Quick Reference Guide WAQUA

Rijkswaterstaat
Ministerie van Infrastructuur en Waterstaat

Quick Reference Guide WAQUA

Version number : Version 10.66, June 2019
Maintenance : see `www.helpdeskwater.nl/waqua`

Log-sheet

Table 1: Log-Sheet

document versie	datum	change-nr	wijzigingen ten opzichte van de vorige versie
10.46	10-05-2007	M308193	Verbetering m.b.t. optie windid-file Waqwnd
10.47	29-11-2007	C71236	Verwijdering Waqbhd
10.48	09-01-2008	C77580	Aanpassingen DDVERT voor Waqpre, hostmap voor Waqpro
10.49	04-02-2008	C77580	Aanpassingen m.b.t. fases gekoppelde runs Waqpro
10.50	14-02-2008	C77580	Toevoeging automatische partitie van subdomeinen voor domein decompositie
10.51	22-08-2008	M353161	Fixed broken reference
10.52	28-11-2008	C84610	Toevoeging opties m.b.t. Waqua-met-Costa systeem
10.53	05-12-2008	C85905	Toevoeging optie **-xtra_comm**
10.54	20-05-2010	C3210	NETCDF support in waqwnd; hoofdstuk waqpos verwijderd
10.55	10-09-2010	C3410	Verwijdering Waqpan
10.56	27-09-2010	C3384	Hostmap mappings Packed en Compact, keep_subdomsds toegevoegd en conversie naar LaTeX
10.57	11-10-2010	C3398	Nieuwe opties use_pinning en use_pbs_aware beschreven; bestaande opties precision en local_only beschreven, leesbaarheid run data parallel rekenen en domein decompositie verbeterd.
10.58	11-11-2010	C3452	Default convert2stress van waqwnd cf implementatie
10.59	14-03-2011	C3535	Waqua-met-Costa wordt Waqua through OpenDA
10.60	21-06-2011	C3395	Default partitioning is Automatic
10.61	22-02-2012	M3705	Default voor precision is nu double op linux64
10.62	15-10-2013	3964	Aanpassen OpenDA documentatie
10.63	12-05-2015	3940	Toevoeging beschrijving use_pinning voor Intel MPI
10.64	21-05-2015	4292	Defaults voor precision en pinning aangepast
10.65	07-10-2015	4311	Time zone read from NetCDF
10.66	19-06-2019	4767	Added time_unit to waqwnd

Contents

Chapter 1

About the quick reference guide

This part is not yet available

Chapter 2

General

2.1 Introduction

This Quick Reference Guide contains descriptions of several scripts to run WAQUA subsystems. This chapter gives some general remarks.

2.2 Background runs and shells

The behaviour of a script depends on the shell being used. The descriptions of the format of the scripts in this Quick Reference Guide are all based on the use of the Bourne-shell. If the Korn-shell or the C-shell is used, there are some problems in relation to the use of the background options. There are four ways to run a program in the background:

- by using *-back y* in the options list;
- by answering affirmative when (in an interactive run) the run procedure asks whether the program must be started in the background or not;
- by using *&* in the options list;
- by using a user script containing the options list and &.

The third possibility is not recommended when using the Korn-shell or the C-shell. In this case it is necessary to specify a complete options list. If one parameter is not given or is not correct, the run procedure will stop and wait for tty-input. Even though & is specified, the run procedure will always produce screen output. When using the C-shell or the Korn-shell, the procedure will not be able to suppress screen output in case of a background run.

When running a SIMONA program in the background, the user can safely log out, provided the run has been started as specified above. The program will keep on running.

Warning: Foreground processes put to the background by suspending them and giving a ' bg' command, will be cancelled at logout.

In case of an interactive run, a user script or the usage of the Bourne- shell, none of these problems will appear.

Chapter 3

Wind files conversion program WAQWND

3.1 Function

purpose	The subsystem converts the information from KNMI ASCII-files, binary GRIB-files, binary NETCDF-files, TIMESERIES file or the so-called Space Varying Wind and Pressure (SVWP) file, delivered by the KNMI, to the standard SIMONA format and stores this information on the 'wind' SDS file. Some additional information that is not present on the binary SVWP file (eg. grid size) is obtained from the user's input (via the perl procedure parameters) and stored on the 'wind' SDS file in order to complete the data. The information from the 'wind' SDS file is used by the WAQUA pre-processor and processor.

3.2 Files

data flow	**Input files**	**Output files**
	- binary SVWP-file,	SIMONA data storage file
	- timeseries file	message print file
	- 1 or more ASCII/GRIB/NETCDF-file(s)	
file names	**Logical name**	**System name**
	SIMONA data storage file	SDS-<runid>
	message file	waqwnd-m.<runid>
file description	**Logical name**	**Purpose**
	- binary SVWP file, - ASCII-files, - a TIMESERIES-file, - GRIB-files - a NETCDF-file	Storage or description of Space Varying Wind and Pressure data delivered by the KNMI or made by user.
	SIMONA data-storage file	storage of permanent data in SIMONA
	message file	Contains error messages and diagnostics
test input	Refer to the User's Guide WAQWND.	

3.3 Run data

Format for Binary SVWP file

*waqwnd.pl [-inpfmt <inpfmt>] [-runid <runid>] [-exp <exp>] *
*[-KNMI <KNMI-file>][-stress_SVWP <Y/N>] *
*[-date_ref "<day> <month> <year>"] [-timzon <timzon>] *
*[-convert2stress <Y/N>] [-wnd2strfile <wnd2strfile>] *
*[-mmw <mmw>] [-nmw <nmw>] [-angle <angle>] *
*[-grid <P/S>] [-gridmode <gridmode>] *
*[-overwrite <overwrite>] [-windidfile <wind ident file>] *
[-coordsystem <coordsystem>] [-back <back>]

Format for ASCII and GRIB-file(s)

*waqwnd.pl [-inpfmt <inpfmt>] [-runid <runid>] [-exp <exp>] *
*[-KNMI <KNMI-file>] [-date_fc <date_fc>] *
*[-time <time>] [-timzon <timzon>] [-convert2stress <Y/N>] *
*[-wnd2strfile <wnd2strfile>] [-pgem <pgem>] [-stress <stress>] *
*[-overwrite <overwrite>] [-windidfile <wind ident file>] *
*[-coordsystem <coordsystem>] [-gridflag <gridflag>] *
[-date_unit <unit>] \[-back <back>]

Format for a NETCDF-file

*waqwnd.pl [-inpfmt <inpfmt>] [-runid <runid>] *
*[-KNMI <KNMI-file>] [-date_ref <date_ref>] *
*[-convert2stress <Y/N>] *
*[-wnd2strfile <wnd2strfile>] [-stress <stress>] *
[-overwrite <overwrite>] [-back <back>]

