ECOLOPES

ECOlogical building enveLOPES: a game-changing design approach for regenerative urban ecosystems

H2020-FET-OPEN-2021-2025

Action number 964414

D3.2: Draft ECOLOPES platform architecture

Dissemination level:	Public
Contractual date of delivery:	Month 19, 31 October 2022
Actual date of delivery:	Month 19, 31 October 2022
Work package:	WP3
Task:	T3.2, T3.3 and T3.4
Type:	Report
Approval Status:	Final version
Version:	v1.0
Number of pages:	54
Filename:	D3.2_DraftEcolopesPlatformArchitecture_20221031_v1.0.docx

Abstract: The objective of D3.2 is to report on the current state of the work carried out in work package 3 (WP3) after 19 months. The deliverable describes the developments and component integration of the ECOLOPES platform, and it presents first results of the elaborated frameworks such as the computational framework (section 3), the knowledge generation framework (section 4 and 5), and the cloud-based Rhino.Compute framework with the ECOLOPES front-end tool (section 6). Additionally, to contextualise the ECOLOPES computational platform, the conceptual framework is introduced in section 2.

Funded by the European Union

VERSION HISTORY

Version	Date	Reason	Revised by
v0.1	14.09.2022	ToC	Verena Vogler, Jens Joschinski, Wolfgang Weisser
v0.2	21.09.2022	ToC	Verena Vogler, Luis Fraguada
v0.3	04.10.2022	Draft 1	Verena Vogler, Jens Joschinski, Francesca Mosca
v0.4	20.10.2022	Draft 2	Verena Vogler, Jens Joschinski, Francesca Mosca, Luis Fraguada, Surayyn Uthaya Selvan, Shany Barath
v0.4	20.10.2022	Draft 3	Verena Vogler, Jens Joschinski, Luis Fraguada, Surayyn Uthaya Selvan, Shany Barath, Francesca Mosca
v0.5	27.10.2022	Draft 4	Verena Vogler, Jens Joschinski, Luis Fraguada, Surayyn Uthaya Selvan, Shany Barath, Francesca Mosca, Michael Hensel, Defne Sunguroglu Hensel
v1.0	29.10.2022	Final	Verena Vogler, Luis Fraguada, Jens Joschinski, Wolfgang Weisser

AUTHOR LIST

Organisation	Name	Contact information	Contribution
MCNEEL	Verena Vogler	verena@mcneel.com	Main text all sections, figures, GH definitions
SAAD	Jens Joschinski	jens.joschinski@animal-aided-design.de	Ecological Model, KGF, graphs, C++ script
UNIGE	Francesca Mosca	francesca.mosca@edu.unige.it	Section 6.4, GH definition
MCNEEL	Luis Fraguada	luis@mcneel.com	Review, Proofreading, 6.3.2 GH connection
TECHNION	Surayyn Uthaya Selvan	surayyn@campus.technion.ac.il	KGF, design optimisation, GH definition
TECHNION	Shany Barath	barathshany@technion.ac.il	Review
TU VIENNA	Michael Hensel	michael.hensel@tuwien.ac.at	Future design generation, Section 3.4.1
TUM	Wolfgang Weisser	wolfgang.weisser@tum.de	Proofreading, introduction
TUM	Defne Sunguroglu Hensel	defne.hensel@tum.de	Future ontology, Section 3.4.2

EXECUTIVE SUMMARY

The ECOLOPES platform aims to provide design decision support, evaluation, and generation/optimisation of ecological building envelopes. This report addresses theoretical and practical research and development efforts conducted in WP3 for the implementation of the first draft of the ECOLOPES platform and its technical components.

First, the conceptual framework is introduced which helps to guide the development of the ECOLOPES platform. It identifies knowledge gaps and key questions through domain-specific and interdisciplinary reviews on the existing literature, technologies, methods and data (section 2).

Then, the report updates the platform's computational framework (section 3), and reports on the progress regarding software components and environments for generating, evaluating, and optimising designs.

Afterwards, the report presents initial results of the Knowledge generation framework (KGF) that provides missing correlations between architectural, environmental and ecological aspects for design evaluation and optimisation (section 4). The interim Knowledge base (KB) is presented as an outcome from the KGF experiments (section 5). As a repository for the newly generated knowledge it supports human experts/ ecologists in design decision-making.

Lastly, Section 6 of the report presents the draft version of the ECOLOPES computational platform and its system architecture. The section reports on the deployment and integration of the draft version including the backend and frontend tool (ECOLOPES Grasshopper plugin).

The report concludes by drawing future applications (e.g. for a case study site in Vienna), plans for the next stage of the development and integration of the ECOLOPES platform.

ABBREVIATIONS AND ACRONYMS

AEC	Architecture, engineering and construction industry
AFG	Animal functional groups, i.e. groups of animal species with similar effects on ecosystem functioning
AHP	Analytical Hierarchy Process (algorithm)
AI	Artificial intelligence
CAD	Computer-aided design
Computational workflow	The overall process, implemented in the ECOLOPES platform, and composed of chained software components that together resolve ECOLOPES design cases.
Computational framework	The is a computational system as well as applications that bridges the gap between frontline research and practical efforts.
CSV	Comma-separated values (file format for exchange of tabular data)
ECOLOPES	Ecological building envelopes
ECOLOPES platform	A set of interoperable tools and software components that together support the ECOLOPES approach for the design of *envelopes*.
FG	Functional groups
GH	Grasshopper, a visual programming interface for Rhino
KB	Knowledge base
JSON	JavaScript Object Notation (file format for data exchange)
KGF	Knowledge generation framework
KBS	Knowledge-based system
KPI	Key performance indicators
MADM	Multi-attribute decision-making (algorithm)
ML	Machine learning
MOO	Multi-objective optimisation (algorithm)
OWL	Web ontology language (language for knowledge representation)
PFG	Plant functional groups, i.e. groups of plant species with similar effects on ecosystem functioning

RDF	Resource description framework (file format for exchange of graph data)
Rhino	Rhinoceros, a 3D free-form NURBS modelling software and cross-platform open developer platform
Sandbox	Sandbox is a cloud-based playground for testing software components in an agile process.
SDK	Software development kit
SQL	Structured query language (language for relational database queries)
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution (algorithm)
UI	User interface
WP	Work package

TABLE OF CONTENTS

1. INTRODUCTION

In ECOLOPES, we propose a radical change for city development: instead of minimising the negative impact of urbanisation on nature, we aim at urbanisation to be planned and designed such that nature - including humans - can co-inhabit the city. We envisage a radically new integrated ecosystem approach to architecture that focuses equally on humans, plants, animals, and associated organisms such as microbiota. In our ECOLOPES EU FET project, we focus on the envelope, the building enclosure, as it is the buffer zone between the inside and outside that has been treated so far as a single function entity - a barrier between the inside and outside. We will transform the envelope into an *ecolope*, a multi-species living space for four types of inhabitants: humans, plants, animals, and microbiota. ECOLOPES will develop the core technologies for designing *ecolopes* in a systematic way, considering the needs of both humans, as well as of plants, animals and beneficial microbes. To do so, ECOLOPES will make biological knowledge available for the architectural design process, to find architectural solutions that enable synergies and limit conflicts between the inhabitants. The *ecolopes* designed by this multi-species approach will restore the beneficial human - nature relationships in cities.

The progress of the project until month 12 has been summarised in Deliverable D1.3. (Report of year 1), and the results were also presented in Deliverable 4.1 (Technical Report), in preparation of the first review meeting of ECOLOPES in May 2022. A key feature of ECOLOPES is the development of a design approach that is supported by a computational framework implemented in the ECOLOPES platform. In Deliverable 3.1. (Prototype technical requirements report) submitted at month 12, the initial version of the ECOLOPES platform (first prototype: sandbox) was described. Deliverable D3.1 described how a range of expert data-bases, an information model and algorithmic processes and tools are intended to be combined, to result in a data-driven design recommendation system. The ECOLOPES platform makes new knowledge available for design. It includes frontend tools for design, modelling and visualisation, and a computational simulation environment that enables iterative design development integrated with multi-criteria decision-making strategies. Deliverable 3.1 also describes the MiMo experiment which has now been developed into the Knowledge Generation Framework (KGF), an important component of the ECOLOPES platform that helps to generate knowledge on the relationship between architecture and ecology. Motivation and design of the KGF are described in detail in D4.1 and D3.1.

In this deliverable D3.2 we describe the further development of the ECOLOPES platform architecture. In particular, the deliverable provides:

- a more detailed description of the conceptual framework that guides the development of the computational framework (computational workflow) and the Knowledge Generation Framework (KGF), and ultimately the ECOLOPES platform
- the embedding of the platform architecture in the framework
- the further development of the Knowledge Generation Framework (KGF)
- the interim knowledge base (KB) on a cloud server and its current content, in particular the first data for testing the interoperability of the Grasshopper definitions
- The further development and current integration of major technical component in the ECOLOPES platform, in particular
    - the ecological model, described in more detail in D4.1, that is now integrated in a 3D CAD modelling environment
    - A Grasshopper definition for the shading analysis, as part of the ecological analysis and the Knowledge Generation Framework KGF
    - A Grasshopper definition of soil depth modelling (also part of the KGF)
    - A Grasshopper definition of voxelizing 3D geometry for raster inputs
    - A Grasshopper definition to visualise raster output in 3D CAD
    - A Grasshopper definition for human comfort analysis
    - A Grasshopper definition to read and use Key Performance Indicators (KPIs) for optimisation (originally described in D1.3)
- a case study example for the adaptability of the ECOLOPES platform (Vienna case study).

2. The conceptual framework of the ECOLOPES computational platform

This section presents the conceptual framework for the ECOLOPES computational platform. It integrates the main objectives and expectations for the ECOLOPES platform based on a literature review, a review of the state-of-the art in design platform development (see D3.1, section 2), as well as interdisciplinary and domain-specific methods and data. This conceptual framework guides the development of the computational framework (computational workflow) and the Knowledge Generation Framework (KGF). The three frameworks help to build the ECOLOPES platform (Figure 1).

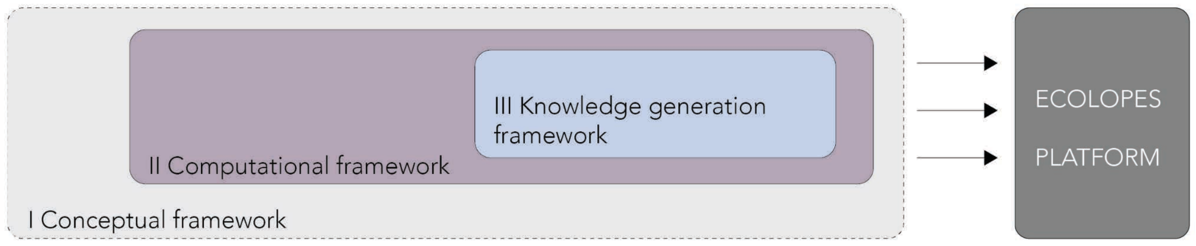

III Knowledge generation framework
II Computational framework
I Conceptual framework
ECOLOPES PLATFORM

Figure 1: The three frameworks that help to build the ECOLOPES platform.

2.1 Description of the conceptual framework

The conceptual framework, that forms the basis for the development of the ECOLOPES platform, describes the expected relationships between the ECOLOPES variables, i.e., the architectural and ecological parameters for designing an *ecolope*. It defines the relevant platform objectives of the research and development process in ECOLOPES and maps out how they come together to draw coherent conclusions. Thus, it is a representation of the relationship we expect to see between these variables, characteristics, or properties of a building envelope with respect to ecological performance. The conceptual framework was developed based on four reviews of the literature, of methods in the various domains of ECOLOPES, and of data (Figure 2).

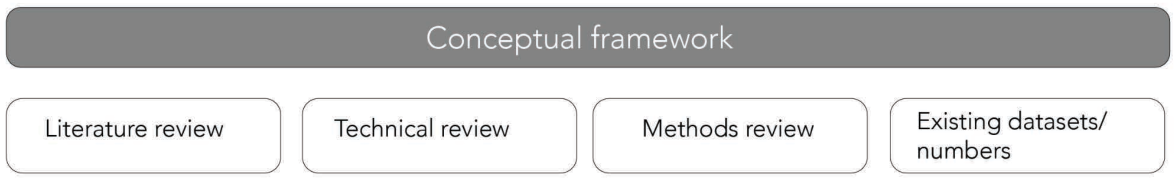


Figure 2: The development of the conceptual framework was based on a number of reviews.

Literature reviews of existing studies about regenerative urban ecosystems and the role of design (described in more detail in D4.1 as a product of WP4) gathered existing knowledge about modelling plant and animal populations, the role of soil on buildings for plant growth and the use of functional groups in modelling. In addition, it reviewed how the urban environment can be suitably classified for modelling. Finally, a review was conducted on how nature can be evaluated, to be able to later test for human acceptance of an *ecolope*. The ecological knowledge accumulated in these reviews resulted in a list of relationships, parameters and datasets that need to be included into the ECOLOPES platform, to e.g. make it possible that an architect creates and artificial nests of the right shape and height to potentially attract a particular bird.

A technical review (described in more detail in section 2 of D3.1 of WP3) discussed the state-of-the-Art for digital data-driven urban planning platforms, concluding that there is an increasing number of decision-support tools for real-time urban design and urban analysis. These cloud-based tools integrate expert systems from interdisciplinary fields (e.g., from catastrophe and climate modelling) and that can be accessed from a web browser as a web application. This review also identified gaps in the existing platforms that need to be developed for the ECOLOPES platform.

A review on the methods already used and needed in the various domains, was carried out by interdisciplinary discussions in WP3, WP4 and WP6. The review found that e.g. multiple tools and datasets already exist to represent environmental conditions in architectural workflows (e.g. illumination, noise), but that models that incorporate ecological community dynamics currently do not exist in the AEC sector. Existing ecological community models like RangeShifter (Bocedi et al. 2021) or Fate-HD (Boulangeat et al. 2014), on the other hand, do not incorporate the spatial scale and available data of urban environments.

A review of necessary datasets, carried out in WP4 and WP6 (and that is discussed in more detail in section 2.1 in D4.1), consisted of a general screening of crowd-sourced and open-databases with data needed for *ecolope* design. The review helped to identify what data are available in what spatial resolution for design tasks at a very local scale, i.e. a building in a city. It was found that the use of some data was restricted due to either non-availability for some of the planned case studies, or in other parts of the world, or the wrong spatial resolution, i.e. mostly too coarse. This necessitated the identification of key parameters required to initiate the computational workflow. A dataset table was compiled including primary datasets with the right spatial resolution and availability for workflow initiation. The selected datasets were validated by the consortium experts (see Appendix A).

Based on the reviews the **following key questions and challenges** were identified that need to be addressed by the new design approach and technology developed in the ECOLOPES project.

1. How to conceptually and computationally correlate architecture and ecological performance? And vice versa, how to consider architectural aspects such as vertical and elevated construction elements such as façades and green roofs, material and high density human spaces within an ecological model?
2. How to bring the newly generated knowledge into an informed design decision making system?
3. What are the relevant key performance indicators (KPIs) from architecture and ecology to appropriately evaluate an ecological-informed design outcome for an *ecolope*?

2.2 Development of the computational framework

From the above, a number of conclusions and necessary developmental steps towards a computational framework can be derived:

The user of the ECOLOPES platform can be both an architect/ urban planner but also an ecologist. However, environmental, ecological, and architectural analysis results and *ecolope* design outcome can only be generated if the many different computational components such as the ecological model and the design generation component are connected in a meaningful and efficient manner. The ECOLOPES computational framework that supports the development of the ECOLOPES platform was developed to meet these challenges. The goal of the computational framework is to connect the different computational components from the environmental, ecological, and architectural modelling practices. Such an undertaking requires the implementation of domain-specific (computational) methods to allow corresponding data to pass through the system and to support a holistic design system. The computational framework has been described in D3.1 and it is further developed below (section 3).

From the conceptual framework and the consideration for the computational framework, a number of requirements for the design of the ECOLOPES platform can be derived. We identified six main challenges that need to be met in order to proceed from the conceptual framework to the implementation of a project-specific computational framework:

(1) User requirements need to be defined and translated into technical requirements for the development of the ECOLOPES platform (see also D3.1, Appendix).

(2) As an interdisciplinary approach, data relevant for the architectural design of building envelopes (e.g., design brief, legal framework) as well as data for urban ecology modelling (e.g., local species pool, plant succession, climate data) need to be accessible for the user and brought into the system (open and expert datasets). Based on these datasets, computational components need to be capable to import, filter and process only ECOLOPES relevant data for designing an *ecolope*.

(3) Most ecological data, analysis, and spatially explicit simulation models represent settings in 2D (e.g., species distribution models, RangeShifter, RFate), while architectural data and urban areas are modelled in 3D. Compressing 3-dimensional data into 2D space is a challenge, as is simulating ecological models in 3D space.

(4) The development and implementation of individual computational components has to be divided into different tasks for each consortium partner, to be able to build on their specific expertise. The outcome has to align to the overall development approach for the ECOLOPES platform and the ambitions of the project.

(5) Due to the complexity of ecological and environmental processes, the processing time for individual analysis and simulation steps limits efficient use. Comparable ecological and environmental simulations run in the order of minutes to hours, and are therefore

not suitable if real-time simulation of results is desired. It is thus a task for the ecological modelling group (WP4) to develop shortcuts as done with the environmental models (D4.1) and generate knowledge that can be stored in and accessed in real time from a Knowledge Base (KB). Generating this knowledge is the task of the Knowledge Generation Framework (KGF, see 4.1., still called the "MiniModel"). The current state of the KGF and the interim KB are described in sections 4 and 5.

In summary, there are a significant number of challenges to be addressed and overcome for the development and the technical implementation of the ECOLOPES platform. These challenges include besides technical aspects (massive datasets, complexity), aspects of interdisciplinary collaboration, as well as the complex and temporal nature of natural systems versus the modelling legacy of 3D CAD systems.

2.3 Conclusions

In summary, the conceptual framework has helped to identify key questions and tasks for the development of the Computational framework and the Knowledge generation framework (KGF), which guides the development of the ECOLOPES platform. The thorough domain-specific and interdisciplinary reviews on the existing literature, technologies, methods and data was vital for the development of the frameworks and has helped to shape the developmental process towards the ECOLOPES platform. As a result, a number of novel approaches are now possible, such as the inclusion of an ecological analysis in a 3D CAD environment for architectural design.

3. The Computational Framework and Its Integrated Components

The computational framework bridges the gap between the conceptual approaches, research and practical efforts. It is unique in showing the interdisciplinary nature of the ECOLOPES research project and the way in which it interacts with emerging technologies and techniques/methods. Thus, this section introduces the computational system envisioned to address challenges including big data, high performance, databases, information systems, integrated and embedded hardware/software components, and networks.

3.1 The computational system and the ontology-driven generative design

The computational system is composed of individual modules developed by each partner. For instance, while the local and regional ecological (green) are developed by the team of ecologists, the form generation and optimisation environment is developed by computational architects (pink). Figure 3 shows the updated version of the computational workflow and all the components that are integrated (black frame).

