

APPROVED: 24 June 2021

doi:10.2903/sp.efsa.2021. EN-6694

The EU Bee Partnership (EUBP) Prototype Platform: data model description

Noa Simón Delso, Gregor Sušanj and Andrés Salazar Abello

BeeLife European Beekeeping Coordination, Louvain la Neuve, Belgium

Abstract

The EU Bee Partnership is an initiative that was created in 2017 to bring together stakeholders who have an interest in data sharing on bee health and beekeeping in the EU. The EU Bee Partnership Prototype Platform aims at collecting and exchanging standardised data on pollinator health and other relevant data based on the Bee Hub Proof of concept initially led by BeeLife European Beekeeping Coordination within the framework of the Internet of Bees Project. This new stage of development responds to the main conclusions of the EU Bee Partnership: the need for data standardisation and communication for the benefit of stakeholders and EFSA. The work developed during the prototyping of the platform funded by EFSA has included the design and implementation of data models for the data integrated in the Prototype Platform (<https://bee-ppp.eu>). A procedure for data and metadata integration has been created and implemented, including validation of the data quality. The platform was enriched with datasets coming from existing projects and public databases. Additionally, several algorithms for exploring the data and extracting ready-to-use information have been proposed in this phase providing examples and avenues for future development. Finally, a reporting feature was added, and some reports were produced, to illustrate the type of information that the analyses of the platform could generate in the future for the benefit of stakeholders, institutions and more sustainable bee health and beekeeping in the EU.

© European Food Safety Authority, 2021

Key words: EU Bee Partnership, bee health, beekeeping, data integration, data standardisation, data processing, data communication

Question number: EFSA-Q-2021-00009

Correspondence: sc.secretariat@efsa.europa.eu

Disclaimer: The present document has been produced and adopted by the bodies identified above as authors. This task has been carried out exclusively by the authors in the context of a contract between the European Food Safety Authority and the authors, awarded following a tender procedure. The present document is published complying with the transparency principle to which the Authority is subject. It may not be considered as an output adopted by the Authority. The European Food Safety Authority reserves its rights, view and position regarding the issues addressed and the conclusions reached in the present document, without prejudice to the rights of the authors.

Acknowledgements: The authors would like to specially thank Nuno Xavier Capella, from the University of Coimbra, Coimbra, Portugal and Michael Rubinigg, from Biene Österreich, Vienna, Austria, for their investment and feedback in the development of this prototype. The authors would like to thank the data providers who engaged with us in this project and the beeXML group for the discussions on data standardisation. The authors would like to thank the EU Bee Partnership for believing in the project and supporting it.

Suggested citation: Simón Delso N, Sušanj G and Salazar Abello A, 2021. The EUBP Prototype Platform: data model description. EFSA supporting publication 2021:EN-6694. 118 pp. doi:10.2903/sp.efsa.2021.EN-6694

ISSN: 2397-8325

© European Food Safety Authority, 2021

Reproduction is authorised provided the source is acknowledged.

Summary

The EU Bee Partnership (EUBP) was established in 2017 to promote stakeholders' engagement on harmonised and interoperable data exchange (i.e. data gathering and sharing), processing and communication in bee health and beekeeping.

EU Bee Partnership Prototype Platform aims at collecting and exchanging standardised data on pollinator health and other relevant data based on the Bee Hub Proof of concept initially led by BeeLife European Beekeeping Coordination within the framework of the Internet of Bees Project. This new stage of development aims at responding to the main conclusions of the EU Bee Partnership: the need for data standardisation and communication (section 1).

The web interface has evolved during the prototyping phase and a number of features have been included in the platform. As a result, the user will find now a blog describing news and activities related to the platform (section 2); a section on the data providers and the projects providing data, providing visibility to the institutions/persons and projects contributing to the platform, respectively; a section dedicated to the data (including metadata, statistics and links with other initiatives of data standardisation); a reporting section including automatic or manual reports showing the results of the analyses of data contained in the database; a data provider metadata management section, where the provider can log in and modify or administer his/her account and datasets (modify, add, erase, change the public sharing status, etc.); and a Data Error Report Alarm, that is setup with an external service, which tracks and reports any errors found.

During the development of the Prototype an important effort has been put on the development of the Data model. There is a large data gap in bee related data standardisation. For this reason, a large part of the project was dedicated to design, create and implement the Data model in a way that can be the most versatile possible and can serve to interconnect datasets of different origins. The objective was that the data, independently of how it was stored in the Prototype Platform, could be used in the future for other purposes with other tools. The approach has been to work on practical cases: specific datasets candidate to be integrated into the platform contributed to the definition of the first data models. In doing so, an evolutive data model is built up, solving the specific situations observed with real data. Due to the need to document the standardisation efforts developed by the prototype, a large part of this report is dedicated to thoroughly describe the different data models.

The data integrated during the prototyping phase has been classified into different categories, namely Demography and Biology, Ecology, Automatic Monitoring, Bee Health, Weather Conditions, Landscape, Environmental Pollutants, Breeding, and Socioeconomic Aspects (section 3). In the future this taxonomy may be modified to better adapt to the data integrated and the comprehension of the users. A total of 16 new datasets have been integrated into the prototyping phase, belonging to several of the groups previously mentioned. These datasets come from scientific publications, private initiatives, (inter-)governmental datasets or communications. Each of these datasets have contributed to further develop the Data model. Metadata about the Data providers, Data Owners and Projects promoting the creation of the data is also collected.

A procedure for the integration of data has been developed to ensure the data quality and usability, which includes the following phases: (1) identification of relevant datasets; (2) Establishment of contact with the data provider; (3) Drafting and signature of data sharing agreement, if needed; (4) Data acquisition; (5) Data and metadata description; (6) Data cleaning and preparation; (7) Import of dataset in the platform; (8) Visualisation of the dataset; (9) Analysis of the datasets in the context of the platform (integration with other datasets, description analyses, correlations, etc.) (section 4).

A total of 11 algorithms have been developed analysing some of the descriptors (section 4). For each algorithm, the authors have identified the problem the algorithm aims to solve, the data used by the algorithm, the formulas used and some proposals for the future use of the algorithms. The algorithms

process data to obtain ready-to-use information in the form of reports which are automatically displayed in the map of the prototype. This information is linked to a given geolocation (point) on the map or within the reporting section of the platform.

During the development of the prototyping phase several activities of engagement with new and potential data providers have been held. The authors are active members of the Apimondia BeeXML WG and maintain a permanent contact with this WG. For this reason, the platform was presented and discussed in several meetings of the WG. The platform has also been presented to the EUBP, and contributions from the partners were considered in the platform. ECPA, EFSA and BeeLife mainly contributed data to this phase. The engagement activities favoured the involvement of further candidate data providers whose data could not be integrated yet, due to a lack of time and resources. These data providers wanted to contribute mainly with non-invasive monitoring devices. Additionally, several datasets have been made available to the team, but they could not be integrated yet. Further work must be undertaken to integrate them in the future.

Finally, a tutorial and several promotional videos have been produced to show the various features of the web interface of the prototype platform. Similarly, the engagement activities carried out during the project are summarised.

Table of contents

Abstract.....	1
Summary	3
1. Introduction.....	8
1.1. Background and Terms of Reference as provided by EFSA.....	8
2. EUBP Prototype Platform: description of the web interface content and access	10
2.1. Welcoming page - The interactive map.....	11
2.2. Blog describing news and activities related to the platform	12
2.3. Data providers	14
2.4. Projects providing data.....	20
2.5. Descriptors	20
2.6. Reporting (i.e. data reports, usability, platform analytical statistics)	22
2.7. Data provider metadata management	23
2.8. Data Error Report Alarm	24
3. Data.....	24
3.1. Description of the imported data	24
3.2. Description of data providers, data owners and data users	27
4. Methodologies	28
4.1. Data integration procedure	28
4.2. Data validation.....	29
4.3. Data model.....	29
4.4. Primary features of the proposed Data model	36
4.4.1. Using polymorphism to save individual data points.....	36
4.4.2. Using polymorphism to save location data	37
4.5. Individual data models explained	37
4.5.1. Metadata.....	37
4.5.1.1. Project model	37
4.5.1.2. Provider model	37
4.5.1.3. Descriptor model.....	37
4.5.1.4. DescriptorCategory model	38
4.5.2. Location <i>Metadata</i>	38
4.5.2.1. Country model	38
4.5.2.2. Region model	38
4.5.2.3. LAU model.....	38
4.5.2.4. Post model	38
4.5.3. Origin <i>Metadata</i>	39
4.5.3.1. Methodology model.....	39
4.5.3.2. MethodologyCategory model.....	39
4.5.3.3. Device model.....	39
4.5.3.4. DeviceCategory model.....	39
4.5.3.5. DeviceVendors model	39
4.5.3.6. Publication model.....	39
4.5.4. Reference libraries	39
4.5.4.1. CurrencyReference.....	40
4.5.4.2. LandUseReference	40
4.5.4.3. PesticideReference	40
4.5.4.4. PesticidesTypeReference	40
4.5.4.5. PollenOriginReference	40
4.5.4.6. CropSpeciesReference	41
4.5.4.7. CropPollinationDependencyReference.....	41

4.5.5. Data models representing the data included into the EUBP Prototype Platform	41
4.5.5.1. Log model	41
4.5.5.2. Data model	41
4.5.5.3. DataModel model	41
4.5.6. Minimum requirements for a valid data point	43
4.5.6.1. Database structure	44
4.6. Data algorithms	44
4.6.1. Daily colony production	44
4.6.1.1. Identified problem and proposal	44
4.6.1.2. Used data points	45
4.6.1.3. Formulas	45
4.6.1.4. Future use	45
4.6.2. Colony production period detection	45
4.6.2.1. Identified problem and proposal	45
4.6.2.2. Used data points	46
4.6.2.3. Formulas	46
4.6.2.4. Future use	46
4.6.3. Colony consumption during the winter	46
4.6.3.1. Identified problem and proposal	46
4.6.3.2. Used data points	46
4.6.3.3. Formulas	47
4.6.3.4. Future use	47
4.6.4. Metabolic resting state of the colony	47
4.6.4.1. Identified problem and proposal	47
4.6.4.2. Used data points	47
4.6.4.3. Formulas	47
4.6.4.4. Future use	48
4.6.5. Sensor statistics on apiary, district, region, and country level	48
4.6.5.1. Identified problem and proposal	48
4.6.5.2. Used data points	48
4.6.5.3. Formulas	48
4.6.5.4. Future uses	49
4.6.6. Available flight time to bees	49
4.6.6.1. Identified problem and proposal	49
4.6.6.2. Used data points	49
4.6.6.3. Formulas	49
4.6.6.4. Future uses	50
4.6.7. Average colony production based on honey production and beehive statistics	50
4.6.7.1. Identified problem and proposal	50
4.6.7.2. Used data points	50
4.6.7.3. Formulas	50
4.6.7.4. Future uses	50
4.6.8. Cumulative honey production	50
4.6.8.1. Identified problem and proposal	50
4.6.8.2. Used data points	51
4.6.8.3. Formulas	51
4.6.8.4. Future uses	51
4.6.9. Colony management detection	51
4.6.9.1. Identified problem and proposal	51
4.6.9.2. Used data points	51
4.6.9.3. Formulas	52

4.6.9.4. Future uses	53
4.6.10. Economic value of pollination	53
4.6.10.1. Identified problem and proposal	53
4.6.10.2. Used data points	53
4.6.10.3. Formulas	53
4.6.10.4. Future uses	57
4.6.11. Beekeeping economic review	57
4.6.11.1. Identified problem and proposal	57
4.6.11.2. Used data points	57
4.6.11.3. Formulas	57
4.6.11.4. Future uses	59
5. Engagement and communication activities	59
6. Conclusions	62
7. Recommendations	62
References	64
Glossary	66
Abbreviations	67
Appendix A – Information on the data providers and datasets available in the EUBP prototype platform	68
Appendix B – Detailed description of the data models for the different Data types considered in the EUBP Prototype Platform	75
Appendix C – Detailed description of dataset import process	98
Appendix D – Pseudocodes of the different data algorithms included in the EUBP Prototype Platform	108

1. Introduction

1.1. Background and Terms of Reference as provided by EFSA

The EU Bee Partnership (EUBP) was established to promote stakeholders' engagement on harmonised and interoperable data exchange (i.e. data gathering and sharing) in the area of bee health and bee-keeping (EFSA 2018). BeeLife has supported the EUBP by setting a Proof of Concept (PoC) for a data/information exchange platform. This PoC is based on the Bee Hub¹ initiated under the umbrella of the Internet of Bees (IoBee) project (io-bee.eu²), a EU-funded Fast-Track-to-Innovation project, to be completed by April 2020. IoBee developed new technologies to improve the monitoring of bees and pollinators (including bee counters for hives and optoelectronic sensors to count pollinator density and diversity) and predictive models for relevant environmental factors such as plant phenology.

The Bee Hub PoC integrates different types of data (1) colony weight and in-hive temperature with digital sensors, (2) climatic with weather stations, (3) Varroa counts reported by beekeepers and (4) honeybee colony winter mortality from beekeepers' surveys. After the end of IoBee in April, 2020 the PoC needs further development to integrate all relevant information, knowledge and data to be collected by, and exchanged among, stakeholders.

The Bee Hub integrates the advances made by the group working on standardisation of bee-related data (BeeXML³), considers other proposals (e.g. the data model and the web application developed by BEEP⁴), and offers a concrete opportunity for interoperable data exchange in the area of bee health, also supported by the EUBP and EFSA. Through the EFSA MUSTB project, in particular on the collection of data coming from the field (EFSA, 2017) on various aspects related to bee health (i.e. pesticides application, SSD2 laboratory analyses including data on infectious agents, chemical residues, sugar concentrations and pollen identification, land cover and structure, hive measurements related to the identification of the specific colony on which measurements are made, colony management and inspections), data models were defined to collect and store data in a harmonised and standardised way.

Through this procurement, EFSA aims to ensure the development of a prototype platform, a user-friendly window on the state of bees in real-time for field practitioners such as beekeepers and farmers, as well as a more direct insight to current conditions and challenges for researchers.

This call is based on EFSA's 2020-2022 draft Work Programme for grants and operational procurements as presented in Annex IX of the draft Programming Document 2020 – 2022, available on the EFSA's website (EFSA, 2019).

This contract/grant was awarded by EFSA to implement and expand the existing Bee Hub into a prototype data platform for stakeholders involved in maintaining healthy stocks of honey bee colonies and sustainable beekeeping in Europe.

This development should include the following tasks:

- 1) Identify 2-5 new data sources;
- 2) Expand the data model and data storage of the existing Bee Hub platform in order to host data from the new sources to be added;
- 3) Include all the possible data from the new sources according to the expanded data model;

¹ <http://bee-hub.org>

² <http://io-bee.eu>

³ <http://beexml.org>

⁴ <https://beep.nl/>

- 4) Validate the quality of the data integrated and establish procedures for data validation;
- 5) Develop further the current visualisation and analysis capacity of the Bee Hub platform to include the new added data sources and reporting;
- 6) Improve the usability and backend stability for data processing.

Contractor/Beneficiary: BeeLife European Beekeeping Coordination.

Contract title: EU Bee Partnership Prototype Platform for collecting and exchanging standardised data on bee health and beekeeping based on the Bee Hub Proof of concept.

Contract/Grant number: NP/EFSA/SCER/2020/01.

The above tasks will be covered under the following work packages (WPs):

WP1. Data integration & acquisition: Identification of data that are made accessible to the contractor and the Bee Hub, favouring public databases (Eurostat), published datasets (scientific publications/technical reports, etc.), data resulting from sample analyses of monitoring efforts of BeeLife members and networks, data from EUBP members, or integration of further data from non-invasive monitoring hive systems. The data included will be verified for its quality before integration with a minimum set of quality criteria to ensure that the platform can be used for scientific analysis (e.g. completeness of the dataset; to be discussed and agreed with EFSA). If a new data source creates regularly new data, it will be synchronized with the Bee Hub in specific intervals. For this, it requires the availability of an API (Automatic Programme Interface). For static sources like Excel spreadsheets or published databases, the data will be imported and processed to an internal data model. For any new datatype a new display logic is built, and all metadata is available to be managed by administrators.

WP2. Data processing: Develop data processing algorithms for the data already integrated in the PoC and the new data sources. Develop algorithms visualising the outcome of the data processing on the platform. Algorithms will be developed with the help of bee experts to show examples of processing both a single or a combination of descriptors.

WP3. Validation and quality control: ensure the quality of the data integrated into the platform so that fully operational analytics algorithms and reporting systems can be achieved by the end of the prototyping. The algorithms developed under WP2 receive here additional testing iterations and validation. Furthermore, quality check methods are thoroughly documented to contribute to data standardisation. For validation purposes, additional datasets (namely non-invasive colony monitoring) need to be included, which will contribute to the complementation of WP1. The inclusion of these datasets requires programming for automatising their integration into the system (API development).

WP4. Prototype platform: Develop a user-friendly web interface showing an interactive map with different factors related to bee health (Figure 1). It should include the following 7 features:

- 1) Blog describing news and activities related to the platform
- 2) Data providers
- 3) Projects providing data
- 4) Data (metadata, statistics and links with the BeeXML advances and activities)
- 5) Reporting (i.e. data reports, usability, administration statistics)
- 6) Data provider metadata management
- 7) Data Error Report Alarm

WP5. Engage with new and potential data providers: During the development of the prototype, contact with the Apimondia BeeXML WG and EUBP. Contact with other data sources including private firms, academia, public authorities, and beekeeping and farmers associations.

WP6. Communicate and disseminate on the prototype platform: At the end of the project, a tutorial and video will be produced to show the various features of the web interface of the prototype platform.

■ **Factors inherent to the bees**

■ **External factors**

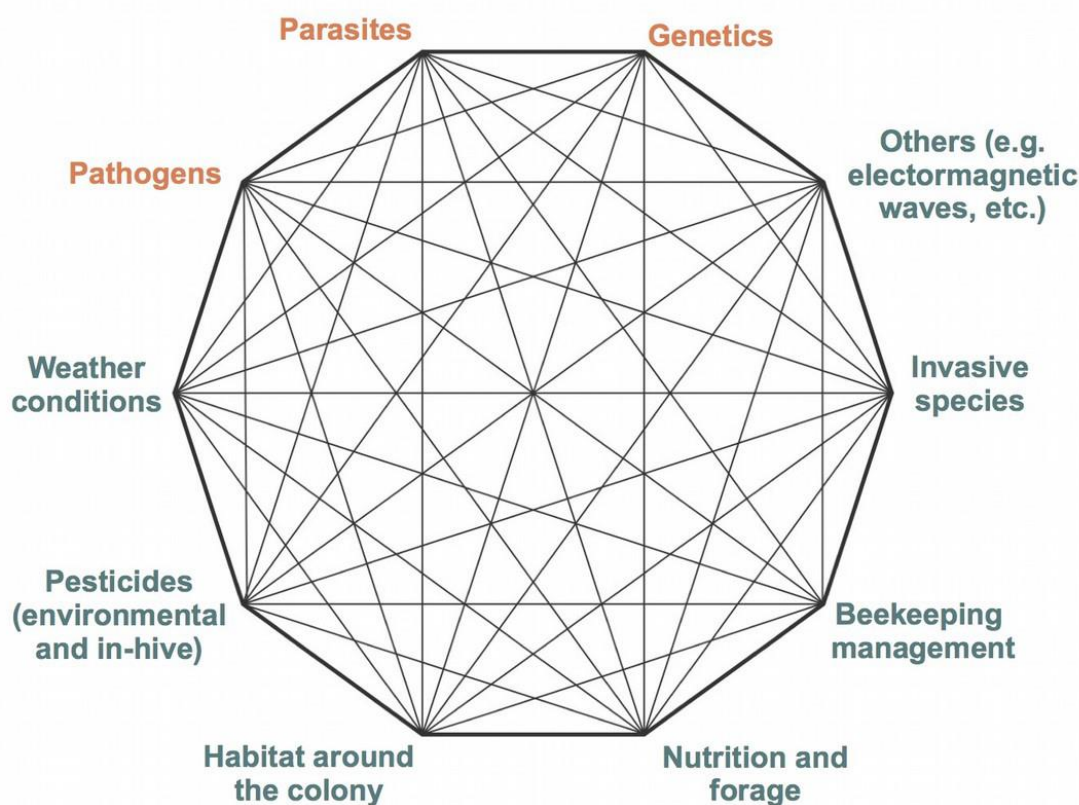


Figure 1: Schematic of the different factors affecting bee health. Source: Simon-Delso 2017

Up to date, mainly honey bee data has been included with the exception of the pollination dependency of crops. However, the long-term objective is to integrate information related to pollinators in general. Furthermore, among the first reports performed during the prototyping phase, information about the economic value of crop pollination has been included. This topic is horizontal to all pollinating species, and not just honey bees.

2. EUBP Prototype Platform: description of the web interface content and access

During the prototyping phase, the web interface has been improved in format and content. The web interface includes the features described hereafter, which are accessible via the link <https://bee-ppp.eu>. By accessing the link through a browser, the visitor/user of the web application will have access to the present developmental status of the Platform.

2.1. Welcoming page - The interactive map

The web application of the EUBP Prototype Platform welcomes the visitor with a map and the list of parameters that can be visualised in the map, for they are integrated into the database (Figure 2). By selecting any parameter on the right, the data available is displayed. A selection bar has been integrated on the left bottom corner of the map so that the visitor can choose what to visualise in the cases where the database contains data for several years and geographical levels for a given parameter. Every parameter for which there is a visualisation is associated with a legend at the bottom right of the map indicating the gradient of feasible results.

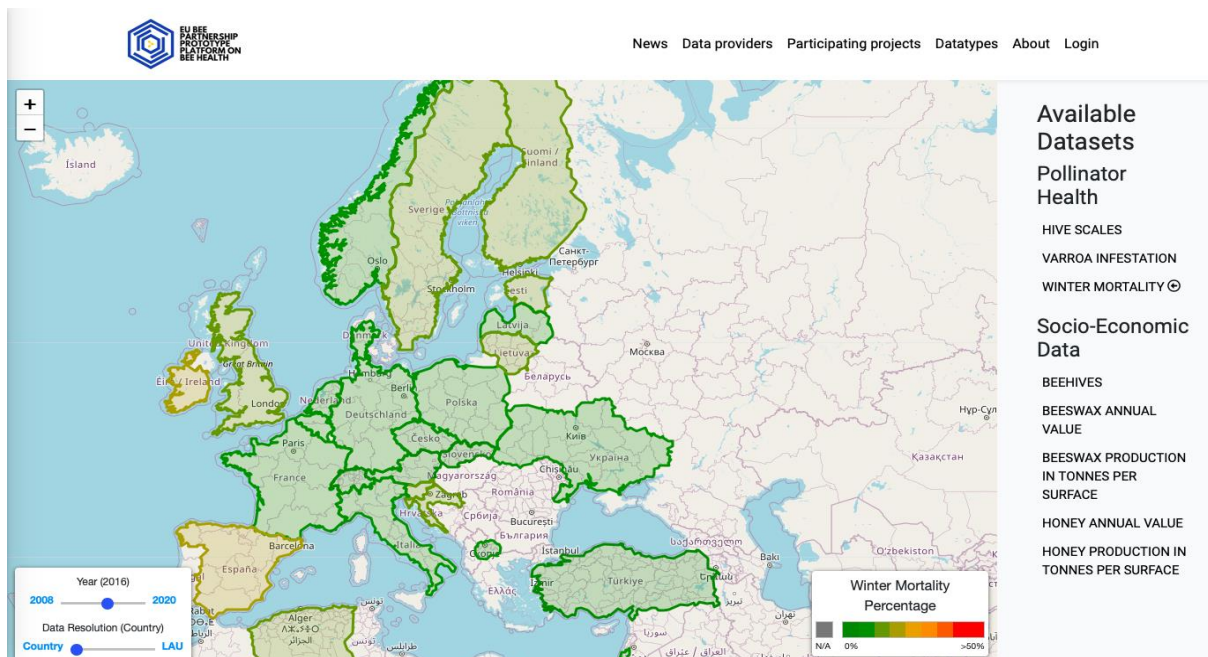


Figure 2: Display of one of the datasets on the welcoming page of the EUBP Prototype Platform

The map uses a colour code for visualising the results, but for colour-blinds, it also displays the aggregated data or results when the pointer of the mouse passes over a specific region.

It is also possible to click on top of a point or a region displaying results. When this is done, a pop-up window appears showing more information about the data displayed, including the value of the data point, important metadata like the location and year the data relates to, the participating project(s) contributing to create the data and the data provider(s) (Figure 3). In the future, as the database is further developed, an alternative display could be envisaged like showing all the data available for every geographical point.

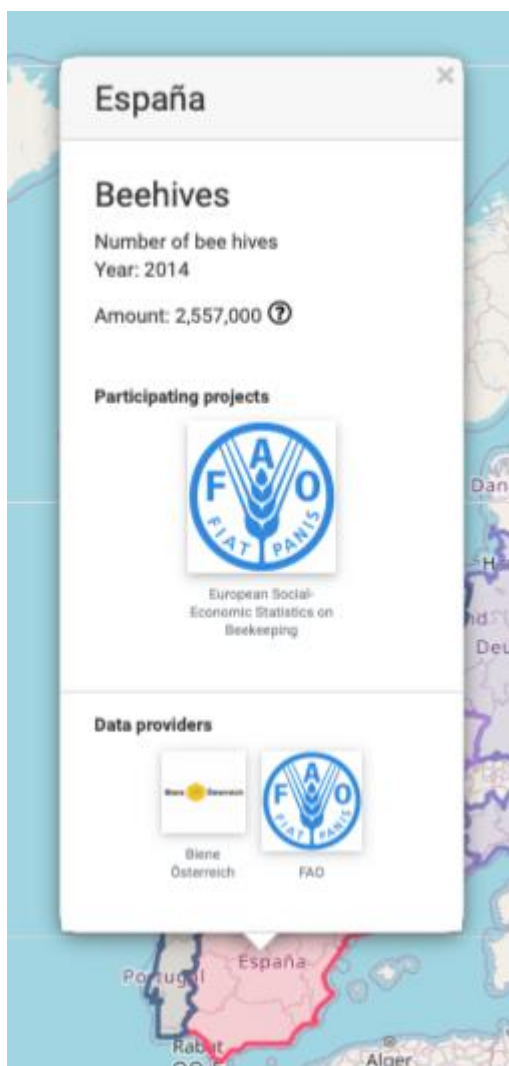


Figure 3: Display of the pop-up window with information of data, year, location, Data provider and Project

2.2. Blog describing news and activities related to the platform

A second feature of the platform includes a blog to inform and update the community about the activities related to the platform. So far, the blog contains a limited number of entries due to the fact that public communication about the platform was not one of the objectives of the prototyping phase.

The blog entries are displayed with a visual, a short text and the date of publication (Figure 4). So far, only English versions of the text are available for the blog and the platform in general. Each blog entry includes a button leading to another site that contains the full text and materials (Figure 5).

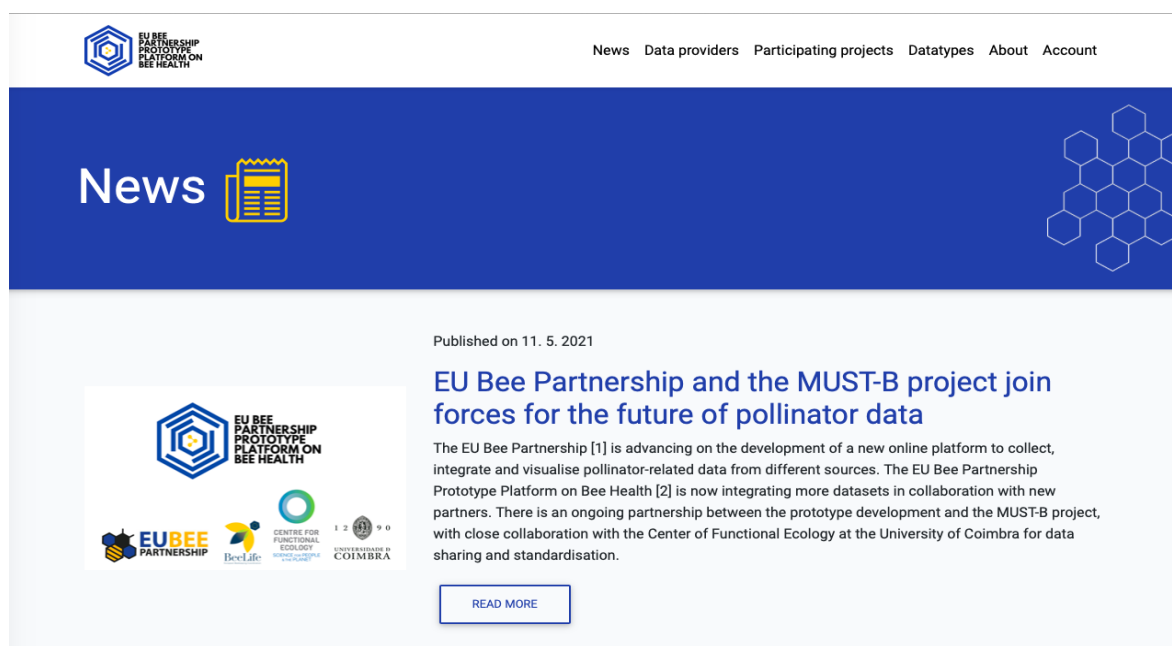


Figure 4: Display of the Blog feature of the EUBP Prototype Platform



Figure 5: Display of a blog sample once the web visitor clicks on the button “Read more”

2.3. Data providers

The data providers section is one of the pillars of the platform, which does not own the data. Data providers agree to share their data with the platform, but they keep on being the data owners. Data shared with the platform cannot be shared or made public unless the data providers agree on it. For this reason and many others like traceability, in case of questions related to the shared data, the data provider should be contacted directly. It is important to document the data sources. In doing so, not only data providers gain visibility for their efforts in data collection, but also anyone can contact them if the data providers allow it.

