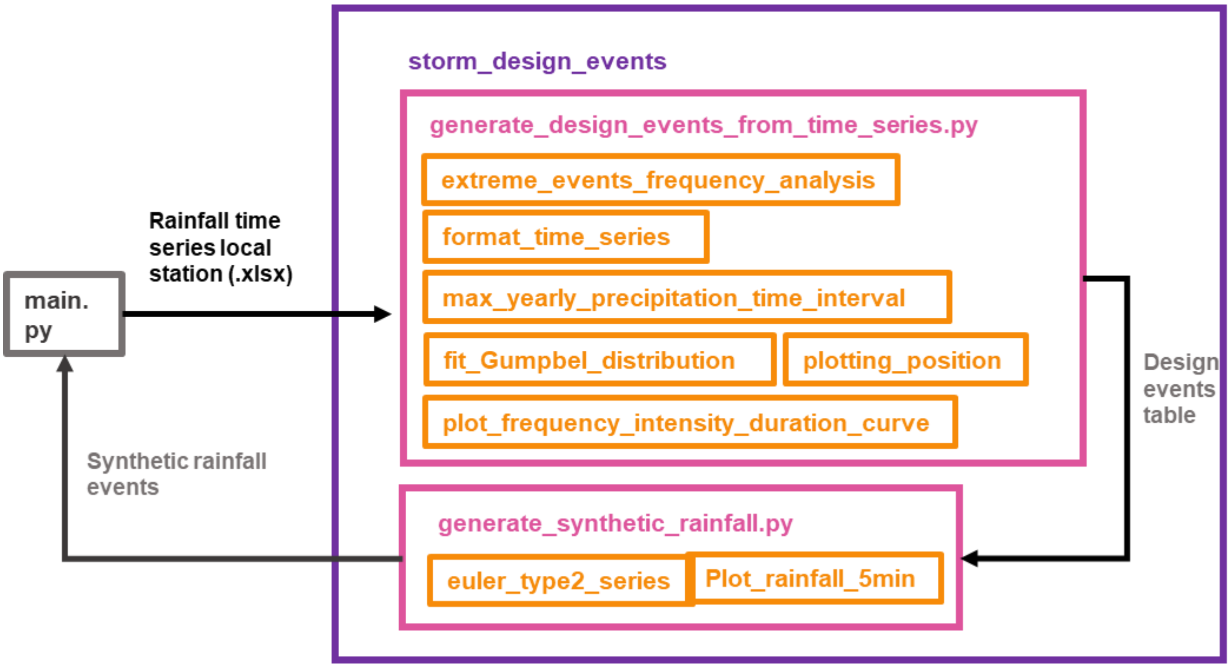
When no design events are available from weather services, e.g. KOSTRA-DWD-2010R dataset from the DWD Climate Data Center for Germany, the tool can read an Excel file containing precipitation data with sub-hourly resolution, a length of 30 years is recommended.

The function **format_time_series** reads the Excel file, defines the time resolution in minutes and fills the missing values with NaNs. The **max_yearly_precipitation_time_interval** takes a defined time interval array (e.g. 5, 10, 20, 30 .. 120) of minutes depending on the time resolution and, for each year, computes the annual maxima. The single events are identified within each year of the time series, and the maximum precipitation depth (mm) is defined for each duration time.

The Gumbel distribution - Extreme Value Type I (EVI) distribution is implemented in the function **fit_Gumpbel_distribution** by estimating the parameters through the momentum method given the statistics of the annual maxima for the different duration times.

The design events are then generated for the return periods T= 2, 5, 10, 20, 30, 50, 100 years and the Intensity-Duration-Frequency (IDF) curves are parametrised and plotted in the function **plot_frequency_intensity_duration_curve**. The dataframe of the design events is exported in Excel. In the module **generate_synthetic_rainfall.py** the design events values are passed to the functions **euler_type2_series** which computes the Euler Type II Hyetograph with a 5 min resolution. The synthetic rainfalls of 120 min duration and return periods T= 2, 5, 10, 20, 30, 50, 100 years are plotted in **plot_rainfall_5min.py**.

Figure 6 Modules and functions of the Design Storm Events tool

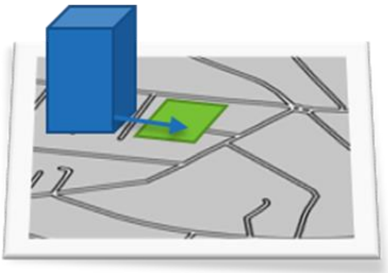


2.3 Urban Stormwater Management

URBAN STORMWATER MANAGEMENT SCENARIOS

Description

The aim of the Urban Stormwater Management Scenarios tool is to spatially locate and size the technologies for urban runoff according to the produced runoff and viable space.