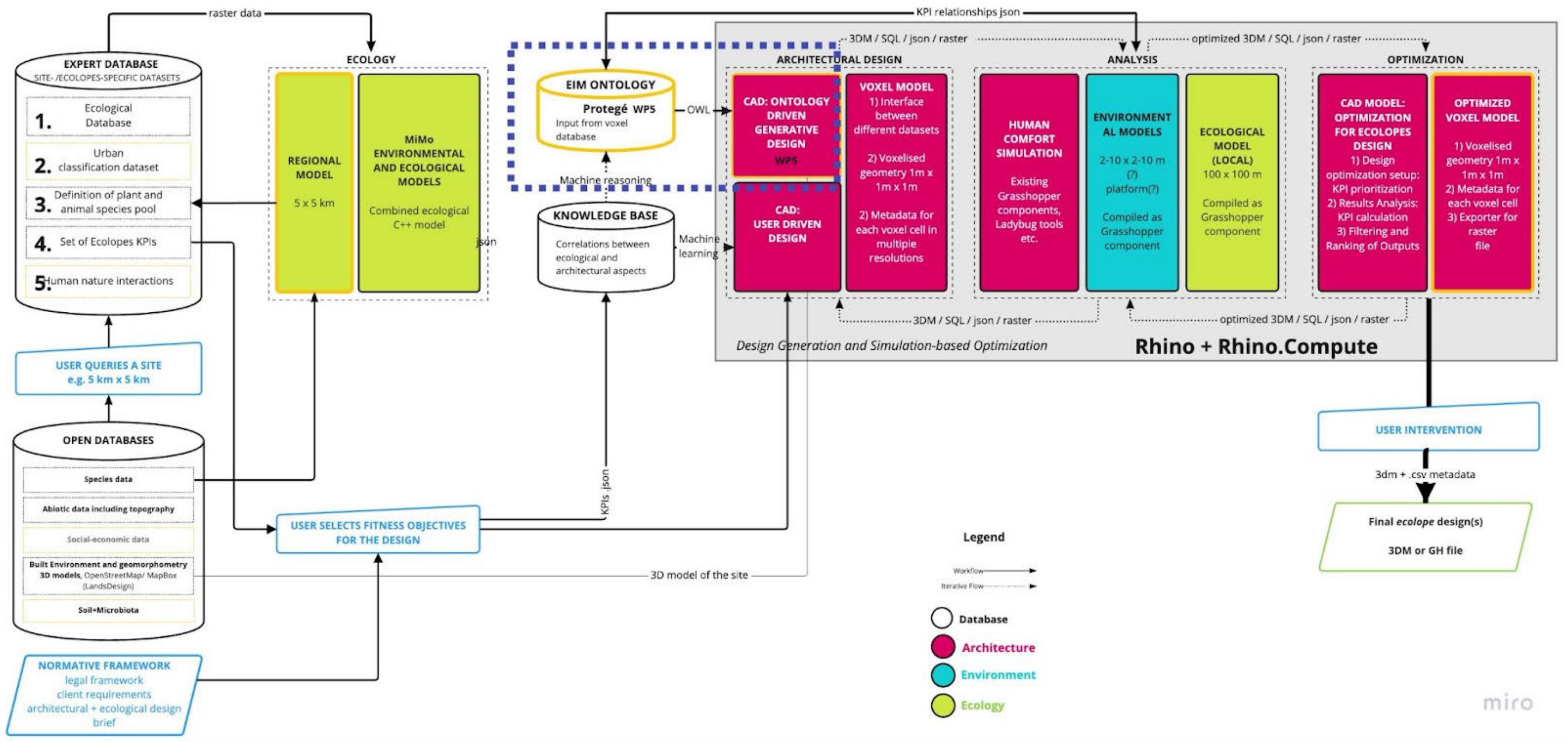

raster data
EXPERT DATABASE
SITE- /ECOLOPES-SPECIFIC DATASETS
1. Ecological Database
2. Urban classification dataset
3. Definition of plant and animal species pool
4. Set of Ecolopes KPIs
5. Human nature interactions
USER QUERIES A SITE e.g. 5 km x 5 km
OPEN DATABASES
Species data
Abiotic data including topography
Social-economic data
Built Environment and geomorphometry 3D models, OpenStreetMap/ MapBox (LandsDesign)
Soil+Microbiota
NORMATIVE FRAMEWORK
legal framework
client requirements
architectural + ecological design brief
ECOLOGY
REGIONAL MODEL
5 x 5 km
MiMo ENVIRONMENTAL AND ECOLOGICAL MODELS
Combined ecological C++ model
json
USER SELECTS FITNESS OBJECTIVES FOR THE DESIGN
KPIs json
3D model of the site
EIM ONTOLOGY
Protegé WP5
Input from voxel database
OWL
Machine reasoning
KNOWLEDGE BASE
Correlations between ecological and architectural aspects
Machine learning
KPI relationships json
3DM / SQL / json / raster
optimized 3DM / SQL / json / raster
ARCHITECTURAL DESIGN
CAD: ONTOLOGY DRIVEN GENERATIVE DESIGN
WP5
CAD: USER DRIVEN DESIGN
VOXEL MODEL
1) Interface between different datasets
2) Voxelised geometry 1m x 1m x 1m
2) Metadata for each voxel cell in multiple resolutions
ANALYSIS
HUMAN COMFORT SIMULATION
Existing Grasshopper components, Ladybug tools etc.
ENVIRONMENTAL MODELS
2-10 x 2-10 m (?) platform(?)
Compiled as Grasshopper component
ECOLOGICAL MODEL (LOCAL)
100 x 100 m
Compiled as Grasshopper component
OPTIMIZATION
CAD MODEL: OPTIMIZATION FOR ECOLOPES DESIGN
1) Design optimization setup: KPI prioritization
2) Results Analysis: KPI calculation
3) Filtering and Ranking of Outputs
OPTIMIZED VOXEL MODEL
1) Voxelised geometry 1m x 1m x 1m
2) Metadata for each voxel cell
3) Exporter for raster file
3DM / SQL / json / raster
optimized 3DM / SQL / json / raster
Design Generation and Simulation-based Optimization
Rhino + Rhino.Compute
USER INTERVENTION
3dm + .csv metadata
Final ecolope design(s)
3DM or GH file
Legend
Workflow
Iterative Flow
Database
Architecture
Environment
Ecology
miro

Figure 3: Computational workflow, including the ontology/ ontology-driven generative design that will be integrated when developed (blue dotted line, TU Vienna)

The core of the computational workflow is the ecological model and the work needed to make ecological analysis available for design decision making and design optimisation. The part developed by TU Vienna (blue dotted line, Figure 3) focuses on the development of a machine reasoning system based on a 2.5D voxel model, an ontology, and an ontology-driven design generation process. The computational workflow and the ontology driven design approach are developed in parallel due to the different requirements for each knowledge domain to develop their modules. In order to allow for bringing in designs that have not passed through the ontology, the computational workflow enables users to analyse their designs based on the ecological and environmental models for optimisation.

There are five largely technical challenges for developing the ECOLOPES platform that are further described in this section: (1) big data, (2) high performance, (3) databases, (4) information systems, and (5) the integration and embedding of hardware and software modules.

(1) ***Big data:*** A number of large datasets including e.g., climate data, species pools, topographical models, and land use data, need to be processed. The ECOLOPES platform needs to help data custodians govern and secure big datasets while empowering end users to analyse that data. Thus, there need to be strategies that can be fulfilled within the scope of the project to guarantee a secure and efficient system/ platform to make big datasets available for processing and analytical purposes.

(2) ***High performance:*** Ecological and environmental analysis of 3D CAD models, and design iteration for KPI- based optimisation are computationally very heavy processes. For instance, the ecological models currently require approximately three hours to simulate a simple ecological community on a 100 x 100 m *ecolope* over the lifetime of 50 years (see section 7.2), a solar radiation analysis of a 1m x 1m x 1m voxel cell in a 3D model requires up to 10 min, and design optimisation requires a couple of hours to compute a representative amount of design iterations for design optimisation. The cloud-based infrastructure of the ECOLOPES platform (D3.1), addresses the required high performance issues of the computational framework and places powerful computational capabilities in the hands of researchers and engineers.

(3) ***Database maintenance:*** Database performance can slow down applications with an impact on end-users. Besides storing and accessing huge volumes of data, it is crucial to solve database performance issues, e.g., by structuring data coming from public data sources in a suitable way. This comes along with the tasks of data management, security, reliability, and uptime of multiple databases.

(4) ***Information systems:*** The ECOLOPES information system turns raw data into meaningful information that can be used for decision making in the design process. The main challenges for the information system are as follows: generating new knowledge from knowledge, data, and information analysis (Knowledge generation

framework, Machine learning and machine reasoning); the input and the maintenance of up-to-date information (data maintenance); and usability (user interface).

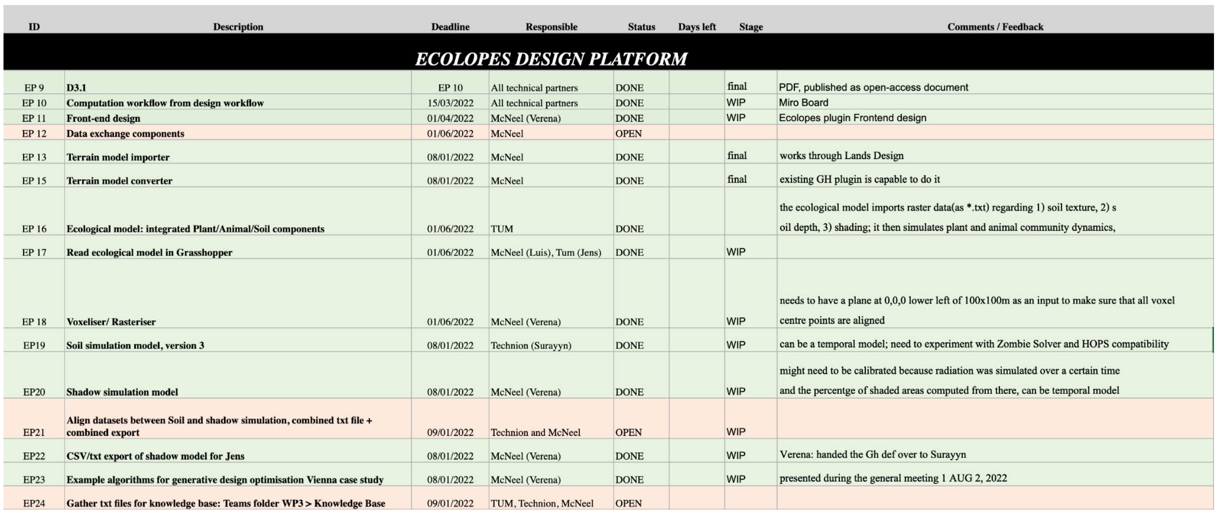
(5) ***Integrated and embedded hardware/ software modules:*** For the development of an interdisciplinary design technology, there are multiple software and hardware modules being used for various purposes. They need to work hand in hand to act as a single system to perform a certain task. This is where the challenges of software integration come into play, especially interoperability, the generation of a common runtime environment for modules written in different programming languages, long running processes on the cloud, data exchange formats, and the integration of an information system within a 3D CAD environment. Additionally, the challenge to integrate different licence types of each module needs to be addressed.

In summary, the challenges described need to be addressed for the ECOLOPES platform development and for the integration of components. In the following sections, integration strategies, integrated modules for the computational workflow, and future integrated modules (ontology-driven design) are described.

3.2 Software development and integration strategies

As described in D3.1, section 5, the software development approach adopted in ECOLOPES takes into account the multiple types of applications, their maturity, intended usage, and lifecycle. As the ECOLOPES project progresses, we continue in WP3 to manage software development based on an **incremental approach** rather than an agile or waterfall system. That way, specific sections or components of the platform are designed, prototyped, integrated, and evaluated in each iteration. This approach also allows us to design and **develop several sections in parallel**, a process supported by the early definition of the system architecture, common data models, and interfacing mechanisms. To ensure consistency with respect to development and integration McNeel coordinates weekly WP3 workshops and a monthly meeting that reports on the latest developments. Furthermore, action points and their state are tracked within a shared Excel table (Figure 4).

This software development approach and integration strategy is reflected in parallel development approaches for the computational workflow and the ontology-driven design. This strategy guarantees an operating system, even though the ontology will be developed and integrated later.



ID	Description	Deadline	Responsible	Status	Days left	Stage	Comments / Feedback
ECOLOPES DESIGN PLATFORM							
EP 9	**D3.1**	EP 10	All technical partners	DONE		final	PDF, published as open-access document
EP 10	**Computation workflow from design workflow**	15/03/2022	All technical partners	DONE		WIP	Miro Board
EP 11	**Front-end design**	01/04/2022	McNeel (Verena)	DONE		WIP	Ecolopes plugin Frontend design
EP 12	**Data exchange components**	01/06/2022	McNeel	OPEN			
EP 13	**Terrain model importer**	08/01/2022	McNeel	DONE		final	works through Lands Design
EP 15	**Terrain model converter**	08/01/2022	McNeel	DONE		final	existing GH plugin is capable to do it
EP 16	**Ecological model: integrated Plant/Animal/Soil components**	01/06/2022	TUM	DONE			the ecological model imports raster data(as *.txt) regarding 1) soil texture, 2) s oil depth, 3) shading; it then simulates plant and animal community dynamics,
EP 17	**Read ecological model in Grasshopper**	01/06/2022	McNeel (Luis), Tum (Jens)	DONE		WIP	
EP 18	**Voxeliser/ Rasteriser**	01/06/2022	McNeel (Verena)	DONE		WIP	needs to have a plane at 0,0,0 lower left of 100x100m as an input to make sure that all voxel centre points are aligned
EP19	**Soil simulation model, version 3**	08/01/2022	Technion (Surayyn)	DONE		WIP	can be a temporal model; need to experiment with Zombie Solver and HOPS compatibility
EP20	**Shadow simulation model**	08/01/2022	McNeel (Verena)	DONE		WIP	might need to be calibrated because radiation was simulated over a certain time and the percentge of shaded areas computed from there, can be temporal model
EP21	**Align datasets between Soil and shadow simulation, combined txt file + combined export**	09/01/2022	Technion and McNeel	OPEN		WIP	
EP22	**CSV/txt export of shadow model for Jens**	08/01/2022	McNeel (Verena)	DONE		WIP	Verena: handed the Gh def over to Surayyn
EP23	**Example algorithms for generative design optimisation Vienna case study**	08/01/2022	McNeel (Verena)	DONE		WIP	presented during the general meeting 1 AUG 2, 2022
EP24	**Gather txt files for knowledge base: Teams folder WP3 > Knowledge Base**	09/01/2022	TUM, Technion, McNeel	OPEN			

Figure 4: Incremental development approach led by MCNEEL.

For the integration, McNeel provided a cloud-based backend infrastructure (D3.1, "The ECOLOPES sandbox", section 7) which is partly a Linux Server (data storage and KB) and a Windows Server running Rhino.Compute, a cloud-based processing environment for geometry models. Furthermore, Rhino is an open development platform for third-party plugin development (Rhino Development, 2018) including a royalty-free Software Development Kit (SDK) as well as guides that can be applied across platforms – Windows and OS X.

3.3 Integrated modules of the computational workflow

There are four major modules of the computational workflow that are already integrated within the ECOLOPES platform: (1) Open and expert databases, (2) the ecological model, (3) the Knowledge base, and (4) the 3D CAD and optimisation environment. Each module responds to specific data and functional requirements.

(1) ***Open and expert databases (WP3-WP7):*** Site-specific environmental data such as open climate data (e.g., .epw, .stat, .txt, .ddy) can be accessed through open street maps in Grasshopper/ Ladybug Tools and used for solar radiation analysis to evaluate adequate areas for plant growth.

(2) ***Ecological model (WP4):*** The ecological model evaluates environmental information (3.3.1) to simulate plant, animal, and soil community development on the site over the life cycle of the *ecolope*. It includes soil suitability for plants, plant demography, competition for light, seed dispersal and herbivory, formation of animal home ranges, and animal mortality. The outputs (spatially and temporally varying plant and animal biomass) will be used to evaluate whether the design meets the ecological and architectural objectives (KPIs). The biggest achievement made since the last deliverable is that the ecological model is now fully integrated in Grasshopper, a visual algorithm editor for Rhino, a standard 3D CAD software in AEC (see D3.1, section 2.4). As a parametric modelling tool, Grasshopper can manipulate geometry to generate multiple design solutions which are informed and analysed by the ecological model.

The ecological model provides variables to define architectural and ecological performance indicators (KPIs) that facilitate multi-objective optimisation processes in Grasshopper to find the most appropriate design solution. Furthermore, the integration of ecological modelling within CAD facilitates the generation of new knowledge through experiments (section 4).

(3) ***Knowledge base (KB) (WP3):*** The ECOLOPES Knowledge base (KB) (see D3.1, section 3.3) collects information and resources generated through the Knowledge Generation Framework (see D3.1 and D3.2, section 4). The computational system in ECOLOPES is dependent on the KB as its repository for the knowledge needed to support decision-making. As a knowledge-based system (KBS), it analyses knowledge, data, and other information from various sources to generate new knowledge. The principal purpose is to capture the knowledge of human experts/ ecologists to support design decision-making (section 4).

(4) ***Design optimisation environment (WP6):*** The optimisation environment will consist of two multi-criteria decision-making (MCDM) strategies, namely, multi-objective optimisation (MOO) and multi-attribute decision-making (MADM). The MOO will produce optimised design solutions while MADM will facilitate the selection of top-performing designs based on architectural and ecological priorities. These strategies will be integrated into the Grasshopper environment by employing existing plugins. Generated volumes from a) a designer or b) resulting from the generative design phase will be optimised using evolutionary solver plugins such as Wallacei or Octopus. The genomes of the evolutionary solvers will be represented by design parameters that describe the geometry of the generated volumes (e.g., surface area, surface inclination, surface roughness). The fitness objectives will be represented by key performance indicators (KPIs) used to evaluate architectural and ecological performances of the generated volumes. Currently, some of these KPI values will be computed using existing Grasshopper plugins that analyse abiotic environmental factors (i.e., Ladybug and Honeybee) or to simulate soil placement using physics simulation plugins (i.e., Kangaroo). KPIs that are not computed within Grasshopper will be accessed from the Knowledge base using .csv or .xlsx reader plugins in Grasshopper. The optimised design solutions will then undergo MADM to identify the best performing solution(s). Algorithms such as the Analytical Hierarchy Process (AHP) or Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) will be constructed using the component in the Maths tab within Grasshopper. By inputting the optimised values of the key performance indicators and identifying weights for each key performance indicator, the MADM Grasshopper script will be able to sort the range of optimised design solutions according to given architectural or ecological objectives.

3.4 Future integrations (WP5)

In this section we present the proposed integration of ontology-driven design. The resulting software components will connect with the open and expert databases and with the KB generated from the computational workflow. The form generation algorithms will be developed within the existing Grasshopper environment (.gh file type) to guarantee interoperability with the rest of the platform. The ontology will be developed in Protegé.

3.4.1 Envisioned ontology-aided design generation environment (TU Vienna)

The ontology-aided design generation environment consists of three elements: (1) the voxel model that integrates design process related data, (2) the EIM ontology that can be queried by the designer, and (3) the two-part generative design process consisting of a *translational* process and a *generative* process, both of which are linked with the ontology.

Project-specific input data for the generative design process is contained in the voxel model and the CAD model. The voxel model is an SQL database. The data contained in the voxel model can be derived from open and expert data bases, the ecological model, the knowledge generation framework, and the ontology. In addition it is often necessary to generate datasets that are not already available. This can include data-generation via simulations in GIS or CAD environments. Only data that can be expressed spatially is contained in the voxel model.

To enable the generation of design variety, i.e. a number of different design solutions that can be ranked and evaluated, it is necessary to extend single value data contained in the SQL database into ranges of values. This is for now done through a three step process:

1. Definition of a range of valid values (for example valid values for an angle: x ϵ [0;360])
2. Definition of thresholds (for example thresholds for an angle based on angle types)
3. Definition of a possible range of values (following the principle for defining the thresholds based on angle types)

The ontology-aided design generation consists of two linked parts: (1) the *translational* process and (2) the *generative* process. In the *translational* process design requirements are, wherever possible, spatialized and then correlated such that a synthesised input into the generative process is derived. Design requirements stated in the design brief are developed into specific datasets that operate as inputs and constraints for the *generative* process. This involves identifying and spatializing items and relations in the form of functional *networks* that are informed and complemented by the dataset *maps* (see Report 5.1 sections 1.5 and 3, and report 4.1 section 3.2). *Networks* or *nodes* and relations have corresponding items in the ontology expressed as knowledge graphs.