Format for a TIMESERIES file

*waqwnd.pl [-inpfmt <inpfmt>] [-runid <runid>] [-exp <exp>] *
*[-KNMI <KNMI-file>] [-timzon <timzon>] *
*[-wnd2strfile <wnd2strfile>] [-overwrite <overwrite>] *
[-windidfile <wind ident file>] [-back <back>]

parameters	**parameter**	**signification and value**
	-angle	angle of the Y-axis with the North-direction; default: 0.0
	-back	y(es): program will be started in the background n(o): program will be started in the foreground
	-convert2stress	Must waqwnd convert windspeeds from the input files into stresses (Y/N); default: N
	-coordsystem	String with extra information on the coordinate system: USER: User defined (default); grid-type: Undefined INDEX: Model coordinates (M and N); grid-type: Undefined

	RDV: "Rijksdriehoeksstelsel-verschoven"; grid-type: Planar ED50: European Datum 1950; grid-type: Spherical WGS84: World Geodetic System 1984; grid-type: Spherical UTM31: Universal Transverse Mercator zone 31; grid-type: Planar UTM32: Universal Transverse Mercator zone 32; grid-type: Planar
-date_fc	Forecast date in format <*yyyymmdd*>; default: "current date". Input with reference time after "date_fc+time" is ignored.
-date_ref	reference date according to the WAQUA conventions (i.e. given in the form *dd mmm yyyy*, e.g. *15 SEP 1993*) The format is <yyyymmdd>; default: "current date" if input format is NETCDF.
-date_unit	Forecast unit; default: "h".
-exp	name of the experiment (simulation)
-inpfmt	Format of input file. Allowed values are SVWP (binary KNMI files), ASCII, GRIB, NETCDF or TIMESERIES
-grid	specification of grid type (either planar(P) or spherical(S))
-gridflag	Grid flag with values 1, 2 or 3 - 1: non-staggered grid - 2: auto detect (default) - 3: staggered grid
-gridmode	specification of grid mode (either staggered(S) or non-staggered(N))
-KNMI	- name of the binary space varying wind and pressure (SVWP) file delivered by KNMI; or - name of the TIMESERIES input file - name(s) of the ASCII/GRIB/NETCDF-file(s) with wind and pressure data (wildcard may be used) It can contain an explicit path name; the use of any indication of a parent directory ('..') is allowed.
-mmw	number of points in the rectangular wind grid in the M-direction
-nmw	number of points in the rectangular wind grid in the N-direction

-overwrite	an already existing experiment on the wind SDS file may be overwritten only if overwrite = yes is specified
-pgem	mean pressure in Pascal; (default: 101200 Pascal)
-runid	code to identify the output files of a run (eg. t02)
-stress	Indicates type of input to be used: -1 = autodetect (default) 0 = use wind speeds 1 = use wind stresses 2 = use cumm. wind stresses 3 = use cumm. wind stresses, times in between 4 = external forces
-stress_SVWP	Flag, indicating whether the KNMI file contains wind speeds (-stress_SVWP N) or wind stresses (-stress_SVWP Y); default: N
-time	time of start forecast in format <*hhmm*>; default: "current time" Input with reference time after "date_fc+time" is ignored.
-timzon	timezone GMT, MET or UNKNOWN; default: UNKNOWN
-windidfile	file containing text lines to identify the wind field (up to 3 lines allowed, maximal text length in each line: 80 characters); it can contain an explicit path name; the use of any indication of a parent directory ('..') is allowed.
-wnd2strfile	Name of input file for wind speed to stress conversion. Is only needed if -convert2stress== Y. if "-" is given default values will be used for the conversion.

- In case of a background run (with &) runid, exp, KNMI, date, mmw, nmw and grid are obligatory parameters and back has no effect.
- In case of an interactive start of the run the procedure will prompt for the parameters mentioned above (except those already given in the run call) including back. All parameters are checked before the program is started.
- If an executable of the program (WAQWND.EXE) is present in the current directory, this executable will be used to run the program.

notes

Each set of space varying space and pressure data on the 'wind' SDS file is identified by means of an experiment name. More than one experiment on the 'wind' SDS file is allowed, unless the input format is NetCDF.

All the information required by the program WAQWND is passed to it by means of the run procedure.

A wind SDS file is different from a WAQUA SDS file created by WAQPRE and WAQPRO. It contains information which is additional to the information on a WAQUA SDS file.

If the wind grid must be rotated the value of angle should be nonzero. The wind grid will be then be rotated, counterclockwise when angle is positive and clockwise when angle is negative. However, the wind vectors are not rotated; they should be given - in the KNMI-file - , with reference to the coordinate system, in the definitive positions of the nodal points as if those were already rotated.

If planar grid is specified, the grid coordinates are expected to be given in meters.
If spherical grid is specified, the grid coordinates are expected to be given in degrees of longitude and latitude.

Chapter 4

Preprocessor WAQPRE

4.1 Function

purpose

The WAQUA preprocessor WAQPRE checks the user input, prepares data for the simulation program WAQPRO (and produces a report).

The run-procedure waqpre.pl is used for standard (sequential and parallel) runs as well as for domain decomposition with vertical and horizontal refinement. The extensions of waqpre.pl for the latter two cases are described separately in paragraph 4.4.

4.2 Files

data flow	**Input files**	**Output files**
	WAQPRE input file	Message print file
	(SIMONA data storage file)	SIMONA data storage file
file names	**logical name**	**system name**
	WAQPRE input file	user defined
	SIMONA data storage file	SDS-<runid>
	Message file	waqpre-m.<runid>

The WAQPRE input file name (including file names used in this file) can contain an explicit path name. The use of any indication of a parent directory ('..') is allowed.

file description	**logical name**	**purpose**
	WAQPRE input file	Contains input in a structure suited for the WAQPRE program.
	SIMONA data storage file	Storage of permanent data in SIMONA.

	Message file	Contains error messages and diagnostics.

user's guide See 'User's Guide pre-processor WAQPRE'.
test input Not yet available.
exceptions **curvilinear computation**

- Limitation on the position of open boundaries: In case of curvilinear computation, all open boundaries should be positioned within the grid which is generated by the orthogonal grid generator. This grid is defined in the depth locations. It means that a water level boundary at for instance m = 1, is situated outside the transformed grid and not allowed. A u-velocity boundary at m = 1 on the other hand is allowed, because u-velocity points are situated between the depth points. The same restrictions apply to the other edges of a transformed grid or subgrid.

4.3 Run data

format standard run
*waqpre.pl [-runid <runid>] [-back <back>] [-input <input>] *
[-bufsize <bufsize>] [-ndom 1] [&]

format horizontal refinement
*waqpre.pl [-runid <runid>] [-back <back>] [-input <input>] *
*[-decomp <areas>] [-npart <npart>] [-partit <partit>] *
[-bufsize <bufsize>] [-buf_prt <bufsize>] [&]

format vertical refinement
*waqpre.pl [-runid <runid>] [-back <back>] [-input <input>] *
*[-ndom <ndom>] [-decomp <decomp>] [-kmax <kmaxes>] *
*[-npart <nparts>] [-partit <partits>] [-bufsize <bufsize>] *
[-buf_prt <bufsize>] [&]

format horizontal & vertical refinement
*waqpre.pl [-runid <runid>] [-back <back>] [-input <input>] *
*[-ndom <ndom>] [-decomp <decomp>] [-kmax <kmaxes>] *
*[-npart <nparts>] [-partit <partits>] [-bufsize <bufsize>] *
[-buf_prt <bufsize>] [&]

parameters	**parameter**	**significance and value**
	-runid	Code to identify the output files of a run.

