# ECOLOPES

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

H2020-FET-OPEN-2021-2025

Action number 964414

# D3.3: Interim ECOLOPES platform architecture

| | |
|---|---|
| **Dissemination level:** | Public |
| **Contractual date of delivery:** | Month 29, 31 August 2023 |
| **Actual date of delivery:** | Month 29, 03 August 2023 |
| **Work package:** | WP3 |
| **Task:** | T3.3 and T3.4 |
| **Type:** | Report |
| **Approval Status:** | Final version |
| **Version:** | v1.0 |
| **Number of pages:** | 60 |
| **Filename:** | D3.3_InterimEcolopesPlatformArchitecture_20230803_v1.0.pdf |

**Abstract:** The aim of D3.3 is to report on the progress made in Work Package 3 over the past 10 months (since D3.2 and Report of year 2). D3.3 is the last WP3 report before submission of the demonstrator in M38. It illustrates the updated ECOLOPES platform including the computational workflow, component integration, and the cloud-based platform architecture (Section 2). Section 3 reports on the progress made with respect to the integration of ecological analysis in a 3D CAD system. Chapter 4 reviews the advances that have been made regarding the Knowledge Generation Framework (KGF), the Knowledge Base prototype (KB), and the ML model. Chapter 5 reports on the development progress of the ECOLOPES frontend tool, a Grasshopper plugin for ecological analysis in CAD. Lastly, Chapter 6 and 7 inform about WP3 related development strategies and dissemination activities.

co-funded by the European Union

## VERSION HISTORY

| Version | Date | Reason | Revised by |
|---|---|---|---|
| v0.1 | 07.06.2023 | ToC | Verena Vogler, Jens Joschinski, Wolfgang Weisser |
| v0.2 | 21.06.2023 | Draft | Verena Vogler, Jens Joschinski |
| v0.3 | 29.06.2023 | Draft | Verena Vogler, Jens Joschinski |
| v0.4 | 17.07.2023 | Draft | Verena Vogler, Jens Joschinski, Francesca Mosca |
| v0.5 | 25.07.2023 | Draft | Verena Vogler, Jens Joschinski, Wolfgang Weisser |
| v1.0 | 03.08.2023 | Final | Verena Vogler, Jens Joschinski, Wolfgang Weisser |

## AUTHOR LIST

| Organisation | Name | Contact information | Contribution |
|---|---|---|---|
| MCNEEL | Verena Vogler | verena@mcneel.com | Development, writing, graphs, tables |
| SAAD | Jens Joschinski | jens.joschinski@animal-aided-design.de | Development, writing/ graphs and table for ecological model |
| UNIGE | Francesca Mosca | francesca.mosca@edu.unige.it | 6.3 Validation Tools |
| MCNEEL | Kay Eckelt | kay@mcneel.com | Development Machine learning model & plots |

# EXECUTIVE SUMMARY

The Interim ECOLOPES platform deliverable aims to provide an overview about the latest research, development and integration efforts conducted in WP3. The report focuses on the integrated technical components, workflows, front-end tools and data outputs that are relevant to the WP3 platform demonstrator (M38) as well as for the WP4-WP7.

After an introduction that links our current work with previously reported content, Chapter 2 elaborates more details concerning the ECOLOPES computational platform based on Rhino.Compute, integrated technical components, and an updated version of our data flow chart (computational workflow).

The following Chapter 3 presents how we achieved to integrate a preliminary version of the Ecological model, a plant model, in a 3D CAD system (Rhino/Grasshopper). This important step enables ecological analysis of 3D building envelopes in CAD. Thus, ecological analysis becomes an intrinsic part of the design process for building envelopes. For the demonstrator, we want to integrate the animal model using the same method.

The next Chapter 4 explains in detail the Knowledge Generation Framework (KGF), preliminary experiments, data outputs to train a Machine learning (ML) model, our ML pipeline for computing KIP ranges and rules for design. Future steps are to generate more data for our ML pipeline which includes 3D modelling of more buildings and write algorithms for additional building features.

Chapter 5 introduces in more depth the preliminary version of our ECOLOPES Grasshopper plugin, the platform's front-end tool. In a future version, parts of Grasshopper HOPS components might need to be substituted by C# components to increase their efficiency.

Lastly, Chapter 6 reports on our development strategies including communication, documentation, version control, and licensing while Chapter 7 presents WP3 related public outreach/ feedback and publishing plans.

The report concludes with recommendations for the technical demonstrator of the ECOLOPES platform in M38.

## ABBREVIATIONS, DEFINITIONS 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 |
| **CFD** | Computational Fluid Dynamics |
| **Computational framework** | It is the ECOLOPES computational system that defines technical components and the way they are connected. |
| **Computational workflow** | The computational workflow is a visual representation of the information that passes through the ECOLOPES computational system. It uses various symbols and labels to depict the movement of data between different technical components. |
| **ECOLOPES** | Ecological building envelopes |
| **ECOLOPES platform** | The ECOLOPES platform is a piece of software components running on a Windows cloud server/ front-end on a local machine. The platform supports 3D parametric modelling, analysis and simulations in Rhinoceros®/ Grasshopper® through cloud computing. |
| **FG** | Functional groups |
| **GH** | Grasshopper®, a visual programming interface for Rhinoceros® developed by McNeel |
| **KB** | Knowledge base |
| **JSON** | JavaScript Object Notation (file format for data exchange) |
| **KGF** | Knowledge generation framework |
| **KPI** | Key performance indicators |
| **ML** | Machine learning |
| **PFG** | Plant functional groups, i.e. groups of plant species with similar effects on ecosystem functioning |

| | |
|---|---|
| **Rhino** | Rhinoceros, a 3D free-form NURBS modelling software and cross-platform open developer platform |
| **SDK** | Software development kit |
| **TOPSIS** | Technique for Order of Preference by Similarity to Ideal Solution (algorithm) |
| **UI** | User interface |
| **Voxel model** | The ECOLOPES computational framework uses two different types of voxel models for different purposes: First, a 3D voxel model to overcome the interoperability between 2D raster data and 3D geometry data (WP3, this deliverable), and second, a 2.5D voxel/point cloud model elaborated for the ontology-aided design process (WP5, D5.1). |
| **WP** | Work package |

## TABLE OF CONTENTS

# 1. INTRODUCTION

In ECOLOPES, we propose a radical change for urban development. Instead of minimising the negative impact of urbanisation on nature, our aim is to create a prototype of a technology to design cities where nature and humans can coexist. The vision entails an integrated approach to architecture that prioritises the requirements of humans, plants, animals, and other associated organisms such as microbiota. Therefore, we specifically focus on the scale of the building envelope, with the goal to convert this envelope into an *ecolope*, a multi-species living space with the potential to accommodate humans, plants, animals, and microbiota. This vision was put into practice during the first two years of the ECOLOPES projects, when technological components were developed by interdisciplinary experts, and partially integrated into a computational platform. By bringing ecological knowledge into the architectural design process, we expect to find solutions that foster synergies and minimise conflicts among the inhabitants. This computational multi-species approach has the potential to design *ecolopes* that restore beneficial relationships between humans and nature, revitalising our cities.
The "Interim version of the ECOLOPES platform" is presented in this deliverable with the focus on the latest technical developments and implementations, enhancements and the missing steps towards the development of the prototype demonstrator of the ECOLOPES platform in M38 of the ECOLOPES project. Topics discussed in this report build on the concepts and developments introduced in the previously presented reports, the public deliverables D3.1 and D3.2, and the Report of the Second Year. Several advances have been made in developing the computational tools of ECOLOPES. The ECOLOPES platform architecture and the computational workflow was completed in 2022 and presented in detail in D3.2 (M19). The Knowledge Generation Framework (KGF), a set of computational experiments to extract relationships between architectural, environmental and ecological parameters was initiated. In the year 2023 we completed the first version of the ECOLOPES Grasshopper plugin (frontend tool) that successfully integrated the Ecological Model (WP4) into a 3D CAD system. Thus, ecological modelling has become an intrinsic part of the form-finding process for parametric building envelopes.
Furthermore, the following technical achievements with respect to the ECOLOPES computational system were made:

1. The integration of open and expert databases
2. Working environmental models that use the database as input, and whose output is used by the ecological model:
   a. A shading model using Ladybug tools in Grasshopper (Ladybug tools 2013)
   b. A soil depth model
3. An ecological model that uses the environmental models and 3D geometry as input, and simulates plant, animal and soil communities

4. The Knowledge base which stores the inputs and outputs of simulations testing the response of the *ecolope* ecosystem to the building geometry (geometries, environmental and ecological model outputs; requiring manual extraction)
5. A design evaluation and optimisation environment based on Wallacei MCDM and MOO (Wallacei, 2020)

Tasks 3.1 "ECOLOPES system architecture" (M1-12) and Task 3.2 "ECOLOPES data warehousing" (M5-M29) were completed with the outcomes of a system architecture for the ECOLOPES design platform and an integrated cloud-based data warehousing infrastructure with a draft version of the ECOLOPES database – the Knowledge Base (KB). Section 2.3 of the report will demonstrate these achievements. All milestones except parts of Milestone 2 "Proof of concept, initial data collection and architecture" (M12) and Milestone 3 "1st release and basic ECOLOPES algorithms" (M20) were reached, as the basic EIM Ontologies (WP4 and WP5) proved unexpectedly challenging. To navigate around the challenge, the ECOLOPES computational framework has been set up to also work without a functional ontology, but operate using a Knowledge base (KB). To generate knowledge on the relationship between architectural design and ecological outcome, ecological, environmental and architectural parameters were combined in the Knowledge Generation Framework (KGF). On a technical level, the KGF successfully resolved challenges associated with interoperability between 2D raster data from the Ecological Model and 3D geometry data (see Chapter 3). Thus, the WP3 related developments, integrations and gathered data presented in this deliverable rather reflect on the implemented KGF approach. In the future, more elaborated EIM ontologies could be integrated additionally to the KGF approach.
Under active development are sub-task related to Task 3.3 "Backend development and integration" (M5-38) as the development of cloud-based backend services as well as a continuous integration and improvement of the platform according to T3.1 (see Section 2.3). Furthermore, Task 3.4 "Frontend development" (M9-38), already outputs an early-stage prototype of a Grasshopper plugin (see Chapter 5) that enables ecological analysis within a CAD system at a resolution of 1m x 1m.
Recommendations from the previous report D3.2 included the development and integration of new Grasshopper components, debugging and optimisation of existing ones, as well as the computational processes, the Ecological model and data exchange formats needed to be more efficient. Furthermore, the developed system was still not 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. Some of these points were successfully addressed and will be presented in this report. With respect to the KGF, correlations between architectural, environmental and ecological parameters, and KPIs needed can now be explored through multiple experiments with changing geometry inputs. Such a process was started and generated a tremendous amount of data points during each experiment (see Chapter 3 and 4). Thus, as a next step in

the development, the KB will include machine learning and reasoning methods to efficiently optimise data and to extract required rules and KPI ranges for design. Lastly, the developed Grasshopper functional components are ready to be compiled and tested as a frontend tool. This report will present an early-stage prototype of the frontend tool which is rather a tool that enables ecological analysis of a custom envelope design than a design recommendation system.

In conclusion, D3.3 will provide information about the latest development activities in relation to the ECOLOPES computational system and its technical prototypes. Additionally, it is the last WP3 report before submission of technical demonstrators/ prototypes in M38. Therefore, another main focus of the report is to introduce guidelines and measurements for the compilation of technical demonstrators of the ECOLOPES platform and the ECOLOPES frontend tool.

# 2. The ECOLOPES computational platform

The ECOLOPES computational platform is a design recommendation system for the design of ecological building envelopes in urban environments with a potential to be extended to the scale of urban regions. The design generation process is informed by a set of rules and data that emerge from a series of computational experiments (KGF), resulting in knowledge that will eventually be implemented in the EIM Ontologies. Architectural and ecological design objectives and KPIs are part of the system to optimise and validate design outcomes.

The ECOLOPES computational platform has many technical components that are developed jointly by the consortium: open and expert databases, environmental models, the ecological model, the Knowledge base (KB), the design optimisation environment, and validation. The following section reports on the updated computational workflow and its integrated components, and on the cloud-based system architecture of the ECOLOPES computational platform (Figure 1). Chapter 3 explains the integration of the Ecological Model in a CAD system.

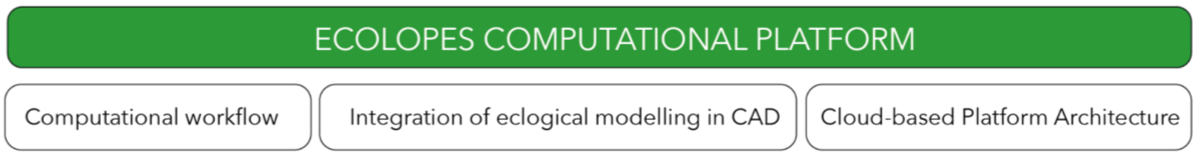

ECOLOPES COMPUTATIONAL PLATFORM
Computational workflow
Integration of eclogical modelling in CAD
Cloud-based Platform Architecture

*Figure 1: The ECOLOPES computational platform.*

## 2.1 Integrated technical components and data flows

### 2.1.1 Computational workflow for the visualisation of data flows

The initial version of the computational workflow was developed in WP3 and introduced in D3.1. The aim of the computational workflow is to map out the information that passes through the ECOLOPES system. This graphical representation of the flow of data within a system is also known as a data flow diagram (DFD). Figure 1 layouts the updated version of the computational workflow which uses various symbols and labels to depict the movement of data between different components of the system:

- ***Rectangles:*** These represent processes or activities that transform or manipulate the data (e.g., the Ecological Model or the Data-driven generative design, or the Voxel Model). These processes are developed by interdisciplinary consortium members. For instance, while the local and regional ecological models (green) are developed by the team of ecologists, the form generation and optimisation environment is developed by architects (pink).
- ***Arrows:*** Data flows are represented by arrows and depict the movement of data between different components in the system (processes, entities, and data stores). Arrows show the direction of data movement, from a source to a destination and data type (file format).
- ***Data stores:*** These are represented by containers and represent a storage point for data. They could be databases, files, or any other data repositories.
- ***Blue parallelogram:*** User intervention.

Throughout the project, the elaborated computational workflow helped in understanding the system's data flow and identifying potential bottlenecks, dependencies, or areas for improvement. It visually explains the ECOLOPES computational system that would be hard to explain in words, and it works for both technical and nontechnical partners.

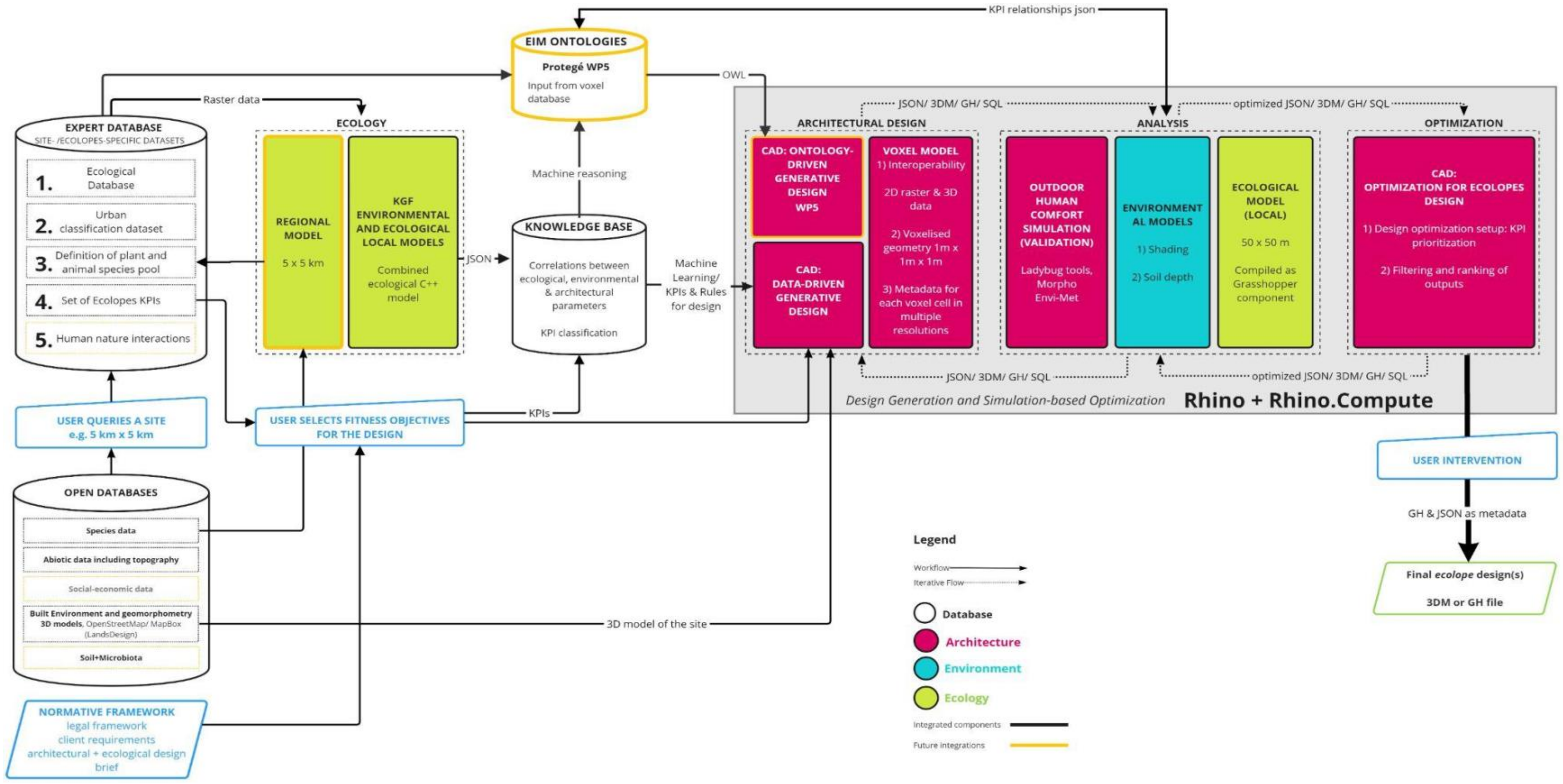

KPI relationships json
EIM ONTOLOGIES
Protegé WP5
Input from voxel database
OWL
Raster data
EXPERT DATABASE
SITE- /ECOLOPES-SPECIFIC DATASETS
1. Ecological Database
2. Urban classification dataset
3. Definition of plant and animal species pool
4. Set of Ecolopes KPIs
5. Human nature interactions
ECOLOGY
REGIONAL MODEL
5 x 5 km
KGF ENVIRONMENTAL AND ECOLOGICAL LOCAL MODELS
Combined ecological C++ model
JSON
Machine reasoning
KNOWLEDGE BASE
Correlations between ecological, environmental & architectural parameters
KPI classification
Machine Learning/ KPIs & Rules for design
JSON/ 3DM/ GH/ SQL
optimized JSON/ 3DM/ GH/ SQL
ARCHITECTURAL DESIGN
ANALYSIS
OPTIMIZATION
CAD: ONTOLOGY-DRIVEN GENERATIVE DESIGN WP5
CAD: DATA-DRIVEN GENERATIVE DESIGN
VOXEL MODEL
1) Interoperability
2D raster & 3D data
2) Voxelised geometry 1m x 1m x 1m
3) Metadata for each voxel cell in multiple resolutions
OUTDOOR HUMAN COMFORT SIMULATION (VALIDATION)
Ladybug tools, Morpho Envi-Met
ENVIRONMENTAL MODELS
1) Shading
2) Soil depth
ECOLOGICAL MODEL (LOCAL)
50 x 50 m
Compiled as Grasshopper component
CAD: OPTIMIZATION FOR ECOLOPES DESIGN
1) Design optimization setup: KPI prioritization
2) Filtering and ranking of outputs
JSON/ 3DM/ GH/ SQL
optimized JSON/ 3DM/ GH/ SQL
Design Generation and Simulation-based Optimization
Rhino + Rhino.Compute
USER QUERIES A SITE
e.g. 5 km x 5 km
USER SELECTS FITNESS OBJECTIVES FOR THE DESIGN
KPIs
USER INTERVENTION
GH & JSON as metadata
Final ecolope design(s)
3DM or GH file
OPEN DATABASES
Species data
Abiotic data including topography
Social-economic data
Built Environment and geomorphometry 3D models, OpenStreetMap/ MapBox (LandsDesign)
Soil+Microbiota
3D model of the site
NORMATIVE FRAMEWORK
legal framework
client requirements
architectural + ecological design brief
Legend
Workflow
Iterative Flow
Database
Architecture
Environment
Ecology
Integrated components
Future integrations

*Figure 2: Updated ECOLOPES Computational workflow and integrated components (black frame)*

Updates from the last version in D3.2 include the limitation of data exchange formats to JSON and Grasshopper files (.gh), the implementation of a Machine Learning Model to optimise the KB, and lastly, the limitation of Environmental Models to “Shading” and “Soil depth”.

