Institut de recherche
pour le développement

# OSMOSE

## Object-oriented Simulator of Marine ecOSystem Exploitation

---

## *Release Note*

---

**Version 3 Update 2**

Website: www.osmose-model.org

April 2015

## TABLE OF CONTENTS

# 1) Introduction

This new release is an update of Osmose 3 and, as such, does not lead to major changes in the parameters.

This document highlights the most important changes from Osmose 3 Update 1 to Osmose 3 Update 2, and lists all the changes, additions and deprecations.

# 2) Release history

This section provides of brief history of Osmose versions. Source code and default configuration files are all available from the subversion server.

## 2.1) Osmose 3 Update 2

The release improves the initialisation of the model in order to avoid oscillation (that may lead to premature species collapse) and make it lighter and faster. It also provides a new parametrisation for managing the input flux of incoming species more accurately. This release automatically updates the configuration file from previous versions (Osmose 3 and Osmose 3 Update 1). Osmose 3 Update 2 also comes will clear guidelines for the calibration procedure (separate document).

Source code

https://svn.mpl.ird.fr/osmose/code/branches/stable-3

Default configuration template

https://svn.mpl.ird.fr/osmose/code/branches/config/osm/osm_v3u2

## 2.2) Osmose 3 Update 1

This release updates and clarifies the mortality algorithms, hereafter called ITERATIVE and STOCHASTIC algorithms (see section below). Both stochastic and iterative algorithms now guarantee Finput=Foutput (so no need to back calculate fishing mortality from output as we used to do before). Fishing mortality and natural mortality accept any type of time variability (constant, seasonal, interannual, interannual + seasonal). The release introduces the option of parameterizing fishing by input catches (this could be useful if you would like to simulate fishing quotas for example). Effort for unifying the output format has been made (not finished though).

Source code

https://svn.mpl.ird.fr/osmose/code/tags/ov3u1

Default configuration template

https://svn.mpl.ird.fr/osmose/code/branches/config/osm/osm_v3u1

## 2.3) Osmose 3

Osmose 3, workshop March 2013, Sète, France

This is the first public release of Osmose 3. It is a stabilized and debugged version of Osmose that has been presented during the Osmose Workshop 2013.

It uses the ITERATIVE (previously called Case1) mortality algorithm by default (iterative algorithm presented by Ricardo at the 2013 workshop). Most input parameters can be in the form of time series so that the application works on interannual mode.

Source code

https://svn.mpl.ird.fr/osmose/code/tags/ov3

Default configuration template

https://svn.mpl.ird.fr/osmose/code/branches/config/osm/osm_v3

## 2.4) Osmose 2, 2013

This version gathers together all the work done from February 2013 (limit being the email sent for MEECE simulations early February 2013) until mid-year 2013.

Both mortality algorithm CASE1 and CASE3 are implemented (what will become ITERATIVE and STOCHASTIC in Osmose 3 Update 1), conversely to ov2_2012.

It fixes several bugs from ov2_2012 and improves input formats for fish spatial distribution maps.

Source code

https://svn.mpl.ird.fr/osmose/code/tags/ov2_2013

Default configuration template

https://svn.mpl.ird.fr/osmose/code/branches/config/osm/osm_v2_2013

## 2.5) Osmose 2, 2012

This version gathers together all the work done from mid-2011 (start of Philippe's contract) to beginning of 2013 (limit being the email sent for MEECE simulations early February 2013). It incorporates the new mortality algorithms but only CASE3 (what will become the stochastic mortality algorithm in Osmose 3 Update 1) seems to work properly.

As for the input files, it takes the old format (even though the habitat/area file accepts some CSV files). It is a big update from Osmose WS2009, with a lot of bug fixing and the first steps for improving the mortality algorithm, without major changes in the parametrization.

Source code

https://svn.mpl.ird.fr/osmose/code/tags/ov2_2012

Default configuration template

https://svn.mpl.ird.fr/osmose/code/branches/config/osm/osm_v2_2012

## 2.6) Osmose 2, 2009

Osmose 2, workshop May 2009, Cape Town, South Africa

This version gathers all the work done by Morgane, Yunne and collaborators during

Morgane PhD (2006-2009). Main additions concern the coupling of Osmose to the biogeochemical model ROMS-NPZD.

Source code

https://svn.mpl.ird.fr/osmose/code/tags/ws2009

Default configuration file

https://svn.mpl.ird.fr/osmose/code/branches/config/ben/afs_ws2009

### 2.7) Osmose 1

Yunne PhD and early career (Shin and Cury 2001, 2004). Not archived on subversion server.

## 3) Update configuration file

### 3.1) From Osmose 2 to Osmose 3

From Osmose 2, 2012 or Osmose 2, 2009, you need to update manually your configuration files to be compatible with Osmose 2, 2013. They are no major changes though.

From Osmose 2, 2013 to Osmose 3, there is a utility that will convert automatically your configuration file to the new Osmose 3 format. Open Osmose 3 Update 1 with Netbeans and expand package fr.ird.osmose.util. Open class ConfigurationConverter.java. There is two variables to set up:

- inputFile that provides the path to the Osmose 2, 2013 INPUT.txt
- outputPath that provides the path of the directory for the converted Osmose 3 configuration.

