Notes About the Training, by J. Gregersen

The OpenMI training material included in the .zip file was created in connection with the OpenMI training at NTUA in Greece – October 2006. The training was part of the OpenMI LIFE project deliverables.

The training was a combined end users and developers training. However, all participants attended both the end users and the developers training. The duration of the training was three days, where the first day mainly focused on the main concepts and how to use OpenMI as an end user. Linking of the simple river and the simple ground water model was demonstrated and the participants repeated this on their own PC's.

Another exercise was to connect two rivers, where one of the rivers was a tributary to the other river. The participants checked the output files to see if the results were as expected. Data from the data directory was used (Rhine and "small river").

The developers' training was focused on migrating the simple river model. A visual studio solution was prepared, so all projects, classes and method definitions were done beforehand. The participants implemented the methods in the IEngine interface and after each implemented method, the unit test was executed. All unit tests were also prepared beforehand.

The whole exercise could be done using Microsoft visual studio express, which can be downloaded for free. Running unit test from this environment was a bit tricky, because the express version does not allow plug-ins and attach process (for unit test, it is important to be able to debug the individual test when something goes wrong). This was solved by use of a unit test project

(see the exercise solution

…OpenMITraining\Athen\Examples\RiverModel\Exercise\Rivermodel.sln).

The order of method implementations is defined in the document exercise.doc.

After completing the implementation of the river model and checking that the omi file was correct, all participants loaded the RiverModel into the OpenMI GUI and made the model run.

Most of the participants had very limited experience with c# and oo. This was the reason for preparing the solutions, classes and method definitions beforehand. In this way we could focus on OpenMI and not on how to write c# code.