-bufsize	The size of SIMONA blank common in million words, i.e. the space needed to store all SIMONA arrays; default: depending on the total size of the siminp and the include files.
-back	y(es): program will be started in the background n(o): program will be started in the foreground
-input	File name of WAQPRE input file
-ndom	Number of subdomains in case of a domain decomposition run with vertical refinement. The value should then be equal to the number of subdomains specified in the decomposition (decomp). The value 1 is used to specify that a standard (sequential or parallel) run is intended.
-decomp	File name of decomposition of the global domain into subdomains for domain decomposition. Also called areas-file.
-kmax	List of the number of layers per subdomain in case of domain decomposition with vertical refinement. This list should contain ndom entries, separated by commas. The number of layers of the last subdomain must be the maximum over all subdomains.
-npart	Number of parts for automatic partitioning of subdomains. In case of a DDHOR run *decomp* should contain one subdomain and *npart* should be one value. For DDVERT, a separate value *npart* is required for each subdomain in the decomposition.
-partit	Automatic partitioning method to be used per subdomain. Possible values are "strip", "orb", "strip_row", "orb_row", "strip_col" and "orb_col", "automatic" and "-" for no partitioning. A comma-separated list of *ndom* values is required for DDVERT, a single *partit* is used for DDHOR.
-buf_prt	The size of SIMONA blank common in million words for the auxiliary program COPPRE for domain decomposition. Default: 10.

notes

- The type of run is firstly determined through the occurrence of the reserved word %KMAX% in the WAQPRE input file (see paragraph 4.4). If this word is found the run must be using vertical refinement, else it is a standard run or using horizontal refinement.
- The distinction between a standard run or a run with horizontal refinement is made via the options decomp (signals horizontal refinement) and ndom (1=standard, >1 horizontal refinement), when given. If both are unspecified then in a background run (with &) a standard run is assumed, whereas in an interactive run the procedure asks which type of run is intended.
- In case of a background run (with &) runid and input are obligatory parameters and back has no effect. In a run with vertical refinement also the parameters ndom, kmax and decomp are obligatory. In a run with horizontal refinement the parameter decomp is required.
- In case of an interactive start of the run the procedure will prompt for the parameters mentioned above (except those already given in the run call) including back. All parameters are checked before the program is started.
- In contrary to previous versions (Simona2005-02 and older), no new executable is created if a non-default buffer size is requested.

4.4 Extensions for domain decomposition with horizontal and vertical refinement

overview

The run-procedure waqpre.pl is also used for the pre-processing phase of domain decomposition runs with horizontal and vertical refinement.

vertical refinement

The strategy towards vertical refinement is that the user prepares a WAQPRE input file for a single grid, but with all information that is related to the layer distribution concentrated in include-files. Next a decomposition of the domain is specified. Finally different versions of the include-files are delivered for all subdomains.

The run-procedure then executes WAQPRE repeatedly for each of the subdomains, each time with the appropriate include-files filled in. The program COPPRE is used to extract the data for the subdomains from the SIMONA data storage file for the complete domain with the given layer distribution. COPPRE is also used for splitting subdomains into multiple parts for parallel computation.

include files

The number of layers per subdomain and the include files for information that varies between different subdomains are entered into the WAQPRE input file via the reserved strings %KMAX% and %DOM%.

So, for example, layer thicknesses may be specified as:

VERTICAL
INCLUDE FILE="layerfile.%DOM%" In this case the files layerfile.1, layerfile.2, are used for subdomains 1 and 2, respectively.

Also weir-definitions must be treated in this way because they are allowed in TRIWAQ in runs with a single layer only.

horizontal refinement

The strategy towards horizontal refinement is that the user prepares multiple WAQPRE input files with different grids that can be matched on certain grid lines ("interfaces"). In order to combine different grids/domains different sections of each grid may be marked as inactive, these areas must then be filled in by one of the other domains. The active part of the grid is specified through a decomposition- or areas-file. Such an areas-file may be created with the help of IPW.

The areas-file may also prescribe a splitting of the active part of the grid into multiple subdomains for parallel computing. When the areas-file uses only a single subdomain, this may be partitioned automatically using partitioner COPPRE.

The run-procedure waqpre.pl is then executed separately once for each WAQPRE input file and accompanying areas file. The program WAQPRE creates a SIMONA data storage file for the complete domain, and the program COPPRE is then used to extract the data for all active subdomains.

automatic partitioning of subdomains

The run-procedure waqpre.pl provides options for automatic partitioning of subdomains given in a decomposition input file. The user then has to define the relevant information for domain decomposition only, and the system takes care of partitioning for parallel computing.

For domain decomposition with horizontal refinement this is restricted to decomposition input files in which only a single subdomain is described. The user should add the option "-npart <np>" to the run-procedure call, with <np> the number of parts to be used for automatic partitioning of subdomain 1. The option -partit is available for selecting the method for the partitioning, which defaults to Automatic.

For domain decomposition with vertical refinement the decomposition input file describes multiple (ndom) subdomains with different layer distributions. In this case a comma-separated list of ndom values must be provided, e.g. "-npart 1,2,6". The partitioning method should also be given per subdomain, in a comma-separated list. Here "-" may be used for subdomains that should not be split. An example argument is "-partit -,orb,orb", which is actually the default.

data flow	**Additional input files**	**Additional output files**
	decomposition input file	decomposition report file
		SIMONA data storage file per subdomain
file names	**logical name**	**system name**
	Decomposition input file	user defined
	SIMONA data storage file per subdomain	SDS-<runid>-<dom>, with <dom> the three-digit subdomain number
	Decomposition report file	coppre-r.<runid>
file description	**logical name**	**purpose**
	Decomposition input file	Gives the decomposition of the global grid of the WAQPRE input file into subdomains.
	SIMONA data storage file	Storage of permanent data in SIMONA.
	Decomposition report file	Contains an overview of the decomposition used.

user's guide	See 'User's Guide for Parallel WAQUA/TRIWAQ and Domain Decomposition'.
test input	Not yet available.

exceptions

Additional restrictions for vertical refinement

- Interfaces between subdomains must be at least three grid points away from each other, from any open boundary and from barriers.
- The subdomain with the highest number must also have the maximum number of layers.
- Layer interfaces of a coarse domain must continue into neighbouring finer domains; only layer interfaces from a finer domain may stop at the interface with a coarser domain.
- Fixed layers (with thickness given in meters) in one subdomain must connect to one or more fixed layers in neighbouring subdomains. Layers with variable thickness (thickness given in percentages) in one subdomain must connect to one or more layers with variable thickness in neighbouring subdomains. The only exception is that a subdomain with one (variable) layer may connect to a subdomain with both fixed and variable layer thicknesses.
- If a subdomain with only one layer connects to a subdomain with both fixed and variable thickness layers, the velocity and transport checkpoints in the subdomain with one layer must also be water level checkpoints. This is necessary for the interpolations done by the collector program COPPOS.
- The number of layers in a fine subdomain that connect to one layer in a neighbouring coarse domain must be at most four. If this number is higher, the functionality should still work, but numerical artefacts may become serious.
- The use of weirs exactly on subdomain interfaces is not advised, especially because in future versions the interfaces may be assigned to the subdomain with the highest number of layers.

