MIKE ECO Lab

Short Scientific Description

DHI A/S headquarters
Agern Allé 5
DK-2970 Hørsholm
Denmark

+45 4516 9200 Telephone
mike@dhigroup.com
www.mikepoweredbydhi.com

Company Registration No.: DK36466871

PLEASE NOTE

CONTENTS

MIKE ECO Lab
Short Scientific Documentation

1 Introduction

MIKE ECO Lab functions as a module in the MIKE simulation software developed by DHI. It is a piece of numerical simulation software for Ecological Modelling. It is an open and generic tool for customising aquatic ecosystem models to simulate for instance water quality, eutrophication, heavy metals and ecology.
Open and generic refers in this case to the fact, that the underlying model equations are not concealed in hardcoded, inaccessible machine code but are visible to and changeable by the user. MIKE ECO Lab also supports agent- or individual-based modelling (ABM/IBM) in connection with selected hydrodynamic simulation engines.

The module is developed to describe processes and interactions between chemical and ecosystem or agent state variables. Also, the physical process of sedimentation or buoyancy transport of state variables can be described (moves a state variable vertically in the water column).

The module is coupled to the Advection-Dispersion Modules (AD) of the DHI hydrodynamic flow models, so that transport mechanisms based on advection-dispersion can be integrated in the MIKE ECO Lab simulation. Agent based modelling is analogue coupled to the Particle Transport Module (PT)[1].

The description of the ecosystem or agent state variables in MIKE ECO Lab is formulated as a set of ordinary coupled differential equations describing the rate of change for each state variable based on processes taking place in the ecosystem. All information about MIKE ECO Lab state variables, processes and their interaction are stored in a generic text-based file, so-called MIKE ECO Lab template.

The scientific descriptions of DHI supported MIKE ECO Lab templates, i.e. the documentation of specific MIKE ECO Lab differential equations and process equations is usually available as a PDF file linked to template in question and in the general DHI documentation. For DHI projects with tailor-made MIKE ECO Lab templates for specific projects, the scientific description of the used MIKE ECO Lab equations typically will be described in the project report.

1.1 Classic MIKE ECO version 1 / version 2.0

MIKE ECO Lab currently supports the classic MIKE ECO Lab kernel 1.0 and since MIKE release 2025 also MIKE ECO Lab version 2.0[1]. From a user perspective, both kernels are equivalent though new developments will focus on version 2.0 and beyond. The main difference is that version 2.0 is better suited for modern, multi-threaded computation.

[1] Currently only available in the MIKE FM series

2 What Is Behind MIKE ECO Lab?

MIKE ECO Lab is implemented as a sharable, dynamic loaded library/module (.so/.dll). The MIKE ECO Lab library is generic and utilised by a number of different DHI flow model systems and editors. All work with MIKE ECO Lab is centred around MIKE ECO Lab template[2] files that contain the model definition. The MIKE ECO Lab kernel library consists of an in-/output object to create, read and store MIKE ECO Lab templates, a parser/ interpreter that translates the equation expressions into lists of instructions that enables the computational engine of the library to evaluate the expressions in the template.

During simulation, the model system integrates one time step by simulating the transport of advective state variables or ABM particles based on hydrodynamics. Initial concentrations or updated AD concentrations, coefficients/constants and updated forcings are loaded into the ECO Lab module. Then the ECO Lab module evaluates all the expressions, integrates one time step, and returns updated variable values and particle movement vectors to the general flow model system that advances one time step. An illustration of the data flow is shown in Figure 2.1.

