OCEAN
Operator-Centred Enhancement of Awareness in Navigation

D4.3 - Framework for regular whale aggregation prediction

31 December 2024

Co-funded by
the European Union

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101076983. UK participants in Project OCEAN are supported by UKRI grant numbers 10038659 (Lloyds Register) and 10052942 (The Nautical Institute).

Project information

Project acronym:	OCEAN
Full title:	Operator-Centred Enhancement of Awareness in Navigation
Grant agreement:	101076983
Programme and call:	Horizon Europe, CL5-2022-D6-01
Project coordinator:	HVL
Contact:	ocean@hvl.no
Project duration:	36
Project website:	www.ocean-navigation-awareness.eu

Deliverable information

Deliverable number	D4.3
Deliverable title:	Framework for regular whale aggregation prediction
Dissemination level:	Public
Deliverable type:	Other
License:	N/A
Status:	Submitted
Due date:	31/12/2024
Submission date:	05/12/2024
Work Package:	04
Lead Beneficiary:	UAC

Disclaimer

Co-funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Climate, Infrastructure and Environment Executive Agency. Neither the European Union nor the granting authority can be held responsible for them. UK participants are supported by the UK Research and Innovation. The OCEAN consortium is committed to publish accurate and up to date information and take the greatest care to do so. However, the OCEAN consortium, or any of its members, cannot accept liability for any inaccuracies or omissions, nor do they accept liability for any direct, indirect, special, consequential or other losses or damages of any kind arising out of the use of this information.

Copyright and license information

Revision History

Date	Version	Author	Comment
01/06/2024	V0.01	Rui Prieto, UAC	Creation of document outline
01/11/2024	V0.02	Rui Prieto, Sergi Pérez-Jorge, Mónica Almeida e Silva, UAC	First draft of report
02/11/2024	V0.03	Rui Prieto	Expanded 'Validation and testing' section with more information and images. Corrected some typos.
04/11/2024	V0.03	Simon Berrow, IWDG	Reviewed and commented
04/11/2024	V0.04	Rui Prieto, UAC	Addressed comments
05/11/2024	V0.05	Rui Prieto, UAC	Small changes to formatting, including acknowledgements in the front page
25/11/2024	V0.06	Erik Styhr Petersen, HVL	Coordinator's review; very minor formatting support.
29/11/2024	V0.07	Rui Prieto, UAC	Improved text according Coordinator's review.
03/12/2024	V1.0	Risk and Quality Manager, HVL	Final check and approval for submission

Author List

Institution	First name and Name	Contact information
UAC	Rui Prieto	rcabprieto@gmail.com
UAC	Sergi Pérez-Jorge	
UAC	Mónica Almeida e Silva	

Abbreviations and acronyms

Abbreviation or acronym used in this document	Explanation
4D-SAD	OCEAN 4D Situation Awareness Display
API	Application Programming Interface
app	Software application (usually for handheld equipment)
AVHRR	Advanced Very High-Resolution Radiometer
CMEMS	Copernicus Marine Environment Monitoring Service
DOI	Digital Object Identifier
DOM	Dynamic Ocean Management

DRA	Dynamic Routing Area(s)
ENHI	European Navigational Hazard Infrastructure
ENM	Environmental Niche Model(s)
GAM	Generalized Additive Model (model family)
NW	Navigational warning(s)
OS	Operating System
SST	Sea Surface Temperature
TLR	Technology Readiness Level
WP	Work Package

Glossary

Term	Definition used or meaning in the Acronym project	Reference or source for the definition if applicable
Biota	In the context of this document, the fauna of a region	https://www.merriam-webster.com
Covariate	A variable (= a quantity that can change) that may affect the result of what is being studied.	https://dictionary.cambridge.org
Taxon	Any unit used in biological classification (taxonomy). Plural: ‘taxa’.	https://www.britannica.com
Vagile	A taxon that can move freely.	https://www.merriam-webster.com

Executive Summary

This document reports the work developed under Task 4.3, between months 18 to 27 of the OCEAN project, and relates to Deliverable D4.3 (predictive tool framework software) that is made public through the GitHub repository in https://github.com/AzWhaleLab/OCEAN.

The description of the work planned for Task 4.3 in the Grant Agreement is reproduced below:

> "A framework to regularly run the ENMs created in Task 4.2 will be created, through a dedicated server that will be connected to remote-sensed and derived data products used as model co-variates (e.g. sea surface temperature, ocean colour, primary production, SEAPODYM lower- and mid-trophic prey). The feasibility of interactively feeding new sightings reported by the voluntary reporting system or satellite imagery on each iteration (Task 6.6) to improve model predictive power will be evaluated. The regularity of model iterations will be decided based on frequency of co-variate updates. After each model iteration, the occurrence of values of high probability whale occurrence, above a given threshold to be decided on basis of whale ecology and expert knowledge, will trigger the issuing of Dynamic Routing Areas (DRAs) for which boats are requested to either reduce speed or avoid altogether. DRAs based on ENMs predictions will be valid until a new model iteration is complete and will be issued through the *European Navigational Hazard infrastructure* described in WP6."

Shortly, the overall goal of this task is the creation of a working software tool that can automatize the production of predictions of areas with high probability of whale occurrence.

The tool is based on an Environmental Niche Model (ENM), developed for the sperm whale (*Physeter macrocephalus*) in Task 4.2. The model fitting procedure is detailed in its own document (Report for Deliverable 4.2[1]). This model serves as the motor to create the regular predictions. Given that the model includes dynamic covariates (predictive environmental variables that change over time), the predictions are not static in time, comprising a good example for application of a predictive tool for dynamic ocean management (DOM; in which management areas adapt to changes in the distribution of the resource).

Thus, by inputting a set of predictive covariates that translate the conditions in the environment at a given time, areas of high probability of whale occurrence can be plotted using model results.

To automatize that process, a workflow was created, consisting of several steps that comprise the main software components to the system:

- *Covariate data gathering and scheduling*: a routine that sets the frequency of predictions (daily, weekly, monthly, etc.), and collects dynamic predictive covariates data from external data provider(s).
- *Model predictions*: a routine to create the predictions based on the ENM and new predictive covariates, that are plotted as a raster file with continuous (0-1) values representing probability of whale occurrence.
- *Polygon extraction and classification*: a routine to convert the continuous values in the raster file created in the prior step into polygons representing areas of high probability of whale occurrence, based on a given threshold value. This step is necessary to guarantee that the information can be transmitted using current protocols for Navigation Warning messages.
- *Posting*: a routine to send the resulting polygons (termed here as Dynamic Routing Areas) to the *European Navigational Hazard Infrastructure* (ENHI) developed in OCEAN's Work Package 6.

The use of assimilative models, in which ENMs are automatically updated using new sightings from, for example, voluntary reporting or satellite imagery was considered at an

[1] https://ocean-navigation-awareness.eu/wp-content/uploads/2024/10/D4.2-Environmental-Niche-Models-v1.01.pdf

earlier phase but was discarded. That decision was based on discussions with specialists during the *Marine Mammal Ship Strike Mitigation Workshop* held under Task 4.1 (D4.1) and the realization that there are methodological challenges in combining different data types to fit the same models that need to be resolved first.

Working in close coordination with Work Package 6, all the code necessary for the components identified in the high-level design phase was developed and tested during the period predicted in the Grant Agreement.

The goals of producing regular (daily in this case) predictions that can be distributed through the ENHI to navigators using the reporting app (OCEAN Task 4.4) or the 4D-SAD (OCEAN WP7), or by Navigational Warnings (NW), were met completely, and the software was fully tested before being transferred to a running workstation for full deployment.

The software is now ready for the demonstration (Task 4.7), supported and coordinated by WP9.

Table of Contents

1 Introduction 8

1.1 Task goals and approach 8

1.2 Intended readership 9

1.3 Structure of the document 9

1.4 Relationship with other deliverables 9

2 High-level design 11

2.1 Description of the near-real time predictive tool framework 11

3 Environment setup 13

3.1 Platform 13

3.2 Programming languages 13

3.3 Additional libraries 13

3.4 APIs 13

4 Implementation 14

4.1 Description of the routines 14

4.1.1 Covariate data gathering and scheduling 14

4.1.2 Model predictions 16

4.1.3 Polygon extraction and classification 17

4.1.4 Posting 18

5 Validation and testing 20

6 Conclusions 23

6.1 Main results 23

6.2 Limitations 23

6.3 Further work 23

7 Deliverable form and availability 24

8 References 25

9 Annex 1: The Consortium 26

10 Annex 2: Project Summary 27

11 Annex 3: Software Code 28

List of Figures

Figure 1: High-level framework of the near-real time predictive tool. 11

Figure 2: Example of the R code used to download environmental data from the Copernicus Marine Data Store using the Copernicus Marine Toolbox. Code available in https://github.com/AzWhaleLab/OCEAN. 14

Figure 3: Example of the R code used to activate the data gathering and scheduling routine. Code available in https://github.com/AzWhaleLab/OCEAN. 15

Figure 4: Example of the code to reprocess covariates for a single reference grid. Code available in https://github.com/AzWhaleLab/OCEAN 16

Figure 5: Example of maps of predicted daily sperm whale occurrence probability for the Azores region. A) probability of occurrence, and B) Standard deviation. 17

Figure 6: Map of grid cells with values higher than the quantile 95 threshold applied to daily predictions of sperm whale occurrence probability for the Azores region. 17

Figure 7: A) Contiguous grid cells merged to obtain the polygons. B) Final polygons selected after small polygons are eliminated. 18

Figure 8: Example of data sent to the ENHI with the information of the selected polygons (partial view of the data table). 19

Figure 9: Example of diagnostic file received from the ENHI after submission of message containing multiple polygons, showing the expected output ["I002", null] meaning that the report was received successfully for each polygon sent. 20

Figure 10: Example of ENHI daily activity log file with record of login and transmissions by the predictive tool...21

Figure 11: Example of database query with messages sent by the predictive tool defining a polygon. Only the first line for the second polygon is shown at the bottom of the image. Quantity is set to '0' on purpose, as this is the standard for polygons.....................................21

Figure 12: Screenshot of the voluntary reporting app (developed under Task 4.4) centred in the Azores archipelago, showing polygons for areas with high sperm whale occurrence probability issued by the predictive tool. 22

List of Tables

Table I: Relationship with other deliverables 9

Table II: Technical description of dynamic covariates used in the sperm whale model, derived from Copernicus Marine Environmental Monitoring Service. 15

1 Introduction

1.1 Task goals and approach

The primary goal of Work Package 4 – *Detection and tracking of Marine Mammals in high-density areas* (WP4), is to use different and complementary predictive and detection methodologies to produce reliable information that can increase mariner's awareness of marine mammal ship strike risk along their journeys.