Data providers, including their picture/logo (if they agree), name and a button that provides access to further details about the provider are displayed in this section. In case the sources of data are scientific publications, data providers are identified by a "Research" icon (Figure 6). By clicking on the button, the visitor accesses a site in which more information is given about the data provider, including the name of the providers, a description of their nature and activities, contact details and website (if any), and the projects they are contributing data with (Figure 7). In the future, the datatypes of the data shared by each data provider could as well be integrated within the information from the individual providers.

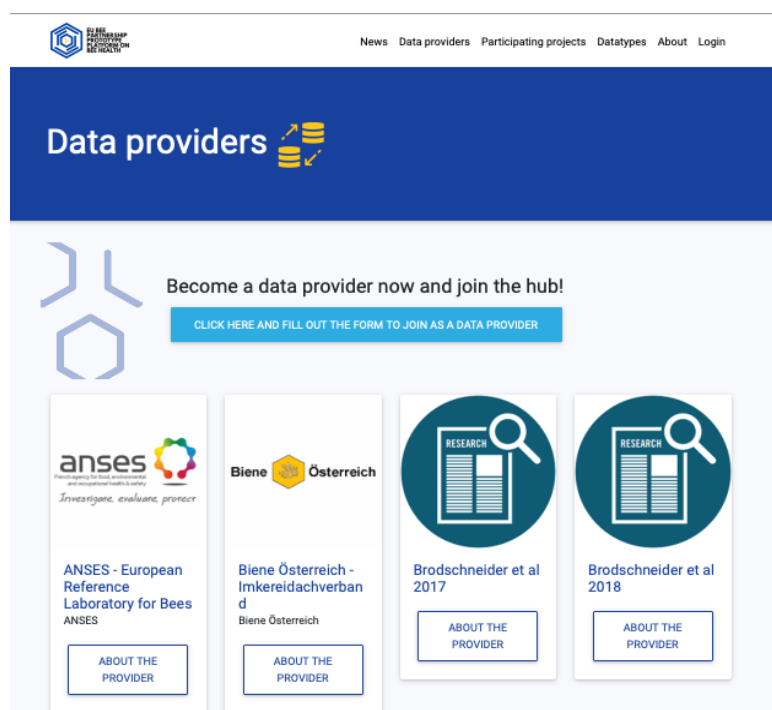


Figure 6: Display of the data providers section on the EUBP Prototype Platform

The visitor has the option to read details about each of the data providers and go back to the data provider's section by clicking on the "Data provider" hyperlink on the top of the website. Should the visitor want to go back to the Home page (the map and selection of parameters) he/she can also do it by clicking on the "Home" hyperlink at the top of the website.

During this prototyping phase, automatization of the procedure to become a data provider has been achieved. For this purpose, a button has been added in this section leading to the procedure to become a data provider and join the hub. In the future, data providers can automatically start the procedure of contacting the platform and sharing their data without contacting BeeLife and manually sign a Data Sharing Agreement, which is the current procedure. A virtual procedure has been established with the objective of (1) gathering the Data providers, Projects and Data metadata, (2) Agree with the Terms and Conditions of the data sharing signed between the individual data provider and BeeLife European Beekeeping Coordination establishing the conditions of the sharing and use of the data) and (3) specify the data sharing options: e.g. "Raw data publicly available", "available under request" or "not available".



The screenshot displays the user interface of the EUBP Prototype Platform. At the top, there is a navigation bar with links: Home, Data providers, Participating projects, Datatypes, About, and Login. The main content area is titled "CARlasbl - Beekeeping Center of Research and Information". It features a logo for CARlasbl, a section titled "About the provider" with a detailed description of the organization, and a "Contact details" section with an email address and phone number. Below this, there is a "Participating projects" section with three project cards: "National Beekeeping Programme - Wallonie (BE)", "BeeWallonie", and "Project" (Experimental approach to unexplained mortality of bee colonies in Wallonia). Each card includes a logo and a "MORE INFORMATION" button.

Figure 7: Display of one example of data provider information

To become a data provider for the platform, the user is guided through a number of steps. The first one presents what the platform is, its objectives and thanks the user for its participation. By pressing a button, the user starts the process (Figure 8).

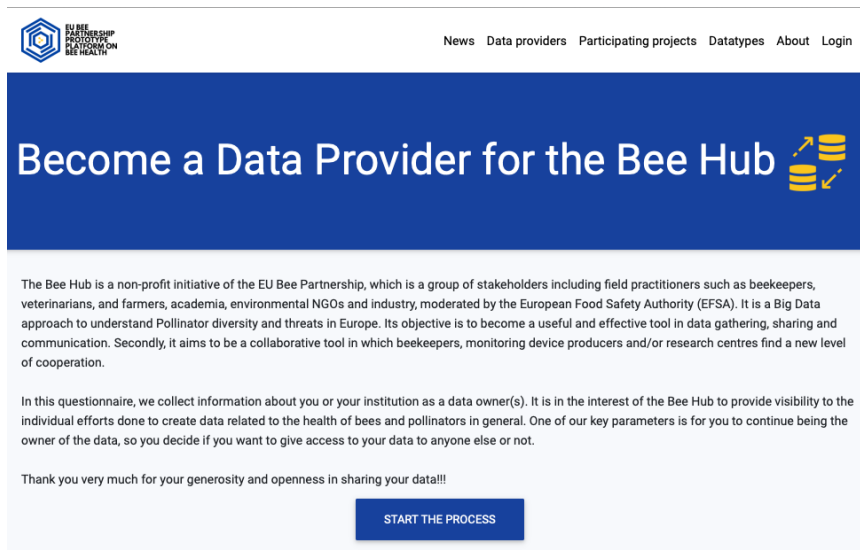


Figure 8: Display of the feature “Become a Data Provider for the Bee Hub” to become a data provider

The system today considers that the data provider shares either data that he/she has gained through participation in a project in which a collaboration exists among several institutions or that is solely owned by he/she. In the first case, it is considered that the data provider (co-)owns the data created through a project and the first information requested is linked to it. During the prototyping phase, only the first procedure has been developed. The second case will need to be established in the future (Figure 9).

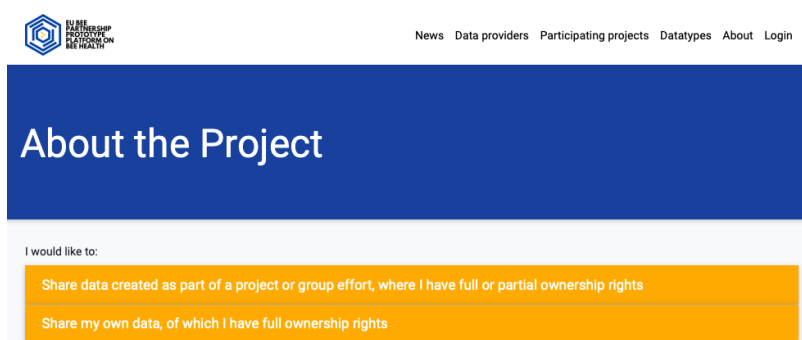


Figure 9: Display of the first step to become a data provider

Several details about the Project that generated the data are requested firstly (Figure 10). It includes information like the title (mandatory information) and acronym of the project, description, duration, its logo (if applicable) and information that will be publicly displayed on the platform. It is the intention that in the future, the operative platform will allow revealing or making an inventory of the different projects at the international, national and local level generating data related to pollinators.

The next step in the process aims at characterising the type of data that will be shared with the platform. A multiple-choice format has been chosen for this part to ease the user registration, standardise the input and reduce errors to a maximum extent. The system allows the possibility to introduce descriptors that are not proposed in the multiple-choice. The list of descriptors can be adapted in the future if need be (Figure 11).

Share data created as part of a project or group effort, where I have full or partial ownership rights

Project Name

Type in the name of the project or working group you participate in

Project Acronym

If the project has a short acronym, type it here

Project Description

B

I

S

L

¶

¶¶

<>

≡

≡

≡

≡

🔗

↶

↷

Write a description about the project here. You can use simple formatting to improve the appearance of the description.

Start of the project

Select date

📅


End of the project

Select date

📅

Note: fill out if the dates are relevant to your data. Please fill out the date in the form year-month-day with leading zeroes, for example 2020-04-20.

Upload Logo



UPLOAD A FILE

Figure 10: Display of the questions gathering data related to the Project that generated the data with the potential to be shared with the platform

About the Data

What kind of data are you willing to share

Smart Hives related data

- | | |
|---|-------------------------------------|
| <input type="checkbox"/> In-hive temperature | <input type="checkbox"/> CO2 |
| <input type="checkbox"/> In-hive relative humidity | <input type="checkbox"/> Sounds |
| <input type="checkbox"/> Weight | <input type="checkbox"/> Vibrations |
| <input type="checkbox"/> Bee counts at the entrance of the hive | <input type="checkbox"/> Odors |
| <input type="checkbox"/> External temperature | |
| <input type="checkbox"/> External relative humidity | |

Other

Please specify other

Stress factors related data

- | | |
|---|--|
| <input type="checkbox"/> Pesticide residues | <input type="checkbox"/> Weather information |
| <input type="checkbox"/> Pathogen loads | <input type="checkbox"/> Landscape information (CORINE Land cover, satellite images, crops, etc) |
| <input type="checkbox"/> Varroa infestation level | <input type="checkbox"/> Genetics |
| <input type="checkbox"/> Nutritional information | <input type="checkbox"/> Invasive species |

Other

Please specify other

Figure 11: Display of the questions gathering data related to the Data that will be shared with the platform

Once the data provider has chosen the type of data to be shared with the platform, he/she will be requested to specify the way in which the data sharing will happen. There are four options: (1) as a single file or archive (up to 10 MB), (2) as a link to a remote repository or file, (3) as a remote API⁵ connection, (4) or using other formats (Figure 12). The data provider will be requested to determine the access he/she wants to give to his/her data. Three options are proposed: (1) Open data (raw data can be openly shared and downloaded), (2) Restricted access (raw data can only be accessed by contacting the Data provider directly), and (3) Closed access (raw data cannot be shared or accessed through the platform). Finally, the contact details of the person in charge of data integration is requested in case problems would appear during data integration into the platform (Figure 12).

The final step aims at gathering data about the data owner, who can be the same as the data provider or not. An example in which the data provider and the data owner are not the same person/entity is as follows: the MUST-B project has generated bee-related data, some of which has been integrated into the platform. This data has been sponsored by EFSA (who is the data owner), but has been provided by one of the researchers in charge of creating the data with the agreement of EFSA. The system allows for one or several data owners to be registered. The information provided here is publicly displayed as part of metadata information of the data. The provision of this information is not compulsory (Figure 13).

⁵ API Application Programming Interface

How will the data be provided

☐ As a single file or archive, up to 10MB

☐ As a link to a remote repository or file

☐ Via a remote API connection

☐ Other formats

Would you like to share your raw data with other users?

☐ Open: My raw data can be openly shared and downloaded by other users without restrictions.

☐ Restricted: My raw data can only be shared by direct contact.

☐ Closed: My raw data can not be shared.

Data integration contact

Contact person concerning data integration

Email address of the data integration contact person

This email address will not be published publicly but will be passed to a developer.

Figure 12: Display of the questions linked to the format of data transfer and sharing of raw data

The possibility to join the platform as data provider is permanently displayed on the other features of the website (i.e. Home, News, Participating Projects, etc.) at the bottom of the web application, together with the link of the platform with other social media platforms.

About the ownership

Who is/are the owner-s of the data?

These fields will be publicly displayed as part of the meta information that describes your data. All of these fields are optional but recommended.

Data Owner 1

Full name

Short name or acronym

Address, post number and country

Legal registry number

Required for non-persons only

Project Description

Figure 13: Display of the questions related to data ownership

2.4. Projects providing data

A great number of projects at the international, national and local level generate data related to pollinators. This feature aims at providing visibility to each of these projects that contribute data to this platform, which aims to be a referent in terms of pollinator related data. This part of the website summarises the projects which shared data within the platform. For each participating project, a logo/picture (if any), its title, acronym and a button to read more information are displayed. Should the project not have any logo, a generic "Project" icon is displayed (Figure 14).

By clicking on the button to learn more information about the project, the visitor is taken to a new page in which the description of the project can be found as well as the data providers that participated in the project and co-own the data (with the possibility to read more about it/them) and the descriptors that the project contributed with (with the possibility to read more about these Descriptors).

On the top of the site, the visitor can find the hyperlink to return to the Participating Projects page and to the Home page.

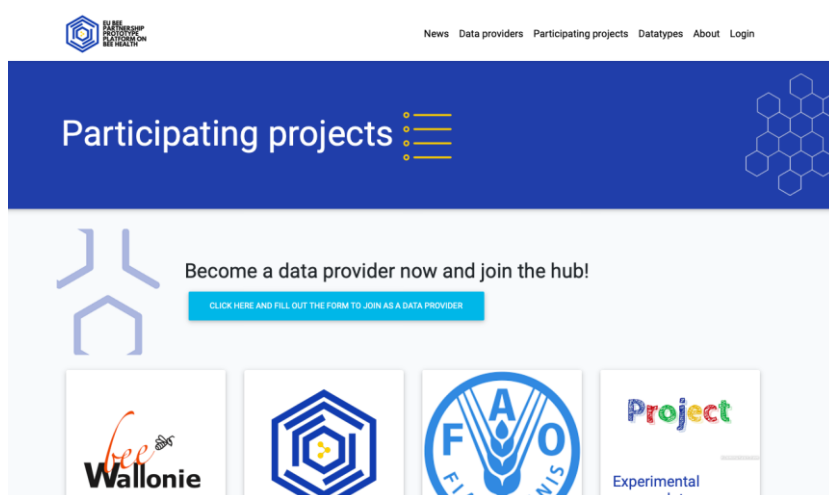


Figure 14: Display of the Participating projects features of the EUBP Prototype Platform

The prototyping phase has contributed to generating new datasets, thanks to the algorithms developed (see section 4.6). For the moment, these new datasets have been linked to their own projects and data owners. In the future, when more datasets are available, different datasets from different projects and data owners can be used for data analyses or data processing. Should the outcome create a new dataset, it will be linked to a Participating Project called "EU Bee Partnership Prototype Platform" (or whatever the platform will be named in the future), having EFSA as data provider. A procedure will need to be developed in succeeding steps to ensure the traceability of the data providers and projects that contributed to the processed data.

2.5. Descriptors

A total of 574 descriptors were integrated so far in the Prototype Platform. However, only 66 have been made public to ease the reading of the web application. The choice of the descriptors to be displayed publicly has been based on the consideration of meaningful parameters to potential visitors. Descriptors

have been organised by groups to ease their accessibility and user-friendliness. Each data type is identified by its name, category, and icon. By clicking on each, so far the most meaningful descriptors contained in the dataset, containing both raw data integrated and calculated data resulting from the implementation of algorithms (Figure 15) of the descriptors or on the button "More information" a new page appears in which detailed information is given related to the data type, including metadata like the name, definition, units, icon and the information and link to the projects contributing to the database; statistics like the total amount of data points, the amount of unique data points and the number of related projects (Figure 16). Unique data points are values of variables that the platform registers only once despite being repeated one or more times (e.g. during the day, the temperature of 25°C can be registered at different hours of the day. The platform will register the unique data point of 25°C). In the future, thanks to the close collaboration with the beeXML group⁶, the descriptor standards will be updated to meet the standards proposed by the group. EFSA proposed standards in Healthy-B (EFSA AHAW Panel, 2016) that can be taken into account in the further development of the platform.

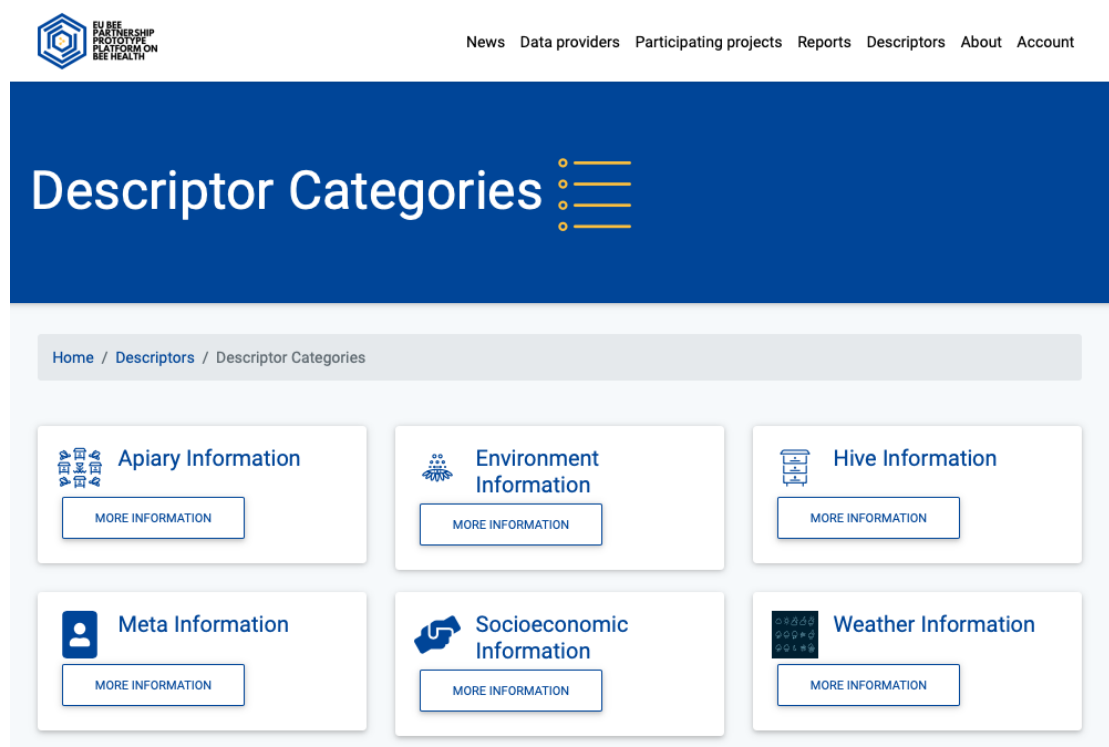


Figure 15: Display of the most relevant descriptor categories integrated into the database or calculated from the developed algorithms

⁶ <http://beexml.org>

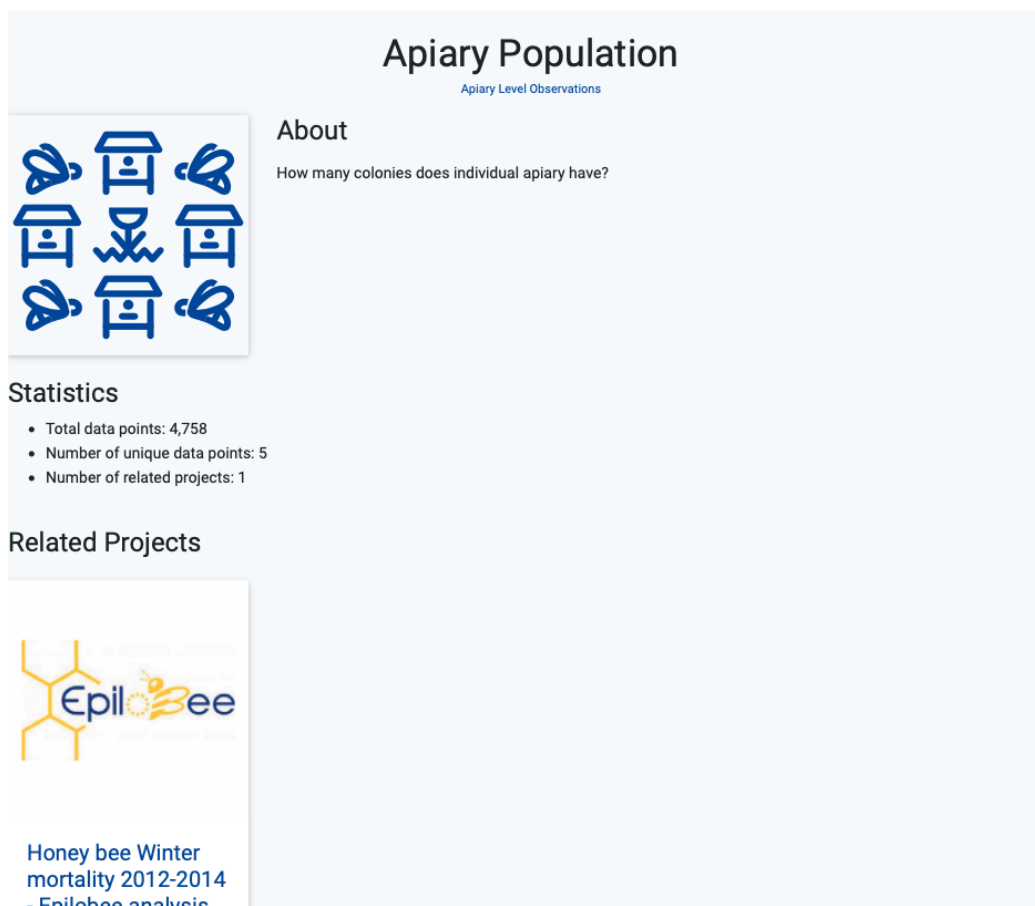


Figure 16: Display of an example of a descriptor included in the EUBP Prototype Platform

2.6. Reporting (i.e. data reports, usability, platform analytical statistics)

The Reporting section includes *ad hoc* reports made on the basis of the data integrated in the platform database. Automatic reports are the result of the implementation of some of the algorithms developed within the prototyping phase and are displayed on the interactive map. A couple of *ad hoc* reports have been produced as examples showing the potential reporting that could be developed from the data included into the database (Figure 17).

In theory, reports can have different formats, from traditional scientific reports following a typical scientific structure (i.e. title, authors, affiliation, abstract, key words, introduction, methodology, results, conclusions, and references), to graphs, maps or infographics. Future work on the platform should be devoted to automatise reporting, while keeping the possibility to develop *ad hoc* reports and infographics.



Figure 17: Display of the Report section of the EUBP Prototype Platform with one example report

2.7. Data provider metadata management

One of the main objectives of the prototyping phase has been to automatise the gathering of data providers' information. For this reason, it was key to develop a data provider management section where data providers of the EUBP Prototype Platform can now modify on their own the metadata included regarding projects, data provided, or add new data. They can also administer the accessibility to their data as they want and modify the password. All this can be done by accessing the "account" feature (Figure 18).

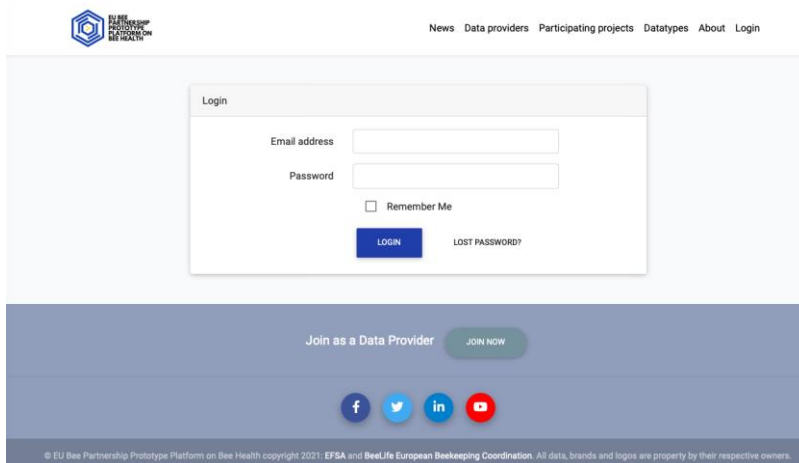


Figure 18: Display of the data provider metadata management of the EUBP Prototype Platform

2.8. Data Error Report Alarm

As the platform grows, so does the complexity and amount of code required to run it. To ease the management of it by administrators and developers, a simple version of error reporting has been implemented with an external tool called "Sentry⁷". Because of the nature of application and server errors, they have to run on a separate service, web application or server, in order to properly log and handle them. As the platform is still in the prototype stage and the number of expected reports will not be very high, we can use the "Developer" tier of the service. In this, it is included:

- Logging any kind of errors or reports, coming from the production server;
- Categorising, grouping and managing the reports as issues, which can be further analysed, assigned and managed;
- Email and push notifications for any severe or urgent level of events;
- Support both front end (JavaScript) and back end (PHP) errors.

A EUBP Prototype Platform Administrator Manual has been created to guide the administrator through the platform interface (see Annex 1-EUBP Prototype Platform Administrator Manual).

3. Data

3.1. Description of the imported data

A number of datasets were integrated in the platform during the PoC phase at the beginning of the procurement process (Appendix A), namely hive scales readout every 2 hours (i.e. every two hours one data point), hive sensor data (in-hive temperature) and weather sensor data (i.e. temperature, relative humidity and rainfall) from Belgium, regional varroa infestation data from Austria, colonies autumn and spring counts, colony mortality count by natural disasters, colony mortality count by queen problems and colony mortality count by empty/dead from Belgium and winter mortality from the Epilobee project

⁷ <https://sentry.io/welcome/>

at EU level. The table in Appendix A shows the phase of implementation and the date on which the data was integrated.

During the development of the prototyping phase, additional datasets were integrated into the EUBP Platform. For each dataset, the data supplier is described as well as the project that enabled the data acquisition, if known. When data has been obtained from a publication, the first author of the publication and publication bibliography are included in the data providers. The additional datasets integrated into the platform are included in Table 1.

The PoC included a preliminary data taxonomy, grouping datasets that belong to the same “nature”. As far as possible, we have followed the proposal of the EFSA in HEALTHY-B (EFSA AHAW Panel, 2016), with some modifications in the terminology that we consider more accessible to the general public and reducing the hierarchical grouping. In the future, we aim to get in line with the taxonomy used in the B-GOOD project⁸. The proposed data grouping includes a number of parameters linked to the colony attributes, such as (1) demography and biology (incl. colony development measured as the number of bees and brood, bee/insect populations, etc.); (2) automatic monitoring (done with scales and sensors, etc.); (3) bee health (incl. diseases, parasites, winter mortality, etc.); and (4) breeding (including information about breeds and races preferentially based on DNA analysis, etc.). Other data groups are linked to external drivers, such as (5) ecology (interaction between bees and plants (pollination, nutrition); (6) weather conditions (including temperature, rainfall, solar radiation, etc.); (7) landscape (including land use, CORINE land cover data, crops, prairies, forests, plant phenology, pollen botanical origin, etc.); (8) environmental pollutants that come in contact with bees (including residues of pesticides, biocides, veterinary products, in environmental matrices or industrial emissions, etc.). Finally, a third group is linked to the colony outputs, but goes beyond them, referred to as (9) socioeconomic aspects (including data about the beekeeping sector, production, crop pollination dependence, beekeeping/crop/pollination management, etc.).

Table 1: Table summarising the datasets integrated so far in the EUBP Prototype Platform and the related data groups, suppliers and projects

Data Group ^(a)	Dataset	Supplier or Source	Project
AUTOMATIC MONITORING	Hive scales readout every 2 hours and hive sensors data (in-hive temperature) (Belgium)	CARI	National Beekeeping Programmes
WEATHER CONDITIONS	weather sensors data (i.e. temperature, relative humidity and rainfall) (Belgium)	CARI	National Beekeeping Programmes
GEOLOCALISATION	Population and surface size for most local administrative units (LAU) and NUTS regions levels 0 – 3 of European countries.	World Geodetic System 1984 and Nomenclature of territorial units for statistics within EU (Eurostat)	-
BEE HEALTH	Regional varroa infestation alerts data (Austria)	Biene Österreich and Ländliches Fortbildungsinstitut Österreich	VarroaAlert

⁸ <https://b-good-project.eu>

BEE HEALTH	Colonies autumn and spring counts, Colony mortality count by natural disasters, Colony mortality count by queen problems and Colony mortality count by empty/Dead (Belgium)	CARI, HoneyBeeValley, CRA-W	BeeWallonie (CARI, CRA-W), National Beekeeping Programmes (HoneyBee Valley)
BEE HEALTH	Winter mortality (EU)	ANSES	Epilobee
AUTOMATIC MONITORING	Hive scales weight readout every hour, hive management and sampling information, hive sensors data (i.e. in-hive temperature, relative in-hive humidity) (Portugal and Denmark)	EFSA (Portuguese data provided via collaboration with University of Coimbra, Danish data is in the pipeline)	MUST-B
WEATHER CONDITIONS	weather sensors data (i.e. temperature, relative humidity, rainfall, dew point, wind speed, wind gust and solar radiation) (Portugal and Denmark)	EFSA (Portuguese data provided via collaboration with University of Coimbra, Danish data is in the pipeline)	MUST-B
DEMOGRAPHY AND BIOLOGY	Colony developmental data (Portugal and Denmark)	EFSA (Portuguese data provided via collaboration with University of Coimbra, Danish data is in the pipeline)	MUST-B
BEE HEALTH	Mean winter loss rate in percentages, mean winter loss lower and upper limits (CI 95%), number of correspondents and total number of alive colonies (Worldwide data)	Brodschneider et al., 2018, 2016; Gray et al., 2020, 2019; van der Zee et al., 2014, 2012	COLOSS network
ENVIRONMENTAL POLLUTANTS	Pesticide content in honey bee collected pollen pellets (Belgium)	Simon-Delso et al., 2017	DEPAB
ECOLOGY	Botanical diversity in honey bee collected pollen pellets (Belgium)	Simon-Delso et al., 2017	DEPAB
LANDSCAPE	Land use in the 3 km surroundings of apiaries (Belgium)	Simon-Delso et al., 2017	DEPAB
ECOLOGY	Faunistic and floristic recordings of bee (<i>Apoidea</i>) visits on specific plant species including location and time (200 countries Worldwide).	Klein et al., 2007	ALARM Project
ECOLOGY	Pollination dependency of crops (200 countries Worldwide)	Klein et al., 2007	ALARM Project
BEE HEALTH	An eight-year survey of <i>Varroa destructor</i> infestation rates of honeybee	Rubinigg et al., 2021	

	colonies and the spatial dimension, temporal dimension and weather factors that impact these infestation rates (Austria)		
BEE HEALTH	Outbreaks of European Foulbrood and American Foulbrood (Belgium)	BeeLife ⁹	
BEE HEALTH	Outbreaks of European Foulbrood and American Foulbrood (Austria)	Biene Österreich ¹⁰	
DEMOGRAPHY AND BIOLOGY	Colony developmental data (Germany, not published)	CORTEVA ¹¹	
AUTOMATIC MONITORING	Hive scales weight readout every hour, hive management and sampling information, hive sensors data (i.e. in-hive temperature, relative in-hive humidity)(Germany, not published)	CORTEVA	
DEMOGRAPHY AND BIOLOGY	Colony developmental data (Austria, not published)	dr. Mag. rer. nat. Michael Rubinigg	
SOCIO-ECONOMIC DATA	In the EU27 Member States: Number of Beehives per country; Crops requiring pollination, their production and value; Volume of honey production; Value of honey production; Volume of honey trade; Volume of wax production; Value of wax trade; Value of wax production (EU).	FAOSTAT	

–: no information available.