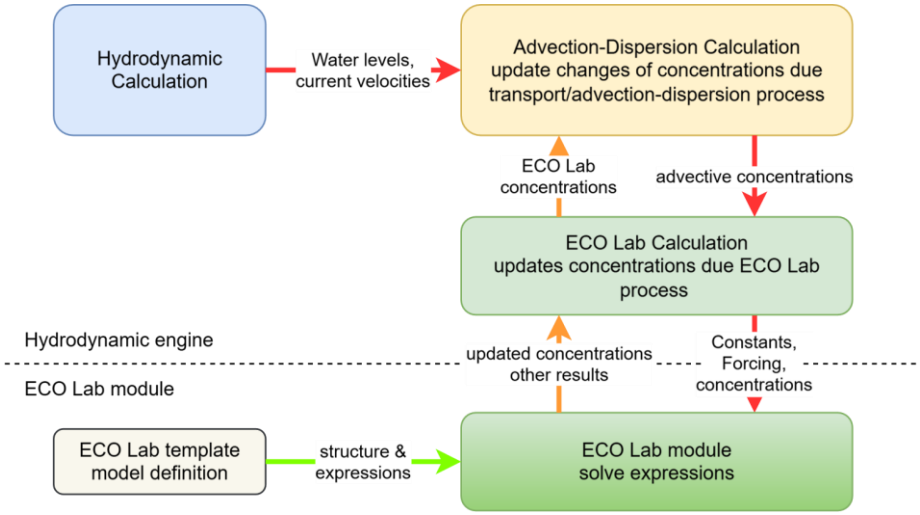

Figure 2.1 Data flow between the hydrodynamic flow model and MIKE ECO Lab.

[2] 1 A MIKE ECO Lab template contains the mathematical definition of a MIKE ECO Lab model. It contains information about the included state variables, constants, forcings, processes and the state variables' rate-of-change differential equations.

3 MIKE ECO Lab Set of Ordinary Differential Equations

In general, for each state variable an ordinary differential equation (ODE) is specified. The ODE summaries the processes involved for the specific state variable, i.e. it describes which and how process determine the rate of change of the state variable in time. It usually consist of several separate formulated process[3]. If a process affects more than one state variable (i.e. the same process is referred by several state variable equations), or the state variables affect each other, the differential equations are said to be coupled with each other.

$$ODE_C: \quad \frac{dC}{dt} = \sum_{i=1}^{n} process_i \tag{3.1}$$

ODE_C	ODE of MIKE ECO Lab state variable C
C:	MIKE ECO Lab state variable
n:	Number of processes involved for specific state variable

3.1 Example of ordinary MIKE ECO Lab differential equation:

Cyanide is assumed only to be affected by one temperature dependent decay process in this simple example;

$$\begin{aligned} \frac{dc_{cyanide}}{dt} &= -decay \\ decay &= K * \theta^{(temperature-20.0)} * C_{cyanide} \end{aligned} \tag{3.4}$$

$C_{cyanide}$	state variable representing the cyanide concentration
$\frac{dc_{cyanide}}{dt}$	ODE expression describing the change of the state variable due a process (changes due AD transport is calculated by the HD engine)
decay:	(named) process
K:	decay coefficient (per day)
Θ:	Arrhenius temperature coefficient

3.2 ECO Lab variable types

ECO Lab differentiates between a couple of semantic different variable types.

State variables represent unusually the “information of interest”. Their development over time is described by an ODE expression composed of one or more process.

Processes always describe the rate at which something changes. They are described by mathematical expressions using arguments such as numbers, constants, forcings and state variables or other calculated values (auxiliary expressions or other known process).

[3] In fact, besides process a state variable expression may also refer to other state variables, constants/forcing and mathematical expressions.

Constants and forcings represent numerical parameters. In this context, constants are values always constant in time, and forcings are values that can be varying in time[4].

Auxiliary expressions are mathematical expressions with the same type of arguments as process but may be represented in any unit. It is generally the template authors obligation to ensure that all units are correct and consistent throughout the template. There is no form of automatic unit conversion within the ECO Lab module[5].

Derived outputs can be formulated as additional mathematical expressions. They can represent additional information based on the values of other variables at the end of the current time step, i.e. after the numerical integration of all ODEs.

State variable expressions and process are always describing the change over time. The base time unit in ECO Lab is one day. Thus, their unit shall be specified as 'per day' in the template. For state variables expressed as concentration this equals to a change rate in "concentration/per day", or, expressed as mass, "mass/per day" etc.