Additional restrictions for horizontal refinement

- Interfaces between subdomains must stay away from each other, from any open boundary and from barriers by at least 3.5 grid spaces w.r.t. the coarsest subdomain involved.
- Openings and line-barriers and all information needed for dynamic barrier steering may not be partially in an active subdomain and partly in the inactive part of the domain.

notes for vertical refinement

- The value of ndom must match the number of subdomains specified in the decomposition input file; subdomains must be numbered consecutively.
- In domain decomposition runs, WAQPRE.EXE and COPPRE.EXE are run separately for each distinct value of %KMAX% (when %DOM% is not used) or for each subdomain (when %DOM% is used). In each run, all occurrences of the string "%KMAX%" in the WAQPRE input file are replaced by the corresponding value of kmaxes, and all occurrences of the string "%DOM%" are replaced by the subdomain number.
- Note that the number of times that WAQPRE.EXE is run is less, i.e. the preprocessing phase is faster, when %DOM% is not used in the input-file.

notes for horizontal & vertical refinement

- It is not allowed that a (sub)domain is finer horizontally but coarser vertically than a neighbouring (sub)domain.

Chapter 5

GEN_WAQPRO.PL

5.1 Function

purpose	GEN_WAQPRO.PL is used in combination with the user routine WASUST (refer to the User's Guide WAQPRO: 'Subroutine WASUST').	
	The utility GEN_WAQPRO.PL generates a new executable file WAQPRO_SINGLE.EXE, or WAQPRO_DOUBLE.EXE. This executable is called by the WAQPRO run procedure WAQPRO.PL (refer to chapter 6).	
data flow	Input files	Output files
	source file(s) of WASUST	WAQPRO_SINGLE.EXE or WAQPRO_DOUBLE.EXE
format	*gen_waqpro.pl file1 [file2 [...]] [&]*	
note	The current directory (.) must be added to PATH before $SIMONADIR/bin.	

Chapter 6

Processor WAQPRO

6.1 Function

purpose

The WAQUA processor WAQPRO computes water levels and currents as well as the concentration and dispersion of constituents resulting from the time varying effects of tide level, wind, discharges, concentrations, movable barriers and weirs, using either a rectilinear or a curvilinear grid. The assimilation of observed data may be used to improve model predictions. For this WAQPRO contains different Kalman filtering techniques. In the original implementation of this, the Kalman filter is implemented as an extension of WAQPRO, and the configuration of the filter is provided to the pre-processor WAQPRE and stored on the SDS-file. A new implementation is under development where the open source system OpenDA is used for the implementation of Kalman filtering. In this case part of the configuration is stored in the SDS-file whereas other parameters are specified on the command line for WAQPRO.
The system allows for employing multiple computers or processors in a single run (parallel computing). In this case the input data is distributed automatically, multiple WAQPRO processes are started which cooperatively perform the simulation, and then the output data is gathered together.
The system further allows for using domain decomposition with vertical refinement, i.e. for using different numbers of layers in different TRIWAQ subdomains. In this case the input data is given separately for the subdomains. Again multiple WAQPRO processes are started. Finally the output data per subdomain are converted to the maximum vertical resolution used and gathered together in a single output file.

Finally the system also allows for using domain decomposition with horizontal refinement, in which different global domains/grids are computed simultaneously. In this case input data must be given separately for one or more subdomains per global domain and a configuration file must be given that describes the global domains to be simulated. Again multiple WAQPRO processes are started. In this case the output data are gathered in a single output file per global domain.

6.2 Files for standard (single processor, sequential) runs

data flow	**Input files**	**Output files**
	SIMONA data storage file	SIMONA data storage file
	Debug settings file (optional)	Report file
		Message file
file names	**logical name**	**system name**
	SIMONA data storage file	SDS-<runid>
	Report file	waqpro-r.<runid>
	Message file	waqpro-m.<runid>
file description	**logical name**	**purpose**
	SIMONA data storage file	Storage of permanent data in SIMONA.
	Report file	User controlled print with initial array data, history prints and computed values throughout the grid.
	Message file	Contains error messages and diagnostics.

user's guide	See 'User's Guide processor WAQPRO'.
test input	Not yet available.

6.3 Run data for standard (single processor, sequential) runs

format

*waqpro.pl [-runid <runid>] [-npart 1] [-ndom 1] *
*[-bufsize <bufsize>] [-back <back>] *
[-precision <single/double>] [-nmdbg <nmdbg>] [&]

parameters	**parameter**	**meaning and values**
	-runid	Code to identify the output files of a run.
	-npart	Switch to activate the parallel functionality (=1: not parallel, >1: parallel).

-ndom	Switch to activate the domain decomposition functionality (=1: no domain decomposition, >1: using vertical refinement).
-bufsize	The size of SIMONA blank common in million words, i.e. the space needed to store all SIMONA arrays. The default value is 10.
-back	y(es): program will be started in the background n(o): program will be started in the foreground
-nmdbg	The name of the file with debug settings. The input syntax is explained below.
-precision	Switch between single and double precision for the computational core. Possible values: single or double. Default: double.

notes

- For execution of a standard (single processor, sequential) run, npart and ndom must be 1, and the file SDS-<runid>-000 should not exist.
- In case of a background run (with &) runid and npart are obligatory parameters, and back has no effect.
- In case of an interactive start of the run the procedure will prompt for the first five parameters mentioned above. These parameters are checked before the program is started.
- In contrary to previous versions (Simona2005-02 and older), no new executable is created if a non-default buffer size is requested.

general

- The viscosity field is uniform.
- Initial values for a restart are entirely computed by the preprocessor WAQPRE. For the processor no operational difference exists between a restart and a standard simulation.

curvilinear computation

- The SDS file contains spatial data referring to the transformed plane. (History data always refer to the physical plane).

6.3.1 Debug settings file

The executable program of waqpro has elaborate debug facilities, allowing developers to detect errors and users to obtain extra information concerning the progress of a calculation. The debug input file can be used to request exactly the necessary information. The input is based on SIMONA keyword structure. Refer to "About SIMONA" in Section 1 "General Information" and to "Input Description" in the User's Guide WAQPRE. The debug settings file has the following syntax:

```
POINTS
   <P[ival]:  M=[ival] N=[ival] K=[ival]
DOMAIN=[ival]
      | MODE = [ival]
      <
      | MODES_UNTIL = [ival]
   >

ROUTINES
   <R[ival]:
      GENERAL: [NAME]
         | TIME= [val] :  [val]
         <
         | NST = [ival] :  [ival]
         XDIR YDIR
         VERBOSITY = [ival]
         PRINT_SYSTEM
      | POINTS: P[ival]
      |
      <
      | DD_POINT: <POINT: P[ival], D[ival]
   >
```

Explanation:

POINTS	M	The keyword under which the points may be specified, about which extra information is needed. Solver-routines (WASSUC, WASUXC, WASDFC, TRSSUW, TRSCUE, TRSDIF, TRSTUR, WASKCH) print very compact lines with the calculated values at the given points. The specified points may also be selected in the subkeywords P of the keyword ROUTINES.
P	M	The keyword under which one point is specified.
M	M	M-coordinate of the specified point.
N	M	N-coordinate of the specified point.
K	M	K-coordinate of the specified point.