Right click on ConfigurationConverter.java from the “Projects” panel and click on “Run File”. The conversion starts and should last one or two seconds.

This converter has been written as a convenience for the developer to upgrade quickly several configurations and was not intended at first to be shared. Even though it should work without any problem for the end user, be aware that it is not a fancy tool and you must double check the new configuration files and look for any inconsistency.

### 3.2) From Osmose 3 to current release

Since Osmose 3 Update 1, the configuration manager automatically updates any configuration (no older than Osmose 3) to the current release. The update proceeds incrementally :

- detect the version of the configuration (check parameter 'osmose.version', if the parameter does not exist, Osmose assumes this is Osmose 3)
- loop and update from one version to the following one until the newest release

Any configuration file that is modified by the update is backed up, for instance osm_param-output.csv is copied as osm_param-out.csv.bakyyyyMMddHHmm (with yyyyMMddHHmm the time of the backup). Every changes is commented in the configuration file and detailed in the log of the simulation.

Please pay careful attention to the log after the update process as it will give valuable information about missing parameters or assumption made by Osmose in order to run the new version.

## 4) CSV input file separator

Many Osmose parameters are paths to CSV file, for instance:

*movement.map0.file*

*mortality.fishing.rate.byDt.byAge.file.sp#*

*reproduction.season.file.sp#*

In Osmose 3 and Osmose 3 Update 1 these CSV input files had to be semi-colon separated. Since Osmose 3 Update 2, CSV input file separators can be any of the following characters:

- equals =
- semi-colon ;
- coma ,
- colon :
- tab \t

Osmose will detect the separator automatically and independently for every CSV file. It means that one CSV input file may be coma separated and an other one may be tab separated, this is perfectly fine since Osmose 3 Update 2.

## 5) Decimal separator

Osmose is quite flexible in terms of separators for the configuration files (automatically detected among = , ; : \t ), the CSV output files (user-defined by parameter output.csv.separator) and the CSV input files (automatically detected among = , ; : \t ). On the contrary it restricts the decimal separator to dot, and only dot.

*Exemple given: 3.14159265 or 1.618*

Any other decimal separator (COMA for instance as in French locale) will be misunderstood and will unmistakably lead to errors.

One must be careful when editing CSV input files (either parameters or time series) with tools such as spreadsheets that may automatically replace decimal separator depending on the locale settings. Future Osmose release might allow the use of specific locale but for now remember that DOT is the only accepted decimal separator.

# 6) Population initialisation

## 6.1) Seeding approach

Until Osmose 3 Update 1, the main way of initialising the population consisted in providing target biomass for every species. Osmose would create an age-structured population following an exponential decay by estimating the total annual mortality from the fishing and natural mortality parameters. This initialisation method shows several drawbacks:

- it provides a fully structured population though no assumption should be made about this structure. Indeed we want the model to build up this structure given some basic laws at individual level.
- it often leaves the system in a highly unstable state and therefore leads to premature and artificial species collapses or explosion in the first year of the simulation.
- it slows down the simulation because the initial population contains a big number of schools (it creates nschool for every age class of the species, from eggs to old schools).

We must first acknowledge that there is no ideal solution for initialising Osmose but it should be done buy making as little assumptions as possible, keeping the spin-up time as short as possible and leave the model dynamics as untouched as possible.

Keeping this three points in mind, Osmose 3 Update 2 proposes a new "seeding" mechanism for initialising the population. It works the following way: the system starts from a pristine state, with no schools in the domain. For a few years (user-defined) Osmose will spawn some eggs for every species. As soon as the eggs reach sexual maturity, the reproduction process takes over. Osmose stops the seeding, unless the spawning stock biomass gets depleted. In that case Osmose resumes the seeding by releasing some eggs until there are again mature individuals in the system for carrying on the reproduction process. Osmose completely ceases the seeding when the simulation reaches the maximal number of year for seeding, defined in the configuration file.

By following this approach

- we do not make any assumption about the structure of the population
- we give full command to the individual based model for building up the structure of the population
- we keep the model light-weight as the population grows from scratch (the first years are the fastest to run, conversely to the target biomass approach)
- we minimize the amplitude of the population oscillations
- we make the spin-up period shorter

## 6.2) Seeding parameters

The new initialisation process is controlled by two parameters, the seeding biomass and the seeding duration.

*population.seeding.biomass.sp#*

This parameter is the SSB that Osmose guarantees during the initialisation process when there are no mature adults to ensure the reproduction process. The number of eggs to be released in the reproduction process are computed the following way:

neggs = sex_ratio * alpha * season * SSB

with SSB = sum(biomass of mature individuals) or population.seeding.biomass.sp# if sum(biomass of mature individuals) is equal to zero.

As a first estimate population.seeding.biomass.sp# can take the same value as the previous parameter population.initialisation.biomass.sp#. This parameter could be calibrated.