In MIKE ECO Lab there are three kinds of processes: transformation and settling- or buoyancy processes. Transformation is a point description of a process not dependent on neighbouring points. Settling is a process transporting state variables to neighbouring points down the water column. Analogue, buoyancy process transport state variables to neighbouring points up the water column. The calculation of a state variable with a settling or buoyancy process is therefore dependent on information from vertically neighbouring points. Any lateral transport is handled by the AD transport in the hydrodynamic simulation engine. ABM state variables are "bound" to particles and only support transformation process. Vertical movement of the whole particle is controlled by a settling velocity expression. Horizontal particle movement is controlled by the sum of up to 6 horizontal movement expressions using a polar notation (i.e. defining direction and speed of movement).

Also, the light forcing needs special handling to calculate the light penetration in the water column. A special built-in function can be used for this purpose. MIKE ECO Lab can also handle that some processes only take place at specific positions in the water column. For instance, should reaeration (exchange with the atmosphere) only take place in the water surface. In other parts of the water column the reaeration is not active.

3.3 Special handling of settling and buoyancy process

The calculation of vertical movements needs information from the layer below or above in multi layered systems. In MIKE ECO Lab, it is possible to specify process with vertical movement: settling or buoyancy process. A settling process for example is transporting the state variable vertically towards the bottom. As for transformation processes, an expression must be specified describing the 'concentration change' from actual cell to cell below [mg/l/d]. When looking at a 'Settling' process directly in output from a MIKE ECO Lab simulation the output will show the result of the specified expression (the same as if it was a transformation process). The difference between a 'Settling' and a 'Transformation' process will appear in output of the affected state variable. This is because the numerical solution of a state variable affected by a 'Settling' process is different than if it was a 'Transformation' process. The definition of sign for a settling process is so that it should be specified as minus in the differential equation in order to transport the state variable correctly

[4] For the current calculation grid node, i.e. it is possible that a "constant" can vary between grid nodes but the value for a specific grid will not change during the simulation period.

[5] The DHI MIKE software utilises a unit management system known as EUM unit system, to automatically convert input- and output data; however, ECO Lab supports only a limited set of all EUM units and marks most expressions with a user- or undefined unit. There is never any internal, implicit unit conversion inside the ECO Lab module.

down the water column. The solution of a state variable with a settling process in a multi-layered system takes into account that a contribution to the state variable is received from the layer above (if not top/surface layer). Any variation in vertical discretization is also included in the numerical solution of differential equation involving a settling process. When solving a differential equation containing a settling process, MIKE ECO Lab substitutes the settling process expression in the differential equation with the following expression:

$$\frac{dc_n}{dt} = \frac{-settling_{n-1} * dz_{n-1} + settling_n * dz_n}{dz_n} \tag{3.2}$$

Where

settling$_{n-1}$is the user specified expression for 'rate of change' of the state variable concentration in layer n caused by a settling process transporting from layer n-1 to layer n [g·m-3·d-1]. It is usually a function of the concentration in layer n-1.

settling$_n$ is the user specified expression for 'rate of change' of the state variable concentration in layer n caused by a settling process transporting from layer n to layer n+1 [g·m-3·d-1]. It is usually a function of the concentration in layer n.

dz$_n$ is the thickness of layer *n* [m] and *dz*$_{n-1}$ is the thickness of layer *n*–1 [m].

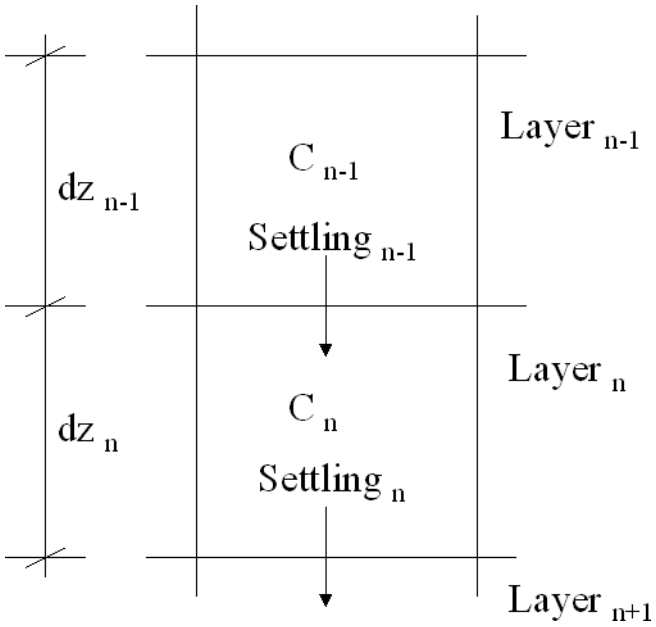

Figure 3.1 Schematic illustration of settling process

Buoyancy processes are handled analogue but "upwards".