Among these methodologies, environmental niche models (ENM) are included as a powerful way of informing navigation about areas of the increased probability of occurrence of marine mammals (or other large biota), where direct detection is difficult or unavailable.

Overall, the goal of Task 4.3 is to operationalize a tool for near-real time prediction (now-casting) of areas with enhanced probability of marine mammal occurrence, using the sperm whale model developed in Task 4.2 as a working example. Here, near-real time does not necessarily mean very short (hours) periods, and can vary with the environment dynamics and the strength of response to those changes by the organisms.

Vagile species often respond to periodic or stochastic changes in the environment by changing their distribution. This is especially relevant in marine environments that are more dynamic than terrestrial ones.

Thus, to inform management, specifically the operation of shipping activity, an approach based on dynamic ocean management (DOM; Maxwell et al., 2015) is more efficient than traditional designation of static management areas. Using ENMs at appropriate temporal and spatial scales can help optimizing the balance between ecological and economic objectives (Maxwell et al., 2015).

The framework developed in the OCEAN project applies the DOM principle, by informing mariners about the changes in the habitat quality for a given species using the European Navigational Hazard Infrastructure (ENHI), developed in WP6 to relay that information either through the reporting app (OCEAN Task 4.4) or the 4D-SAD (OCEAN WP7). It also complies with current standards for Navigational Warnings (NW) messages, so that the information can reach all mariners, even in the absence of the tools referred above.

Areas flagged as having high probability of whale occurrence by ENMs can be viewed as dynamic routing areas (DRA), in which ship strike risk reduction measures may vary from enhancing attentiveness, to speed reduction or even outright area avoidance. By receiving the information, mariners can take proper action when necessary but avoid unnecessary route changes or speed reductions, when that is not warranted.

To that effect, an ENM for the sperm whale (*Physeter macrocephalus*) was developed in the scope of Task 4.2 (hereby termed 'sperm whale model'), for the Azores archipelago (Portugal). A detailed description of the modelling process and results is given in the Report for Deliverable 4.2[2]. The sperm whale model is representative of other similar models for different species or areas that may, eventually, be integrated with the ENHI developed in WP6.

The sperm whale model developed in Task 4.2 is driven by a set of static (such as seafloor depth and distance to nearby seamounts) and dynamic (such as sea surface temperature and chlorophyll-a concentration from satellite remote sensed data) covariates. The inclusion of dynamic variables as predictive covariates means that the predicted areas of increased probability of occurrence for the species change over time. As such, the DOM principle is applicable, which calls for a recurring deployment of the model to update the predictions accordingly with changes in the dynamic covariates.

It is noteworthy to mention that during the planning phase of the project, the idea of using data obtained in near-real time from the voluntary sightings reporting app (D4.4) or satellite

[2] https://ocean-navigation-awareness.eu/wp-content/uploads/2024/10/D4.2-Environmental-Niche-Models-v1.01.pdf

imagery to continuously improve predictive models (termed assimilative models), was considered for feasibility evaluation. This idea was discussed with other specialists during and after the *Marine Mammal Ship Strike Mitigation Workshop*[3] held under Task 4.1. The common understanding was that, while this will probably be a common practice in the future, it is still at an early phase of development and must be further investigated. There are challenges in combining data from different sources with different data quality that must be addressed, making that approach still experimental. Thus, the idea was discarded for the present work.

1.2 Intended readership

This report details the objectives, methodology and results of Task 4.2. The methodologies developed and applied during the development of Task 4.2 may be of interest to different stakeholders:

- European Commission;
- OCEAN partners;
- Science practitioners;
- Decision/policymakers;
- Shipping industry;
- Environmental NGOs.

1.3 Structure of the document

This document is structured in several sections to facilitate consultation and readability. The high-level design and main modules of the predictive tool framework developed within Task 4.3 are presented in Section 2. Section 3 describes how the environment to run the predictive tool was set up, detailing what are the requirements to set it up in another environment. This is followed by a detailed description of each of the modules is given in Section 4, and the testing methodology and results in Section 5.

1.4 Relationship with other deliverables

The main relationships with other deliverables are presented in Table I.

This document and D4.3 (see Section 7) are supported by deliverables D4.1 and D4.2 and will support deliverables D4.7 and D6.4. The outputs of the predictive tool presented here will be displayed in the At-Sea Reporting App (D4.4) and 4D-SAD (D7.4). There was a co-development of D4.2, D4.3, D4.4, D6.1, D6.2 and D6.3 since all these have interdependencies.

Table I: Relationship with other deliverables

Id	Title	Available to public
D4.1	Workshop Report (including specialist recommendations)	April 2023
D4.2	Environmental Niche Models for selected whale species	October 2024
D4.4	Sightings-at-sea reporting app	January 2025
D6.1	Overall infrastructure design	October 2023
D6.2	Tracking navigation hazard databases (Design & Implementation)	April 2024

[3]https://ocean-navigation-awareness.eu/wp-content/uploads/2023/07/D04_1_ShipStrikesWorkshop_OCEAN_v2.1-1606KB.pdf

D6.3	Ready to Publish European Navigational Safety Database (Design & Implementation)	September 2024
D6.4	European Navigational Safety Database	October 2024
D7.5	Usability testing of the 4D-SAD	April 2025

2 High-level design

2.1 Description of the near-real time predictive tool framework

Development of the near-real time predictive tool framework started by defining the high-level component architecture (Figure 1).

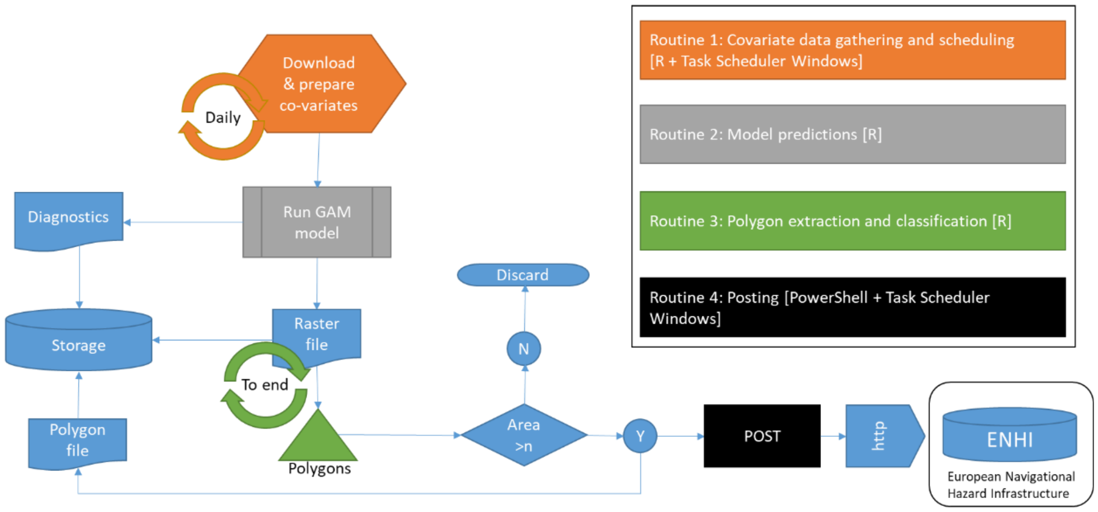

Figure 1: High-level framework of the near-real time predictive tool.

The predictive tool rests on a pre-existing statistical environmental niche model fitted to a set of predictive covariates, in this case comprised by the sperm whale model developed under Task 4.2 (GAM Model).

To produce a new prediction, up to date data from dynamic covariates needs to be obtained from third-party data providers, which can be executed using an Application Programming Interface (API) to communicate with the data provider(s) (detailed in the next section). These data are combined with data from static covariates stored locally, and the model is run as a subsequent step.

In the case of the sperm whale model developed under Task 4.2, results are calculated as predicted probability of occurrence, but the framework is agnostic in relation to the units, even for unitless predictions (such as for habitat quality scores in some modelling approaches).

Model results are plotted as a raster file comprised of gridded integer or floating-point values. These are stored locally and further processed to create polygons that can be transmitted using current marine NW standards and utilize a much lower bandwidth than raster files.

Polygons are created by defining a threshold value to create a simple 0/1 geolocated value table that is then converted to a polygon by uniting each cell central coordinates. To avoid noise from very small areas, single cell polygons are eliminated and a final file defining the polygons is produced and stored locally.

Finally, the resulting polygon file is formatted according to the requirements of use-case 010 (UC-010) defined by the ENHI using the PUSH method by way of the API described in D6.3[4] and transferred to the ENHI over the internet.

[4]https://ocean-navigation-awareness.eu/wp-content/uploads/2024/10/D6.3-OCEAN-Ready-to-publish-European-Navigational-Safety-Databse-Design-Implementation-v1.0.pdf

The steps above are performed through a set of routines (Figure 1) that are performed sequentially:

1. Covariate data gathering and scheduling
2. Model predictions
3. Polygon extraction and classification
4. Posting

The code to run and operationalize the predictive tool is publicly available through the GitHub repository in https://github.com/AzWhaleLab/OCEAN. The code is run using the statistical programming language R (R Development Core Team, 2015).