Input data

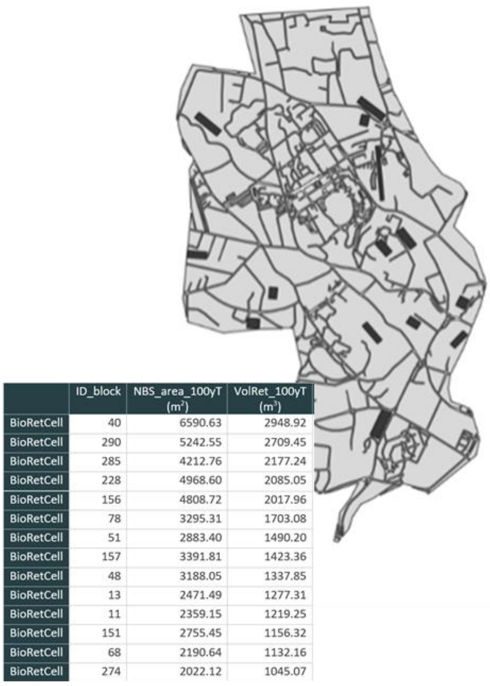
-Urban blocks (.shp) and NBS rectangle area (.shp)
-Extreme rainfall events (.xlsx)
-[opt] Target volumes to reduce at the catchment

Output data

-Decentral scenario (.shp)
-Decentral scenario excel table (.xlsx)
-Centralised scenario

Products

- Locations of the sites, area of the technology and retained runoff volume for the design return period
-Sizing given technologies parameters
-Hydrological simulation for an extreme year [dev]
- Set of scenarios: centralised/decentralised and grey/NBS
-Comparison total area and total volume



	ID_block	NBS_area_100yT (m^2)	VolRet_100yT (m^3)
BioRetCell	40	6590.63	2948.92
BioRetCell	290	5242.55	2709.45
BioRetCell	285	4212.76	2177.24
BioRetCell	228	4968.60	2085.05
BioRetCell	156	4808.72	2017.96
BioRetCell	78	3295.31	1703.08
BioRetCell	51	2883.40	1490.20
BioRetCell	157	3391.81	1423.36
BioRetCell	48	3188.05	1337.85
BioRetCell	13	2471.49	1277.31
BioRetCell	11	2359.15	1219.25
BioRetCell	151	2755.45	1156.32
BioRetCell	68	2190.64	1132.16
BioRetCell	274	2022.12	1045.07

Figure 5 Directory Structure for the Urban Stormwater Management Scenarios

```
Planning Tools
|
+---stormw_managem_scenarios
|       central_GREY.py
|       central_NBS.py
|       compute_inflow.py
|       decentral_GREY.py
|       decentral_NBS.py
|       parameters_BioRetentionCell.py
|       parameters_ConstructedWetland.py
|       parameters_InfiltrationShaft.py
|       parameters_RetentionTreatmentBasin.py
|       __init__.py
|
+---data
|   +---export
|   |   |
|   |   |
|   |   \---urban_stormwater_management_scenarios
|   |       |   decentral_NBS_scenario_5yT.xlsx
|   |       |
|   |       \---decentral_NBS_scenario_5yT
|   |               decentral_NBS_scenario_5yT.cpg
|   |               decentral_NBS_scenario_5yT.dbf
|   |               decentral_NBS_scenario_5yT.prj
|   |               decentral_NBS_scenario_5yT.shp
|   |               decentral_NBS_scenario_5yT.shx
|   |
|   |
|   \---raw
|       |
|       +---design_storm_events
|       |       Synthetic_rainfall_events.xlsx
|       |
|       +---urban_environment
|               [City]_blocks.cpg
|               [City]_blocks.dbf
|               [City]_blocks.prj
|               [City]_blocks.shp
|               [City]_blocks.shx

|               [City]_NBS_rectangles.cpg
|               [City]_NBS_rectangles.dbf
|               [City]_NBS_rectangles.prj
|               [City]_NBS_rectangles.shp
|               [City]_NBS_rectangles.shx
```

The **urban stormwater management scenarios** tool comprises several modules for scenario computation and parameter files for the technologies. The design parameters of the technologies, which have been explained in D5.3 “Urban water management scenarios developed”, are in separate files so that they are easily accessible to users to change layer depth or material properties. These files **parameters_BioRetentionCell.py**, **parameters_ConstructedWetland.py**, **parameters_InfiltrationShaft.py** and **parameters_RetentionTreatmentBasin.py** are called within the scenarios modules to pass the parameters of the technologies.

The **compute_inflow.py** module contains the function **read_design_event** to read the excel file in output from the Storm Design Event tool and format the synthetic rainfall event for the following operations. The **runoff_sealed_surf** generates the runoff from the sealed surfaces of the blocks. These functions are called within the modules of the scenarios.

The **decentral_NBS.py** module has the most complex structure, having a larger number of functions implemented. The **sizing_BioRetCell** function takes in input the parameters of the bio-retention cell and computes the minimum surface area for which there is no generation of the overflow volume outside of the bio-retention cell. The required storage volume is determined from the maximum difference between the cumulated runoff volume arriving at the bioretention cell and the volume that the soil layer is able to infiltrate into the storage layer. The maximum area is limited by the NBS rectangle area in output from the urban_blocks tool.

In the function **generate_scenario_decentral_NBS** given the effective NBS area to retain the runoff and the volume of water retained within the blocks, a selection criterion has been developed for the allocation of NBS sites, prioritising those with effective retention of higher runoff volumes. In this way, priority is given to those blocks with extensive sealed surfaces that generate runoff and that have the potential NBS areas to retain the volumes. Using this criterion, the number of planned sites for the scenario is reduced to contain the target volume for the catchment.