### 2.1.2 Integration of technical components

For the integration, McNeel elaborated on the use of *Rhinoceros®* as an open development platform for the ECOLOPES plugin, allowing to write a library of custom algorithms in Grasshopper and C++, as well as for the development of a frontend tool (see Chapter 5). *Rhino.Compute* is used as a backend service to bring process-intensive algorithms into the cloud, and of *Hops* to wrap up algorithms for communicating with Rhino.Compute (see Section 2.3).
At this stage, there are five major modules of the computational workflow integrated within the ECOLOPES computational platform: open and expert databases, environmental models, the Ecological model, the Knowledge base (KB), the CAD and Voxel model, the design optimisation environment, and validation.

- ***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 environmental analysis to evaluate adequate areas for plant growth.
- ***Environmental models (WP3):*** Environmental models compute shading, soil depth (indirectly water retention), and shelter for parametric building envelopes at a specific site. This information serves as input for the Ecological model to calculate spatio-temporal dynamics of functional groups in a resolution of 1 sqm. The Environmental models provide variables to define performance indicators (KPIs) for multi-objective optimisation processes (shading, soil depth, information from local weather file).
  ***Ecological model (WP4):*** The Ecological model is written in the C++ programming language. It is a program compiled from interconnected sub-models (soil, plants and animals) that model the community dynamics of functional groups. As an input, it requires definitions of plant functional groups and animal functional groups, information which of these groups occur in the region and can potentially colonise the site, the soil types used on the site, and the Environmental model outputs. A realistic and matching set of 200 Plant Functional Groups (PFG), 18 Animal Functional Groups, and 42 soil classes was created via analysis of literature, trait databases and species occurrence data sets. The classification system allows for higher precision in the prediction of species abundance and system dynamics. The Ecological model provides variables (PFGs, AFGs in a resolution of 1 sqm) to define ecological KPIs.
- ***Knowledge base (KB) (WP3):*** The ECOLOPES Knowledge base (KB) stores data generated through the Knowledge Generation Framework experiments (KGF, D3.1 and

D3.2) in the JSON file format. The computational system in ECOLOPES uses the KB as its repository but the extraction of knowledge as such (relationships between parameters, KPIs and rules for decision-making) requires data optimisation strategies from Machine Learning. This can be attributed to the massive amount of spatio-temporal data generated by each experiment with over 1.500.000 data points.

- ***Voxel model for interoperability (WP3):*** The voxel model developed in WP3 differs in concept and implementation from the 2.5D voxel/point cloud model elaborated for the ontology-aided design process by TU Vienna. The voxel model presented in this deliverable is used as a method to overcome the interoperability between 2D raster data and 3D geometry data (Chapter 3).
- ***Ecological analysis in CAD (WP3-WP4):*** Ecological analysis in CAD refers to the novel possibility to analyse any kind of building envelopes for species abundance and biomass in Rhino and Grasshopper (Chapter 3 and Chapter 6).
- ***Optimisation environment (WP6):*** The optimisation environment inputs optimised KPI values and identifies KPI weights. Currently 4 KPIs are defined which are shading, soil depth, shading heterogeneity, and soil depth heterogeneity. A developed MADM Grasshopper component then sorts the range of optimised design solutions according to given architectural or ecological objectives defined. For this purpose, decision making and optimisation algorithms (TOPSIS) were developed in WP6 and are further explained in D6.1.
- ***Validation (WP7):*** Human thermal outdoor comfort analysis (see D7.1).

These components can already partly be connected. Exchange formats are JSON file format for numerical data and Grasshopper file format for 3D geometry data. In the year of 2023, components have been updated to accordingly adapt to this holistic data schema. Current obstacles are related to the massive amount of data points generated by each run which affects the calculation of optimised KPI ranges and the definition of rules for design. The obstacle ties in with risk 3 listed in the Grant Agreement (EIM overly complex, WP4). The challenge can be addressed by an additional machine learning and/or machine reasoning layer or due to a further simplification of the Ecological model (WP4).
***Future integrations:*** Lastly, there are components related to the ontology-aided generative computational design process (WP5) and the ontologies (WP4) that are prototyped and validated in parallel. Once developed and tested, they will be integrated in our computational system. In D4.2 (M30, Interim EIM Ontology), an updated version of the Ontologies developments including an in parallel to the Ecological model (validated C++ model written by ecologists) developed Ecological Networks Ontology (developed by architects and computer scientists) in Protegé, and an exported GraphDB for querying and reasoning are presented. Additionally, D5.3 (M30) elaborates on the ECOLOPES Computational Model for the ontology-aided generative computational design process including algorithms and custom processes involved. Another risk is related to time, as the technical demonstrator of the ECOLOPES

platform will be elaborated in the third year of the project and be submitted by M38, due to its complexity, both the ontology-aided generative computational design process (WP5) and the ontologies (WP4) might not be fully integrated.

## 2.2 Updated ECOLOPES platform architecture

The ECOLOPES platform is a software system for the design of ecologically sound building envelopes. It includes frontend tools in Rhino/Grasshopper (Windows only), and a backend that connects integrated technical components and caches and storages JSON, Grasshopper (.gh) and Rhino files (.3dm). In the report of Year 2, we explained that ecological and environmental analysis and simulations are computationally heavy processes, and that for this reason, a cloud-based system with scalable infrastructure was put into place. Another benefit of utilising a cloud-based system is the ability to leverage parallel computing. This type of architecture allows employing several processors with scalable CPUs to efficiently carry out numerous smaller computations in parallel. These computations are derived from a larger and more intricate problem, which is broken down into manageable tasks for simultaneous execution.
The cloud-based ECOLOPES platform was built on top of the Rhino.Compute, a framework developed by McNeel for 64 bits Windows applications that run in Rhino 7 (Figure 3) (Rhino.Compute 2021). Technically, Rhino.Compute is a web server that can perform geometry calculations using Rhino's geometry library (Rhino.Inside 2021). Rhino.Compute operates in the following manner: HTTP/HTTPS requests are sent to the Rhino.Compute server (in the case of the ECOLOPES project, a cloud server hosted by McNeel). The server then sends responses and clients such as Hops (Grasshopper on PC), the App.Server (Rhino web applications) as well as iRhino (Rhino on iPad) can communicate with the Rhino.Compute Server. Thus, geometry calculations can be performed on an external instance while users can interface with the Grasshopper front-end tools locally on their machines.

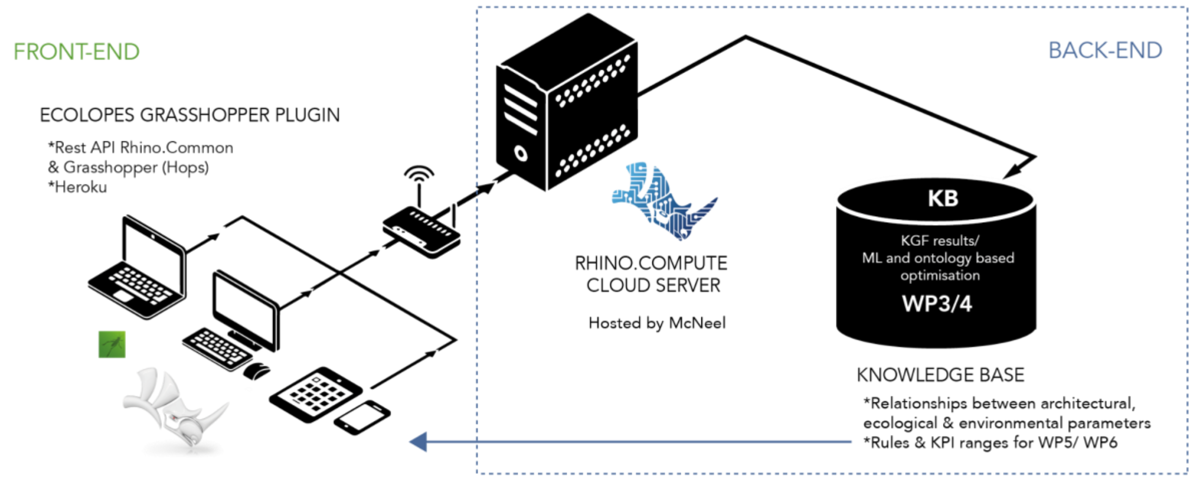

FRONT-END
ECOLOPES GRASSHOPPER PLUGIN
*Rest API Rhino.Common & Grasshopper (Hops)
*Heroku
BACK-END
RHINO.COMPUTE CLOUD SERVER
Hosted by McNeel
KB
KGF results/ ML and ontology based optimisation
WP3/4
KNOWLEDGE BASE
*Relationships between architectural, ecological & environmental parameters
*Rules & KPI ranges for WP5/ WP6

*Figure 3: Cloud infrastructure and ECOLOPES platform architecture based on Rhino.Compute.*

Grasshopper frontend tools are written as Hops components (Chapter 6), and their early versions are operational in Rhino 7 for Windows. Hops, the Rhino.Compute App.Server and Rhino.Compute were already introduced and further described in D3.1 and D3.2. The cloud-based infrastructure is fully scalable but depended on financial resources.
By the time of the delivery of D3.2, the system was tested for ecological analysis of building envelopes, KGF experiments, and to generate data for the KB (Chapter 5). The prototype of the KB is hosted on a Linux cloud server. It stores millions of data points providing information about the relationships between architectural, environmental, and ecological parameters, as well as KPI ranges and rules for design decision support.

## 2.3 Conclusions

Both the computational workflow with its technical components and the ECOLOPES platform architecture are guidelines for the joint development pipeline in the ECOLOPES project. While the computational workflow ensures data flow between technical components (WP3-WP7) throughout the project, the ECOLOPES platform with its cloud-based architecture is used as a test bed for early stage development. A prototype of the ECOLOPES platform and the Frontend tools for design will be demonstrated in M38 “Platform demonstrator”. Until this point, future developments with respect to the ECOLOPES platform include debugging and further development of existing components as well as the integration of the components that are still under development (Regional model, EIM ontologies, decision support system for design optimisation). In the project proposal, risks related to the ECOLOPES platform were laid out. For instance, Risk 4 highlights the hazard of the *“poor integration of different computational tools for WP5 and WP6”* (ECOLOPES, 2020). Furthermore, it points out that *“a common computational platform requires a gradual build-up of tools based on the extensive experience of VIE, TECH and partners involved”* (ECOLOPES, 2020). Deliverables D5.2, D5.3, D6.1, D7.1 address these risks and report on the current development state of non-integrated technical components.
With respect to the Ecological model component, the integration processes in a 3D CAD system is described in detail in the following Chapter 3.

# 3. Integration of Ecological Modelling in a 3D CAD System

An important and novel aspect of the ECOLOPES computational workflow is the Ecological model and the work needed to make ecological analysis available for decision-support and design optimisation. In the second year of the ECOLOPES project, we achieved the goal of technically integrating ecological modelling in a 3D CAD system which can then be used to inform urban planners to make more efficient design decisions that also consider the requirements of the urban ecosystem. In the first step, the Ecological model was developed by TUM, UNIGE, and SAAD. The model was then integrated into the Rhino/Grasshopper environment by McNeel. This section describes in more detail the updates of the developed computational system, which reflects the uniqueness, and interdisciplinary nature of the ECOLOPES research project and the way in which it interacts with emerging technologies and techniques/methods. Besides more general challenges for the computational system such as big data, high performance, databases, information systems, integrated and embedded hardware/software components, and networks, the main obstacle was to address interoperability issues between 2D raster file format of the Ecological model (which is a 2D representation of geographic information) and the 3D file format used for architectural design and analysis. For instance, if we look at an urban section, precise georeferenced and geometry information for the vertical parts of the urban landscape is missing in the raster file format. But especially these areas have a great potential to become multi-species habitats for non-human species (Figure 4).

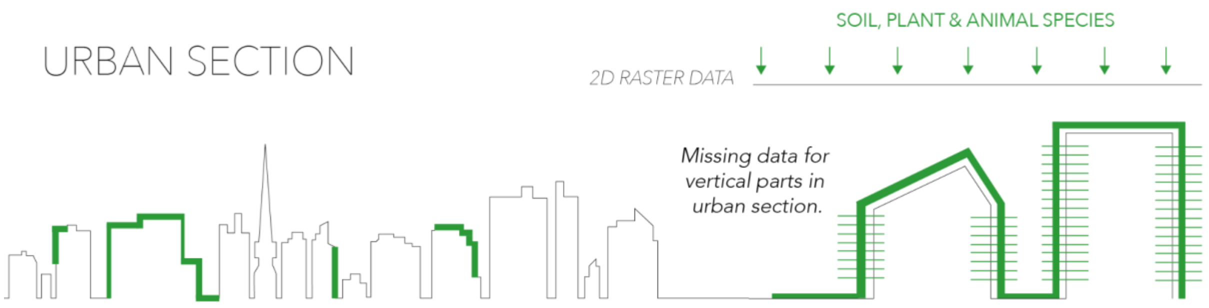

URBAN SECTION
SOIL, PLANT & ANIMAL SPECIES
2D RASTER DATA
Missing data for vertical parts in urban section.

*Figure 4: Urban section reveals vertical areas with a potential to become multi-species habitats. However, data for these areas is still missing and needs to be generated.*

To tackle this challenge, we developed a system that can generate this missing data for the ecological model input by using **voxel cells** (Figure 5). The term "voxel" is derived from combining the words *"volume"* and *"pixel"* and can be thought of as the 3D equivalent of pixels. Voxels are three-dimensional units to represent data in a volumetric space. Voxel-based representations offer advantages such as straightforward spatial indexing, easy manipulation, and the ability to capture complex structures and details. However, they also come with challenges, such as memory requirements, computational complexity, and limitations in representing smooth surfaces or fine geometric features (Heeramaglore & Kolbe, 2022). Furthermore, voxel cells are counterparts to the raster file format used in the Ecological model.
This section describes with the help of an example building envelope the entire process of CAD integration using voxel cells, new algorithms for shading and soil depth analysis, as well as the Ecological model.

## 3.1 CAD integration through a 3D voxel model

When architects, urban planners and designers develop concepts for building envelopes, they use sketches, physical models, and conceptual digital 3D models in their creative design process. This process already involves the consideration of normative constraints (e.g., location, site, type of building, max height, number of built square metres/ cubic metres), as well as early stage decision making with respect to the structural design but also sustainability (e.g., materials used, orientation of the building for a max. exploitation of daylight, carbon footprint, etc.). By integrating ecological analysis in a 3D CAD system, feedback about the suitability of a building envelope design for species inhibition can be gained during the early design stage. Thus, ecological modelling becomes an intrinsic part of the design decision making process. Moreover, it is possible to assess the suitability of existing building envelopes within urban environments to support the needs of non-human species.
In the following example, a random building envelope design was used as reference to explain the integration process more clearly (Figure 5). Our example building has a building height of 10,5 m, and a total floor area of 1097,22 m$^2$. Approximately 62 % of the building is made of concrete and 38 % of the building consists of glass/steel. Further metadata related to the building design is shown in Table 1. Each floor has an external area with the potential to become a terrace/green space and on the building's roof top a layer of soil could be placed. Furthermore, our test building is located on a 50 x 50 metre site in central Spain.

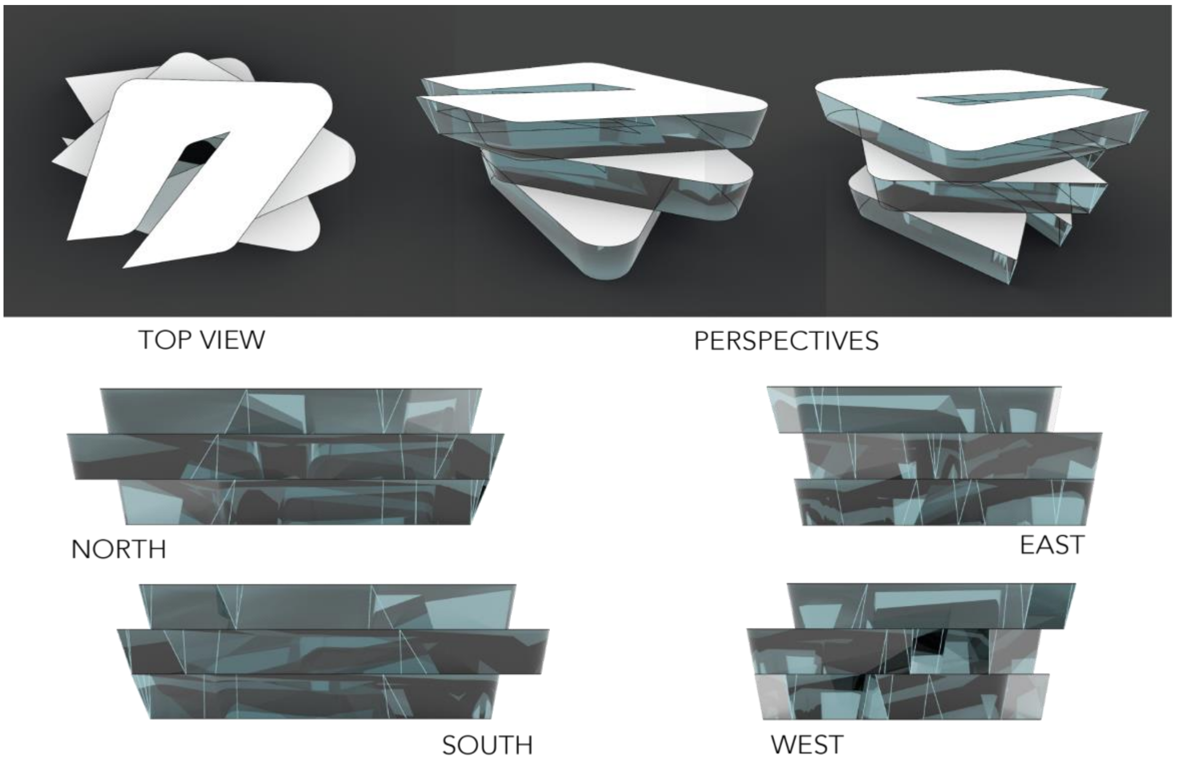


*Figure 5: Example building envelope geometry for CAD integration. It shows the top view, perspectives, and all elevations.*

*Table 1: Example of extraction of metadata related to the architectural design.*

| Architectural parameters | Values |
|---|---|
| **Geolocation** | Madrid, mid July, 15.00 pm |
| **Square metres site/ footprint** | 2500 m2/ 365,74 $m^2$ |
| **Building mass** | 3471,17 $m^3$ |
| **Building height** | 10, 5 $m^2$ |
| **Number of floors** | 3 floors |
| **Square metre floors** | 1097,22 $m^2$ |
| **Square metre façade/material glass** | 1207,70 $m^2$ |
| **Square metre material concrete** | 1984,73 $m^2$ |
| **Square metre roof** | 365,74 $m^2$ |

The building was modelled in Rhino using NURBS geometry. NURBS are mathematical representations of 3D geometry that can accurately describe any shape. Thus, NURBS models allow geometry related analysis at highest precision at any point of the 3D model. However, the GIS data available for environmental and ecological analysis purposes is constrained to a certain resolution (Raster resolution of 1 sqm). In order to be able to analyse 3D geometry models, the resolution of both systems must be identical. Thus, the system of the lowest resolution must be the reference. In our example, a custom developed geometry converter voxelises 3D geometry inputs to reduce the resolution to the one required as an input by the Ecological model. Table 2 shows the required inputs for such a process and Figure 6 shows the result.