Each of these routines is described on the following sections.

3 Environment setup

3.1 Platform

Development was performed on a personal computer with Windows 10 Pro 64-bit operating system (OS). After all components were tested, they were migrated to a physical workstation (Dell Precision Tower 7920 CTO) running Windows 11 Pro for workstations OS.

3.2 Programming languages

All main components of the predictive tool framework are coded in *R*, an open source (under GNU S license) language and run-time environment for statistical computation and graphics (R Development Core Team, 2024).

Pre-compiled binary distributions of *R* are available for Linux, macOS and Windows. Code written in R is transferable among these operating systems, with no or only small changes.

For the present implementation, scheduling was achieved using *R* scripts invoking Windows Task Scheduler[5]. When using other OS, different scheduling strategies should be applied.

3.3 Additional libraries

Additionally, to the large number of statistical procedures contained in the core *R* distribution, functionality can be extended by additional libraries ("add-on packages") from different repositories (for more information visit the R project homepage: https://www.r-project.org).

The predictive tool framework presented here uses the next add-on packages:

- *lubridate*: Library containing functions to work with date-times and timespan (Spinu, Grolemund & Wickham, 2023*)*.
- *mgcv*: Library to fit Generalized additive (mixed) models, some of their extensions and other generalized ridge regression with multiple smoothing parameter estimation (Wood, 2017).
- *raster*: Library for reading, writing, manipulating, analysing and modelling spatial data (Hijmans, 2024).
- *sf*: Library supporting standardized encoding of spatial vector data (Pebesma, 2018).
- *taskscheduleR*: Library to schedule scripts/processes with Windows task scheduler (Wijffels & Belmans, 2023).

3.4 APIs

To communicate with external data providers and the ENHI, the framework utilizes APIs.

To download environmental data from the Copernicus Marine Data Store (see Section 4.1.1), it was necessary to install the Copernicus Marine Toolbox[6], which it is a free and easy-to-use, cross-platform, tool that interoperates with the Copernicus Marine Data Store.

The final step of the framework presented here (Figure 1) is partially performed by a PowerShell[7] routine to run the API that interfaces with the ENHI as described in Deliverable 6.3[4]. PowerShell is a cross-platform (Linux, macOS and Windows) task automation tool, built on the .NET Common Language Runtime, and optimized for dealing with structured data (e.g. JSON, CSV, XML, etc.), REST APIs, and object models.

[5]https://learn.microsoft.com/en-us/windows/win32/taskschd/task-scheduler-start-page

[6]https://help.marine.copernicus.eu/en/articles/7970514-copernicus-marine-toolbox-installation

[7]https://learn.microsoft.com/en-us/powershell/scripting/overview?view=powershell-7.4

4 Implementation

4.1 Description of the routines

4.1.1 Covariate data gathering and scheduling

In this step, it is necessary to acquire a regular collection of near real-time data of the dynamic environmental variables required by the sperm whale model developed in Deliverable 4.2. The dynamic covariates driving the sperm whale model are obtained from the Copernicus Marine Environmental Monitoring Service (CMEMS, the Earth observation component of the European Union's Space programme; https://marine.copernicus.eu). All marine data products (i.e., physical, biogeochemical) delivered by CMEMS are publicly available. Products from environmental data without data gaps (Level 4 products provide spatially complete fields) were chosen and integrated into the automated workflow.

For this routine, we developed an automated workflow to acquire the dynamic environmental variables from the Copernicus Marine Data Store at a daily time-step. Other models may require interfacing with different data providers which should be easily integrated in the code.

The two dynamic variables included on the sperm whale model were sea surface temperature and chlorophyll-a concentration, which are used here to illustrate the workflow (Figure 2;

Table II).

```
# ================ Variables for your query ================

###SST sea surface temperature######

# Product and dataset IDs
productId = "cmems_obs-sst_glo_phy_nrt_l4_P1D-m"

# Ocean Variable(s)
#Please keep the space at the beginning (" --variable "variable name"")
variable_sst <- c(" --variable analysed_sst")

# Time range
date_min = ymd(get_date) # start_date
date_max = ymd(get_date) # end_date

# Geographic area and depth level
lon = list(-60, 35)  # lon_min, lon_max
lat = list(-40,70) # lat_min, lat_max
depth = list(0, 1) # depth_min, depth_max

# Output filename
#out_name_chl = paste("cmems_obs-oc_glo_bgc-plankton_nrt_l4-gapfree-multi-4km_P1D_",date_min,".nc", sep="")

out_name_sst = paste("SST", ".nc", sep="")

command_sst <- paste (path_copernicusmarine, " subset -i", productId,
                      "-x", lon[1], "-X", lon[2],
                      "-y", lat[1], "-Y", lat[2],
                      "-t", date_min, "-T", date_max,
                      "-z", depth[1], "-Z", depth[2],
                      variable_sst, "-o", tmpdir_sst,
                      "-f", out_name_sst, "--force-download",
                      sep = " ")

print(command_sst)

system(command_sst, intern = TRUE)
```

Figure 2: Example of the R code used to download environmental data from the Copernicus Marine Data Store using the Copernicus Marine Toolbox. Code available in https://github.com/AzWhaleLab/OCEAN.

Table II: Technical description of dynamic covariates used in the sperm whale model, derived from Copernicus Marine Environmental Monitoring Service.

Variable:	SST (Sea Surface Temperature)	Chl-a (Chlorophyll-a)
Full name:	ODYSSEA Global Sea Surface Temperature Gridded Level 4 Daily Multi-Sensor Observations	Global Ocean Colour (Copernicus-GlobColour), Bio-Geo-Chemical, L4 (monthly and interpolated) from Satellite Observations
Product ID:	SST_GLO_PHY_L4_NRT_010_043	OCEANCOLOUR_GLO_BGC_L4_MY_009_104
Dataset ID:	cmems_obs-sst_glo_phy_nrt_l4_P1D-m	cmems_obs-oc_glo_bgc-plankton_my_l4-gapfree-multi-4km_P1D
Variable code:	analysed_sst	CHL
Source:	Satellite observations	Satellite observations
Spatial extent:	Global Ocean Lat -79.95° to 79.95°Lon -179.95° to 179.95°	Global OceanLat -90° to 90°Lon -180° to 180°
Spatial resolution:	0.1° × 0.1°	4 × 4 km (~0.04 degrees)
Temporal extent:	31 Dec 2020 - ongoing	1997-ongoing
Temporal resolution:	Daily	Daily; Monthly
Link:	https://data.marine.copernicus.eu/product/SST_GLO_PHY_L4_NRT_010_043/description	https://data.marine.copernicus.eu/product/OCEANCOLOUR_GLO_BGC_L4_MY_009_104/description

The coverage of these data products is global. As such, the files are heavy and to improve efficiency, the data is downloaded only for the area of interest, defined by a bounding box and maximum depth (Figure 2). Temporal resolution in this case is set to a daily time-step, but that can be changed according to the product definitions and user needs.

The R library *taskscheduleR* is used to automate the running of this and all other processes of the predictive tool framework, by invoking Windows Task Scheduler (Figure 3). As mentioned earlier, if working with a different OS, the automation must be performed using other appropriate approach.

```
##Automate R script##


library(taskscheduleR)

#taskscheduleR:::taskschedulerAddin()

fichero <- "C:/R/Projects/OCEAN/Tool_R_codes/DOM/New_codes_DOM/Download_data_from_CMEMS_DOM_tool.R"

taskscheduler_create(taskname = "download_CMEMS_data",
                     rscript = fichero,
                     schedule = "DAILY",
                     starttime = format(Sys.time(), "11:39:00"),
                     startdate = format(Sys.time(), "30/04/2024"))

#taskscheduler_delete("download_CMEMS_data")
```

Figure 3: Example of the R code used to activate the data gathering and scheduling routine. Code available in https://github.com/AzWhaleLab/OCEAN.

4.1.2 Model predictions

The first step to create model projections is to define the study area, in this case the Azores region. Since not all covariates have the same spatial resolution or are referenced to the same geodetic datum, a single grid with the appropriate spatial resolution is defined for the study region and all covariates are reprocessed in reference to that grid (Figure 4). This step also includes a check to ensure that all necessary covariates are available prior to running the model (Figure 4).

```
#prediction grid. Define resolution, area and coordinate system.
template=raster()
res(template)=0.1
xmin(template)=-33
xmax(template)=-22
ymin(template)=35
ymax(template)=42
crs(template)="+proj=longlat +datum=WGS84 +no_defs"


grid<-read.csv(paste0(staticdir, "grid_cntr.csv",sep=""))
grid$X<-NULL
pma_base<-data.frame(Long=grid$Lng_cntwgs84,
                     Lat=grid$Lt_cntwgs84,
                     observer=1)

# Resample and transform dynamic variables
#chl
chl_nc <-sample(list.files(paste(envdir,"netcdfs_raw/temp_",get_date,"/chl",sep="")), 1)
file_chl<-paste(envdir,"netcdfs_raw/temp_",get_date,"/chl/",chl_nc,sep="")
chla<-raster(file_chl, varname="CHL")
chla_new<-log(raster::resample(chla, template, method="bilinear")+0.001)
writeRaster(chla_new,paste(finaldir,"/chl",sep=""),overwrite=T)


#sst
sst_nc <-sample(list.files(paste(envdir,"netcdfs_raw/temp_",get_date,"/sst",sep="")), 1)
file_sst <-paste(envdir,"netcdfs_raw/temp_",get_date,"/sst/",sst_nc,sep="")
sst<-raster(file_sst, varname="analysed_sst")
sst=(raster::resample(sst, template, method="bilinear")-273.15)
writeRaster(sst,paste(finaldir,"/sst",sep=""),overwrite=T)

#Check if no environmental data is missing
FileList_get_date=list.files(paste(envdir,get_date,sep=""),pattern="*.grd$")
FileList_full=c("sst.grd","chl.grd","chl_2m.grd")
FileList_missing=setdiff(FileList_full, FileList_get_date)
FileList_final=list.files(paste(envdir,get_date,sep=""),patter="*.grd$",full.names=T)
return_list=list("FileList_final"=FileList_final, "FileList_missing"=FileList_missing)
#if FileList_missing = 0 then the model can run
return_list
```

Figure 4: Example of the code to reprocess covariates for a single reference grid. Code available in https://github.com/AzWhaleLab/OCEAN.