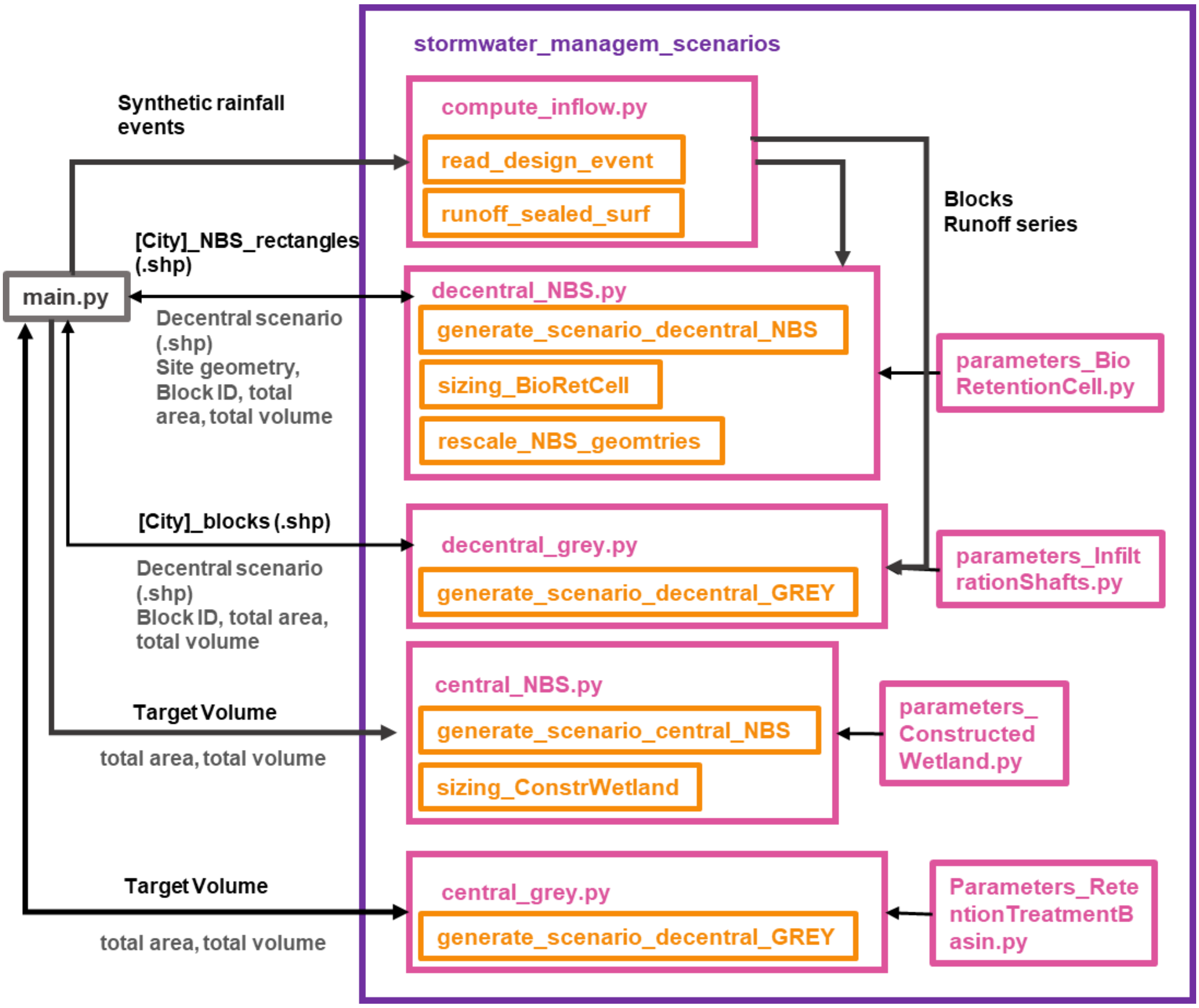
For the selected sites, the function **rescale_NBS_geometries** resizes the geometries of the NBS sites according to the necessary dimensioning for the design event and total target volume within the urban catchment. The geodata frame with the dimensioned NBS area is exported for each design storm scenario. The decentral scenarios are also exported into an Excel file to visualise the data relative to the block, the NBS area and the retained volume for storm design events.

In the **decentral_GREY** module, the block runoff is again first computed through the compute_inflow.py module. The infiltration shafts are sized to be distributed homogeneously along the blocks of the urban catchment, consistently accounting for the runoff generated within the blocks and up to the target volume retention of the urban catchment. The blocks with a sealed surface area of less than 50 m^2 are excluded, as they would generate for a 5-year return period a runoff volume below the minimum capacity of the infiltration shaft. Within the function **generate_scenario_decentral_GREY**, the geodata frame of the scenario is exported with centroids as geometry for the infiltration shafts, sized in proportion to the retained volume from the blocks.

The **central_NBS** and **central_GREY** modules read in input the target stormwater volume to reduce at the urban catchment. In **sizing_ConstrWetland** function, the method of the maximum retention volume is implemented given the sewer inflow volume and the outflow rate characteristic of the constructed wetland. The outcome gives similar results to those using the hydraulic loading rate parameter for sizing. Currently, no geodata frame is exported to the user by the module; only the information on the total area and total volume of the constructed wetland is returned. Also, in the **generate_scenario_central_GREY** function, the total area and total volume are returned from the sizing of the retention treatment basin by detaining the whole target volume from the sewer.

For the semi-decentralised scenarios, preliminary considerations and results are reported in *D5.3 - Urban water management scenarios developed*. However, the workflow for this scenario is not reported here since it is strictly dependent on the HDM (cf. section 2.6), and there might be further developments.

