VORtech Computing

Industrial and Scientific Computing

P.O. Box 260
2600 AG Delft
The Netherlands

tel. +31(0)15-285 0125
fax. +31(0)15-285 0126
vortech@vortech.nl

MEMO	EL/M07.056
Date	December 12, 2007
Author(s)	dr.ir. G.E. Loots
Subject	Making WAQUA/TRIWAQ OpenMI-Compliant

Document information

Version	Author	Date	Description	Review
0.9	EL	27-08-2007	Concept report	
1.0	EL	12-12-2007	Adjusted report	
File location:		/home/vortech/Memos		

1 Introduction

RIKZ is participating in the 3-years European project called "OpenMI-Life". This is a demonstration project for OpenMI, a standard that allows time-dependent models to exchange data at run-time. When the standard is implemented, existing models can be run together and exchange information at each timestep.

In this case, two flow models of a different structure, each describing a separate region, from two different organisations will be coupled. The first model is WAQUA/TRIWAQ, used by RIKZ, is called 'Kust-Zuid' and covers the Dutch part of the Schelde basin. The other model, Mike11, from DHI, solved the flow in the Flemish part of the Schelde basin.

This document describes the work that has been performed for the extension of WAQUA and is also a 'getting-started' to perform the first simple test cases.

2 Changes to WAQUA

- The program file waqpro.f has been split in three subroutines: waqpro_init, waqpro_loop and waqpro_finalize. One of the arguments of waqpro_init is the openMI-flag. WAQUA can still be run in the old way by running the new program waqpro.f which only call the above mentioned subroutines, with the openMI-flag set at zero. The program waqpro.exe is generated from the project `waqpro`. Thus, it is still possible to use the old functionality of WAQUA.

- Two other projects have been added: `waqprodll` and `waqprodlltest`. `waqprodll` uses all `wapqro-` files and contains also a set of OpenMI-dll-functions (`OpenMIDLLFuncts.F90`).

These functions use information from two modules: a high-level module `waqpro_in_open_MI` which contains time information and where the three functions Initialize, Perform-Timestep and Finalize are defined. The module has access to the waqpro-interior via intgda. The second module `waqpro_data_in_open_MI` exists on a lower level and contains the exchange-administration. Via this administration, i.e. a buffer containing all exchange-items, communication between the WAQUA-kernel and upper wrapper layers takes place.

- In the initialize-phase, all possible exchange items are identified and administrated. That means that later, at run-time, the exchanges can easily take place using the administration. Insize the WAQUA code, a few routines contain the openMI-flag and the possibility to retrieve or provide information towards the administration buffer. Those routines use the module `waqpro_data_in_open_mi`.

- At this moment, the following exchange items exist:

  - **source**. A location inside the computational domain provides a discharge. In `WAQPRE`, it is defined by FORCINGS → DISCHARGES. Its role in OpenMI is `ACCEPTING`. In normal circumstances, the timeseries of this source is used in the routine `wassdr` which provides a value of the discharge at each timestep. If there is a connection (the openMI-flag is true and a value from outside is provided) the value will be overruled.
  - **open boundary** This is a strip defined by a begin-point and an end-point (in m,n) coordinates). In `WAQPRE`, it is defined by FORCINGS → BOUNDARIES . Its role in OpenMI is `ACCEPTING`. Possible quantities are waterlevel, (automatic) discharge,and velocities. In fact, the type of quantity is not very important. It is assumed that one or two timeseries are originally provided. In the routine `wassov`, the time series is processed further. In the case of an openMI-flag, the value (per opening point, which is acquired by linear interpolation from the two end points) is overruled from outside. Later, in the case of automatic discharges, the routine `wasfqa` distributes these values in the normal way.
  - **checkpoint** A checkpoint is a point of which information is written to the SDS-file. Its role in OpenMI is `PROVIDING`. At this moment, only the waterlevel is given. Note that the information is available at every timestep, not just at the (possibly very few) time instances when data is written to the SDS-file.
  - **Special exchange item: grid** The very last exchange item that will be added in the initialization phase is the grid with respect to the waterlevel. the grid exists in the `waqpro_data_in_openMI` module. It is possible to perform a GetValues call for the grid *before* the time loop. See the `waqprodlltest` explained below for details. Note that the administration of each exchange item contains the (m,n) co-oordinate as well as its corresponding (x,y) value.

- An important aspect is that all exchange items need a unique ID. This can be provided in the simpinp input file by providing a name of the points (discharge point, checkpoint) or open boundaries: BOUNDARIES → OPENINGS: `open1 = line(p1,p2, 'name_of_opening')`.