*Table 2: Inputs required for the voxelisation process of the CAD model in Rhino/ Grasshopper.*

| Input parameters | Outputs |
|---|---|
| Envelope geometry and site | JSON: Voxel cells for the site and building envelope in logical order |
| Size of the voxel cells | Number of voxel cells |
| | Voxel cells air |

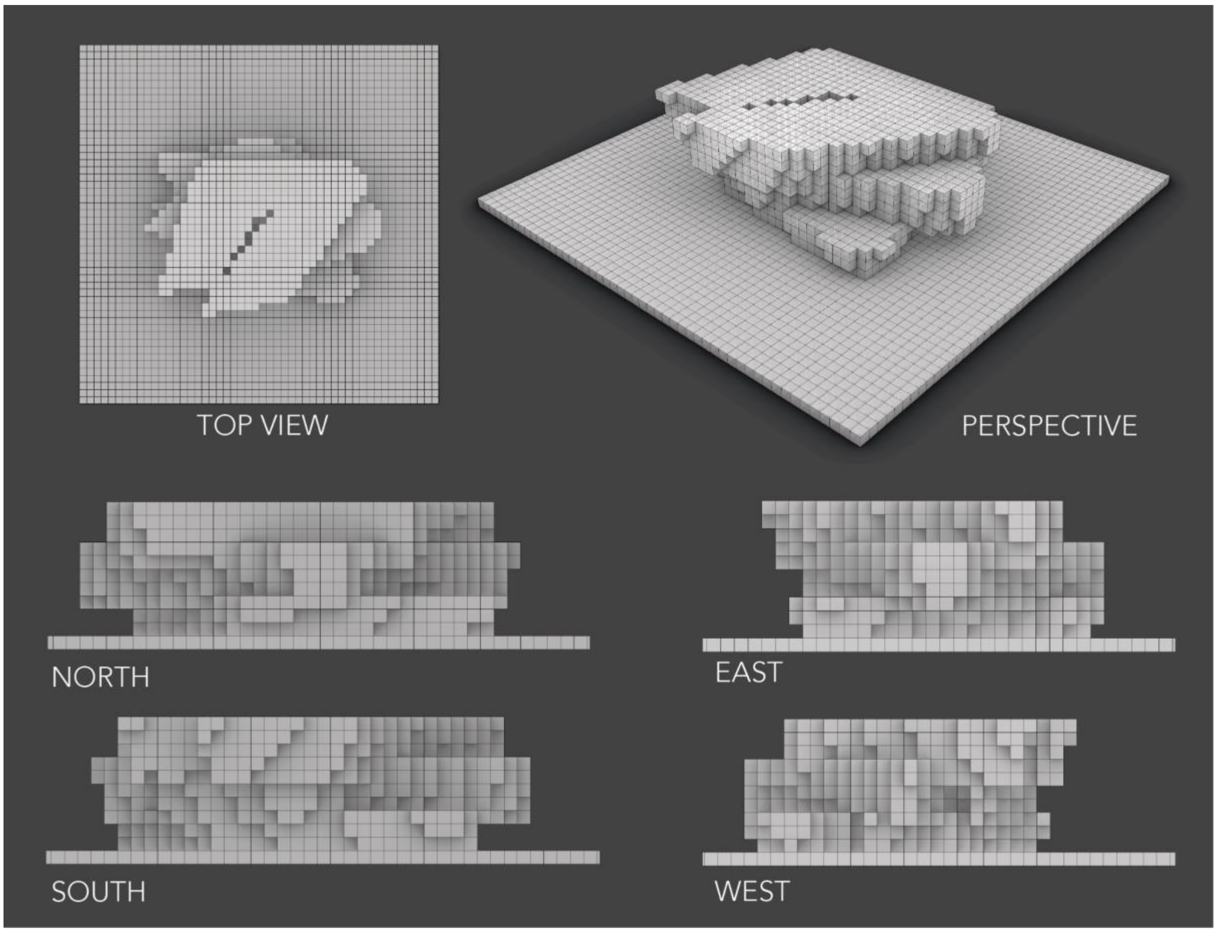


*Figure 6: Voxelisation of geometry inputs (example building envelope) with cell sizes of 1 cubic metre and their limitations capturing intricate details of the envelope geometry.*

The result is a voxel model of the building envelope and the site. Each voxel cell now represents mean geometry related values.

## 3.2 CAD integration shading analysis

In the next step, geometry related inputs required for the Ecological model need to be generated. These inputs are *"shading"* and *"soil depth"*. **Shading values** were computed using Ladybug tools in Grasshopper and mapped as metadata to each voxel cell. Table 2 shows the input parameters for our calculations. The colour gradient in Figure 7 provides an idea about the distribution of shading values (in percent) throughout the voxel model. Our results demonstrate that shading values are not only linked to the ability of plants to grow, but also to the intricate shape of the building and its geolocation. In the following step, shading values are associated with the corresponding voxel cell reference and are written into a JSON file which is then used as input for the ecological model. Furthermore, these values are used as data points for the Knowledge Base (Chapter 5).

*Table 2: Inputs and outputs of shading analysis of the CAD model in Rhino/ Grasshopper.*

| Input parameters | Outputs |
|---|---|
| Geolocation | JSON: Shading values for each voxel cell from 0.0 to 1.0 (JSON file) |
| Envelope geometry and site | Coloured mesh |
| Voxel cells | RGB values for each mesh cell |

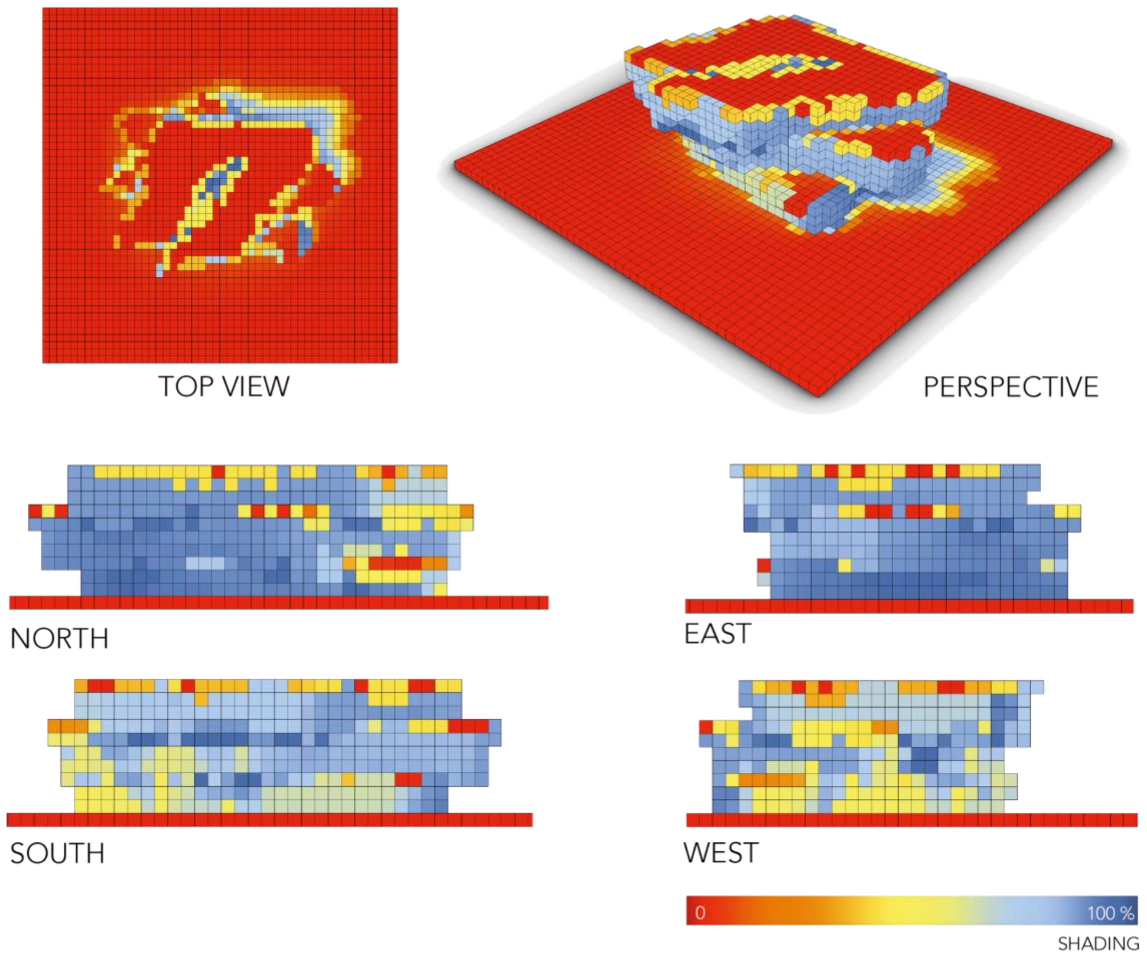


*Figure 7: The colour gradient displays voxel cells of a higher percentage of shading in blue and areas with no shadow in red.*

## 3.3 CAD integration of soil depth analysis

The process described in the previous section is repeated to compute **soil depth values** for each voxel cell. A maximum soil volume and different soil types are used as inputs to calculate the amount of soil that can be distributed on the building envelope. Soil depth values are not only relevant for plant growth, but are also determined by the form of the building envelope: First, areas that are covered by another floor slab and thus, do not receive natural irrigation through rainfall, have zero soil depth values. Second, areas that are too steep to accommodate soil without artificial planters or panels have zero soil depth values. Third, areas located on rooftops have different types of soil. Lastly, areas on edges of the building have lower soil depth values. Our results for soil depth values are visualised in Figure 8. Input parameters for our calculations can be seen in Table 3. Individual soil depth values associated with each voxel cell are written into a **JSON file**. Also this file is an input for the Ecological model and part of the KB.

*Table 3: Inputs and outputs of soil depth analysis of the CAD model in Rhino/ Grasshopper.*

| **Input parameters** | **Outputs** |
|---|---|
| Max. soil volume available for the project | JSON: Soil depth values for each voxel cell in mm (JSON file) |
| Soil type | Coloured mesh |
| Envelope geometry and site | RGB values for each mesh cell |
| Voxel cells | |

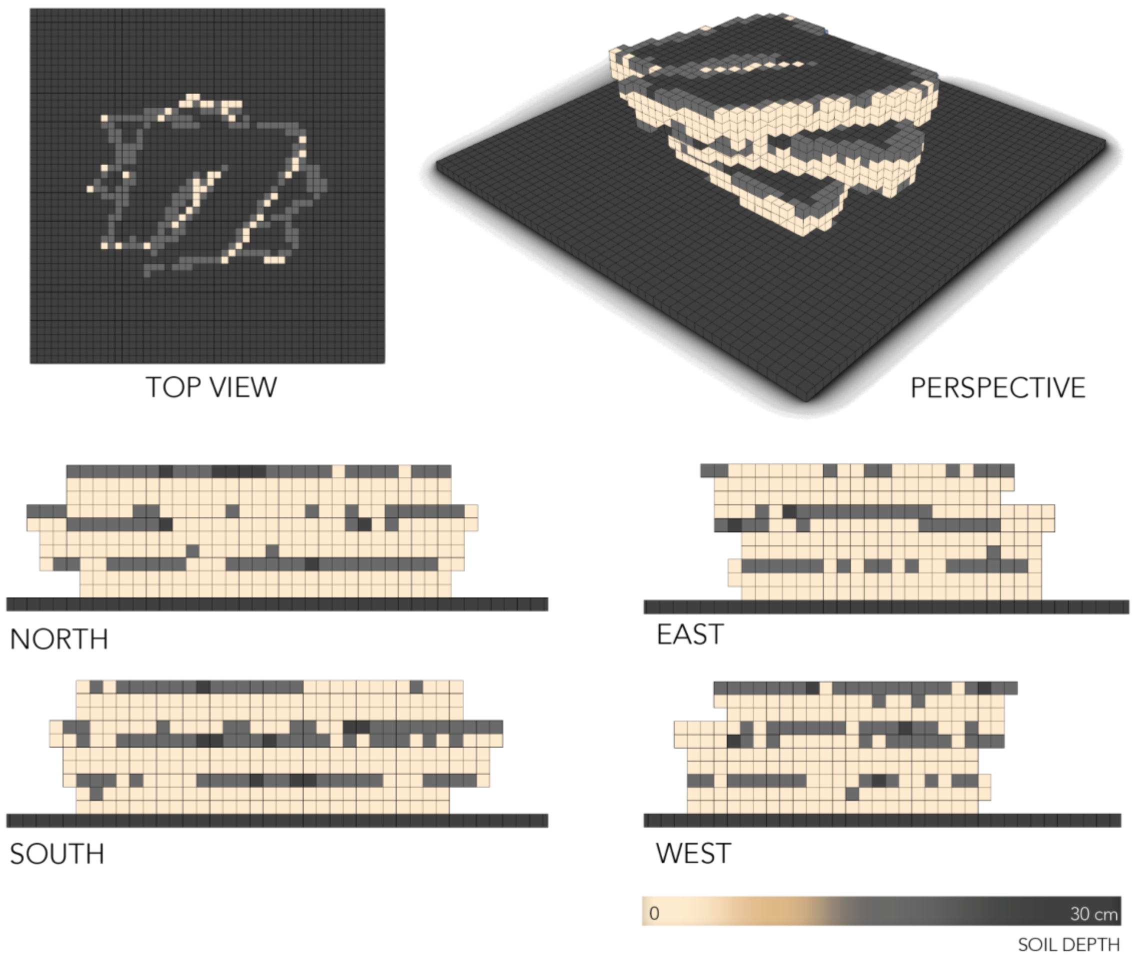


*Figure 7: The colour gradient displays soil depth values for each voxel cell. Areas of darker colour have the potential to hold thicker layers of soil, and thus, can become multi-species habitats. Beige areas are not prone to hold soil without artificial planters or special panels.*

### 3.4 CAD integration ecological analysis and meta data of PFGs

The previous steps described how meaningful input values were generated for the Ecological model. The compiled JSON files were exactly formatted in a way the Ecological model can process them. Only with these generated inputs, community dynamics between soil, plants, and animals can be modelled for this specific shape (Figure 8). In our test, we used only the PFG submodel, which was easier to adapt to 2.5D data. Outputs are plant biomass per voxel cell, and PFGs per voxel cell as a JSON file. This information can then be written in Rhinoceros/ Grasshopper to preview the analysis results using the colour gradient method for plant biomass, and metadata method to display PFGs/ voxel cells in the CAD environment. To bring this information back closes the circle for CAD integration. Architects, urban planners and designers can use ecological analysis results to better understand how likely their design is to plant species abundance. Table 4 shows the required inputs and Figures 8 shows a colour gradient representing biomass and Figure 9 individual PFGs as metadata.

*Table 4: Inputs and outputs of ecological analysis of the CAD model in Rhino/ Grasshopper.*

| Input parameters | Outputs |
|---|---|
| JSON files “Shading” and “Soil depth” | Biomass plants |
| Voxel cells | Name and number of PFGs for each voxel cell (metadata) |
| | Coloured mesh |
| | RGB values for each mesh cell |

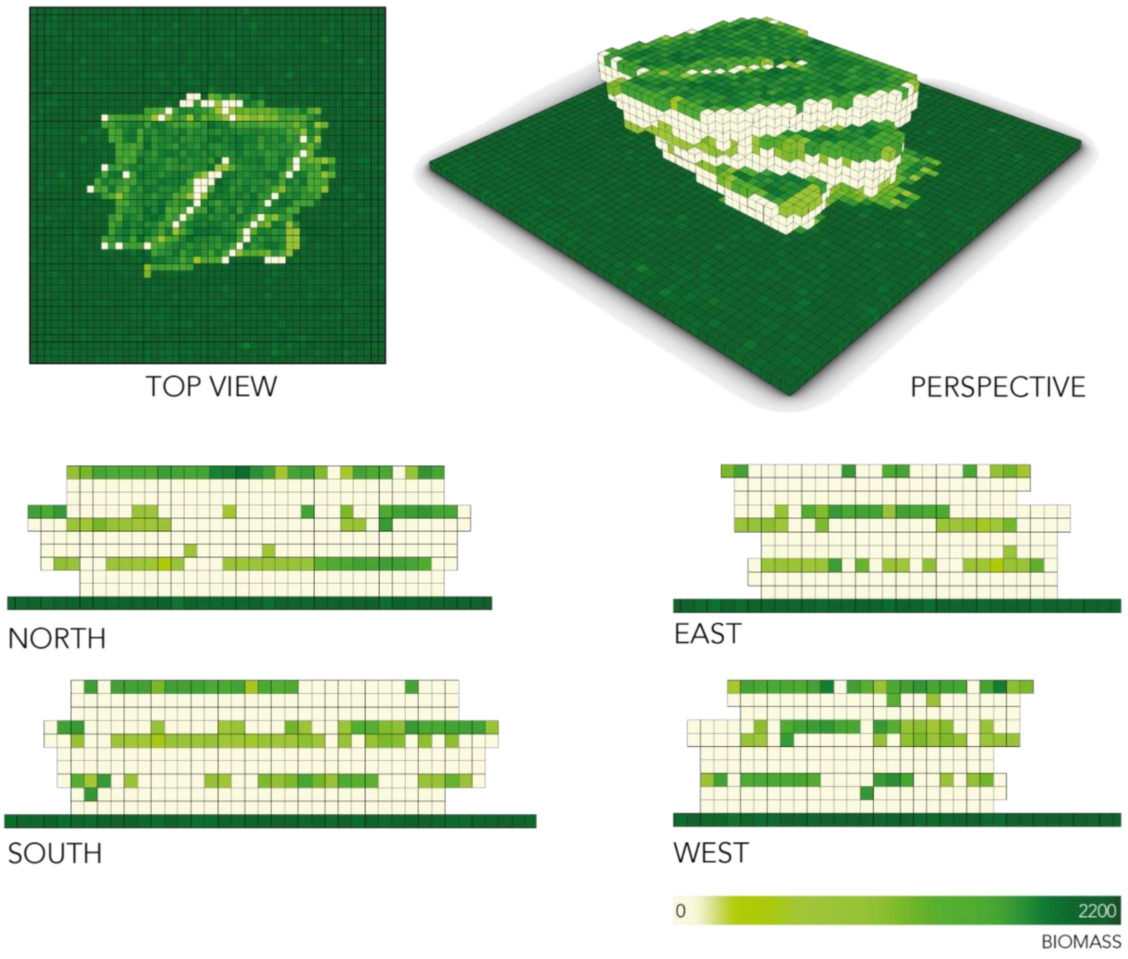


*Figure 8: Visual output of the ecological analysis in Rhinoceros/ Grasshopper. Voxel cells of dark green colour are more likely to host a larger number of species and PFGs. Cells of a beige colour are not considered as habitats for plants.*

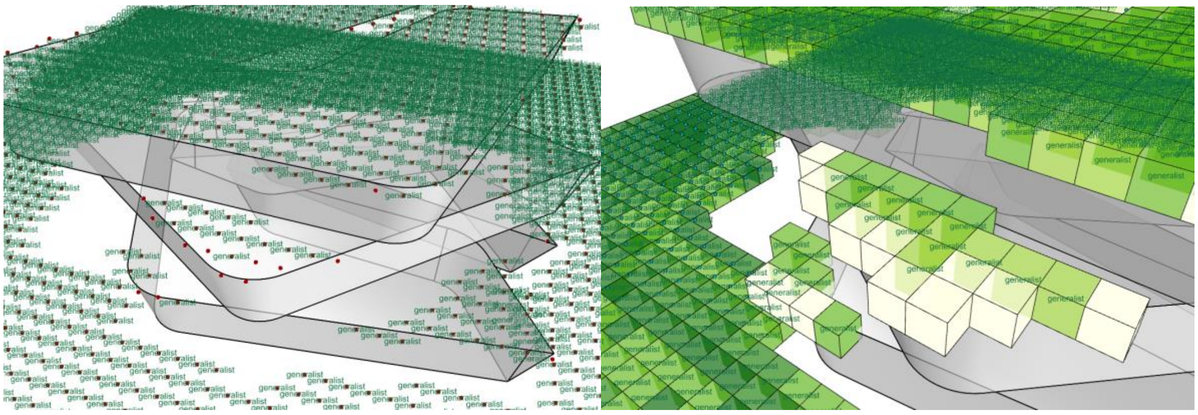


*Figure 9: Names of PFGs mapped as metadata to the 3D model in a CAD system. Screenshots (left and right) show different visualisation options for metadata display in CAD.*

Besides spatial dynamics, another important step was the inclusion of temporal dynamics linked to reproduction and growth of plant functional groups These dynamics are computed in the Ecological model and for each time step a JSON file can be generated. The visualisation of the data in CAD is crucial for various reasons: First, to understand if the generated outputs are correct (context wise and order). Second, to visualise the impact to certain spatial configurations over time. Third, to have a visual way of presenting this data to stakeholders and decision makers. Table 5 shows the abundance of PFGs over 9 years. A graphical representation of these numbers is shown in Figure 10.