Figure 8 Modules and functions of the Urban Stormwater Management Scenarios tool



2.4 Cost Module

COST MODULE

Description

The aim of the Cost module is to derive project costs at their present value and annual costs accounting for the long horizon of the project

Input data

-Investments, Operation & Maintenance, Reinvestment costs

-Useful life time of the infrastructure

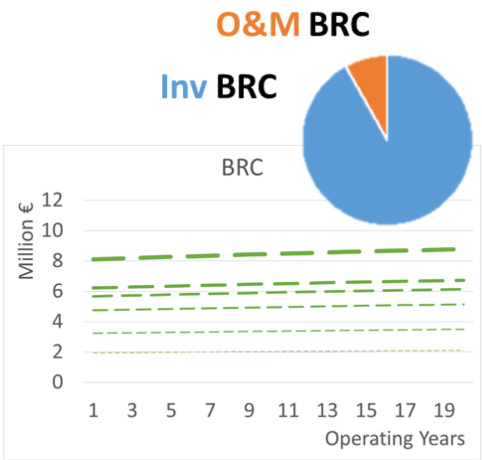
Output data

- Project cost present value PCPV
- Annual Costs AC

Products

-Comparison of project costs for urban stormwater management scenarios given the number and sizing of infrastructures

-Project cost for the useful life time of the infrastructure



2.5 Octopus

OCTOPUS

Description

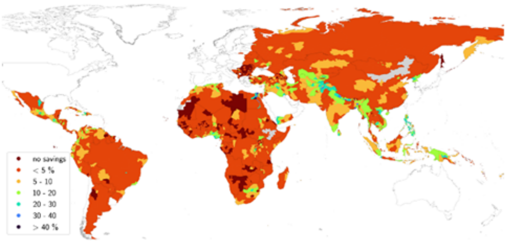
OCTOPUS identifies regional treatment plant clusters by optimizing settlement networks.

Input data

- **Global settlement data**
- **Administrational boundaries**
- **Cost assumptions**

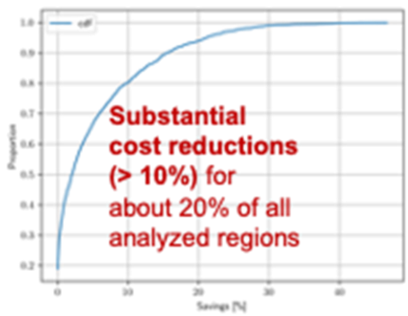
Output data

- **Cluster network maps per study region**
- **Cost saving estimates**



Products

- **Optimized network maps**
- **Cost reduction per region**
- **WWTP cluster maps**

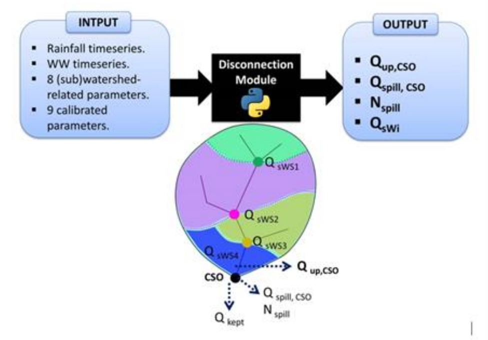


2.6 Hydraulic Disconnection Module

HYDRAULIC DISCONNECTION MODULE

Description

The aim of the Disconnection Module is to quantify the volume needed to be disconnected to meet the local regulation about Combined Sewer Overflow (CSO) management.



Input data

-Rainfall time series
-Dry wastewater flow time series
-8 (sub)watershed parameters
-9 calibrated parameters

Output data

-Watershed and sub-watersheds outflow
-CSO discharge at the watershed's outflow
-Number of days with CSO discharge

Products

- CSO reduction scenarios by varying input parameters at the catchment
-Identification of the sub-watershed(s) to be disconnected in priority
-Volume to be disconnected per sub-watershed to meet CSO objective regulation-wise

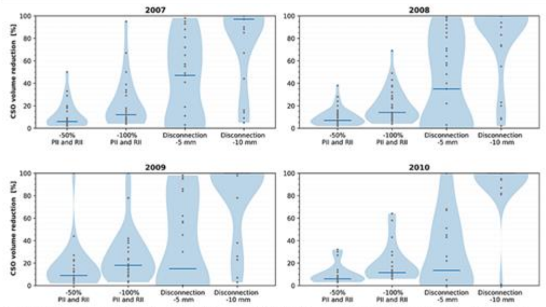
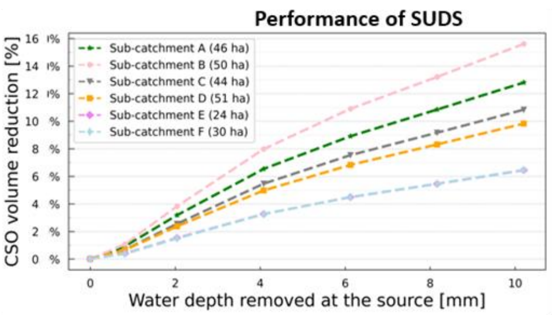


Fig. 12. CSO volume reduction for four different scenarios. Two permanent infiltration inflow and rain-induced infiltration (PII and RII) reduction scenarios were considered, a partial and complete reduction of PII and RII of the sewer. Two disconnecting scenarios: infiltrating the first 5 and 10 mm of surface rainfall runoff. Four years were simulated (from 2007 to 2010). The CSO volume reduction is represented by violin plots displaying the distribution and density of the data at different values. The data represent all rainfall events that occurred during the study years which resulted in a CSO event.