3.2. Description of data providers, data owners and data users

The prototype platform identifies three ways of interaction with the data: data providers, data owners and data users. Data providers are the persons or institutions that directly interact with the platform to provide the data. Data owners are the persons or institutions that own the data. In most occasions data providers are the same as data owners, although this does not always need to be the case. A practical example is the case of the MUST-B project. The owner of this data is EFSA, while the data provider has been one of the researchers from the University of Coimbra in charge of generating the data. In this

⁹ <http://bee-life.eu>

¹⁰ <https://www.biene-oesterreich.at>

¹¹ <https://www.corteva.com>

case, EFSA has agreed on sharing data with the platform and has favoured the researcher to build up a collaboration with the team developing the platform so the data is integrated.

A similar situation could exist for data that are automatically generated by non-invasive monitoring systems, beekeeping management applications, etc. In this case, the data owner is the person or institution owning the data, but the data provider is the company that, through an API, transfers the data to the platform. To date, the Prototype Platform considers these somehow legal differences and the information is registered as metadata. However, for technical purposes the platform considers data providers and owners as equal for the displaying and management of data. In the future, should there be a need, this could be modified.

Data users are the people that can have access to the data or information to generate reports or use the data for other purposes without necessarily having an account or without having contributed with data. So far, only the team developing the platform has been the data users; the data integration cannot be accessed yet through the platform (i.e. the platform does not have the feature to access and download the data). In the future, data providers will determine the sharing typology of their preferences and the platform will allow downloading in a standardised way the data that is publicly available. A system must be created in the next developmental phases of the platform to establish contacts between potential data users and data providers/owners choosing to have restricted access to their data (upon request).

4. Methodologies

4.1. Data integration procedure

The methodology for the integration of datasets into the platform follows a number of steps that are necessary to ensure their quality and usability. The formal steps are described hereafter. Each of the datasets is processed in the same way, and Appendix A shows the status of the different datasets in the process. The final aim would be to automatise as much as possible the data integration, which would suppress or shorten steps 1-4. During the prototyping phase, the procedure for a data provider to automatically upload his/her data into the platform was already established. The procedure is currently being tested by a small group of new data providers.

- 1) Identification of relevant datasets;
- 2) Establishment of contact with the data provider;
- 3) Drafting and signature of data sharing agreement, if needed;
- 4) Data acquisition;
- 5) Data and metadata description;
- 6) Data cleaning and preparation;
- 7) Import of dataset in the platform;
- 8) Visualisation of the dataset;
- 9) Analysis of the datasets in the context of the platform (integration with other datasets, description analyses, correlations, etc.).

The focus is on generating information from the data included in the platform which is useful for the targeted groups, namely field practitioners (such as beekeepers, farmers or veterinarians), the general public, institutions and decision-makers.

The algorithms included in the platform aim at transforming the data integrated into different shapes or groups:

- Aggregated data, which improves comparisons between individual areas and allows better statistical analysis. For example, winter mortality data from Belgium is available and registered by postal code. This information can be upscaled to the district, regional and country level;
- Processed data leading to ready-to-consume information, which can provide additional, previously unknown information. The algorithms presented hereafter belong to this type;
- Analytical metadata, which describes existing data and improves the context of it. These are mostly used in reporting to give a clear picture of the data that was used to get the results. One such use case is displaying the relevance of data by displaying the number of data points. This indicates the relevance of data based on the number of data points.

4.2. Data validation

Manual validation of datasets is performed in a three-tier process, which is documented in a validation protocol:

Tier I: Assessment of the technical correctness, integrity and consistency of the raw data.

Tier II: Import of the raw data into a MySQL database or a statistical software suite (*e.g.* SAS[®] Viya[®] Data Studio, Microsoft[®] Excel[®], or any other suitable software), followed by the generation of a data profile for each field of each table and comparison with initially defined assumptions on the data:

- Total number of records
- Data type
- Number and percentage of unique entries, NULL and blank values (all data types)
- Minimum and maximum value (all numeric data types)
- Mean, standard deviation, coefficient of variation (all numeric data types)
- Minimum and maximum length (only characters)

Tier III: Generation of a report, based on an initially defined goal and assumptions on the data:

- Performance of descriptive and predictive analyses on the dataset covering all initially defined goals.
- Discussion of the results, including literature review, examination of plausibility and accuracy.
- Documentation of concerns regarding data quality and any other potential issue (legal, ethics).

4.3. Data model

The EUBP Prototype Platform Data model has the objective to standardise the structure and organisation of the data collection that the platform will integrate and the relationships among these data.

This data model has been constructed during the PoC phase development of the platform (The Bee Hub¹²), by using four datasets as case studies. The present data model is the basis for data integration of other datasets. The additional datasets integrated during the prototype phase of the development of the platform served to further finetune the present data model.

The Data model was built from the ground up, due to the fact that there is no existing (at least to our knowledge) data model that implements all or most bee-related data, and openly integrates different sources of data. Based on previous experience both linked with bee-related projects and beyond, we

¹² <http://bee-hub.org>

consider the present implementation of the data model as the most relevant and adequate to the current use and should be flexible enough in the future. This data model was developed with consultation from the Apimondia BeeXML project (Apimondia WG on Bee Data¹³).

The EUBP Prototype Platform Data model includes a number of individual models and structures that render it fully operational for the datasets already integrated and paves the ground for scaling up in the addition of datasets. Figures 19-23 show the Unified Modelling Language (UML) diagram of the EUBP Prototype Platform Data model, its inheritance and relationships between individual models. It is separated into 6 parts (coloured):

- 1) Blue: Metadata for participating Projects, DataProviders, Descriptors, DescriptorCategorization and Management metadata.
- 2) Black: individual recorded logs in relation to the participating project with a collection of data points.
- 3) Purple: many-to-many relationship pivots.
- 4) Brown: location data based on Eurostat¹⁴ and GeoNames¹⁵.
- 5) Orange: data origin Metadata like data collection methodologies or sensor devices, units, etc.
- 6) Crimson red: the data point storage for various data types.
- 7) Gray: Reference database and reference data models

¹³ <http://beexml.org>

¹⁴ <https://ec.europa.eu/eurostat/web/qisco/geodata/reference-data/administrative-units-statistical-units/nuts>, <https://ec.europa.eu/eurostat/web/nuts/correspondence-tables/postcodes-and-nuts>

¹⁵ <https://www.geonames.org/>

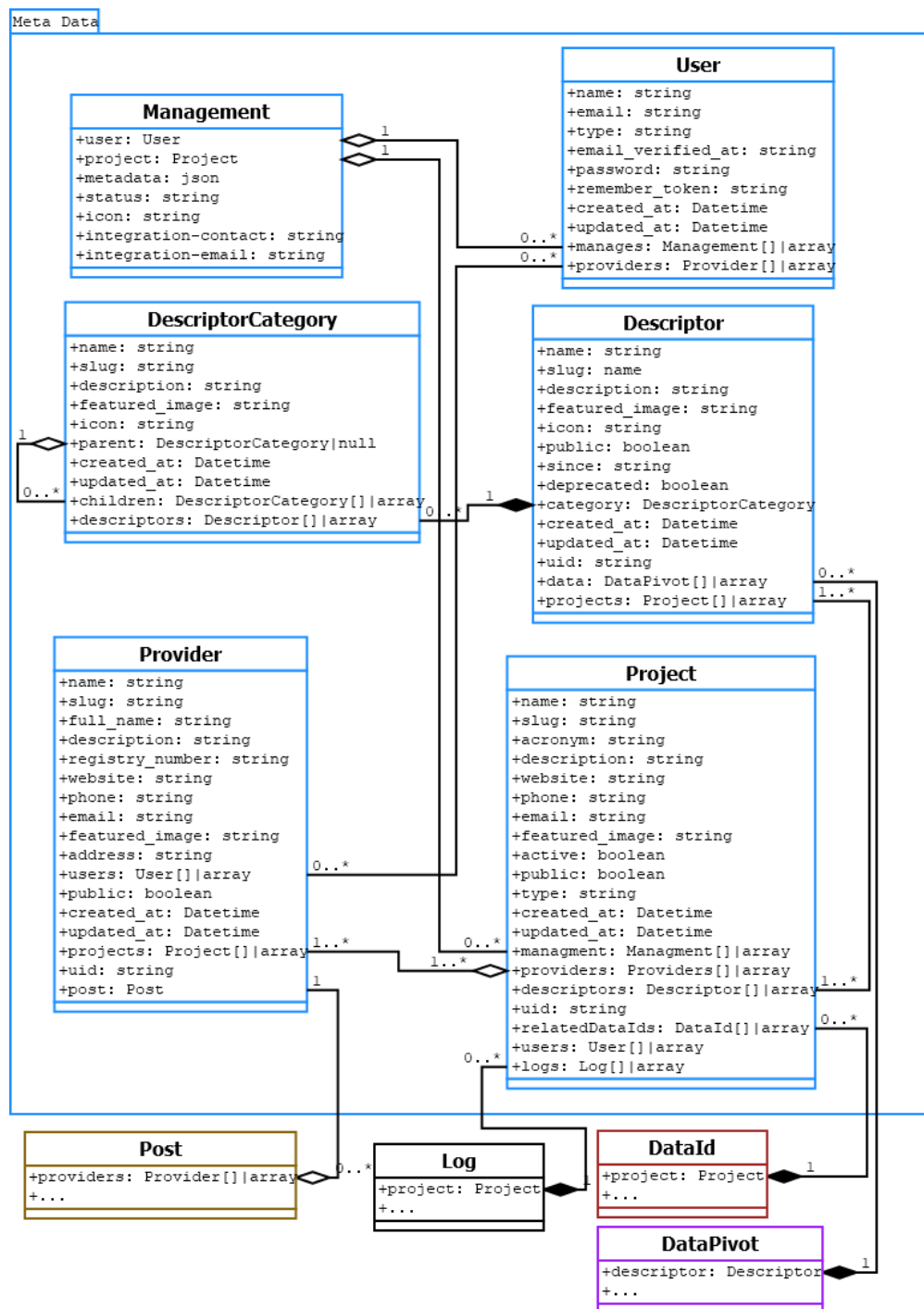


Figure 19: Data model UML diagram displays inheritance and relationships between individual models within the first (blue) metadata section

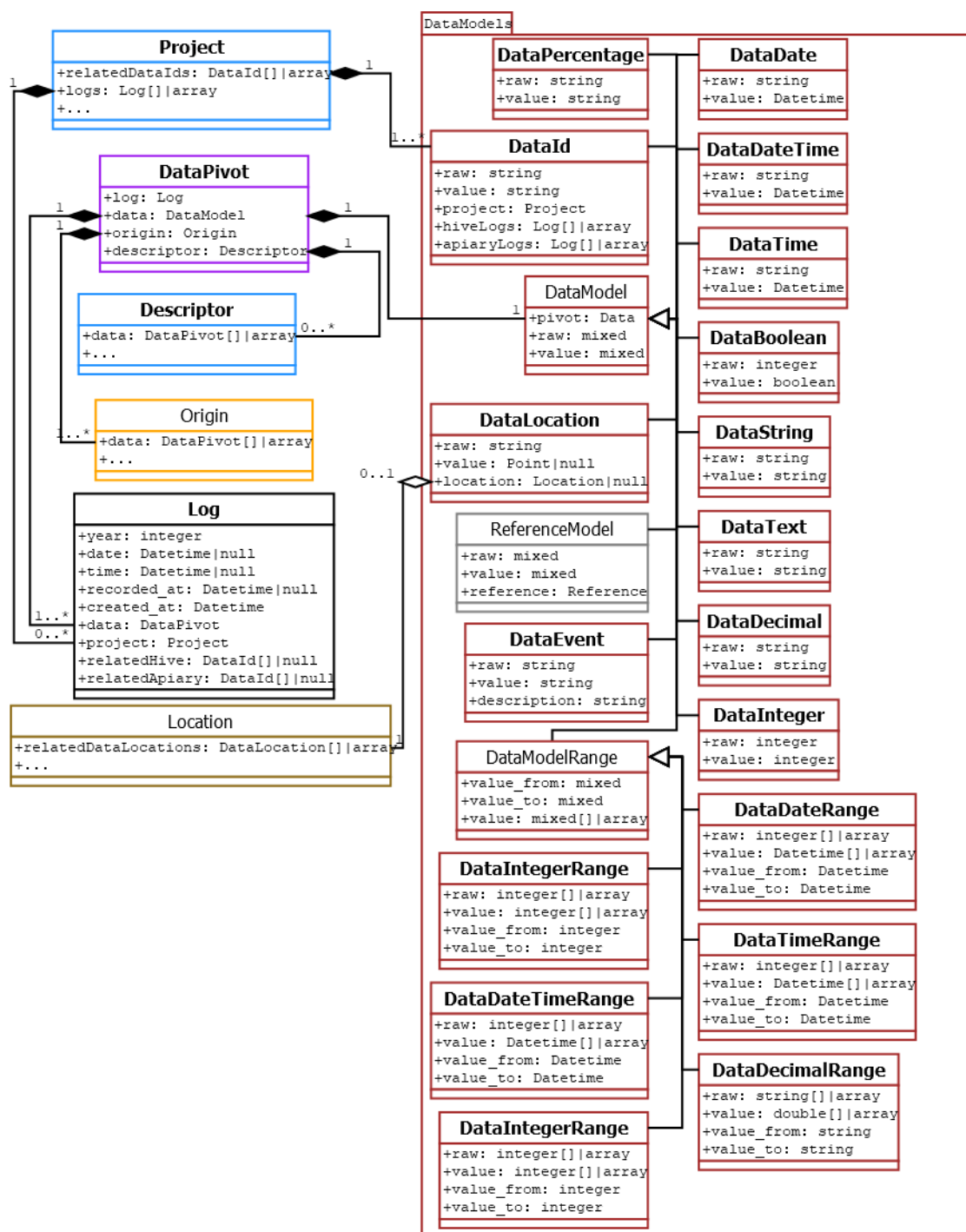


Figure 20: Data model UML diagram displays inheritance and relationships between individual models within the second (black), third (purple) and sixth (red) sections. These represent logs, data points and their relationship

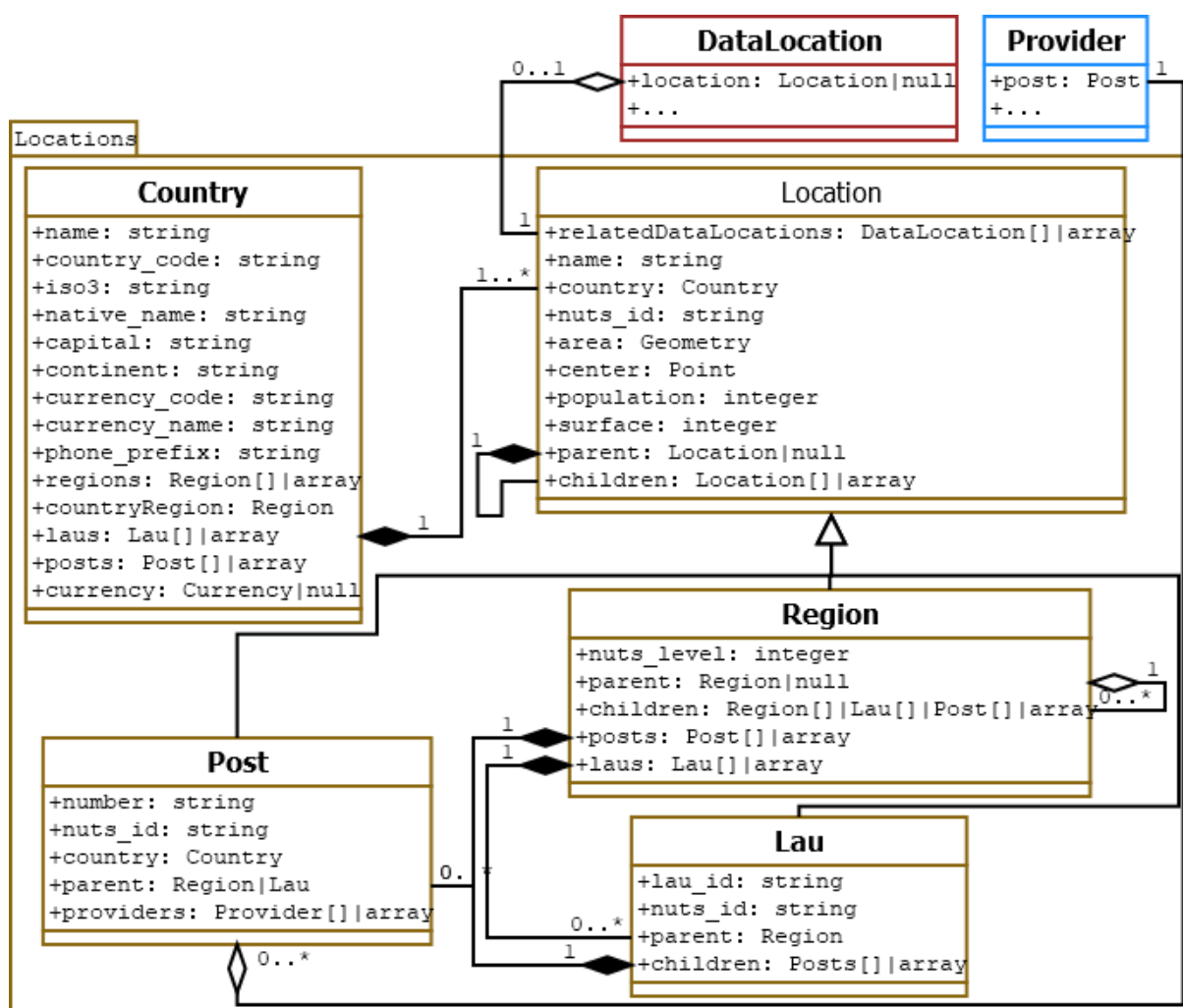


Figure 21: Data model UML diagram displays inheritance and relationships between individual models within the fourth (brown) data location section

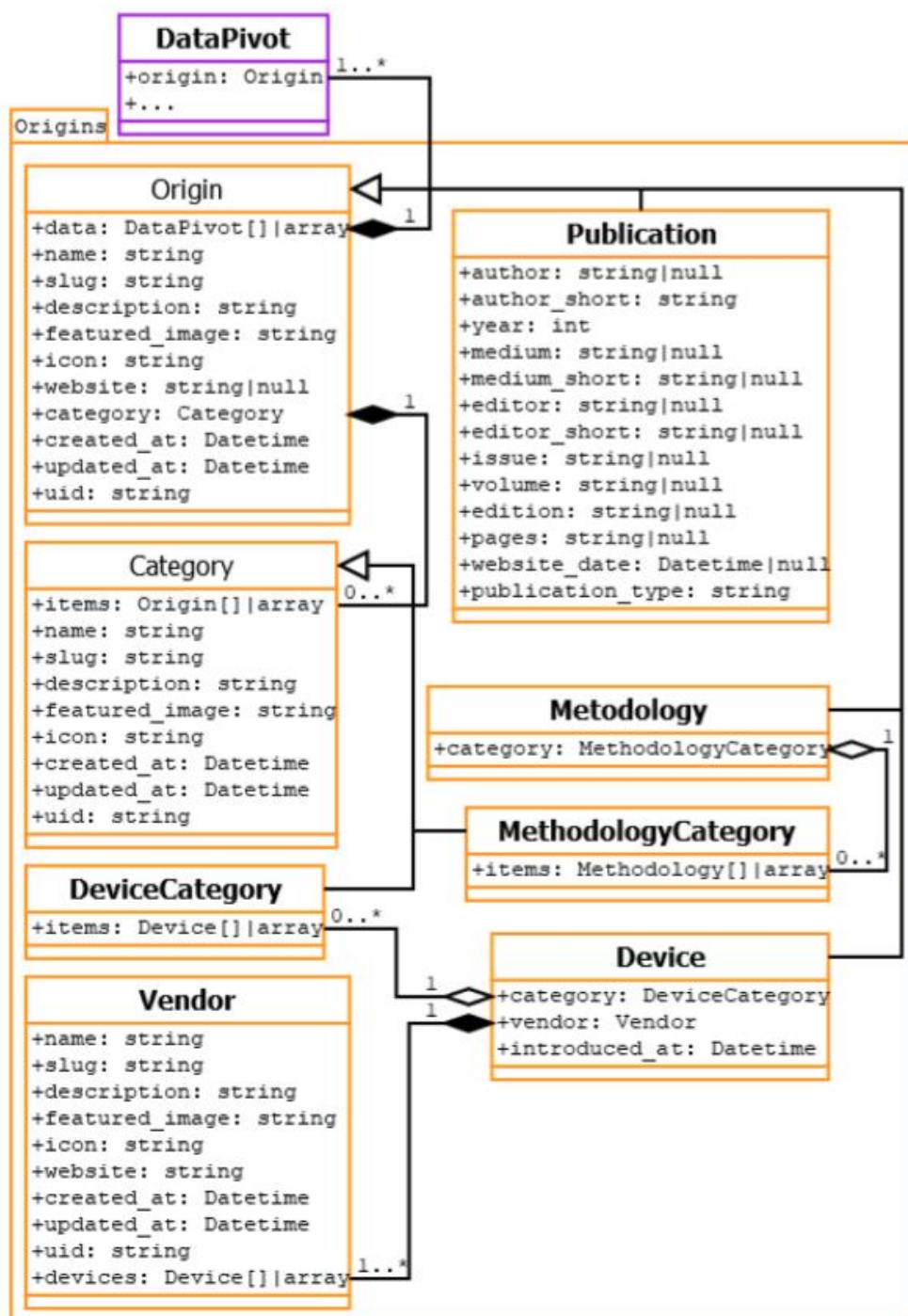


Figure 22: Data model UML diagram displays inheritance and relationships between individual models within the fifth (orange) data origin section

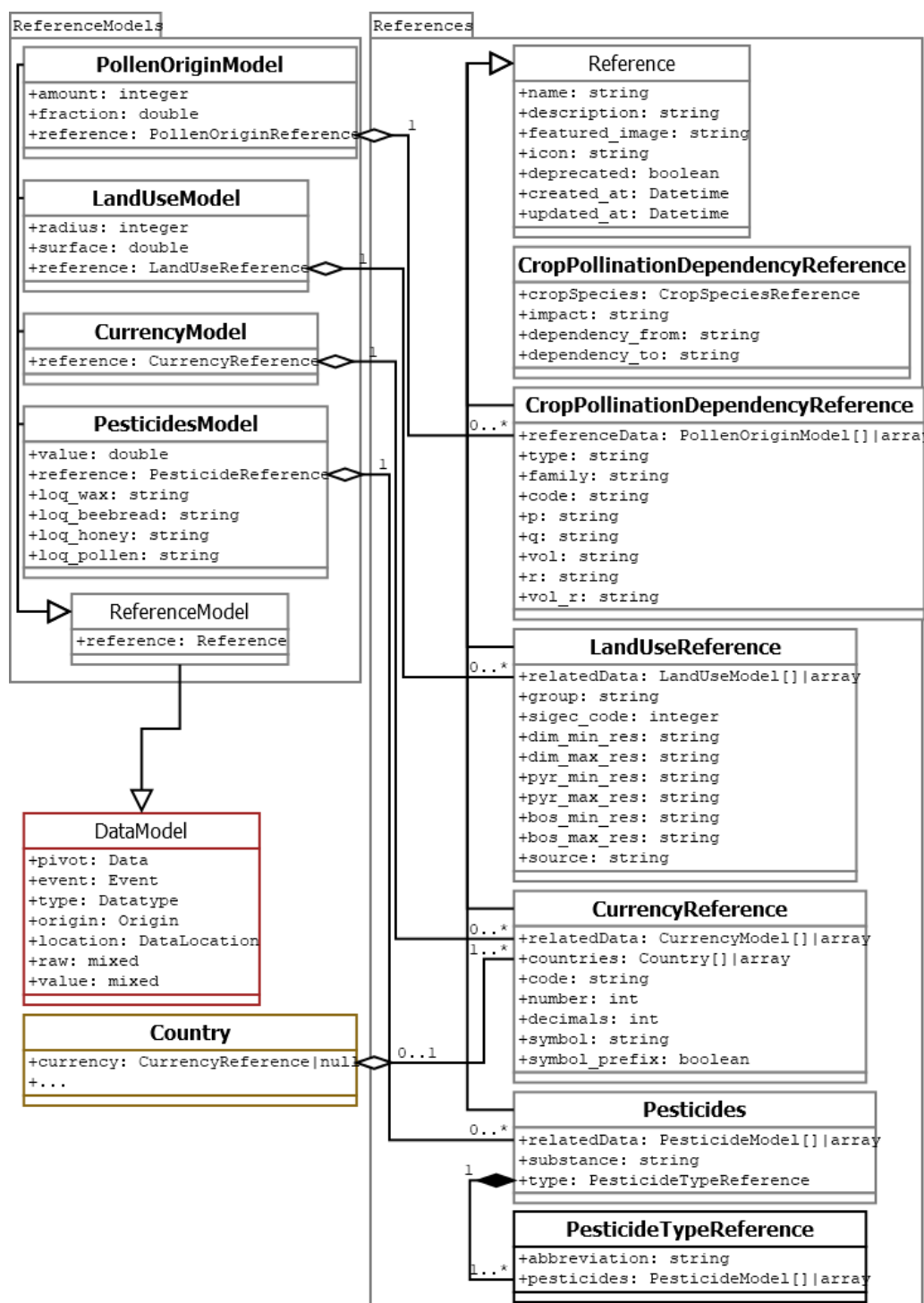


Figure 23: Data model UML diagram displays inheritance and relationships between individual models within the seventh (gray) reference data section

4.4. Primary features of the proposed Data model

4.4.1. Using polymorphism to save individual data points

In programming languages and type theory, polymorphism is the provision of a single interface to entities of different types or the use of a single symbol to represent multiple different types (Milner, 1978). In other words we allow a single model to relate to a variable sub-model. This is known as polymorphic relationships that allows us to return different sub-models based on available data, e.g. a Log model has always a related location, but among different data providers, the precision can greatly differ. This is why we allow the location to be WGS84¹⁶ coordinates and/or the NUTS Region¹⁷ (of any level) and/or the LAU Region¹⁸ and/or Postal code. Usually when we have coordinates available, we try to bind it to a location by using geolocation libraries. Currently, all Regions and some LAUs contain geographic boundaries.

All data in the platform is saved using polymorphic relationships, which means that depending on the type of the information stored, a different column in the database might be used. For example, when saving a full number (like number of beehives in a region), the `log_data_integers` table is used which contains an integer as value datatype. While saving real numbers (up to 20 decimal spaces) like hive scales weight data, the `log_data_decimals` table is used which contains decimal as value data type.

Advantages of separating data points into individual data types

By separating data points into individual data types we can optimize our database in several ways:

- Size: by saving data in appropriate raw form, we can greatly reduce the size of the database;
- Speed: the fastest way to access individual values is by using raw data types that are supported by the database;
- Extendibility: at any time, we can add more scalar or compound data types based on need, as they are described by the Descriptor;
- Repetition: many data points are repeatable, so we can save any different value only once, then reference them via Pivot. This enables saving disk size and memory while allowing simple data grouping;

Disadvantages of separating data points into individual data types

There are certain disadvantages in separating data points into individual data types:

- Complexity: to manage individual Types, certain code complexity is required, which reduces code readability and might introduce more, and harder to find errors;
- Performance: by separating data points it becomes necessary to perform queries to multiple tables to retrieve individual Log data. This increases response times and memory usage;

The advantages are believed to outweigh the disadvantages. Therefore, a number of measures have been set up to minimize the impact of the latter. To compensate for the increased complexity, the data model has been implemented and documented, carefully commenting every step of it. To compensate for performance, cache has been introduced, which greatly improves the response speed of the application. Another measure used is to pre-calculating algorithms and aggregates, which saves valuable loading time.

¹⁶ World Geodetic System 1984

¹⁷ Nomenclature of territorial units for statistics within EU

¹⁸ Local Administrative Unit

4.4.2. Using polymorphism to save location data

The precision of individual datasets in terms of location data may greatly vary. Therefore, different levels of location precision are supported. The lowest supported resolution is NUTS level 0 or country level, while the highest supported precision is WGS84 coordinates. The full list of location precision levels is as follows:

- 1) NUTS level 0 or country level
- 2) NUTS level 1, defining major socio-economic regions
- 3) NUTS level 2, defining small regions of the application of regional policies
- 4) NUTS level 3, defining small regions for specific diagnoses
- 5) LAU or local administrative units
- 6) Postal code
- 7) WGS84 coordinates, usually up to 2nd decimal precision, which approximates to 1.1km.

4.5. Individual data models explained

4.5.1. Metadata

The EUBP Prototype Platform emphasizes that most of the data is not owned by the platform managers but by the data Providers. That is why *Metadata* is an important part of the Data model, so the origin of the data can be traced back. This possibility will be updated in the future. Appendix B contains specific details about the attributes of each of the data models described in this section. All metadata models track in detail the history of changes and have the possibility to be removed by the owner or administrator.

4.5.1.1. Project model

Whenever a new dataset is included in the EUBP Prototype Platform, a new Project is created. The Project describes why, how, by whom and where the data has been created, edited, or deleted. It is uniquely identified using a slug¹⁹.