*Table 5: PFG abundance over time linked one voxel cell.*

| | **Generalist** | **Ground_17** | **Ground_7** | **Ground_No** | **Total** | **r_17** | **r_7** | **r_no** |
|---|---|---|---|---|---|---|---|---|
| **0** | 288 | 132 | 386 | 282 | 1088 | 52 | 58 | 124 |
| **1** | 461 | 66 | 485 | 402 | 1414 | 26 | 29 | 62 |
| **2** | 776 | 0 | 691 | 655 | 2122 | 0 | 0 | 0 |
| **3** | 1267 | n/a | 1097 | 997 | 3361 | n/a | n/a | n/a |
| **4** | 1854 | n/a | 1557 | 1449 | 4860 | n/a | n/a | n/a |
| **5** | 1643 | n/a | 2152 | 2025 | 5820 | n/a | n/a | n/a |
| **6** | 0 | n/a | 2090 | 2105 | 4195 | n/a | n/a | n/a |
| **7** | 2536 | n/a | 834 | 878 | 4248 | n/a | n/a | n/a |
| **8** | 2536 | n/a | 953 | 1011 | 4500 | n/a | n/a | n/a |

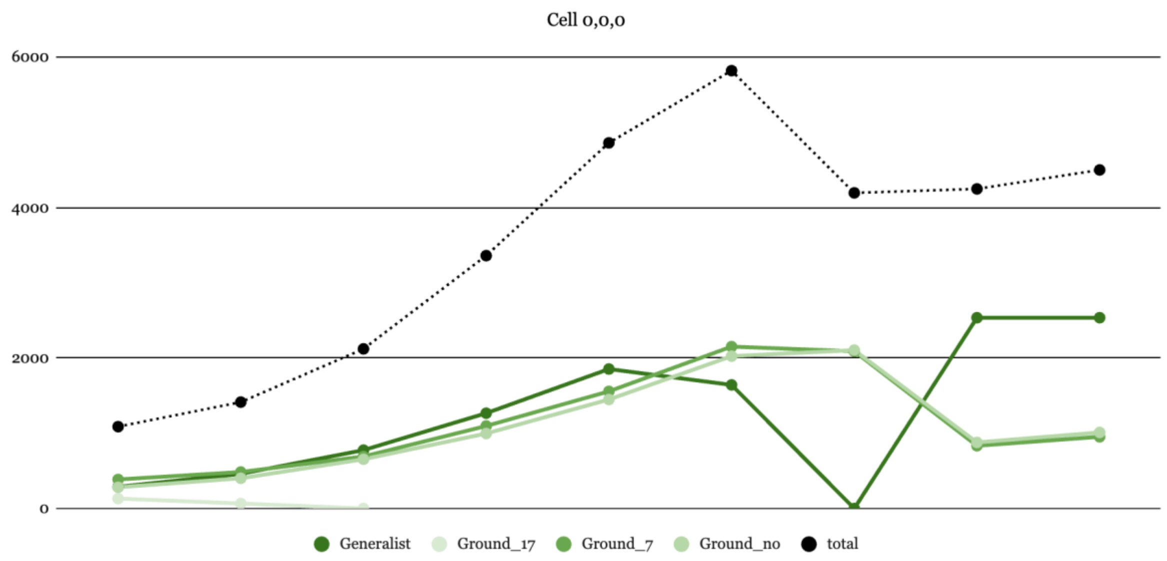


*Figure 10: Temporal PFG dynamics over a time of 9 years for one voxel cell.*

Our results (for voxel cell (0,0,0) ) show that the first PFG dies out because soil depth is too small with only 1 cm. PFGs “Ground_7” and “Ground_no” only show differences when it comes to soil depth, but generally have the same preconditions. PFG “Generalist” (dark green) becomes shaded by the other PFGs and dies out in time step 6/7. The black line demonstrates the total biomass. The total biomass of all PFG is one important parameter to understand the effects of vegetation on the building.

Figures 11 and 12 visualise the spatio-temporal PFG dynamics linked to our example building design over a time of 12 years.

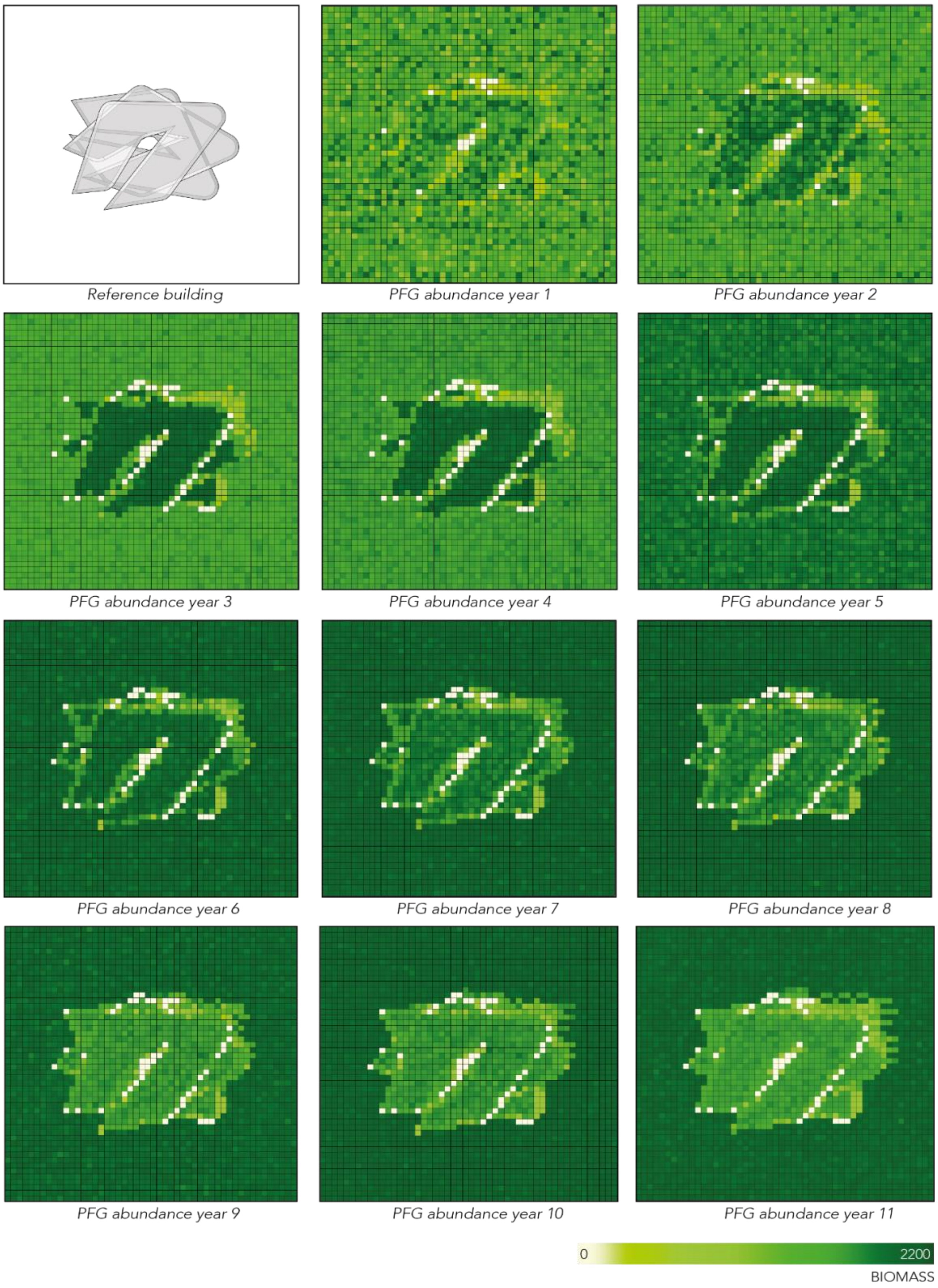


*Figure 11: Top view of spatio-temporal dynamics of PFG total biomass in Rhino/ Grasshopper. Areas of dark green are more likely to promote a larger number of biomass than lighter green areas. Beige areas are considered in our modelling approach as uninhabitable areas for PFGs.*

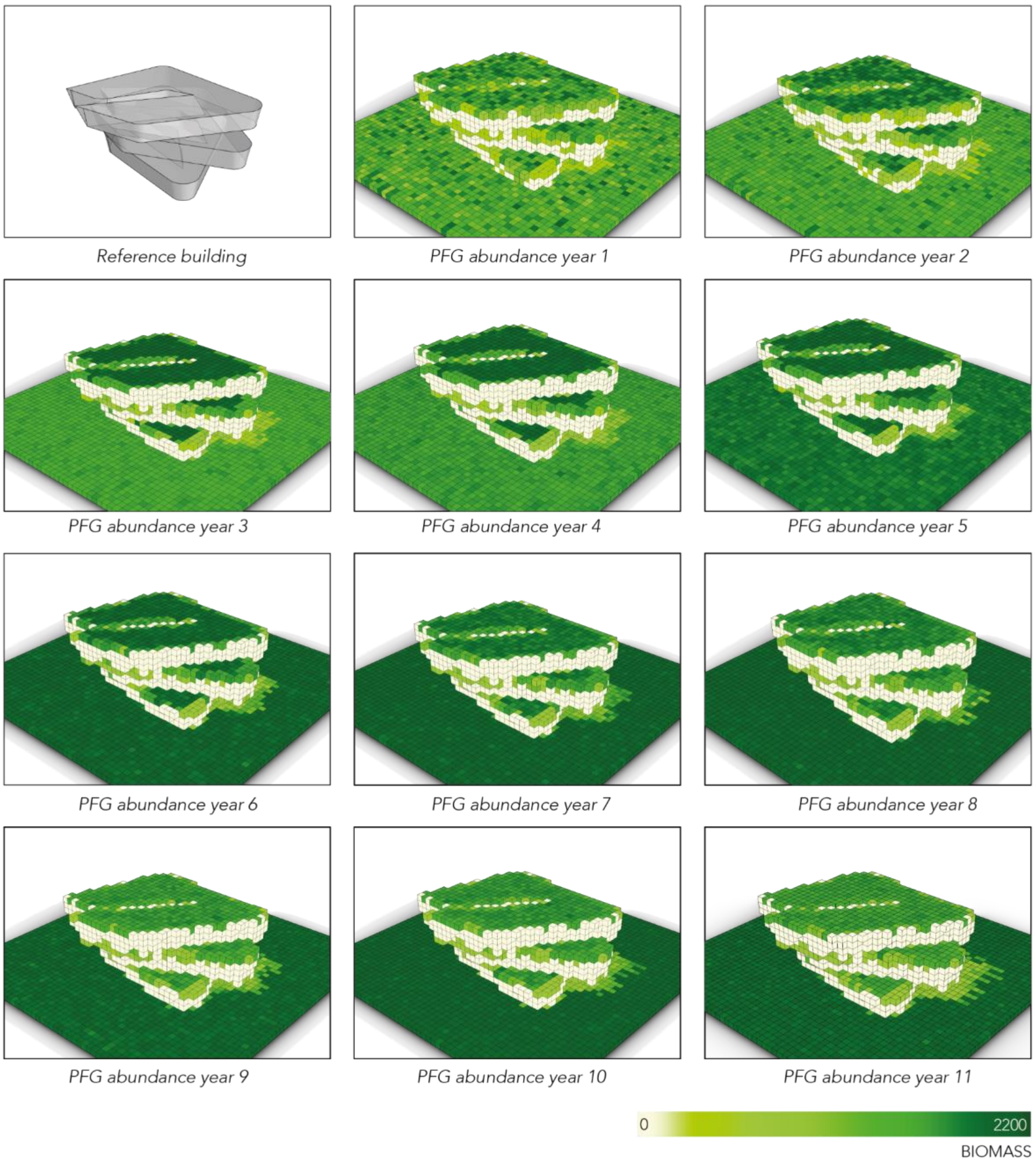


*Figure 12: Perspective of spatio-temporal dynamics of PFG total biomass in Rhino/Grasshopper. Areas of dark green are more likely to promote a larger number of biomass than lighter green areas. Beige areas are considered in our modelling approach as uninhabitable areas for PFGs.*

In the next step, we used analysis results to project this information as a rendering mesh (ecological analysis) to the initial NURBS CAD model (Figure 13).

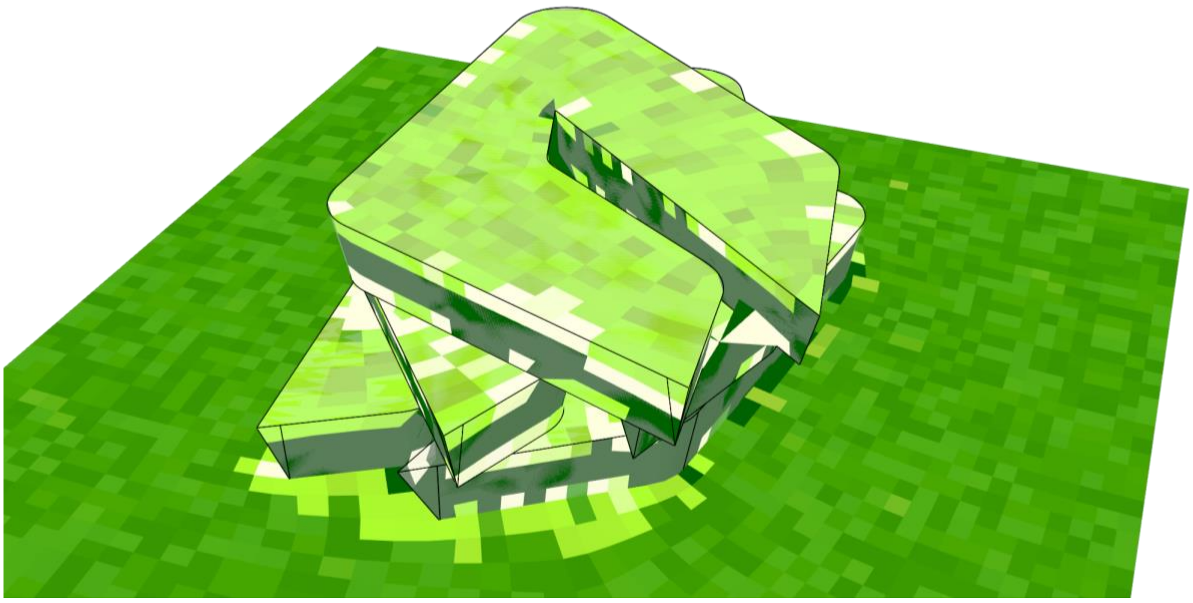


*Figure 13: Generated rendering mesh that displays ecological analysis results on CAD model (NURBS geometry in Rhino).*

To test and validate the developed integration approach, we used a more intricate building envelope design as an input to conduct the same analysis steps as in our previous tests. The 3D model of the building envelope was inspired by an AI generated image in MidJourney, a generative artificial intelligence program developed by the independent research lab Midjourney, Inc (Midjourney 2022) (Figure 14).

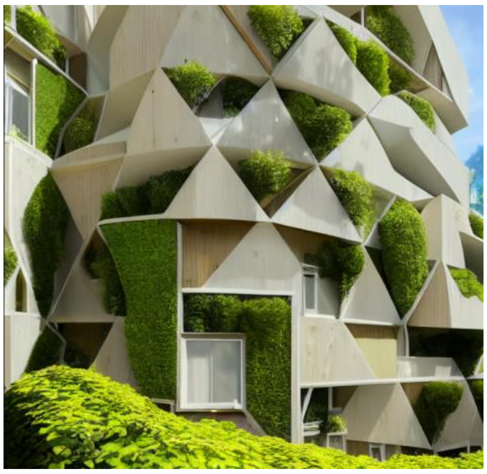


*Figure 14: Inspiration for a green building envelope, image generated through Mid Journey PROMPT: "modular building, mid rise apartment, sustainable, green space balcony, rooftop garden, wall climbing ivy and plants, parametric design, net zero, exoskeleton structure frame, wood and precast white finish smooth wall, high performance facade window and glazing, high resolution, biomimicry, organic and dynamic space, healthy, 2K, lower camera angle, Vray render, high tech, futuristic, located near urban park" (MidJourney Prompt 2023).*

Our model was developed in Rhino and it differs in height and facade heterogeneity (material, shape, ratio between more and less exposed construction elements) from our first test building envelope (Figure 15).

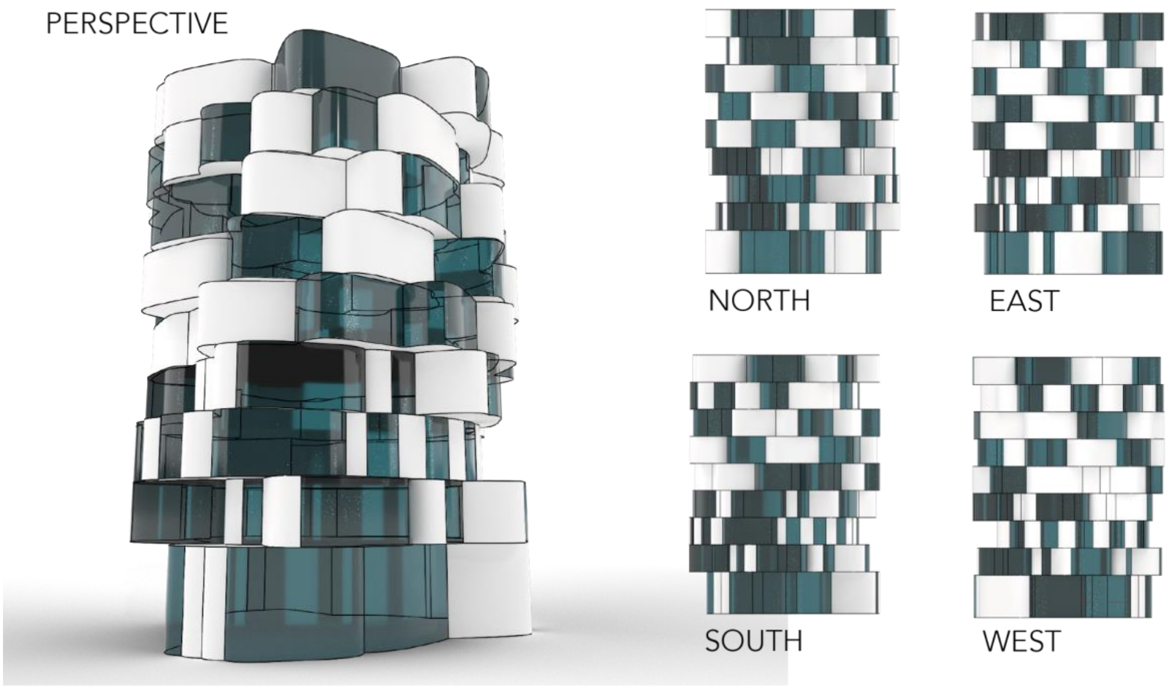


*Figure 15: 3D CAD model (Rhino) of a reference building envelope with a more heterogeneous facade design than the first example.*

Analysis results and inputs for the Ecological model, shading and soil depth, were calculated and displayed in Rhino/ Grasshopper (Figure 16). Analysis grid shows non-shaded areas on balconies and the rooftop. The north face of the building and areas below more exposed facade elements are shaded. Soil depth values seen on larger flat areas are greater than the flat areas on balconies. The majority of the vertical areas of the facade & glass panels are not likely to hold soil without additional artificial planters.

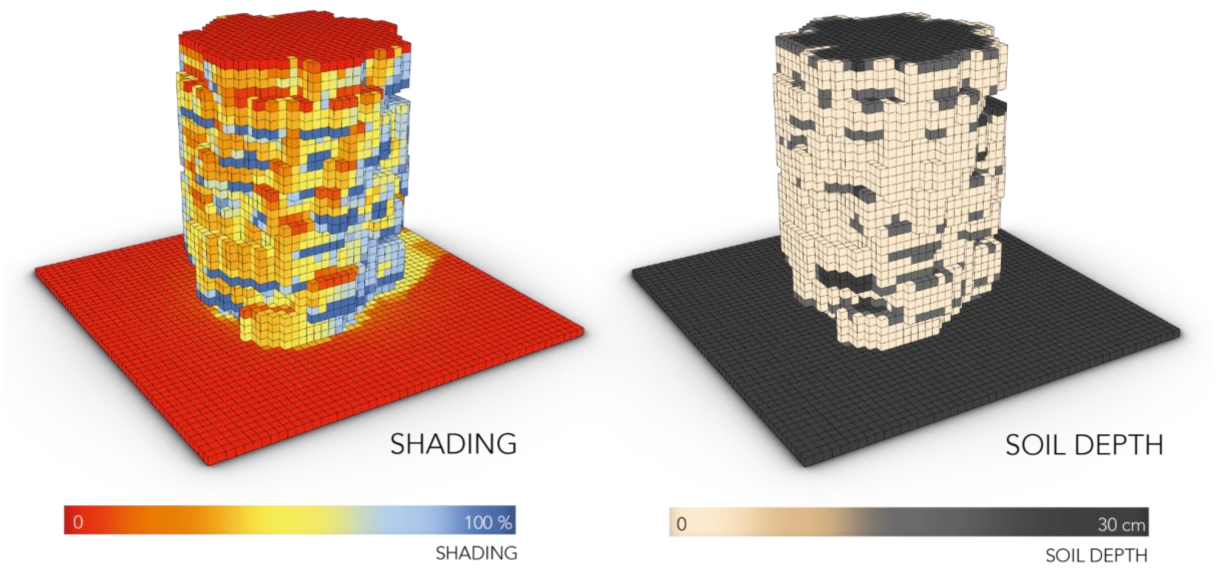


*Figure 16: Shading and soil depth analysis results in Rhino/ Grasshopper.*

In the next step, spatio-temporal dynamics for PFG biomass values and PFGs were computed for voxel cells using shading and soil depth values as an input (Figure 17). Our first results confirm the ones in our initial test, that areas of a larger soil depth promote more biomass, and that over time green areas located on the building envelope promote less biomass than areas on the plot.