Correlations of requirements are structured and visualised using networks and matrices. Data is assigned to different "layers" of information that are represented by a correlation chart (Excel spreadsheet) and its corresponding *network* that consists of items and the relations between them. A correlation chart supports decision-making about relationships between items. Each item has an index that is needed to create networks.

The designer can start by selecting and localising individual items (*nodes*) or by selecting predefined *network* types and defining and localising *nodes* of the selected *network(s)*, i.e. via drop down menus for *nodes* and for *networks* in Grasshopper. *Nodes* and *networks* in the drop-down menus are organised according to different categories (i.e. architectural program nodes / network; stakeholder nodes / networks; etc.). The content of the drop-down menus coincides with items, relations and networks contained in the ontology. This makes it possible for the designer to query the ontology when selecting and relating *nodes* and *networks*.

The development of a classification system for nodes and relations is currently underway along the following system:

- **NodeClass:** defines the general classification of a node (physical/non-physical entity; assigned, flexible or associated location, etc.)
    - **NodeLayer:** defines the information layer of a node (eg. functional building programme, soil volume, biomass volume, etc.)
        - **NodeType:** defines in which (main) data type the node is represented (integer, float, string, boolean)

From the spatialization of nodes and relations, networks are created in the Rhino Grasshopper environment. Selected nodes have a corresponding algorithmic procedure that serves to spatialize and / or relate nodes to each other. This process is supported by the voxel model (SQL database) that contains corresponding datasets (*maps*) that can be called by the designer or by the ontology as an outcome of a query. The *translation* process is completed when a project-specific set of networks that represent the translation of architectural and ecological design requirements and their corresponding maps has been defined and verified by querying the ontology.

The plug-in "SpaceChase" (https://www.food4rhino.com/en/app/spacechase) is used to create dynamic networks with referenced Rhino points, while the information about the relations is provided through a JSON (=**J**ava**S**cript **O**bject **N**otation) file. The JSON file contains the information from the correlation chart and is a direct input for the Grasshopper definition used to create a graph in Rhino. The JSON file is compiled through a GhPython component. The Python script loads and reads the data contained in the JSON file and creates lists with start points and end points, which define the edges of the network. A start point is created for each item with an index in the JSON file. The outputs of the GhPython component "startPoints" and "endPoints" are read through a list component in Grasshopper, which passes this information on to a line component. “SpaceChase” was used as a bridge between 3D points in Rhino, which are the nodes of the network and can be assigned with a name tag and area, and the edges, which are an input from a JSON file and created in Grasshopper. In this way, the position of nodes can be dynamically changed, while the connections remain unchanged.

The *generative* process serves to locate volumes and partition volumes (e.g., architectural, biomass, and soil volumes) and to generate geometry described as *landform* (see report 5.1 sections 1.4, 2 and 4 and report 4.1 section 3). For the generative process two cases that are typical for architectural projects are identified that can be executed separately or in sequence. Case 1 concerns the development of larger areas for which construction plots have been defined, while building footprints, FAR, maximum volume, etc have not yet been determined. Primary building volumes need to be defined in terms of number, dimensions, and locations. This is combined with an overall strategy for defining the site and building geometry jointly as variegated landform (primary landform) (see Report 5.1 Sections 1.4, 2 and 4) to enable connectivity and occupation by different species across the site and building(s). Case 2 follows the guidelines of an already existing master-plan and focuses on the more detailed development of an individual building. This involves the spatial organisation and partitioning of the defined possible building volume and is followed by its geometric articulation (secondary landform) to enable connectivity and occupation of different parts of the building envelope.

In Case 1 the volumes are placed by the designer into an initial configuration, but are changed in location and dimension as part of the generative design process. The solution space for volume distribution can be controlled by selecting fixed and changeable parameters. The designer can choose which parameters are “locked”, for instance, when such parameters are defined by existing planning restrictions.

To enable relations between architectural and ecological requirements, three types of volumes are defined: (1) architectural volumes, (2) soil volumes, and (3) biomass volumes.
These volumes feature associated parameters that can be linked with associated parameters of *networks* and KPIs. This enables identification of possible locations of volumes in relation to the previously defined *networks* and *maps*. For instance, solar exposure may be one key parameter associated with the *networks*, *maps*, and KPIs and indicate where on a given site a volume can be located without preventing required conditions. In this context thresholds (data values and ranges) need to be specified that have influence on KPIs and possible design outcomes.

For Case 2 the generative process focuses on generating different subdivisions of the volume together with different geometries for each subdivision option. In this case strategic choices, ecological objectives, and KPIs can be selected to delimit the search space.

In parallel to deriving benchmarks for the generative process from the design brief, the knowledge generation framework, etc. it is necessary to compare design results with current state of the art solutions. In order to develop such supplementary benchmarks we are currently analysing state of the art projects selected for case 1 (i.e., *ACROS Fukuoka Prefectural International Hall*, Fukuoka, Emilio Ambasz, 1995; *The Mountain Dwellings*, Copenhagen, BIG, 2008; etc.) and case 2 (i.e., *Flower Tower*, Paris, Edouard François, 2004; *M6B2 Tour de la Biodiversité*, Edouard François, 2010-16; *Bosco Verticale*, Milan, Stefano Boeri Architects, 2007-14; etc.). Focuses is placed on massing strategy (for case 1) and geometry of the building(s), distance to neighbouring buildings together with orientation and solar exposure, calculation of proportional relation between architectural volume, biomass volumes (plants) and soil volumes, resources provisions (i.e., water system, etc.), species selection, and maintenance strategy of green systems. From this analysis three insights are derived: (1) trade-offs specific to particular design approaches, (2) architectural design validation criteria, and (3) numerical values that serve as benchmarks in the generative process. The findings regarding trade-offs will inform the ontology and the findings regarding design validation criteria and related numerical values will inform the generative design process and the respective development of the related algorithm(s).

3.4.2 Future ontology (TU Vienna)

The software Protégé is used to build the EIM Ontology with OWL2 Manchester Syntax. If necessary some ontology programming might be carried out in Python. To a limited extent the ontology will be open source and will continue to evolve beyond the project time, fully complying with the FAIR principles.

The EIM ontology aids the generative design process that can produce schematic design to define a project-specific search space and input data for design optimization (i.e., in Rhino Grasshopper using Wallacei). As described in 3.4.1 the tasks of the ontology in the early design phase involve aiding:

(1) The configuration of spatialized 3D networks in CAD environment and the generation of corresponding maps;
(2) The spatial distribution of architectural, soil and biomass volumes in CAD environment according to networks and maps;
(3) The generation of primary and secondary landforms in the CAD environment according to networks, maps and volumes.

There are five main sources of data and information that feed into the ontology, including:

- **Ecological model output:** Data generated by the ecological model will be used to instantiate the ontology. The datasets include:
    - The regional species pool: Plant (soil microbiota) and animal functional groups that are in a particular region. This is a list, so can be an Excel or csv file, which can be imported as an instance of the regional species pool class in Protege. This dataset is static, in other terms, provides one-time input for design
    - The local species pool: Plant (soil microbiota) and animal functional groups that can reach an ecolope site. This is a list, so can be an Excel and csv, which can be imported as instances of local species pool class in Protege. It is static, in other terms, provides one-time input for design.
    - Local species distribution and abundance: Plant (soil microbiota) and animal functional groups' spatial and temporal occurrences in an ecolope over design iterations and in a certain timeframe. This is a list, so can be an Excel or csv file or stored in an SQL Database and can be imported as instances of the voxel cell class in Protege. This is spatialized data, which is structured in an SQL Database, where each data point in the list corresponds to a volumetric data point, or a voxel. It is dynamic, thus, the data property values can change at each design iteration and over a specified timeframe and can provide more than one-time input for design.
- **KGF output:** Information generated by the KGF will be used to model relationships between ecological and architectural parameters. Since the output can be stored in a relational database or as RDF triples, they can be imported into the ontology as Excel, csv file or loaded in as RDF file.
- **GIS output:** Georeferenced datasets are stored in an SQL Database. It can provide one-time and more than one-time input for design. There are Protege plugins to connect the ontology with relational databases such as Ontop.
- **CAD output:** Geometric data and analysis results (i.e., using Ladybug Rhino Grasshopper plugin) can be stored in the voxel model (SQL Database) and accessed by the ontology as instances of geometric types (3D CAD geometry-Geomorphon / typological patterns) and environmental variables (i.e., solar radiation data).
- **Design brief output:** The brief will help to specify the design objectives and KPIs (some of which can be spatially explicit and thus be located in the voxel model) and constraints, which can be represented in a table format. This is a list, so can be an Excel or csv file, which can be imported as instances of design objectives, KPIs and constraints classes in Protege.

The tasks are aligned with the three stages of the algorithmic design process, in the generation of ecological building envelopes. The role of the ontology and the corresponding interfaces in each of these steps are as follows:

(1) Networks: The ontology will facilitate representing and reasoning over knowledge graphs. This configures a semantic network where the nodes in the graph are entities, corresponding to the classes, class instances and data property values and ranges, and edges are relationships defined by object and data properties (consistent with networks in 3D CAD) in Protégé. The designer will query the ontology in Protégé and export inference results to configure networks in CAD, which can be exported as csv files that can directly feed into Rhino Grasshopper definitions. Once networks are configured, the ontology will guide generating maps according to network, and thereby will assist configuring datasets in the voxel model (SQL Database).

(2) Volumes: In the second step, the ontology will be queried to capture input into the algorithmic process, whose task at this stage is to spatially distribute volumes in CAD according to networks and maps generated in the previous step. The query results from the ontology can directly feed into the SQL Database, which stores volumetric data (voxel model), and can be exported as csv files that can directly feed into Rhino Grasshopper definitions for example.

(3) Landform: Once the volumes are configured in the voxel/CAD model, according to networks and maps, the last step in the generative process will be to turn architectural and soil volumes into landform geometry. The ontology will use geomorphons / topographic patterns (consistent with the algorithmic setup) to describe geometric form and suggested geometric changes. The competency questions, which will help define the content and scope of the ontology will be established according to the form generation rules. Accordingly, the method for exporting query results to feed into the computational algorithmic process will be chosen. The computed geomorphology data will need to be translated into CAD model data to provide input into design optimisation.

The conceptual approach to the ontology-aided design generation has been laid out above and specific functionalities of the ontology and interfaces with the voxel model and CAD environment that require programming have been identified. TU Wien is now hiring a postdoctoral researcher with computer and information science background, and more specifically expertise in various programming languages related to ontologies (OWL), as well as generative algorithms (Python). The next steps include (1) programming the link between translational process, i.e. configuration of networks, and the ontology, (2) programming the process by which the ontology calls and / or transfers datasets to the voxel model, (3) programming the geometry generating algorithm and the way the ontology interacts with this part of the process.

3.5 Conclusions and recommendations

In summary, section 3 introduced the computational system and its challenges resulting in incremental development and integration strategies for WP3. We also reported on the integration of interdisciplinary modules for enabling data passing through the system. The computational workflow integrates ecological modelling in a 3D CAD environment as a core achievement. The goal is that ecological modelling becomes an intrinsic part of a parametric and data/information-driven design process that considers the requirements of multiple species. The developed system is non-deterministic as it constantly generates knowledge which can be used for more effective design-decision support. However, the system's major challenges are related to performance issues, as each process (ecological analysis, environmental analysis, optimisation) is computationally heavy. Additionally, the generated knowledge in the KB needs to be extracted automatically rather than manually for design analysis and optimisation.

With respect to the ontology-driven generative design approach, interoperability with the ECOLOPES platform will be ensured during the developments. The main challenges for the ontology are to generate (based on the KB) an interface engine that processes data throughout the system that acts similarly to a search engine by locating relevant information based on queries, and to generate a user-interface that allows users to interact with the system. Another challenge is related to integration of the developed system within the Rhino/Grasshopper environment.

4. THE KNOWLEDGE GENERATION FRAMEWORK (KGF)

Section 4 focuses on the Knowledge generation framework[1] (KGF), which is key to generating new knowledge from ecological and architectural data, knowledge, and information analysis effectively enabling the ECOLOPES design approach and technology. Knowledge-based systems such as the KGF are useful for providing expertise to designers, architects, and planners who require it, especially for making more efficient design decisions. In this context, the KGF is useful for providing recommendations for urban planning, and its potential may continue to grow as the technology evolves. The knowledge generation framework is developed by TUM, MCNEEL, SAAD and the TECHNION.

KGF can be understood as a premise for the Computational framework (section 3), because it generates new knowledge about the correlations between architectural and ecological parameters for a new design approach. An example of these relationships are, how the form of a building or building mass increases biodiversity (= total number of functional groups –

[1] The Knowledge generation framework was referred to as MiMo in D3.1.

FG), or how the inclination of a façade fosters biomass in urban neighbourhoods. Thus, the KGF bridges the gap between ecological modelling and CAD. The goal of knowledge generation is to inform the design process towards given ecological and/or human comfort objectives. These objectives are addressed by a) enabling the initiation of architectural and ecological variables to be meaningfully correlated, and b) serving as an open-ended accumulative knowledge generator for the ECOLOPES project (Figure 5).

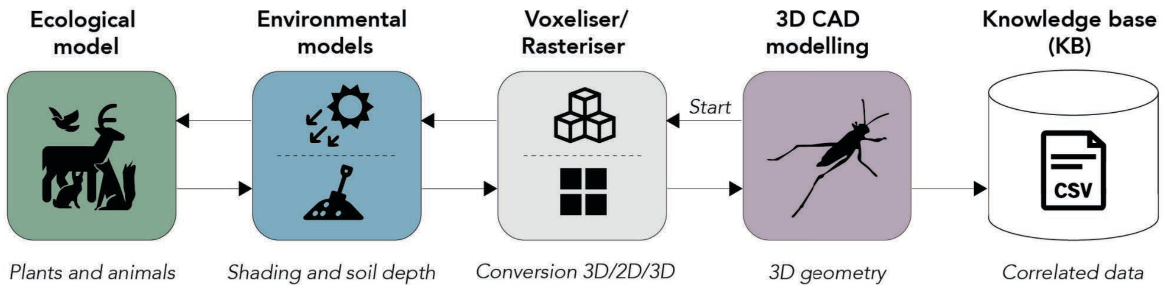


Figure 5: The Knowledge generation framework (TUM, SAAD, McNeel, TECHNION).

The following sections will present the goals of the KGF, methods for the ecological and environmental models, as well as the Grasshopper integration and the solutions to overcome interoperability obstacles between ecological and 3D geometry modelling.

4.1 Set-up of the knowledge generation framework

In D3.1, we described the KGF objectives which were to understand how given building geometries impact environmental conditions (such as soil depth and water retention, radiation input, and general connectivity of the *ecolope*) as well as the structure and composition of the ecological communities (biomass, abundance of functional groups) of the *ecolope.* Therefore, by extracting general relationships between architecture and ecology, these relationships can be used to guide a computational design decision-making process. We defined variables describing the geometry of the building as geometry is expected to influence the environmental conditions on the *ecolope*, and therefore the species that can inhabit it. The method we proposed in D3.1 is now further developed and initial results could be generated.

The overarching objectives in the development of the KGF were (1) to generate new knowledge that will be stored in the Knowledge base (KB), (2) to test and evaluate the layouted computational system, and lastly, (3) to enable the ECOLOPES platform. We have now technically advanced the system so that the KGF can meet these objectives. We developed a system that is linking a geometry typology and mass (architecture) to soil depth, shading (environment), biomass and number of functional groups (ecology) computationally. In the next step, as technicalities mature, the developed process is repeated with changing geometry variables and geometry typologies. Figure 6 shows the associated information as well as the datasets and knowledge involved in this knowledge generation process.

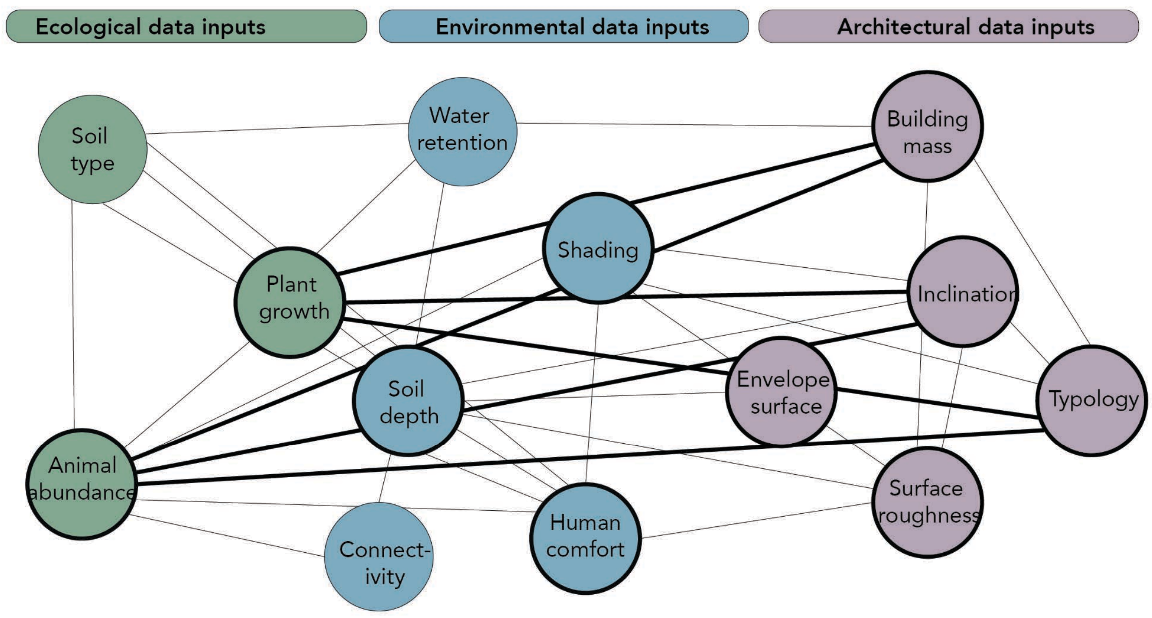


Figure 6: Associated ecological, environmental and architectural information in the KGF. The elements with the black frame are already considered in the code.

The KGF has been tested and initial results reveal first patterns and associations of the relationships between architecture and ecology as well as environmental aspects (section 4.5).

4.2 Adaptation of the ecological and environmental models for KGF

Although the ecological and environmental models are intended to understand complex relationships between biotic, abiotic, site-specific information as well as the form of a building on many levels, they are rather simplified models of ecological communities. Nevertheless, a minimal set of inputs is required, including local species information and information about the site information (currently a 100m x 100m planar site divided into 1m x 1m raster cells): spatialised soil types, a converted 3D model of a selected building form/ building mass with information about shading percentage and build depth for each raster cell. To start generating knowledge and deriving principal relationships, simple versions of the soil, plant and animal submodels are integrated into the KGF, which will be prosynthetic building designs are created, leaving all information apart from the building geometry constant, so only selected variables are allowed to change under certain conditions.