4.5.1.2. Provider model

The provider is representing institutions, working groups or individuals that are directly providing any kind of data to the platform. There can be several organisations, institutions or individuals participating in a data collection project and each organisation, institution or individual can participate in several projects. Data providers do not necessarily correspond to the actual owners of the data. A data provider can own the data, and if not, they need to have the agreement of the data owner to share the data. In that case, the data provider becomes a sort of data manager. Future developments of the platform will need to clarify these roles.

4.5.1.3. Descriptor model

Data can be of many different types and nature, e.g. Varroa infestation level, pollen contamination, number of beekeepers, hive weights, etc. It can be static or be created with a certain frequency, it can

¹⁹ A slug is a string of text with lowercase alphanumeric characters. It can be used as part of an URL, to link individual models on the web.

refer to a period of time (e.g. winter mortalities) or to a specific point in time (e.g. temperature). The descriptor model describes metadata of the data, i.e. how the data is stored in the database, in which unit and includes any required annotations.

4.5.1.4. DescriptorCategory model

Descriptor categories are organised based on the work developed by the BEEP team²⁰ and adapted to be used within the EUBP Prototype Platform. In its simplest form, it allows to group individual Descriptors, for example, the socio-economic data containing the pricing of honey and beeswax, which are each separate Descriptors, while they both belong to the “goods pricing” DescriptorCategory. In the future, this model might be merged with the Descriptors themselves, depending on the most efficient way to use descriptors.

4.5.2. Location Metadata

As previously mentioned, the location of the data point is very important for each dataset, given that the EUBP Prototype Platform envisages to display the data availability in a map. A known location allows us to make a statistical analysis and comparisons of different national, regional or local data. Location data is separated into 4 tables which contain geographical reference libraries from various sources (see below). Not all sub-data is available for all locations.

4.5.2.1. Country model

The countries’ table is populated from geonames.org, which contains a list of all countries included in ISO 3166²¹ countries list. The available data is English name, ISO and ISO3 country codes, Capital name, total area, total population, currency, continent, currency code, phone prefix number and language.

4.5.2.2. Region model

The list of regions is based on the Eurostat NUTS 2016 database. Additionally, we include Geographical data of each region of all levels, also available from Eurostat. In the future this model will require detailed documentation of the sources and document changes, as the NUTS database changes over time (the 2021 version is already available at the time of writing).

4.5.2.3. LAU model

Local Administrative Units (LAUs) is a list of administrative units defined within countries, hierarchically right under level 3 of NUTS regions. Similar to NUTS, the list of available LAUs is based on the Eurostat LAU 2019 definition, which itself is compatible with NUTS 2016. Also similar to NUTS, these can change over time and will require proper documentation in the future.

4.5.2.4. Post model

Postal codes, or internally “Post”, are so far the most precise administrative units handled by the EUBP Prototype Platform. Currently two sources are used to fill out the table: postal code to NUTS 2016 database by Eurostat, geographical information from Geonames.org. As with Regions and LAUs, these can change and will require proper documentation in the future.

²⁰ For more information about BEEP project, visit <https://beep.nl/home-english>.

²¹ <https://www.iso.org/fr/iso-3166-country-codes.html>

4.5.3. Origin Metadata

During the development of the proof of concept, it was clear that the same data can be collected using different methodologies. Additionally, different vendors of similar sensors can provide different results and contain different error margins. That is why, it is of key importance to include Origin *Metadata* for each value saved in our database during the development of the prototype. Currently Origin *Metadata* is separated into 3 types:

- 1) Methodology
- 2) Device
- 3) Publication

Each of them is organised within a category system, which might be merged with Descriptor categories (DescriptorCategory) in the future.

4.5.3.1. Methodology model

This model can describe in detail, how and which principles were used when a data point was collected. It also explains how the data was processed within the platform, whether it used statistical methods or algorithms. It can also contain information about the methodology of data collection, e.g. the COLOSS winter mortality data was collected using a questionnaire with extensive validation. In the future it might relate to an external publication like the COLOSS BeeBook (Dietemann et al. 2013a; 2013b) or HEALTHY-B standards (EFSA AHAW Panel, 2016).

4.5.3.2. MethodologyCategory model

The methodology category or internally "MethodologyCategory" helps us organise common methodologies, and to compare and extrapolate their results. For example, there are several methods to measure varroa infestation, but all would fit in the same category and could be theoretically compared.

4.5.3.3. Device model

In the last few years, a lot of new manufacturers and devices for bees and beekeeping monitoring emerged on the market. Each device collects data in its own way and may produce different results when comparing similar devices. This is the reason why it is important to determine the device that is generating the data.

4.5.3.4. DeviceCategory model

The device category, or internally "DeviceCategory" model, helps us organise common devices used for the collection of data from honey bee colonies, in order to compare and extrapolate their results, e.g. hive scale would be a DeviceCategory and a specific model of hive scale would be a Device.

4.5.3.5. DeviceVendors model

A single manufacturer might produce different devices or several versions of a single device. This is the reason why we introduce a *DeviceVendor* model.

4.5.3.6. Publication model

As a lot of information can be retrieved from publications, this model allows us to properly reference these.

4.5.4. Reference libraries

During development of the platform various reference libraries related to pollinators, socio-economic or taxonomical data started to be collected that can be used to further detail the data from within the platform, contextualise it or create some recalculating options. A reference library is a collection of data to reference the datasets with. For example, the annual figures available for human population (inhabitants) per country, the land area per country and region or the toxicological profile per pesticide active ingredient, which can be related to when performing calculations. Currently we have stored information of the following types of libraries.

4.5.4.1. CurrencyReference

Reference library with a list of all known currencies and a relation with countries using the currency. It also stores some additional metadata related to the currency like number of decimals, the symbol used to represent it, and if it is positioned before or after the value.

4.5.4.2. LandUseReference

Reference library for the agricultural land use types. It will provide references to various crops occupying different agricultural fields each year or the type of land use existing on a certain area and year (e.g. forest, urban, agriculture, etc.), using external identification libraries e.g. SIGeC²², CORINE²³ land cover, or Agridata²⁴ data. A first draft of reference library was uploaded with the data from Simon-Delso, et al. (2017) and sets the basis for future development of this Data model. For the completion of this model related to LandUse further datasets from different locations and years are needed. Only after adding further data can the model be fulfilled and become operative and usable.

4.5.4.3. PesticideReference

This reference library of pesticides contains names of pesticide active ingredients in English and French found in honey bee matrices. In the future, the toxicological and physico-chemical profiles of individual active ingredients should be included. The data was created thanks to the DEPAB project and was uploaded with the data from the publication Simon-Delso, et al. 2017.

4.5.4.4. PesticidesTypeReference

This is a reference library of the pesticides types, based on the mode of action of pesticides (insecticides, fungicides, herbicides, etc.) and it completes the previous one. The reference library was uploaded with the data from the publication Simon-Delso, et al. (2017). The reference library needs to be completed with a full list of pesticides active ingredients and their mode of action.

4.5.4.5. PollenOriginReference

Reference library representing botanical pollen types with information about their dimensions, prevalence and taxonomic classification. The reference library is so far based on the result of palynological analyses of pollen pellet analysis (microscopic identification) and was uploaded with the data from the publication Simon-Delso et al. (2017). For the completion of this model related to pollen further datasets from different locations, analytical methods, bee matrices are needed.

²² https://ec.europa.eu/info/food-farming-fisheries/key-policies/common-agricultural-policy/financing-cap/financial-assurance/managing-payments_fr

²³ <https://land.copernicus.eu/pan-european/corine-land-cover>

²⁴ <https://agridata.ec.europa.eu/extensions/iacs/iacs.html>

4.5.4.6. CropSpeciesReference

This is a reference library representing the different plant species cultivated worldwide, based on the FAOSTAT classification and the database GBIF (<https://www.gbif.org/fr/>)”

4.5.4.7. CropPollinationDependencyReference

This is a reference library of the dependency on pollination of the yield of different crops, based on the classification provided in Klein et al. 2017.

4.5.5. Data models representing the data included into the EUBP Prototype Platform

Raw data included in the EUBP Prototype Platform is saved within the following models, because it enables high extensibility and scalability.

Data itself is separated into 3 parts:

- Log
- Data
- DataModel

4.5.5.1. Log model

When any datum from data providers is recorded, it always starts with a Log within the database. The Log indicates when the data was recorded. Note: most logs contain the related apiary and hive identifiers, so currently for the purpose of optimisation we include them here as well to speed up the platform. In the future, this will be removed as this information is already stored in the data model.

4.5.5.2. Data model

For each data point, the EUBP Prototype Platform saves a value in *DataModel* and a value in *DataPivot* model. With the help of this pivot, we also include Descriptor and Origin models to further describe the data point.

4.5.5.3. DataModel model

DataModel is an abstract model representing a data point value. Using polymorphism we support the following data types:

Scalar Types:

- Boolean – a true or false value;
- Integer – a whole number plus all the negatives of natural numbers;
- Percentage – decimal value with up to 5 decimal numbers;
- Decimal – a real number;
- String – a sequence of characters, up to 191 characters long;
- Text – a sequence of characters, up to 16.777.215 characters;
- Date – a date in format YYYY-MM-DD (e.g. 2020-09-30);
- Time – a time value in format hh:mm (e.g. 13:04);
- Datetime – a date and time in format YYYY-MM-DD hh:mm (e.g. 2020-09-30 13:04);

Note: all dates and times are stored in UTC-0.

Compound data types, which are made up of several individual Scalar data types:

- Integer range – 2 integer values, which create a from-to range;
- Decimal range – 2 decimal values, which create a from-to range;
- Location – contains at least 1 item (or both):
 - 1) Reference to Postal code, local administrative unit LAU or NUTS region;
 - 2) WGS84 GPS coordinates rounded to 4th decimal number;
- ID – a unique data identifier in relation to data source, a sequence of characters of up to 191 characters long. It usually represents Hive or Apiary Identity but can be any other ID references in the source as well;
- Event - a logged monitoring or action done to an apiary or hive;
- Date range - a from - to date in format YYYY-MM-DD;
- Time range - a from - to time value in format hh:mm;
- Datetime range - a from - to date and time in format YYYY-MM-DD hh:mm;

Reference data types, which reference our internal database

- CurrencyModel reference
- LandUseModel reference
- PesticideModel reference
- PollenModel reference
- CurrencyModel reference

These are the individual data types and their attributes, in addition to what is described above. Further attributes are described for each data type in Appendix B.

DataBoolean - Stores simple true or false values.

DataInteger - Stores numerical data.

DataPercentage - Stores an increase or decrease in percentages.

DataDecimal - Stores real numbers.

DataString - Stores short text strings.

DataText - Stores longer text strings.

DataDate - Stores Dates.

DataTime - Stores Time of day or elapsed time.

DataDatetime - Stores Date with time of day.

DataIntegerRange - Stores a range of integers. One of the values can be null to represent infinity.

DataDecimalRange - Stores a range of decimals. One of the values can be null to represent infinity.

DataLocation - Stores GPS coordinates or reference to a Location.

DataID - It allows for identification of separate collection points. They are unique for each Project. A single group of data points can have multiple IDs assigned to them (for example Hive ID, Apiary ID).

DataEvent - When an observation or an interaction is done within an apiary or a hive, an Event is logged. There are several event types available, additionally, observation notes and value changes (with a unit like kg or litre) can be saved. In the future a list of all possible events will be compiled to properly sort and group similar events.

DataDateRange - Stores a range (from - to) of dates.

DataTimeRange - Stores a range (from - to) Time of day.

DataDatetimeRange - Stores a range (from - to) Date with time of day.

CurrencyModel - Stores a currency value like annual value of honey in a country with a link to the CurrencyReference model.

LandUseModel - The LandUse Model describes information about the use of land in terms of the crops that are planted, their dimensions (in hectares), location and time. The model included in the prototyping phase is very limited, for it has been created from the data available at the publication Simon-Delso et al. (2017), which covers only the crops planted in certain areas of the Walloon region (Belgium) in 2011. The typology of crops used may therefore not be standardised. Further efforts of data collection need to be carried out to achieve a more complete model. The LandUseModel has a link with reference to the LandUseReference model.

PesticideModel - The Pesticide Model describes the attributes linked to pesticide analyses performed on, *a priori*, pollinator matrices. As for the LandUse one, it is preliminary and future developments should go towards the operationalisation of the SSD2 model recommendations proposed by EFSA, 2017. At this stage, the model includes, including parameters like: name of pesticide active ingredient, matrix analysed, date of sample collection, analytical method, and the individual Limits of Quantification (LOQs) of different matrices (i.e. wax, beebread and honey, pollen pellets). Limit Of Detection (LOD) and LOQ attributes represent the detection and quantification thresholds of the chemical analyses of different pollutants (pesticides, heavy metals, combustion products, etc.) in their respective units (e.g. mg/kg). The PesticideModel has a link with reference to the PesticidReference model.

PollenOriginModel - The Pollen origin model describes the attributes linked to the plants from which bees collected either pollen or nectar. So far the model has been based on the information available at Simon-Delso et al. (2017), which was based on palynological analyses of pollen pellets. Other parameters included in the model are the date of the sample collection, the name of the pollen family or genus, sometimes the morphological description of the pollen grains, the amount of grains counted and the representativity of individual pollen grains on the sample. In the future the model could be inspired by the SSD2 model proposed by EFSA, 2017, with necessary adaptations to capture the nature of this variable. The PollenOriginModel has a link with reference to the PollenOriginReference model.

4.5.6. Minimum requirements for a valid data point

Each recorded Log must have at least 4 attributes:

- 1) Date (as a year, date or date with time);
- 2) Unique identification or ID;
- 3) Location information (a Postal code, NUTS ID, LAU ID, Country name or WGS84 coordinates);
and
- 4) The data point value of a specific variable (e.g. "25" if the in-hive temperature is 25°C).

In some cases, these minimum requirements have not been met. In such situations, the data is reclassified as a reference library, which might not have date, location or ID information. One such case is crop pollination dependency data acquired from Klein et al. (2007).

4.5.6.1. Database structure

Using the Data model in The Bee Hub as a basis, the EUBP Prototype Platform database structure has been constructed using the Data model in and the relationships are based on database capability. Currently, the selected database is MySQL. The reasons why MySQL has been selected for the database are:

- Standardisation: MySQL is the world's most used client-server relational database management system.
- Simplicity: it is simple to set up and develop on.
- Familiarity: most applications developed by the programmer behind the structural conception of the EUBP Prototype Platform were developed with MySQL.
- Versatility: due to the dynamic language capabilities of the selected framework (Laravel written in PHP), it is possible to switch the database engine in the future with minor modifications to the application.

Appendix C describes the procedure of importing a dataset through a practical example.

4.6. Data algorithms

The implemented algorithms use the data integrated both in the PoC and Prototype phase to produce information readily available to the public. The algorithms developed use colony weight and weather sensors, winter colony mortality, varroa infestation levels, and socioeconomic data linked to beekeeping and agriculture.

Each described algorithm consists of the following parts:

- 1) Identified problem and proposal;
- 2) Used data points;
- 3) Formulas;
- 4) Future uses

The pseudocode of the different algorithms can be found in Appendix D.

4.6.1. Daily colony production

4.6.1.1. Identified problem and proposal

For a beekeeper the most important information is to know the production of his/her colonies. According to the data provider CARI asbl, the scale information is the part of their website that is most visited by the members. Beekeepers seek to have a benchmark for their production, and they survey the evolution of the weight of referenced monitored hives in their region. If we can calculate average productivity within regions, we can provide an indication about how much honey a colony may produce now and in the future to the beekeepers in the region.

In order to estimate it, the daily production of a colony needs to be calculated. Because the amount of collected pollen, nectar and honeydew, and consequently produced honey, and wax, varies greatly between species, weather, geographical location and time, it is possible to calculate net production of all colony resources per day. Furthermore, the liquid/semi liquid component of the hive contributes the most to the change of weight of the hive.

Farmers may be willing to have access to the daily weight gain (or trends in time) of the colonies located in the surroundings of their fields as from the hive weight gain, they may get an indication about the

efficacy of their cropping practices (e.g. planting edges and flowering strips, selecting specific varieties of crops, managing their spraying activity, etc.)

Citizens may be curious to explore the daily weight gain of the honey bee colonies (or trends in the period) in their surroundings for educational or recreational purposes, because they are curious to see how the bees of their beekeeper neighbours are developing or because they know that if bees thrive, the environment in which they are living is healthy.

4.6.1.2. Used data points

To calculate daily production, (bi-)hourly colony weight data and additional colony management information are required, which can be provided by the beekeeper or detected automatically as described in 4.6.9. Colony management detection.

4.6.1.3. Formulas

A colony is active (foraging) only during daylight. It can be safely assumed that a colony is at its peak weight at sunset. But getting data reliably at variable sunset time is challenging, that is why it was decided to use the midnight weight of a colony to calculate daily production. It is important to take into account time zones. As a result, the local midnight time for each colony is compared. So daily production of a colony is the difference of weight (Δm_p) between 2 consecutive days as recorded at midnight (m_t and m_{t-1}).

$$\Delta m = m_t - m_{t-1}$$

But this calculation does not take into account beekeeping management. That is why, it is needed to exclude the weight change (Δm) of the beekeeping management operation from the final result.

$$\Delta m_p = m_t - m_{t-1} - \Delta m$$

4.6.1.4. Future use

The daily colony production is used in most algorithms following this one. It can establish a ranking of production areas and comparisons among regions. It can be used to establish if the year is average with regards to the historical production in a certain location. When meteorological, landscape and geographical data is integrated into the platform, predictive estimations of production may become available and we will be able to establish correlations between agricultural practices and the impact on the production capacity and development of honey bee colonies. In doing so, the efficacy of implementation of policies affecting the landscape composition may be inferred.

4.6.2. Colony production period detection

4.6.2.1. Identified problem and proposal

For a beekeeper it is important to know when bees are most productive and how he/she needs to adapt their management practices to work in synergy with the development of their colonies (e.g. add honey supers or prevent swarming). Also, it is important for them to understand when it is best to harvest the honey supers, in order either to maximise their productions or to get to an optimum balance of harvesting and feeding.

Additionally, decision makers may be interested in using the honey bee colonies as bio-indicators of the landscape and apply this concept to different policies affecting the landscape, such as the agricultural policy, the habitats directive, or the sustainable use of pesticide directive. The former two, for example, affect the landscape profile in increasing or reducing the flowering resources available to pollinators. A

poor evolution of the colony weight in some periods of the year and in certain locations, may indicate a problem in the implementation of the latter, even though there are more adapted approaches to verify it. By watching trends of colony production, we can discover increases or decreases in production trends within a period of time or even along the season.

4.6.2.2. Used data points

To detect colony production period, we need to have access to historical and live weight scale sensors. With the help of 4.6.1 Daily colony production we can calculate weekly trends.

4.6.2.3. Formulas

It is interesting to calculate the total increase in daily production per week ($\sum_{t=1}^7 \Delta mp$) above a threshold (m_{th}). For the implementation in the EUBP Prototype Platform, two thresholds have been used: (1) one is set at 2kg for increased production and (2) set at 5kg for highest production periods. These thresholds are based on MUST-B data and will have to be adjusted in the future for other bee species, hive types and climates.

$\sum_{t=1}^7 \Delta mp > m_{th}$; then a period of higher colony production was detected.

With more experimentation and different hive types, the threshold may be changed in the future or adapted for specific contextual conditions (bees' breed, hive type, etc.).

4.6.2.4. Future use

Beekeepers can see an interest in knowing the trends of production of reference hives along the year. In the future, meteorological data and landscape data (linked to resources, such as CORINE, land parcel used for farming subsidies, observational data on flowering crops/trees, etc.) should be integrated and environmental context should be provided to understand the trends, based on which predictive estimations will be made available to the users.

Similar analyses, but for different purposes, can be made available for farmers, decision making and dissemination and outreach purposes. Knowing the weather data and the crops in the surroundings of the hives, it is possible to estimate the potential risk of crop pests the potential risk from the use of pesticides for their control. This will require the introduction of data related to pesticide authorisation and toxicological profile. The trends in the colony production or weight in a period of time may indicate potential intoxication events. Additionally, it could be used for the detection of ecosystem/landscape changes and availability of resources at specific times of the year.

4.6.3. Colony consumption during the winter

4.6.3.1. Identified problem and proposal

The productive beekeeping season finishes with the last harvesting of honey. Afterwards it is common practice to estimate the level of honey reserves of the colonies to check if they have enough to survive the winter. Should this not be the case, then the beekeepers normally feed their colonies with a sugar-content syrup. Nonetheless, these feeding events might be done too late or may not provide the necessary amount and compromise colony winter survival. Additionally, it could also happen that the colony consumes reserves at a rate that is too high and where the beekeepers need to adapt the colony feeding behaviour for its survival.

4.6.3.2. Used data points

To detect winter colony consumption, we need to have access to historical and live weight scale sensors. Historical data can be used to calculate honey consumption in that area from previous winters and live

data can be used to detect if a colony is short on resources to survive, creating alarm systems for beekeepers. During field testing it was noticed that rain data needed to be included in order to exclude false positives of production (weight increase) during rain. The values could go usually up to 0.3 kg.

4.6.3.3. Formulas

Colony wintering consumption (m_w) starts after the last honey extraction (t_0) until the colony starts producing again (next spring positive trend, t_n) in the next year.

$$m_w = \sum_{t=0}^n t m_t$$

4.6.3.4. Future use

The minimum amount of honey a colony requires to survive the winter in a particular region and climatological conditions is possible. In doing so, the beekeeper can adapt his/her honey harvest or feeding activities so the winter mortality linked to hunger is minimised. Also, based on the weight loss during winter, alarm systems could be set to inform beekeepers that the colony urgently needs feeding to survive the winter. It is possible to evaluate if a colony dies during the winter as well.

Future developments and integration of data could fine-tune the data analyses and alarm systems by integrating climatological and landscape data. Indeed, it is common practice in agriculture to plant catch crops containing flowering plants such as mustard or phacelia. Should the weather be clement, bees could go and forage in these crops, which would either replace the need to feed the colonies or could supplement it. Furthermore, depending on the detail of landscape data, the interests or risks of winter crops/catch crops on the hives can be established, considering the impact of mass-flowering on the pre-wintering period (Simon-Delso et al., 2017).

4.6.4. Metabolic resting state of the colony

4.6.4.1. Identified problem and proposal

Inside the colonies, honey bees rely on nectar and pollen consumption for their own survival or to feed their brood and the queen. During the day in the active foraging period, their high foraging activity creates large weight variations and the consumption information gets mixed with the foraging. As a result, it is impossible to calculate the amount of reserves that the colony is consuming. Nonetheless, during the night, there is no foraging activity, all the bees are inside the colony and the work of the bees is devoted to evaporating the water of the nectar collected during the day and basal consumption. Outside the active foraging season, when there is no foraging and therefore no work on transformation of nectar into honey, the weight of the colony can give us an indication about the metabolic resting state. The amount of activity of bees is however reduced because there is less caring work to deal with, as the queen reduces her laying activity. The colony metabolic resting state by the whole colony could be estimated at a level between the weight during the night's active period and the inactive period.

4.6.4.2. Used data points

To calculate metabolic resting state, the access to historical and live weight scale sensors is necessary.

4.6.4.3. Formulas

It is possible to calculate the value of metabolic resting state (Δmr) by calculating the difference of weight between sundown (end of colony weight increase m_{t-1}) and sunrise (start of higher colony weight decrease m_t).

$$\Delta mr = m_t - m_{t-1}$$

4.6.4.4. Future use

Information about Metabolic Resting State could be interesting in biology research. To achieve adequate estimates of the Metabolic Resting State, it would be required to have high precision in the weight measurements, meteorological data providing the context of the colony and possibly an experimental approach in order to be able to isolate the Metabolic Resting State during the season from other activities such as nectar evaporation.

4.6.5. Sensor statistics on apiary, district, region, and country level

4.6.5.1. Identified problem and proposal

Historical and live data for various hives (sometimes even several hives within a single apiary) are stored in the database of the EUBP Prototype Platform and can be used for calculating statistics. Hive weight, hive temperature, hive humidity and various weather values can be used to calculate several statistics for apiary, district, regional and even country level averages (based on available data).

This information could be useful to the beekeeper to assess the hive performance and compare to other beekeepers within a local or larger area.

4.6.5.2. Used data points

To calculate these statistics, live or historical sensor data from hive scales with internal temperature and humidity sensors are needed. Historical sensor data is the one provided in the MUST-B project, for example. Data was registered by sensors, stocked in a server and shared with the platform manually by a data provider after the period of observation. Live data is data usually gathered by an API²⁵ shared simultaneously and frequently with the platform. Live data becomes "historical" once it stays recorded in the database.

Currently, all available data is provided by the CARI asbl National Beekeeping Programme in Wallonia (Belgium) and the MUST-B Portugal field data. Algorithms calculating yearly values can only be applied in historical data.

4.6.5.3. Formulas

We take the values supplied in the following table from each hive and use the written methodology to calculate common values on apiary, district, region (NUTS 3 - 1) and country level.

Available data	Methodology
Hive Weight Scales data	Calculating daily midnight (last known of the day) values
Colony production data	Calculating daily and periodical (weekly, monthly, yearly) median values
Hive temperature data	Calculating daily minimum and maximum values
Hive humidity data	Calculating daily median values
Temperature data	Calculating daily minimum and maximum median values
Humidity data	Calculating daily median values
Rain data	Calculating daily sum of values
Solar radiation data	Calculating daily sum of values
Wind speed data	Calculating daily median and maximum values
Wind gust data	Calculating daily median and maximum values
Wind direction data	Calculating daily median value
Colony winter consumption	Calculating yearly median values

²⁵ Automated Programming Interface

For the purpose of statistical analysis within the platform, median values are preferred over mean values.

4.6.5.4. Future uses

With the help of average regional data, the performance of individual hives as well as apiaries or even distinct districts or regions can be compared and rated. If the dataset grows enough, it would allow us to measure the suitability of the environment, a kind of bioindicator for landscape fitness even for other pollinators.

4.6.6. Available flight time to bees

4.6.6.1. Identified problem and proposal

The production of a beehive depends highly on the available flight time of bees and the available resources in the environment. If it is possible to calculate the amount of available flight time based on available sensor data, better insights could be given into what influences honey production. This information provides the potential for foraging coming from the colony. For example, if two apiaries are positioned in two highly different elevations and environments, with the help of the temperature, humidity, rain and solar radiation sensors one could determine if the cause of lower production is directly caused by reduced flight time in this less suitable environment. At a later stage, it is possible to complement the calculation with climatic and landscape data if this data is made available, so that the availability of resources can be taken into account.

4.6.6.2. Used data points

In order to calculate the available flight time, information about individual location, sunset and sunrise time, temperature, humidity, rain and solar radiation sensors can be used. Additionally, to validate the results, they can be compared with the change of hive weight sensors, which should inform if the bees are actually foraging at that time.

The relevant datasets are provided by CARI asbl and MUST-B Portugal field data.

4.6.6.3. Formulas

To determine a certain time frame as available flight time, certain conditions should be met. Based on beekeeping practices and observations, it must be first determined when is the most suitable time for bees to forage:

- 1) The time between sunrise and sunset
- 2) Minimum solar radiation - parameters defined and validated based on the scientific literature review and field observations.
- 3) Minimum and maximum temperature for active foraging in honey bees, i.e. 12°C and 55°C - parameters defined and validated based on scientific literature review and field observations.
- 4) No rainfall.

The available flight time is then calculated for every day according to above conditions.

More research will be required to improve the performance of the algorithm by including more parameters in the calculations.

4.6.6.4. Future uses

The flight time calculations could be invaluable when using meteorological prediction algorithms and feed them to this algorithm. In doing so, one could create a short term prediction model of honey production based on the available meteorological forecast. Future periods of higher colony production could be determined and assist beekeepers in their management tasks.

Knowing the flight time, a more precise calculation of the foraging activity can be done as well. However, the latter requires a more robust prediction algorithm, based on models such as the one currently being developed by APISRAM (EFSA, 2016), BEEHAVE (Becher et al., 2016), HOPOMO (Schmickl and Crailsheim, 2007), etc. or the one described in Clarke and Robert, 2018.

4.6.7. Average colony production based on honey production and beehive statistics

4.6.7.1. Identified problem and proposal

Honey production data on a country level can be misleading considering the number of colonies and country size. For a more comprehensive comparison, it is better to calculate the amount of honey produced per colony in each country.

To do so, we can use data on honey production in EU27 and the number of beehives in EU27 (FAOSTAT) to calculate the average amount of honey produced in each colony per country.

4.6.7.2. Used data points

In order to calculate the colony production per country, data from the number of beehives and on honey production in EU27 was used. This data is available for download on the FAOSTAT website.

4.6.7.3. Formulas

For colony country production we would have to divide the production (honey in kg) by the number of beehives.

4.6.7.4. Future uses

The level of accuracy and the determination of deviation from averages can be validated by comparing the results of the algorithm to the FAOSTAT data. It can also be used to determine a data saturation point, where it represents the point in which the analysis output will not change even if more data is added. In doing so it can be established if there are enough sampling points/colonies/apiaries for the results to be trustworthy, reliable and representative.

The validity of the data can be tested (and proved), if one crosses the saturation point in the future, by having enough available datasets in the platform database.

4.6.8. Cumulative honey production

4.6.8.1. Identified problem and proposal

One of the main information important for beekeeping management is to know the amount of product, mainly honey, that exists in a hive. However, it is often problematic to have a precise estimate of this amount because the overall weight gain of the colony includes not only honey but also the weight of bees, pollen, wax, brood, etc.

Therefore, in order to estimate the amount of honey a colony may produce, the following parameters need to be assumed: (1) a minimum value of weight at the end of the winter that represents the empty hive, (2) 1.5 kg of bees and (3) 3 kg of honey leftover. These data come from field observations carried

out within the MUST-B project (Capela, pers. comm.). Additionally, since the platform counts with precise data on colony development, the evolution of the weight of the bees and brood over the year can be estimated. The remaining weight represents an approximate value of products within a beehive. By using automatically detected management events, one can track any events that change the hive weight and exclude them from calculations.