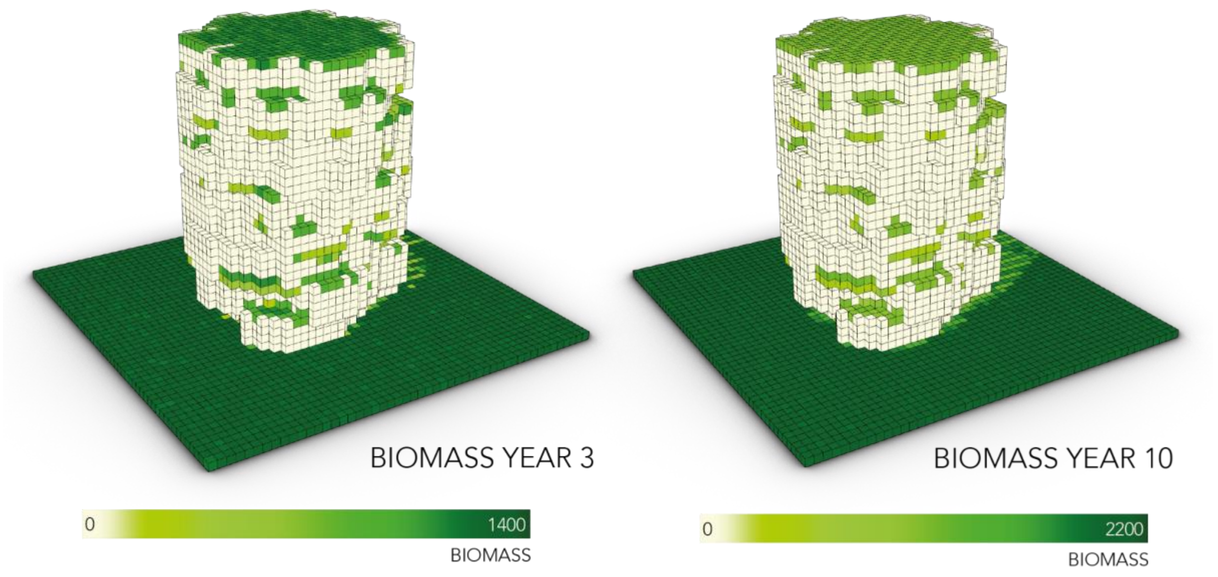


*Figure 17: Ecological model analysis result shows PFG dynamics after 3 and 10 years.*

Also in our second example, PFG metadata reveals more precise information about the classification of plants (taxonomy). This understanding is crucial to first, have the opportunity to 3D model these plants in CAD, and second, to apply form generation and decision-making algorithms in a data driven design process (Figure 18).

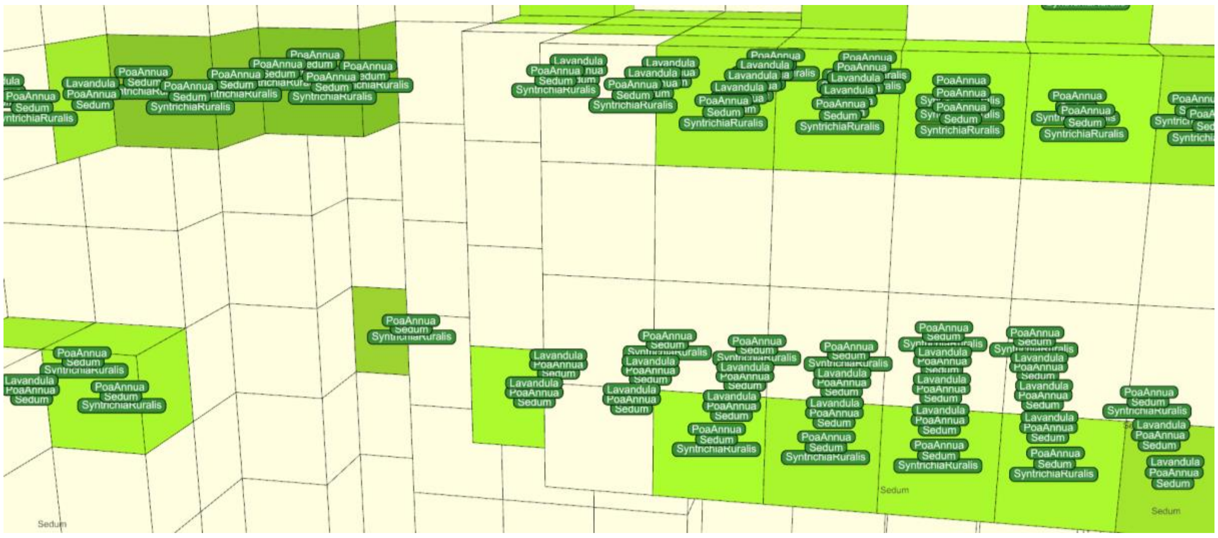


*Figure 18: Visualisation of PFG related metadata (plant species) for each voxel cell of 1 cubic metre.*

The second example shows that our modelling approach is already applicable to different building envelope designs. Analysis results linked to shading, soil depth and PFG biomass, and PFG groups are directly linked to the form of the building envelope, which proves that for the start the correct parameters were selected. Even if all three analysis methods are simplified modelling approaches and do not reflect the complexity that is involved for analysing green building envelopes, the approach starts to show that the understanding of the relationships between shading, soil depth, species abundance and the form of the building is key. This aspect will be further addressed in the Knowledge Generation Framework in the following Chapter 4.

## 3.5 Conclusions

In summary, our results show that through the integration of the ecological model in a 3D CAD system, ecological modelling becomes an **intrinsic part** of the form finding process of building envelopes in our cities. Furthermore, as a parametric system, the approach works with multiple geometry inputs and surrounding buildings. Lastly, the implemented cloud architecture with Rhino.Compute was essential to enable and guarantee an intuitive computational process for users of our system. At present, the ecological model is integrated in a simple way, for example, no effects of windows on bird strikes can be investigated. However, the set-up of the computational platform allows the integration of more and more realistic plant and animal models.

For the submission of the demonstrator in M38, we will work on how to make this process more efficient by introducing parallel computing resources and through the optimisation of the developed algorithms. Architects will also need a validated ecological model to apply the developed approach in design.

At a higher level, our CAD integration approach could already contribute to answering fundamental questions concerning the behaviour of dynamic systems and the emergence of form when designing for nature and humans.

## 4. KNOWLEDGE GENERATION FRAMEWORK PROTOTYPE (KGF)

In the literature, there is little systematic information on how the architectural form affects Ecological Communities (Weisser et al. 2023). The existing knowledge linking architecture to ecology only exists in fragmented and specific ways. For instance, we might know from the literature review the needs in terms of food and shelter resources of a given bird species. Such knowledge can be used by architects to create artificial nests of the right shape and height, to attract particular bird species. However, an artificial nest may only enable an individual of this species to live on the building if a number of other conditions are met, i.e., if the environment provides the other factors necessary for the species to complete its life cycle, such as access to sufficient food resources or mates (Weisser & Hauck 2017). This species-specific knowledge is not necessarily straightforward to translate at the FG level. Understanding whether an *ecolope* provides sufficient resources for particular FGs is a requirement for design. Similarly, we also need to understand how the architectural form influences biodiversity variables at a higher organisational level of the *ecolope* ecosystem, i.e., at the community level (e.g., diversity of FGs, presence of predators). Ideally, such information is provided intrinsically in the design process by means of an ontology that encapsulates the relevant relationships. However, as very little is known about the relationship between architecture and ecology, such knowledge needs to be generated. The Knowledge Generation Framework (KGF) was designed to address this knowledge gap. By simulating how architecture (e.g., form, building mass and height, material, facade heterogeneity/inclination) drives the environmental conditions on the *ecolope* (e.g., shading, soil depth), and hereby the ecology (e.g., identity and abundance of FGs), novel correlations between different parameters can be generated. Besides asking questions such as how architectural form promotes species richness, biomass, and ecological performance in our cities, a variety of questions concerning the effect of architecture on ecology are and will be addressed by the KGF, for instance:

- How strong is the influence of shade vs. soil type and depth on the distribution of plants on the building?
- How does heterogeneity in soil type or soil depth affect plant community development?
- What are the FGs suitable for the chosen building envelope and what is the suggested initial distribution of the FGs on the building surfaces?

The KGF is designed to systematically and computationally find relationships between architectural, environmental and ecological parameters. Through a series of computational experiments with changing geometry inputs, datasets consisting of data representing certain parameter values are generated. These datasets are stored in the KB. In the following step, KB datasets are optimised using algorithms from ML (WP3) and machine reasoning (WP5) to extract the missing knowledge about interrelations for design decision support (for WP6 (Figure 19).

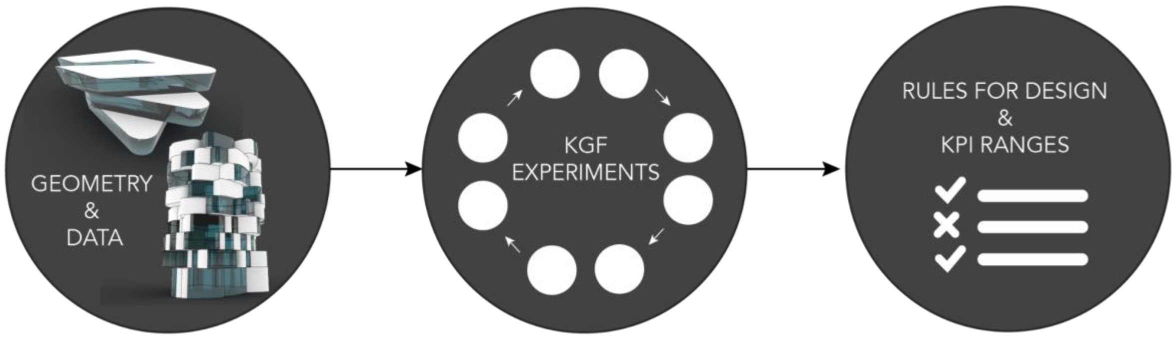


*Figure 19: Architectural, environmental and ecological data of changing building geometry inputs are generated and stored in a database (KB). In the next step, ML algorithms extract rules and KPI ranges for design decision support.*

In this section, we discuss the latest update of the KGF which concerns changes made to the Ecological model (Section 4.1), KGF Grasshopper functions (Section 4.2), KGF parameters (Section 4.3), updates about KGF experiments (Section 4.4), and the Knowledge base (KB) including ML-based optimisation (Section 4.5).

## 4.1 Reduced Ecological Model for KGF

Because model development is asynchronous, version 0.2 of the KGF concentrates on the plant effects only, disregarding changes in soil over time, and also ignoring animals. This allows setting up the framework and workflows first, before attempting to retrieve meaningful correlations. During development it became clear that various amendments to the connection between architectural and ecological models are necessary, particularly in regard to reading files. These amendments are developed in a separate branch of the Ecological model that breaks compatibility between plant and animal model. Furthermore, for version 0.2, various bugs were resolved and minor issues corrected while building the KGF branch. The following amendments are being made for the KGF: 3D version of Ecological model (Section 4.1.1), and changes to JSON input and output files (Section 4.1.2 and 4.2.3).

### 4.1.1 Addition of Z-dimension

In the base version (version 0.1) a square 2D input map was required, and all coordinate keys within the square needed to be present (no holes allowed). The z-dimension of the input was silently ignored. In the KGF version 0.2, a 3-Dimensional input is required. The user also needs to specify the spatial extent in x, y and z direction. The model checks whether all provided keys fall within the spatial extent, but no longer checks for completeness of keys. This allows providing a map with 3D "holes" or a non-rectangular (potentially ragged) map. The intention with this setup is threefold: (1) One can provide a full 3D map of the terrain, but leave the interior of buildings out, so they do not get colonised by the plants; (2) the model will simulate plant growth on all cells, so it is able to model e.g. balconies or terraces (it is still only correct on horizontal surfaces, though); (3) one does not need to provide the cells that only contain air or are buried (a 50 x 50 x 10 site would have 25000 cells; but the surface may only cover 2500 cells).

### 4.1.2 Changes to JSON input files

The changes to the coordinates caused a few changes to the way the JSON files are read in.

1. Because the Ecological model does not check the keys for completeness anymore, the shading and soil depth maps might accidentally contain different holes but have the same number of cells. The Ecological model is not guaranteed to find these problems or it may issue a warning that is not informative. Special care must be taken by Grasshopper (Section 4.2) to provide correct maps.
3. In previous versions of the Ecological model, a 50 x 50 map with random soil classes was provided. This class was used with any input coming from Grasshopper. Because the 3D model of the building on a site was flattened down to 50 x 50 (the Z-dimension was set to zero), the keys of the soil class map fitted with the other inputs. Now that the Z-dimension can be any other arbitrary value, this approach does not work anymore. Thus, a separate script was written that uses the soil class definitions and a Grasshopper input file (shading or soil depth) to create a map with randomly distributed soil classes.

### 4.1.3 Changes to JSON output files

The outputs of the Ecological model for a simple test case were approximately 90 MB for the plant model, and further 90 MB for the animal model. The JSON files contained the information of 107 PFGs, at 50 x 50 cells, and for 10 time steps (approx 250.000 data points). The file size increases linearly with the number of cells, i.e., to the third power with spatial extent. The output was hence reduced for the KGF. The user can now specify for which individual time steps an output is required. For each required time step, the model will create two JSON files: One file outputs the summed biomass of all PFGs per cell, and the other file outputs the identities of the PFGs whose biomass is larger than 0.0. A third file concatenates the detailed information as before, but only for the time steps that the user selects.

## 4.2 Updated KGF Grasshopper functions

To enable the KGF, custom Grasshopper components were developed by MCNEEL. These components include algorithms for data conversion between the Ecological model and the 3D CAD model, a JSON reader and writer adapted to the in- and output requirements by the Ecological model, as well as display components for PFG biomass and metadata in CAD, and algorithms for shading, solid depth and facade inclination analysis. The majority of them now also work for Rhino 8, to be released soon.

### 4.2.1 Changes to output files for Ecological model

Since KGF version 0.1, the following updates were made for KGF version 0.2:

1. The voxelisation process of 3D models returned an error with increasing complexity of the building envelope geometry. This issue is fixed for version 0.2 by rewriting the algorithm. Furthermore, the voxel drape algorithm that only provides the voxel cells for a 2D 50 x 50 raster input in KGF version 0.1 is now substituted by a full voxel model.
2. Shading values used for KGF version 0.1 were calculated from indirect irradiance. However, these values were not precise enough and the algorithm for shading analysis was adjusted for the Ecological model input.
3. The soil depth model in KGF version 0.2 now roughly understands different soil types and can distribute soil depending on the type of soil and its location. This process required segmentation algorithms. The voxel model can be segmented into (ecologically) meaningful areas (e.g., rooftop, terraces, balconies, facade, site) and soil types applied to these areas. Inputs for soil depth analysis are max soil volume and type for distribution.
4. The "JSON writer" now writes JSON files in the format that the Ecological model can directly input.

### 4.2.2 Changes to display Grasshopper components of PFG biomass and metadata

Ecological outputs can only be evaluated once visualised and displayed in CAD on the original 3D model input. Thus, the following changes were made for KGF version 0.2:

1. Existing preview component was adjusted to read 3D inputs in the JSON file format.
2. Two additional components were added:
    a. To display PFG metadata for each voxel cell in CAD,
    b. and to display the analysis grid for PFG biomass as a rendering mesh on the original 3D model input.

## 4.3 KGF parameters

KGF version 0.2 supports six parameters in the categories: architecture, environment and ecology. They are further explained in Table 6.

*Table 6: Cross-disciplinary parameters that are considered in the current KGF version 0.2.*

| **Category** | **Parameters** | **Description** |
|---|---|---|
| Inclination | Architecture | Facade inclination of building envelope/ voxel cell |
| Soil type | Architecture | Segmentation of the 3D model into areas of terraces, balconies, roof tops, outdoor areas for vegetable patches, etc./ voxel cell |
| Shading | Environment | Shading values directly related to the geometry of the building/ voxel cell |
| Soil depth | Environment | Soil depth values directly related to the geometry of the building/ voxel cell |
| Biomass | Ecology | Biomass values are the sum of PFG biomass/ voxel cell |
| PFGs | Ecology | PFG names and number/ voxel cell |

To calculate each parameter at a resolution of 1 voxel cell (1 cubic metre), different computational tools were developed (Chapter 3). One of the tools developed is a tool to compute facade inclination for intricate building geometries (Figure 20). Inputs are the NURBS/ mesh and voxel model (Table 7).

*Table 7: Inputs and outputs of inclination analysis of the CAD model in Rhino/ Grasshopper.*

| **Input parameters** | **Outputs** |
|---|---|
| Envelope geometry and site | JSON: Inclination values for each voxel cell in degrees. |
| Voxel cells | Coloured mesh |

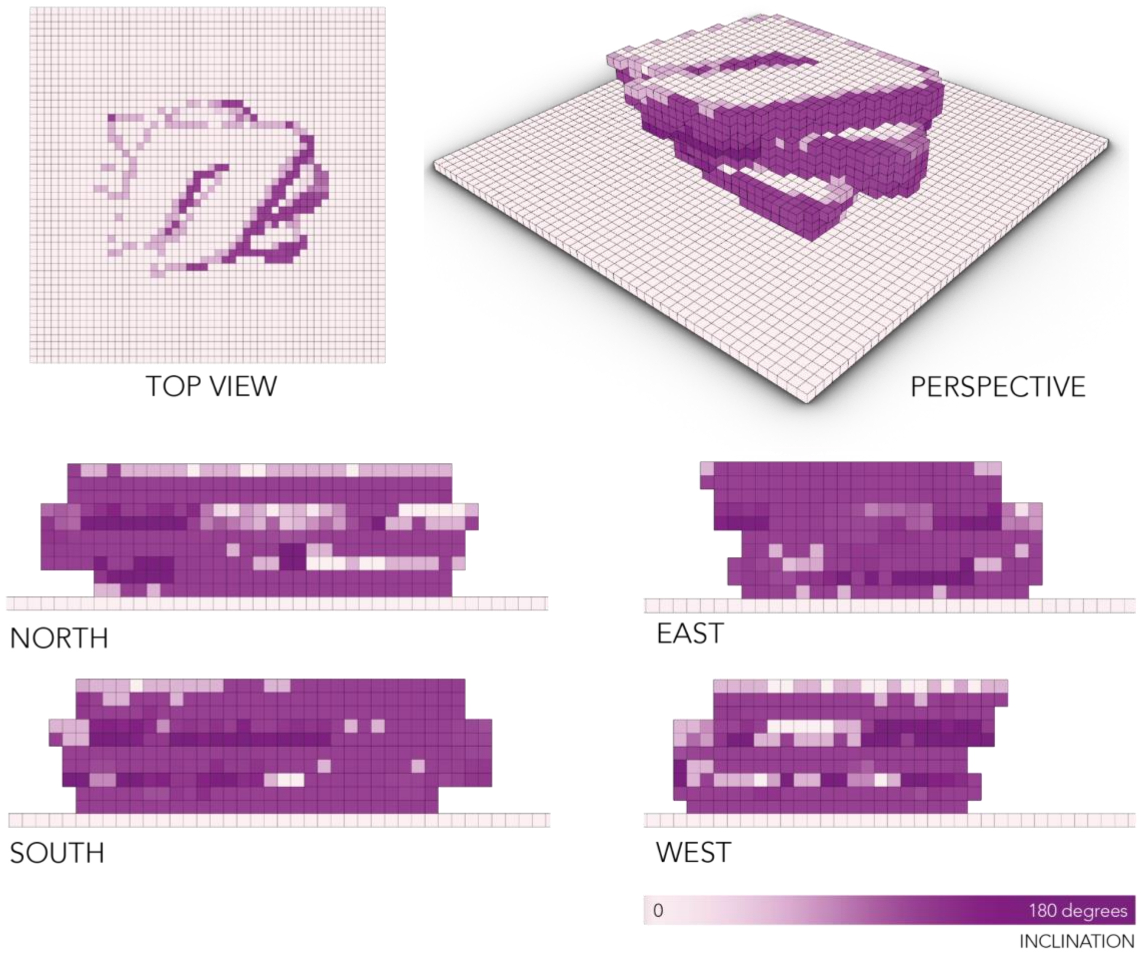


*Figure 20: Inclination values in degree. The darker areas in the colour gradient indicate overhanging parts of the facade, while lighter areas represent planar areas.*

Figure 21 illustrates KGF parameters for both example buildings. Already by a visual representation of the computed data, relationships between shading and biomass, as well as inclination and soil depth, or soil depth and biomass can be supposed. To more precisely understand relationships and thresholds between more than two parameters, the next sections introduce KGF experiments and our ML approach for the KB.