With all predictive covariates ready, the model can be run to obtain a predictive map, projected on a two-dimensional space, covering the study area and using the spatial resolution set by the pre-defined grid, as a raster file. It is noteworthy that the quantities for each cell in the raster file are defined by the model. In the case illustrated here, the sperm whale model calculates the probability of occurrence in a scale ranging from 0 to 1, but depending on the data and modelling approach used, the quantities could as well represent a habitat quality score. Nevertheless, since the prediction tool framework is agnostic in relation to the units used, this has no effect in the outcome of the process.

For the sperm whale model illustrated here, two outputs are created: 1) daily prediction of mean probability of occurrence; and 2) standard deviation (SD) showing the measure of stability of the predicted distribution, with stable and unstable habitat preferences represented by low and high SD (Figure 5). This second output is used internally only to enable evaluating results and is not processed further. As such, it is not required for the workflow and can be commented out of the script or replaced by other evaluation method.

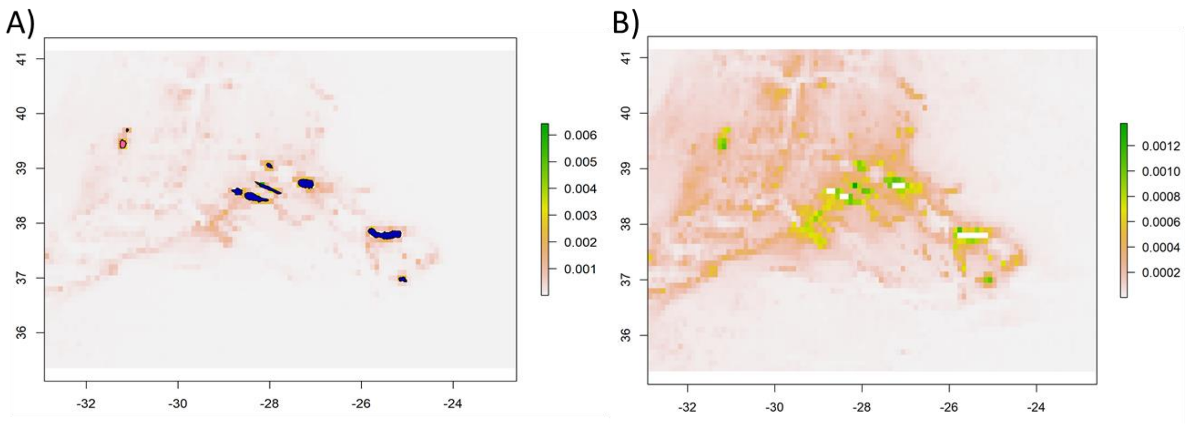

Figure 5: Example of maps of predicted daily sperm whale occurrence probability for the Azores region. A) probability of occurrence, and B) Standard deviation.

4.1.3 Polygon extraction and classification

Raster files are not supported by current NW standards and, consequently, the predictions in raster format created in the previous step cannot be used for that purpose.

To circumvent that limitation, the subsequent step is to run a process to convert the information contained in the raster files to a set of polygons that can be transmitted as NW.

The grid cell values in the raster file produced in the previous step are extracted along with the cell geographic coordinates. Those are processed to keep only values above a defined threshold that represent the areas with higher probability of whale occurrence (Figure 6).

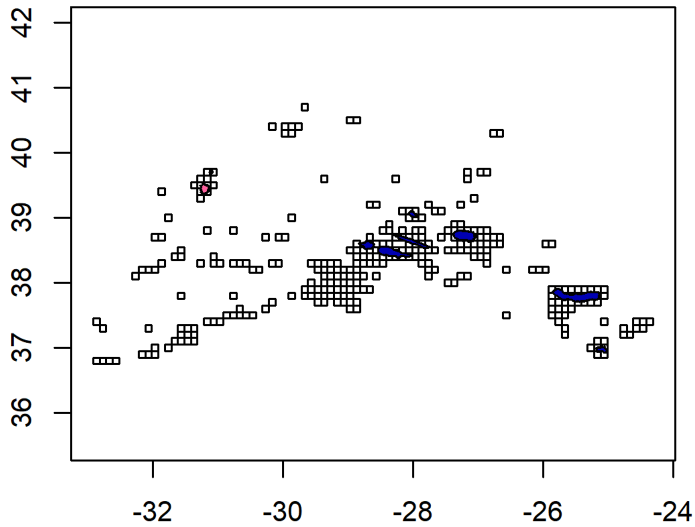

Figure 6: Map of grid cells with values higher than the quantile 95 threshold applied to daily predictions of sperm whale occurrence probability for the Azores region.

For the sperm whale model, a threshold of values above the quantile 95% was chosen as an illustrative example. However, it must be stressed that that value was not subject to any systematic test. It is used here to illustrate the process but can be considered arbitrary without further research. The choice of this threshold is critical to balance the goal of reducing ship strikes with the goal of not affecting shipping operations negatively more than necessary. For more detail on the importance of a careful choice of thresholds, the reader is referred to D4.1 (Report on the *Marine Mammal Ship Strike Mitigation Workshop*[8]).

To simplify the geometries further (consequently reducing file size) a 'dissolve' procedure is applied to merge contiguous grid cells, and polygons containing only one grid cell (~93km²) are discarded (Figure 7).

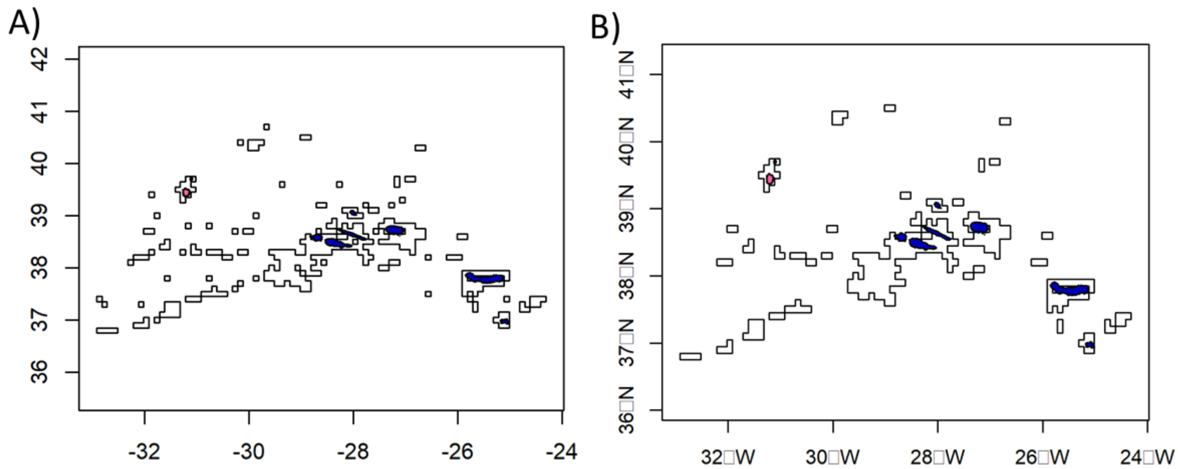

Figure 7: A) Contiguous grid cells merged to obtain the polygons. B) Final polygons selected after small polygons are eliminated.

The resulting georeferenced polygons are stored and fed into the final step, below.

4.1.4 Posting

The objective of this last routine of the framework is to send the information of each polygon created in 4.1.3 to the ENHI. Using a PowerShell script, the polygon data is formatted according to the requirements of UC-010[4] and merged with the metadata required by the API:

- Current date (ts): date that the file was created. The format of the date follows the international standards recommended for writing the date and time as year, month, day and then hours, minutes and seconds: YYYY-MM-DD hh:mm:ss
- Expiration date (exp): date of expiration of the file. Same format as the "Current date".
- Type (type): describe the type of the message sent. In this case, MM for marine mammals
- Category (cat): describe the category of the message sent. In this case, LW for large whales.
- Species (spc): describe the species of the message sent. In this case, SPW for sperm whales.
- Number of individuals (qty): describe the number of individuals on that polygon. In this case, we set it to '0', as we only have probability of presence data, ranging from 0 to 1).
- Boundaries of the polygon (loc): includes the position of the boundaries of the polygon. The position is formatted as Decimal Degrees (DDD.DDDDD°, DDD.DDDDD°; Latitude and Longitude information, respectively).