4.6.8.2. Used data points

To calculate honey production, live or historical sensor data from the hive scales is required. Additionally, the accuracy of the algorithm significantly increases if there is data about the management events available, for there is no need to estimate them (4.6.9.). Management events are used to justify a weight change. When they are detected and registered by the beekeeper, it is possible to verify them and remove their impact on the output data.

Currently, all available data is provided by the CARI asbl National Beekeeping Programme in Wallonia (Belgium) and MUST-B Portugal field data. Since the MUST-B data also contains honey production and scale information, this data will be used to validate this algorithm.

4.6.8.3. Formulas

The calculation starts by finding the minimum weight value in spring, then any addition of weight is proportionally added as product weight. With the help of the colony development model, created from the MUST-B Portugal field data, the weight increase of the beehive throughout the year can be estimated.

4.6.8.4. Future uses

This information can then be used to compare local beekeepers' production in comparison with the district, region or country level averages (from 4.6.1.) to check if production is above or below average figures. One simply needs to subtract colony country production with colony/apiary production to see if it is above or below average.

4.6.9. Colony management detection

4.6.9.1. Identified problem and proposal

Weight (scales) and temperature sensors gathered do not include colony management data, i.e. what the beekeeper did on what day or at what time. Without this information, the context of what is happening within the hives can only be deduced from the data of hive scales and inside temperature. Additionally, the weather station data can be used to exclude natural events introducing a weight change like snowing or raining.

Based on years of beekeeping management practices it is possible to identify certain management events happening within the colony based on the scale data.

These kinds of assumptions are not without a risk, data could be misinterpreted. This was the most complex algorithm and certain modifications and fine tuning were introduced to improve the accuracy and functionalities. The present version of the algorithm is still highly experimental and requires further testing for refinement.

4.6.9.2. Used data points

To identify management decisions, hourly historical and live colony hive scales data, inside and outside temperature, humidity and rainfall sensors are needed. All data is provided by the CARI asbl National

Beekeeping Programme in Wallonia (Belgium). In addition, the MUST-B field data was used to test and validate the algorithm, for the management events are well described within the dataset.

4.6.9.3. Formulas

For most management events was difference of weight (Δm) between two data points are looked for, measured within an hour (t and t-1).

$$\Delta m = m_t - m_{t-1}$$

Additionally, Table 2 was used to narrow possible matches based on type of weight change, date, time of day and temperature. The following example data is based on Langstroth Beehive for Portugal.

As beekeeping practices are different from region to region and there are several types of material used such as the beehive model, in the future it will be necessary or recommendable to prepare such a table for each hive type.

Table 2: Possible matches between weight change, date, time of the day and temperature described for hives type Langstroth in Portugal

Management	Weight increase or decrease	Weight range	Date	Time of day
Adding or removing brood frame	Both	0,5 - 2,5kg	March - October	Sunlight
Adding or removing a honey frame	Both	0,5 - 3,3 kg	March - October	Sunlight
Adding an empty frame	Increase	0,4 - 0,5 kg	March - October	Sunlight
Adding honey super with wax comb frame	Increase	7,0 - 9,0 kg	March - July	Sunlight
Adding Honey super with laminated wax frame	Increase	5,0 - 7,5 kg	March - July	Sunlight
Adding upper nest and frames	Increase	10,0 - 12,0 kg	March - June	Sunlight
Feeding	Increase	2,5 kg	September - March	Sunlight
Adding or removing pollen trap	Both	2,0 - 3,0 kg	March - June	Sunlight
Honey collection	Decrease	5,0 - 60,0 kg	July - October	Early morning

The following step is to detect false positives by imposing a number of conditions (Table 3). In the future, additional conditions will be added based on new acquired data.

Table 3: Conditions or criteria established to remove false positives

Conditions	Explanation
Nighttime	The largest amount of management practices in the hives, apart from transhumance, is done during the day.
Sudden weight drop and an equal rise within an hour	Probably a management event that removed pieces of the Hive and repositioned them after the work is done

4.6.9.4. Future uses

If live data is considered and a possible event happens, it is possible to inform the data provider about it to verify our conclusions. Additionally, this could be used to detect swarming events, natural disasters, equipment malfunctions or robbery/vandalism of the hive. However, due to reliability concerns, testing and validation is required in the future.

4.6.10. Economic value of pollination

4.6.10.1. Identified problem and proposal

The economic value of pollinators consists of two components: (1) Direct value of beekeeping products mainly produced with the Western Honeybee (*Apis mellifera*) and (2) indirect value created by the pollination of agricultural crops. The major beekeeping products produced with *A. mellifera*, for which at least rough estimates are available, are honey and, to a lesser degree, beeswax. Pollination dependency of agricultural crops, in turn, has been roughly estimated (Klein et al., 2007). According to a method proposed by Gallai et al. (2009) the economic value of insect pollination has been calculated. This value may be subject to both long- and short-term changes due to changes in crop production and prices. The proposed algorithms (Gallai et al., 2009) have therefore been extended to assess the change in economic value of crop production.

4.6.10.2. Used data points

Estimates on the economic value of honey and beeswax production as well as on the amount of managed honey bee colonies were obtained from data provided by the Food and Agriculture Organization of the United Nations (FAO) and by a local beekeeping association. A rough assessment of the economic value of crop pollination was made by a method proposed by Gallai et al. (2009), based on pollination dependency ranges for different crops provided by Klein et al. (2007).

Data on honey, wax (t) and agricultural crop production (t, ha) in the EU from 1961 to 2018 was obtained from FAO at NUTS0 level (FAOSTAT). Producer prices (US\$) for honey, beeswax and agricultural crops in the EU from 1991 to 2018 were obtained from FAO at NUTS 0 level (FAOSTAT). The number of managed colonies was obtained from FAO (1961 to 2018) at NUTS0 level (FAOSTAT) and from the Austrian Beekeepers Federation *Biene Österreich* (2001-2019) for Austria at NUTS2 level. The pollination dependency of agricultural crops was obtained from Klein et al. (2007). A table containing the scientific names of plant species belonging to a particular agricultural crop as well as their accession number at the Global Biodiversity Information Facility (GBIF) was provided for possible future integration of ecological information (e.g. interactions between pollinator species and crop species).

To calculate the economic value of insect pollination (*EV*) the prices and quantities for a given crop produced in a given region for a given year need to be accessed. This data can be assessed at the website of the Food and Agricultural Organization Corporate Statistical Database (FAOSTAT). Besides, the dependency of a given crop from insect pollination is required. This value has been assessed by Klein et al. (2007).

4.6.10.3. Formulas

The total economic value (EV_{ixy}) of a given crop (i) in a given EU member state (x) in a given year (y) is calculated according to the following formula where Q_{ixy} is the quantity produced of crop i in EU member state x in year y and P_{ixy} is the annual average price of crop i per unit of quantity produced in EU member state x in year y :

$$EV_{ixy} = P_{ixy} \times Q_{ixy}$$

The minimum ($PEVmin_{ixy}$), maximum ($PEVmax_{ixy}$) and average ($PEVavg_{ixy}$) economic value of pollination of a given crop (i) in a given EU member state (x) in a given year (y) is calculated according to the following formulas, where for each crop $i \in [1, I]$, Q_{ixy} is the quantity produced of this particular crop in EU member state x in year y , P_{ixy} is the annual average price of this particular crop per unit of quantity produced in EU member state x in year y , $Dmin_i$ is the minimum and $Dmax_i$ the maximum dependence ratio of crop i on insect pollinators.

$$\begin{aligned} PEVmin_{ixy} &= P_{ixy} \times Q_{ixy} \times Dmin_i \\ PEVmax_{ixy} &= P_{ixy} \times Q_{ixy} \times Dmax_i \\ PEVavg_{ixy} &= \frac{PEVmax_{ixy} + PEVmin_{ixy}}{2} \end{aligned}$$

The relative economic value of plant pollination ($RPEVavg_{ixy}$) for a given crop (i) in a given EU member state (x) in a given year (y) is then calculated as the percentage of average economic value of plant pollination for the given year y ($PEVavg_{ixy}$).

$$RPEVavg_{ixy} = \frac{PEVavg_{ixy} \times 100}{EV_{ixy}}$$

The total economic value (EV_{iy}) of a given crop (i) in a given year (y) in all EU member states is calculated according to the following formula where for each EU member state $x \in [1, X]$, EV_{ixy} is the total economic value of crop i in EU member state x in year y :

$$EV_{iy} = \sum_{x=1}^X (EV_{ixy})$$

The minimum ($PEVmin_{iy}$), maximum ($PEVmax_{iy}$) and average ($PEVavg_{iy}$) economic value of pollination for a given crop (i) in a given year (y) in all EU member states is calculated according to the following formulas, where for each EU member state $x \in [1, X]$, $PEVmin_{ixy}$ is the minimum economic value of plant pollination for crop i in EU member state x in year y and $PEVmax_{ixy}$ is the maximum economic value of plant pollination for crop i in EU member state x in year y .

$$\begin{aligned} PEVmin_{iy} &= \sum_{x=1}^X (PEVmin_{ixy}) \\ PEVmax_{iy} &= \sum_{x=1}^X (PEVmax_{ixy}) \\ PEVavg_{iy} &= \frac{PEVmax_{iy} + PEVmin_{iy}}{2} \end{aligned}$$

The relative economic value of pollination ($RPEV_{iy}$) for a given crop (i) in a given year (y) in all EU member states is calculated according to the following formula where $PEVavg_{iy}$ is the average economic value of pollination for a given crop (i) in a given year (y) in all EU member states and EV_{iy} is the total economic value of a given crop (i) in a given year (y) in all EU member states.

$$RPEV_{iy} = \frac{PEVavg_{iy} \times 100}{EV_{iy}}$$

The average relative economic value of pollination ($RPEV_{iy1Y2}$) for a given crop (i) in all EU member states for the years Y_1 through Y_2 is calculated according to the following formula where $RPEV_{iy}$ is the

relative economic value of pollination for a given crop (i) in a given year (y) in all EU member states and n is the number of years:

$$RPEV_{iY1Yn} = \frac{\sum_{y=Y1}^{Yn} (RPEV_{iy})}{n}$$

The total economic value of crop production (EV_{xy}) in a given EU member state (x) in a given year (y) is calculated according to the following formula where for each crop $i \in [1, I]$, EV_{ixy} is the total economic value of crop i in EU member state x in year y :

$$EV_{xy} = \sum_{i=1}^I (EV_{ixy})$$

The minimum ($PEVmin_{xy}$), maximum ($PEVmax_{xy}$) and average ($PEVavg_{xy}$) economic value of crop pollination in a given EU member state (x) in a given year (y) is calculated according to the following formulas, where for each crop $i \in [1, I]$, $PEVmin_{ixy}$ is the minimum economic value of plant pollination for crop i in EU member state x in year y and $PEVmax_{ixy}$ is the maximum economic value of plant pollination for crop i in EU member state x in year y .

$$PEVmin_{xy} = \sum_{i=1}^I (PEVmin_{ixy})$$

$$PEVmax_{xy} = \sum_{i=1}^I (PEVmax_{ixy})$$

$$PEVavg_{xy} = \frac{PEVmax_{xy} + PEVmin_{xy}}{2}$$

The relative economic value of crop pollination ($RPEV_{xy}$) for a given EU member state (x) in a given year (y) is calculated according to the following formula where $PEVavg_{xy}$ is the average economic value of crop pollination for a given EU member state (x) in a given year (y) and EV_{xy} is the total economic value of crop production for a given EU member state (x) in a given year (y).

$$RPEV_{xy} = \frac{PEVavg_{xy} \times 100}{EV_{xy}}$$

The average relative economic value of crop pollination ($RPEV_{xY1Yn}$) for a given EU member state (x) for the years Y_1 through Y_n is calculated according to the following formula where $RPEV_{xy}$ is the relative economic value of crop pollination for a given EU member state (x) in a given year (y) and n is the number of years:

$$RPEV_{xY1Yn} = \frac{\sum_{y=Y1}^{Yn} (RPEV_{xy})}{n}$$

The total economic value of crop production in all EU member states (EV_y) in a given year (y) is calculated according to the following formula where for each crop $i \in [1, I]$ and for each EU member state $x \in [1, X]$, EV_{ixy} is the total economic value of crop i in EU member state x in year y :

$$EV_y = \sum_{x=1}^X \sum_{i=1}^I (EV_{ixy})$$

The minimum ($PEVmin_y$), maximum ($PEVmax_y$) and average ($PEVavg_y$) economic value of crop pollination in all EU member states in a given year (y) is calculated according to the following formulas,

where for each crop $i \in [1, I]$ and for each EU member state $x \in [1, X]$, $PEVmin_{ixy}$ is the minimum economic value of plant pollination for crop i in EU member state x in year y and $PEVmax_{ixy}$ is the maximum economic value of plant pollination for crop i in EU member state x in year y :

$$PEVmin_y = \sum_{x=1}^X \sum_{i=1}^I (PEVmin_{ixy})$$

$$PEVmax_y = \sum_{x=1}^X \sum_{i=1}^I (PEVmax_{ixy})$$

$$PEVavg_y = \frac{PEVmax_y + PEVmin_y}{2}$$

The minimum ($RPEVmin_y$), maximum ($RPEVmax_y$) and average ($RPEVavg_y$) relative economic value of crop pollination in all EU member states in a given year (y) is calculated according to the following formulas where $PEVmin_y$, $PEVmax_y$ and $PEVavg_y$ are the minimum, maximum and average economic value, respectively, of crop pollination in all EU member states in a given year (y) and EV_y is the total economic value of crop production in all EU member states in a given year (y):

$$RPEVmin_y = \frac{PEVmin_y \times 100}{EV_y}$$

$$RPEVmax_y = \frac{PEVmax_y \times 100}{EV_y}$$

$$RPEVavg_y = \frac{PEVavg_y \times 100}{EV_y}$$

The total area of cultivated crops for which the dependency on insect pollination according to Klein et al. (2007) is greater than 0, subsequently referred to as pollinated crops (PA_{xy}) in a given EU member state (x) in a given year (y) is calculated according to the following formulas, where the pollinated crop $i \in [1, I]$ and A_{ixy} is the total area of a pollinated crop i in EU member state x in year y :

$$PA_{xy} = \sum_{i=1}^I (A_{ixy})$$

The number of beehives per total area of pollinated crops (BPA_{xy}) for a given EU member state (x) in a given year (y) is calculated according to the following formula where B_{xy} is the total amount of beehives kept in a given EU member state (x) in a given year (y) and PA_{xy} is total area of pollinated crops in a given EU member state (x) in a given year (y):

$$BPA_{xy} = \frac{B_{xy}}{PA_{xy}}$$

The average number of beehives per total area of pollinated crops (BPA_{xy1Yn}) for a given EU member state (x) for the years $Y1$ through Yn is calculated according to the following formula where BPA_{xy} is the number of beehives per total area of pollinated crops for a given EU member state (x) in a given year (y) and n is the number of years:

$$BPA_{xy1Yn} = \frac{\sum_{y=Y1}^{Yn} (BPA_{xy})}{n}$$

4.6.10.4. Future uses

Information on the economic value of pollination should be considered by policy makers when they weigh up risks for pollinators caused by the application of plant protection products or changes in land use and economic consequences for agriculture. It may also be used to predict future developments and future significance of the beekeeping sector and its impact on crop production in the member states of the European Union.

4.6.11. Beekeeping economic review

4.6.11.1. Identified problem and proposal

The beekeeping sector lacks a comprehensive overview on its economic and social performance (size, productivity, economic significance, regional differences), its development in the past and future prospects.

4.6.11.2. Used data points

Estimates on the economic value of honey and beeswax production as well as on the amount of managed honey bee colonies were obtained from data provided by the Food and Agriculture Organization of the United Nations (FAO) and by a local beekeeping association. Data on honey, wax (t) and agricultural crop production (t, ha) in the EU from 1961 to 2018 was obtained from FAO at NUTS0 level (FAOSTAT). Producer prices (US\$) for honey, beeswax and agricultural crops in the EU from 1991 to 2018 were obtained from FAO at NUTS 0 level (FAOSTAT). The number of managed colonies was obtained from FAO (1961 to 2018) at NUTS0 level (FAOSTAT) and from the Austrian Beekeepers Federation *Biene Österreich* (2001-2019) for Austria at NUTS2 level.

4.6.11.3. Formulas

The total number of beehives in a member state aggregate (BA_{yX1Xn}) in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, B_{xy} is the number of beehives in a single member state (x) in a given year (y):

$$BA_{yX1Xn} = \sum_{x=X1}^{Xn} (B_{xy})$$

The total number of beehives in a member state aggregate (BAA_{yX1Xn}) per total area in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, A_{xy} is the total area of a single member state (x) in a given year (y):

$$BAA_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{B_{xy}}{A_{xy}} \right)$$

The total number of beehives in a member state aggregate (BAI_{yX1Xn}) per total number of inhabitants in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, I_{xy} is the total area of a single member state (x) in a given year (y):

$$BAI_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{B_{xy}}{I_{xy}} \right)$$

The total amount of honey production in a member state aggregate (HA_{yX1Xn}) in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, H_{xy} is the amount of honey produced in a single member state (x) in a given year (y):

$$HA_{yX1Xn} = \sum_{x=X1}^{Xn} (H_{xy})$$

The total amount of wax production in a member state aggregate (WA_{yX1Xn}) in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, W_{xy} is the amount of wax produced in a single member state (x) in a given year (y):

$$WA_{yX1Xn} = \sum_{x=X1}^{Xn} (W_{xy})$$

The total amount of honey production in a member state aggregate (HBA_{xy}) per beehive in a given year (y) is calculated according to the following formula:

$$HBA_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{H_{xy}}{B_{xy}} \right)$$

The total amount of honey production in a member state aggregate (HIA_{xy}) per inhabitant in a given year (y) is calculated according to the following formula:

$$HIA_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{H_{xy}}{I_{xy}} \right)$$

The total amount of beeswax production in a member state aggregate (WBA_{xy}) per beehive in a given year (y) is calculated according to the following formula:

$$WBA_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{W_{xy}}{B_{xy}} \right)$$

The total value of honey production in a member state aggregate (HPA_{yX1Xn}) in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, PH_{xy} is the producer price of honey produced in a single member state (x) in a given year (y):

$$HPA_{yX1Xn} = \sum_{x=X1}^{Xn} (H_{xy} \times PH_{xy})$$

The total value of honey production in a member state aggregate ($HPBA_{yX1Xn}$) in a given year (y) per beehive is calculated according to the following formula:

$$HPBA_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{H_{xy} \times PH_{xy}}{B_{xy}} \right)$$

The total value of beeswax production in a member state aggregate (WPA_{yX1Xn}) in a given year (y) is calculated according to the following formula, where for each member state $x \in [X1, Xn]$, PW_{xy} is the producer price of beeswax produced in a single member state (x) in a given year (y):

$$WPA_{yX1Xn} = \sum_{x=X1}^{Xn} (W_{xy} \times PW_{xy})$$

The total value of beeswax production in a member state aggregate ($WPBA_{yX1Xn}$) in a given year (y) per beehive is calculated according to the following formula:

$$WPBA_{yX1Xn} = \sum_{x=X1}^{Xn} \left(\frac{W_{xy} \times PW_{xy}}{B_{xy}} \right)$$

4.6.11.4. Future uses

Information on the economic and social performance of the beekeeping sector should be used by representatives of the beekeeping industry and policy makers as a guideline for the implementation of future policies.

5. Engagement and communication activities

The initial contract for the development of the Prototype platform included a limited number of activities that can be found in Table 4. Additionally, the development team has carried out other engagement and communication activities from which there is interest in continuing the development of the platform in the future (Table 5).

Table 4: Main engagement and communication activities carried out during the contract period

Activity	Information
EUBP Prototype Platform Promotional Video (Main video)	Promotional video on the EUBP prototype platform explaining the concept, development status, how it works, and inviting potential data providers to share their data. Link to online video (BeeLife's YouTube channel): https://youtu.be/4JGkoaIC_fo Original video file + Subtitles file and Storyboard: http://bit.ly/EUBP-Final-Video-Files
EUBP Prototype Platform Promotional Video - Version for potential data providers	Promotional video on the EUBP prototype platform. Extended version targeting potential data providers with a specific call on interesting types of providers. The video presents specific benefits for each category of data providers. Link to online video (BeeLife's YouTube channel): https://youtu.be/HBLBc41Pk6Y Original video file + Subtitles file and Storyboard: http://bit.ly/EUBP-Final-Video-Files
EUBP Prototype Platform Promotional Video - Social Media version	Abridged version of the promotional video with appropriate length and rhythm for social media. Link to online video (BeeLife's YouTube channel): https://youtu.be/18wCuIyYH_M Original video file + Subtitles file and Storyboard: http://bit.ly/EUBP-Final-Video-Files
EUBP Prototype Platform Tutorial Video	Tutorial video for users to navigate the platform and register as a data provider. Link to online video (BeeLife's YouTube channel): https://youtu.be/k1Zigaxz3vk Original video file + Subtitles file: http://bit.ly/EUBP-User-Tutorial-Video
Participation to EUBP 7th (Virtual) Meeting	Presentation of developments of the platform and invitation to other stakeholders to become data providers.

	BeeLife also requested a letter of support from the EU Bee Partnership to seek future funding for further development. October 27, 2020 - Online Meeting
Participation to EUBP 8th (Virtual) Meeting	June 10, 2021 Online Meeting Presentation of final results of the platform development and discussion of further involvement by partners and stakeholders.

Table 5: Additional engagement and communication activities carried out during the contract period

Activity	Information
Bilateral meeting w/ Bee tech services (BeeK-ing)	May 25, 2021 Bee Tech has a monitoring and data collection mobile app for beekeepers. During the meeting it was discussed the possibility for Bee Tech to create an API to allow sharing data with the platform, as well as promoting data sharing with their customers.
Participation in the Apimondia Bee Data Group - BeeXML	<ul style="list-style-type: none"> • May 5, 2021 • January 20, 2021 • September 22, 2020 • August 24, 2020 <p>The Apimondia Bee Data Group is a group focused on the standardisation of bee health related data for improved data communication and sharing.</p> <ul style="list-style-type: none"> • Presentation of advances in the platform development. • Discussion on possible forms of data standardisation. • Workshops for data standardisation efforts. • Creation and editorial for a bee data standardisation (Bee XML) peer-reviewed journal.
Bilateral meeting w/ Finnish Beekeepers Association	February 1, 2021 The Finnish Beekeepers Association counts with a network of digital beehive monitoring systems. The possibility for them to become data providers was discussed. Additional information was sent for the Association's Board of Directors to review. A new meeting after the peak of the beekeeping season is expected.
Bilateral meeting w/ ARPAT - Agenzia Regionale per la Protezione Ambientale	October 29, 2020 ARPAT agreed to develop an API to share data from their network of beehive digital monitoring systems with the platform.
Bilateral meeting w/ ASPROMIELE - Associazione Produttori di Miele Piemonte	October 29, 2020

	ASPROMIELE agreed to provide access to the server with all the data from their network of beehive digital monitoring systems with the platform.
Bilateral meeting w/ APIS/Beeyard Technology	March 16, 2021 Exploratory meeting for potential collaborations and data sharing.
Bilateral meeting w/ ECPA - European Crop Protection Association	June 19, 2020 Potential collaborations for data sharing and synergies between efforts towards data standardization
Bilateral meeting with BeeSecure	November 2, 2020 Discussion on potential data sharing from BeeSecure

6. Conclusions

The EUBP Prototype Platform achieved a major technological breakthrough during the present contract. The inclusion of further datasets has enabled us to complete and improve the platform's data models, leading the efforts for the standardisation of bee related data. The further development of the data models contributes actively towards establishing a data standard, in synergy with the BeeXML's group work. Advancements in the data model definition are shared with the group, which hopefully will enrich the standardization process. However, further efforts in data integration still need to be achieved: further descriptors, years, geographical coverage, etc.

Furthermore, it was possible to design a clear protocol for data integration and further features and functionalities have been integrated into the web application. It is clear at this point that the lack of standardisation in data/metadata documentation and communication makes the task of data integration the most time consuming one in the process, as the procedure for a large majority of datasets still requires an initial manual check. The prototyping phase has aimed at reducing the time and administrative burden of the procedure of becoming a data provider. The prototype has aimed at mainstreaming the data sharing process. Initially, a direct contact was done between the development team and the data provider and a Data Sharing Agreement was signed by both parties. Through the data integration experiences in the platform development, metadata collection is further standardised and less time consuming.

Thanks to new datasets integration, a number of data gaps were identified, and the team realised the lack of quality of many of them. The lack of standardisation in data collection, documentation and communication is a fact that hampers the re-use of multiple datasets for other purposes than those for which they were generated. Metadata is often poorly documented, e.g. description of the methodology followed, generator of the data, units, etc. These gaps require in some cases further collaboration between the team and the data providers.

The development team has shown the plausibility of many datasets to be visualised and be transformed into information readily consumable by end users. The algorithms created will be further improved in the future thanks to the addition of datasets, sources and parameters.

7. Recommendations

Thanks to the work developed so far, the contractor would like to recommend a number of issues to favour the future development of an operational platform integrating, processing and communicating bee-related data.

Some recommendations linked to the development of the platform are:

- A number of datasets have remained in the implementation pipeline, meaning that these datasets are integrated, but not processed nor visualised. In addition, new data providers agreed to share their data with the platform, but the procedure of data acquisition was not finalised. Future efforts should prioritise the full implementation of these datasets;
- Visualisation of the data in the platform is essential to strengthen the usability of the platform. These visualisations need to be further implemented and improved, possibly through collaborations with designers. In addition, the platform speed and visuals, in terms of branding, should be enhanced;
- At one point in the future, more disciplines should be involved in the process (e.g. data scientists, web designer, GIS experts, etc.);
- Future versions of the platform could benefit from a more in-depth and complete platform management interface for administrators;

- The reflection about the difference between data provider and data owner, whenever they are not the same person/institution, needs to be clarified;
- To ensure the long-term functionality of the platform, it is recommended that in the future it operates in tight connection with the EFSA Knowledge Junction (Zenodo) or similar, mainly to store raw, cleaned and processed data for external use;
- In the future, it would be ideal to integrate a conversion tool to BeeXML in the platform, a ready-to-use tool in which anyone with data interesting for the platform can upload his/her data from an excel file or similar and an XML version of the data would be created and would be ready to be integrated.

Other recommendations more linked to the data are:

- More datasets need to be included. If possible, they should include the same descriptors as those already in the platform to further complete the existing data models, but also different kinds to extend the standardisation exercise to other topics;
- Further improve and add more/complete reference libraries: e.g. lists of names of pesticide active ingredients in different languages, lists of species of bees and other insect pollinators in latin and other languages, lists of species of plants and crops, lists of habitat types, etc.;
- For doing so, extensive engagement activities with field practitioners, researchers and public and private institutions will need to be carried out.

Recommendations about the data processing are:

- In the future platform, a system that allows the traceability of data owners and projects needs to be created whenever data of different sources is processed, resulting into new datasets or outputs;
- The algorithm performance should be improved (enhance calculation speed, functionality and usability) and there is room for many additional algorithms to be created.

Recommendations about the future of pollinator-related/environment-related data production are:

- To avoid the problem of lack of data standardisation and to ensure the reuse value of any data produced from any EU pollinator-related or environment-related projects, projects may involve the EUBP. Exploring synergies and stimulating discussion will help establish and inform minimum requirements to communicate/store data. For instance, metadata needs to be provided in an XML format, well documented and ready to be integrated into the platform or to be shared with other initiatives;
- Large communication should be done on the data standards and good data management and data communication practices;
- The organisation of specific pollinator-data standardisation and data identification workshops and dissemination are of key importance to be able to continue the integration efforts

References

- Becher MA, Volker G, Pernille T, et al., 2014. BEEHAVE: A Systems Model of Honeybee Colony Dynamics and Foraging to Explore Multifactorial Causes of Colony Failure. *Journal of Applied Ecology* 51(2): 470–482.
- BEEP - Bijenmonitoring, n.d. <https://beep.nl/>, accessed April 1, 2021.
- BeeXML – Exchanging Data about Bees and Beekeeping, <http://beexml.org/>, accessed April 1, 2021.
- Brodschneider R, Gray A, Adjlane N, et al., 2018. Multi-Country Loss Rates of Honey Bee Colonies during Winter 2016/2017 from the COLOSS Survey. *Journal of Apicultural Research* 57(3): 452–457.
- Brodschneider R, Gray A, Van der Zee R, et al., 2016. Preliminary Analysis of Loss Rates of Honey Bee Colonies during Winter 2015/16 from the COLOSS Survey. *Journal of Apicultural Research* 55: 375–378.
- Clarke D, and Daniel R, 2018. Predictive Modelling of Honey Bee Foraging Activity Using Local Weather Conditions. *Apidologie* 49(3): 386–396.
- Dietemann, V, Ellis JD, and Neumann P, 2013a. The COLOSS BEEBOOK Volume I, Standard Methods for *Apis Mellifera* Research: Introduction. *Journal of Apicultural Research* 52(4). Taylor & Francis: 1–4.
- Dietemann, V, Ellis JD, and Neumann P, 2013b. The COLOSS BEEBOOK Volume II: Standard Methods for *Apis Mellifera* Pest and Pathogen Research, vol.52. International Bee Research Association IBRA.
- EFSA AHAW Panel, 2016. Assessing the Health Status of Managed Honeybee Colonies (HEALTHY-B): A Toolbox to Facilitate Harmonised Data Collection. *EFSA Journal* 14(10): e04578.
- EFSA, 2016. A mechanistic model to assess risks to honeybee colonies from exposure to pesticides under different scenarios of combined stressors and factors. EFSA supporting publication 2016:EN-1069. 116 pp
- EFSA, 2017. Specifications for Field Data Collection Contributing to Honey Bee Model Corroboration and Verification. EFSA Supporting Publications 14(5): 1234E.
- EFSA, 2018. Terms of Reference for an EU Bee Partnership. EFSA Supporting Publications 15(5): 1423E.
- EFSA, 2019. Programming Document 2020-2022. Trusted Science for Safe Food. Protecting Consumers' Health with Independent Scientific Advice on the Food Chain. Parma, Italy: EFSA.
- EUBP Prototype Platform, N.d. EU Bee Partnership Prototype Platform on Bee Health. <https://bee-ppp.eu/>, accessed April 1, 2021.
- Gallai N, Salles JM, Settele J, and Vaissière BE, 2009. Economic Valuation of the Vulnerability of World Agriculture Confronted with Pollinator Decline. *Ecological Economics* 68(3): 810–821.
- Gray A, Adjlane N, Arab A, et al., 2020. Honey Bee Colony Winter Loss Rates for 35 Countries Participating in the COLOSS Survey for Winter 2018–2019, and the Effects of a New Queen on the Risk of Colony Winter Loss. *Journal of Apicultural Research* 59(5): 744–751.
- Gray A, Brodschneider R, Adjlane N, et al., 2019. Loss Rates of Honey Bee Colonies during Winter 2017/18 in 36 Countries Participating in the COLOSS Survey, Including Effects of Forage Sources. *Journal of Apicultural Research*: 1–7.
- IoBee | IoT Application, N.d. <https://io-bee.eu/>, accessed April 1, 2021.

