DOCUMENTATION: INTERFACES

Note: The asterisk indicates that the argument is optional.

- **Transient 1D reactive transport class**:

Subroutine	Arguments			Description
	Name	Kind	Description	
`read_time_discretisation`	`root`	String	Path of the file to be read	This subroutine reads the file ending in "_WMA_discr.dat" that contains the time discretization for the WMA: number of time steps, time step(s) and method integration of chemical reactions.
`write_RT_1D`	`root`	String	Path of the file where the results are to be written	This subroutine writes the data and results in the final time of all variables to the file ending in ".out".
	`path_py*`	String	Path of the directory where the results needed to use them in Python are to be written	
`set_transport`	`transport_obj`	Transient 1D transport class	Transient 1D transport class object.	Assigns the `transport` attribute of this class.
`set_chemistry`	`chemistry_obj`	Chemistry class	Chemistry class object	Assigns the `chemistry` attribute of this class.

- **Transient 1D transport class**:

read_transport_data_WMA	root	String	Root of the file to be read	This subroutine reads the file ending in "_WMA_lambdas.dat" that contains the porosity, mixing ratios, and water indices corresponding to the mixing ratios.

- **Chemistry Class**:

read_chemistry	root	String	contains the root of the data entry files of the problem to be solved	This subroutine reads all the chemical information. It first calls the chemical system's reading subroutine via its attribute of the chemical system class, then reads the types of initial waters, solids, and gases, and finally reads the target waters association. In addition, it assigns objects in the *Reactive Zone* class belonging to the *Chemistry* class.
	path_DB	String	contains the directory of chemical databases	
solve_reactive_mixing_lump	root	String	Root for output file	This is the main reactive mixing solver. Performs the reactive mixing calculations with the WMA for a given time interval. At the local level, the calculations are made by the *Aqueous Chemistry* class.
	mixing_ratios	Real Array Class	Mixing ratios of concentrations	
	mixing_waters_indices	Integer Array Class	Mixing water indices corresponding to mixing ratios	
	time_discr	Time discretization class	Time discretization used to solve transport	
	int_method_chem_reacts	Integer	Temporal integration method for chemical reactions. 1: Euler Explicit 2: Euler Fully Implicit	
solve_reactive_mixing_cons	root	String	Root for output file	
	mixing_ratios_conc	Real Array Class	Mixing ratios of concentrations	
	mixing_ratios_Rk_init	Real Array Class	Initial mixing ratios of reaction amounts	

	mixing_waters_indices	Integer Array Class	Mixing water indices corresponding to mixing ratios	
	time_discr	Time discretization class	Time discretization used to solve transport	
	int_method_chem_reacts	Integer	Temporal integration method for chemical reactions. 1: Euler Explicit 2: Euler Fully Implicit	
	mixing_ratios_Rk	Real Array Class	Final mixing ratios of reaction amounts	
interfaz_comps_vars	u_tilde	Real matrix	Component concentrations after mixing iteration	This is an interface to solve a reactive mixing iteration using **variables** as **arguments**. The client must solve transport beforehand, mix the waters and then call this subroutine, at every time step.
	Delta_t	Real scalar	Time step	
	u_new	Real matrix	Component concentrations after reactive mixing iteration	
interfaz_comps_arch	num_comps	Integer	Nº components	This is an interface to solve a reactive mixing iteration using **files** instead of variables. The client must solve transport beforehand and write the mixed component concentrations in a file. Then, he or she must call this subroutine.
	file_in	String	File with concentrations of components after mixing iteration	
	Delta_t	Real scalar	Time step	
	file_out	String	File where concentrations of components after reactive mixing iteration will be written	

Sequence in the main file

1) Define path names: problem to be solved and databases
2) Choose problem to solve. If it is not among the options, it must be added.
3) Initialize transport:
    a. Transient 1D Reactive Transport Class Object `my_RT_trans` calls subroutine *`set_transport`* using Transient 1D Transport Class Object `my_tpt_trans` as argument
    b. `my_RT_trans` calls subroutine *`read_time_discretisation`*
    c. `transport` attribute in `my_RT_trans` calls subroutine *`read_transport_data_WMA`*
4) Initialize chemistry:
    a. `my_chem` calls `read_chemistry`
5) Solve reactive mixing and assign chemistry class object:
    a. `my_chem` calls global reactive mixing solver, which can be one of the following subroutines depending on the activity coefficients model (ideal or not) and the WMA variant (lumped/consistent):
        i. `solve_reactive_mixing_ideal_lump`
        ii. `solve_reactive_mixing_ideal_cons`
        iii. `solve_reactive_mixing_lump`
        iv. `solve_reactive_mixing_cons`
    b. `my_RT_trans` calls *`set_chemistry`* using `my_chem` as argument
6) Write data and results:
    a. `my_RT_trans` calls *`write_RT_1D`*