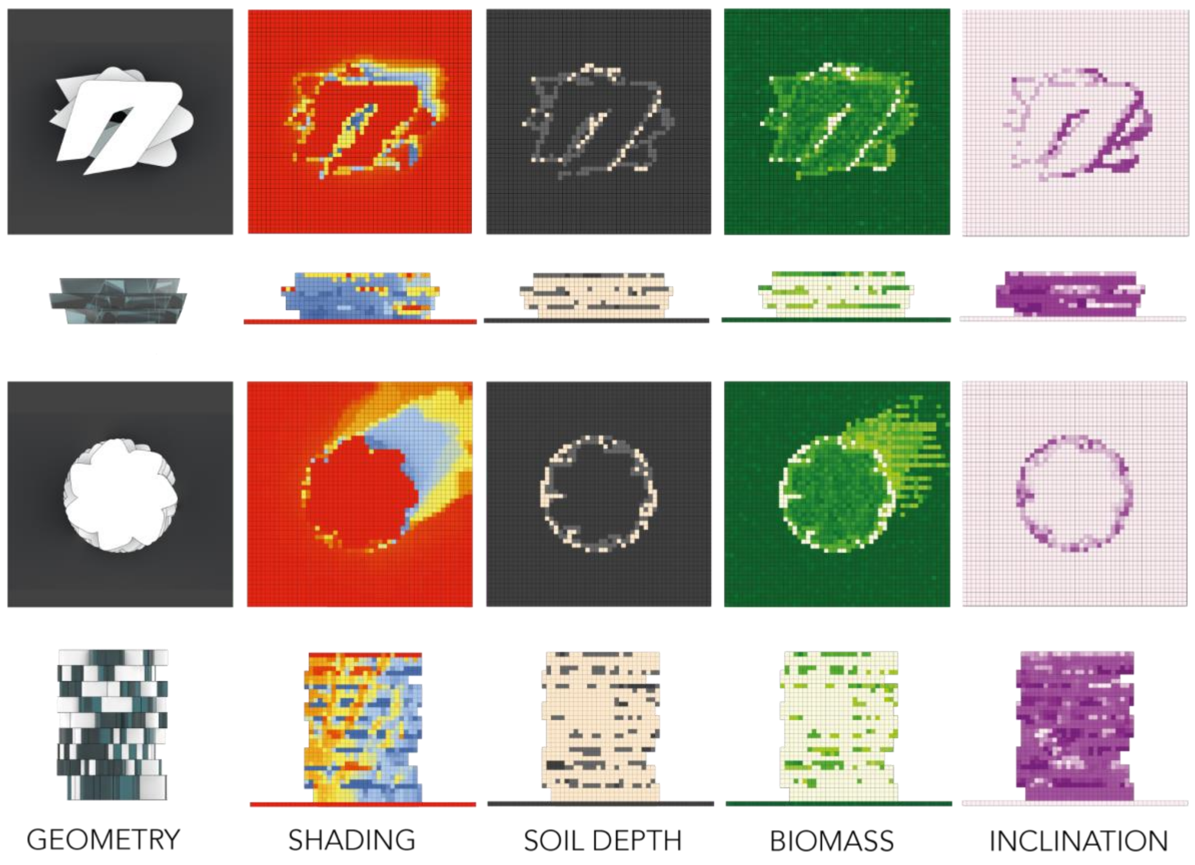


*Figure 21: Visual observations of relationships between different parameters.*

## 4.4 KGF experiments

Since the report of Year 2 (D1.5, M24), we addressed a number of obstacles to provide adequate conditions for an efficient computation of the KGF experiments: First, the computation time of the Ecological model (plants only at the moment) was drastically reduced from approximately 30 min to 5 min by TUM /SAAD (Section 4.1). Second, the ecological plant model can now read 3D data inputs instead of only 2D data inputs. Thus, it is now possible to compute and visualise PFGs for the entire building envelope including vertical parts (Section 4.1). Third, algorithms related to interoperability between 2D raster and 3D data, analysis of 3D geometry input, and display of results in CAD were entirely rewritten by MCNEEL (Section 4.2). Furthermore, additional converters to guarantee JSON file exchange were developed on both ends by MCNEEL and TUM/SAAD (Section 4.1 and 4.2). Lastly, the Knowledge Base was developed as a storage of JSON files. Algorithms from ML optimise the KB to extract relevant relationships (Section 4.5).

The Knowledge Generation Framework is designed to run the Computational workflow (including the Ecological model) systematically on hundreds of generic shapes, in order to derive correlations between form, environment and ecological function. The KGF is also the first attempt to run all integrated components without user intervention.

The current version of the KGF (Version 0.2) can generate data in a resolution of 1 cubic metre related to: (1) architectural form (facade inclination, material); (2) environment (shading, soil depth, soil type); and (3) PFG biomass and PFG abundance. In its current state, the developed system is capable of analysing 3D models of existing building envelopes and new building designs.

Figure 22 illustrates the iterative computational process of KGF experiments. First, building design is used as an input. Then, the NURBS/ mesh geometry input is converted into data points using a voxelisation process. In the next step, environmental metadata is computed for shading and soil depth, architectural metadata for inclination, and ecological metadata for biomass and PFGs during different timesteps. These values are then stored as a JSON file (KB). To validate if the computed ecological data is correctly written into the JSON file, a preview component displays ecological analysis outputs in CAD. KGF experiments are repeated for the same geometry inputs at different locations (Vienna, Munich, Haifa, Genoa).

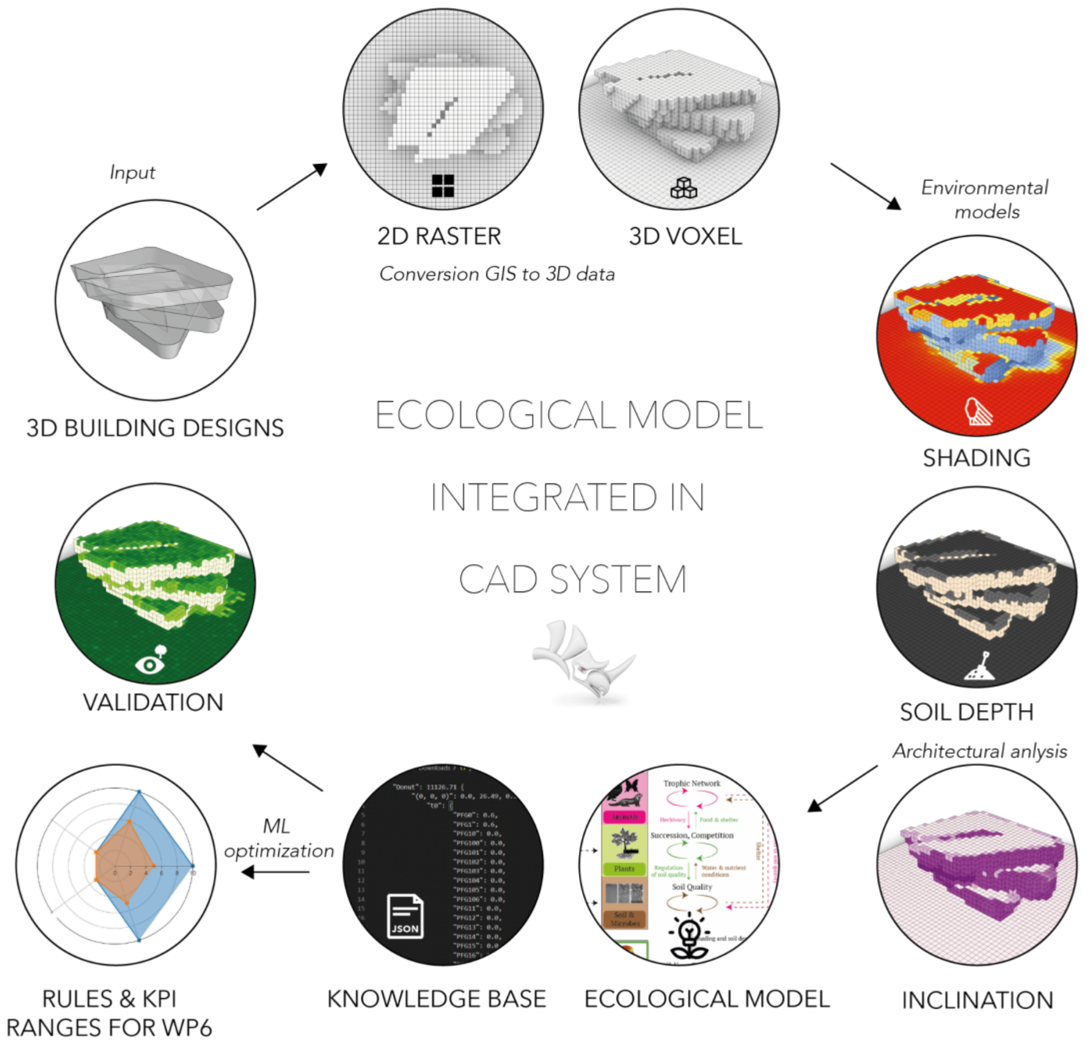


*Figure 22: Iterative computational experiments in the Knowledge generation framework (KGF).*

### 4.5 Knowledge base prototype (KB) and Machine learning model

Our interim KB 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 a key element for further knowledge extraction during future design generation and optimisation processes. In our initial experiments, we generated test datasets that contain architectural, environmental and ecological metadata for each voxel cell (building 1 = 6815 voxel cells, building 2 = 6864 voxel cells) as inputs. The amount of data points calculated from these inputs increases with the amount of possible PFGs (120 PFGs for final experiments) and AFGs (20 AFGs for final experiments). This means that for each experiment, we can assume more than **1.5 million data points**. In order to analyse vast amounts of data, recognise patterns, and make predictions with high accuracy, Machine Learning (ML) models offer numerous advantages. Once trained, ML models can adapt to new data and generalise their knowledge to make predictions on unseen examples. This flexibility allows them to handle dynamic and changing environments effectively. While traditional rule-based systems that rely on human-crafted rules and domain knowledge might struggle to handle massive and complex datasets, ML models can process and extract valuable insights from such data with relative ease. Another advantage is that ML models can learn from new data, allowing them to improve their performance over time. This process, known as retraining, ensures that the model stays up-to-date and accurate. To discover valuable insights and patterns that might not be obvious through traditional analysis methods (statistical analysis methods), ML models reveal these insights with the potential for better decision-making and a deeper understanding of the underlying processes. Despite these advantages, ML models also have limitations and challenges, such as the need for high-quality data, potential biases, and the requirement for careful monitoring and interpretability. However, when used appropriately and responsibly, ML models can provide significant benefits and open up new opportunities for problem-solving (Sarker 2021).
This section will present our initial ML approach for (1) the optimisation of our KB database, (2) the discovery of patterns, (3) extraction of KPI ranges and rules for design for WP5 and WP6, (4) potential of feature engineering (impact of different parameters, parameter weights), and (5) the potential for making predictions.

#### 4.5.1 Machine learning pipeline

We built a ML pipeline starting with the KGF generated data from virtual building envelopes and finishing by predicting ecological or architectural features. We choose the Python language because it is the state-of-the art environment for ML projects. Figure 23 shows our ML pipeline.

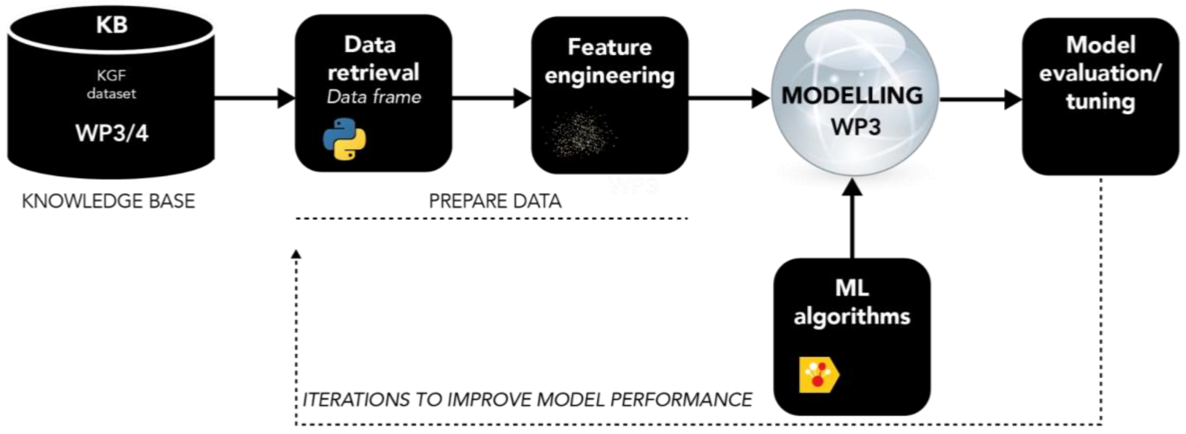
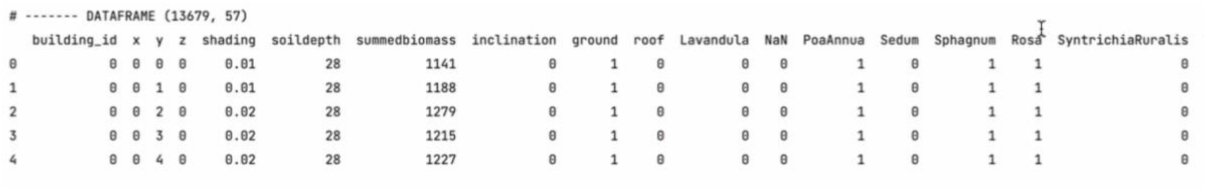


*Figure 23: ECOLOPES ML pipeline for KB optimisation.*

In the first step, we combine and transform the data generated in the KGF (architectural, environmental and ecological parameters) (Figure 24).



```
# ------- DATAFRAME (13679, 57)
   building_id  x  y  z  shading  soildepth  summedbiomass  inclination  ground  roof  Lavandula  NaN  PoaAnnua  Sedum  Sphagnum  Rosa  SyntrichiaRuralis
0            0  0  0  0     0.01         28           1141            0       1     0          0    0         1      0         1     1                  0
1            0  0  1  0     0.01         28           1188            0       1     0          0    0         1      0         1     1                  0
2            0  0  2  0     0.02         28           1279            0       1     0          0    0         1      0         1     1                  0
3            0  0  3  0     0.02         28           1215            0       1     0          0    0         1      0         1     1                  0
4            0  0  4  0     0.02         28           1227            0       1     0          0    0         1      0         1     1                  0
```

*Figure 24: Each row contains data from a voxel and the columns represent the features (inclination, soil depth, shading, biomass, PFGs, etc.).*

For exploratory analysis purposes, the data inputs can be plotted (Figure 25).

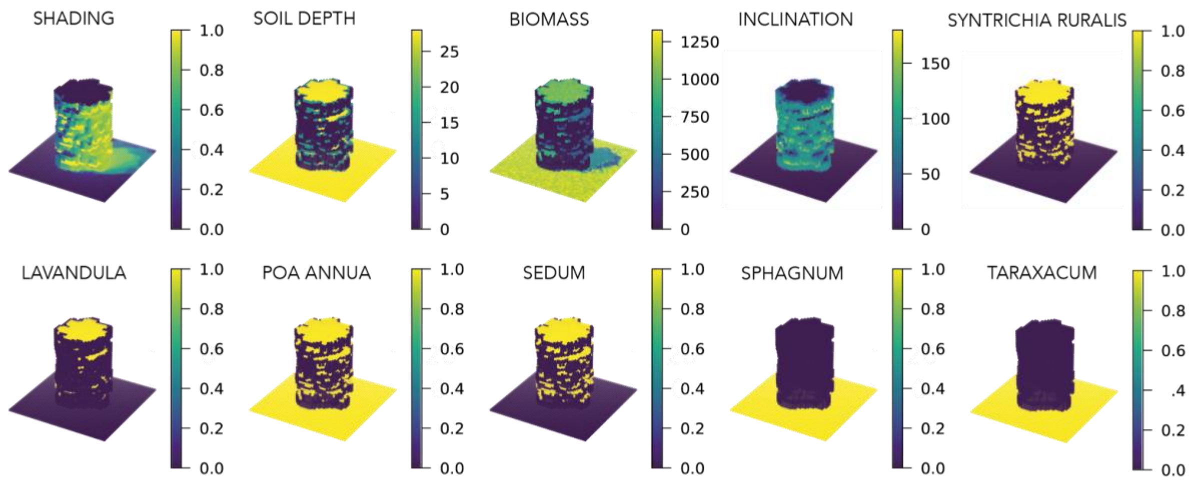
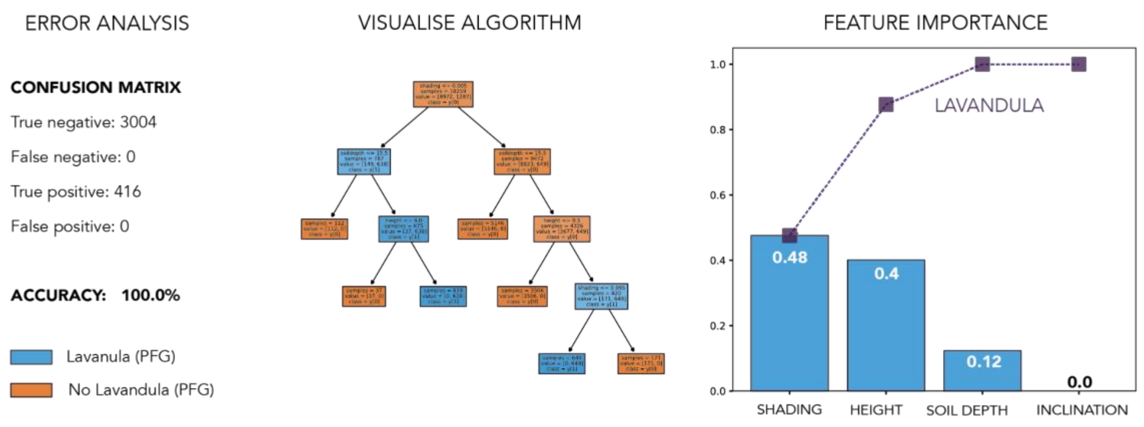


*Figure 25: Example for exploratory analysis of architectural and ecological features of Building 2.*

Although not used in our initial tests, we also establish tools for feature engineering, e.g., to calculate new features from **voxels** and **voxel neighbours**. This will increase the probability of finding novel important features/relationships and improve the quality of the predictions. In the next step, we implement ML algorithms and analysis tools. We choose the *decision-tree algorithm* from the scikit-learn package (Pedegrosa et a., 2011) and the *catboost algorithm* from Yandex technologies (CatBoost 2023 & Prokhorenkova et al 2018) which performs gradient boosting on decision trees by subsequently improving weak learners and converts them into strong learners (features). Both algorithms have been proven to (1) work stable on Python and being well documented and maintained, (2) to give highly accurate results and (3) provide a rich toolset for analysing the algorithms decisions and importance of features on the result. Each algorithm provides a version for classifications which are used for prediction of discrete classes (yes/no, 0/1) and a so-called *regressor* to predict continuous variables. In the last step, we confirmed the functionality and usefulness of our pipeline by processing data from two buildings and building some simple predictive models by using the methods described above. These first tests and results of the pipeline are described in the following section.

### 4.5.2 ML experiments

The objective of our experiments is to test the developed ML pipeline with KGF datasets to retrieve first patterns and predictions between parameter relationships. This section describes our experiments and first results step by step: We loaded and transformed the KB data and explored the features with visualisation and statistical tools. Then, we split the dataset into training and test datasets and balanced the dataset (increase the number of samples of the minority group) if necessary. The algorithm was trained and a model was built which contains the instructions on how to predict the outcome from the data. The model was then applied to the test dataset and predictions were obtained and analysed. In one concrete example we predicted the presence of **one specific PFG "Lavandula"** from architectural parameters – inclination and height, and environmental parameters – soil depth and shading. The model predicted the presence/no presence of Lavandula in all voxels correctly (accuracy: 100%) and found that 'shading' and 'height' account for 88 % of the total feature importance (Figure 26).