DOMAIN	M	Domain number of the specified point (in calculations without horizontal refining, the (only) domain is domain 1.
MODE	M	Index of the Kalman-mode about which the output is needed. In calculations without Kalman filter, the only mode is mode 0 (the solution itself).
MODES_UNTIL	M	Largest index of the Kalman-modes about which the output is needed. In calculations without Kalman filter, the only mode is mode 0 (the solution itself).
ROUTINES	M	Keyword in which the waqpro-subroutines are listed from which extra debug output is needed.
R	M	Keyword in which one subroutine is named for extra debug output.
GENERAL	M	Keyword under which all debug settings are given except the selected point.
[**NAME**]	M	Name of the subroutine from which extra debug output is wanted
TIME	M	Time span [minutes] for which extra debug output is wanted
NST	M	Time span [time steps] for which extra debug output is wanted
XDIR	O	Extra debug output selector for calculations in x-direction, for routines which can be called in x and in y-direction.
YDIR	O	Extra debug output selector for calculations in y-direction, for routines which can be called in x and in y-direction.
PRINT_SYSTEM	O	Output selector for printing the entire linearised system before solving. This keyword is only relevant for routines which set up linear systems and solve them.
VERBOSITY	D	Output selector (verbosity level) for the routine. Default level is 1.
POINTS	M	Point about which extra information is needed. In domain decomposition runs, it is possible to specify one point for each domain using the keyword DD_POINTS instead.
DD_POINTS	M	Points about which extra information is needed (max. 1 per domain).
P	M	Point number of the selected point: refers to values of keyword P in main keyword POINTS.
D	M	Domain number of the selected point. This value must correspond to the value of keyword DOMAIN in main keyword POINTS.

6.4 Additional options for using Waqua through OpenDA

Overview

The assimilation of observed data may be used to improve model predictions. For this Waqpro contains different Kalman filtering techniques. In the original implementation of this, the Kalman filter is implemented as an extension of Waqpro, and the configuration of the filter is provided to the pre-processor Waqpre and stored on the SDS-file. A new implementation is under development where the open source system OpenDA is used for the implementation of Kalman filtering. In this case part of the configuration is stored in the SDS-file whereas other parameters are specified on the command line for Waqpro.

It is **important** to ensure a proper OpenDA environment is available before using Waqua through OpenDA. Otherwise, an error will be written to waqpro's message file.
This paragraph describes the options that are relevant for the new implementation using Waqua through OpenDA.

format

waqpro.pl [...] [*-openda yes*] \
[*-nmode <nmode>*] [*-obsxml <xmlfile>*] [*-gui <yes/no>*] [*-parallel <yes/no>*] *[&]*

parameters

parameter	meaning and values
-openda	When set to "y(es)", the OpenDA system will be started and Waqua will be run within the OpenDA environment.
-nmode	Parameter for the data assimilation method that is used: number of modes.
-obsxml	The name of the XML file with references to observed data to be incorporated in the simulation.
-opendaconfig	The name of the XML file with the OpenDA configuration. This is the XML file that is passed to Application.sh in case OpenDA is run without waqpro.pl.
-opendafilterconfig	The name of the XML file with the OpenDA filter configuration. This is the XML file that is specified in the `<algorithm><configString>` section of the OpenDA configuration file (it replaces placeholder `OPENDACOSTACONFIG`).
-gui	A yes/no flag indicating whether the OpenDA GUI should be started. See the OpenDA documentation for more information. The default value is `no`.

notes

- When "-openda yes" is specified, the options -nmode and -obsxml are mandatory.
- If no observed data are needed for the run, the value "-" may be used instead of the name of an observed data file.
- The number of modes must be two or more, unless a simulation is run. In the latter case, the number of modes must be 0.

restrictions

- The Waqua through OpenDA implementation currently has a number of restrictions when compared to the original implementation of Kalman filtering in WAQPRO.

  The steady state filters cannot be used.

  Only waterlevel measurements can be processed.

  And finally the system currently works only on the Linux platform.

6.5 Extensions for parallel computing

overview

A parallel run consists of three phases:

1. "PRT", partitioning of the computational grid and of the input SIMONA data storage file, by running the partitioner program COPPRE,
2. "PRO", the actual simulation, and
3. "COL", gathering of the output data into a single SIMONA data storage file, by running the collector program COPPOS.

The actual simulation is carried out by multiple WAQPRO-processes, one per part/subdomain. Each of the WAQPRO-processes uses its own SIMONA data storage file with data for its own subgrid. The WAQPRO-processes exchange information on subgrid interfaces with each other via a communications subsystem using the MPI library. They are started by MPI and managed by the executive/master process COEXEC. The partitioning of the computational grid can be performed using an automatic partitioning method or can be specified via an input file.

data flow

Additional input files	**Additional output files**
Partitioning input-file	Partitioning report file

		SIMONA data storage file for distribution tables
		SIMONA data storage file per part
		Report file per part
file names	**logical name**	**system name**
	Partitioning input-file	free
	Partitioning report file	coppre-r.<runid>
	SIMONA data storage file for distribution tables	SDS-<runid>-000
	SIMONA data storage file per part	SDS-<runid>-<part>, with <part> the three-digit part number
	Report file per part	waqpro-r.<runid>-<part>
file description	**logical name**	**purpose**
	Partitioning input-file	Contains a user-defined partitioning of the computational grid into parts/subgrids for a parallel computation.
	Partitioning report file	Contains the partitioning that is used in a parallel run.
	SIMONA data storage file for distribution tables	Contains information determined by COPPRE and needed by COPPOS

user's guide See 'User's Guide for Parallel WAQUA/TRIWAQ and for Domain Decomposition'.

test input Not yet available.

exceptions

- In parallel/domain decomposition runs the report file will be generated separately for each part/subdomain, and will use the local numbering for that part/subdomain. The names of the files are then 'waqpro-r.<runid>-<part>', where '<part>' stands for the part/subdomain number in three decimal digits.
- The SIMONA data storage file for each part/subdomain is called 'SDS-<runid>-<part>', with '<part>' the part/subdomain number. The file 'SDS-<runid>-000' is used for transferring information from the partitioner to the collector. These files may be removed as soon as the collector has finished successfully.
- The partitioning input-file is needed only for manual steering of the partitioning and not if one of the automatic partitioning methods is used, see paragraph 6.9 below.

6.6 Extensions for domain decomposition with vertical refinement

overview vertical refinement

A domain decomposition run with vertical refinement proceeds largely similar to a parallel computation. One notable exception is that the decomposition (parallel run: partitioning) is necessarily user-defined and needs to be taken into account in the preprocessing phase (see chapter 4, preprocessor WAQPRE).
Therefore a domain decomposition run consists of two phases:

1. "PRO", the actual simulation, and
2. "COL", gathering of the output data into a single SIMONA data storage file.