*population.seeding.year.max*

The number of years for running the seeding process. From year 0 to year seeding max, Osmose will guarantee that some eggs will be release even though there are no mature individuals in the system. From year seeding max to the end of the simulation, the seeding ceases completely. If the parameter does not exist, Osmose will set it by default to the lifespan of the longest lived species of the system.

### 6.3) Deprecated parameters

The seeding mechanism replaces the initialisation from biomass. As a consequence the following parameters are deprecated

*population.initialisation.biomass.sp#*

*population.initialisation.method*

Indeed we also deleted the initialisation from spectrum as it has never been used since Osmose 2 and therefore the only way to initialise the population is the seeding mechanism. One can still start the simulation from a NetCDF file though. Refer to the next section.

### 6.4) Initialisation from NetCDF file

The population can be initialised by providing a NetCDF file that contains a complete description of every single school of the population. This approach may be useful for defining initial condition for an inter-annual run, for instance. Refer to the user manual for details.

*population.initialization.file*

Be aware that initialisation from NetCDF file and seeding mechanism are independent one from an other and they may interfere if set up inconsistently. If one initialises the simulation from a NetCDF file, then the seeding process should be disabled (by setting seeding biomass or seeding maximal year to zero).

## 7) Migration

Osmose 3 Update 2 redefines how to input incoming flux of biomass in the system.

Since Osmose 2 2012, the incoming flux of biomass was defined by a biomass, an incoming age or length and a seasonality file. Since Osmose 3 Update 2, the user provides time series of  biomass by size/age class, in CSV files.

## 7.1) Incoming flux process

### 7.1.1) Deprecated parameters

- flux.incoming.season.file
- flux.incoming.season.file.sp#
- flux.incoming.biomass.sp#
- flux.incoming.size.sp#
- flux.incoming.age.sp#

### 7.1.2) New parameters

*flux.incoming.byDt.byAge.file.sp#*

ou

*flux.incoming.byDt.bySize.file.sp#*

Format of the CSV file

```
Time step / Age;0;2;3;4
0;0;500;800;0
1;0;500;800;0
2;0;400;700;0
3;0;400;700;0
...
```

The age classes (year) are automatically scanned by Osmose. In this case there are 4 classes: [0 2[, [2 3[, [3 4[ and [4 lifespan[. Osmose will sets the incoming age at the middle of the interval: 1 year, 2.5 year, 3.5 year, etc. The value of the time step does not matter, Osmose assumes there is one line per time step. The number of time steps in the CSV file must be a multiple of the number of time steps per year. If the time series is shorter than the duration of the simulation, Osmose will loop over it. If the time series is longer than the duration of the simulation, Osmose will ignore the exceeding steps.

In the above example, for the first time step, Osmose will input 500 tonnes of 2.5 year old school and 800 tonnes of 3.5 year school. The incoming biomass should be calibrated.

Size classes are handled the same way than age classes.

*simulation.nschool.sp#*

This parameter takes a slightly different meaning for the incoming flux process. It still controls the number of schools created during the reproduction process (which may occur independently of the incoming flux process, depending on your configuration parameters) but it also controls the number of schools created for each age/size class and time step.

The meanings are close enough so as not to worry about the value of this parameter and its order of magnitude depending on whether it controls reproduction, incoming flux or both.

## 7.2) Several processes to account for migration

In your simulated domain, some species might not fulfil a complete life cycle inside the domain but are too significant in terms of biomass or impact upon the other species to be ignored by the simulation. Such species will be considered as migrating species in Osmose and several mechanisms co-exists in order to depict a broad range of situation.

Careful parametrisation (and combination) of incoming flux process, reproduction process, movement process and lifespan should allow the user to represent any type of migration.

### 7.2.1) Incoming flux process

Some schools come from outside the simulated domain at a given age/size and time. Osmose does not control where and when these schools have been spawned neither what happened to them in previous stages. This process brings them in. What happens to these schools inside the simulated domain is then up to the movement process and reproduction process.

### 7.2.2) Temporarily out of the simulated zone

Osmose can relocate some schools outside the simulated domain, at given age and time. What happens outside the simulated domain is not explicitly represented, Osmose applies a user-defined total mortality rate and a species specific growth rate and the reproduction process is disabled. The movement parameters may be such that some schools never come back in the simulated domain. It is not advisable though because such schools would unnecessarily clutter the memory.

### 7.2.3) Permanent departure

Some species may leave the simulated domain at a given age and never come back. As mentioned in the previous point, this could be achieved in Osmose with the movement process by leaving them permanently out of the simulated domain, but it is not recommended for memory reason. Osmose does not offer yet a specific process to trigger the permanent departure of some schools at a given age/size. The only trick to render this behaviour would be to purposefully shorten the lifespan of the species. It works but it is not satisfactory because it corrupts the meaning of the lifespan parameter.

### 7.2.4) Reproduction process

The process controls which schools can spawn inside the simulated domain.

# 8) Reproduction process

Updated ReproductionProcess.java so that the seasonality files can only be provided by species.

Deprecated parameter:

*reproduction.season.file*

Valid parameter:

*reproduction.season.file.sp#*