Lenormand É., Montoya Coronado V.A., Álvarez Velásquez M. J., Lipeme Kouyi G. (2023) Disconnection module. MULTISOURCE Milestone MS16, H2020 grant no. 101003527

Montoya-Coronado, V. A., Tedoldi, D., Castebrunet, H., Molle, P., & Kouyi, G. L. (2024). Data-driven methodological approach for modeling rainfall-induced infiltration effects on combined sewer overflow in urban catchments. Journal of Hydrology, 130834.

2.7 Technology Selection Tool

TECHNOLOGY SELECTION TOOL

Description

The aim of the Technology Selection tool is to provide the User with the most appropriate nature-based solutions for his water management needs

Input data

-Type of water
-Inflow
-People served
-BOD concentration
-Available surface
-Temperature, Climate
-Ecosystem services preferences, technical constrains [opt]

Output data

Optimal technology with:
-Surface required
-Costs
-Co-benefits
-O&M required
-Energy required
-Manpower, Skills
-Risks
Case studies

Products

Enhanced Natural Treatment Solution App

https://nat4wat.icradev.cat/home

3.0 PLANNING PLATFORM WORKFLOWS

As mentioned in the overview section, the use of the Planning Platform and the selection of tools used for planning urban stormwater strategies might vary depending on the specific objectives and the data availability of the specific urban context.

3.1 Decentral stormwater potential

The workflow of the Planning Platform applied to the Ecully catchment, Lyon, is shown in the code below. This workflow assesses the maximum volume of stormwater that the urban catchment can retain if it is completely diverted from the sewer system. Therefore, this type of workflow is entirely based on the **urban_block** and **design_storm_events** tools, plus the module **decentral_NBS** from the tool **storm_managem_scenarios**, and is implementable with readily available urban data. Below is reported the workflow to compute the decentral stormwater potential for the Ecully Catchment.

```
import os
from pathlib import Path
from planning_tools.urban_environment import geodata_transform as gdtr
from planning_tools.urban_environment import urban_blocks as urb
from planning_tools.urban_environment import urban_NBS_area as urb_nbs
from planning_tools.storm_design_events import generate_design_event_from_time_series as gen_event
from planning_tools.stormw_managem_scenarios import decentral_NBS as dc_NBS

# USER’s INPUT#################################################################

# define the city name to plan with the Planning Platform
city_name= ‘Ecully’
# define the name of the URBAN ATLAS geopackage downloaded at:
https://land.copernicus.eu/en/products/urban-atla
urban_atlas_name='FR003L2_LYON_UA2018_v013.gpkg’
# define the name of the WSF raster file downloaded at: https://download.geoservice.dlr.de/WSF2019/
WSF_name='WSF2019_lyon.tif’
# define whether the Planning Platform should be applied to only one urban catchment, and the boundary is
provided (.shp). note also that CRS should be in WGS84 - UTM zone
boundaries = 'Ecully_catchment_epsg32631.shp'

# define return period for the storm design events of the Planning Platform
return_periods= [5, 10, 20, 30, 50, 100]
# define the level of data availability to compute the stormwater design events
storm_data_availability=0
#storm_data_availability=0 only rainfall time series from a local station is available. The time
resolution should be sub-hourly and series longer than 30 years.
#storm_data_availability=1 tabulated values of design storm events are available (e.g. COSTRA SERVICE for
Germany). Provide Excel file with design storm events.
#storm_data_availability=2 The user runs the Planning Platform on a synthetic stormwater time series that
has at disposal (single event)

rain_file_name = "30 years Rain 6min.xlsx" # define the name of the file. Time should be sub-hourly and
Precipitation in mm
```

```
header_rows= 6 # define the number of rows in the Excel file before the time series starts
date_col = “Date” #Enter the name of the series corresponding to the Date - Time
precipitation_col = “27 CHAMPAGNE MONT D’OR” #Enter the name of the series corresponding to Precipitation
start_year=1992
end_year=2022
years_to_exclude=[2005, 2006, 2008, 2009, 2010] #Enter the years that have incomplete series or anomalies
so that should be excluded by the analysis
# in case storm_data_availability=1 define
#design_events_existing_file

#COMPUTATION URBAN BLOCKS AND URBAN NBS AREA

working_directory = os.path.abspath(os.getcwd())
ua_path = working_directory + '\\data\\raw\\urban_environment\\' + urban_atlas_name
rst_path= working_directory + '\\data\\raw\\urban_environment\\' + WSF_name
bound_path= working_directory + '\\data\\raw\\urban_blocks\\boundaries\\' + boundaries
hist_rain_path= working_directory + '\\data\\raw\\design_storm_event\\' + rain_file_name
rst_rp_path= working_directory +'\\data\\processed\\reprojected_' + city_name + '.tif'
rst_hr_path= working_directory +'\\data\\processed\\reprojected_higher_res' + city_name + '.tif'

epsg = gdtr.epsg_finder(city_name)
dstCrs = {'init': f'EPSG:{epsg}'}
gdtr.reproject_raster(rst_path, dstCrs, rst_rp_path)
upscale_factor=2
gdtr.upscale_rst(rst_path, upscale_factor, rst_hr_path) #upscale_data==1 if values of raster with higher
resolution have to be upscaled

export_path=working_directory +'\\data\\export\\urban_environment\\'+ city_name
blocks_path, potential_area_path= urb.compute_blocks(city_name, rst_hr_path, ua_path, epsg, export_path,
bound_path)
NBS_rect_path = urb_nbs.find_NBS_area(potential_area_path, rst_hr_path, export_path, step=1)

#COMPUTATION STORM DESIGN EVENTS

match storm_data_availability:
    case 0:
        design_event_excel_path=gen_event.extreme_events_frequency_analysis (hist_rain_path, header_rows,
date_col, precipitation_col, start_year, end_year, years_to_exclude, working_directory)
        print('Excel file created in ' +  design_event_excel_path)
    case 1:
        design_event_excel_path = working_directory +'\\data\\processed\\' + design_events_existing_file

gen_event.generate_synthetic_rainfall (design_event_excel_path, working_directory)

#COMPUTATION DECENTRAL STORMWATER POTENTIAL
rain_design_path= working_directory+ '\\data\\export\\Synthetic_rainfall_events.xlsx'

gdf_NBS_rect = gpd.read_file(NBS_rect_path)
```

```
gdf_max_conn = gpd.read_file(potential_area_path)

for rp in return_periods:

    index = return_periods.index(rp)

    #compute decentral stormwater potential for largest rectangle area
    NBS_area, Vol_Ret = dc_NBS.sizing_BioRetCell(rp, rain_design_path, gdf_NBS_rect)
    gdf[f'NBS_area_{rp}yT']=NBS_area
    gdf[f'VolRet_{rp}yT']=Vol_Ret
    resized_polygons=rescale_NBS_geomtries(gdf, return_period)
    gdf_decentral_NBS = gpd.GeoDataFrame(geometry=resized_polygons, data=gdf.data, crs=gdf.crs )
    gdf_decentral_NBS.to_file(export_path + f'NBS_rect_potential_scenario_{rp}yT' )

    #compute decentral stormwater potential for max open connected area
    NBS_area, Vol_Ret = dc_NBS.sizing_BioRetCell(rp, rain_design_path, gdf_NBS_max_conn)
    gdf[f'NBS_area_{rp}yT']=NBS_area
    gdf[f'VolRet_{rp}yT']=Vol_Ret
    resized_polygons=rescale_NBS_geomtries(gdf, return_period)
    gdf_decentral_NBS = gpd.GeoDataFrame(geometry=resized_polygons, data=gdf.data, crs=gdf.crs )
    gdf_decentral_NBS.to_file(export_path + f'max_conn_Area_potential_scenario_{rp}yT' )
```