Another difference with parallel runs is that, because of the different vertical resolutions used, the output data cannot be fitted immediately onto a single global grid. Therefore they are converted to the maximum vertical resolution used by the collector program COPPOS.

data flow

Additional input files	**Additional output files**
	SIMONA data storage file for distribution tables
	SIMONA data storage file per subdomain
	Report file per subdomain

file names/description

See the description for parallel runs, note that in this case the parts of a domain are called subdomains.

user's guide

See 'User's Guide for Parallel WAQUA/TRIWAQ and for Domain Decomposition'.

test input

Not yet available.

exceptions

- See the exceptions for parallel runs.
- The SIMONA data storage file SDS-<runid>-<part> for each subdomain uses the local vertical resolution.

6.7 Extensions for domain decomposition with horizontal refinement

overview horizontal refinement

A domain decomposition run with horizontal refinement proceeds largely similar to a run using vertical refinement. Again the decomposition of the grid is necessarily user-defined and is therefore taken into account in the preprocessing phase (see chapter 4, preprocessor WAQPRE). A run with horizontal refinement therefore also consists of two phases:

1. "PRO", the actual simulation, and
2. "COL", gathering of the output data into a single SIMONA data storage file.

A difference with other types of runs is that a run now may concern multiple global domains or "runid's". Therefore the input to the run-procedure must be entered into a configuration file instead of at the command line. A further difference is that a number of checks concerning the consistency of the different global domains cannot be performed in the pre-processing phase, which is carried out separately per global domain. Therefore the executive process COEXEC that manages the computing processes in the actual simulation has been extended with checks for domain decomposition with horizontal refinement. The option check_only is provided by the run-procedure to carry out just these checks, and not start the actual simulation.

data flow

Additional input files	**Additional output files**
process configuration file	SIMONA data storage file for distribution tables
	SIMONA data storage file per subdomain
	Report file per subdomain

file description

logical name	**purpose**
process configuration file	contains a list of global domains ("runid's") that must be included in the simulation and relevant parameters per domain.
other files	see the description for parallel runs.

user's guide

See 'User's Guide for Parallel WAQUA/TRIWAQ and for Domain Decomposition'.

6.8 Extensions for domain decomposition with horizontal and vertical refinement

overview A domain decomposition run with both horizontal and vertical refinement proceeds similar to a computation with horizontal refinement. Note: domains that are created with vertical refinement are handled as such in the "COL" phase.

6.9 Run data for parallel and domain decomposition runs

format parallel runs

*waqpro.pl [-runid <runid>] [-npart <npart>] *
[-partit <partit>] [-bufsize <bufsize>]

format vertical refinement

*waqpro.pl [-runid <runid>] [-ndom <ndom>] *
[-bufsize <bufsize>]

format horizontal refinement

*waqpro.pl [-runid <runid>] [-config <config>] *
[-check_only <ckonly>]

format horizontal & vertical refinement:

The format for horizontal and vertical refinement is identical to the format of horizontal refinement.

optional parameters for parallel runs and all kinds of domain decomposition:

extra options:

*[-hostmap <hostmap>] [-fases <fases>] *
*[-buf_prt <buf_prt>] [-keep_subdomsds <Y/N>] *
*[-use_pinning <Y/N>] [-xtra_comm <Y/N>] *
*[-use_pbs_aware <Y/N>] [-buf_exc <buf_exc>] *
*[-local_only <Y/N>] [-col_only <Y/N>] *
*[-col_online <Y/N>] *
*[-precision <single/double>] *
[-nmdbg <nmdbg>] [-back <Y/N>] [&]

parameters

Parameters	meaning and values
-back	y(es): program will be started in the background n(o): program will be started in the foreground
-bufsize	The size of SIMONA blank common in million words per WAQPRO computing process; a single value can be entered as well as a list of "npart/ndom" values separated by commas. The default value is 10.

	Note that when using domain decomposition with horizontal refinement, the bufsize is given in the configuration file.
-buf_prt	The size of SIMONA blank common in million words to be used by the partitioner and collector programs. Default: 10.
-buf_exc	The size of SIMONA blank common in million words to be used by master process COEXEC. Default: 10.
-check_only	Relevant in simulations with horizontal refinement only, requests to perform the consistency checks of different global domains only (yes/y) or also to perform the actual simulation (no/n, default).
-col_online	y(es): collect partial results during a parallel or domain decomposition computation, n(o) collect results after all waqpro processes are finished. Default: yes
-col_only	y(es): an alias for "-fases col", n(o): an alias for "-fases all".
-config	File name of the process configuration file to be used in case of horizontal refinement, which lists among others the "runid's" per global domain in the simulation.
-fases	Detailed control over the separate phases of a parallel or domain decomposition computation. Possible values: all, prt (only in parallel runs), pro, col. Default: all.
-hostmap	The mapping of subdomains onto hosts or computing nodes. For each subdomain the host-number (1..num_hosts) may be given, separated by comma's. When the option -col_online Y is used the hostmap should include additional entries at the end of the list, one for each coppos process. Also the mappings Compact, Packed or Round-Robin may be given. Default: Packed mapping mechanism. When the hostmap is not correct Round-Robin is used as a fallback.
-keep_subdomsds	After successful collection of the subdomain SDS files in one overall SDS file, the subdomain SDS files will be removed. This can be avoided by using this option with the value yes.
-local_only	Compute only on local machine (Windows only).

-nmdbg	The name of the file with debug settings (see 6.3.1).
-ndom	Number of subdomains used in a domain decomposition run. Also used as switch to activate the domain decomposition functionality (=1: no domain decomposition, >1: domain decomposition).
-npart	Number of parts to be created by the partitioner in case an automatic partitioning method is used. Also used as switch to activate the parallel functionality (=1: not parallel, >1: parallel).
-partit	Partitioning to be used in a parallel run. Possible values: automatic, orb, orb_row, orb_col, strip, strip_row, strip_col, <*filename*>. Default: automatic.
-precision	Switch between single and double precision for the computational core. Possible values: single or double. Default: double.
-runid	Code to identify the output files of a run. In simulations with horizontal refinement this code is used for the message file only, then the runid's per global domain are given in the configuration file.
-use_pbs_aware	This option can be used when running Open-MPI in combination with the taskbroker PBS. It avoids complications when running on many cores ($>$ 100), but it neglects the option hostmap.
-use_pinning	Pin the waqpro processes onto a core. This may result in a higher performance. It is only implemented for MPICH2, open-mpi and Intel MPI on linux. For Intel MPI, pinning is enabled by default. Note that also coexec is pinned, so you need 1 slot more then without pinning. When multi-threading is set on, this is not a problem.
-xtra_comm	The communication scheme to be used. The (default) value 0 chooses the standard, efficient scheme. Larger values cause more communication, and should lead to identical results as the default. This option exists to facilitate testing the communication scheme.

notes

- The type of run is determined firstly determined through the occurrence of the option config, which signals a run using horizontal refinement. Else if ndom>1 a run with vertical refinement is assumed, if npart>1 or fases=PRT or COL then a parallel run is executed, and if npart=1 and ndom=1 are given, a standard sequential run is performed.
- In case of a background run (with &) runid is an obligatory parameter. When not using horizontal refinement (if option config is not given) then npart and partit (if npart>1) are obligatory parameters; ndom has the default value 1, and back has no effect.
- If a run is not a domain decomposition or parallel run (config not given, fases not given, ndom=npart=1), then the parameters check_only, buf_prt and partit have no effect. They need not be given and will not be asked for in both background and interactive runs.
- In case of an interactive start of the run the procedure will prompt for all required parameters mentioned above. All parameters are checked before the program is started.