ERROR ANALYSIS

**CONFUSION MATRIX**

True negative: 3004

False negative: 0

True positive: 416

False positive: 0

**ACCURACY: 100.0%**

🟦 Lavanula (PFG)

🟧 No Lavandula (PFG)

*Figure 26: Error analysis, visualisation of algorithms, feature/parameter importance for predicting PFG "Lavandula".*

These results were expected because Lavandula is a mock-up PFG that was parametrised such that (1) it overgrows and outcompetes other PFGs and is hence not limited by shading; (2) it requires at least 16 cm deep soil; and (3) it can only live on the soil type "roof", which was distributed on any voxel higher than 1 m.
In summary, this simple and controlled experiment confirms the usefulness of the ML pipeline and highlights its potential. With more data, features and complexity included in the dataset we will be able to accurately forecast different scenarios and architectural design features from both modelled and real world datasets.

For the next version of the ML model, we will put our focus here:

1. Increase the number of building envelopes with different characteristics (volume, material, etc.), and building features (Rhino/ Grasshopper WP3).
2. Increase the number of features and establish dimensionality reduction and feature extraction methods such as principal component analysis or lasso.
3. Implement more algorithms and strategies for predictions (neural networks).
4. Speed up the calculations by engineering the scripts and implementing ray parallel computing.

### 4.5.3 KPIs and "rules for decision support" (WP3)

From our ML pipeline meaningful **KPI ranges** (shading, soil depth, height) can be extracted. In contrast to KPIs, **"rules for decision support"** are a combination of different parameters that lead to a certain result.
In our example, we targeted the PFG "Lavandula" to compute KPI ranges and "rules for decision support". The question asked is: **"What are the requirements to develop a building envelope that maximises the occurrence of the Lavandula PFG?"?** The decision tree results provide an excellent tool to find those conditions as they were learned from the training data.

The decision tree in Figure 26 revealed that a soil depth greater than 15.5 cm would support "Lavandula", that heights greater than 4.0 metres have a positive influence on the occurrence of Lavandula. Another indicator of success for Lavandula are shading values smaller than 99.5 %. These results are in line with the relationships programmed into the Ecological model.

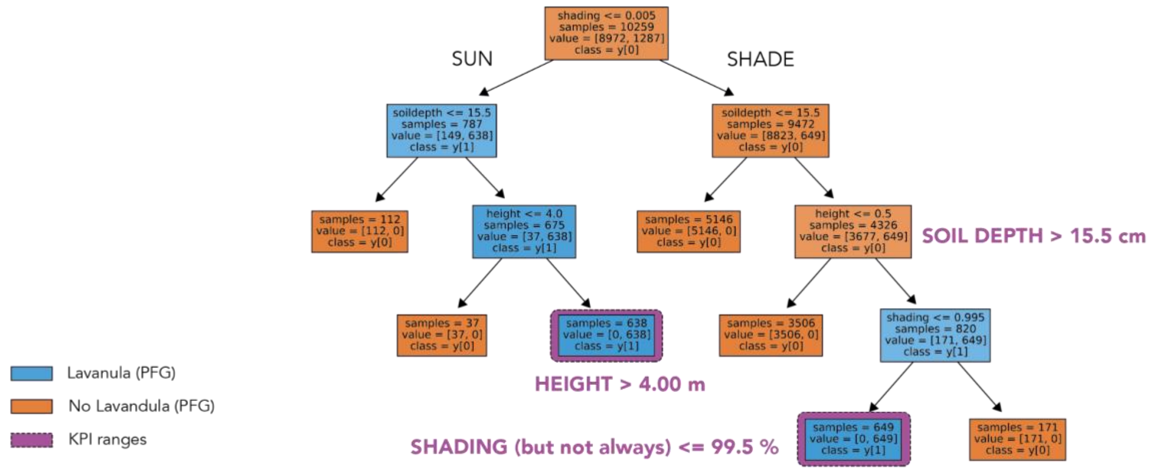


*Figure 27: Preliminary ML decision tree with 'learned' rules to predict the presence of Lavandula in a voxel cell.*

Grasshopper algorithms for generative design and optimisation processes developed in WP5 & WP6 need to reflect the ML decision tree. As the KPIs and feature importance change depending on the target feature (e.g., PFG "Lavandula), these changes need to be considered in the Grasshopper definitions for design and optimisation, too.

To translate ML outputs to Grasshopper, we recommend the following method: "KPI ranges" and "rules for decision support" can be referred to as "objectives" and "data" in Grasshopper. **Wallacei** is a free evolutionary engine that allows users to run evolutionary simulations in Grasshopper. It utilises highly detailed analytic tools coupled with various comprehensive selection methods to assist users to better understand their evolutionary runs, and make more informed decisions at all stages of their evolutionary simulations; including setting up the design problem, analysing the outputted results and selecting the desired solution or solutions for the final output (Wallacei 2020). In Wallacei, **KPIs** are referred to as the final aim and goal for optimisation and thus, considered in the Wallacei X component as "objectives". In contrast, **"rules for decision support"** were translated into a Grasshopper definition of a parametric 3D model of Building 2 and are imputed as "data". "Data" refers to geometry parameters. Additionally, feature importance values can be applied as **KPI weights** according to the results from "feature importance analysis" (Figure 28).

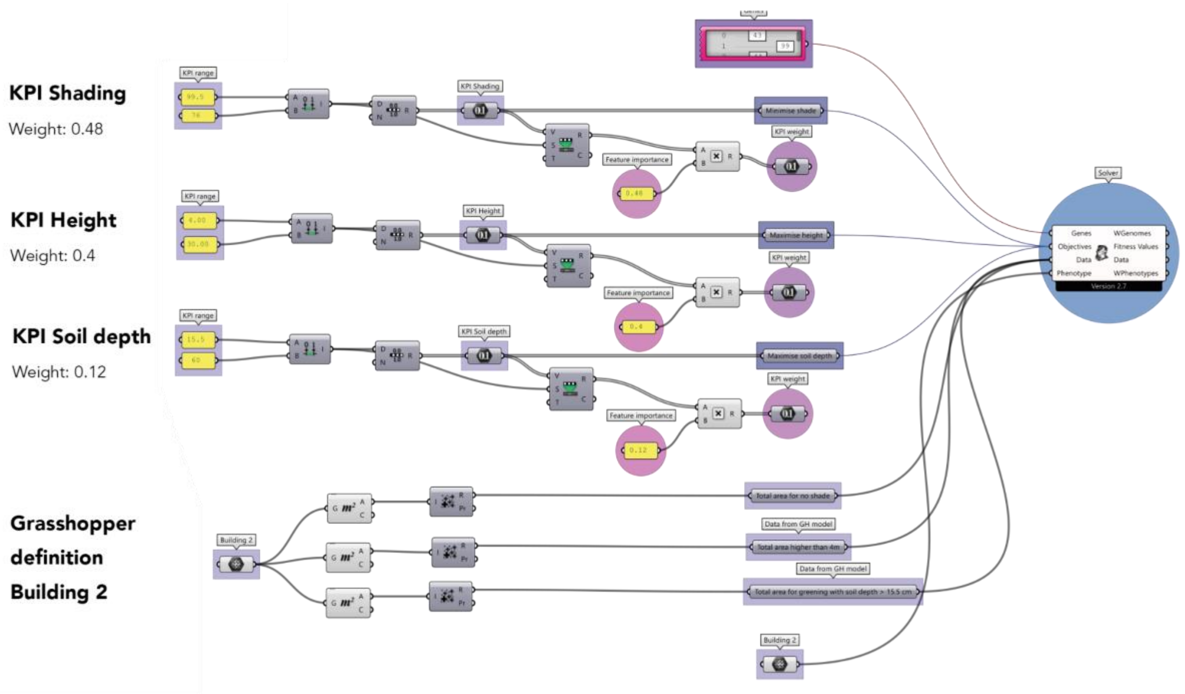


*Figure 28: ML model outputs (KPI ranges and extracted rules for design) to be used as inputs for Wallacei GH plugin (NSGA-2 algorithm, Wallacei x GH component).*

In summary, the output of the ML pipeline is an optimised KB with KPI ranges and rules for decision support which can then inform the EIM ontology (WP4), the generative design process (WP5) and optimisation (WP6).

## 4.6 Conclusions

In summary, the KGF is unique and a major achievement, because it represents the first systematic modelling approach to analyse the relationship between architecture, environment, and ecology. It has now been developed to the point that initial experiments lead to major insights into how architecture affects ecological communities. The KGF also allowed us to derive general rules for decision support and KPI ranges that can be used in a systematic design process. This will be valuable well beyond the ECOLOPES project. However, the developed system and its results need to be validated through real-world data and experiments. With respect to our ML pipeline, in future versions, by increasing the number of buildings and building features more accuracy can be achieved. Additionally, the ML model will be further developed (implementation of strategies, more algorithms and ray parallel computing). For the next version, we need to include a 3D animal model in our process, as well as algorithms for facade heterogeneity and more precise information about building structure and materials. Additionally, we need to define meaningful 3D geometry inputs.

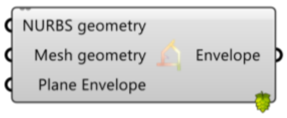
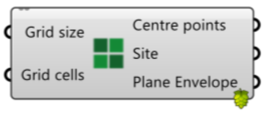
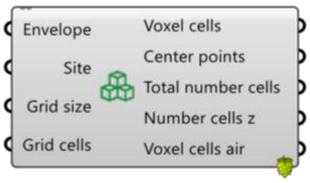
# 5. ECOLOPES Grasshopper Plugin prototype (Front-end tool)

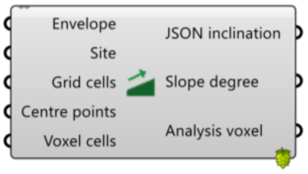
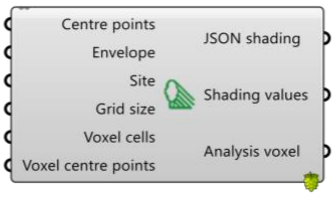
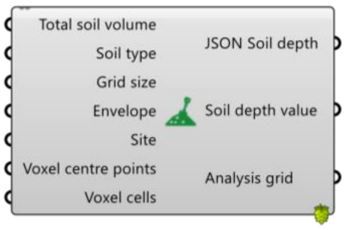
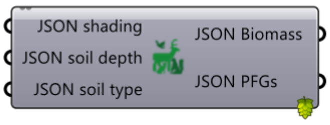
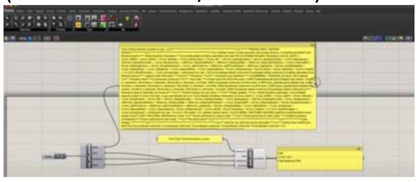
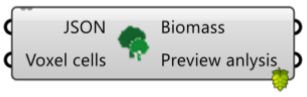
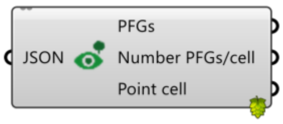
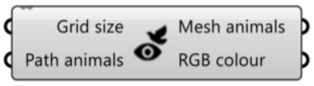
Task 3.4 *“Frontend development (M9-38)”* deals with the development of the ECOLOPES frontend tools based on Rhino/ Grasshopper. So far, we achieved the development of a tool that can visualise the Ecological model output on a 3D building. The tool generates visual information about the spatio-temporal distribution of biomass as well as metadata in a resolution of 1 cubic metre. Additionally, the tool enables KGF experiments with the goal to extract computationally relevant relationships, rules and KPI ranges for design decision support (Section 5.1). For the future version of the ECOLOPES frontend tool, a Grasshopper plugin, more functions for ontology-aided design generation (Section 5.2.1), optimisation (Section 5.2.2), and validation (Section 5.3) will be added.

## 5.1 Grasshopper plugin for ecological analysis in CAD (WP3)

After the submission of D3.2, we developed the first prototype of the ECOLOPES Grasshopper plugin. Grasshopper components were built by utilising McNeel’s .NET APIs as well as web applications leveraging Rhino.Compute and JavaScript APIs. There are components for data exchange between 2D raster and 3D voxel data, components for data conversion from metadata to JSON files, components for architectural, ecological and environmental analysis, and preview components to display community dynamics between plants (3D) and animals (2D) in Rhino, a 3D CAD environment. These components were written as Grasshopper Hops components (Hops 2023). Hops simplifies complex Grasshopper algorithms and adds additional functions, e.g., functions for parallel computing and cloud computing of Grasshopper algorithms (Table 8).

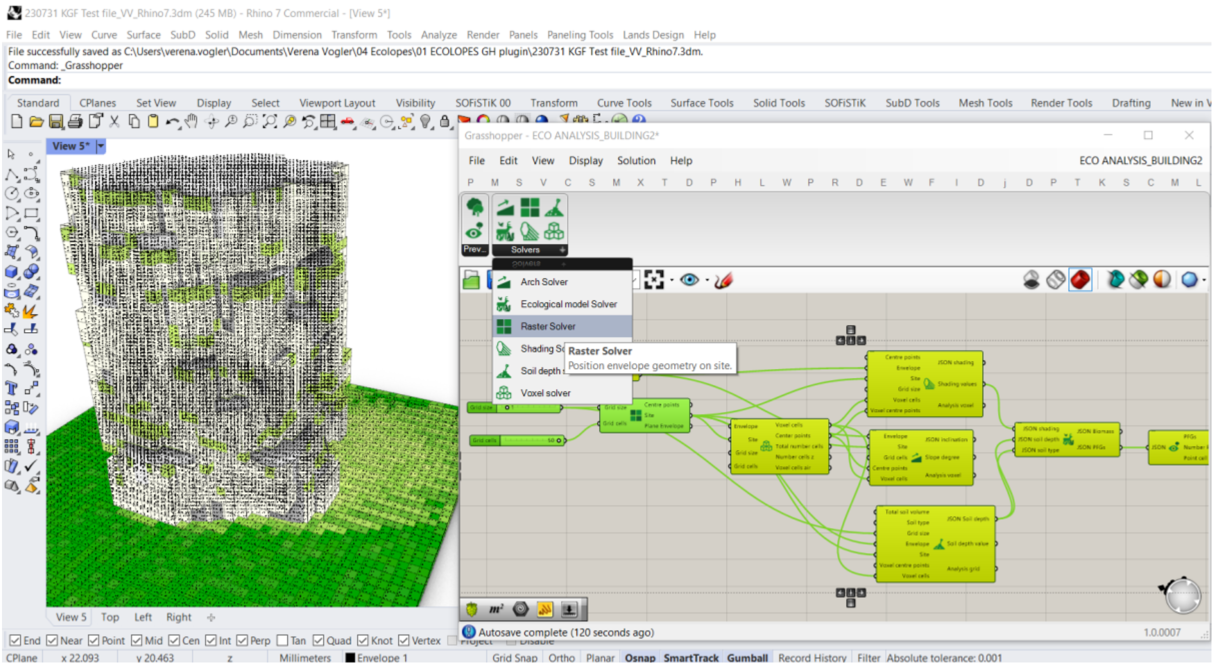
*Table 8: Updated components of the ECOLOPES Grasshopper plugin developed by MCNEEL.*

| Grasshopper Hops component | | Description |
|---|---|---|
|  NURBS geometry<br>Mesh geometry<br>Envelope<br>Plane Envelope | | **Envelope_Reader:** Inputs NURBS/mesh geometry and converts it into an “envelope geometry” for analysis. |
|  Grid size<br>Grid cells<br>Centre points<br>Site<br>Plane Envelope | | **Raster_Solver:** Positions 3D model for KGF experiments and outputs only KGF relevant information for data conversion. The component is much faster than in the previous version. |
|  Envelope<br>Site<br>Grid size<br>Grid cells<br>Voxel cells<br>Center points<br>Total number cells<br>Number cells z<br>Voxel cells air | | **Voxel_Solver:** Converts building envelope and the site into scalable voxel cells. This component runs much faster in the current version (C# script). |

| | | |
|---|---|---|
|   Envelope<br>Site<br>Grid cells<br>Centre points<br>Voxel cells<br>JSON inclination<br>Slope degree<br>Analysis voxel | | **Arch_Solver**: Computes inclination values at resolution of the voxel model. The output is a correctly formatted JSON file with metadata (Inclination values/ voxel cell). |
|   Centre points<br>Envelope<br>Site<br>Grid size<br>Voxel cells<br>Voxel centre points<br>JSON shading<br>Shading values<br>Analysis voxel | | **Shading_Solver:** Computes percentage of shading for a defined location, time of the day for each voxel cell. The output is an analysis mesh and a correctly formatted JSON file with metadata (Shading values/ voxel cell). |
|   Total soil volume<br>Soil type<br>Grid size<br>Envelope<br>Site<br>Voxel centre points<br>Voxel cells<br>JSON Soil depth<br>Soil depth value<br>Analysis grid | | **SoilDepth_Solver:** Computes soil depth and soil type values for each voxel cell. The output is an analysis mesh and a correctly formatted JSON file with metadata (Soil depth and type values/ voxel cell). |
|   JSON shading<br>JSON soil depth<br>JSON soil type<br>JSON Biomass<br>JSON PFGs | | **PFG_Solver:** Runs C++ Ecological model on Windows cloud server and outputs a correctly formatted JSON file with metadata (Biomass and PFG/ voxel cell).  |
|  | | **PFGBiomass_Preview3D:** Displays Ecological model outcome on 3D building envelope geometry as a colour gradient. Outputs biomass values/ voxel cell. |
|  | | **PFGMetada_Preview3D:** Displays Ecological model outcome as on 3D building envelope geometry as text. Outputs PFG number and names/ voxel cell. |
|  | | **AFGMetada_Preview2D:** Displays Animal model outcome as 2D colour gradient in CAD. Outputs number of AFGs/ raster cells. (OLD) |

The developed Hops components were tested on a local machine and on our Windows cloud server. Hops is a relatively new technology for Rhino/ Grasshopper and is still under active development. Besides the advantage of including existing external Grasshopper plugins (e.g., Ladybug tools) for the development of new Hops components, there are a few issues with Hops: Hops sends and receives requests from a Rhino.Compute server. This process adds to

the processing time of the Grasshopper definitions that are referenced by Hops. Another issue is that versioning and debugging of Hops components is more time consuming as the reference definitions do not display errors. Lastly, there are still bugs in Hops which result in unexpected crashes and non-working components. However, the MCNEEL development team is aware of these issues and they will be improved with time and with the increasing number of Hops/ Rhino.Compute users that rely on this technology.
To improve the performance of our Grasshopper components, especially for KGF experiments, parts of the visual GH code are replaced with C# code. Additionally, existing components can be compiled as a Grasshopper plugin. In its early test version it works for ecological analysis in CAD (Figure 29).


230731 KGF Test file_VV_Rhino7.3dm (245 MB) - Rhino 7 Commercial - [View 5*]

File successfully saved as C:\Users\verena.vogler\Documents\Verena Vogler\04 Ecolopes\01 ECOLOPES GH plugin\230731 KGF Test file_VV_Rhino7.3dm.
Command: _Grasshopper
Command:

Grasshopper - ECO ANALYSIS_BUILDING2*

Raster Solver
Position envelope geometry on site.

Autosave complete (120 seconds ago)

*Figure 29: Screenshot of compiled ECOLOPES Grasshopper plugin (MCNEEL) running in Rhino. The developed GH components were added as functions with descriptions.*

Additional functions for decision support and validation are under development (WP5-WP7). They will shortly be introduced in the following sections. Lastly, in M38, WP3 will submit a demonstrator of the ECOLOPES platform and its integrated front-end tools. We will try to have an early prototype running for test users, and use cases.

## 5.2 Future Grasshopper functions

### 5.2.1 Ontologies (WP4/ WP5)

Grasshopper Hops components related to the EIM Ontology and the ontology-aided design process are further explained in D4.2, D5.2, and D5.3 (all M30).

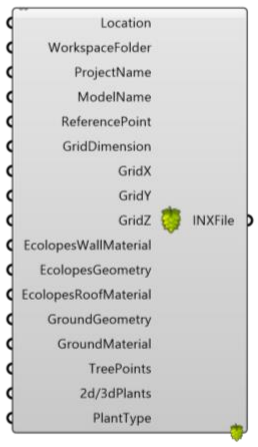
### 5.2.2 Optimisation algorithms (WP6)

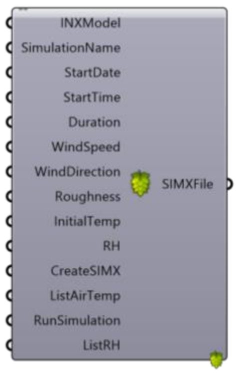
Grasshopper Hops components related to the optimisation process are further explained in D6.1 (M30).

## 5.3 Validation tools (WP7)

UNIGE DAD developed three Grasshopper Hops components to assess human thermal comfort computationally (see D3.2, Section 6.4 Human comfort analysis for validation). Assessing human thermal comfort ensures that the *ecolope* design outcome to evaluate how humans are affected by the resulting changes to the microclimate and outdoor thermal conditions on the site. The Grasshopper components were developed using existing Grasshopper plugins such as Ladybug tools, Dragonfly Legacy and Morpho (Morpho 2020), a plugin to create Envimet 2.5D and 3D models (INX), write configuration files (SIMX), run simulation and read results. Envi-Met is a high-resolution 3D modelling software package that accurately simulates complex microclimatic processes (ENVI-met 2014). Through Morpho it is possible to to run simulations, e.g., wind speed CFD simulations, in Rhino and Grasshopper. UNIGE DAD compiles three Grasshopper Hops components to first, convert *ecolopes* geometry into an .INX file for Envimet; second, to read these files into Envi-met, and lastly, to to visualise the simulations results in Rhino.