- Klein AM, Vaissiere BE, Cane JH, et al. 2007. Importance of Pollinators in Changing Landscapes for World Crops. *Proceedings of the Royal Society of London B: Biological Sciences* 274(1608): 303–313.
- Milner R, 1978. A Theory of Type Polymorphism in Programming. *Journal of Computer and System Sciences* 17(3). Elsevier: 348–375.
- Rortais A, Arnold G, Dorne JL, et al., 2017. Risk Assessment of Pesticides and Other Stressors in Bees: Principles, Data Gaps and Perspectives from the European Food Safety Authority. *Science of The Total Environment* 587–588: 524–537.
- Rubinigg M, MacDonald M, Davenport V, et al. 2021. Predicting Varroa: Longitudinal Data, Micro Climate, and Proximity Closeness Useful for Predicting Varroa Infestations. *Data & Analytics For Good*(1). <https://data-for-good.pubpub.org/pub/aawkkv33>.
- Schmickl T and Crailsheim K, 2007. HoPoMo: A Model of Honeybee Intracolony Population Dynamics and Resource Management. *Ecological Modelling* 204(1–2): 219–245.
- Simon-Delso N, 2017. Fungicides and Bees: A Story of the Unexpected. Towards a Better Understanding of Inexplicable Bee Colony Disorders in the Walloon Region (Belgium). Prom.: Debier, Cathy; Bruneau, Etienne. Université Catholique de Louvain. <http://hdl.handle.net/2078.1/195698>.
- Simon-Delso N, San Martin G, Bruneau E, et al., 2017. The Challenges of Predicting Pesticide Exposure of Honey Bees at Landscape Level. *Scientific Reports* 7(1). Nature Publishing Group: 3801.
- van der Zee R, Pisa L, Andonov S, et al., 2012. Managed Honey Bee Colony Losses in Canada, China, Europe, Israel and Turkey, for the Winters of 2008-9 and 1009-10. *Journal of Apicultural Research and Bee World* 51(1): 100–114. <http://strathprints.strath.ac.uk/37337/>
- van der Zee R, Brodschneider R, Brusbardis V, et al., 2014. Results of International Standardised Beekeeper Surveys of Colony Losses for Winter 2012–2013: Analysis of Winter Loss Rates and Mixed Effects Modelling of Risk Factors for Winter Loss. *Journal of Apicultural Research* 53(1): 19–34.

Glossary

DataModel	An abstraction of the raw <i>Datatype</i> value saved within a database.
DataPivot	Model representing a Pivot between models: Log, Descriptor, DataModel and Origin.
Data point	A single value of a dataset. <i>Data points</i> are raw values which by itself have no real meaning, so they have to be described by <i>Descriptor Metadata</i> .
Datatype	In computer science and computer programming, a data type or simply type is an attribute of data which tells the compiler or interpreter how the programmer intends to use the data.
Descriptor	<i>Metadata</i> describing each data point by including information about data type and unit.
DescriptorCategory	Categorization of <i>Descriptor</i> model.
Log	A collection of <i>data points</i> , that are recorded on the same date/time and location.
Metadata	Data that provides description of other data.
Model	A simplified representation of a part of reality that contains mutually dependent elements.
Origin	Metadata on how the data was collected like Methodology or Device.
Pivot	Represents a connection between 2 or more Models. Used mainly in many to many relationships.
Post	Model representing <i>Post</i> code of a country.
Project	Also known as 'Data Source' <i>Metadata</i> . It is the direct source of the data.
Provider	Also known as "Data Owner". It is the actual owner of the data, which usually can be an institution, organization or even an individual. A single <i>Data Provider</i> can participate in several <i>Projects</i> , and share several <i>Datatypes</i> .
Reference library	A collection of data to reference the datasets with. Reference libraries are for example, lists of correspondences of pesticides names in different languages, or the toxicological profile of each pesticide active ingredient. Another example is a list of crop species with links to external descriptions (GBIF).

Abbreviations

API	Application Programming Interface
COLOSS	Prevention of honey bee COLony LOSSes
ECPA	European Crop Protection Association
EFSA	European Food Safety Authority
EUBP	European Bee Partnership
EU	European Union
FAO	Food and Agriculture Organization of the United Nations
FAOSTAT	Statistical service of the FAO
GBIF	Global Biodiversity Information Facilities
GPS	Global Positioning System
ID	Identification
ISO	International Organisation for Standardisation
LAU	Local Administrative Unit
LOD	Level of Detection
LOQ	Level of Quantification
MUST-B	EFSA project on the holistic risk assessment of MULTiple STressors in Bees
NUTS	Nomenclature of territorial units for statistics
PHP	Hypertext Preprocessor
PoC	Proof of Concept
SIGeC	Système intégré de gestion et de contrôle
SSD2	Standard Sample Description 2
UML	Unified Modelling Language
UTC	Coordinated Universal Time
WG	Working Group
WGS84	World Geodetic System 1984

Appendix A – Information on the data providers and datasets available in the EUBP prototype platform

Data provider	About the provider	Phase of integration	Project	About the project	Implementation	Implemented Descriptors	Descriptors to be implemented
Biene Österreich and Ländliches Fortbildungsinstitut Österreich	https://bee-ppp.eu/providers/biene-oesterreich-im-kereidachverband and https://bee-ppp.eu/providers/laendliches-fortbildungsinstitut-oesterreich	Proof of Concept (PoC)	VarroaAlert	https://bee-ppp.eu/projects/varroaalert-varroa-warndienst	Implementation includes daily API query to bienengesundheit.at to get infestation warnings and regional infestation data. Data is available per district (NUTS2).	Regional Infestation alerts, infestation counts and statistics	None
CARI	https://bee-ppp.eu/providers/cariasbl-bee-keeping-center-of-research-and-information	PoC	National Bee-keeping Programmes	https://bee-ppp.eu/projects/national-beekeeping-programme	Implementation includes daily API query to cari.be and historical data since 2012. Data is available from colonies in various locations of the Walloon region (Belgium).	Hive scales weight readout every 2 hours Hive sensors data: in-hive temperature Weather sensors data (outside the hive): • Temperature • Relative humidity • Rain	None
CARI, CRA-W and Honey-Bee Valley	CARI - https://bee-ppp.eu/providers/cariasbl-bee-keeping-center-of-research-and-information	PoC	BeeWallonie and Flemish bee-keeping programme	https://bee-ppp.eu/projects/prevention-of-honey-bee-colony-losses and https://bee-ppp.eu/projects/prevention-of-honey-bee-colony-losses	Implementation includes import of several Excel databases, one for each year (2015 - 2020) within Belgium. Data is available for postal code (precision level 6) within Belgium.	Autumn and spring counts of number of alive colonies Colony mortality count by natural disasters Colony mortality count by queen problems (e.g. drone laying, supersedure, etc.)	There is a possibility to use counts that would give insights into the possible reasons for the observed colony losses, but data quality is not good enough.

	Honeybee Valley https://bee-ppp.eu/providers/honeybee-valley Centre de Recherches Agronomiques - Wallonie https://bee-ppp.eu/providers/cra-w			ppp.eu/projects/national-beekeeping-programs-vlaanderen-be		Colony mortality count by number of colonies found either depopulated (empty hive) or with dead bees	
ANSES	https://bee-ppp.eu/providers/anses-european-reference-laboratory-for-bees	PoC	Epilobee	https://bee-ppp.eu/projects/honey-bee-winter-mortality-2012-2014-epilobee	Implementation includes import of a single Excel database. Data is available for various European Member States at country level	Winter mortality of honey bee colonies	None
World Geodetic System 1984 and Nomenclature of territorial units for statistics within EU	-	PoC	-	-	Implementation includes a collection of .gejson files, which include geographical data related to country, region or district (all NUTS levels and some LAU) boundaries and geographical centers compatible with NUTS 2016 / LAU 2019. Additionally there is a collection of geographical centers for most Postal codes.	Human population and surface size for most local administrative units (LAU) and NUTS regions levels 0 – 3 of European countries.	

Nuno Xavier Jesus Capela	https://bee-ppp.eu/providers/nuno-xavier-jesus-capela	Prototype	MUST-B	https://bee-ppp.eu/projects/must-b-field-data-collection-for-honey-bee-colony-model-evaluation	Implementation includes import of several Excel files with historical data from 2019 until 2020. Data is available from 30 colonies in 2 EU Member States (Portugal and Denmark) but only implemented for Portugal	<p>Hive scales weight readout every 2 hours</p> <p>Hive management and sampling information</p> <p>Hive sensors data:</p> <ul style="list-style-type: none"> • In-hive temperature • In-hive relative humidity <p>Weather sensors data (readout every 15 minutes):</p> <ul style="list-style-type: none"> • Temperature • Relative humidity • Rain • Dew point • Wind speed • Wind direction • Wind gust • Solar radiation <p>Colony development data (readout every two weeks: nr. of bees, nr. of cells with brood, nr. of cells with pollen) along two seasons.</p> <p>Hive material's weight</p>	<p>Danish data</p> <p>Various data sampling related to and around the colonies:</p> <ul style="list-style-type: none"> • Pollen composition of the pollen brought back by bees • Pesticide analysis of the samples: beebread, Land usage around the apiary (1.5 km radius from the hives)
-------------------------------------	---	-----------	--------	---	--	--	---

Publications about winter mortality from the COLOSS network: Brodschneider et al. 2018, 2016, Gray et al., 2019, 2020, van der Zee et al., 2012, 2014 (EU-World wide data)	Example of one of the publications: https://bee-ppp.eu/providers/brodschneider-et-al-2016	Prototype	COLOSS country level	https://bee-ppp.eu/projects/prevention-of-honey-bee-colony-losses-2015	Implementation includes import of a merged Excel databases with country level data between 2008 and 2017	<ul style="list-style-type: none"> • Number of correspondents • Total number of alive colonies • Mean winter loss rate in percentages • Mean winter loss lower limit (CI 95%) • Mean winter loss upper limit (CI 95%) 	
---	--	-----------	----------------------	---	--	--	--

Simon-Delso et al., 2017	https://bee-ppp.eu/providers/simon-delso-et-al-2017	Prototype	DEPAB - Experimental approach to unexplained mortality of bee colonies in Wallonia region, Belgium	https://bee-ppp.eu/projects/experimental-approach-to-unexplained-mortality-of-bee-colonies-in-wallonia	Implementation includes several .csv files with raw data. Data is available for the duration of the project in several colonies within Belgium	Individual sampled colonies with following information: <ul style="list-style-type: none"> • Land usage around the apiary (3 km radius from the apiary) • Pollen composition of the samples • Pesticide analysis of the samples 	Imported libraries: <ul style="list-style-type: none"> • Pesticide reference library • Pollen botanical origin reference library • Land usage reference library for Belgium
Klein et al., 2007	https://bee-ppp.eu/providers/klein-et-al-2007	Prototype	ALARM Project	https://bee-ppp.eu/projects/project-alarm-goce-ct-2003-506675	Implementation includes several .csv files with raw data.	Pollination dependency of crops	None
Rubinigg et al., 2021	https://bee-ppp.eu/providers/rubinigg-et-al-2021	Prototype	-	-	Implementation includes several .csv files with raw data.		An eight-year survey of <i>Varroa destructor</i> infestation rates of honey bee colonies with the spatial and temporal dimensions and weather factors that impact these infestation rates

Corteva	https://www.corteva.com And https://beeppp.eu/providers/corteva	Prototype	-	Corteva feeding study with colony development https://beeppp.eu/projects/corteva-feeding-study-with-colony-development	Implementation includes two Excel files and a .doc file explaining the methodology and providing the metadata	Hive scales weight readout once per day at 11.30 pm Weekly colony assessment (nr. of bees, nr. of cells with brood, nr. of cells with food). All data is collected for 5 apiaries between 25 May and 12 Oct 2016 and between 23 May and 30 Oct 2018	None
FAOSTAT	https://beeppp.eu/providers/fao	Prototype	Socioeconomic data related to beekeeping and agriculture	-	Implementation includes import of historic data up to 2018 from FAO-STAT as collection of .csv files.	In EU27 Member States: number of beehives per country; Crops requiring pollination, their production and value; Volume of honey production; Value of honey production, Volume of honey trade, Volume of wax production, Volume of wax trade, Value of wax production	Import latest (2019) data.
BeeLife European Beekeeping Coordination	https://beeppp.eu/providers/beelife-european-beekeeping-coordination and http://bee-life.eu	Prototype	-	-	Excel files and metadata		Outbreaks of European Foulbrood and American Foulbrood (Belgium)

Biene Österreich	https://bee-ppp.eu/providers/biene-oester-reich-im-kereidachverband	Prototype	-	-	Excel files and metadata		Outbreaks of European Foulbrood and American Foulbrood (Austria)
dr. Mag. rer. nat. Michael Rubinigg	https://bee-ppp.eu/providers/dr-mag-rer-nat-michael-rubinigg	Prototype	-	-	Excel files and metadata	Colony developmental data (nr. of bees, nr. of cells with brood)	

Appendix B – Detailed description of the data models for the different Data types considered in the EUBP Prototype Platform

In addition to the information included in section 4.5.1, this appendix describes in detail the fields and attributes of each data model considered within the EUBP Prototype Platform.

B.1. Generic fields

This section describes fields that are repeated across many models.

B.1.1. Id field

The Id or identifier field is an integer used in most models as a unique representation of the model within a table. It is used by the database to create various relationships between models, as well as enabling editing and deletion of models. It is always automatically generated by the database, always starts at 1 and is auto incremented for each new row.

B.1.2. Uid field

The Uid or universal identifier is developer generated, non-editable field of text in length up to 100 characters. Primary role of the field is to help the developer reference various models while working with the database, as referencing Ids might not be reliable enough when developing in a local environment and subsequently running scripts on a production environment. Uid should never be displayed to the front-end users.

B.1.3. Slug field

It represents part of a URL which identifies a particular page on a website in an easy to read form. It is commonly used on the platform with models that require their own reference pages with a link. The link is then generated based on the model type and language with the addition of the Slug field. For example, the details of a Hive scales Data type can be found on <https://bee-ppp.eu/data/hive-weight>, where the "https://bee-ppp.eu" represents the domain of the platform, the "/data/" represents localized "location" of the Data type model and the "/hive-weight" represents the Slug that is saved in the database.

B.1.4. Name field

It represents user-friendly naming of the model, which is usually displayed on the front-end. It is always required and must be at most 191 characters long.

B.1.5. Description field

It represents a longer description of the model that can be edited with a visual "what you see is what you get" editor. It can contain HTML elements to further style the text and can be up to 16,777,215 characters long.

B.1.6. Public field

The field represents a flag, either the model should be displayed on the front-end. Sometimes we want to hide certain models, either they are not being used or are used internally. For that we use this flag to set it either as 1 (true, visible) or 0 (false, hidden).

B.1.7. Created at field

A date and time field representing the exact moment a model was first created in the database. It is always created automatically when the model is created.

B.1.8. Updated at field

A date and time field representing the exact moment a model was last changed in the database. It is always updated when there are any changes to the model.

B.1.9. Featured image field

A field containing the url to the uploaded image that represents the model.

B.1.10. Icon field

A field containing the url to the uploaded icon that represents the model on certain occasions.

B.2. Individual Data models

As individual models vary a lot in structure, here are additional details about them.

B.2.1. Metadata

Models related to models representing metadata.

B.2.1.1. Project model

The "Project model" describes the main characteristics of the project (by whom and within what project has the data been created and processed) and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Acronym	String [varchar(50)], null	Up to 50 characters.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Website	String [varchar(191)], null	Up to 191 characters.
Phone	String [varchar(191)], null	Up to 191 characters.
Email	String [varchar(191)], null	Up to 191 characters.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Active	Boolean [tinyint]	If true, it synchronizes data.
Public	Boolean [tinyint]	If true, it is displayed on the website.
Type	String [varchar(8)]	Can be either live, static or remote.
Last synced at	Datetime [timestamp], null	
Sync period	String [varchar(8)], null	Can be either null, hourly, daily, weekly, monthly, quarterly or yearly.
Historic Sync	Boolean [tinyint], null	If set, historic data exists. If true, it is synced to the EUBP Prototype Platform.
UID	String [varchar(100)]	Up to 100 characters.
Started at	Datetime [timestamp], null	
Ended at	Datetime [timestamp], null	
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.
Deleted at	Datetime [timestamp]	Autogenerated, when soft-deleted.

It also contains the following relations with other models:

- Many to Many relation to *Provider*
- Many to Many relation to *Descriptor*
- Many to One relation to *DataAggregate*
- One to Many relation to *DataId*

Additional field explanations

Acronym - a shorter version of the name.

Website - an url as link to the projects website.

Phone - a telephone number that could be used to contact the project manager.

Email - an email address that could be used to contact the project manager.

Active - whether we allow continuous data synchronization via an API (if available). Can be used to disable the process due to errors.

Type - internal representation of project, which can be live (is regularly querying live data via an API), static (the data is contained within local files) or remote (the data is hosted remotely).

Last synced at - The exact date and time the project was last updated from a remote or live source.

Historic Sync - If true, the project has already received any historical data, if false, the process is not yet started, if null,

Started at - optional date a project has started

Ended at - optional date a project has ended

B.2.1.2. Provider model

The "Provider model" describes the characteristics of the provider (i.e. institutions, working groups or individuals that are directly providing any kind of data to the platform) and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Full name	String [varchar(191)], null	Up to 191 characters.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Registry number	String [varchar(192)], null	Up to 191 characters, can contain any kind of national registration number.
Website	String [varchar(191)], null	Up to 191 characters.
Phone	String [varchar(191)], null	Up to 191 characters.
Email	String [varchar(191)], null	Up to 191 characters.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Address	String [text], null	Up to 16,777,215 characters, should not contain HTML.
Post	Post [bigint], null	Relates to Post object.
Public	Boolean [tinyint]	If true, it is displayed on the website.

UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.
Deleted at	Datetime [timestamp]	Autogenerated, when soft-deleted.

It contains the following relations with other models:

- Many to Many relation to *Project*
- One to Many relation to *Post*

Additional field explanations

Full Name - a longer (sometimes official) version of the name.

Registry Number - an official registry number, required only for legal entities.

Website - an url as link to the projects website.

Phone - a telephone number that could be used to contact the project manager.

Email - an email address that could be used to contact the project manager.

Post - an address related Post id, used to identify country and location of the provider.

Address - The address of the provider, required only for legal entities.

B.2.1.3. Descriptor model

The "Descriptor model" describes the characteristics of the model (i.e. which data type it is, its unit and description) and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Public	Boolean [tinyint]	If true, it is displayed on the website.
Since	String [varchar(10)]	Up to 10 characters, version information
Deprecated	Boolean [tinyint]	If true, the data from this Descriptor is not being used anymore and is there for historical reasons.
Category	DescriptorCategory [bigint]	Relates to Descriptor category.
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- Many to Many relation to *Project*
- One to Many relation to *Data*
- Many to Many relation to *Log*
- Polymorphic Many to Many relation to *DataModel*

- Many to one relation to *DescriptorCategory*

Additional field explanations

Since - version information, when the Descriptor was added.

Deprecated - Similar to soft-deletes but Descriptors should never be considered deleted. The deprecated value represents a model that exists and is used in the past but should no longer be used.

Category - Represents relation to Descriptor Category, via id.

Table_name - Used internally to identify, in which database table data is stored for current Descriptor.

B.2.1.4. DescriptorCategory model

The "DescriptorCategory model" describes the characteristics of the category of the model and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Parent	DescriptorCategory [bigint], null	Relates to a <i>DescriptorCategory</i> parent.
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

The DescriptorCategory model contains the following relations with itself and with other models:

- One to Many relation to *Descriptor*
- Many to one relation to parent *DescriptorCategory*
- One to many relation to children *DescriptorCategory*

Additional field explanations

Parent- Represents relation to model in its own table, Descriptor Category, via id. A null parent is the highest level Descriptor Category available in the platform.

B.2.2. Location Metadata

B.2.2.1. Country model

The "Country model" describes the characteristics of the country from which the data were collected and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .

Name	String [varchar(50)]	Up to 50 characters, required field.
Country code	String [varchar(2)]	Up to 2 characters, ISO country code.
Iso3	String [varchar(3)]	Up to 3 characters, ISO3 country code.
Native name	String[varchar(100)]	Up to 100 characters.
Capital	String[varchar(30)]	Up to 30 characters.
Continent	String[varchar(2)]	Up to 2 characters. One of AF, AN, AS, EU, NA, OC or SA.
Currency	Currency [bigint]	Relates to Currency model.
Phone prefix	String[varchar(20)]	Up to 20 characters.

It contains the following relations with other models:

- One to Many relation to *Region*
- One to One relation to NUTS level 0 Region of corresponding country
- One to Many relation to *Post*
- One to Many relation to LAU
- Many to one relation to Currency

Additional field explanations

Country code - ISO 3166-1 alpha-2, a two letter country code representation

Iso3 - ISO 3166-1 alpha-2, a three letter country code representation

Native name - Country name in native language

Capital - The capital city of the country

Continent - The continent the country resides in, can be either AF - Africa, AN - Antarctica, AS - Asia, EU - Europe, NA - North America, OC - Oceania, SA - South America.

Currency - relation to the Currency model in order to identify the currency used in the country.

Phone prefix - international phone number prefix

B.2.2.2. Region model

The "Region model" describes the characteristics of the region from which the data were collected and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(50)]	Up to 50 characters, required field.
Country	Country [bigint]	Relates to Country.
Nuts ID	String [varchar(5)]	Up to 5 characters, official region NUTS ID.
Nuts level	Integer [smallint]	A number between 0 and 3, corresponding to the official NUTS region level.
Parent	Region [bigint], null	Relates to Region parent of 1 nuts level higher.
Population	Integer [bigint], null	
Surface	Integer [bigint], null	In square meters.

Center	Point [point]	Geographical centre of the region in WGS86.
Area	Geometry [Geometry]	A list of points in WGS86, surrounding the region.

It contains the following relations with itself and other models:

- Many to one relation to *Country*
- Many to one relation to parent *Region*
- One to many relations to children *Regions*
- One to many relations to LAU
- One to many relations to *Post*
- Many to many polymorphic relations with *Log*

Additional field explanations

Country - Relates to the Country model, that represents the country the region is situated in.

Nuts ID - The id based on nomenclature of territorial units for statistics within EU

Nuts level - The precision level the region represents within Nuts. 0 for country, 1 for major socio-economic regions, 2 for small regions of the application of regional policies and 3 for small regions for specific diagnoses

Parent - The Nuts nomenclature contains hierarchy, this relation describes the parent, current region relates to. The parent is always one level above the current model level.

Population - Population of the area in 2019.

Surface - The size in square meters of the surface in 2019.

Centre - WGS86 coordinates of geographical centre of the model.

Area - A list of geometries describing the boundaries of the model in WGS86 coordinates.

B.2.2.3. LAU model

The "LAU model" describes the Local Administrative Units (LAU) from which the data were collected and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(150)]	Up to 150 characters, required field.
Country	Country [bigint]	Relates to Country.
Lau ID	String [varchar(13)]	Up to 13 characters, official LAU ID.
Nuts ID	String [varchar(5)]	Up to 5 characters, official region NUTS ID.
Population	Integer [bigint], null	
Surface	Integer [bigint], null	In square meters.
Center	Point [point]	Geographical centre of the region in WGS86.
Area	Geometry [Geometry]	A list of points in WGS86, surrounding the region.

It contains the following relations with other models:

- Many to one relation to *Country*
- Many to one relation to Region with NUTS level 3
- One to many relations to *Post*
- Many to many polymorphic relations with *Log*

Additional field explanations

Country - Relates to the Country model, that represents the country the region is situated in.

Nuts ID - The related region represented as parent to this model.

Lau ID - The national ID of the local administrative unit.

Population - Population of the area in 2019.

Surface - The size in square meters of the surface in 2019.

Centre - WGS86 coordinates of geographical centre of the model.

Area - A list of geometries describing the boundaries of the model in WGS86 coordinates.

B.2.2.4. Post model

The "Post model" describes the postal code from which the data were collected and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(50)]	Up to 50 characters, required field.
Number	String [varchar(15)]	Up to 15 characters, required field.
Country	Country [bigint]	Relates to Country.
Lau ID	String [varchar(13)]	Up to 13 characters, official LAU ID.
Nuts ID	String [varchar(5)]	Up to 5 characters, official region NUTS ID.
Population	Integer [bigint], null	
Surface	Integer [bigint], null	In square meters.
Center	Point [point]	Geographical centre of the region in WGS86.
Area	Geometry [Geometry]	A list of points in WGS86, surrounding the region.

It contains the following relations with other models:

- Many to one relation to *Country*
- Many to one relation to Region with NUTS level 3
- Many to one relation to LAU
- Many to many polymorphic relations with *Log*
- One to many relations to *Provider*

Additional field explanations

Number - The number of the model, may contain numbers as well, depending on national standards.

Country - Relates to the Country model, that represents the country the region is situated in.

Lau ID - The related local administrative unit as parent to this model.

Nuts ID - The related region represented as backup parent to this model. If Lau ID is not available, this is used as the actual parent.

Population - Population of the area in 2019.

Surface - The size in square meters of the surface in 2019.

Centre - WGS86 coordinates of geographical centre of the model.

Area - A list of geometries describing the boundaries of the model in WGS86 coordinates.

B.2.3. Origin Metadata

B.2.3.1. Methodology model

The "Methodology model" describes the characteristics of the methodology used to generate the data and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Website	String [varchar(192)], null	Up to 191 characters.
Category	Category [bigint]	Relates to <i>MethodologyCategory</i> .
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with itself and other models:

- One to many relations to *DataPivot*
- Many to one relation to *MethodologyCategory*

Additional field explanations

Website - an url as link to the methodology website.

Category - The category that this model belongs to.

B.2.3.2. MethodologyCategory model

The "MethodologyCategory" model describes the category of the methodology used to generate the data and contains the following fields:

Field name	Datatype [Database type]	Limitations
------------	--------------------------	-------------

Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relations to *Methodology*

B.2.3.3. Device model

The "Device model" describes the characteristics of the device used to collect the data and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Category	Category [bigint]	Relates to <i>MethodologyCategory</i> .
Website	String [varchar(191)], null	Up to 191 characters.
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.
Vendor	Vendor [bigint]	Relates to device Vendor.
Introduced at	Datetime [timestamp]	

It contains the following relations with other models:

- One to many relations to *Data*
- Many to one relation to *DeviceCategory*

Additional field explanations

Website - an url as link to the methodology website.

Category - the category that this model belongs to.

Vendor - the vendor (manufacturer) this device belongs to.

Introduced at - The date this device was introduced to market. A single device could be created in several versions under the same name but with different specifications.

B.2.3.4. DeviceCategory model

The "DeviceCategory model" describes the characteristics of category of the device used to generate the data and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relation to *Device*

B.2.3.5. DeviceVendor model

The "DeviceVendor model" describes the characteristics of the vendor of the device or the versions of a single device used to generate the data and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Slug	String [varchar(191)]	Up to 191 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Website	String [varchar(192)], null	Up to 191 characters.
UID	String [varchar(100)]	Up to 100 characters.
Created at	Datetime [timestamp]	Autogenerated, when created
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains following relations with other models:

- One to many relations to *Device*

Additional field explanations

Website - an url as link to the methodology website.

B.2.4. Reference libraries

B.2.4.1. CurrencyReference model

The "CurrencyReference model" model is based on the "Currency reference model". It represents a value from the database of currencies, which lists all known currencies (based on ISO 4217) and a relation with countries using the currency. It contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(191)]	Up to 191 characters, required field.
Code	String [varchar(191)]	Up to 191 characters, required field.
Number	Integer [int]	Accepts values between 0 to 2.147.483.647, required field.
Decimals	Integer [int], null	Accepts values between 0 to 2.147.483.647.
Symbol	String [varchar(5)], null	Up to 5 characters.
Symbol prefix	Boolean [tinyint]	Accepts values between 0 and 1.
Created at	Datetime [timestamp]	Autogenerated, when created
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relations to *LandUseRModel*

Additional field explanations

Code - ISO 4217 alphabetic currency code.

Number - ISO 4217 numeric currency code.

Decimals - number of decimal places the currency uses.

Symbol - the symbol a currency uses.

Symbol Prefix - whether the symbol is before (true) or after (false) the value.

B.2.4.2. LandUseReference model

The "LandUseReference model" is based on the "LandUse data model" obtained from the publication Simon-Delso et al. (2017). This model represents a value from the database of land use data (references to various crops occupying different agricultural fields each year or the type of land use existing on a certain area and year) and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(200)]	Up to 200 characters, required field.
Group	String [varchar(100)]	Up to 100 characters, required field.