Domain decomposition with horizontal and optionally vertical refinement

- The process configuration file largely consists of a list of global domains. For each domain must be given a NAME, the RUNID that has been used in the pre-processing phase (WAQPRE), an EXECutable name, and the required BUFSize.
- The executive program COEXEC figures out itself how many active subdomains are used per global domain. This number can be >1, so that horizontal refinement can be combined freely with parallel computing. Combination with vertical refinement is supported too.
- For each domain listed in the configuration file WAQPRE must have been executed successfully, and consequently the SIMONA data storage files SDS-<runid>-<part> must exist for all subdomain-numbers <part> that are used. Further the presence of the file SDS-<runid>-000 is verified by the run-procedure.

Domain decomposition with vertical refinement

- Whether a run can be using domain decomposition with vertical refinement is derived from the presence of a file SDS-<runid>-000 in the current working-directory and from the values of options ndom and npart.
- Parallel computing and domain decomposition cannot be specified simultaneously, therefore a value ndom>1 specifies that domain decomposition is to be used and either ndom=1 or npart>1 specify that domain decomposition is not to be used. In background runs, the default value ndom=1 will be used if ndom is not given on the command line. The value specified for ndom must be equal to the number of subdomains created in the preprocessing phase (see chapter 4, preprocessor WAQPRE). It is an error to specify both ndom>1 and npart>1.
- In case the options ndom and npart are inconclusive with respect to the use of domain decomposition (i.e. in interactive runs only) and in case the file SDS-<runid>-000 exists, the procedure asks whether domain decomposition is to be used or not. If so, it asks for the number of domains ndom.

Parallel computation

- The partitioning of the computational grid can be done through one of the automatic partitioning methods ORB and STRIP, with the initial splitting-direction left open or specified (_ROW=along rows, _COL=along columns). Also the complete partitioning may be specified by giving the part-number for each grid-point in a partitioning input-file. The method AUTOMATIC chooses between ORB and STRIP, and the initial splitting-direction.

General notes on buffer sizes

- An estimate of the buffer size that is needed by TRIWAQ for a subdomain can be obtained from the parameters MMAX, NMAX and KMAX of the subdomain:

  buffersize $\approx$ (85*KMAX + 110)*(MMAX + 6)*NMAX

  The actual buffer size that is needed per subdomain is printed at the end of a simulation, together with the report of simulation start and end dates and the total amount of CPU-time used.

  If the given buffersize is less than needed, it will be automatically increased.

Message file, phases of a computation

- A parallel run consists of two phases: partitioning of the input SIMONA data storage file by running the partitioner (coppre.exe), the parallel computation itself which employs MPI, the executive (coexec.exe) and multiple computing processes (waqpro_<precision>.exe) and one or more collecting processes (coppos.exe). Optionally, the gathering of the output data by running the collector program (coppos.exe) can be done as third (post-process) phase (see -col_online). For a domain decomposition the partioner is not used.

- The message-file is removed only at the start of the first phase of a computation: before the partitioning phase of a parallel run or at the start of the processing step of a sequential or domain decomposition run. In case only the processing or collecting phases are executed for a parallel run, or in case only the collecting phase is executed for a domain decomposition run, the output of waqpro_<precision>.exe and coppos.exe is appended to an existing message-file.

- Usually all phases are started together, but occasionally it may be necessary/desired to start only one of the phases. This is especially so in case a parallel or domain decomposition run was terminated abnormally, such that the output data have not been gathered using the collector program. Therefore in case the file SDS-<runid>-000 exists and the run is not a domain decomposition run, the run-procedure asks whether you want to run the collector only. In other circumstances the option fases must be given at the command line, otherwise the default value fases=all is used.

Start-up of coupled (parallel / domaindecomposition) runs

- The processes that are needed in a coupled run are one instance of coexec.exe and coppos.exe and <num_prc> instances of waqpro_<precision>.exe one for each subdomain. These processes are started using the message passing interface MPI. Multiple implementations of MPI are available, more or less comparable to the various Fortran compilers that can be used for generating executables. The MPI-implementation that is used in SIMONA is configured in the file Settings.inc that is located in the directory etc/linux or etc/win32 of the SIMONA-installation.

- The runprocedure waqpro.pl determines an appropriate mapping of all subdomains over the available machines. The available machines may be obtained through a batch scheduler (PBS) or through an auxiliary configuration file "hostfile.gen". Normally waqpro.pl will allocate each subdomain in turn to the next available cpu in a Packed fashion. For this the number of processors or cores per host (ppn) may be specified in "hostfile.gen". The mapping may also be specified by the user using the option -hostmap. For more details see the 'User's Guide for Parallel WAQUA/TRIWAQ and Domain Decomposition'.

Chapter 7

Program SIMPAR

7.1 Function

SIMPAR is a post-processing program that simulates the transport of floating particles and particles that are representative for dissolved substances. SIMPAR calculates the displacement of particles due to advection, diffusion and wind. The contribution of advection is taken care of by the water movement model of WAQUA. SIMPAR uses the same grid as WAQUA.

7.2 Files

Data flow.

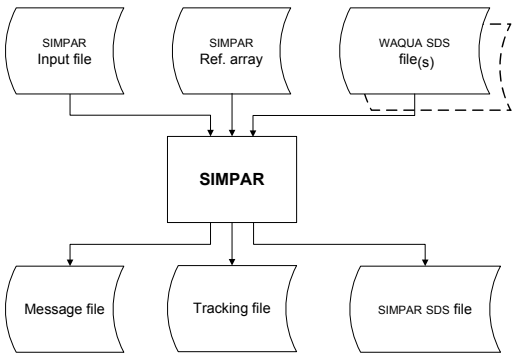

Figure 7.1: Flow chart of SIMPAR

The following table gives the names of the input and output files as used by the run procedure and the purpose of these files. The output files of SIMPAR are collected in the current directory.

Logical name	System name	Purpose
input file	parinp.<runid>	particle model definition
reference table	simparref.arr	input file checking
WAQUA SDS file	SDS-<old fantasy>	storage permanent data
message file	simpar-m.<runid>	programme output
tracking file	simpar-t.<runid> / <user's fantasy>	storage particle trajectories
SIMPAR SDS file	SDS-<new fantasy>	storage permanent data

7.3 Run data

format

Procedure call:
*simpar.pl [-runid <runid>] [-input < input >] [-back < back >] *
[-debug < debug >] [-bufsize <bufsize>] [&]

parameters

parameter	**signification and value**
-runid	Code (alpha-numeric) to identify the output files; e.g.: -runid pab17
-input	File name of simpar input file; e.g.: -input parinp.pab17
-back	y(es): program is started in the background n(o): program is started in the foreground
-debug	y(es): program run in debugging mode
-bufsize	The size of SIMONA blank common in million words, i.e. the space needed to store all SIMONA arrays; default: 10

notes

The procedure will:

- in an interactive start of the run automatically prompt for any of the above mentioned parameters, if not given in the procedure call.
- in a background run denoted by &, necessitate the presence of the runid and input parameters and ignore the back parameter.
- forbid to debug in the background.
- use the first found SIMPAR executable in the PATH environment variable to run the program.
- check the parameters before the run is started.
- In contrary to previous versions (Simona2005-02 and older), no new executable is created if a non-default buffer size is requested.