For the generation of knowledge, the ecological model computes for different geometry/ building typologies/ masses the biomass, and number of functional groups over a period of 50 years (yearly time stamps) (Figure 7). Left aside are changing environmental conditions e.g, during seasons, hydrological conditions and aquatic areas on the site, site topography, as well as the capacity of vertical spaces to contain soil (e.g., architectural elements such as buckets), and the material of a façade. The aim of the experiment is to learn more about the role of a building form with respect to ecological and environmental performance. A number of initial questions will be formulated to guide the generation of useful knowledge on the relationship between architecture and ecology, such as *how can surface inclination support soil depth for plant growth?*

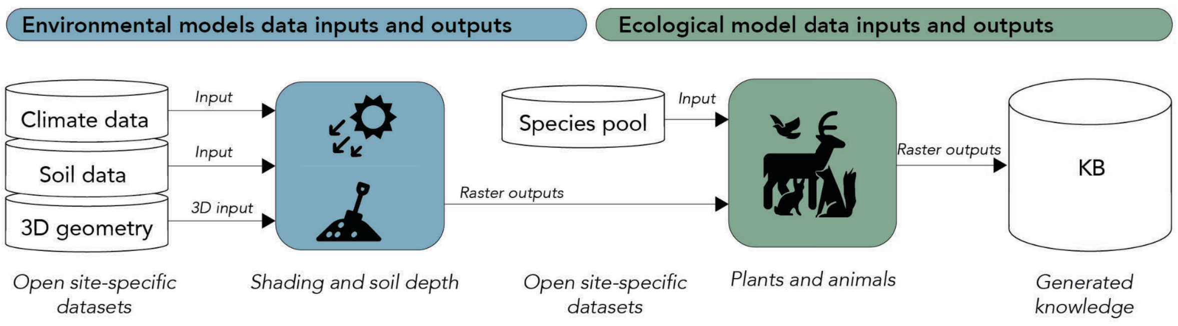
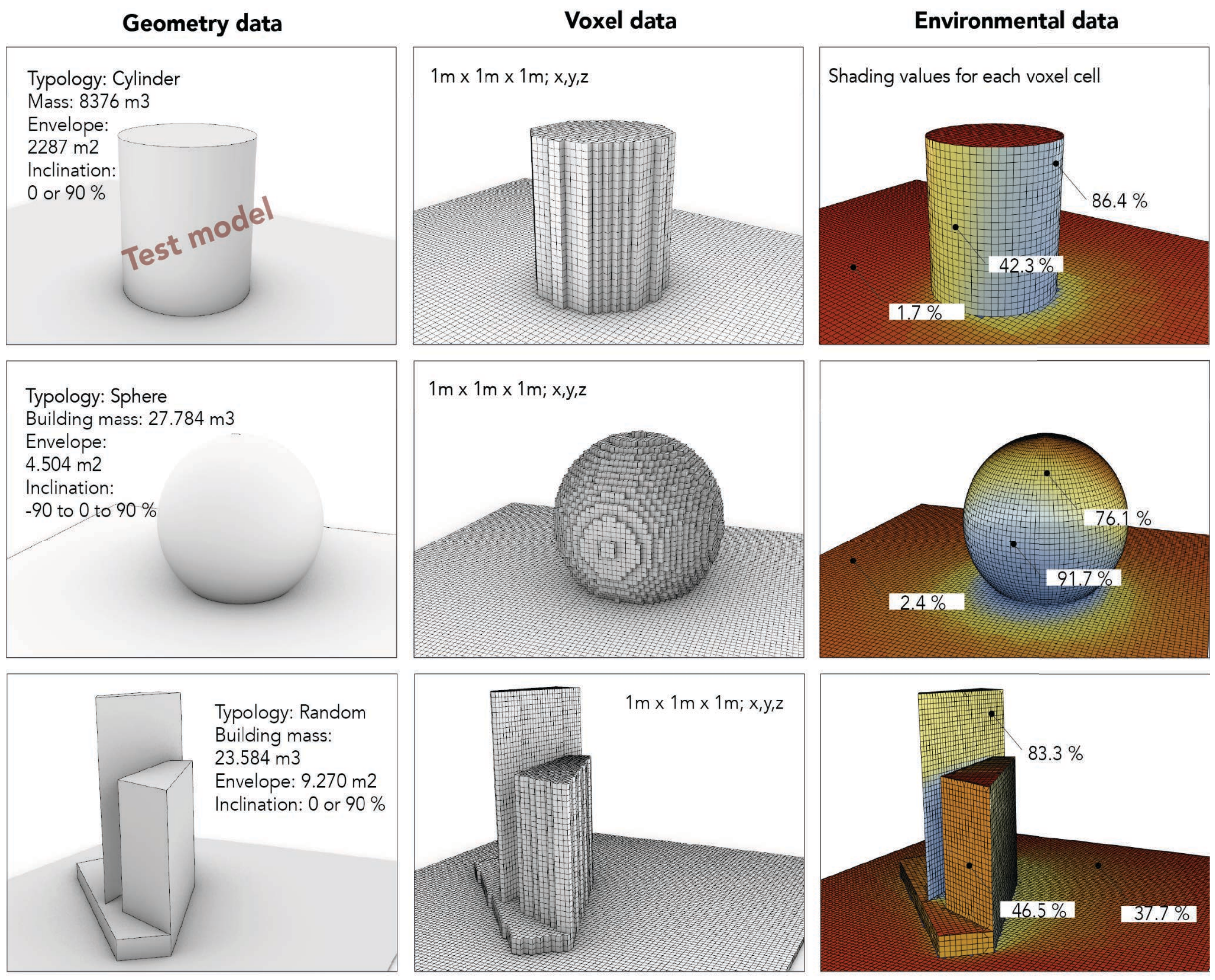


Figure 7: Required input data and outputs for the ecological and environmental models.

4.3 Grasshopper integration methods

To enable the computation of the relationships between building form and environmental and ecological parameters, the form has to be analysed based on these parameters. Thus, ecological and environmental modelling needs to be integrated within a 3D CAD environment. For the KGF, we used Ladybug tools (https://www.ladybug.tools/), an environmental analysis tool in Grasshopper, to analyse different typologies of simple shapes (e.g. cylinder., box, sphere, see D3.1), Shading values (in percentage) were computed for units of the 3D model (voxel units of 1m x 1m x 1m) as the ecological model requires such an input format. The output values (building: true/false; voxel location; shading values; shading true/false) were exported for each voxel cell as a .csv file (Figure 8).

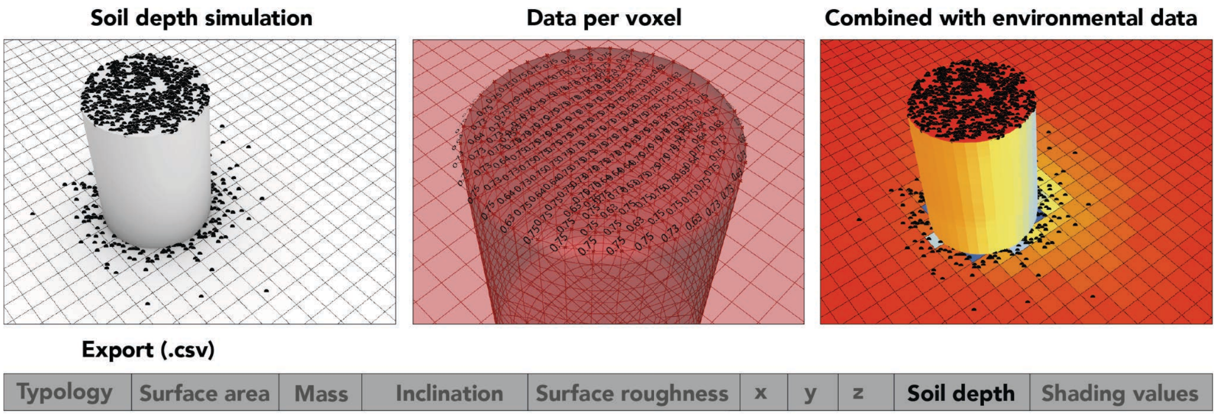


Export (.csv)

Typology	Surface area	Mass	Inclination	x	y	z	Shading values

Figure 8: Computation of output values of shading analysis for different typologies, and .csv export.

Furthermore, a physical simulation model was set up using Kangaroo Physics, a physical simulation engine in Grasshopper (http://kangaroo3d.com/). The Grasshopper definition simulates directional soil particles on different building shapes (Figure 9). Lastly, the computed soil depth values are exported together for each voxel cell.

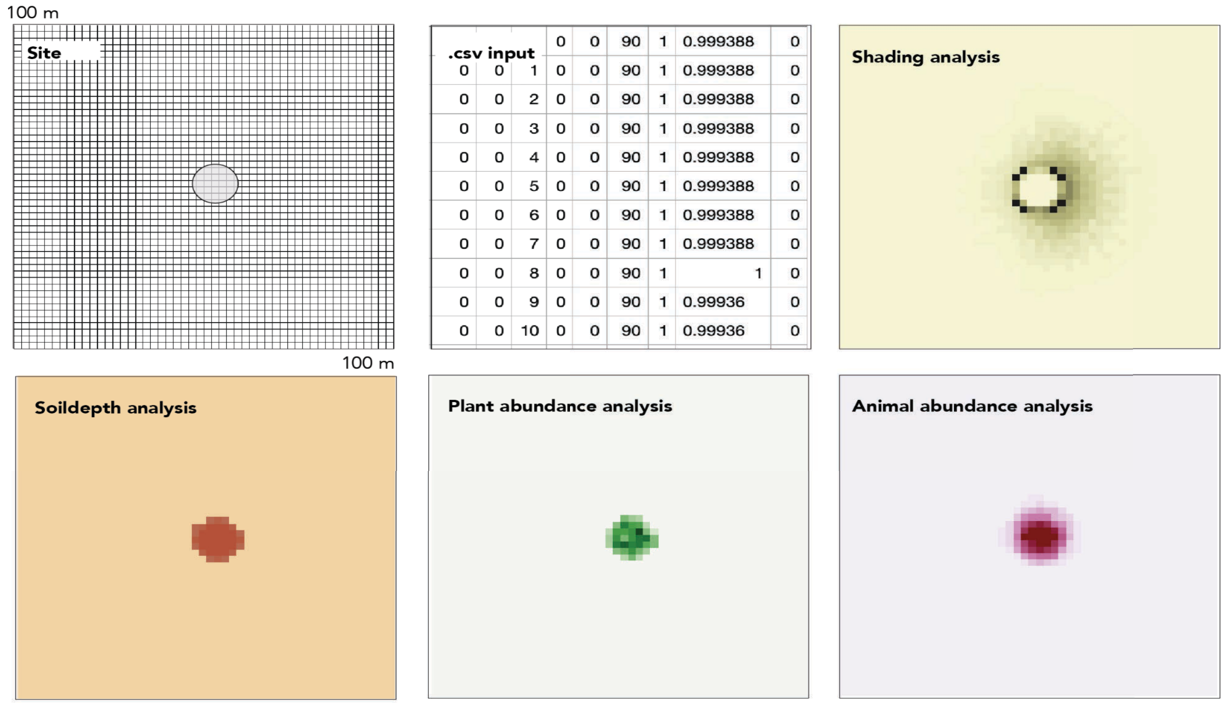


Typology	Surface area	Mass	Inclination	Surface roughness	x	y	z	Soil depth	Shading values

Figure 9: Computation of soil accumulation potential for changing forms stored in each voxel cell.

Besides changing environmental parameters such as shading values and soil depth, architectural parameters such as the building mass, surface inclination and surface roughness values Ra are exported for each voxel cell in the same .csv file (see Grasshopper script in Appendix B).

The ecological model reads the .csv file as an input to compute functional groups abundance and their spatio-temporal distribution (Figure 10). Finally, the results of the ecological analysis are associated with the input data (environmental and architectural data) to be stored in the KB (section 5).



.csv input			0	0	90	1	0.999388	0
0	0	1	0	0	90	1	0.999388	0
0	0	2	0	0	90	1	0.999388	0
0	0	3	0	0	90	1	0.999388	0
0	0	4	0	0	90	1	0.999388	0
0	0	5	0	0	90	1	0.999388	0
0	0	6	0	0	90	1	0.999388	0
0	0	7	0	0	90	1	0.999388	0
0	0	8	0	0	90	1	1	0
0	0	9	0	0	90	1	0.99936	0
0	0	10	0	0	90	1	0.99936	0

Figure 10: Shading(C) and soil depth(D) were calculated using Ladybug tools and Kangaroo Physics, while the plant(E) and animal(F) abundances were simulated by the ecological model. The ecological model produces large quantities of spatially and temporally explicit data; here only one PFG and one AFG are presented at one iteration step (t = 15)

4.4 Data exchange format methods

To achieve interoperability between raster and 3D geometry data, the MiMo method from D3.1 was applied. After some considerations of using Geotiff as an exchange format (interpreted by the Geospatial Data Abstraction Library, GDAL), for draft work, the .txt/.csv file format was used, as parsing the data required no additional dependencies. In the current version, .csv is used as an in- and output file format for exchanging information between Grasshopper and the ecological model.

4.5 Results and first questions to be answered by the KGF

Our first results show raster image plots of the spatio-temporal distribution and the abundance of functional groups per square metre in Grasshopper. These plots are associated with the corresponding metadata (architectural, ecological, environmental data), a .csv file, for each time stamp (Figure 11). Having the first results, the objective for the next step is to find patterns for different building typologies/ masses with each interaction of the process, and to store this information in the KB (section 5).

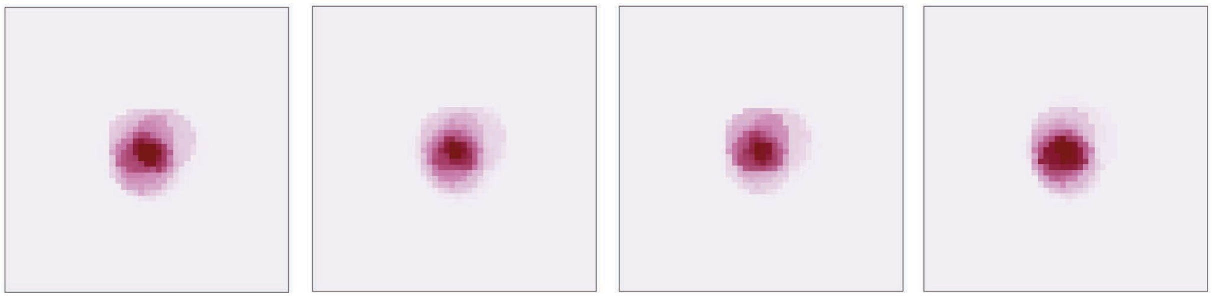


Figure 11: KGF plots of animal abundance over time (=Spatio-temporal raster data output).

Once the KGF has generated enough information, initial questions can be answered using statistical analysis. For instance, the eco-architectural correlations revealed could answer more precisely how the inclination of the façade and abundance of plants are related or how building mass influences the abundance of species on the plot (Figure 12).

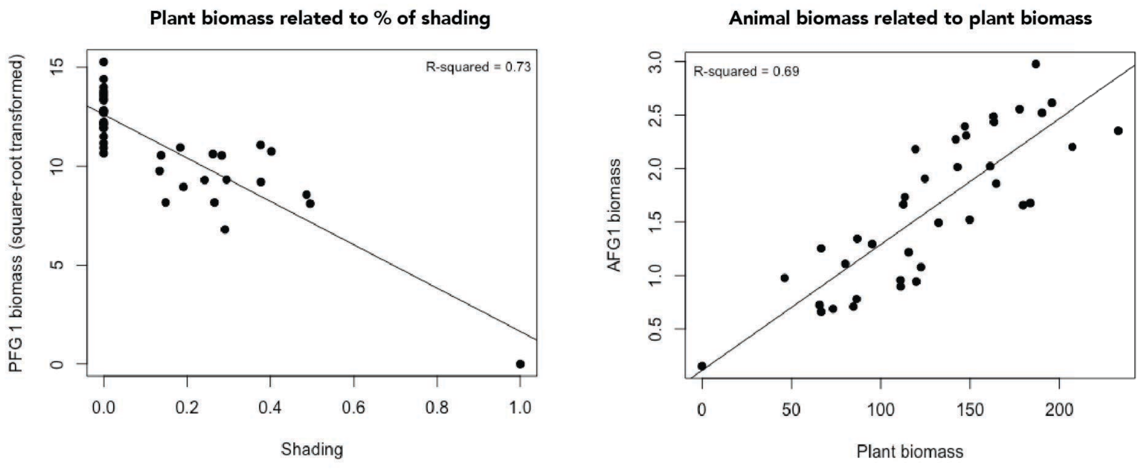


Figure 12: Correlations between plant biomass and shading values (left), and between animal biomass and plant biomass (right).

4.6 Conclusions

Key questions such as how architectural form promotes biodiversity, biomass, and ecological performance in our cities are investigated in the ECOLOPES project. Answering these questions is directly addressed in the KGF. The KGF integrates knowledge from the environment, ecology and architecture to generate new knowledge about the correlations between different parameters as the main contribution of the KGF. Besides an experimental workflow for knowledge generation using interchanging information between raster and 3D geometry data, an Ecological Model, developed by experts from ecology, was integrated as a key component within a 3D CAD system. By the Grasshopper integration, ecological modelling has become an intrinsic part of a parametric and data/information-driven design process that considers the requirements of multiple species.

Furthermore, the KGF has the potential to apply AI concepts (such as ML) to solve problems, which may be useful for assisting with human learning and making decisions. Thus, the KGF could have built-in problem-solving capabilities that allows an understanding of the context of the data reviewed and processed to make informed decisions based on the knowledge stored in the KB.

5. INTERIM KNOWLEDGE BASE (KB) AS RESULT OF THE KGF

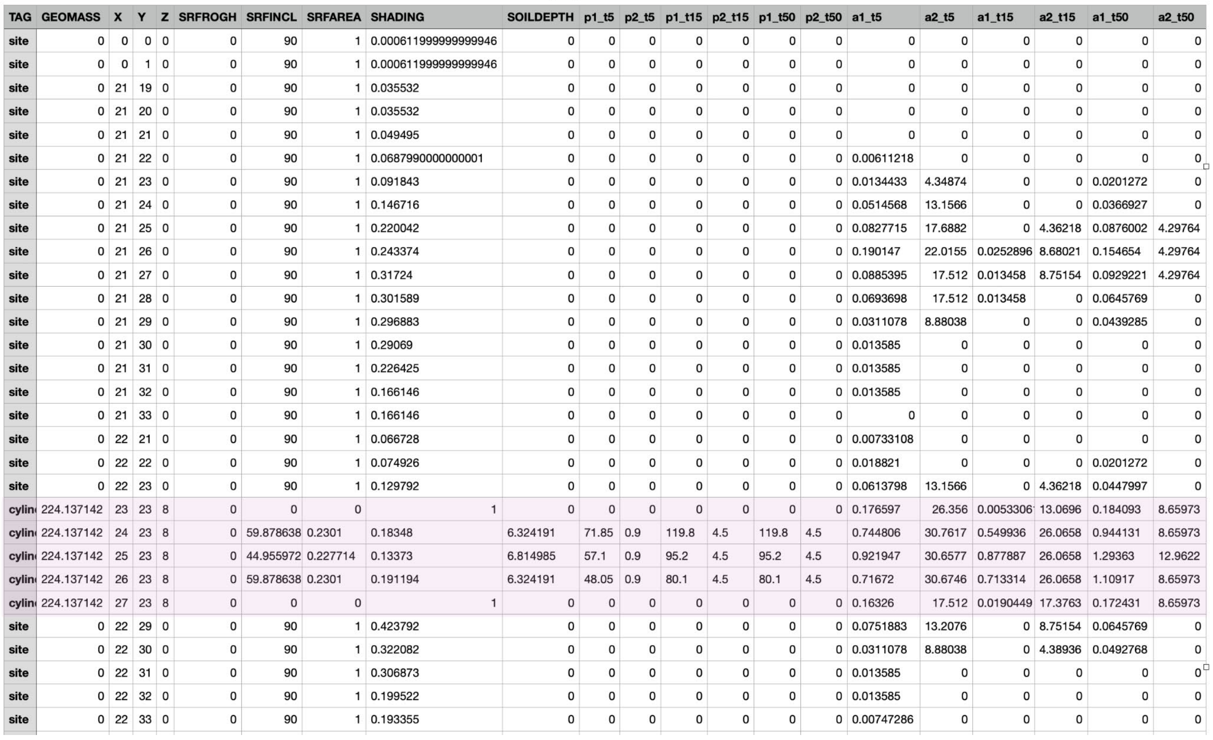
The interim Knowledge base (KB) is the direct outcome from the KGF experiments. As a repository for the newly generated knowledge it captures the knowledge of human experts/ ecologists to support design decision-making. In the future the interim KB will be an .xml file that stores in each row values for each raster cell/ time which is associated with ecology, environmental and architectural related data types in each column (for compatibility reasons we currently resort to .csv files). The data stored in the KB has the potential to be used for machine learning (ML) that leverages the data to improve performance by using it as sample or training data to make predictions and decisions. Furthermore, the KB can be used for machine reasoning (ontology) which empowers machines to make connections between different facts and observations (networks) and to apply human-like common sense to analyse and translate vast knowledge into clear explainable insights. Thus, the KB is the key element to apply a variety of AI concepts with built-in problem-solving capabilities. The knowledge base content is developed by TUM, MCNEEL, and the TECHNION.