Figure 6 Results of the decentral stormwater potential workflow within the largest NBS rectangle area. The maps reported are the exported geospatial files for the retention of design storm events with 5 years return period and 100 return period.

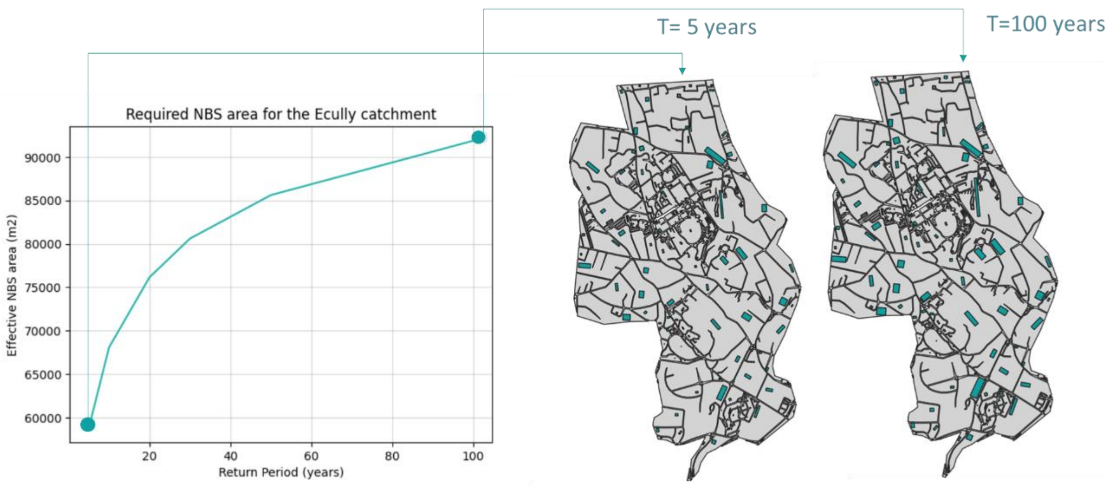


Figure 7 Results of the decentral stormwater potential workflow within the maximum open connected area. The maps reported are the exported geospatial files for the retention of design storm events with 5 years return period and 100 return period.

3.2 Ecully urban stormwater scenarios

The workflow of the Planning Platform, always applied to the Ecully catchment, to compute the urban stormwater scenarios is shown in the code below. This workflow generates the grey/NBS and central/decentral urban stormwater management scenarios, taking the CSO spill as the target volume to be reduced in an equivalent manner in each scenario. This type of workflow is therefore entirely based on the **urban_block** and **design_storm_events** tools, **storm_managem_scenarios**, plus the results of the simulations of the Hydraulic Disconnection Module for the synthetic rainfall events. The first part of the code is the same as the one reported above, thus it is illustrated the code to import and export the results of the storm_managem_scenarios tool. The results of this Planning Platform workflow are illustrated in the deliverable *D5.3 - Urban water management scenarios developed*.