Chapter 8

Program SLIB3D

8.1 Function

purpose The SLIB3D processor SLIB3D computes displacement of the centre of the suspended matter

8.2 Files

data flow

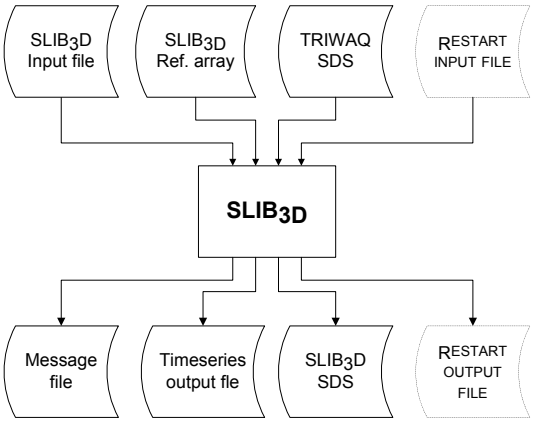

Figure 8.1: Flow chart of SLIB3D

Input files	**Output files**
SIMONA data storage file	SLIB3D SDS-file
SLIB3D input file	Message file
Restart input file	Restart output file
	SLIB3D output file

		Time series file
file names	**logical name**	**system name**
	TRIWAQ SDS-file	SDS-<runid>
	SLIB3D input file	slibinp.<runid>
	SLIB3D SDS-file	SDS-SLIB3D-<runid>
	Message file	slib3d-m.<runid>
	Restart input file	user defined
	Restart output file	user defined
	SLIB3D output file	user defined
	Time series file	user defined
file description	**logical name**	**purpose**
	TRIWAQ SDS-file	Contains data from the WAQUA/TRIWAQ program.
	SLIB3D input file	Contains input in a structure suited for the SLIB3D program
	Restart input file	Contains restart input in a structure suited for the SLIB3D program.
	SLIB3D SDS-file	SLIB3D output data.
	Message file	Contains error messages and diagnostics.
	Restart output file	Contains restart output in a structure suited for the SLIB3D program.
	SLIB3D output file	Contains mass (moments) or concentration.
	Time series file	Contains time series.

user's guide See 'User's Guide SLIB3D'.
test input Not yet available.

8.3 Input and output

Input files TRIWAQ SDS-file:

In this SDS-file the hydrodynamical input data is stored. This data is made by the WAQUA/TRIWAQ program. Therefore it is necessary to run the WAQUA/TRIWAQ program first when the correct SDS-file is not available (see "User's Guide WAQUA").

Slib input file:

This file contains the input data necessary to run the SLIB3D program. The input is based on the keyword structure. The explanation of these keywords is given in the "User's Guide SLIB3D" Section 2 "Input description".

Restart input file:

This file contains the input data for restart. This file is made by the SLIB3D program in an earlier run.

Output files

SLIB3D SDS-file:

In this SDS-file all the output data is stored.

Note: Apart from the SDS-file there are two (temporary) "Aquavision"-files: the "SLIB3D output file" and the "Time series file". Both files are filled with ASCII-data. However, these two files are not indispensable, contents are not verified or tested.

Restart output file:

In this file the restart data is stored. The data is stored in blocks
Each block contains
at line 1 : number of moments in the waterlayers and in the bottom layer
at line 2 : layer number, M number gridcell, N number
gridcell, mass, x-coordinate mass center, y-coordinate mass center, z-coordinate mass center, variance in x-direction, variance in y-direction, variance in z-direction, x,y covariance
The next lines are the same as line 2.
After storing the values of all the waterlayers the values of the bottom layer will be stored. The format is almost the same format for storing the values of the waterlayers, except there is no layer number.

SLIB3D output file:

In this file the concentration or the moments, depending on the given value of 'lsli' .

In case of storing moments, blocks of values are stored.
Each block contains
at line 1 : '$CYCLE ' number of the cycle
at line 2 : '*'

at line 3 : 2 characters, layer number, N number grid cell, M number grid cell, mass, x-coordinate mass centre, y-coordinate mass centre, z-coordinate mass centre, variance in x-direction, variance in y-direction, variance in z-direction, x,y covariance and (0.0 , 0.0).
The 2 characters can be: Pb, Ph or Pv depending on the values of the variance.
The next lines are the same as line 3.
The last line of each block is '###'

After storing the values of all the water layers the values of the bottom layer will be stored. The format is the same.
Notice that the value of the z-coordinate mass centre and variance in z-direction are 0.

In case of storing concentration, also blocks of values are stored.
Each block contains
at line 1 : '$CYCLE ' number of the cycle
at line 2 : '*'
at line 3 : 'Pc', layer number, N number grid cell, M number grid cell, mass, vertical surface area of the grid cell, layer thickness, concentration and (0.0, 0.0).
In the bottom layer the layer thickness = 1.0
The next lines are the same as line 3.
The last line of each block is '###'

Time series file:

In this file values of a cross section are stored.
For (M=constant) cross sections data in the U direction is stored, otherwise (N = constant) cross sections data in the V direction is stored.
Also in this file the data is stored in blocks.
Each block contains
at line 1 : '$CYCLE ' number of the cycle
at line 2 : '*'
at line 3 : 'Pc', layer number, N number grid cell, M number grid cell, mass, vertical surface area of the grid cell , layer thickness, flow and grid distance.
In the bottom layer the flow = 0.0 and the layer thickness = 1.0
The next lines are the same as line 3.
The last line of each block is '###'

8.4 Run data

format *slib3d.pl [-runid <runid>] [-input < input >] [-back < back >] *
[-debug < debug >] [-bufsize <bufsize>] [&]

parameters	**parameter**	**signification and value**
	-runid	Code (alpha-numeric) to identify the output files; e.g.: -runid slib01
	-input	File name of SLIB3D input file; e.g.: -input slibinp.test1
	-back	y(es): program is started in the background n(o): program is started in the foreground
	-debug	y(es): program run in debugging mode
	-bufsize	The size of SIMONA blank common in million words, i.e. the space needed to store all SIMONA arrays; default: 10

notes

The procedure will:

- in an interactive start of the run automatically prompt for any of the above mentioned parameters, if not given in the procedure call.
- in a background run denoted by &, necessitate the presence of the runid and input parameters and ignore the back parameter.
- forbid to debug in the background.
- use the first found SLIB3D executable in the PATH environment variable to run the program.
- check the parameters before the run is started.
- In contrary to previous versions (Simona2005-02 and older), no new executable is created if a non-default buffer size is requested.