3.4 Handling of site-specific processes

Some processes only take place in specific layers of the water column, and such processes are handled by calculating the process at the relevant layer where the process takes place and setting the process to zero in other layers. Examples of this could be a process such as re-aeration in the surface layer or accumulation at the bed layer.

3.5 Handling of built-in constants and forcings

Constants or forcings such as for instance temperature can be specified in different ways. They can be user specified, as constant values or as timeseries, map series, or volume series. As an alternative to use user specified values of constants and forcings, it is also possible to use built-in constants and forcing.
Built-in constants and forcings can be picked from a list in the ECO Lab template editor. These values are already estimated in the hydrodynamic model, and they can be used as arguments in MIKE ECO Lab expressions. During simulation the built-in forcings and constants will be updated with the calculations in the hydrodynamic simulation. Examples of built-in forcings are temperature, flow velocities, salinity, wind velocity.

Please note that even if defined as “built-in”, some constants or forcings may not be present in the applied hydrodynamic model. For example, a 1D or 2D hydrodynamic model cannot provide data for vertical flow information and temperature can only be provided if heat balance calculations are enabled. When the hydrodynamic model in question cannot provide appropriate data for a “built-in” constant/ forcing, it will be handled as a normal user defined value, i.e. it will require proper user input.

3.6 Particle classes in ABM simulations

When utilizing ABM simulations, an ABM particle is a “container” for all variables assigned to and moved with a particular particle. These variables can represent (Lagrange/ ABM) state variables, constants and arithmetical expressions. Particles with the same set of variables belong to the same particle class. Depending on the use, arithmetical expressions may be seen analogue to process or auxiliary expressions. A particle can have additional variables, so called restrict-area-search-functions or RASF, allowing them to “sense” their environment and expressions to characterise its movement. The actual movement is computed by the hydrodynamic engine using the movement vector it receives from MIKE ECO Lab.

3.7 Special handling of light penetration in MIKE ECO Lab

Light penetration in the water column can be solved with a Lambert Beer built-in function in MIKE ECO Lab. In multi-layered systems with vertical varying extinction coefficients, the Lambert Beer expression must be calculated for each layer, and therefore, the Lambert Beer expression as argument uses the result of the Lambert Beer expression in the layer above.

$$I_n = I_{n-1} * e^{\eta_n * dz_n} \tag{3.3}$$

where I_n is the light available for primary production in the actual layer *n*, I_{n-1} is the irradiance in the layer above, η_n is the extinction coefficient and dz_n the layer thickness.

The way MIKE ECO Lab handles this problem is by using a so-called built-in function that is special designed to handle this ‘Lambert Beer’ problem. The functions are called:

LAMBERT_BEER_1(surface radiation, layer height, light extinction coefficient).
The function returns the solar radiation in top of each layer of the water column.

LAMBERT_BEER_2(surface radiation, layer height, light extinction coefficient).
The function returns the solar radiation in bottom of each layer of the water column.

4 Integration With AD Engines

The dynamics of advective MIKE ECO Lab state variables can be expressed by a set of transport equations, which in non-conservative form can be written as:

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} + v\frac{\partial c}{\partial y} + w\frac{\partial c}{\partial z} = D_x\frac{\partial^2 c}{\partial x^2} + D_y\frac{\partial^2 c}{\partial y^2} + D_z\frac{\partial^2 c}{\partial z^2} + S_c + P_c \tag{4.1}$$

c:	The concentration of the MIKE ECO Lab state variable
u, v, w:	Flow velocity components
Dx, Dy, Dz:	Dispersion coefficients
Sc:	Sources and sinks
Pc:	MIKE ECO Lab processes

The state variables may be coupled linearly or non-linearly to each other through the MIKE ECO Lab source term P_c

The transport equation can be rewritten as

$$\frac{\partial c}{\partial t} = AD_c + P_c \tag{4.2}$$

where the term AD_c represents the rate of change in concentration due to advection, dispersion (including sources and sinks).