*Table 9: Grasshopper Hops components for outdoor human thermal comfort analysis with the external Grasshopper plugins Ladybug tools, Dragonfly, and Morphon (UNIGE DAD, WP7).*

| Grasshopper Hops component | Description |
|---|---|
|  Location<br>WorkspaceFolder<br>ProjectName<br>ModelName<br>ReferencePoint<br>GridDimension<br>GridX<br>GridY<br>GridZ<br>INXFile<br>EcolopesWallMaterial<br>EcolopesGeometry<br>EcolopesRoofMaterial<br>GroundGeometry<br>GroundMaterial<br>TreePoints<br>2d/3dPlants<br>PlantType | This component allows the conversion of the *ecolope* geometry into an INX file, a 2.5D model with the building geometry, related materials and the position of vegetation on the building envelope and on the surrounding area. |

| | |
|---|---|
|   INXModel<br>SimulationName<br>StartDate<br>StartTime<br>Duration<br>WindSpeed<br>WindDirection<br>Roughness<br>InitialTemp<br>RH<br>CreateSIMX<br>ListAirTemp<br>RunSimulation<br>ListRH<br>SIMXFile | This component allows the definition of the parameters for the outdoor thermal comfort simulation in Envi-met. The inputs are the INX file generated by the first component and the climatic data for the simulation. |

All components require as inputs the computed voxel model and the Ecological model outputs (Section 5.1). Thus, the spatial distribution of PFGs on the building envelope and the site generates an INX model that can be read in Envi-met (Figure 30).

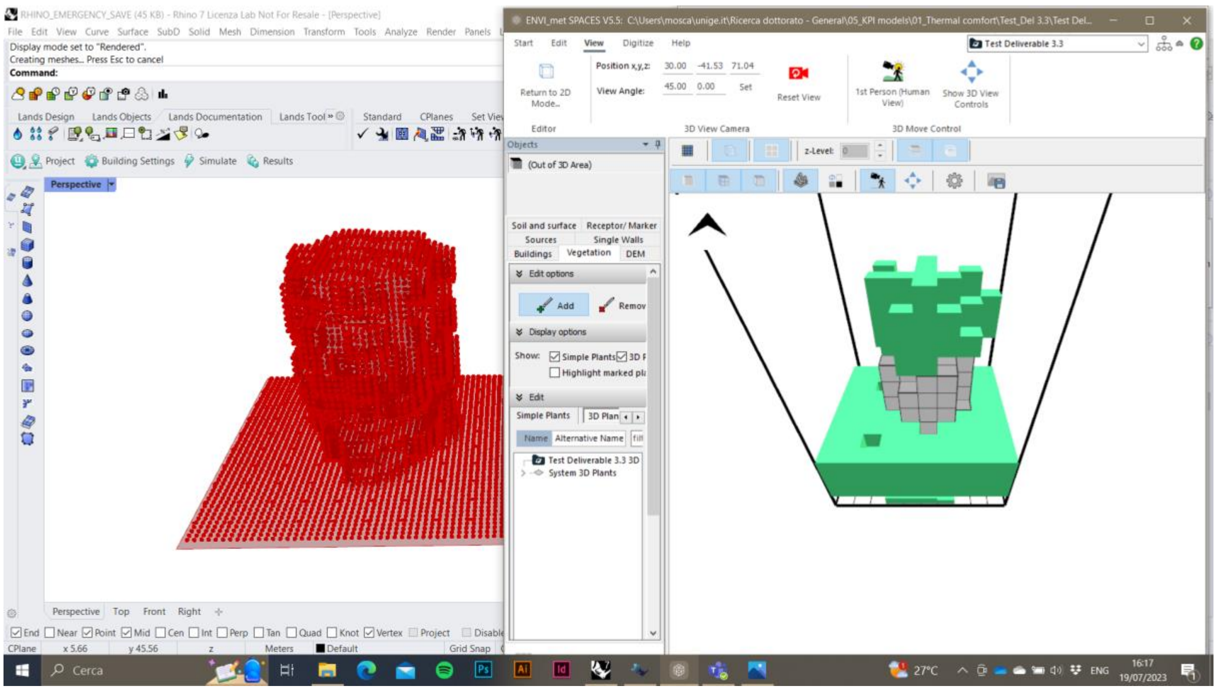


*Figure 30: Screenshot of Initial tests of outdoor thermal comfort simulations depending on the green on the "envelope" of Building 2. Inputs are the building geometry and the centre points of the voxel model (UNIGE DAD).*

## 5.4 Conclusions

The ECOLOPES front-end tool is a Grasshopper plugin based on Rhino.Compute/ Hops. It is at a very early development stage. In the project it will be used for (1) KGF experiments, (2) ecological analysis of CAD models, (3) decision support for *ecolopes* design (WP5 and WP6), and (4) for validation (WP7). However, the Ecological model outcome needs to be validated, and unstable (Hops) components substituted by C# components. For the next versions the objective is to make a stable demonstrator (M38) which can be tested by our key accounts from AEC. Additionally, it is crucial to document development steps, establish a version control system, and licensing policies for the ECOLOPES front-end tool and also for the ECOLOPES platform. These strategies are introduced in the following Chapter 6.

# 6. SOFTWARE DEVELOPMENT AND INTEGRATION STRATEGIES

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 to manage software development based on an incremental approach rather than an agile or waterfall system. This approach 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.
Chapter 6 will present our software development and integration strategies with a focus on communication (Section 6.1), documentation (Section 6.2), version control (Section 3), data storage and distribution (Section 6.4), and lastly, licensing (Section 6.5).

## 6.1 Communication

During the past months, we focussed on discussions within a smaller team of members involved in development. To ensure consistency with respect to development and integration McNeel coordinates technical workshops every two weeks, and the consortium organises technical meetings every 3 months that address more general issues.

## 6.2 Documentation

To understand processes and decisions made during development and integration, documentation is necessary. Documentation can emphasise the bigger picture of the current development state and helps to identify existing bottlenecks and problems. Our current documentation includes, (1) the documentation of the Ecological model (WP4), (2) documentation of the CAD integration (WP3), (3) the KGF (WP3), (4) the computational workflow, (5) the ECOLOPES platform (WP3), and (6) the front-end tools (WP3). Our documents include process descriptions and summary of interim results (D3.1- D3.3, reports of year 1 and 2, diagrams, screenshots, graphs and powerpoint presentations). Additionally, we document our WP3 meetings in the form of protocols and powerpoint presentations on Microsoft Teams.

## 6.3 Version Control

Besides documentation, it is crucial to be able to track changes using version control for development. For the current ECOLOPES plugin (Version 0.4) and the KGF (version 0.2), we used a shared Excel table for these purposes. Additionally, changes made to project files of the Ecological model are tracked in private GitLab projects (https://gitlab.com/ECOLOPES). In the future, we might consider moving documentation and all project files to GitLab, and to use an environment manager, for instance, Conda (Conda 2023).

### 6.4 Data storage and distribution

How WP3 related raw and analysed data is stored was layouted in D1.1 (M6). The idea was that all ECOLOPES data is stored on a shared platform (Microsoft Teams, and the cloud-based data storage). At the moment, WP3 related data is still not centralised in one storage. This needs to be done in the future for the WP3 demonstrator to comply with the data management plan.

### 6.5 Licensing

Different technical components for the ECOLOPES platform will require a software licence (e.g., ECOLOPES Grasshopper plugin) or even a patent (Ecological model). There are options for free software licensing (e.g., GNU) but also commercial licensing options. For instance, there is the possibility to embed parts of the plugin into an existing Rhino plugin for Landscape architects with an existing user community. The Ecological model is currently only shared privately and contains no free licences (all rights reserved). An open licence and public access to the code will be granted upon publication.

### 6.6 Conclusions

Development methods with respect to communication, documentation, data storage, version control and licensing were summarised in each section of Chapter 6. In the current development method, all steps are documented and changes to the software versions are tracked, but this data must be made available centrally. Additionally, licensing will need to be discussed.

# 7. WP3 PRESENTATIONS AND PUBLICATIONS

The following Chapter 7 will provide more information about WP3 dissemination activities. Since the submission of D3.2, we presented the CAD integration of the Ecological model publicly (Section 7.1). Additionally, we developed a publishing plan for the next academic year (Section 7.2).

### 7.1 WP3 presentations

The purpose of public presentations was to communicate the development efforts made for the integration of ecological analysis in Rhino/Grasshopper. Another objective was to understand how these efforts are received by the user community, and how we can get them involved in the discussion as well as testing (Table 10). A crucial experience was that we received positive feedback from students, researchers and the faculty of Master degrees in computational design and architecture (EmTech Master class Architectural Association London, and CITA Master class Copenhague), as well as from professional architecture and engineering firms such as Herzog & de Meuron architects, and Thornton Tomasetti. The majority of questions were related to when they will be able to test the ECOLOPES Grasshopper plugin to apply it to their own designs, and about the precision and validity of the computed results.

*Table 10: Public presentations at international conferences & symposiums.*

| Conference | Topic of the presentation | Presenter | Type | Institution | Reference |
|---|---|---|---|---|---|
| Digital Landscape Architecture (DLA) Conference 2022, Harvard Graduate School of Design, Cambridge, Massachusetts | Artificial coral reef design & ecosystem-aware design | V. Vogler | Talk | McNeel | https://2022.dla-conference.com/home/ |
| Rhino User Meeting Copenhague 2022, Denmark | Rhino developments and Research projects | L. Fraguada | Talk | McNeel | https://www.eventbrite.com/e/rhino-user-meeting-copenhagen-2022-tickets-347818543627 |
| American Society of Landscape Architecture (ASLA) conference 2022: Designing a better future, San Francisco, USA | Assessment of Environmental Aspects of Landscape Projects Using Parametric Design | V. Vogler | Educational accredited session | McNeel | https://www.aslaconference.com/ |
| URBIO International Conference, 2022, Leipzig, Germany | Integrating ecological modelling in a 3D CAD system for urban planning and for regenerative urban ecosystems | V. Vogler | Talk | McNeel | https://www.urbionetwork.com/ |
| EmTech lecture series, Architectural Association London, 2023 EmTech Master Class | Horizon 2020 Research Projects/ Developments | V. Vogler | Lecture | McNeel | https://www.facebook.com/EmtechStudio/ |
| Rhino User Meeting Basel @Herzog & de Meuron Architects Base 2023, Switzerland | Horizon 2020 Research Projects @ McNeel Europe | V. Vogler | Talk | McNeel | https://events.mcneel.eu/rhino-user-meeting-basel/ |

| | | | | | |
|---|---|---|---|---|---|
| Symposium "ArchitectureNature – NatureArchitecture" 2023, Department of Architecture TUM School of Engineering and Design Technical University of Munich | Artificial coral reef design & underwater monitoring strategies/ multi-species design | V. Vogler | Talk | McNeel | https://www.arc.ed.tum.de/en/arc/about-us/news/news-single-view-en/article/symposium-architecturenature-naturearchitecture/ |

## 7.2 WP3 publishing plan

The objective of the publishing plan is to discuss attribution, authorship and publication responsibilities as early as possible. WP4 publications (PFG paper by UNIGE, Animal model paper by TUM, Ecological model paper by TUM/SAAD) need to be published before WP3 papers can be published. This publishing plan is a point of reference for assessing participants' roles (Table 11).

*Table 11: Planned WP3 papers.*

| Order | Topic of the paper | Journal/ estimated date for submission |
|---|---|---|
| **1.** | Integration of ecological modelling in a 3D CAD System paper (WP3) | Journal of Digital Landscape Architecture (JoDLA) paper, abstract submission November, 2023 (MCNEEL, TUM & SAAD) |
| **2.** | Knowledge Generation Framework paper (WP3) | Journal unclear (interdisciplinary journal if possible: architecture, data science, ecology), mid 2024, (MCNEEL, TUM & SAAD) |

# 8. CONCLUSIONS AND RECOMMENDATIONS FOR DEMONSTRATOR

This report has described the interim version of the ECOLOPES platform development. The last chapter of this report summarises the achievements made in WP3 since the submission of D3.2 and the report of year 2 (D1.5, M24). Additionally, we make suggestions how the current version of the ECOLOPES platform including back-end and front-end services can be improved until the submission of the demonstrator in M38.

In **Chapter 2**, we updated on the cloud-based backend and frontend deployment with Rhino.Compute, and advances made with respect to integration of technical components and data flows in the ECOLOPES computational system. Moving forward, the future developments concerning the ECOLOPES platform encompass three main aspects: debugging and enhancing existing components, as well as seamlessly integrating the components that are currently in the development phase (Regional model, EIM ontologies, decision support system for design optimisation).

In **Chapter 3**, we layouted the step-by-step processes behind a successful integration of the ecological model in a 3D CAD system (Rhino/ Grasshopper) using two case study buildings. The results of our achievements are that ecological modelling becomes an intrinsic part of the form finding process for building envelopes. We overcame the obstacle from 2D to 3D analysis results. For the demonstrator, we will work on how to make this process more efficient by introducing parallel computing resources and through the optimisation of the developed algorithms. Additionally, the currently integrated ecological model is a simplified version of a more complex model. At the moment it only includes PFGs and soil classification. Also architectural and environmental parameters are adapted to plants. AFGs and AFG relevant parameters (e.g., facade material and possibility for artificial shelter) need to be introduced in the next version. Lastly, architects will also need a validated ecological model to apply the developed approach in design.

**Chapter 4** shows the latest achievements made with respect to the Knowledge Generation Framework (KGF). The approach is unique because it represents the first systematic modelling approach to analyse the relationship between architecture, environment, and ecology. At its current stage of development, initial computational experiments have yielded significant insights into how architecture impacts ecological communities. These experiments resulted in datasets with more than 1.5 million data points per experiment. To reveal patterns and the impact of parameters, a ML model was developed. Our initial results show that by the developed method, we have been able to extract design principles and KPI ranges for design decision support. This achievement will be valuable well beyond the ECOLOPES project. However, the developed system and its results need to be validated through real-world data and experiments that can be initiated once we have a more efficient animal model.

In **Chapter 5**, we show the early version of the ECOLOPES front-end tool, a Grasshopper plugin based on Rhino.Compute/ Hops. In the project the plugin will be useful for (1) KGF experiments, (2) ecological analysis of CAD models, (3) decision support for *ecolopes* design (WP5 and WP6), and (4) for validation (WP7). For the next versions, the Ecological model outcome needs to be validated, and unstable (Hops) components substituted by C# components. The objective is to present a stable demonstrator (M38) which can be tested by our key accounts from AEC.

**Chapter 6** focused on software development methods highlighting communication, documentation, data storage, version control and licensing. At the moment, each development step is documented, and any modifications to the software versions are tracked. However, for the next versions, it is crucial to ensure that this data is accessible, that projects can be modified on GitLab, and that licensing options will be discussed.

The report concludes with **Chapter 7**, which highlights the importance of public presentations and, more importantly, audience feedback on our endeavours. Additionally, it lays out a publishing plan for our first WP3 papers about CAD integration and the KGF.

## REFERENCES

ECOLOPES (2020), Grant agreement ID: 964414, EXCELLENT SCIENCE - Future and Emerging Technologies (FET), coordinated by Technical University Munich. DOI: 10.3030/964414.

**Research Papers:**

Heeramaglore M., Kolbe T. H. (2022). Semantically enriched voxels as a common representation for comparison and evaluation of 3D building models. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume X-4/W2-2022. https://doi.org/10.5194/isprs-annals-X-4-W2-2022-89-2022.

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. https://doi.org/10.1080/24751448.2022.2040305.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). Catboost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems* (Vol. 2018-December, pp. 6638–6648). Neural information processing systems foundation.

Sarker, I.H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science* 2, 160. https://doi.org/10.1007/s42979-021-00592-x.

Weisser, W.W., Hauck, T.E. (2017). ANIMAL-AIDED DESIGN – using a species' life-cycle to improve open space planning and conservation in cities and elsewhere. *bioRxiv* 150359. https://doi.org/10.1101/150359.

Weisser, W. W., Hensel, M., Barath, S., Culshaw, V., Grobman, Y. J., Hauck, T. E., Joschinski J., Ludwig F., Mimet, A., Perini, K., Roccotiello, E., Schloter, M., Shwartz, A., Hensel, D. S., & Vogler, V. (2023). Creating ecologically sound buildings by integrating ecology, architecture and computational design. *People and Nature*, 5, 4–20. doi.org/10.1002/pan3.10411.

**Software libraries and tools:**

CatBoost (2023). State-of-the-art open-source gradient boosting on decision trees library.[Online]. Available: https://yandex.com/dev/catboost/. [Accessed: 23-July-2023]

Conda (2023). Package, dependency and environment management. [Online]. Available: https://docs.conda.io/en/latest/. [Accessed: 22-July-2023].

ENVI-met (2014). A microscale 3D software model for simulating complex urban environments for Windows. [Online]. Available: https://www.envi-met.com/. [Accessed: 30-June-2023].

Grasshopper by David Rutten (2008). [Online]. Available: https://www.grasshopper3d.com/. [Accessed: 30-June-2023].

Grasshopper component (2018). [Online]. Available: https://developer.rhino3d.com/guides/grasshopper/what-is-a-grasshopper-component/. [Accessed: 17-June-2023].

GitHub Hops (2021). [Online]. Available: https://github.com/mcneel/compute.rhino3d/blob/7.x/src/hops/HopsComponent.cs. [Accessed: 6-June-2023].

GitHub Rhino.Compute McNeel (2021). REST geometry server based on RhinoCommon and Grasshopper [Online]. Available: https://github.com/mcneel/compute.rhino3d. [Accessed: 12-June-2023].

GitHub Rhino.Inside (2021). [Online]. Available: https://github.com/mcneel/rhino.inside. [Accessed: 15-June-2023].

Hops component (2023). Hops adds external functions to Grasshopper. [Online]. Available: https://developer.rhino3d.com/guides/compute/hops-component/. [Accessed: 15-July-2023].

Ladybug Tools (2013). Free environmental design knowledge and tools. [Online]. Available: https://www.ladybug.tools/. [Accessed: 12-June-2023].

MidJourney (2022). Generative artificial intelligence program and service developed by Mindjourney Inc. [Online]. Available: https://www.midjourney.com/home/?callbackUrl=%2Fapp%2F. [Accessed: 12-October-2022].

Mid Journey Prompt (2023). [Online]. Available:https://www.grasshopper3d.com/. https://www.prompthunt.com/prompt/cl9g9641m3501r5oxr4yc0vhn?selectedAsset=cl9g964mi3557r5oxhpuszrxq [Accessed: 30-June-2023].

Morpho (2020). A plugin to create Envimet 2.5D and 3D models (INX), write configuration files (SIMX), run simulation and read results. [Online]. Available: https://github.com/AntonelloDN/Morpho. [Accessed: 12-June-2023].

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: 10-June-2023].

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: 1-June-2023].

RhinoCommon API 2023 https://developer.rhino3d.com/guides/rhinocommon/what-is-rhinocommon/. https://developer.rhino3d.com/api/ [Accessed: 15-June-2023].

Rhino Development (2018). Rhino and Grasshopper Developer Documentation. [Online]. Available: https://developer.rhino3d.com/. [Accessed: 12-June-2023].

Rhino.Inside McNeel (2021). Rhino and Grasshopper inside 64 bit applications for Windows. [Online]. Available: https://www.rhino3d.com/features/rhino-inside/. [Accessed: 10-June-2023].

Rhino.Inside.Revit McNeel (2021). Rhino and Grasshopper inside Revit. [Online]. Available: https://www.rhino3d.com/inside/revit/beta/. [Accessed: 12-June-2023].

Rhino SDK (2018). The Rhino C/C++ Software Development Kit. [Online]. Available: https://developer.rhino3d.com/guides/cpp/what-is-the-cpp-sdk/. [Accessed: 9-June-2023].

Rhino.Python (2018). [Online]. Available: https://developer.rhino3d.com/guides/rhinopython/what-is-rhinopython/. [Accessed: 8-June-2023].

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay E. (2011). Scikit-learn: Machine Learning in Python. *The Journal of Machine Learning Research*. 12, null, 2825–2830.

Wallacei (2020). An Evolutionary Multi-Objective Optimisation and Analytic Engine for Grasshopper. [Online]. Available: https://www.wallacei.com/. [Accessed: 5-June-2023].

**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].