[8]https://ocean-navigation-awareness.eu/wp-content/uploads/2023/07/D04_1_ShipStrikesWorkshop_OCEAN_v2.1-1606KB.pdf

ts	exp	type	cat	spc	qty	loc
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[36.84972838,-32.81638076],[36.84972838,-32.71638076],[36.84972838,-32.61638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[37.14972838,-31.61638076],[37.24972838,-31.61638076],[37.24972838,-31.51638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[37.34972838,-31.11638076],[37.34972838,-31.21638076],[37.44972838,-31.21638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[37.44972838,-30.81638076],[37.44972838,-30.91638076],[37.54972838,-30.91638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.24972838,-30.81638076],[38.34972838,-30.81638076],[38.34972838,-30.71638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[39.54972838,-31.31638076],[39.64972838,-31.31638076],[39.64972838,-31.21638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[37.84972838,-29.71638076],[37.94972838,-29.71638076],[37.94972838,-29.61638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.14972838,-30.31638076],[38.14972838,-30.41638076],[38.14972838,-30.51638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.64972838,-29.91638076],[38.64972838,-30.01638076],[38.64972838,-30.11638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.94972838,-29.81638076],[38.94972838,-29.91638076],[38.84972838,-29.91638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[39.64972838,-29.51638076],[39.64972838,-29.41638076],[39.64972838,-29.31638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[40.24972838,-29.81638076],[40.24972838,-29.91638076],[40.14972838,-29.91638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[40.34972838,-29.31638076],[40.34972838,-29.41638076],[40.44972838,-29.41638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[40.64972838,-29.61638076],[40.64972838,-29.71638076],[40.74972838,-29.71638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[39.24972838,-28.71638076],[39.24972838,-28.61638076],[39.24972838,-28.51638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[40.44972838,-28.91638076],[40.44972838,-29.01638076],[40.54972838,-29.01638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.04972838,-27.41638076],[38.04972838,-27.31638076],[37.94972838,-27.31638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.34972838,-27.71638076],[38.24972838,-27.71638076],[38.24972838,-27.61638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.54972838,-28.01638076],[38.54972838,-28.11638076],[38.64972838,-28.11638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.84972838,-27.41638076],[38.94972838,-27.41638076],[38.94972838,-27.31638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.64972838,-27.21638076],[38.64972838,-27.11638076],[38.64972838,-27.01638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[38.74972838,-28.31638076],[38.74972838,-28.41638076],[38.74972838,-28.51638076],
30/10/2024 17:24	31/10/2024 18:24	MM	LW	SPW	0	[[39.14972838,-28.01638076],[39.14972838,-27.91638076],[39.04972838,-27.91638076],

Figure 8: Example of data sent to the ENHI with the information of the selected polygons (partial view of the data table).

To finalize the process, a web request is made to communicate with the API in the ENHI and, upon response of the server, the diagnostic file is stored locally.

5 Validation and testing

During the development stage, isolated testing of each of the components was carried out by running the scripts and testing for the expected outcome. If results were in the form of georeferenced products (raster files and polygons), a Geographic Information System (ArcGIS) was used to visually inspect products and check for the correct use of geographic projection.

Subsequently, the scripts were tested in sequence, to calculate processing time for each script and allow for adequate time intervals between tasks to guarantee that the predictive tool framework runs properly.

In a close collaboration with WP6, communication tests with the ENHI API were performed and some bugs detected and corrected.

Finally, the entire predictive tool framework was moved to the workstation and deployed unattended for several days.

Proper communication with the ENHI was confirmed using the diagnostic files produced by the ENHI.

At the predictive tool side, confirmation messages sent by the API allow for immediate confirmation of successful transmission (Figure 9).

On the other hand, on the ENHI side, all communications are recorded in the daily activity logs, offering a further way of confirming communications and data transmissions (Figure 10).

Furthermore, since a diagnostic line is produced for each submitted polygon, it was possible to cross-check the number of polygons produced by the predictive tool infrastructure with those received by the ENHI.

```
___________________________
Processing line: 1
200, OK, ["I002",null]

___________________________
Processing line: 2
200, OK, ["I002",null]

___________________________
Processing line: 3
200, OK, ["I002",null]

___________________________
Processing line: 4
200, OK, ["I002",null]

___________________________
Processing line: 5
200, OK, ["I002",null]

___________________________
Processing line: 6
200, OK, ["I002",null]

___________________________
Processing line: 7
200, OK, ["I002",null]
```

Figure 9: Example of diagnostic file received from the ENHI after submission of message containing multiple polygons, showing the expected output ["I002", null] meaning that the report was received successfully for each polygon sent.

```
10:40:57.413;INFO;I.800;Creating new log file [20241028.log]
10:40:57.414;INFO;I.801;Attempting to vacuum older log files
10:40:57.422;INFO;I.900;Module initialization [input]
10:40:57.453;INFO;I.900;Module initialization [db]
10:40:57.475;INFO;I.901;Database successfully connected
10:40:57.764;INFO;I.000;Start of application
10:40:57.765;INFO;I.902;Selecting entry point for use-case [010] [xxxxxxxxx@gmail.com_1730112057.314417]
10:40:57.765;INFO;I.903;Specific entry point for use-case (UC) [010]
10:40:57.765;INFO;I.904;Specific entry point for sub-process (SP) [100]
10:40:57.834;INFO;I.001;User validated successfully [xxxxxxxxx@gmail.com]
10:40:57.839;INFO;I.904;Specific entry point for sub-process (SP) [003]
10:40:57.847;INFO;I.003;Record inserted or updated [4915] [DATA]
10:40:57.854;INFO;I.002;Report received successfully
10:40:57.856;INFO;I.905;Request Extra Data is only executed for (O)bservations and (D)etections
10:40:57.856;INFO;I.999;End of application [Finished in 0.0919203758239746l seconds]
10:40:58.343;INFO;I.900;Module initialization [input]
10:40:58.374;INFO;I.900;Module initialization [db]
10:40:58.395;INFO;I.901;Database successfully connected
10:40:58.677;INFO;I.000;Start of application
10:40:58.677;INFO;I.902;Selecting entry point for use-case [010] [xxxxxxxxx@gmail.com_1730112058.2464597]
10:40:58.677;INFO;I.903;Specific entry point for use-case (UC) [010]
10:40:58.677;INFO;I.904;Specific entry point for sub-process (SP) [100]
10:40:58.747;INFO;I.001;User validated successfully [xxxxxxxxx@gmail.com]
10:40:58.751;INFO;I.904;Specific entry point for sub-process (SP) [003]
10:40:58.757;INFO;I.003;Record inserted or updated [4916] [DATA]
10:40:58.761;INFO;I.002;Report received successfully
10:40:58.763;INFO;I.905;Request Extra Data is only executed for (O)bservations and (D)etections
10:40:58.763;INFO;I.999;End of application [Finished in 0.08591914176940918 seconds]
```

Figure 10: Example of ENHI daily activity log file with record of login and transmissions by the predictive tool.

Finally, integrity of the data was confirmed in two ways: 1) by querying the database (Figure 11) and 2) by checking the polygon shapes sent by the ENHI to the voluntary reporting app against the original shapes produced by the predictive tool (Figure 12).

```sql
select d.id as ID, d.qty, d.type, d.category, d.species, d."generated", d.expires, l.seq as SEQ, l.lat, l.lon
from data d, location l
where d.src_user = 'rcabprieto@gmail.com' and d.id=l.id_data and "generated">'2024-10-28 00:00:00'
union
select da.id, da.qty, da.type, da.category, da.species, da."generated", da.expired, la.seq, la.lat, la.lon
from data_archive da, location_archive la
where da.src_user = 'rcabprieto@gmail.com' and da.id=la.id_data and "generated">'2024-10-28 00:00:00'
order by ID, SEQ
```

id	qty	type	category	species	generated	expires	seq	lat	lon
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	0	-31.496422	37.163094
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	1	-31.51638	37.149727
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	2	-31.51638	37.24973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	3	-31.51638	37.349728
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	4	-31.41638	37.349728
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	5	-31.41638	37.24973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	6	-31.316381	37.24973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	7	-31.316381	37.149727
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	8	-31.316381	37.04973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	9	-31.41638	37.04973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	10	-31.51638	37.04973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	11	-31.61638	37.04973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	12	-31.716381	37.04973
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	13	-31.716381	37.149727
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	14	-31.61638	37.149727
4,915	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	15	-31.51638	37.149727
4,916	0	MM	LW	SPW	2024-10-28 09:50:00.000	2024-10-29 09:50:00.000	0	-31.130676	37.392593

Figure 11: Example of database query with messages sent by the predictive tool defining a polygon. Only the first line for the second polygon is shown at the bottom of the image. Quantity is set to '0' on purpose, as this is the standard for polygons.

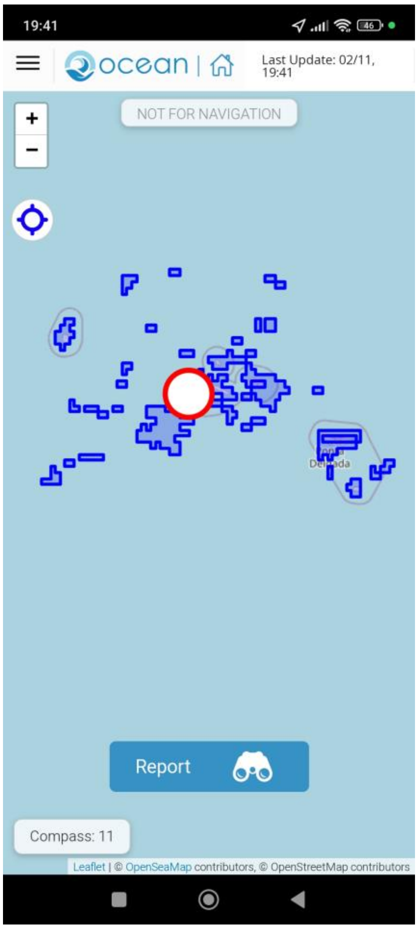

Figure 12: Screenshot of the voluntary reporting app (developed under Task 4.4) centred in the Azores archipelago, showing polygons for areas with high sperm whale occurrence probability issued by the predictive tool.

As the predictive tool framework passed all tests over a test period of one week, it was considered ready for publication.

6 Conclusions

6.1 Main results

The work developed under Task 4.3 resulted in the successful creation of a stand-alone predictive tool of increased probability of occurrence of marine mammals, having the following capabilities:

- Streamlines and automates the production of predictive maps of whale (or other organisms) habitat, regardless of modelling approach.
- Allows for conversion of raster maps to georeferenced polygons for transmission using standard protocols for navigational warnings.
- Integrates flawlessly with the ENHI.
- Due to the modular structure, this tool is easily customizable for other user cases (the user can use all or just part of the code, and can easily change parameters to fit specific cases); is scalable (can run multiple models); and can easily be updated with new models whenever those become available.