The MIKE ECO Lab numerical equation solver makes an explicit time-integration of the above transport equations, when calculating the concentrations to the next time step.

An approximate solution is obtained in MIKE ECO Lab by treating the advection-dispersion term as AD_c as constant in each time step.

The coupled set of ordinary differential equations defined in MIKE ECO Lab are solved by integrating the rate of change due to both the MIKE ECO Lab processes themselves and the advection-dispersion processes.

$$c(t+\Delta t) = \int_t^{t+\Delta t} (P_c(t) + AD_c) + \partial t \tag{4.3}$$

The advection-dispersion contribution is approximated by

$$AD_c = \frac{c^* + (t+\Delta t) - c^n(t)}{\Delta t} \tag{4.4}$$

where the intermediate concentration c* is found by transporting the MIKE ECO Lab state variable as a conservative substance over the time period Δt using the AD module.
The main advantage of this approach is that the explicit approach resolves coupling and non-linearity problems resulting from complex source MIKE ECO Lab terms P_c, and therefore the MIKE ECO Lab and the advection-dispersion part can be treated separately.
An implicit approach of solving the transport equations is not possible yet in MIKE ECO Lab.

5 Integration Methods

The following integration methods are generally available in MIKE ECO Lab: Euler, Runge Kutta 4. In the classic ECO Lab kernel a third method, Runge Kutta with quality check, is also available.

5.1 Euler integration method

A very simple numerical solution method for solving ordinary differential equations. The formula for the Euler method is:

$$y_{n+1} = y_n + h * f(x_n, y_n) \tag{5.1}$$

which advances a solution y from x_n to $x_{n+1} = x_n + h$

5.2 Runge Kutta 4th order

A classical numerical solution method for solving ordinary differential equations. It has usually higher accuracy than the Euler method but requires longer simulation times. The fourth order Runge-Kutta method requires four evaluations of the right-hand side per time step.

$$y_{n+1} = rk4(y_n, f(x_n, y_n), x_n, h) \tag{5.2}$$

The function is solved this way:

$$\begin{aligned} k_1 &= h * f(x_n, y_n) \\ k_2 &= h * f(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \\ k_3 &= h * f(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \\ k_4 &= h * f(x_n + h, y_n + k_3) \\ y_{n+1} &= y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} - O(h^5) \end{aligned} \tag{5.3}$$

which advances a solution y from x_n to $x_{n+1} = x_n + h$

5.3 Runge Kutta 5th order with quality check

A numerical solution method for solving ordinary differential equations. The accuracy is evaluated, and the time step is adjusted if results are not accurate enough.

This method is only available when using the classic ECO Lab 1.0 kernel.

$$y_{n+1} = f(y_n, f(x_n, y_n), x_n, h, \varepsilon, yscale) \tag{5.4}$$

The function is solved as described below.

First take two half steps:

$$\begin{aligned} h_2 &= 0.5 * h \\ x_{n+1/2} &= x_n + h_2 \\ y_2 &= rk4(y_n, f(y_n, x_n), x_n, h_2) \\ y_2 &= rk4\left(y_2, f(y_2, x_{n+1/2}), x_{n+1/2}, h_2\right) \end{aligned} \tag{5.5}$$

Compare with one full time step

$$y_1 = rk4(y_n, f(y_n, x_n), x_n, h) \tag{5.6}$$

Then estimate the error:

$$\begin{aligned} y_1 &= y_2 - y_1 \\ err &= \max\left(abs\left(\frac{y_1}{yscale}\right), \varepsilon\right) \end{aligned} \tag{5.7}$$

If the error is (err <= 1.0) the function returns

$$y_{n+1} = y_2 + \frac{y_1}{15} \tag{5.8}$$

Which advances a solution y from x_n to $x_{n+1}=x_n+h$ or else the time step is reduced, and the function tries again.