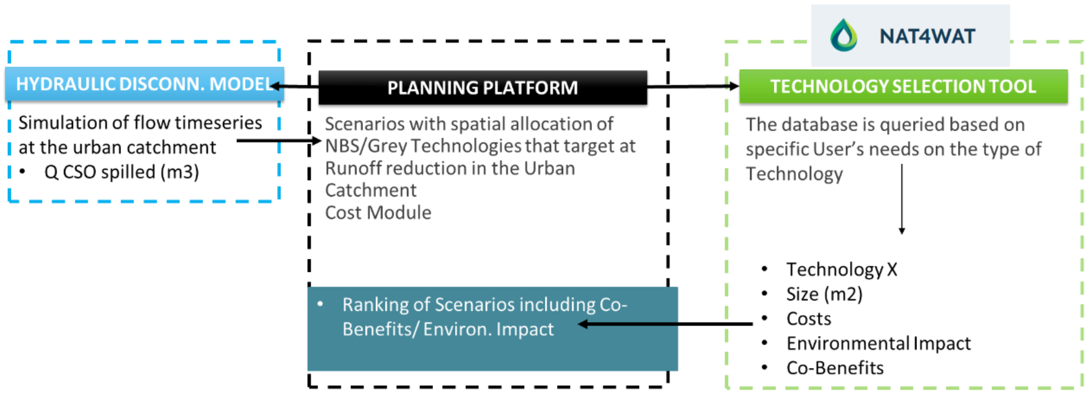
```
from planning_tools.stormw_managem_scenarios import decentral_NBS as dc_NBS
from planning_tools.stormw_managem_scenarios import decentral_GREY as dc_GREY
from planning_tools.stormw_managem_scenarios import central_NBS as ce_NBS
from planning_tools.stormw_managem_scenarios import central_GREY as ce_GREY

# USER’s INPUT######################################################################
# define total target Runoff volume to reduce at the Urban Catchment
Target_Vol_scenarios=[4832, 8817, 13039, 15530, 18683, 22985] #Stormwater volume to reduce for design
event are output from the HDM

##############################################################################

#COMPUTATION URBAN WATER MANAGEMENT SCENARIOS

cso_design_path=working_directory+ '\\data\\export\\RESULTS_SIMULATION_PROJECT_RAINS.xlsx'
export_path=working_directory +'\\data\\export\\urban_stormwater_management_scenarios\\'

results_dec_NBS=[]
results_dec_GREY=[]
results_ce_NBS=[]
results_ce_GREY=[]

for rp in return_periods:
```

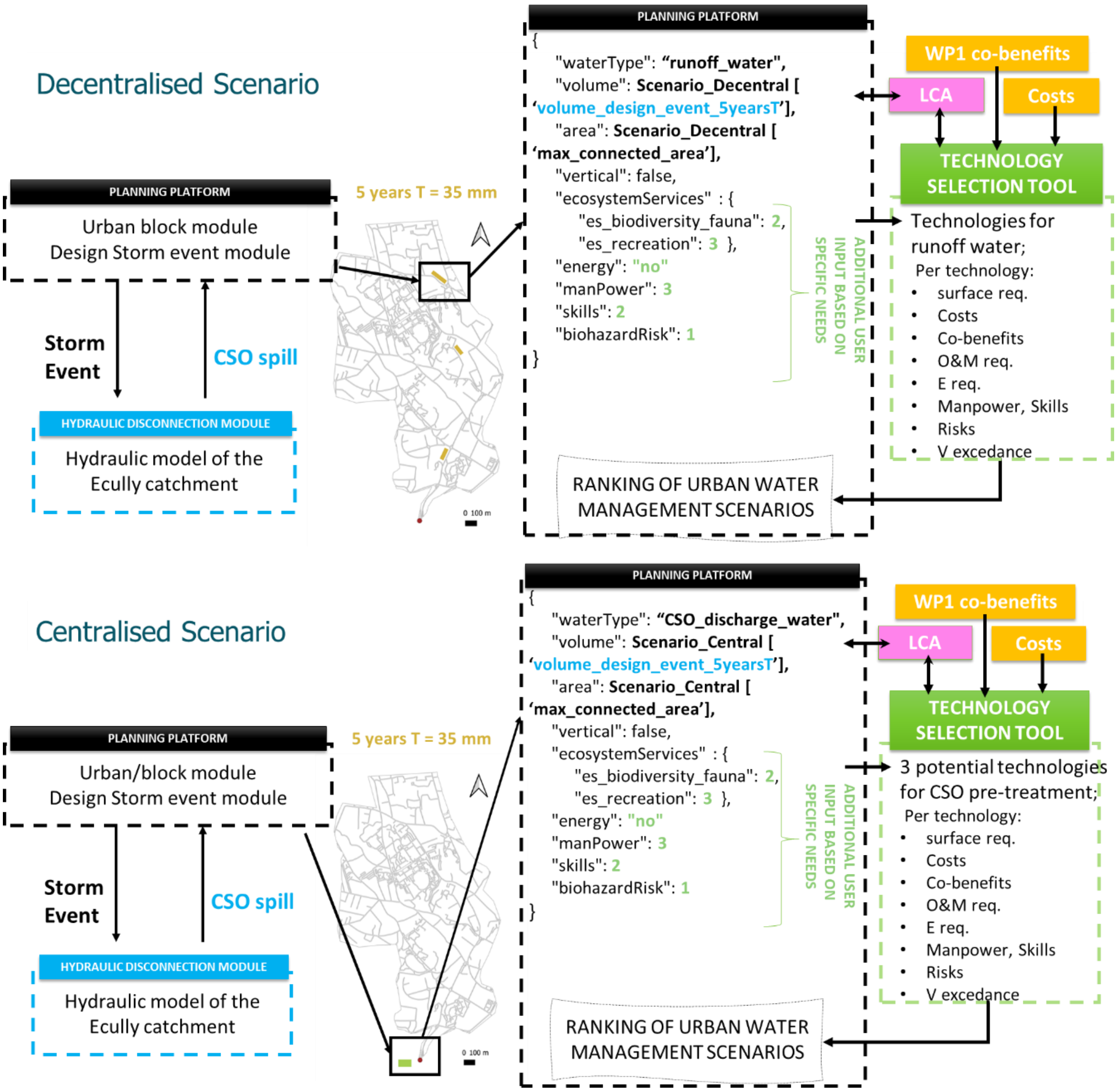
```
    index = return_periods.index(rp)

    total_volume, total_area, gdf_decentral_NBS = dc_NBS.generate_scenario_decentral_NBS (rp,
Target_Vol_scenarios[index], rain_design_path , NBS_rect_path, export_path)
    result_dec_NBS.append(total_volume, total_area)
    total_volume, total_area, gdf_decentral_GREY = dc_GREY.generate_scenario_decentral_GREY (rp,
Target_Vol_scenarios[index], rain_design_path , NBS_rect_path, export_path)
    result_dec_GREY.append(total_volume, total_area)
    total_volume, total_area = ce_NBS.generate_scenario_central_NBS (rp, Target_Vol_scenarios[index],
cso_design_path , export_path)
    result_ce_NBS.append(total_volume, total_area)
    total_volume, total_area = ce_GREY.generate_scenario_central_GREY (rp, Target_Vol_scenarios[index],
cso_design_path , export_path)
    results_ce_GREY.append(total_volume, total_area)
```