6.2 Limitations

The predictive tool framework is well suited to the goals set forth in the Grant Agreement, namely the transmission of information on DRA based on ENM results. The way the information is presented is currently limited by bandwidth and, most importantly, the specifications of NW protocols.

These limitations dictate that the information sent is formatted as a polygon with binary information on whale probability of occurrence. This forces the use of thresholds that in many cases may be difficult to select (see discussion in 4.1.3). However, that limitation is not intrinsic to the predictive tool framework and as such cannot be resolved at this level. In fact, the way the predictive tool is designed allows for the submission of mapped information as floating point raster files, if the NW protocols are changed to allow that.

The tool was designed to be as cross-platform as possible, with only few tasks depending on OS conditioned codes, namely Windows Task Scheduler. Nevertheless, these can be easily changed to utilize other scheduling services running on other OS.

Most of the codding was done in the statistical language *R* because it is well-known in the scientific community, specifically for environmental niche modelling. Arguably, the processing speed can be improved by re-writing the scripts in another more efficient language. However, currently processing speed was not considered an issue, especially because the model utilized is relatively simple. More complex models, or multiple models running in parallel may prove to be more processor intensive and benefit from using a more efficient programming language. In that respect, translating from *R* to other programming languages is not complex and there are even tools that help in automating that translation (e.g. https://www.codeconvert.ai).

6.3 Further work

Software development and testing ends with the delivery of D4.3 and this report (see section 7 below). The software is now ready for demonstration (Task 4.9), supported and coordinated by WP9.

7 Deliverable form and availability

The Deliverable for Task 4.3 (D4.3) is the code for executing the predictive tool, along with the description of the framework presented in this report. All the code, as well as the sperm whale model (D4.2) and data to reproduce it are all available through the GitHub repository in https://github.com/AzWhaleLab/OCEAN.

8 References

Hijmans, R. J. (2024). raster: Geographic Data Analysis and Modeling V3.6-30 (Version 3.6-30). http://CRAN.R-project.org/package=raster. Retrieved from http://CRAN.R-project.org/package=raster

Maxwell, S. M., Hazen, E. L., Lewison, R. L., Dunn, D. C., Bailey, H., Bograd, S. J., . . . Crowder, L. B. (2015). Dynamic ocean management: Defining and conceptualizing real-time management of the ocean. *Marine Policy, 58*, 42-50. doi:https://doi.org/10.1016/j.marpol.2015.03.014

Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal, 10*(1), 439-446. doi:10.32614/RJ-2018-009

R Development Core Team. (2024). R: A Language and Environment for Statistical Computing. Retrieved from http://www.R-project.org

Spinu, V., Grolemund, G., & Wickham, H. (2023). lubridate: Make Dealing with Dates a Little Easier V1.9.3 (Version 1.9.3). http://CRAN.R-project.org/package=lubridate. Retrieved from http://CRAN.R-project.org/package=lubridate

Wijffels, J., & Belmans, O. (2023). taskscheduleR: Schedule R Scripts and Processes with the Windows Task Scheduler V1.8 (Version 1.8). http://CRAN.R-project.org/package=taskscheduleR. Retrieved from http://CRAN.R-project.org/package=taskscheduleR

Wood, S. N. (2017). *Generalized Additive Models: An introduction with R. Second Edition* (Second edition ed.). Boca Raton: Chapman & Hall/CRC.

9 Annex 1: The Consortium

Høgskulen på Vestlandet	Inndalsveien 28 P.O Box 7030 5020 Bergen, Norway	https://www.hvl.no/en/
KYSTVERKET NORWEGIAN COASTAL ADMINISTRATION	Flathauggata 12 5523 Haugesund, Norway	www.kystverket.no
	Heroon Polytechniou 9 Zographou campus 15780 Athina, Greece	www.ntua.gr
CIMNE EXCELENCIA SEVERO OCHOA	C Gran Capitan, Edifici C1, Campus Nord UPC SN 08034 Barcelona, Spain	www.cimne.com
TELEDYNE RESON Everywhereyoulook	Fabriksvangen 13 3550 Slangerup, Denmark	www.teledynemarine.com/reson
OKEANOS	Rua Da Mae De Deus Reitoria Apatado 1422 9501 801, Ponta Delagada S Miguel Acores Portugal	www.uac.pt
KONGSBERG	Kongsberg Maritim: Strandpromenaden 50 Po box: 1009 3183 Horten, Norway	www.kongsberg.com
KONGSBERG	Kongsberg Seatex: Pirsenteret 7462 Trondheim, Norway	www.kongsberg.com
UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH	Calle Jordi Girona 31 08034 Barcelona, Spain	www.upc.edu
IRISH FERRIES	Alexandra Road Ferryport D01 W2F5 Dublin, Ireland	www.irishferries.com
IWDG	Merchants Quay V15 E762 Kilrush, Clare Ireland	www.iwdg.ie
The Nautical Institute	Lambeth road 202, SE 7LQ London, United Kingdom	www.nautinst.org
LR	Fenchurch street 71, EC3M 4BS, United Kingdom	www.lr.org

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101076983. UK participants in Project OCEAN are supported by UKRI grant numbers 10038659 (Lloyds Register) and 10052942 (The Nautical Institute).

10 Annex 2: Project Summary

The OCEAN project is focused on enhancing operator awareness in navigation, to reduce the frequency of severe accidents like collision and grounding, to mitigate ship-strike risks to marine mammals, and to mitigate the risk presented by floating obstacles to ships.

The OCEAN project will contribute to an improved understanding of accident root causes, and will strive to reduce the resulting human, environmental and economic losses through socio-technical innovations supporting ship navigators.

The OCEAN consortium, coordinated by Western Norway University of Applied Sciences, includes 13 partner organizations across 7 different European countries from the industry, academia, NGOs and end users.

Around 3.000 maritime incidents occur every year in the European maritime fleet. 28% of these accidents are categorised as severe or very severe accidents, resulting in the loss of life onboard, pollution, fire, collisions or grounding. Navigational accidents are dominant in these statistics according to the European Maritime Safety Agency, be it for cargo, passenger or service ships.

The OCEAN project ambition is to contribute to the mitigation of navigational accidents by supporting the navigators to do an even better job than they do presently. The OCEAN consortium will address the most pertinent factors that may contribute to events becoming accidents: training, technical, human or organisational factors, operational constraints, processes and procedures, commercial pressures, and will recommend improvements and amendments to regulations, standards and bridge equipment design approaches.

OCEAN seeks to enhance navigational awareness “on the spot” and to improve the performance of evasive manoeuvring to avoid collision with near-field threats. The project will deliver and demonstrate several human centred innovations. For example, the 4D Situation Awareness Display which will be developed in the OCEAN project will improve the visualisation of navigational hazards, integrating current bridge information systems with marine mammal and lost floating containers detection and tracking capacity specifically developed by the project.

Going further, the project will design and implement a European navigational hazard data infrastructure to feed multi-source observations and hazard predictions relating to floating containers and large aggregations of marine mammals into the existing distributed maritime warning infrastructure. OCEAN seeks to transfer this data ecosystem to relevant European organisations for deployment and maintenance.

Co-funded by Horizon Europe, the European Union’s research and innovation programme, the consortium of 13 members represents 7 European countries, Norway, Greece, Spain, Denmark, Portugal, Ireland and UK, all located on major European coastal regions. Members include a coastal administration, a ship operator, maritime safety and transport researchers, marine mammal ecology and conservation experts, companies specialised in maritime information systems and sensors, a professional organisation, a risk and safety management organisation, as well as data infrastructure, data fusion and satellite imaging specialists.

UK participants are supported by UK Research and Innovation Grant Number 10038659 (Lloyd’s Register) and Grant Number 10052942 (The Nautical Institute).

11 Annex 3: Software Code

```
The codes to run and operationalize the framework described in this document are
reproduced in below:

1.     “Download_data_from_CMEMS_DOM_tool” (R code)

2.     ”Run_model_and_create_raster_automatize” (R code)

3.     “Create_final_output_automatize” (R code)

4.     “sendData_Acores.ps1” (Power Shell code)



1.     “Download_data_from_CMEMS_DOM_tool” (R code)



# Set working directories and load R packages



path="path where the DOM tool is stored"

envdir=”path to store environmental data”

azores_shp="path to the Azores shapefile”

chl_dir=”path to the folder with the Chlorophyll products”

sst_dir=”path to the folder with the Sea Surface Temperature products”



#Path to copernicusmarine (object of the Copernicus Marine Toolbox)

path_copernicusmarine: “path directory”



USERNAME<-"xxxxx"

PASSWORD<-"xxxxx"



#get actual date

get_date=Sys.Date()

#lag date (2 months)
```