The goal for the initial KB is to be an integrated part of the ECOLOPES platform, where the newly generated knowledge can be used for (1) and design decision support, and (2) KPI calculations.

5.1 Data for design generation

The interim KB contains newly generated datasets that can be used for design generation. Shading values are directly linked to plant and animal biomass, as well as the building mass, typology and surface roughness of the envelope are related to soil depth and total amount of soil on the test plot (Table 1).

Table 1: The interim KB and the data stored for the first computed typology (cylinder, see highlighted field). Columns 1-6 indicate architectural data, columns 7-9 environmental, and columns 10-21 ecological data.



TAG	GEOMASS	X	Y	Z	SRFROGH	SRFINCL	SRFAREA	SHADING	SOILDEPTH	p1_t5	p2_t5	p1_t15	p2_t15	p1_t50	p2_t50	a1_t5	a2_t5	a1_t15	a2_t15	a1_t50	a2_t50
site	0	0	0	0	0	90	1	0.000611999999999946	0	0	0	0	0	0	0	0	0	0	0	0	0
site	0	0	1	0	0	90	1	0.000611999999999946	0	0	0	0	0	0	0	0	0	0	0	0	0
site	0	21	19	0	0	90	1	0.035532	0	0	0	0	0	0	0	0	0	0	0	0	0
site	0	21	20	0	0	90	1	0.035532	0	0	0	0	0	0	0	0	0	0	0	0	0
site	0	21	21	0	0	90	1	0.049495	0	0	0	0	0	0	0	0	0	0	0	0	0
site	0	21	22	0	0	90	1	0.0687990000000001	0	0	0	0	0	0	0	0.00611218	0	0	0	0	0
site	0	21	23	0	0	90	1	0.091843	0	0	0	0	0	0	0	0.0134433	4.34874	0	0	0.0201272	0
site	0	21	24	0	0	90	1	0.146716	0	0	0	0	0	0	0	0.0514568	13.1566	0	0	0.0366927	0
site	0	21	25	0	0	90	1	0.220042	0	0	0	0	0	0	0	0.0827715	17.6882	0	4.36218	0.0876002	4.29764
site	0	21	26	0	0	90	1	0.243374	0	0	0	0	0	0	0	0.190147	22.0155	0.0252896	8.68021	0.154654	4.29764
site	0	21	27	0	0	90	1	0.31724	0	0	0	0	0	0	0	0.0885395	17.512	0.013458	8.75154	0.0929221	4.29764
site	0	21	28	0	0	90	1	0.301589	0	0	0	0	0	0	0	0.0693698	17.512	0.013458	0	0.0645769	0
site	0	21	29	0	0	90	1	0.296883	0	0	0	0	0	0	0	0.0311078	8.88038	0	0	0.0439285	0
site	0	21	30	0	0	90	1	0.29069	0	0	0	0	0	0	0	0.013585	0	0	0	0	0
site	0	21	31	0	0	90	1	0.226425	0	0	0	0	0	0	0	0.013585	0	0	0	0	0
site	0	21	32	0	0	90	1	0.166146	0	0	0	0	0	0	0	0.013585	0	0	0	0	0
site	0	21	33	0	0	90	1	0.166146	0	0	0	0	0	0	0	0	0	0	0	0	0
site	0	22	21	0	0	90	1	0.066728	0	0	0	0	0	0	0	0.00733108	0	0	0	0	0
site	0	22	22	0	0	90	1	0.074926	0	0	0	0	0	0	0	0.018821	0	0	0	0.0201272	0
site	0	22	23	0	0	90	1	0.129792	0	0	0	0	0	0	0	0.0613798	13.1566	0	4.36218	0.0447997	0
cylin	224.137142	23	23	8	0	0	0	1	0	0	0	0	0	0	0	0.176597	26.356	0.0053306	13.0696	0.184093	8.65973
cylin	224.137142	24	23	8	0	59.878638	0.2301	0.18348	6.324191	71.85	0.9	119.8	4.5	119.8	4.5	0.744806	30.7617	0.549936	26.0658	0.944131	8.65973
cylin	224.137142	25	23	8	0	44.955972	0.227714	0.13373	6.814985	57.1	0.9	95.2	4.5	95.2	4.5	0.921947	30.6577	0.877887	26.0658	1.29363	12.9622
cylin	224.137142	26	23	8	0	59.878638	0.2301	0.191194	6.324191	48.05	0.9	80.1	4.5	80.1	4.5	0.71672	30.6746	0.713314	26.0658	1.10917	8.65973
cylin	224.137142	27	23	8	0	0	0	1	0	0	0	0	0	0	0	0.16326	17.512	0.0190449	17.3763	0.172431	8.65973
site	0	22	29	0	0	90	1	0.423792	0	0	0	0	0	0	0	0.0751883	13.2076	0	8.75154	0.0645769	0
site	0	22	30	0	0	90	1	0.322082	0	0	0	0	0	0	0	0.0311078	8.88038	0	4.38936	0.0492768	0
site	0	22	31	0	0	90	1	0.306873	0	0	0	0	0	0	0	0.013585	0	0	0	0	0
site	0	22	32	0	0	90	1	0.199522	0	0	0	0	0	0	0	0.013585	0	0	0	0	0
site	0	22	33	0	0	90	1	0.193355	0	0	0	0	0	0	0	0.00747286	0	0	0	0	0

With every change in geometry typologies, new ecological values are calculated. These values are then stored in the KB.

The goal of the interim KB is to a) compute KPI ranges and values (see 5.2), and b) to provide a database to extract the newly generated knowledge for design decision support (machine learning or machine reasoning methods).

5.2 KPI generation for application in design optimisation

The results of the KGF would be output in the .xml format which can be read into the Grasshopper environment through plug-ins such as Lunchbox (https://apps.provingground.io/lunchbox/). These results can be formatted into numerical and textual information representation which would facilitate in formulating KPIs. Mainly, the output of the environmental (e.g., Shading and Soil Depth) and ecological model (e.g., Plant and Animal Functional Groups) will be used as KPIs while the correlated architectural parameters (e.g., surface roughness, surface inclination, and surface area) would determine geometrical thresholds. Within the context of the optimisation set-up, the architectural thresholds define the geometrical limits of a parameterised design. These thresholds then correspond to a range of target environmental and ecological performance values to achieve. At present, the environmental parameters will be utilised as fitness objectives in the optimisation phase by which the optimised values are correlated with the ecological outputs in the KGF.

5.3 Conclusions

In summary, the interim KB is currently a .csv file that stores in the KGF newly generated knowledge about the relationships between architecture, ecology and the environment. It is a non-deterministic database as the constantly generated data is stored each time a new geometry input is analysed. Thus, the KB is the result of the KGF but also the key element for further knowledge extraction during future design generation and optimisation processes. Besides statistical analysis approaches (Figure 12), the generated data can be extracted automatically rather than manually (machine learning and reasoning) for design decision support.

6. The draft ECOLOPES platform architecture and technical components

Section 6 will introduce the draft version of the ECOLOPES design platform and its integrated technical components. The Sandbox, the 1st prototype of the ECOLOPES platform, has a cloud-based infrastructure to enable real-time cloud computing, visualisation of analysis results in open web interfaces, and the implementation of decision-support tools for urban design and urban analysis. The ECOLOPES platform is developed by MCNEEL. D3.1 presented the first version of the ECOLOPES platform architecture. After an update about the State-of-the-Art for data-driven urban planning platforms (section 6.1), this report will focus on the improvements made in respect of the integration of developed software components including the ECOLOPES frontend tools (section 6.2).

6.1 State-of-the-art update

One of the main challenges with respect to the ECOLOPES technology is to combine cross-disciplinary expert datasets with 3D CAD models, and to enable a user interface to interact with the system. In urban design research and software development practice, similar challenges were encountered and addressed. The latest data/AI-driven urban planning platforms for generative design, analysis and optimisation is called SPACIO (https://www.spacio.ai/). SPACIO is developed by a team of computational and environmental designers based in Sweden and is currently available as a beta version. SPACIO is designed as a cloud-based system accessible through a standard web browser. As such, the system is hardware and software independent as well as easy to use for non-experts. Real-time analysis and simulations (environmental, energy, structural, spatial, sound, wind, storm water, metrics, human comfort, landscape and LCA) are available for design decision support and AI-based generative design and optimisation of urban spaces (Figure 13 and Figure 14).

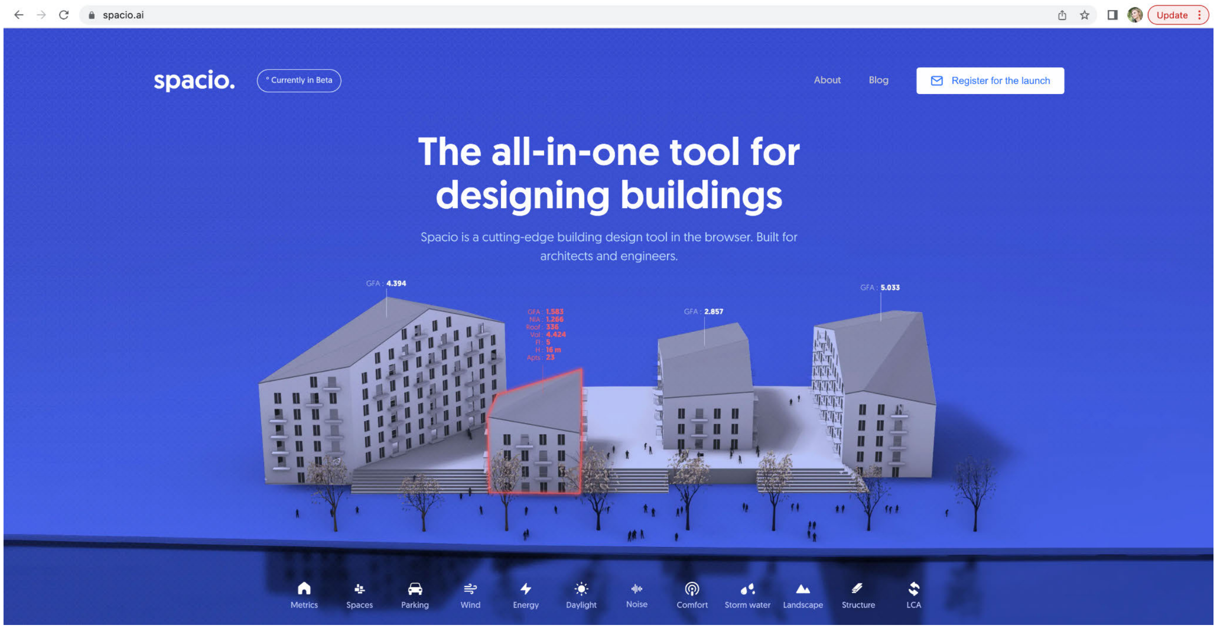


Figure 13: Browser-based web interface of the SPACIO design platform, a (in beta) technology for urban AI-driven design and optimisation (screenshot).

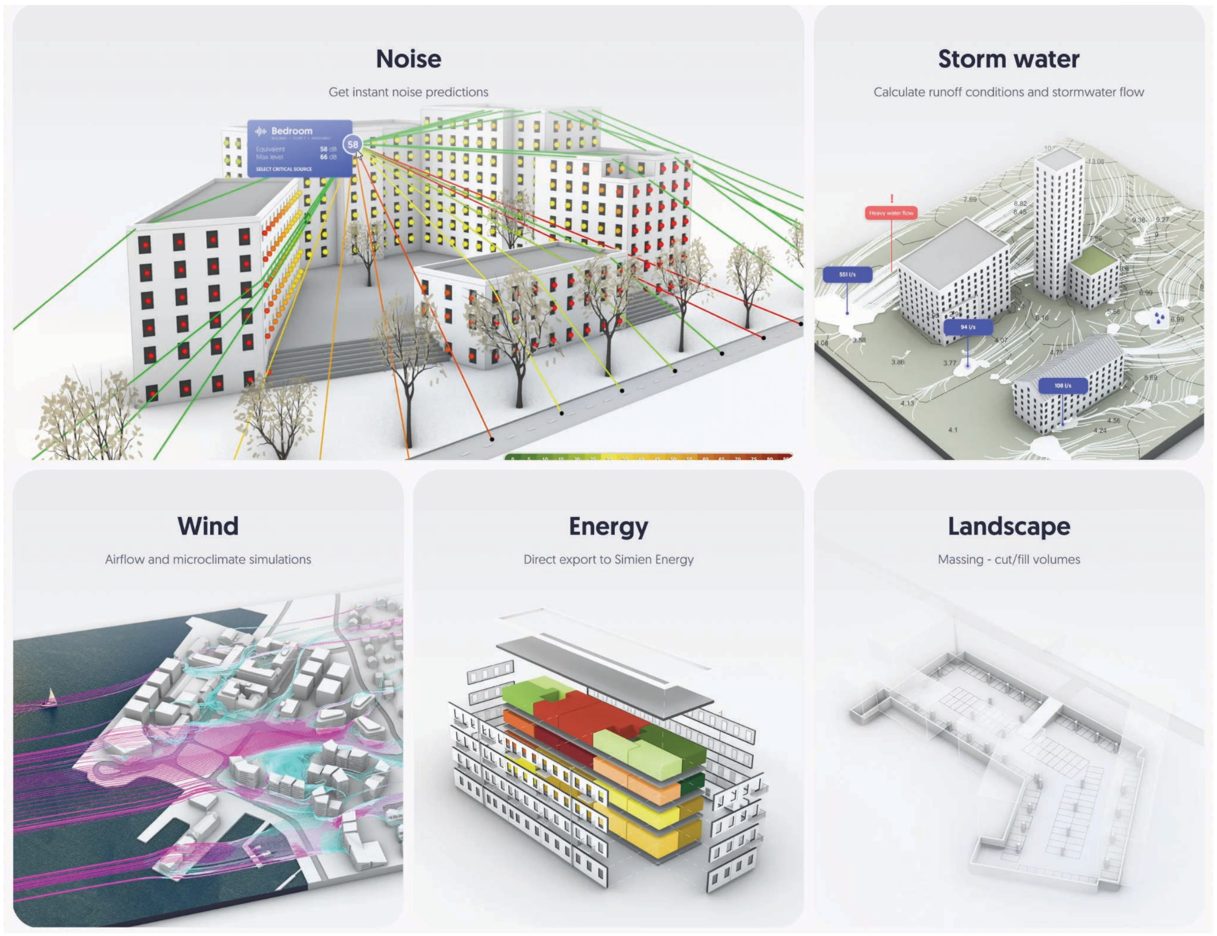

Noise
Get instant noise predictions
Bedroom
Storm water
Calculate runoff conditions and stormwater flow
Wind
Airflow and microclimate simulations
Energy
Direct export to Simien Energy
Landscape
Massing - cut/fill volumes

Figure 14: Real-time feedback from analysis and simulations with an impact for building performance (screenshot).

A uniform data representation (described in DataTrees) allows to connect all elements and to be aware of neighbouring elements, meaning they react and respond to changes. This structure allows any building to be represented and documented in a standardised manner and thus becoming comparable, measurable and reusable. SPACIO can import and export CAD and BIM data formats (.obj, .dwg, .3dm, .ifc) as well as metadata in the .json file format.

SPACIO also generates new knowledge based on ML processes. For instance, thousands of floor plans are in their database that were trained to assist the design process. This gathered data can be compared to the KB that we are developing in ECOLOPES.

In summary, the SPACIO design platform seems to offer an all-in-one solution to support a more efficient design process and to increase the performance of building and urban environments based on the criteria defined (KPIs). However, the inclusion of building optimisation based on ecological analysis for an integrated multi-species design approach as envisioned in the ECOLOPES project is still not part of their system. This fact underlines the importance to focus on the integration of ecological modelling and expert-knowledge from ecology in CAD to enable more informed building envelopes that correspond to the requirements of multiple species in urban environments.

6.2 Overview of the developed system

The developed system architecture results from the layouted computational workflow. Figure 15 shows the improvements made from the first version in D3.1. The main advances include (1) the implementation of the black framed components, (2) the connection of components and processes to ensure that data flow from the beginning until the end is operative (black arrows), and (3) the definition of the types of front-end tools to be developed (section 6.3).

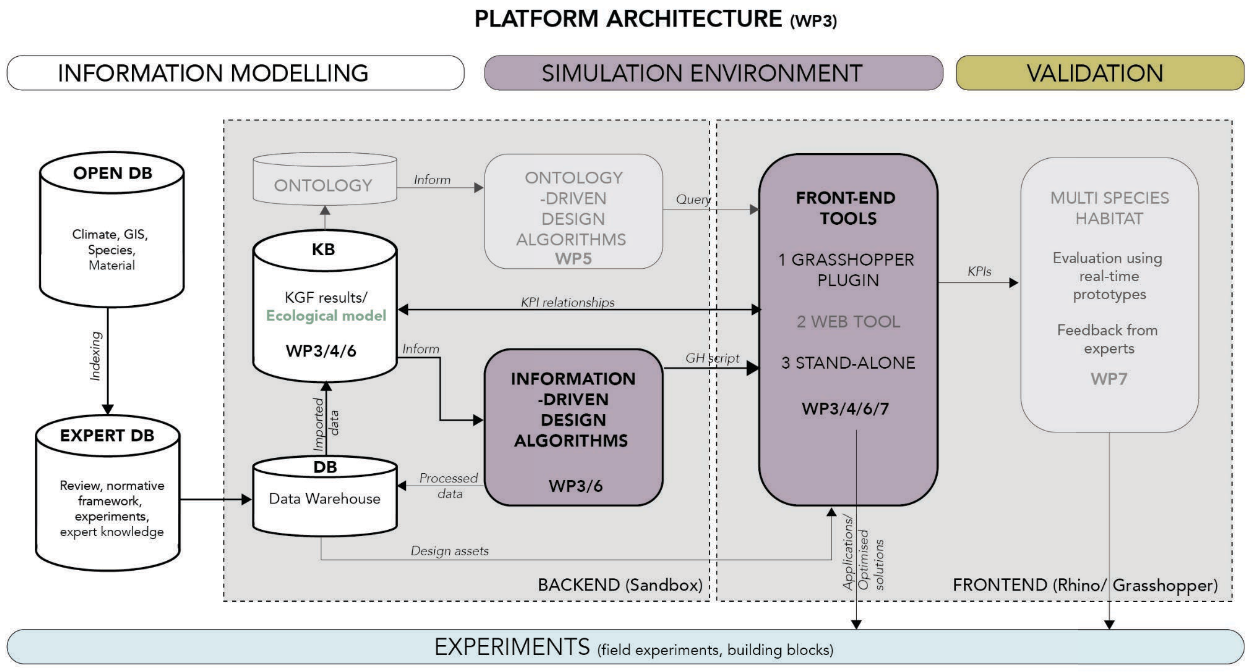

PLATFORM ARCHITECTURE (WP3)
INFORMATION MODELLING
SIMULATION ENVIRONMENT
VALIDATION
OPEN DB
Climate, GIS, Species, Material
Indexing
EXPERT DB
Review, normative framework, experiments, expert knowledge
ONTOLOGY
Inform
ONTOLOGY -DRIVEN DESIGN ALGORITHMS WP5
Query
KB
KGF results/ Ecological model
WP3/4/6
KPI relationships
Inform
Imported data
DB
Data Warehouse
Processed data
INFORMATION -DRIVEN DESIGN ALGORITHMS
WP3/6
GH script
FRONT-END TOOLS
1 GRASSHOPPER PLUGIN
2 WEB TOOL
3 STAND-ALONE
WP3/4/6/7
KPIs
MULTI SPECIES HABITAT
Evaluation using real-time prototypes
Feedback from experts
WP7
Design assets
Applications/ Optimised solutions
BACKEND (Sandbox)
FRONTEND (Rhino/ Grasshopper)
EXPERIMENTS (field experiments, building blocks)

Figure 15: The draft version of ECOLOPES functional architecture and its components.