Sigec Code	Integer [int], null	A standard Sigec code reference.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Dim Min Residues	Integer[int], null	Accepts values between 0 to 2.147.483.647.
Dim Max Residues	Integer[int], null	Accepts values between 0 to 2.147.483.647.
Pyr Min Residues	Integer[int], null	Accepts values between 0 to 2.147.483.647.
Pyr Min Residues	Integer[int], null	Accepts values between 0 to 2.147.483.647.
Bos Min Residues	Integer[int], null	Accepts values between 0 to 2.147.483.647.
Bos Min Residues	Integer[int], null	Accepts values between 0 to 2.147.483.647.
Source	String [varchar(100)]	Up to 100 characters.
Deprecated	Boolean [tinyint]	Accepts only 0 or 1.
Created at	Datetime [timestamp]	Autogenerated, when created
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relations to *LandUseModel*

Additional field explanations

Group - grouping of the Sigec code in major categories

DimMinRes and DimMaxRes - theoretical maximum and minimum amount of Dimethoate residues possible on that crop according to the authorized uses, doses and number of treatments

PyrMinRes and PyrMaxRes - theoretical maximum and minimum amount of Pyrimetanyl residues possible on that crop according to the authorized uses, doses and number of treatments

BosMinRes and BosMaxRes - theoretical maximum and minimum amount of Boscalid residues possible on that crop according to the authorized uses, doses and number of treatments

B.2.4.3. PesticidesReference model

The "PesticidesReference model" represents a value from the database of pesticide data (reference library of pesticides containing names of pesticide active ingredients in English and French found in honey bee matrices) obtained from the publication Simon-Delso et al. (2017). It contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into <i>Database</i> .
Name	String [varchar(100)]	Up to 100 characters, required field.
Substance	String [varchar(100)]	Up to 100 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Pesticides Type	PesticidesTypeReference Model [bigint]	Relates to PesticideType
Deprecated	Boolean [tinyint]	Accepts only 0 or 1.

Created at	Datetime [timestamp]	Autogenerated, when created
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relations to *PesticidesReference*
- Many to One relation to *PesticidesType*

Additional field explanations

Substance - name as used in referenced dataset

Pesticides Type - refers to the PesticideType model.

Deprecated - Similar to soft-deletes but Descriptors should never be considered deleted. The deprecated value represents a model that exists and is used in the past but should no longer be used.

B.2.4.4. PesticideTypeReference model

The "PesticidesTypeReference model" represents a value from our database of pesticide types data (reference library of the pesticides types, based on the mode of action of pesticides (insecticides, fungicides, herbicides, etc.) and it completes the previous one) obtained from the publication Simon-Delso et al. (2017). It contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(100)]	Up to 100 characters, required field.
Abbreviation	String [varchar(2)]	Up to 2 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Deprecated	Boolean [tinyint]	Accepts only 0 or 1.
Created at	Datetime [timestamp]	Autogenerated, when created
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relations to *PesticidesModel*
- Many to one relation to *PesticidesTypeReference*

B.2.4.5. PollenOriginReference model

The "PollenOriginReference model" represents a value from the database of Pollen data (reference library representing botanical pollen types with information about their dimensions, taxonomic classification, etc.) obtained from the publication Simon-Delso et al. (2017). It contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(100)]	Up to 100 characters, required field.

Type	String [varchar(100)]	Up to 100 characters, required field.
Family	String [varchar(100)]	Up to 100 characters, required field.
Code	String [varchar(10)]	Up to 10 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.
Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Polar size	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
Equatorial size	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
Ellipsoid volume	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
Mean radius	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
Volume by Radius	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
Deprecated	Boolean [tinyint]	Accepts only 0 or 1.
Created at	Datetime [timestamp]	Autogenerated, when created
Updated at	Datetime [timestamp]	Autogenerated, when created or edited.

It contains the following relations with other models:

- One to many relations to *PollenOriginModel*

Additional field explanations

Type - clean pollen categories to be used (with some groupings).

Family - Botanical family of the pollen species.

Code - short code for Pollen_type.

Polar size - polar size effectively used (for ranges we used the mean of the extreme values).

Equatorial size - equatorial size.

Ellipsoid volume - volume of the ellipsoid calculated from r & q as $\frac{4}{3} * r/2 * (q/2)^2$.

Mean radius - mean radius used to calculate the volume as a sphere.

Volume by Radius - volume used based on r.

Deprecated - Similar to soft-deletes but Descriptors should never be considered deleted. The deprecated value represents a model that exists and is used in the past but should no longer be used.

B.2.4.6. CropSpeciesReference model

The "CropSpeciesReference model" describes the reference related to the crop species data and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Name	String [varchar(100)]	Up to 100 characters, required field.
Description	String [text], null	Up to 16,777,215 characters, can contain HTML.

Featured image	String [varchar(512)], null	Up to 512 characters, autogenerated.
Icon	String [varchar(512)], null	Up to 512 characters, autogenerated.
Species name	String [varchar(100)]	Up to 100 characters.
GBIF id	String [varchar(10)]	Up to 10 characters.
Parent	CropSpecies [bigint]	Relates to itself.
Created at	Datetime [timestamp]	Autogenerated, when created.
Updated at	Datetime [timestamp]	Autogenerated, when created. NULL if no update exists.

It contains the following relations with other models:

- Many to one relation to *CropSpecies*
- One to one relation to *CropSpeciesReference*
- One to one relation to *CropPollinationDependencyReference*

Additional field explanations

Species name - Scientific name of the crop species.

GBIF id- Relates to identifiers of the species at GBIF (<https://www.gbif.org>).

Parent- relates to itself.

B.2.4.7. CropPollinationDependencyReference model

The "CropPollinationDependencyReference model" describes the reference related to the crop pollination dependency data provided by Klein et al. (2017) and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Crop Species	Integer [bigint]	Relates to Item model
Impact	String [varchar(100)]	Relates to CropSpecies model
Dependence from	Integer [bigint]	Required field.
Dependency to	Integer [bigint]	Required field.
CreatedAt	Datetime [timestamp]	Autogenerated, when created.
UpdatedAt	Datetime [timestamp]	Autogenerated, when created. NULL if no update exists.

It contains the following relations with other models:

- One to one relation to *Item*
- One to one relation to *CropSpecies*
- One to one relation to *Unit*
- One to one relation to *References*
- One to one relation to *Annotation*
- One to one relation to *PollinationDependency*

Additional field explanations

Crop Species - relates to the CropSpecies model.

Impact - Brief summary of pollination impact on crop.

Dependency from - to - lower and upper range of pollination dependency of crop.

B.2.5. The Data included into the EUBP Prototype Platform

B.2.5.1. Log model

The “Log model” represents a group of data points created at the same time and place and contains the following fields:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once inserted into Database.
Year	Datetime [year]	Required field.
Date	Datetime [date], null	
Time	Datetime [time], null	
Project	Project [bigint]	Relates to Project, required.
Created at	Datetime [timestamp]	Autogenerated, when created.
Apiary	Data Id Model [bigint, null]	Relates to DataId
Hive	Data Id Model [bigint, null]	Relates to DataId

It contains the following relations with other models:

- Many to one relation to *Project*
- One to many relations to *Data*
- Many to many relation to *DataModel*, via *DataPivot*
- Many to many relations to *Descriptor*, via *DataPivot*
- Many to many relations to *Origin*, via *DataPivot*
- Many to one relation with *DataId* (Apiary)
- Many to one relation with *DataId* (Hive)

Additional field explanations

Year - represents the year, the datapoint relates to. Should always be present, even if the date is present, it should point to the same year.

Date - optional date the datapoint relates to.

Time - optional time information, the datapoint relates to.

Project - Related Project the datapoint was created in.

Apiary - Optional related Apiary ID, that this datapoint relates to.

Hive - Optional related Hive ID, that this datapoint relates to.

B.2.5.2. Data model

The “Data model” represents a relationship of each data point value (DataModel) to Descriptor (what it is) and Origin (how it was created) and contains the following fields:

Field name	Datatype [Database type]	Limitations
Log	Log [bigint]	Relates to Log, required.
Descriptor	Descriptor [bigint]	Relates to Descriptor, required.
Data	Data Model [bigint]	Relates to DataModel, which is a datapoint value.
Origin	Origin [bigint], null	Relates to Origin.

It contains the following relations with other models:

- Many to one relation to Log
- Many to one relation to DataModel
- Many to one relation to Descriptor
- Many to one relation to Origin

Additional field explanations

Descriptor - relates to the Descriptor model that describes this data point.

Data - relates to the data value, this datapoint contains.

Origin - relates to the Origin meta data, of how the datapoint was created.

B.2.5.3. DataModel model

Data Model is an abstract model representing a data point value. Using polymorphism we support the Scalar and Compound Types (i.e. made up of several individual Scalar Types).

Reference types, which reference our internal database

The following table displays attributes, that are shared between all individual DataModels:

Field name	Datatype [Database type]	Limitations
Id	Integer [bigint]	Autogenerated once it's inserted into the Database.

It contains the following relations with other models:

- One to many relations to DataPivot
- Many to many relations to Log, via DataPivot
- Many to many relations to Descriptor, via DataPivot
- Many to many relations to Origin, via DataPivot

The following tables represent individual DataTypes and their attributes, in addition to what is described above.

DataBoolean

The DataBoolean model represents a true or false data point value. It can also be used as a "yes - no", "on-off" or any other opposite representation.

Field name	Datatype [Database type]	Limitations
Value	Boolean [tinyint]	Accepts only 0 or 1.

DataInteger

The DataInteger model represents a whole number data point value.

Field name	Datatype [Database type]	Limitations
Value	Integer [bigint]	Accepts values between -9223372036854775808 to 9223372036854775807.

DataPercentage

The DataPercentage model represents a percentage data point value.

Field name	Datatype [Database type]	Limitations
Value	String [decimal]	Accepts values between -99.999.999,999.99 - 99.999.999,999.99 but is usually between 0 and 100.

DataDecimal

The DataDecimal model represents a finite decimal number data point value.

Field name	Datatype [Database type]	Limitations
Value	String [decimal]	Accepts positive and negative values with up to 40 significant digits and 20 digits following the decimal point.

DataString

The DataString model represents a short text with any characters.

Field name	Datatype [Database type]	Limitations
Value	String [varchar(200)]	Up to 200 characters.

DataText

The DataText model represents a long text with any characters.

Field name	Datatype [Database type]	Limitations
Value	String [text]	Up to 16.777.215 characters.

DataDate

The DataDate model represents a single date.

Field name	Datatype [Database type]	Limitations
Value	Datetime [date]	In format YYYY-MM-DD.

DataTime

The DataTime model represents a single time data point value, with up to seconds precision. It can represent both a specific point in time or a length.

Field name	Datatype [Database type]	Limitations
------------	--------------------------	-------------

Value	Datetime [time]	In format hhh-mm-ss, allows values between -838:59:59 to 838:59:59.
-------	-----------------	---

DataDatetime

The DataDatetime model represents a single date and time data point value. It can represent both, specific moment in time and date or a duration with precision of up to seconds.

Field name	Datatype [Database type]	Limitations
Value	Datetime [time]	In format YYYY-MM-DD hh-mm-ss.

DataIntegerRange

The DataIntegerRange model represents a from-to whole number data point value.

Field name	Datatype [Database type]	Limitations
Value from	Integer [bigint]	Accepts values between - 9.223.372.036.854.775.808 to 9.223.372.036.854.775.807. Must be smaller than Value to.
Value to	Integer [bigint]	Accepts values between - 9.223.372.036.854.775.808 to 9.223.372.036.854.775.807. Must be larger than Value from.

DataDecimalRange

The DataDecimalRange model represents a from-to finite decimal number data point value.

Field name	Datatype [Database type]	Limitations
Value from	String [decimal]	Accepts positive and negative values with up to 40 significant digits and 20 digits following decimal point. Must be smaller than Value to.
Value to	String [decimal]	Accepts positive and negative values with up to 40 significant digits and 20 digits following decimal point. Must be larger than Value from.

DataLocation

The DataLocation model represents a location of the data point collection or as a value. It can use geographic coordinates or reference to a location, or both.

Field name	Datatype [Database type]	Limitations
Value	Point [point], null	Accepts WGS84 coordinates.
Location	Location [bigint]	Relates to any Location.

In addition to the Data Model, it contains following relations:

- Many to one relation to any Location (can be Post, LAU or Region).

Additional field explanations

Location - relates to any of the Location precision sub-models that this datapoint represents.

DataID

The DataId model represents a unique identifier of the datapoint. A single datapoint can have multiple ids. It accepts a short sequence of any characters and is unique to project.

Field name	Datatype [Database type]	Limitations
Value	String [varchar(192)]	Up to 192 characters.
Project	Project [bigint]	Relates to Project.

In addition to the Data Model, it contains following relations:

- Many to one relation to Project

Additional field explanations

Project - relates to a Project this ID should be Unique for.

Data Event

The DataEvent model represents an event that was logged within an apiary or hive.

Field name	Datatype [Database type]	Limitations
Value	String [decimal]	Accepts positive and negative values with up to 40 significant digits and 20 digits following the decimal point.
Unit	string [varchar(10)]	Up to 10 characters, can be empty.
Description	String [text]	Up to 16.777.215 characters, can be empty.

Additional field explanations

Unit - relates to any unit the value should be represented in.

DateDateRange

The DataDecimalRange model represents a from-to date data point value.

Field name	Datatype [Database type]	Limitations
Value_from	Datetime [date]	In format YYYY-MM-DD
Value_to	Datetime [date]	In format YYYY-MM-DD

DataTimeRange

The DataDecimalRange model represents a from-to time data point value.

Field name	Datatype [Database type]	Limitations
Value_from	Datetime [time]	In format hhh-mm-ss, allows values between - 838:59:59 to 838:59:59.
Value_to	Datetime [time]	In format hhh-mm-ss, allows values between - 838:59:59 to 838:59:59.

DataDatetimeRange

The DataDecimalRange model represents a from-to date and time data point value.

Field name	Datatype [Database type]	Limitations
Value_from	Datetime [time]	In format YYYY-MM-DD hh-mm-ss.
Value_to	Datetime [time]	In format YYYY-MM-DD hh-mm-ss.

Following models represent the relationship of the data with reference libraries.

CurrencyModel

The currency model represents data point values in different currencies.

Field name	Datatype [Database type]	Limitations
Value	String [decimal]	Accepts positive and negative values with up to 40 significant digits and 20 digits following the decimal point.
Currency	CurrencyReference [bigint]	Relates to CurrencyReference

It contains the following relations with other models:

- Many to one relation to CurrencyReference

Additional field explanations

Currency - relates to the CurrencyReference to define the type of currency the model represents.

LandUseModel

The LandUseModel model represents the actual land use types in a specific location at a given time.

Field name	Datatype [Database type]	Limitations
Surface	Integer [int]	Accepts values between 0 to 2.147.483.647.
Radius	Integer [int]	Accepts values between 0 to 2.147.483.647.
Land Use	Land Usage [bigint]	Relates to LandUseReference

It contains the following relations with other models:

- Many to one relation to LandUseReference

Additional field explanations

Surface - amount of square meter the land use model represents.

Radius - the radii from which the model is located from the apiary

Land Usage - relates to the Land Usage model to further define the type of land the model represents.

PesticideModel

The pesticide model represents the actual values of the different active ingredients' residues found in specific relevant matrices at a given time.

Field name	Datatype [Database type]	Limitations
Value	String [decimal], null	Accepts positive with up to 6 significant digits and 2 digits following the decimal point.
LOQ Wax	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
LOQ Bee Bread	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
LOQ Honey	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
LOQ Pollen	String [decimal]	Accepts positive values with up to 6 significant digits and 2 digits following decimal point.
Pesticide	PesticidesReference [bigint]	Relates to PesticidesReference

It contains following relations with other models:

- Many to one relation to PesticideReference

Additional field explanations

LOQ Wax - detection threshold of analyses in mg/kg

LOQ Bee Bread - detection threshold of analyses in mg/kg

LOQ Honey - detection threshold of analyses in mg/kg

LOQ Pollen - detection threshold of analyses in mg/kg

Pesticides - relates to the Pesticide model to further define the type of land the model represents.

PollenOriginModel

The pollen origin model represents the actual amount of pollen grains of different plant species identified in specific beekeeping samples.

Field name	Datatype [Database type]	Limitations
Fraction	String [decimal]	Accepts positive with up to 6 significant digits and 2 digits following decimal point.
Amount	Integer [int]	Accepts values between 0 to 2.147.483.647.
PollenOrigin	PollenOriginReference [bigint]	Relates to PollenOriginReference

It contains following relations with other models:

- Many to one relation to PollenOriginReference

Additional field explanations

Fraction - a percentage fraction of total pollen this model represents in the whole sample.

Amount - the amount of this pollen contained in the sample.

PollenOrigin - relates to the PollenOriginReference model to further define the type of land the model represents.

Appendix C – Detailed description of dataset import process

The present section uses the EFSA MUST-B project data from Portugal, provided by one of the researchers participating in the project, Nuno Xavier Jesus Capela, as an example to present the current data import process. Before the data can be actually imported, several steps need to be completed:

C.1. Searching and evaluating eligibility of data

During the early stage of the project development, potential data sources are identified. For this purpose, the development team has selected data sources that are compliant with EFSA's requirements as specified at the start of the project (i.e. a combination of new and existing data sources developed within the proof of concept phase). Importing many different data sources may become counter-productive as they would not properly show the potential of the platform (developed as a prototype, i.e. based on a subset of data sources and analytics). Therefore, as hive scale and weather station data were already available in the platform, it was decided to further integrate similar datasets from MUST-B with additional kinds of data, for continuing the development of the platform. As a result, a contact with a data provider was established and a first data sample was shared, thanks to which the development team established if the data quality requirements were fulfilled.

C.2. The metadata

The next step was to gather all related metadata, for that we used a Google Form survey (later it was moved to a more open platform hosted by Limesurvey²⁶, and currently the web application includes its own form) to gather required metadata related to the MUST-B project, the data owner and the data provider.

C.3. Transforming scientific raw data to a format suitable for import

Before the import of data can actually be made, cleaned and prepared files are required. This process took many weeks to be completed. Currently, the platform does not contain an automated data import process, which should be planned in the future, once a data standard is adopted. Because of that, a narrow collaboration was established between the provider and the development team to structure the data with the least amount of effort and with the simplest structure possible for the developer to manage it. The actions included here are:

- Encoding of date and time fields;
- Encoding of the decimal delimiter;
- Removal of invalid data (e.g. Relative humidity values of 225%);
- Inclusion of additional metadata (the devices used to collect the data);
- Combining the data from several hives within a single apiary;
- Proper documentation and inclusion of the colony management data;
- Clear description of the data types included in the dataset.

The dataset includes data from 2 separate locations (Denmark and Portugal) and years (2019 and 2020). For this reason, it was decided to import the data in four different files, one for each year and each location. Tables 6–9 represent examples of data from within a single file containing data from apiary at Lousa in one of the 2 registered years (i.e. 2019).

²⁶ <https://www.limesurvey.com>

Table 6: Table showing an example of the layout of the dataset related to colony development

Date	Colony num-ber	Capped cells	Uncapped cells	Larva cells	Egg cells	Total brood cells	Egg rate	Pollen cells	Weight Honey/nec-tar (kg)	Harvested honey (yield)	Number of bees	Hive materi-als (kg)	Queen pres-ence (1 - presence, - p absence)
2019-03-10	1	6972	5169	3229	1940	12141	578	5277	2.65		15806	24.7	1
2019-03-28	1	11825	6143	3889	2254	17968	856	2844	6.62		19032	24.7	1
2019-04-16	1	13731	9022	5586	3436	22753	1083	3866	0.35		36774	34.77	1
2019-05-04	1	13475	8612	5113	3499	22087	1052	6068	5.24		46452	36.15	1
2019-05-22	1	15011	6959	3974	2985	21970	1046	7491	11.75		34435	36.21	1
2019-06-09	1	11030	9172	4895	4277	20202	962	7491	14.88		39516	36.32	1
2019-06-29	1	13441	7558	4659	2899	20999	1000	8172	15.79		33145	36.35	1
2019-07-18	1	9772	5731	3469	2262	15503	738	10662	20.65		30161	36.36	1
2019-08-06	1	8512	5964	3930	2034	14476	689	8638	21.16		24839	36.4	1
2019-08-25	1	6770	3911	2553	1358	10681	509	8328	20.04		16774	36.33	1
2019-08-30	1									11.77			
2019-09-13	1	3004	2841	1294	1547	5845	278	8488	5.97		18226	24.7	1

Table 7: Table showing an example of the layout of the dataset related to the hive scales and colony weight

Interior temperature (°C)	Scale weight (kg)	Relative humidity (%)	Date	Time
27.6	34.2	48	2019-03-18	11:32
24.2	34.5	39	2019-03-18	13:39
22.8	34.6	33	2019-03-18	15:44
18.9	34.7	25	2019-03-18	17:52
15.3	34.6	40	2019-03-18	19:58
14.8	34.7	51	2019-03-18	22:00
15.7	34.6	42	2019-03-19	00:11
15	34.5	33	2019-03-19	02:17
14.6	34.5	50	2019-03-19	04:24
14.2	34.5	50	2019-03-19	06:31
21	34.3	50	2019-03-19	08:39
25.3	33.9	53	2019-03-19	10:45

Table 8: Table showing an example of the layout of the dataset related to the hive management and beekeeping practices

Date	Time (if available)	Type	Quantity	Unit	Description
2019-03-28		Input	10	kg	10 frames Upper nest with wax
2019-03-29		Output			Remove scale for upgrade
2019-04-05		Input			Install scale
2019-04-16		Output	2	stripes	Remove spring treatment (Apivar)
2019-08-30		Output	23.4	Kg	Remove upper nest for honey collection
2019-09-13		Input	3.8	Kg	Empty honey super for Apiguard treatment
2019-09-13	15:00	Output	2.1	Kg	Remove video box
2019-10-11	08:00	Output	3.8	Kg	Remove empty honey super
2019-10-11	08:00	Input	1	kg	Sugar paste
2019-11-12	15:00	Input	2.5	kg	Sugar paste
2020-01-21					Dead colony (food bag was ripped on top and created air flow)

Table 9: Table showing an example of the layout of the dataset related to weather conditions

Solar radiation (wat/m ²)	Relative humidity (%)	Temperature (*C)	Rainfall (mm)	Wind direction (Deg)	Wind gust (km/h)	Wind speed (km/h)	Dew point (*C)	Date	Time
0	72.4	9.1	0	125	91	17	4.4	2019-03-01	01:00
0	59.7	9.3	0	108	16	11	1.9	2019-03-01	01:15
0	59.6	9.2	0	111	19	11	1.8	2019-03-01	01:30
0	59.4	8.9	0	38	12	6	1.5	2019-03-01	01:45
0	52	9.6	0	49	9	4	0.2	2019-03-01	02:00
0	51.3	10.4	0	40	17	9	0.8	2019-03-01	02:15
0	52.7	10.4	0	33	19	9	1.2	2019-03-01	02:30
0	52.5	10.4	0	22	14	8	1.1	2019-03-01	02:45
0	52.5	10.4	0	26	12	8	1.1	2019-03-01	03:00
0	53.1	10.3	0	31	12	6	1.2	2019-03-01	03:15
0	54.09	10.1	0	39	14	4	1.5	2019-03-01	03:30

C.4. The import

C.4.1. The files

Once the file format is finalized, the next step is the writing of the import script, based on these parameters:

- A set of four files, each representing data from the individual year and location
- The file format is Excel Workbook (.xlsx)
- The data representing the different measurements (as shown in Tables 6–9) is displayed by sheets for each file
- The first row of each sheet always represents headers (i.e. the different measurements)
- Each row contains data from a single point in time (date and/or time).

A script file is created, usually named by project and sometimes by year, and the data is represented. In this case, EFSAMUSTBSeeder.php has been created under the database/seeder/Connectors folder. Raw data is moved into the database/seeder/Connectors/raw/EFSa folder.

C.4.2. The metadata

Before importing the actual data, all metadata must be imported into the corresponding data models, as the metadata needs to be related during the import process.

- 1) Create a new project;
- 2) Create new providers (EFSa as owner and Nuno Capela as provider);
- 3) Create or load data types used in this project (hive id, apiary id, location, hive weight, in-hive relative humidity, in-hive temperature, outside temperature, rain, solar radiation, wind direction, wind gust, dew point, monitoring events, scale weight changes, hive sampling, other hive management, capped cells, uncapped cells, larva cells, egg cells, brood cells, egg laying rate, beebread cells, amount of produce (honey and nectar), extracted honey, hive population, weight of hive materials, queen presence);
- 4) Create new device categories (weather station and in-hive weight sensor);

- 5) Create new device vendors (Apis technology for hive weight and spectrum technologies for weather station);
- 6) Create new devices (B-Hive standalone hive weight and Watchdog 2900ET weather station);
- 7) Register 2 new locations (EFSA Castelo Branco and EFSA Lousa) based on GPS coordinates and Post number.

C.4.3. The data

With the help of a third-party library Excel files are read and data is imported, file by file and sheet by sheet. For each kind of data, a separate handler function needs to be written to import the data. Figures 24–28 show diagrams displaying how the file columns are mapped to the data model and the database.

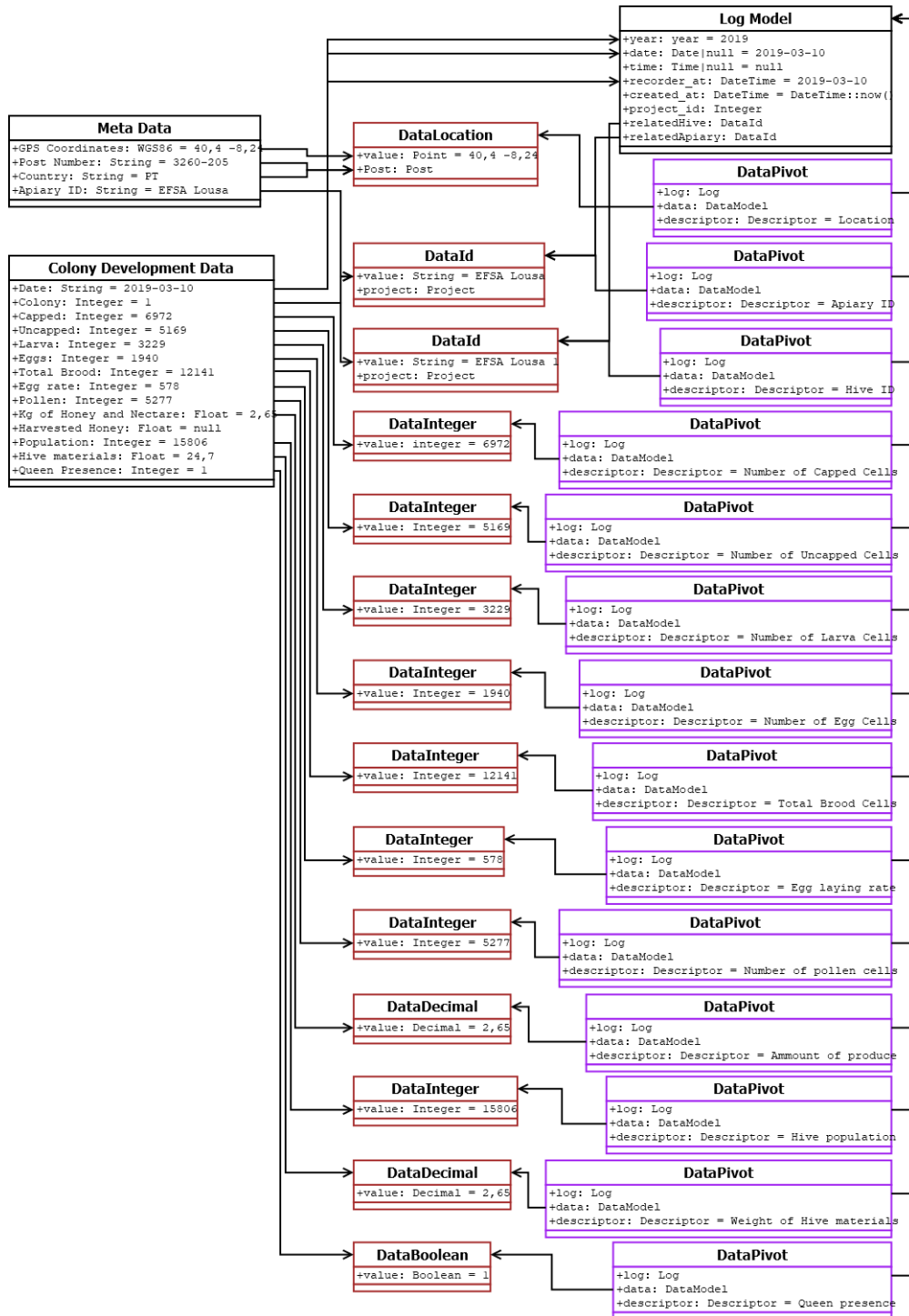


Figure 24: Mapping of a row related to colony development to the data model

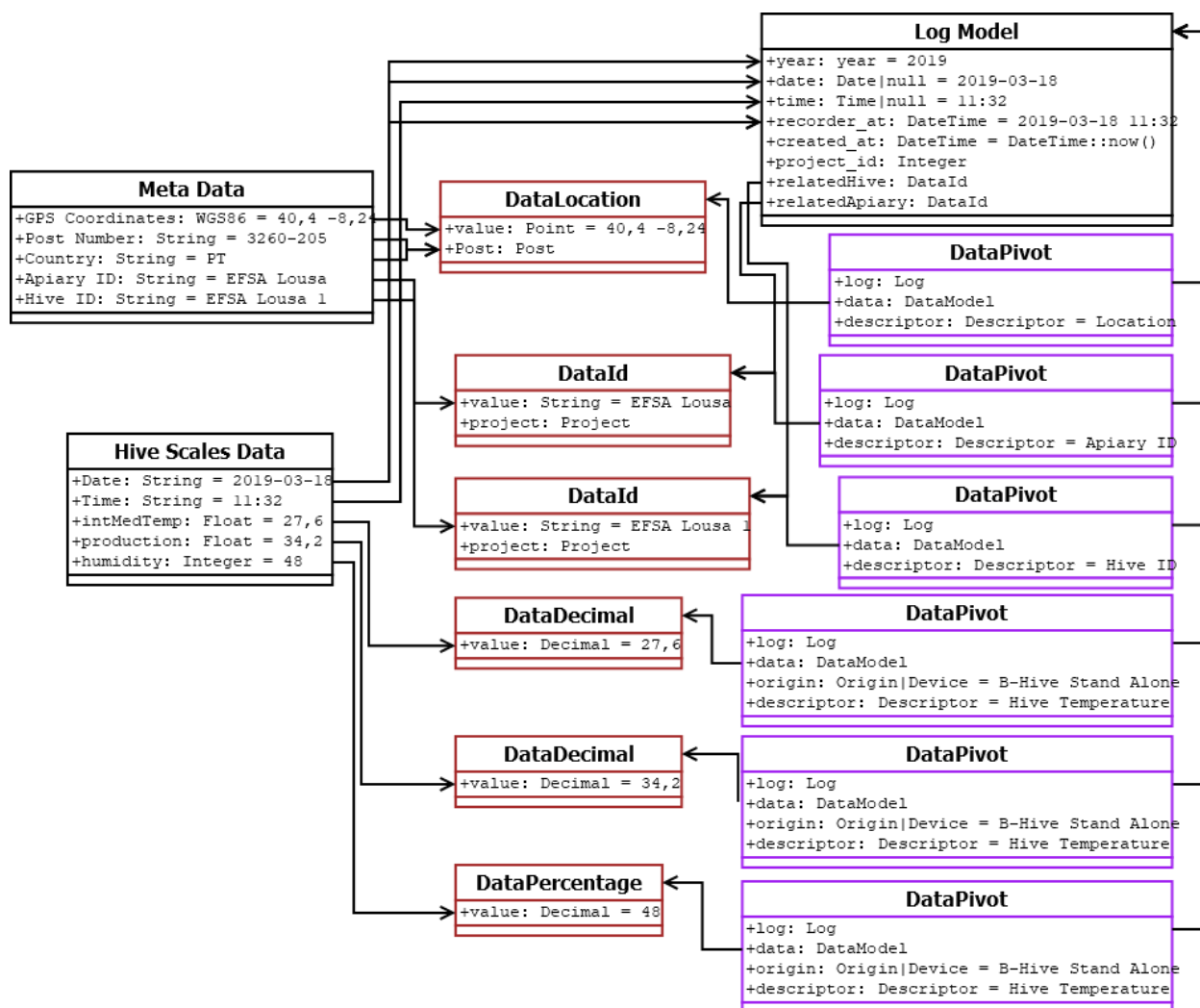


Figure 25: Mapping of a row related to hive scales and colony weight to the data model