This project has received funding from the European Union’s Horizon Europe research and innovation programme under grant agreement No. 101076983. UK participants in Project OCEAN are supported by UKRI grant numbers 10038659 (Lloyds Register) and 10052942 (The Nautical Institute).

```
most_recent=as.character(Sys.Date()-60)


###Create folders to store daily products (netcdf files).

###Create individual folder for each product


tmpdir=paste(envdir,"netcdfs_raw/temp_",get_date,sep="")

if (!file.exists(tmpdir)){

  dir.create(tmpdir)

}


tmpdir_chl=paste(envdir,"netcdfs_raw/temp_",get_date,"/chl",sep="")

if (!file.exists(tmpdir_chl)){

  dir.create(tmpdir_chl)

}


tmpdir_chl_2m=paste(envdir,"netcdfs_raw/temp_",get_date,"/chl_2m",sep="")

if (!file.exists(tmpdir_chl_2m)){

  dir.create(tmpdir_chl_2m)

}


tmpdir_sst=paste(envdir,"netcdfs_raw/temp_",get_date,"/sst",sep="")

if (!file.exists(tmpdir_sst)){

  dir.create(tmpdir_sst)

}


# ================ Variables for your query ================


###SST sea surface temperature######
```

```
# Product and dataset IDs

productId = "cmems_obs-sst_glo_phy_nrt_l4_P1D-m"


# Ocean Variable(s)

#Please keep the space at the beginning (" --variable "variable name"")

variable_sst <- c(" --variable analysed_sst")


# Time range

date_min = ymd(get_date) # start_date

date_max = ymd(get_date) # end_date


# Geographic area and depth level

lon = list(-60, 35)  # lon_min, lon_max

lat = list(-40,70) # lat_min, lat_max

depth = list(0, 1) # depth_min, depth_max


# Output filename

#out_name_chl = paste("cmems_obs-oc_glo_bgc-plankton_nrt_l4-gapfree-multi-
4km_P1D_",date_min,".nc", sep="")


out_name_sst = paste("SST", ".nc", sep="")


command_sst <- paste (path_copernicusmarine, " subset -i", productId,

                      "-x", lon[1], "-X", lon[2],

                      "-y", lat[1], "-Y", lat[2],

                      "-t", date_min, "-T", date_max,

                      "-z", depth[1], "-Z", depth[2],

                      variable_sst, "-o", tmpdir_sst,
```

```
                       "-f", out_name_sst, "--force-download",

                       sep = " ")


print(command_sst)


system(command_sst, intern = TRUE)


###CHL ######


# Product and dataset IDs

productId = "cmems_obs-oc_glo_bgc-plankton_nrt_l4-gapfree-multi-4km_P1D"


# Ocean Variable(s)

#Please keep the space at the beginning (" --variable "variable name"")

variable_chl <- c(" --variable CHL")


# Time range

date_min = ymd(get_date) # start_date

date_max = ymd(get_date) # end_date


# Geographic area and depth level

lon = list(-60, 35)  # lon_min, lon_max

lat = list(-40,70) # lat_min, lat_max

depth = list(0, 1) # depth_min, depth_max


# Output filename

out_name_chl = paste("CHL", ".nc", sep="")
```

```
command_chl <- paste (path_copernicusmarine, " subset -i", productId,
                  "-x", lon[1], "-X", lon[2],
                  "-y", lat[1], "-Y", lat[2],
                  "-t", date_min, "-T", date_max,
                  "-z", depth[1], "-Z", depth[2],
                  variable_chl, "-o", tmpdir_chl,
                  "-f", out_name_chl, "--force-download",
                  sep = " ")


print(command_chl)


system(command_chl, intern = TRUE)


###CHL the previous 2 months######


# Time range
date_min = ymd(most_recent) # start_date
dates<-as.POSIXct(paste(date_min, "00:00:00", sep=""))


date_min<-as.Date(dates-1*24*60*60*11)


date_max = ymd(most_recent) # start_date
dates<-as.POSIXct(paste(date_max, "00:00:00", sep=""))


date_min<-as.Date(dates-1*24*60*60*11)


# Geographic area and depth level
lon = list(-60, 35)  # lon_min, lon_max
```

```
lat = list(0,70) # lat_min, lat_max

depth = list(0, 1) # depth_min, depth_max


# Output filename

out_name_chl_2m = paste("CHL_2m", ".nc", sep="")


command_chl_2m <- paste (path_copernicusmarine, " subset -i", productId,

                  "-x", lon[1], "-X", lon[2],

                  "-y", lat[1], "-Y", lat[2],

                  "-t", date_min, "-T", date_max,

                  "-z", depth[1], "-Z", depth[2],

                  variable_chl, "-o", tmpdir_chl_2m,

                  "-f", out_name_chl_2m, "--force-download",

                  sep = " ")


print(command_chl_2m)


system(command_chl_2m, intern = TRUE)


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

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


2.     ”Run_model_and_create_raster_automatize” (R code)


# Set working directories and load R packages


path="path where the DOM tool is stored"

envdir=”path to store environmental data”
```

```
staticdir=”path to static variables data”

gam_dir=”path to GAM outputs”

oceandir= ”path to daily predictions”

azores_shp="path to the Azores shapefile”

chl_dir=”path to the folder with the Chlorophyll products”

sst_dir=”path to the folder with the Sea Surface Temperature products”


#get actual date

get_date=Sys.Date()


###Create folders to store daily products (netcdf files).


finaldir=paste(envdir,get_date,sep="");dir.create(finaldir)

if(!file.exists(finaldir)){

  dir.create(finaldir)

}


#prediction grid. Define resolution, area and coordinate system.

template=raster()

res(template)=0.1

xmin(template)=-33

xmax(template)=-22

ymin(template)=35

ymax(template)=42

crs(template)="+proj=longlat +datum=WGS84 +no_defs"


grid<-read.csv(paste0(staticdir, "grid_cntr.csv",sep=""))

grid$X<-NULL
```

```
pma_base<-data.frame(Long=grid$Lng_cntwgs84,
                     Lat=grid$Lt_cntwgs84,
                     observer=1)


# Resample and transform dynamic variables

#chl

chl_nc <-sample(list.files(paste(envdir,"netcdfs_raw/temp_",get_date,"/chl",sep="")), 1)

file_chl<-paste(envdir,"netcdfs_raw/temp_",get_date,"/chl/",chl_nc,sep="")

chla<-raster(file_chl, varname="CHL")

chla_new<-log(raster::resample(chla, template, method="bilinear")+0.001)

writeRaster(chla_new,paste(finaldir,"/chl",sep=""),overwrite=T)


#sst

sst_nc <-sample(list.files(paste(envdir,"netcdfs_raw/temp_",get_date,"/sst",sep="")), 1)

file_sst <-paste(envdir,"netcdfs_raw/temp_",get_date,"/sst/",sst_nc,sep="")

sst<-raster(file_sst, varname="analysed_sst")

sst=(raster::resample(sst, template, method="bilinear")-273.15)

writeRaster(sst,paste(finaldir,"/sst",sep=""),overwrite=T)


#Check if no environmental data is missing

FileList_get_date=list.files(paste(envdir,get_date,sep=""),pattern="*.grd$")

FileList_full=c("sst.grd","chl.grd","chl_2m.grd")

FileList_missing=setdiff(FileList_full, FileList_get_date)

FileList_final=list.files(paste(envdir,get_date,sep=""),patter="*.grd$",full.names=T)

return_list=list("FileList_final"=FileList_final, "FileList_missing"=FileList_missing)

#if FileList_missing = 0 then the model can run

return_list
```

```
capture.output(return_list, file =paste(envdir,'return_list.csv'))


#Extract dynamic variables data from rasters


varname<-c("log_chl","sst")

coordinates(grid)<-~Lng_cntwgs84+Lt_cntwgs84

for(i in 1:2){

  r<-raster(return_list$FileList_final[i])

  e<-raster::extract(r,grid)

  df<-as.data.frame(e)

  names(df)<-varname[i]

  if(i==1){

    df1<-df} else {

      df2<-cbind(df1,df)

      df1<-df2

    }

}


dym_var<-df2


### Load GAM model ###


pma_gam<-read.csv(paste0(gam_dir,"pma_final_gam2.csv"),header=T)

gam<-gam(presence ~ s(sst)+s(log_chl_2m)+s(depth)+s(sqrt_dist_smnt), data=pma_gam,
family=binomial, method="REML")

summary<-summary(gam)


### Predict daily distribution of sperm whales using best GAM model ###
```

```
#Open static variables

stat_var<-read.csv(paste0(staticdir,"stat_var_grid.csv"),header=T)



#Combine dynamic and static variables data for prediction

pred_df<-cbind(dym_var, stat_var)



#Transform variables from the prediction data

pred_df$sqrt_dist_smnt<-sqrt(pred_df$dist_smnt)



pred_df$log_chl_2m<-pred_df$log_chl



#predict outputs

pred<-data.frame(predict(gam, se.fit=T, newdata=pred_df, type="response",
backtransform=F))



write.csv(pred,paste(envdir, "pred.csv"))



grid<-as.data.frame(grid)

lon<-grid$Lng_cntwgs84

lat<-grid$Lt_cntwgs84

pred_map<-cbind(lon, lat, pred)



write.csv(pred_map,paste(envdir, "pred_map.csv"))



#Create prediction map (raster)

pred_fit<-pred_map[-c(4)]

sp::coordinates(pred_fit) <- ~lon+lat

sp::proj4string(pred_fit) <-CRS("+proj=longlat +datum=WGS84 +no_defs")

# coerce to SpatialPixelsDataFrame
```

```
gridded(pred_fit) <- TRUE

# coerce to raster

rasterDF <- raster(pred_fit)

writeRaster(rasterDF, paste(oceandir,"Predictions_fit_",get_date, sep=""),format =
"GTiff", overwrite = TRUE)



png(paste0(oceandir,"Predictions_fit_",get_date,
sep="",'.png'),width=8,height=6.5,units="in",res=300)

plot(rasterDF)

dev.off()



##add Azores

azores <- read_sf(paste0(azores_shp,'/Azores.shp'))



#Create prediction map with Azores (.png)

png(paste0(oceandir,"Azores_Predictions_fit_",get_date,