3 Testcase: kustzuid_wq

This model covers the entire Scheldt basin and a part of the southern North sea. A discharge (point 321 called BRON) exists upstream (`/randvw/flow/discharge`). Further, a lot of open boundaries exist (`randen/openings-combi`). Finally, checkpoints are defined (`locaties/points-combi`).

The waqpre-part, which provides the necessary SDS-file, can be re-run using start.bat in the `Berekeningen` directory. The simulation date is 02.01.2004. Start time is 08h00. It lasts 20 minutes with one-minute time steps. This can of course be changed by adjusting the siminp-file.

Note that a few actions need to be performed to run the test:

- The SIMONADIR environment variable needs to be set.
- simona.env and SIMETF need to be copied to the working directory Berekeningen.
- The input file omi-invoer.txt (in Berekeningen) needs to be filled. It consists of five lines:
  1. name of the SDS-file (here: SDS-kzv3)
  2. name of the report file (here: waqpro-r.kzv3)
  3. name of the experiment (here: kustzuid-v3)
  4. buffersize in Mw (here: 10)
  5. offset for file unit numbers. This is important is two WAQUA-runs are performed simultaneously. One of the runs should have an offset unequal zero, for example 10.

The last two actions are necessary because the `waqpro.pl` cannot be used. Normally, the information above is given by `waqpro.pl` as command line options to `waqpro.exe`.

- **First test: test waqprodlltest** Here we make not yet use of the OpenMI wrappers. The functions that are needed are already filled in. The program waqprodlltest simply does the same what OpenMI is also supposed to do:
  1. Initialize: ask for all exhchange items
  2. Loop:do some timesteps:

  - Setvalues: ask for information from outside. In this test, the original value of BRON of 250.0 will be overruled to be 1.0.
  - Performtimestep
  - Getvalues: provide the values in the checkpoints. In this case, the waterlevel in Westkapelle is given.
3. Finalize.

The test can be run using the batch testdll.bat in `kustzuid_wq/ Berekeningen`.

- **Second test: via OpenMI** Now we use the OpenMI editor to establish a similar test, where the discharge will be replaced by a sinus. Here the upper wrapper layers will be used, which are the same as the standard WL wrappers for other applications. In fact, the Waqua project which is used is almost the same as the other projects (Delft3Dflow and Sobek).

  The test can be started by running the OmiEd.exe in `Wrappers /OpenMIWrapper/bin`.

  - First, run start0.bat in `kustzuid_wq/Berekeningen` which copies the waqpre-generated SDS-file to the Berekeningen directory (this is necessary because the SDS-file is changed during the simulation)
  - Menu File: Open: choose SinusplusWaqua.opr
  - By right-clicking on the yellow models the exchange items are visible.
  - By right-clicking on the connections the connections are visible. In this case, the (H,anywhere) is connected to (discharge, BRON)
  - the composition is defined in omi-files (in the `kustzuid_wq` directory) and the mentioned opr-file in the same directory.
  - Menu Composition: Run: This runs a simulation. The trigger time must be inside the Waqua simulation interval.

- **Third test: two coupled Waqua-simulations**

  In two different directories, `kustzuid_wq` and `kustzuid2_wq`, identical simulations are defined. OmiEd can be started and the file Waquapluswaqua.opr can be loaded. This defines two models, defined in `test_waqua1.omi` and `test_waqua2.omi`. The Getvalues of the second model (here: the water level at Vlissingen) is the Setvalues of the first model (the discharge BRON).

  To prevent interference between the two models, the following measures have been taken:

  - Them second model is called `Waqua2`.
  - the WLDelft.OpenMI.Wrapper.dll has been changed. For the model `Waqua2` it uses `waqpro_2.dll` which is in fact a copy of `waqpro.dll`

- in the omi-file, a unit number offset is needed to prevent the two dll's to write the same SDS-file.
- MPI is turned off. This is done by using the `mpidum` option in WAQUA.
- log messages have been redirected to each of the `waqprodll-m.kustzuid_v3` files.

4 Current status and future activities

At this moment, the `waqpro` DLL has been made OpenMI-compliant in the same fashion as at least two WLDelft applications: Sobek and Delft2D. In fact, it is now merged into a solution space with these other two applications.

Adjustments of the outer wrapper for providing geometrical information will be made in the near future. This will also be performed on a more general level, not just for WAQUA.

More exchange items for the Setvalues and the GetValues can be added in a similar way as described above. Each specific exchange item needs a small extension in the computational kernel of `waqpro`. The OpenMI module will be used in these routines.