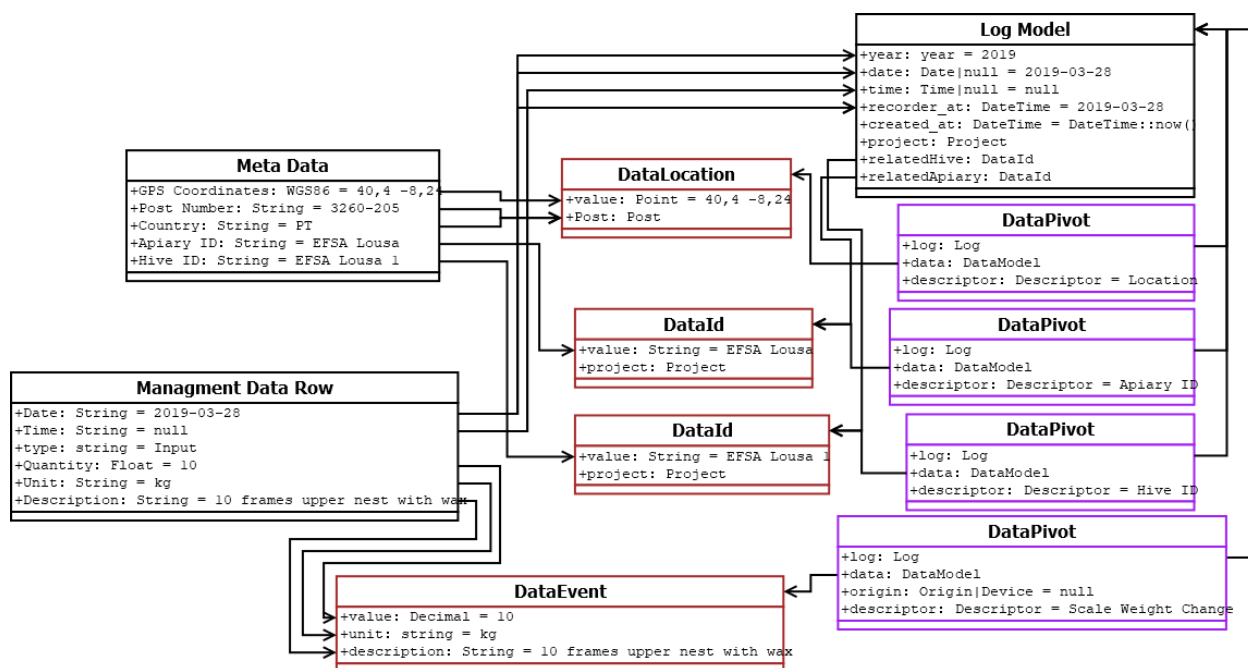


Figure 26: Mapping of a row related to hive management and beekeeping practices to the data model

Figure 27 shows the procedure to determine the type of the event.

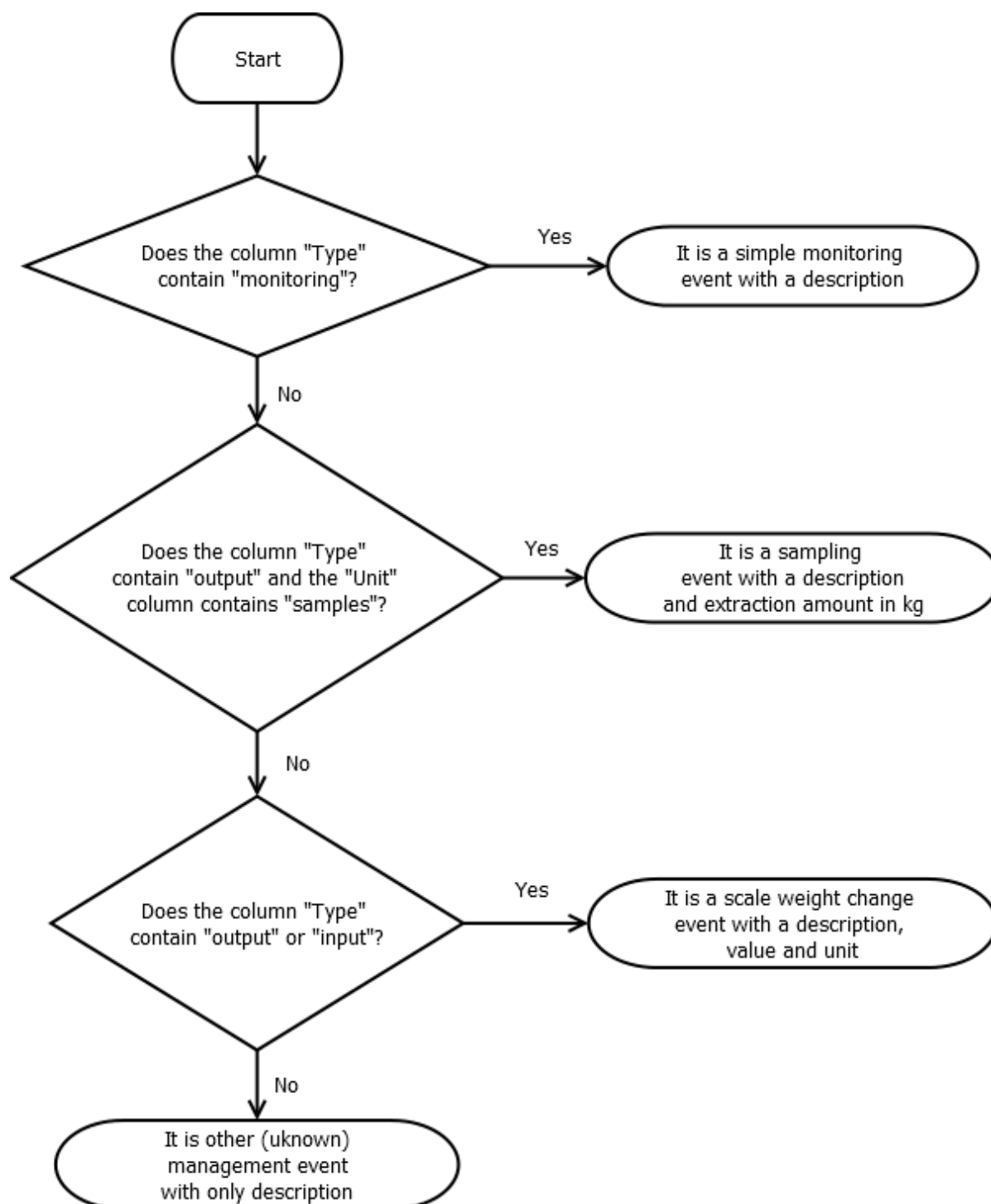


Figure 27: Process to determine the type of the event related to the hive management and beekeeping practices

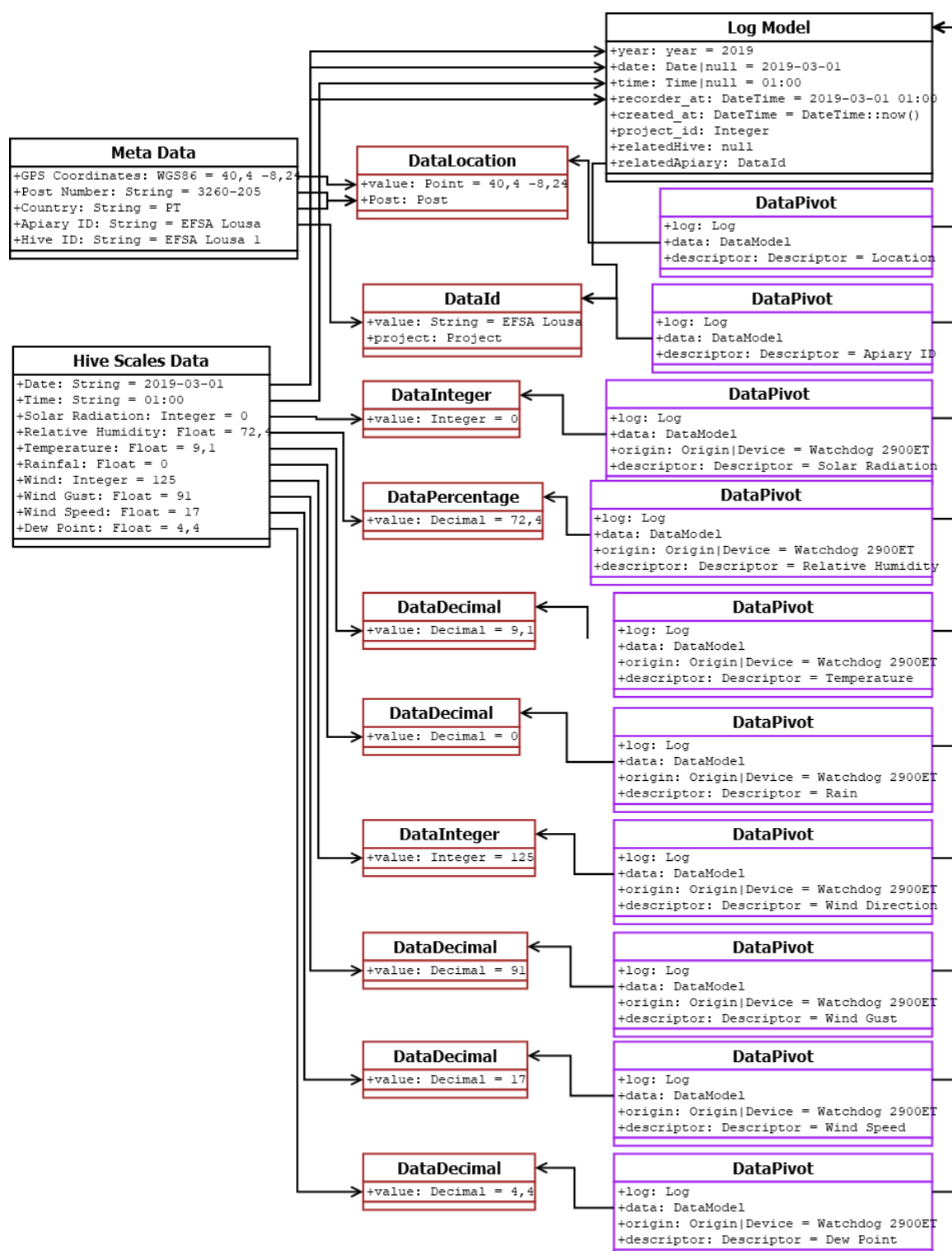


Figure 28: Mapping of a row related to weather conditions to the daModel

C.5. Post processing

Several algorithms have been developed within the prototype platform. Some of them can be used with this dataset as well. MUST-B data is considered as historical data (the import process is run only once and no additional data is being added over time), algorithms can be run only once to achieve the best results. The following algorithms are being run on the data:

- Aggregating daily values of sensors (in-hive weight, in-hive temperature, in-hive humidity, outside temperature, outside humidity, rain, solar radiation, wind speed, wind gust);
- Daily colony production;
- Colony production period;
- Colony winter consumption;
- Metabolic resting state;
- Available flight time;
- Honey yields.

C.6. Conclusions

As seen above, the import process is not trivial. Currently each dataset requires its own import function, created by a developer. That is why the cooperation and contribution to the BeeXML data standardization effort is key with the objective, in the long-term, to simplify the integration process.

Appendix D – Pseudocodes of the different data algorithms included in the EUBP Prototype Platform

This appendix describes in detail some of the developed algorithms within the platform. For most, the pseudocode form is used to describe its functions.

D.1. Pseudocode for the estimation of daily colony production

This pseudocode represents the implementation of section “4.6.1. Daily colony production” of the report.

Algorithm calculateColonyProduction

Input: List of recorded weights at midnight named *weights* keyed by Date,
 Datetime named *startday*,
 List of colony management events weight changes named *events*, keyed by DateTime.

Output: List of daily production named *production*, keyed by Date.

currentDay := *startDay*

While is available weight at midnight of *startday+1* within *weights* do

dailyProduction := *weights*_{*currentDay+1*} - *weights*_{*currentDay*}

If colony management event within *events* happened during *currentDay* then

dailyProduction := *dailyProduction* - sum(*events*_{*currentDay*})

End if

If *dailyProduction* > 0 then

*production*_{*currentDay*} := *dailyProduction*

End if

currentDay := *currentDay* + 1

Return *production*

End function

D.2. Pseudocode for the detection of colony production period

This pseudocode represents the implementation of section “4.6.2. Colony production period detection” of the report.

Algorithm detectedProductionPeriod

Input: List of daily production named *production* keyed by Date,
 Datetime named *startday*,
 Threshold value to detect the trend named *threshold*.

Output: List of detected trends named *trend*, keyed by Date.

currentDay := *startday*

```

While is available production at currentDay-7 within production do
    productionSum := production within last seven days
If productionSum > threshold then
    trend(currentDay) := true
Else
    trend(currentDay) := false
End if
currentDay := currentDay + 1
Return trend

```

End function

D.3. Pseudocode for the estimation of the colony consumption during the winter

This pseudocode represents the implementation of section "4.6.3. Colony consumption during winter" of the report.

Algorithm calculateWinteringConsumption

Input: List of daily production named *production* keyed by Date,
List of colony management events weight changes named *events*, keyed by DateTime,
List of daily rain amounts named *rain* keyed by Date.

Output: List of calculated colony consumptions named *consumptions*, keyed by Year.

```

currentYear := startYear from production
While production is available for startYear+1 within production do
    consumptions(currentYear) := production between extractionDate and first positive produc-
tion value in currentYear+1, while ignoring small positive changes during rainy days
    currentYear := currentYear + 1
End While
Return consumptions

```

End function

D.4. Pseudocode for the estimation of the metabolic resting state of the colony

This pseudocode represents the implementation of section "4.6.4. Metabolic resting state of the colony" of the report.

Algorithm calculateMetabolicRestingStateChange

Input: List of hourly recorded weights named *weights* keyed by Datetime,
Datetime named *startDateTime*.

Output: List of daily night weight changes named *metabolicConsumption*, keyed by Date.

currentDayTime := *startDateTime*

While is available weight at next hour of *currentDayTime+1* within *weights* **do**

If *currentDayTime* is sunset **then**

currentDay := get day from *currentDayTime*

startValue := *weights*_(currentDateime)

End if

If *currentDayTime* is sunrise **then**

*metabolicConsumption*_(currentDay) := *weights*_(currentDateime)

End if

End While

Return *metabolicConsumption*

End function

D.5. Pseudocode for the calculation of sensor statistics on apiary, district, region and country level

This pseudocode represents the implementation of section "4.6.6. Calculate sensor statistics on apiary, district, region and country level" of the report.

This algorithm is run each time for separate levels (apiary, district, NUTS region 3 - 1, country)

Algorithm calculateSensorStatisticsOfGroup

Input: List of daily values and calculations from available data within a group named logs, grouped and keyed by Datetime

 Date named *startDate*

Output: List of calculated statistics named outputLogs, keyed by Datetime.

currentDay := *startDate*

While are data available within logs **do**

For each availableData in logs **do**

 outputLogs_(currentDay)(availableData) := runMethodology(availableData)

End For

currentDay := *currentDay* + 1

End while

Return *outputLogs*

End function

D.6. Pseudocode for the calculation of how many hours of flight time are available to the bees

This pseudocode represents the implementation of section “4.6.7. Calculate how many hours of flight time are available to the bees” of the report.

Algorithm calculateFlyTime

Input: List of hourly recorded solar radiation and temperature sensor data named *sensorData*, grouped by Date, keyed by Time,

List of sunrise/sunset times for current location named *sunInfo*, keyed by Date.

Datetime named *startDateTime*.

Output: List of available fly time per day named *flyTimes*, keyed by Date.

currentDate := *startDate*

While are data available within *sensorData* **do**

flyTimes(*currentDate*) := getFlyTime(*sensorData*, *sunInfo*(*currentDate*))

currentDay := *currentDay* + 1

End while

Return *outputLogs*

End function

Function getFlyTime

Input: the list of sensor data for single day, named *sensorData*, keyed by Time,

A 2 value list with sunrise and sunset time, named *sunInfo*.

Output: amount of fly time for the day the data is provided, named *totalFlyTime*.

totalFlyTime := 0

currentTime := first key of *sensorData*

While is available *sensorData*(*currentTime* + 1) **do**

If *timeLog* is within thresholds then

totalFlyTime := *totalFlyTime* + ((*currentTime* + 1) - *currentTime*)

End If

End while

Return *totalFlyTime*

End function

D.7. Pseudocode for calculating average colony production based on Honey production and Beehive statistics

This pseudocode represents the implementation of section “4.6.7. Calculating average colony production based on honey production and beehive statistics” of the report.

Algorithm calculateAverageColonyProductionFromFAOSTAT

Input: List of total honey production named *totalProduction*, keyed by Country,

list of number of bee hives named *hives*, keyed by Country,

List of countries named *Countries*.

Output: List of honey production per hive named *production*, keyed by Country.

For each country **in** countries **do**

$production_{(country)} := totalProduction_{(country)} \div hives_{(country)}$

End for

Return production

End function

D.8. Pseudocode for the estimation of how much honey a hive contains

This pseudocode represents the implementation of section "4.6.8. How much honey does a hive contain?" of the report.

Algorithm calculateProduceAmount

Input: List of recorded weights at midnight named *weights* keyed by Date,
Date named *startDate*.

Output: List of daily amounts of produce within a hive *produce*, keyed by Date.

$currentDate := startDate$

$minimumWeight := 1000$

While are data available within *weights* **do**

If $minimumWeight > weights_{(currentDay)}$

$minimumWeight := weights_{(currentDay)}$

$currentDay := currentDay + 1$

$currentDate := startDate$

End while

$currentDate := startDate$

While are data available within *weights* **do**

$produce_{(currentDate)} := weights_{(currentDay)} - minimumWeight - weighOfBessOnDay(currentDay)$

$currentDay := currentDay + 1$

End while

Return produce

End function

D.9. Pseudocode for the detection of the colony management events

This pseudocode represents the implementation of section "4.6.9. Calculating average colony production based on hHoney production and bBeehive statistics" of the report.

Algorithm analyzeForManagmentEvents

Input: List of hourly recorded weights named *weights*, keyed by Datetime, Datetime named *startDateTime*.

Output: List of detected events named *events*, keyed by Datetime.

currentDateTime := *startDateTime*

While is available next weight of *currentDateTime+1* within *weights* **do**

weightChange := *weights*_(currentDateTime) - *weights*_(currentDateTime+1)

If not function isFalsePositive(*weightChange*, *currentDateTime*) **then**

If *weightChange* and *currentDateTime* fits within conditions of possible matches **then**

*events*_(currentDateTime) := detected *management event*

End if

End if

currentDay := *currentDay* + 1

return events

End function

Function isFalsePositive

Input: the weight change value, named *weightChange*,
Current date and time of the weight change named *currentDateTime*.

Output: True or false, either is false positive or not.

If values fit to conditions of the false positives table **then**

Return true

Else

Return false

End if

End function

D.10. Pseudocode for the estimation of the economic value of pollination

This pseudocode represents the implementation of the platform report "Socioeconomic data on pollinators", available on <https://bee-ppp.eu/reports/socioeconomic-data-on-pollinators>.

Algorithm getEconomicValueOfPollinationOfCropPerMemberStatePerYear

Input: List of crop pricing in USD named *pricings*, keyed by year, country and crop,
List of crop production amounts named *production*, keyed by year, country and crop,
List of crop species with minimum and maximum pollination dependency named *species*, keyed by crop.

Output: List of total economic values in USD, named total *Value*, keyed by Year, Country and Crop,

List of minimum economic values of pollination in USD, named *minimumValue*, keyed by year, country and crop,

List of maximum economic values of pollination in USD, named *maximumValue*, keyed by year, country and crop,

List of average economic values of pollination in USD, named *averageValue*, keyed by year, country and crop,

List of relative economic values of pollination in percentages, named *relativeValue*, keyed by year, country and crop.

Foreach *year* **do**

Foreach *country* **do**

Foreach *crop* **do**

$totalValue[year][country][crop] = pricings[year][country][crop] * productions[year][country][crop]$

$minimumValue[year][country][crop] = pricings[year][country][crop] * productions[year][country][crop] * species[crop][min]$

$maximumValue[year][country][crop] = pricings[year][country][crop] * productions[year][country][crop] * species[crop][max]$

$averageValue[year][country][crop] = (minimumValue[year][country][crop] + maximumValue[year][country][crop]) / 2$

$relativeValue[year][country][crop] = (averageValue[year][country][crop] * 100) / totalValue[year][country][crop]$

End foreach

End foreach

End foreach

return *totalValue, minimumValue, maximumValue, averageValue, relativeValue*

End function

Algorithm *getEconomicValueOfPollinationOfCropPerYear*

Input: List of total economic values in USD, named *totalCountryValue*, keyed by Year, Country and Crop,

List of minimum economic values of pollination in USD, named *minimumCountryValue*, keyed by year, country and crop,

List of maximum economic values of pollination in USD, named *maximumCountryValue*, keyed by year, country and crop,

List of average economic values of pollination in USD, named *averageCountryValue*, keyed by year, country and crop.

Output: List of total economic values in USD, named *totalValue*, keyed by Year and Crop,
List of minimum economic values of pollination in USD, named *minimumValue*, keyed by year and crop,

List of maximum economic values of pollination in USD, named *maximumValue*, keyed by year and crop,

List of average economic values of pollination in USD, named *averageValue*, keyed by year and crop,

List of relative economic values of pollination in percentages, named *relativeValue*, keyed by year and crop.

Foreach *year* **do**

Foreach *crop* **do**

$totalValue[year][crop] = 0$

$minimumValue[year][crop] = 0$

$maximumValue[year][crop] = 0$

Foreach *country* **do**

$totalValue[year][crop] = totalValue[year][crop] + totalCountryValue[year][country][crop]$

$minimumValue[year][crop] = minimumValue[year][crop] + minimumCountryValue[year][country][crop]$

$maximumValue[year][crop] = maximumValue[year][crop] + maximumCountryValue[year][country][crop]$

End foreach

$averageValue[year][crop] = (minimumValue[year][crop] + maximumValue[year][crop]) / 2$

$relativeValue[year][crop] = (averageValue[year][crop] * 100) / totalValue[year][crop]$

End foreach

End foreach

return *totalValue, minimumValue, maximumValue, averageValue, relativeValue*

End function

Algorithm *getEconomicValueOfPollinationOfCropPerMemberstatePerYear*

Input: List of total economic values in USD, named *totalCountryValue*, keyed by Year, Country and Crop,

List of minimum economic values of pollination in USD, named *minimumCountryValue*, keyed by year, country and crop,

List of maximum economic values of pollination in USD, named *maximumCountryValue*, keyed by year, country and crop,

List of average economic values of pollination in USD, named *averageCountryValue*, keyed by year, country and crop.

Output: List of total economic values in USD, named *totalValue*, keyed by year and Crop,
List of minimum economic values of pollination in USD, named *minimumValue*, keyed by country and year,

List of maximum economic values of pollination in USD, named *maximumValue*, keyed by country and year,

List of average economic values of pollination in USD, named *averageValue*, keyed by country and year,

List of relative economic values of pollination in percentages, named *relativeValue*, keyed by country and year.

Foreach *country* **do**

Foreach *year* **do**

$\text{totalValue}[\text{country}][\text{year}] = 0$

$\text{minimumValue}[\text{country}][\text{year}] = 0$

$\text{maximumValue}[\text{country}][\text{year}] = 0$

Foreach *crop* **do**

$\text{totalValue}[\text{country}][\text{year}] = \text{totalValue}[\text{country}][\text{year}] + \text{totalCountryValue}[\text{country}][\text{country}][\text{year}]$

$\text{minimumValue}[\text{country}][\text{year}] = \text{minimumValue}[\text{country}][\text{year}] + \text{minimumCountryValue}[\text{year}][\text{country}][\text{crop}]$

$\text{maximumValue}[\text{country}][\text{year}] = \text{maximumValue}[\text{country}][\text{year}] + \text{maximumCountryValue}[\text{year}][\text{country}][\text{crop}]$

End foreach

$\text{averageValue}[\text{country}][\text{year}] = (\text{minimumValue}[\text{country}][\text{year}] + \text{maximumValue}[\text{country}][\text{year}]) / 2$

$\text{relativeValue}[\text{country}][\text{year}] = (\text{averageValue}[\text{country}][\text{year}] * 100) / \text{totalValue}[\text{country}][\text{year}]$

End foreach

End foreach

return *totalValue*, *minimumValue*, *maximumValue*, *averageValue*, *relativeValue*

End function

Algorithm *getEconomicValueOfPollinationPerYear*

Input: List of total economic values in USD, named *totalCountryValue*, keyed by year, country and crop,

List of minimum economic values of pollination in USD, named *minimumCountryValue*, keyed by year, country and crop,

List of maximum economic values of pollination in USD, named *maximumCountryValue*, keyed by year, country and crop,

List of average economic values of pollination in USD, named *averageCountryValue*, keyed by year, country and crop.

Output: List of total economic values in USD, named *totalValue*, keyed by year,

List of minimum economic values of pollination in USD, named *minimumValue*, keyed by year,

List of maximum economic values of pollination in USD, named *maximumValue*, keyed by year,

List of average economic values of pollination in USD, named *averageValue*, keyed by year,

List of relative average economic values of pollination in percentages, named *relativeAverageValue*, keyed by year,

List of relative minimum economic values of pollination in percentages, named *relativeMinimumValue*, keyed by year,

List of relative maximum economic values of pollination in percentages, named *relativeMaximumValue*, keyed by year,

Foreach year do

total *Value*[*year*] = 0

minimumValue[*year*] = 0

maximumValue[*year*] = 0

Foreach crop do

Foreach country do

total *Value*[*year*] = total *Value*[*year*] + totalCountry *Value*[*year*][*country*][*crop*]

minimumValue[*year*] = *minimumValue*[*year*] + *minimumCountryValue*[*year*][*country*][*crop*]

maximumValue[*year*] = *maximumValue*[*year*] + *maximumCountryValue*[*year*][*country*][*crop*]

End foreach

End foreach

averageValue[*year*] = (*minimumValue*[*year*] + *maximumValue*[*year*]) / 2

relativeAverageValue[*year*] = (*averageValue*[*year*] * 100) / total *Value*[*year*]

relativeMinimumValue[*year*] = (*minimumValue*[*year*] * 100) / total *Value*[*year*]

relativeMaximumValue[*year*] = (*maximumValue*[*year*] * 100) / total *Value*[*year*]

End foreach

return *totalValue*, *minimumValue*, *maximumValue*, *averageValue*, *relativeAverageValue*, *relativeMinimumValue*, *relativeMaximumValue*

End function

Algorithm getTotalPollinatedArea

Input: List of total area of pollinated crops in square kilometers, named *areas*, keyed by crop, country and year,

List of number of beehives within member states named *beeHives*, keyed by country, and year.

Output: List of total pollinated crops, named *totalArea*, keyed by country and year,

List of number of beehives per total area, named `totalBeehivesPerArea`, keyed by country and year.

Foreach *country* **do**

Foreach *year* **do**

$totalArea[country][year] = 0$

Foreach *crop* **do**

$totalArea[country][year] = totalArea[country][year] + areas[year][country][crop]$

End foreach

$totalBeehivesPerArea[country][year] = beeHives[country][year] + totalArea[country][year]$

End foreach

End foreach

return *totalArea*, *totalBeehivesPerArea*

End function

D.11. Pseudocode for the description of the beekeeping socio-economic characteristics

This pseudocode is currently under development and is not yet available, because the related report "Beekeeping economic review" is still under review and not available.