Links to the other MULTISOURCE tools are being developed for automation. For the Hydraulic Disconnection Model developed by INSA, the combined use of the tools to plan stormwater management strategies in other cities will depend on the transferability of the HDM model to other urban catchments, as INSA is testing their tool for Girona, and also it will depend on the public distribution of the HDM tool. For the technology selection tool, codes have been developed to query the NAT4WAT database through their supplied API. Due to changes in the latest version of their software, the query parameters need to be revised, and this will be done in the upcoming tasks, given that the prototype of Planning Platform is complete and urban stormwater management scenarios have been developed so that quantitative and qualitative benefits can be estimated based on the Planning Platform outcomes.

Below are reported workflows to link the Planning Platform with the Hydraulic Disconnection Module and the Technology Selection Tools for the decentralised and centralised scenario of the Planning Platform. The code writing for data exchange will likely require in-person working sessions between the developers of the MULTISOURCE tools.

Figure 6 Workflows of the linkages among the tools





4.0 OUTLOOK

Some of the different modules and workflows are part of ongoing PhD theses or are being prepared for publication. The final python repositories will be published via GitLab until the end of the project. An exemplary link for OCTOPUS (Friesen et al. 2023) in GitLab for distribution, is provided here as an example: https://git.ufz.de/friesen/octopus.

REFERENCES

DWD Climate Data Center (CDC): Grids of return periods of heavy precipitation (design precipitation) over Germany (KOSTRA-DWD), version 2010R.

Boeing G. (2017). OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, environment and urban systems, 65*, 126-139.

Friesen J, Sanne M, Khurelbaatar G, and van Afferden M. (2023). "OCTOPUS" principle reduces wastewater management costs through network optimization and clustering. *One Earth, 6*(9), 1227-1234.

Montoya-Coronado V, Tedoldi D, Castebrunet H, Molle P, and Kouyi GL. (2024). Data-driven methodological approach for modeling rainfall-induced infiltration effects on combined sewer overflow in urban catchments. *Journal of Hydrology*, 130834.

The overall goal of MULTISOURCE is to, together with local, national, and international stakeholders, demonstrate a variety of about Enhanced Natural Treatment Solutions (ENTS) treating a wide range of urban waters and to develop innovative tools, methods, and business models that support citywide planning and long-term operations and maintenance of nature-based solutions for water treatment, storage, and reuse in urban areas worldwide. The project includes seven pilots treating a wide range of urban waters. Two individual municipalities (Girona, Spain; Oslo, Norway), two metropolitan municipalities (Lyon, France; Milan, Italy), and international partners in Brazil, Vietnam, and the USA will contribute to each of the main project activities: ENTS pilots, risk assessment, business models, technology selection, and the MULTISOURCE Planning Platform. The use of urban archetypes in the Planning Platform will enable users to quickly classify regions (in both developed or developing countries) suitable for the application of nature-based solutions for water treatment (NBSWT) and compare scenarios both with and without NBSWT.

MULTISOURCE

enhanced natural treatment solutions

This project has received funding from the European Union's Horizon H2020 innovation action programme under grant agreement 101003527.