Data from open databases (climate data, GIS, open species pool, material libraries) is indexed and filtered to be stored in the expert database where also the results from the reviews (section 2.1.2), the normative framework and the experiments are stored. This information is still external to the design platform. Once imported to the cloud-based data warehouse (Section 6.3.3), the information can be used to generate knowledge (see Section 5, Knowledge generation framework) to be stored in the interim KB (Section 6). The KB informs a design optimisation process using KPIs and algorithms from generative design. The developed algorithms are Grasshopper (GH) components (either cluster Grasshopper definitions or C# components). The newly generated knowledge, developed processes, and algorithms are then used for the frontend tool development in the Rhino/ Grasshopper environment. The resulting design outcomes for *ecolopes* are provided to be evaluated later. The cloud-based backend infrastructure developed and implemented by McNeel (see The Sandbox, D3.1 Section 7) remains for testing purposes. Until this point, the full potential of the Sandbox has not been tested as the developed components need to be compiled first to run on the cloud-infrastructure. Currently, they are only operational within a local instance of Rhino/ Grasshopper.

6.3 Technical aspects of the developed and integrated software components

Section 6.3 technically describes the development stage of all implemented components, where they are stored, what Operating System (OS) they run in, where the source code is, what language is it written in, any details related to the compilation and packaging of the component, and who is collaborating.

6.3.1 The ECOLOPES frontend tools

Currently, there are three types of ECOLOPES frontend tools (Task 3.4, Frontend development): 1) A Grasshopper plugin, 2) a web tool (uses GH algorithms as backend service to displays results in a web browser, similar to SPACIO), and 3) stand-alone application for ecological analysis. At this stage, the Grasshopper plugin is under development (lead MCNEEL; TUM, TECHNION, UNIGE, SAAD). Grasshopper components are written in the C# or Iron.Python language. Also new GH definitions can be compiled as a cluster to become a new GH component.

There are four groups of Grasshopper components for the plugin: Components for data exchange, components for form generation, components for ecological and environmental (solar radiation, soil depth, water retention and connectivity) analysis, and components that can voxelise geometry models and export it as raster data. Furthermore, there are components for KPI simulation (filtering, ranking, correlations) and optimisation, and lastly, preview components that visualise data for each inhabitant and display the final ecolope design (Figure 16).

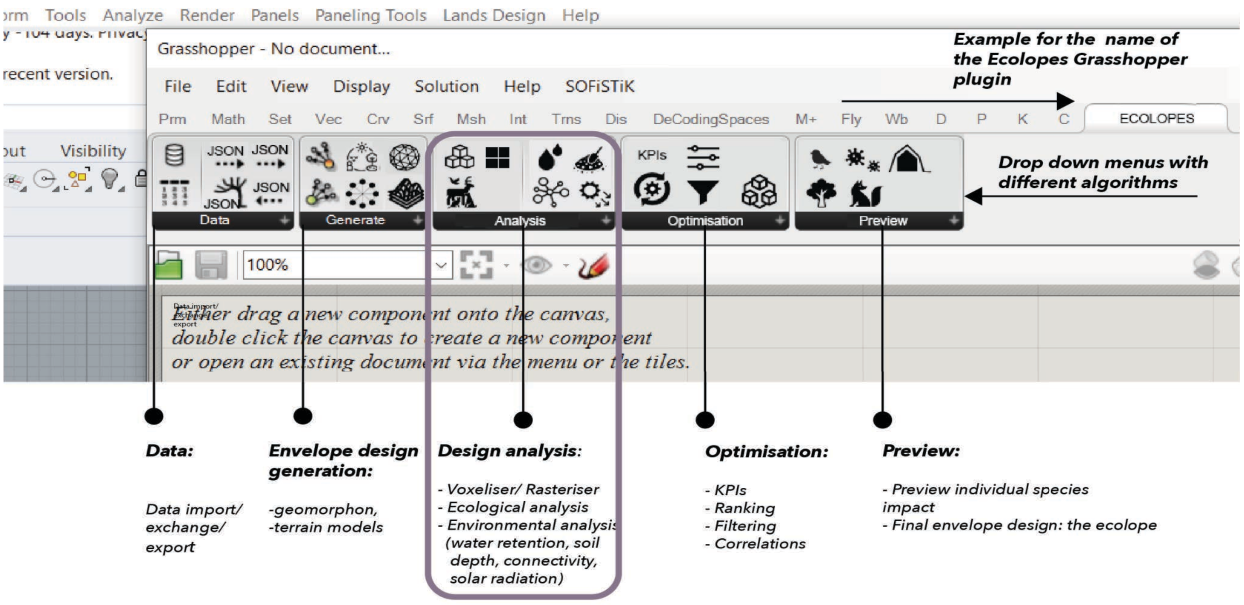

Grasshopper - No document...
File Edit View Display Solution Help SOFiSTiK
Prm Math Set Vec Crv Srf Msh Int Trns Dis DeCodingSpaces M+ Fly Wb D P K C ECOLOPES
Data
Generate
Analysis
Optimisation
Preview
Example for the name of the Ecolopes Grasshopper plugin
Drop down menus with different algorithms
Either drag a new component onto the canvas, double click the canvas to create a new component or open an existing document via the menu or the tiles.
Data:
Data import/ exchange/ export
Envelope design generation:
-geomorphon,
-terrain models
Design analysis:
- Voxeliser/ Rasteriser
- Ecological analysis
- Environmental analysis (water retention, soil depth, connectivity, solar radiation)
Optimisation:
- KPIs
- Ranking
- Filtering
- Correlations
Preview:
- Preview individual species impact
- Final envelope design: the ecolope

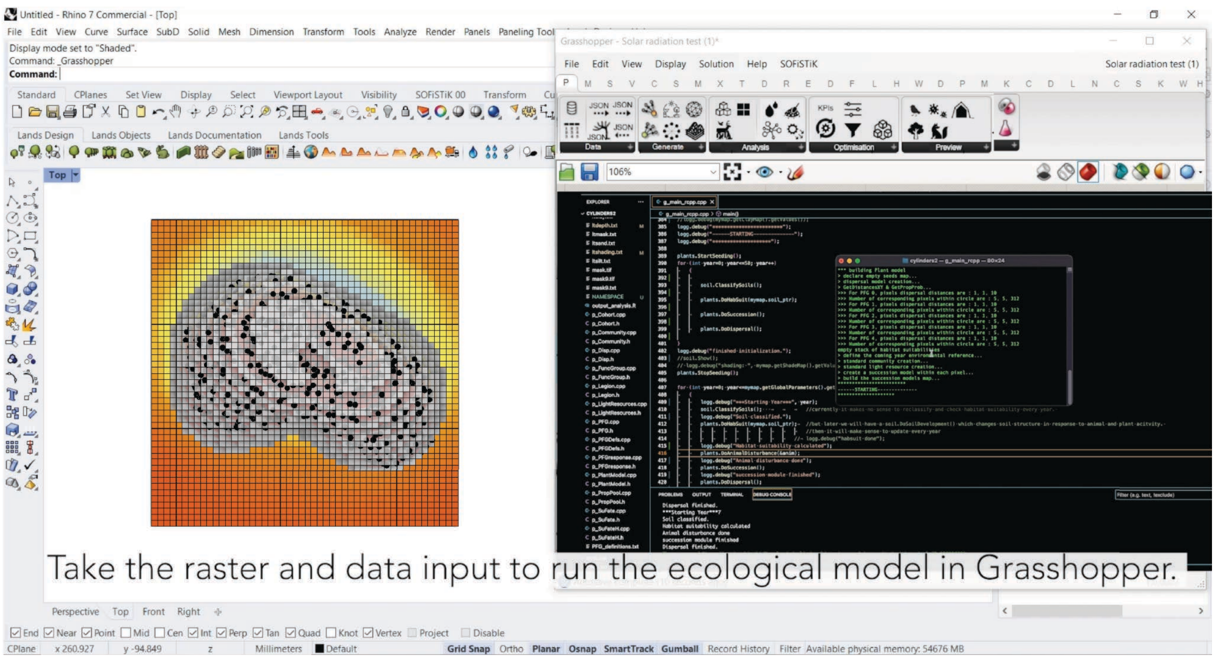
Figure 16: Envisioned UI for the ECOLOPES Grasshopper plugin and the developed GH components

Within the plugin certain Grasshopper components (black icons), that support knowledge generation (KGF) and ecological analysis within a CAD system were prioritised and developed first. Table 2 reports on the development stages of each of these components.

Table 2: Grasshopper components description of the ECOLOPES Grasshopper plugin.

Grasshopper component	Integrated functions	Partner	Short description
☀	Shading analysis	McNeel	The shading analysis component is a GH definition that uses open climate data (.epw) for incident radiation analysis (Ladybug Tools) for parametric 3D geometry. The computed radiation values are converted into shading values and for each voxel cell 1m x1m x1m written in a .csv file. Geometry information such as building typology, building mass, and envelope surface area are adequately computed for each voxel cell and written into the same .csv file.
⛏	Soil depth analysis	Technion	The soil depth analysis component is a GH definition that employs particle simulation on a given geometry using Kangaroo, a physics simulation plugin. The geometry surface is analysed based on a given Soil Type Angle of Repose to deposit soil particles within a predetermined volume. The simulation results in locations where the particles accumulate on the geometry. Using these locations, the average distance between the particle and the surface of the soil volume are calculated. The average soil depth values are correlated to the relevant 1m x 1m x 1m voxel cells to be written in a .csv file.
⧉	Voxelizer	McNeel/ Technion	The voxeliser is a GH definition that voxelises mesh geometry of the building design and terrain into voxel cells of 1m x 1m x1m. The voxel cells are used as an exchange format between 3D geometry and 2D raster data. 3D geometry is split into smaller units to store analysis values at the same resolution as the raster (1m x 1m) required for ecological analysis.
▦	Rasterizer	Technion	The rasteriser is a GH definition that converts the information stored in the 3D voxel model into 2D raster information, that is then written into a .csv file.
🦌	Ecological analysis	TUM/ McNeel	Technically, in the draft version, a C# script component in Grasshopper calls the C++ module and reads simulation results and visualises them in Grasshopper to validate and spatialise data in a 3D CAD environment. In the future version, a series of compiled components will be developed to prepare and serialise input data to be read by C++ module. Furthermore, components to easily visualise data in the Grasshopper environment and contextualised spatially will be developed.
👁	Preview	McNeel	The preview GH component visualises the Ecological model output (raster plots) directly in CAD as a coloured mesh geometry object.
🧍🌡	Human thermal comfort analysis	Unige	The thermal comfort analysis is a GH definition that uses climate data coming from an open database (Ladybug Tools) and geographic coordinates for simulating thermal comfort conditions for the human stakeholders in the environment with 3d parametric geometries. Data coming from the simulation are in .edx format for each hour of the day simulated.

Figure 17 shows the integration of the C++ ecological model in Grasshopper. Raster data is generated from a 3D geometry through a voxel model as it is required as an input for ecological analysis. The operational parts of the ECOLOPES frontend tool were presented to computational designers, landscape architects and at the Digital Landscape Architecture Conference (DLA) 2022 at the Harvard Graduate School of Design, the Rhino User Meeting in Copenhagen, in environmental computational design workshops organised by MCNEEL, and to Master students at the CITA - Centre for Information Technology and Architecture, Royal Danish Academy where it was well received.


Untitled - Rhino 7 Commercial - [Top]
Grasshopper - Solar radiation test (1)*
Data
Generate
Analysis
Optimisation
Preview
Take the raster and data input to run the ecological model in Grasshopper.

```
Display mode set to "Shaded".
Command: _Grasshopper
Command:
```

```
    logg.debug("**********************");
    logg.debug("------STARTING----------------");
    logg.debug("********************");

    plants.StartSeeding();
    for (int year=0; year<=50; year++)
        {

            soil.ClassifySoils();

            plants.DoHabSuit(mymap.soil_ptr);

            plants.DoSuccession();

            plants.DoDispersal();
        }
    logg.debug("finished initialization.");
    //soil.Show();
    // logg.debug("shading: ", mymap.getShadeMap().getValu
    plants.StopSeeding();

    for (int year=0; year<=mymap.getGlobalParameters().get
        {
            logg.debug("***Starting Year***", year);
            soil.ClassifySoils();       //currently it makes no sense to reclassify and check habitat suitability every year.
            logg.debug("Soil classified.");
            plants.DoHabSuit(mymap.soil_ptr);   //but later we will have a soil.DoSoilDevelopment() which changes soil structure in response to animal and plant acitvity.
                                                //then it will make sense to update every year
                                            //  logg.debug("habsuit done");
            logg.debug("Habitat suitability calculated");
            plants.DoAnimalDisturbance(&anim);
            logg.debug("Animal disturbance done");
            plants.DoSuccession();
            logg.debug("succession module finished");
            plants.DoDispersal();
```

```
*** building Plant model
> declare empty seeds map...
> dispersal model creation...
> GetDistancesXY & GetPropProb...
>>> For PFG 0, pixels dispersal distances are : 1, 1, 10
>>> Number of corresponding pixels within circle are : 5, 5, 312
>>> For PFG 1, pixels dispersal distances are : 1, 1, 10
>>> Number of corresponding pixels within circle are : 5, 5, 312
>>> For PFG 2, pixels dispersal distances are : 1, 1, 10
>>> Number of corresponding pixels within circle are : 5, 5, 312
>>> For PFG 3, pixels dispersal distances are : 1, 1, 10
>>> Number of corresponding pixels within circle are : 5, 5, 312
>>> For PFG 4, pixels dispersal distances are : 1, 1, 10
>>> Number of corresponding pixels within circle are : 5, 5, 312
empty stack of habitat suitabilities
> define the coming year environmental reference...
> standard community creation...
> standard light resource creation...
> create a succession model within each pixel...
> build the succession models map...
********************
------STARTING--------------
********************
```

```
Dispersal finished.
***Starting Year***7
Soil classified.
Habitat suitability calculated
Animal disturbance done
succession module finished
Dispersal finished.
```

Figure 17: Screenshot of the Rhino/ Grasshopper modelling environment. It shows the integration of the ecological C++ model in Grasshopper.

6.3.2 The ecological model

The ecological model is coded in C++ (minimum version requirement: C++ 11), requiring several header-only libraries from the boost package (https://www.boost.org/). It compiles platform-independent, having been tested on both windows and macOS. The code is shared with the ECOLOPES consortium via gitlab, and managed by WP4. The memory and CPU requirements depend heavily on the number of functional groups considered, the spatial extent of the ecolope and the number of iterations (years). In a simple example with only 2 plant and 5 animal functional groups the program uses 100% of one CPU (multi-threading is not supported yet) and less than 500 mb RAM, running for approx. 3 hours on a 2.3 GHz MacBook Pro with Intel Core i9 processors. The computational efficiency will be improved in the near future.

Required inputs and input format: The ecological model requires a 2-dimensional representation of environmental variables ("heatmaps"). It has been written under the assumption that the spatial resolution is 1m x 1m and the spatial extent of the ecolope is between 100x100 and approx. 300x300m, though the extent can be flexibly varied. The following variables need to be provided:

1. One file with the USDA soil texture class of the soil covering each grid cell, coded as numerical (integer) values; or three files with the percentages of silt, clay and sand content, respectively, in each grid cell, rounded to the nearest integer;
2. The degree of shading in each grid cell, coded as floating-point number (double precision) between 0 and 1;
3. The depth of the soil in each cell, coded as (positive) integer number indicating the depth in cm.

The soil texture is to be provided as a plain text file, with one value per row, and the number of rows needs to match the spatial extent of the ecolope (10,000 rows for a 100x100m ecolope). Shading and soil depth can be provided in a .csv file (Figure 18), or they can also be provided as individual text files.

Furthermore, the model requires a text file ("filenames.txt") indicating the directory and file names of all inputs. Further setup parameters (e.g., building lifespan, number of CPU cores to use) are currently hardcoded and cannot be changed by the user. Similarly, the definitions of animal and plant functional groups (to be provided in a separate analysis in WP4.2/4.3) are fixed in the model.

Outputs and output format: The main output is the plant biomass and animal home ranges of each plant/animal functional group, in each grid cell and for each simulation iteration (presenting one year of the building's life span) (Figure 18/6). The plant biomass is saved as a text file with one value per row in the same order as the input data, and one text file is produced for each PFG and time step. The animal output (also a text file) summarises the location of each home range at the end of the model run.