sep="",'.png'),width=8,height=6.5,units="in",res=300)

plot(rasterDF)

plot(azores,add=T)

dev.off()



#Create prediction map (standard_error_map)

pred_se<-pred_map[-c(3)]

sp::coordinates(pred_se) <- ~lon+lat

sp::proj4string(pred_se) <-CRS("+proj=longlat +datum=WGS84 +no_defs")

# coerce to SpatialPixelsDataFrame

gridded(pred_se) <- TRUE

# coerce to raster

rasterDF <- raster(pred_se)

rasterDF
```

```
writeRaster(rasterDF, paste(oceandir,"Predictions_se_",get_date, sep=""),format =
"GTiff", overwrite = TRUE)



png(paste0(oceandir,"Predictions_se_",get_date,
sep="",'.png'),width=8,height=6.5,units="in",res=300)

plot(rasterDF)

dev.off()



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

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



3.     “Create_final_output_automatize” (R code)



# Set working directories and load R packages

path=&quot;path where the DOM tool is stored&quot;

envdir=”path to store environmental data”

oceandir= ”path to daily predictions”

azores_shp=&quot;path to the Azores shapefile”

path_powershell=&quot;path to powershell output”

#get actual date

get_date=Sys.Date()

# Specify the path to your raster file

rasterDF&lt;-raster(paste(oceandir, &quot;Predictions_fit_&quot;,get_date,
&quot;.tif&quot;,sep=&quot;&quot;))

# Extract values from raster

data_values &lt;- rasterDF[]

#Eliminate NA values

data_values &lt;- data_values[!is.na(data_values)]

write.csv(data_values,paste(envdir, &quot;data_values.csv&quot;))

#Choose threshold for minimum value of grid cells values
```

```
#quantile 95%

threshold_area &lt;- quantile(data_values,0.95)

#Select polygons of the quantile 95%

regions_shp &lt;- rasterToPolygons(rasterDF, fun=function(x){x&gt;threshold_area},na.rm =
T,

dissolve = T)

#convert SpatialPolygonsDataFrame to an sf object (library sf)

regions_shp_sf&lt;-st_as_sf(regions_shp)

#merge adjacent polygons

pol0&lt;-as_Spatial(regions_shp_sf)

pol1&lt;-st_union(st_as_sf(pol0))

# Save as GeoJSON

sf::write_sf(pol1, paste(envdir, &quot;pol1.geojson&quot;), driver = &quot;GeoJSON&quot;)

#Convert polygon to spatial polygons

pol1_sp &lt;- sf:::as_Spatial(pol1)

#write.csv(get_date,paste(envdir, &quot;get_date_mid.csv&quot;))

#Calculate area of each polygon

for(i in 1:length(pol1_sp@polygons[[1]]@Polygons)){

coords&lt;- as.data.frame(pol1_sp@polygons[[1]]@Polygons[[i]]@coords)

area &lt;- round(pol1_sp@polygons[[1]]@Polygons[[i]]@area, digits=2)

area&lt;-as.data.frame(rep(area, nrow(coords)))

ID&lt;-i

ID&lt;-as.data.frame(rep(ID, nrow(coords)))



df&lt;-cbind(coords, area,ID)

if(i==1){

df1&lt;-df} else {

df2&lt;-rbind(df1,df)

df1&lt;-df2
```

```
}

}

colnames(df2)[3] &lt;- &quot;area&quot;

colnames(df2)[4] &lt;- &quot;ID&quot;

#select polygon larger than 1 grid cell

df3 &lt;- df2[which(df2$area &gt; min(df2$area)),]

# Remove the &quot;Area&quot; column

df3 &lt;- subset(df3, select = -c(area))

buildings_df&lt;-df3

# make a list

buildings_list &lt;- split(buildings_df, buildings_df$ID)

# only want lon-lats in the list, not the names

buildings_list &lt;- lapply(buildings_list, function(x) { x[&quot;ID&quot;] &lt;- NULL; x
})

#convert to list of Polygons

ps &lt;- lapply(buildings_list, Polygon)

# add id variable

p1 &lt;- lapply(seq_along(ps), function(i) Polygons(list(ps[[i]]), ID =
names(buildings_list)[i]

))

# create SpatialPolygons object of the selected polygons

my_spatial_polys &lt;- SpatialPolygons(p1, proj4string = CRS(&quot;+proj=longlat

+datum=WGS84&quot;) )

#write.csv(get_date,paste(envdir, &quot;get_date.csv&quot;))

##add Azores

azores &lt;- read_sf(paste0(azores_shp,&#39;/Azores.shp&#39;))

zee &lt;- read_sf(paste0(azores_shp,&#39;/ZEE.shp&#39;))

###plot both plots together

# Set layout using layout()
```

```
##Map of the polygons with Azores islands shapefile##
png(paste0(oceandir,&quot;Azores_Polygons_defined_&quot;,get_date,
sep=&quot;&quot;,&#39;.png&#39;),width=8,height=6.5,units=&quot;in&quot;,res=300)
par(mar=c(4,4,2,2))
layout(matrix(c(1,2,3,3), 2, 2, byrow = TRUE))
plot(regions_shp,axes=T)
plot(azores,add=T)
plot(pol1_sp,axes=T)
plot(azores,add=T)

par(mar=c(2,14,2,14))
plot(my_spatial_polys,axes=T)
plot(azores,add=T)
dev.off()
##EEZ Azores
##Map of the polygons with EEZ Azores shapefile##
png(paste0(oceandir,&quot;Zee_Polygons_defined_&quot;,get_date,
sep=&quot;&quot;,&#39;.png&#39;),width=8,height=6.5,units=&quot;in&quot;,res=300)
par(mar=c(4,4,2,2))
layout(matrix(c(1,2,3,3), 2, 2, byrow = TRUE))
plot(zee$geometry,axes=T)
plot(regions_shp,add=T)
plot(azores,add=T)
plot(zee$geometry,axes=T)
plot(pol1_sp,add=T)
plot(azores,add=T)
par(mar=c(2,14,2,14))
plot(zee$geometry,axes=T)
```

```
plot(my_spatial_polys,add=T)

plot(azores,add=T)

dev.off()

##Reorganize output for sending

#dynamic directories

current_datetime &lt;- format(Sys.time())

current_datetime &lt;- as.POSIXct(current_datetime, format = &quot;%Y-%m-%d
%H:%M:%S&quot;,

tz=&quot;UTC&quot;)

#add 10 minutes to have an overlap of 5 minutes between the creation of the output and
the

time the info is sent to CIMNE

current_datetime_10m&lt;-current_datetime+ 1*10*60

# Eliminate characters after the first space

modified_vector &lt;- sub(&quot; -&quot;, &quot;&quot;, current_datetime_10m)

#Expiration date 25hours after the creation

expiration_date&lt;-current_datetime_10m+ 1*25*60*60

# Eliminate characters after the first space

expiration_date &lt;- sub(&quot; -&quot;, &quot;&quot;, expiration_date)

#expiration_date&lt;-paste(expiration_date, expiration_time, sep= )

# &quot;2024-05-07 10:42:49&quot;

#class &quot;character&quot;

##Change dataframe to the format LAT LONG

df3&lt;-df3[,c(2,1,3)]

#Adapt the polygon info to send to CIMNE

nlev_ID&lt;-nlevels(factor(df3$ID))

lev_ID&lt;-levels(factor(df3$ID))



for(j in 1:nlev_ID){
```

```
df_short &lt;- subset(df3, df3$ID==lev_ID[j])

coords&lt;- as.data.frame(df_short[,1:2])

for(i in 1:nrow(coords)){

coords_vector&lt;- as.character(coords[i,])

vector&lt;- paste0(&quot;[&quot;, coords_vector[1],&quot;,&quot;, coords_vector[2],
&quot;]&quot;)

if(i==1){

df1&lt;-vector}

else {

df2&lt;-paste0(df1,&quot;,&quot;, vector)

df1&lt;-df2}

###Add ID and ; and [] at the stard and end of the vector

df_new&lt;-
paste0(modified_vector,&quot;;&quot;,expiration_date,&quot;;MM;LW;SPW;9999;&quot;,&quot;[
&quot;,df1, &quot;]&quot;)}

if(j==1){

df_middle&lt;-df_new}

else {

df_final&lt;-rbind(df_middle,df_new)

df_middle&lt;-df_final

}

}

##Add ts;exp;type;cat;spc;qty;loc (format of the receiver at CIMNE)###

df_final_2&lt;-as.data.frame(df_final)

row_names&lt;-&quot;ts;exp;type;cat;spc;qty;loc&quot;

df_final_2&lt;-rbind(row_names,df_final_2)

# Remove row names from an existing data frame

rownames(df_final_2) &lt;- NULL

colnames(df_final_2) &lt;- NULL

# Write the data frame to a CSV file
```

```
write.csv(df_final_2,paste(oceandir,&quot;df_final.csv&quot;),row.names = FALSE,
col.names =

FALSE,quote = FALSE)

write.csv(df_final_2,&quot;C:/R/Projects/OCEAN/PowerShell/MM_output_new_old.csv&quot;,row
.n

ames = FALSE, col.names = FALSE,quote = FALSE)

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

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



4.     “sendData_Acores.ps1” (Power Shell code)



# Set working directories and load R packages



path_powershell="path to powershell output”



$timestamp = Get-Date -Format "yyyyMMddHHmmssfff"

$logfile = ".\" + $timestamp + ".txt"

$Uri = 'http://147.XX.YYY.ZZZ:WWWW/sendData'

$Form = @{}

$Form.src = "email address"

$Form.pwd = "password"

$Form.key = "key"

$model_output = Import-Csv -Path " path_powershell\MM_output_new_old.csv" -Delimiter ";"

$line = 0

ForEach ($row in $model_output){

       $Form.ts = $row.ts

       if ($row.exp -ne "") {

              $Form.exp = $row.exp

       }
```

```
        $Form.type = $row.type

        if ($row.cat -ne "") {

                $Form.cat = $row.cat

        }

        if ($row.spc -ne "") {

                $Form.spc = $row.spc

        }

        if ($row.qty -ne "") {

                $Form.qty = $row.qty

        }

        $Form.loc = $row.loc

$line = $line + 1

"__________________________" | Out-File -FilePath $logfile -Append

"Processing line: " + $line | Out-File -FilePath $logfile -Append

$result = "Request now working (probably the server is not running or there is no
internet access)"

$result = Invoke-WebRequest -Uri $Uri -Method Post -Form $Form

$result.StatusCode.ToString() + ", " + $result.StatusDescription + ", " + $result.Content
| Out-File -FilePath $logfile -Append

<# Replace previous line by the next ones in case want to show

custom activity in the terminal window instead of default output:

$Form

$result = Invoke-WebRequest -Uri $Uri -Method Post -Form $Form

$result.Content

#>

Start-Sleep -m 100

}
```