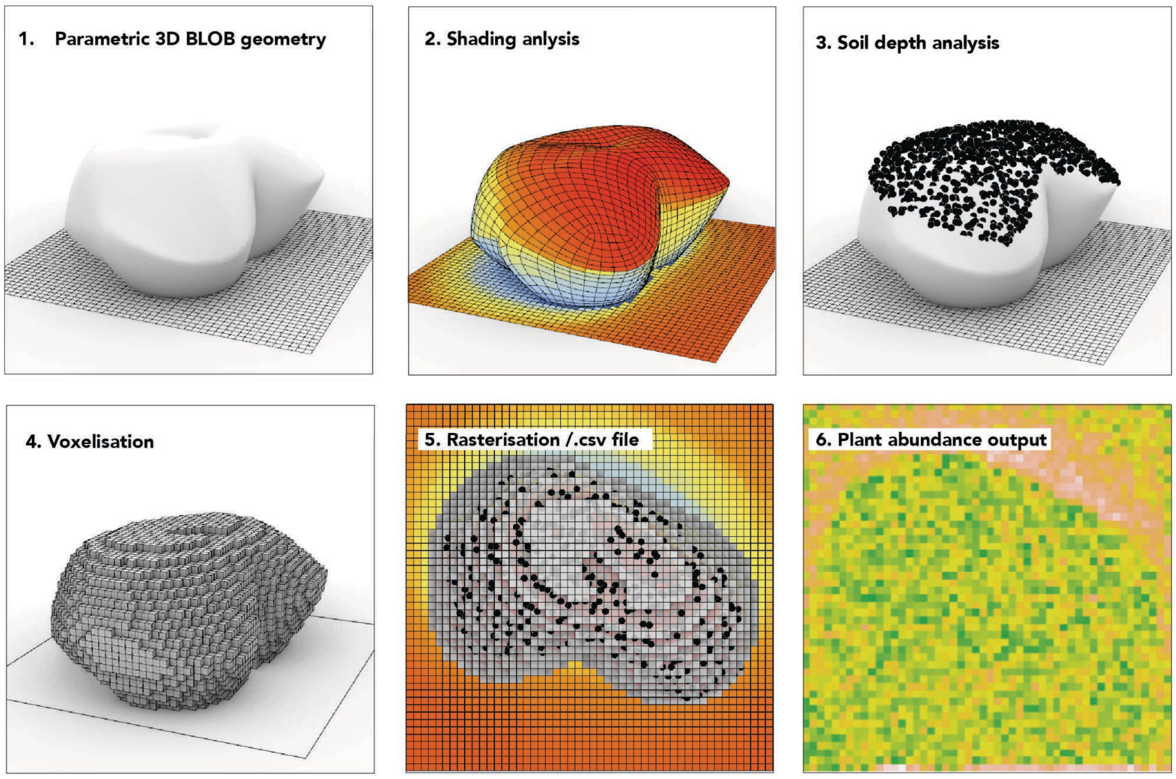
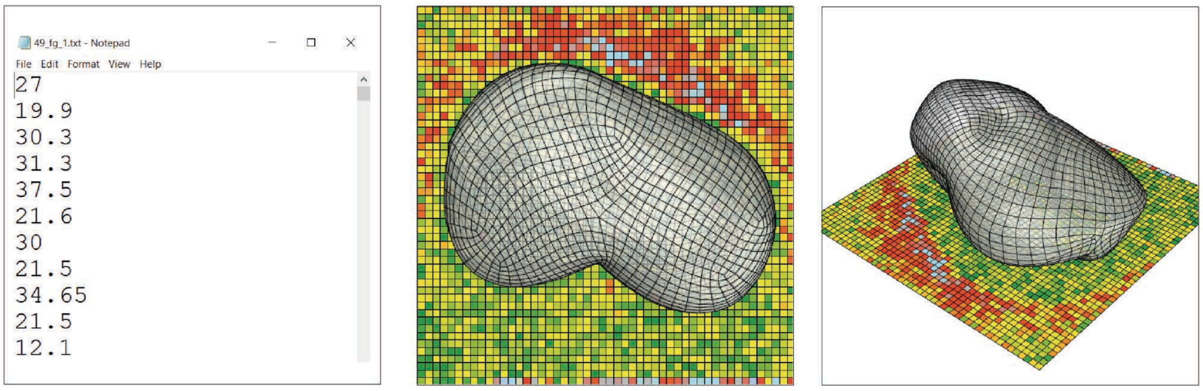

1. Parametric 3D BLOB geometry
2. Shading anlysis
3. Soil depth analysis
4. Voxelisation
5. Rasterisation /.csv file
6. Plant abundance output

Figure 18: GH components and the file exchange described were tested using a random building geometry (Blob architecture example). The ecological model outputs spatio-temporal values (raster plots/ .txt file) for plant abundance on the building envelope and the site.

Connection with the Grasshopper environment: The ecological model described in the previous section was integrated with the Grasshopper visual scripting environment through a process invocation which allows .NET code (written in c#) to execute compiled applications. The initial implementation includes a custom c# scripting component as part of a larger Grasshopper "definition" which prepares the input for the environmental model and eventually visualises the results from the ecological model in the Rhino viewport. The input and output information is formatted as a .csv as described before. When the ecological model has completed its calculations, the results can be explored through Grasshopper and visualised in the Rhino viewport (Figure 19).



```
27
19.9
30.3
31.3
37.5
21.6
30
21.5
34.65
21.5
12.1
```

Figure 19: Preview of raster data outputs from .txt files in Rhino/ Grasshopper.

6.3.3 The ECOLOPES data warehouse

The ECOLOPES data warehouse (D3.1, sections 6.1.3 and 7.2.1) is a Linux cloud server which can be accessed through NextCloud, a free file sharing and collaboration platform. The server enables file storage and the live exchange of analysed data. The data storage can be accessed through NextCloud: https://data.mcneelresearchprojects.com/). An SQL database (MariaDB) is hosted in parallel to the NextCloud data storage. It can be accessed through the Rhino/ Grasshopper Hops interface (D5.1, section 2.4.3).

All Ecolopes-relevant data (for interdisciplinary computational processes) are stored and made available through the data warehouse. It also stores geometry data which includes the voxel model, generative design outcomes, 3D analysis and simulation results, and metadata.

6.3.4 The ECOLOPES computational simulation environment

The ECOLOPES computational simulation environment (D3.1, sections 6.1.4, 7.2.2 and 7.2.3) is hosted on a Windows cloud server. The server can be accessed through the following link: compute.mcneelresearchprojects.com. Rhino.Compute and the Rhino.Compute.AppServer servers enable the computation of the developed GH definitions as a backend service (Task 3.3, Backend development and integration). Rhino.Compute is an open-source project. It uses the Rhino.InsideTM technology (GitHub Rhino.Inside, 2021), another open source project which allows Rhino and Grasshopper to run inside other 64-bit Windows applications. In Rhino.Compute, Rhino SDK functions are accessed via a cloud-based stateless REST API. Through Rhino.Compute developers can create applications that manipulate Rhino (3dm files), solve Grasshopper definitions and python scripts without needing to have Rhino installed on the client devices. Furthermore, it can call over 2400 geometric operations from the RhinoCommon SDK remotely. With respect to the ECOLOPES platform, Rhino.Compute offers the possibility to calculate process-intensive Grasshopper algorithms for geometry generation and analysis within the cloud. Furthermore, it allows cross-disciplinary team members to have access to a 'headless' Rhino version and to work as a team simultaneously on the same project. Another advantage of Rhino.Compute is that Grasshopper definitions can run as a backend service (D3.1, section 2.4.2).

6.4 Human comfort analysis for validation

Within the ECOLOPES front-end tools, the integration of Envi-met Science software has been enabled using *Ladybug, Dragonfly Legacy* and *Morph*o GH plug-ins. This allows designers to run simulations for outdoor thermal comfort (including vegetation cooling effects) and to visualise analysis results directly in Grasshopper.
The first part of the Gh definition defines the model setup and targeted analysis results. Opportunities from this integration are mainly related to the possibility of running simulations in the Grasshopper environment, allowing to compute real time simulations of different

design solutions without modelling each time the building geometry in the Envi-met Science Workspace. Figure 20 shows the computational workflow for our simulation setup, highlighting input and output data.

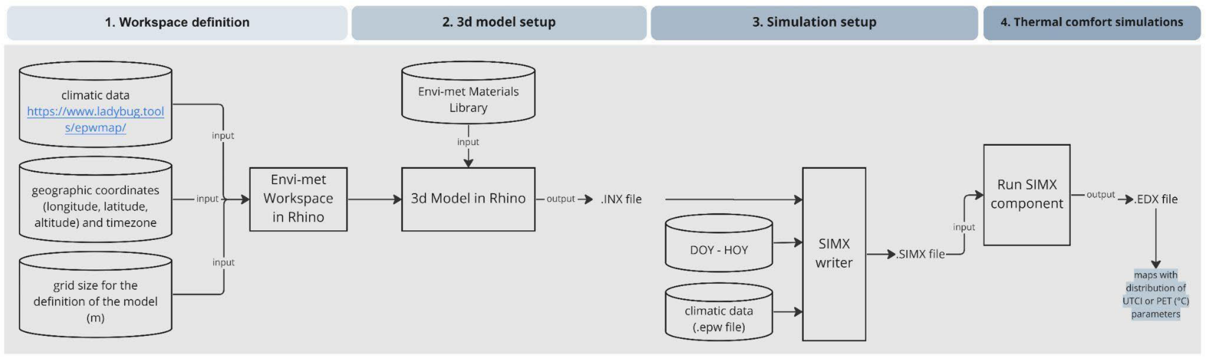
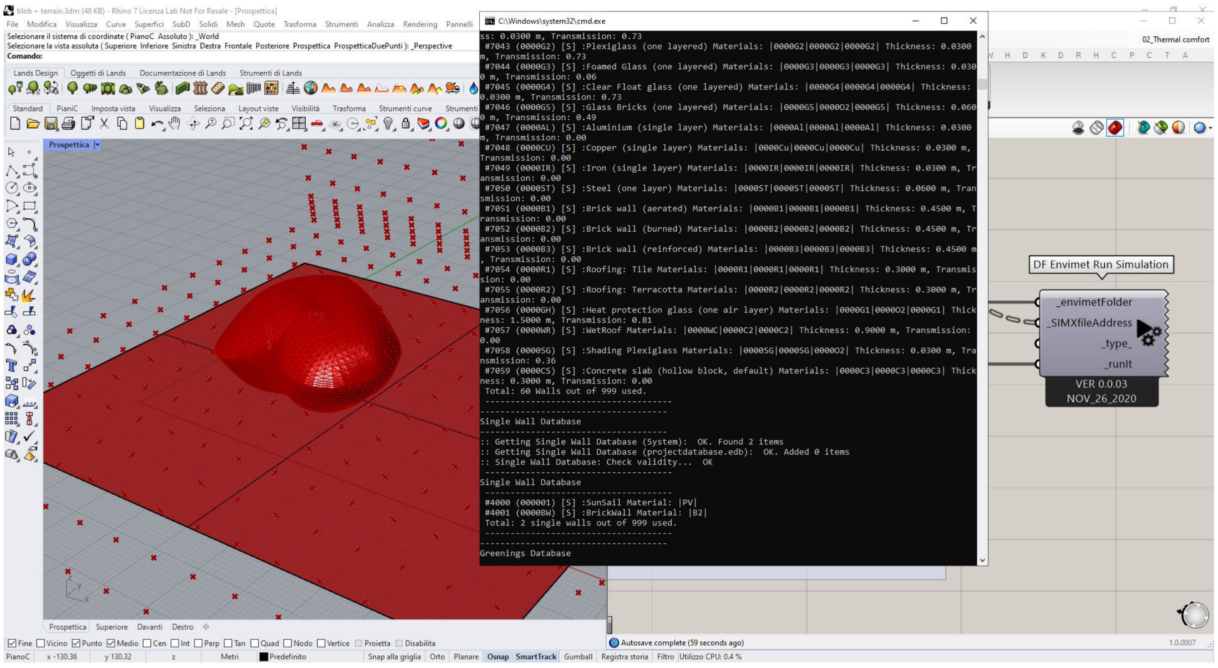

1. Workspace definition
2. 3d model setup
3. Simulation setup
4. Thermal comfort simulations
climatic data https://www.ladybug.tools/epwmap/
geographic coordinates (longitude, latitude, altitude) and timezone
grid size for the definition of the model (m)
input
Envi-met Workspace in Rhino
Envi-met Materials Library
3d Model in Rhino
output
.INX file
DOY - HOY
climatic data (.epw file)
SIMX writer
.SIMX file
Run SIMX component
.EDX file
maps with distribution of UTCI or PET (°C) parameters

Figure 20: Workflow for human thermal comfort simulations in GH.

In particular, input data for model the setup are: (1) climatic data coming from the online database (https://www.ladybug.tools/epwmap/); (2) geographic coordinates of the location (longitude, latitude, altitude) and timezone; and (3) grid size for the definition of the model (in metre). Once the geometry is defined, materials from the Envi-met Materials Library are applied. The output from this first part of the script is an .INX file. The output of the second part of the workflow is a .SIMX file stored in a folder with a series of .EDX files for each hour simulated within a specific day defined during the simulation setup (Fig. 21).

The simulation setup requires the following information: (1) climatic data (.epw file); and (2), day and hour of the day that the software will simulate (LB Calculate HOY component).



```
ss: 0.0300 m, Transmission: 0.73
 #7043 (0000G2) [S] :Plexiglass (one layered) Materials: |0000G2|0000G2|0000G2| Thickness: 0.0300
m, Transmission: 0.73
 #7044 (0000G3) [S] :Foamed Glass (one layered) Materials: |0000G3|0000G3|0000G3| Thickness: 0.030
0 m, Transmission: 0.06
 #7045 (0000G4) [S] :Clear Float glass (one layered) Materials: |0000G4|0000G4|0000G4| Thickness:
0.0300 m, Transmission: 0.73
 #7046 (0000G5) [S] :Glass Bricks (one layered) Materials: |0000G5|0000O2|0000G5| Thickness: 0.060
0 m, Transmission: 0.49
 #7047 (0000AL) [S] :Aluminium (single layer) Materials: |0000Al|0000Al|0000Al| Thickness: 0.0300
m, Transmission: 0.00
 #7048 (0000CU) [S] :Copper (single layer) Materials: |0000Cu|0000Cu|0000Cu| Thickness: 0.0300 m,
Transmission: 0.00
 #7049 (0000IR) [S] :Iron (single layer) Materials: |0000IR|0000IR|0000IR| Thickness: 0.0300 m, Tr
ansmission: 0.00
 #7050 (0000ST) [S] :Steel (one layer) Materials: |0000ST|0000ST|0000ST| Thickness: 0.0600 m, Tran
smission: 0.00
 #7051 (0000B1) [S] :Brick wall (aerated) Materials: |0000B1|0000B1|0000B1| Thickness: 0.4500 m, T
ransmission: 0.00
 #7052 (0000B2) [S] :Brick wall (burned) Materials: |0000B2|0000B2|0000B2| Thickness: 0.4500 m, Tr
ansmission: 0.00
 #7053 (0000B3) [S] :Brick wall (reinforced) Materials: |0000B3|0000B3|0000B3| Thickness: 0.4500 m
, Transmission: 0.00
 #7054 (0000R1) [S] :Roofing: Tile Materials: |0000R1|0000R1|0000R1| Thickness: 0.3000 m, Transmis
sion: 0.00
 #7055 (0000R2) [S] :Roofing: Terracotta Materials: |0000R2|0000R2|0000R2| Thickness: 0.3000 m, Tr
ansmission: 0.00
 #7056 (0000GH) [S] :Heat protection glass (one air layer) Materials: |0000G1|0000O2|0000G1| Thick
ness: 1.5000 m, Transmission: 0.81
 #7057 (0000WR) [S] :WetRoof Materials: |0000WC|0000C2|0000C2| Thickness: 0.9000 m, Transmission:
0.00
 #7058 (0000SG) [S] :Shading Plexiglass Materials: |0000SG|0000SG|0000O2| Thickness: 0.0300 m, Tra
nsmission: 0.36
 #7059 (0000CS) [S] :Concrete slab (hollow block, default) Materials: |0000C3|0000C3|0000C3| Thick
ness: 0.3000 m, Transmission: 0.00
 Total: 60 Walls out of 999 used.
 -------------------------------------
-------------------------------------
Single Wall Database
-------------------------------------
:: Getting Single Wall Database (System):  OK. Found 2 items
:: Getting Single Wall Database (projectdatabase.edb):  OK. Added 0 items
:: Single Wall Database: Check validity...  OK
 -------------------------------------
Single Wall Database
 -------------------------------------
 #4000 (000001) [S] :SunSail Material: |PV|
 #4001 (0000BW) [S] :BrickWall Material: |B2|
 Total: 2 single walls out of 999 used.
 -------------------------------------
-------------------------------------
Greenings Database
```

Figure 21: Screenshot of thermal comfort simulations in GH using the "blob geometry example" (see GH definition, Appendix C)

In summary, the developed GH definition can be compiled as a GH component which can then be used to evaluate *ecolopes* design outcomes based on human comfort analysis. However, there are some limitations of this approach, in particular concerning the possibility to model in a more precise way vegetations' characteristics and vegetation performance. Currently, there is no possibility to model plants directly in GH using the ALBERO database, which is provided by Envi-met. Further research will be addressed on this topic in the WP7 (in particular within Task 7.1: Human comfort and wellbeing).

6.5 Conclusions

Section 6 presented the current development stage of the ECOLOPES platform architecture and its integrated software components including the ECOLOPES frontend tool in Grasshopper. The focus was on an overview and a technical description of the developed software components (e.g., where software components are stored, what Operating System (OS) they run in, where the source code is, etc.). The current version (= Sandbox) of the ECOLOPES platform has an already integrated cloud-based backend which still needs to be connected with the developed ECOLOPES frontend tool in Grasshopper (sections 6.3 and 6.4). This process implies the compilation of the already developed Grasshopper components into the right format and testing, which will be addressed in the next version.

7. Future application – the Vienna case study

An urban example site near the Danube river in the city of Vienna was used to test the layouted computational processes and the ECOLOPES platform. Section 7 briefly presents how the computational system can be applied to a precise site using context-specific data inputs: (1) normative framework (Masterplan provided by the city of Vienna), (2) open-climate data from the city of Vienna, and (3) local species pool data. A Grasshopper definition for KPI-based design generation and optimisation was developed and tested with the goal to integrate KPI-based design optimisation in the ECOLOPES platform (WP3).

First, 2D data and 3D assets from OpenStreetMap and MapBox are imported into Rhino through the Rhino plugins LandsDesign and Ladybug Tools (Figure 22, Figure 23).

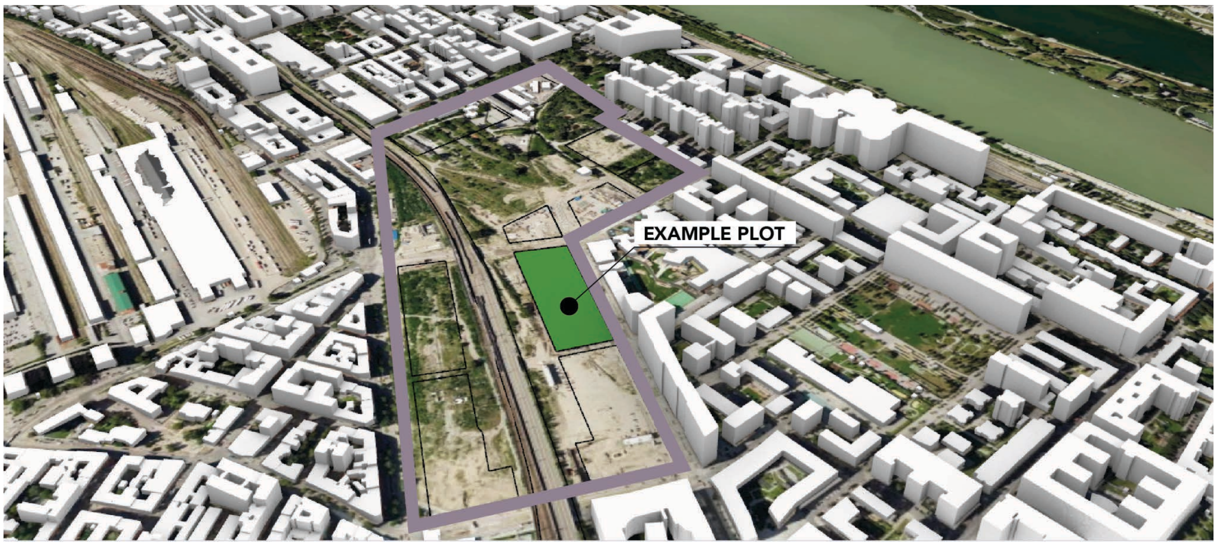
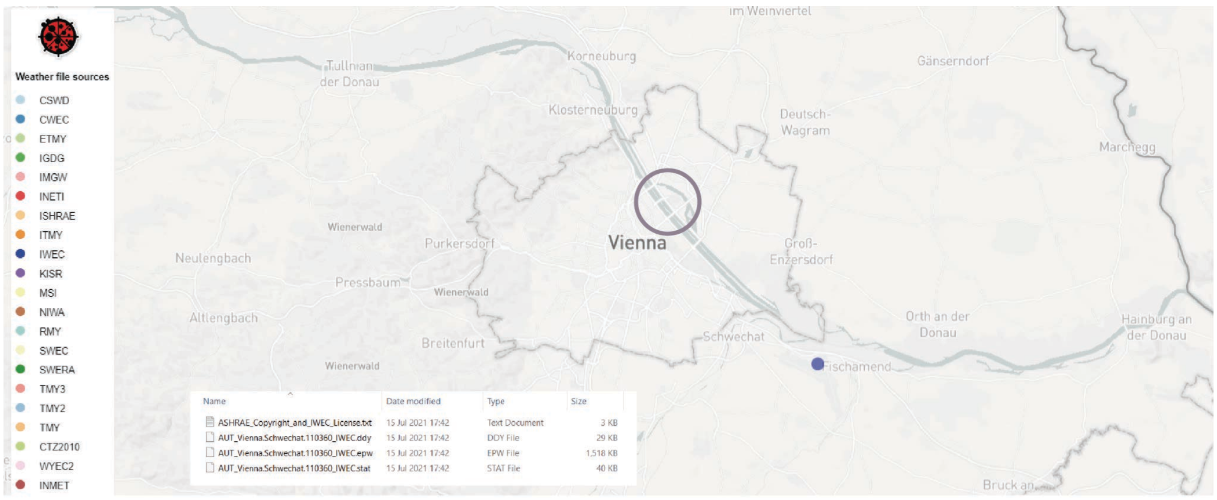


Figure 22: 3D model of an example site in Vienna. 3D geometry data imported through LandsDesign into Rhino/ Grasshopper (MapBox).



Name	Date modified	Type	Size
ASHRAE_Copyright_and_IWEC_License.txt	15 Jul 2021 17:42	Text Document	3 KB
AUT_Vienna.Schwechat.110360_IWEC.ddy	15 Jul 2021 17:42	DDY File	29 KB
AUT_Vienna.Schwechat.110360_IWEC.epw	15 Jul 2021 17:42	EPW File	1,518 KB
AUT_Vienna.Schwechat.110360_IWEC.stat	15 Jul 2021 17:42	STAT File	40 KB

Figure 23: Open climate data import for the city of Vienna. Data imported from MapBox and OpenStreetMaps through Ladybug tools.

In the next step, design generation and optimisation was initiated through Wallacei, an evolutionary solver in Grasshopper. In our example we tried to optimise a building envelope design for plant growth. For testing purposes, we combined architectural parameters (building mass and building height) with shading (from the existing and added reference buildings) as an example environmental parameter with known effect on the ecological community (Figure 14, green geometry). In the future version, the Grasshopper definition will refer to the new correlations between architecture, environment and ecology computed in the Knowledge generation framework (KGF).

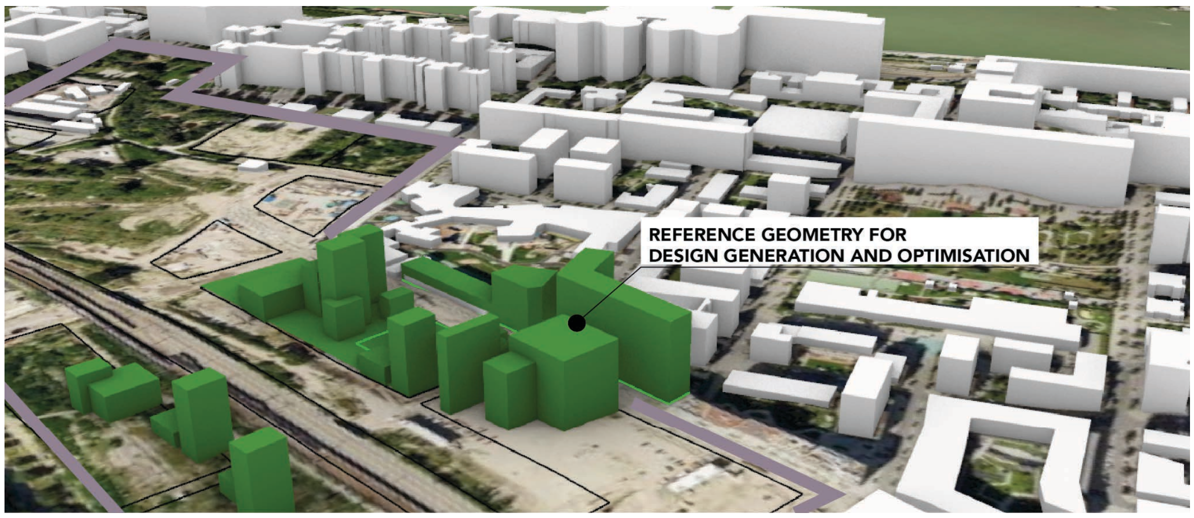


Figure 24: Reference geometry (green) to generate and optimise a design based on architectural and environmental parameters in Wallacei/ Grasshopper.

In the final step, we computed in an iterative process multiple design outcomes with Wallacei, a free evolutionary multi-objective optimisation and analytic engine for Grasshopper (Wallacei, 2020). Our first results show a series of generated design outcomes based on the architectural and environmental input parameters (Figure 25). In an optimisation process, design outcomes are evaluated based on the solutions that are most likely to meet the defined fitness objectives (blue graphs).

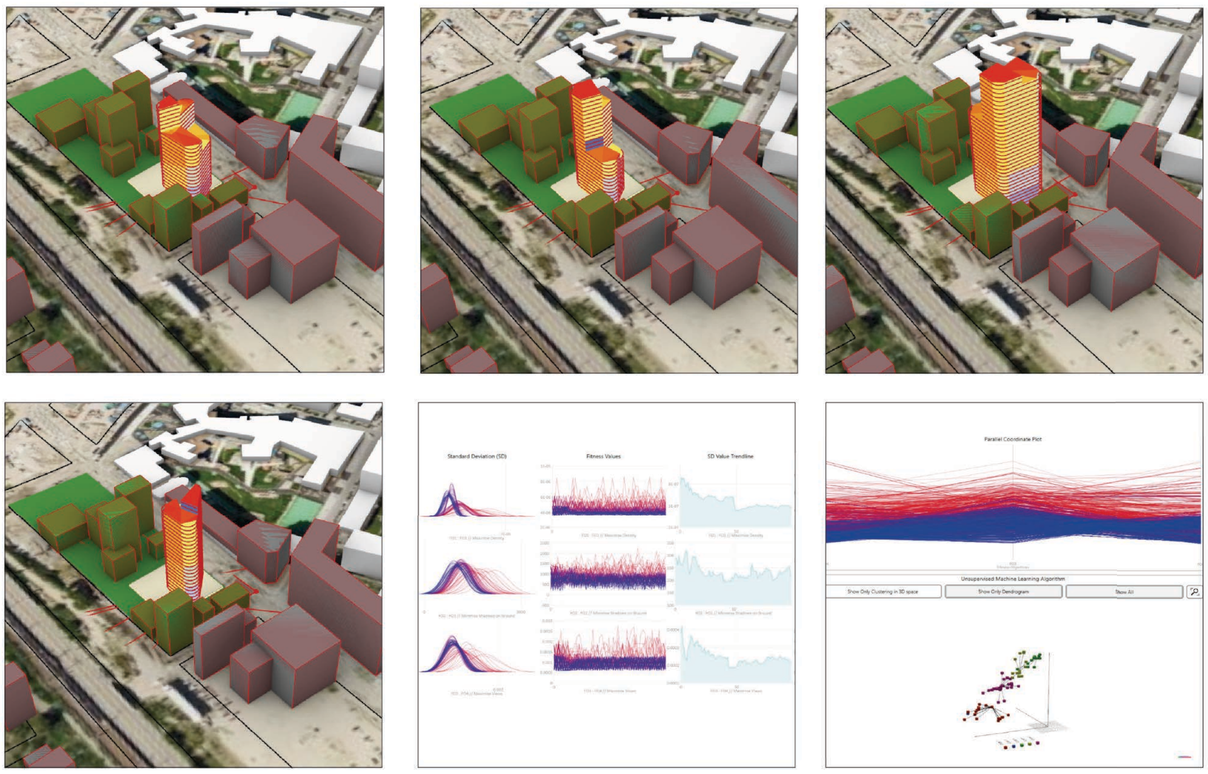


Figure 25: ECOLOPES platform – Grasshopper algorithm for design generation and optimisation that is ready to use KPI inputs from local sites- specific architectural, environmental and ecological conditions.

In summary, a Grasshopper definition to read and use Key Performance Indicators (KPIs) for optimisation was developed and the adaptability of the ECOLOPES platform was tested on an example site in Vienna. In the future version, the algorithm can be adapted to input architectural, environmental and ecological KPIs that will be calculated in the Knowledge generation framework and accessible through the Knowledge base (KB).

8. Conclusions and recommendations for the interim ECOLOPES platform

This report has described the draft version of the ECOLOPES platform development and its backend and frontend deployment. It has showcased the platform architecture and discussed its capabilities with respect to related applications.

It has also presented on the current state of the computational framework and specified how different components of the platform are chained, and how data needs to be managed to support the processes inherent in the ECOLOPES approach.

It described in detail first results of how ecological modelling is intrinsically integrated in design to create the platform's knowledge generation framework, which enables the collection and exploitation of fundamental data pertaining to the relationship between architecture, environment and ecology.

Based on the conceptual, computational and the knowledge generation framework, the requirements in terms of data exchange and interfacing mechanisms were further elicited, documented, and applied for the development of the draft version of the ECOLOPES platform. The drafted system architecture enables the integration of interdisciplinary software components and for data passing through the system. The core achievement is that ecological modelling was integrated in a 3D CAD system (ECOLOPES Grasshopper frontend tool). Thus, ecological modelling has become an intrinsic part of a parametric and data/information-driven design process to consider the requirements of multiple organisms. The developed system is non-deterministic as it constantly generates knowledge which can be used for more effective design-decision support.

In addition, the overall architecture model was developed to support a scalable and flexible (cloud-based) system, capable of supporting the research activities conducted by experts as well as use cases brought forward by architects and designers (Rhino.Compute backend).

In summary, the major achievement for the draft version of the ECOLOPES platform architecture was the development and integration of the main software components considering the technical requirements, technical design, digital infrastructure and 1st prototype defined in D3.1. Platform components were systematically connected end-to-end to enable knowledge generation, first KPI calculations, initial data in the KB, and testing and validation of the developed system. Analytical algorithms (Grasshopper definitions) and programs (C++ ecological model) that support the design, analysis and evaluation of envelopes were deployed. Thus the execution of the ECOLOPES design, analysis, and optimisation processes can partially be processed end-to-end. Furthermore, a first ECOLOPES data model was elaborated that governs how data is exchanged between ecological (raster data) and geometrical components (3D geometry data).

For the next version of the ECOLOPES platform (D3.3), further algorithms have to be developed and integrated, existing ones optimised and computational processes made more efficient. The system needs to become responsive to changing local context-specific and additional data inputs (e.g., local species pool, soil types, building envelope and site materials, terrain models, etc.) for a more precise analysis of building envelopes designs. Such an endeavour leads to an increasing complexity of the system, which requires data management and maintenance strategies. In respect to the Knowledge generation framework (KGF), correlations between architectural, environmental and ecological parameters need to be validated based on multiple experiments with changing geometry typology inputs. Data in the KB needs to be prepared to be used as new knowledge for design decision support which requires the development and implementation of machine learning and reasoning methods in the computational process. Lastly, all developed software components need to be compiled as a series of frontend tools (Grasshopper-, web- , and stand-alone tool) for testing and the design of physical prototypes.

REFERENCES

Research Paper:

Bocedi G, Palmer SCF, Malchow AK, Zurell D, Watts K, & Travis JMJ (2021) RangeShifter 2.0: An extended and enhanced platform for modelling spatial eco-evolutionary dynamics and species' responses to environmental changes. Ecography, 44: 1453-1462.

Boulangeat, I., Georges, D., & Thuiller, W. (2014). FATE-HD: A spatially and temporally explicit integrated model for predicting vegetation structure and diversity at regional scale. *Global Change Biology*, 20, 2368–2378.

Duering, S., Chronis, A., & Koenig, R. (2020). Optimizing urban systems: integrated optimization of spatial configurations. In *Proceedings of the 11th Annual Symposium on Simulation for Architecture and Urban Design* (*SimAUD '20*). Society for Computer Simulation International, San Diego, CA, USA, Article 74, 1–7. https://dl.acm.org/doi/10.5555/3465085.3465159.

Li, Y., Huang, C., Zhang, Z., Yao, J. (2022). Machine Learning Modeling and Genetic Optimization of Adaptive Building Façade Towards the Light Environment. *POST-CARBON, Proceedings of the 27th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA)*, Volume 1, 141-150.

Makki, M., Navarro-Mateu, D., Showkatbakhsh, M. (2022). Decoding the Architectural Genome: Multi-Objective Evolutionary Algorithms in Design. Technology | *Architecture + Design*. 6, p. 68–79.

Petrov, M. and Walker, J. (2020). Optioneering Methods for Optimization - Methods of exploring primary and secondary performance criteria in urban design. *Proceedings of the 38th eCAADe Conference* - Volume 1, TU Berlin, Berlin. pp. 29-36.

Cloud-based platform technologies for building and urban planning:

SPACIO (2022). Cutting-edge browser-based building design tool for architects and engineers (beta version). [Online]. Available: https://www.spacio.ai/. [Accessed: 26-October-2022].

CityPLAIN (2021). A cloud computing tool for urban planning. [Online]. Available: https://www.cityplain.com/. [Accessed: 2-March-2022].

Flux.Broward.Land (2021). Planning for uncertainty preparedness, mitigation & resilience. [Online]. Available: https://broward.flux.land/. [Accessed: 23-March-2022].

InFraReD (2021). Intelligent Framework for Resilient Design. [Online]. Available: http://infrared.city/. [Accessed: 23-March-2022].

KPF UI (2022). KPF urban interface for urban data analytics for informed decision making in the design of buildings and cities. [Online]. Available: https://ui.kpf.com/. [Accessed: 21-March-2022].

Scout.Build (2021). Spatial and temporal urban data analytics for informed decision making in the design of buildings and cities by KPF. [Online]. Available: https://scout.build. [Accessed: 23-March-2022].

Swarm (2021). [Online]. Available: https://swarm.thorntontomasetti.com/. [Accessed: 23-March-2022].

Software libraries and tools:

Grasshopper by David Rutten (2008). [Online]. Available: https://www.grasshopper3d.com/. [Accessed: 23-October-2022].

Grasshopper component (2018). [Online]. Available: https://developer.rhino3d.com/guides/grasshopper/what-is-a-grasshopper-component/. [Accessed: 17-October-2022].

GitHub Rhino.Compute McNeel (2021). REST geometry server based on RhinoCommon and Grasshopper [Online]. Available: https://github.com/mcneel/compute.rhino3d. [Accessed: 10-October-2022].

GitHub Rhino.Inside (2021). [Online]. Available: https://github.com/mcneel/rhino.inside. [Accessed: 14-October-2022].

Iron Python (2009). [Online]. Available: https://ironpython.net/. [Accessed: 21-October-2022].

Ladybug Tools (2013). Free environmental design knowledge and tools. [Online]. Available: https://www.ladybug.tools/. [Accessed: 23-October-2022].

LandsDesign (2021) [Online]. Available: https://www.landsdesign.com/. [Accessed: 21-October-2022].

Lunchbox (2012). Grasshopper for exploring mathematical shapes, panelling, structures, and workflow. [Online]. Available: https://www.food4rhino.com/en/app/lunchbox. [Accessed: 2-October-2022].

Octopus (2012) [Online]. Available: https://www.food4rhino.com/en/app/octopus. [Accessed: 12-October-2022].

Rhino 7 McNeel (2021). [Online]. Available: https://www.rhino3d.com/7/new/. [Accessed: 12-October-2022].

Rhino.Compute Guides McNeel (2021). [Online]. Available: https://developer.rhino3d.com/guides/#compute. [Accessed: 1-October-2022].

Rhino.Compute AppServer (2021). A node.js server acting as a bridge between client apps and private compute.rhino3d servers. [Online]. Available: https://github.com/mcneel/compute.rhino3d.appserver. [Accessed: 19-October-2022].

RhinoCommon, 2018 https://developer.rhino3d.com/guides/rhinocommon/what-is-rhinocommon/. [Accessed: 22-October-2022].

Rhino Development (2018). Rhino and Grasshopper Developer Documentation. [Online]. Available: https://developer.rhino3d.com/. [Accessed: 5-October-2022].

Rhino.Inside McNeel (2021). Rhino and Grasshopper inside 64 bit applications for Windows. [Online]. Available: https://www.rhino3d.com/features/rhino-inside/. [Accessed: 14-October-2022].

Rhino.Inside.Revit McNeel (2021). Rhino and Grasshopper inside Revit. [Online]. Available: https://www.rhino3d.com/inside/revit/beta/. [Accessed: 11-October-2022].

Rhino SDK (2018). The Rhino C/C++ Software Development Kit. [Online]. Available: https://developer.rhino3d.com/guides/cpp/what-is-the-cpp-sdk/. [Accessed: 14-October-2022].

Rhino.Python (2018). [Online]. Available: https://developer.rhino3d.com/guides/rhinopython/what-is-rhinopython/. [Accessed: 5-October-2022].

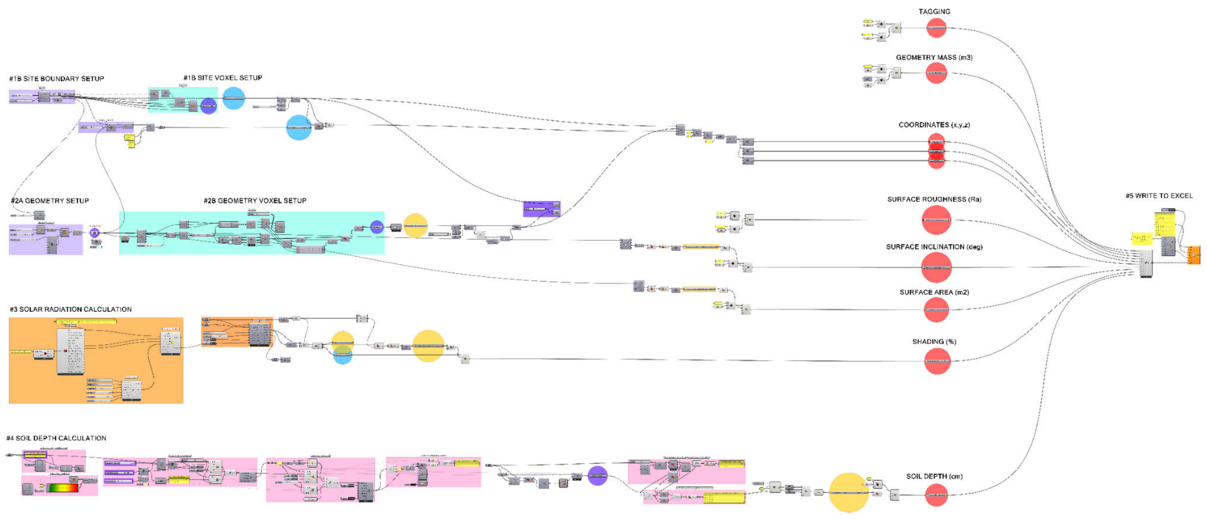
Wallacei (2020). An Evolutionary Multi-Objective Optimisation and Analytic Engine for Grasshopper. [Online]. Available: https://www.wallacei.com/. [Accessed: 23-October-2022].

APPENDIX

A: ECOLOPES_Dataset_ Specification – See deliverable D3.1.

B: KGF Grasshopper definition

The Grasshopper definition (.gh) to generate the shading and soil depth information for each voxel cell that outputs the .csv file that is required as input for ecological analysis.



C: Human comfort analysis Grasshopper definition

The Grasshopper definition (.gh) to simulate human thermal comfort (model setup & workspace definition, materials definition, simulation setup) in a random environment with the “blob example”.