
Karsten Rink\textsuperscript{a}, Lars Bilke\textsuperscript{a}, Benny Selle\textsuperscript{b} and Olaf Kolditz\textsuperscript{ac}

\textsuperscript{a}Dept. of Environmental Informatics, Helmholtz Centre for Environmental Research, Leipzig, Germany (karsten.rink@ufz.de, lars.bilke@ufz.de)

\textsuperscript{b}WESS – Water & Earth System Science, University of Tübingen (benny.selle@uni-tuebingen.de)

\textsuperscript{c}Dept. of Applied Environmental Systems Analysis, Technical University of Dresden, Germany (olaf.kolditz@ufz.de)

Abstract: We present a combined framework for the exploration and assessment of geoscientific data, preparation of hydrological models, simulation and visualisation of the results. A user interface is provided for 3D visualisation of data sets and construction of hydrological models. The structure is modular and functionality is accessed via interfaces, allowing to switch or replace the implementation of parts of the framework (e.g. numerical solver for simulation or filters for visualisation) without any impact on the overall programme. Functionality is accessed via a graphical user interface that guides the user through the modelling process. We present an example workflow for the creation of a model for a groundwater simulation for the Ammer catchment in southern Germany in the scope of the WESS project. We also present the extensive testing environment that has been created to ensure software quality and robust results for a framework with multiple developers.

Keywords: Hydrology; Modelling; Visualisation

1 INTRODUCTION

Faced with the potential impacts of climate change, the simulation of hydrological processes such as overland flow or groundwater recharge has become increasingly important in recent years for the development of long term strategies concerning water management as well as water quality.

While always dependant on certain regional characteristics, the preparation of models to run such simulations always follows a certain pattern. The input data usually consists of a number of heterogeneous data sets, often originating from a large number of sources. Typically, remote sensing data (e.g. digital elevation models (DEM)) is combined with hydrogeological data sets (network of waterbodies, boreholes, precipitation, etc.). By providing a means to easily create such models – either to be used directly or to be refined afterwards based on the aforementioned characteristics – it is possible to deal with the uncertainties related to an imperfect data base (measuring errors, integration of values, etc.), the effects of certain processes or parameters, and the necessity of using...
Table 1: Overview of existing interfaces

<table>
<thead>
<tr>
<th>Data type</th>
<th>Formats / Programmes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raster data</td>
<td>GeoTiff, ASCII, NetCDF, JPEG, PNG, BMP</td>
</tr>
<tr>
<td>Features</td>
<td>Shape</td>
</tr>
<tr>
<td>Meshes</td>
<td>GMS, FEFLOW, TetGen, GMSH</td>
</tr>
<tr>
<td>Modelling software</td>
<td>GMS, Petrel, GoCAD</td>
</tr>
<tr>
<td>Time series data</td>
<td>CSV, WaterML</td>
</tr>
<tr>
<td>Graphic formats</td>
<td>VTK, OpenSG, VRML</td>
</tr>
</tbody>
</table>

only a limited number of input data sets. This allows for the testing of a large number of hypotheses and impacts of certain effects on the model and the subsequent simulation. Visualisation of relevant data at each point of the preparation process is vitally important for verification and assessment as well as to deal with potential problems.

The visualisation of heterogenous data sets is a nontrivial matter that requires high adaptability of the employed methods [Jones et al., 2009; Uhlenkünken et al., 2000]. Once available it supports users considerably in detecting essential details concerning the model and allows for the recognition of errors within the data or inconsistencies between data sets.

We propose a framework designed to provide the described functionality by implementing the necessary interfaces to import data sets, to create models as well as to modify the data or its visualisation for extraction of specific information. In section 2 we will describe typical workflows and how the user is supported by existing interfaces. In section 3 we will show an example of a simulation set up using our framework.

2 Software Concepts for Hydrologic Modelling

Our framework is called OpenGeoSys [Kolditz et al., 2012] and has been originally developed for the simulation of coupled thermal, hydrological, mechanic and chemical processes in fractured porous media. We recently attached a graphical user interface (GUI) – the OpenGeoSys Data Explorer [Rink et al., 2012b] – for the visualisation of input and output data as well as semi-automated generation of models (fig. 1). Although the framework is in ongoing development it has already been used for building models for hydrological simulations in model areas in the Middle East, Germany [Rink et al., 2011] and China [Sun et al., 2012].

2.1 Workflows

While the framework offers a large range of modular functionality, workflows can be devised for typical problems. The following gives an outline of the typical workflow for setting up a hydrological subsurface-model:

1. Import of relevant input data: OpenGeoSys supports a wide range of file formats from popular geoscientific programmes and modelling software (see table 1). It also includes a XML-parser which is used for native files but can easily be extended to
existing XML-based formats such as WaterML [Whiteaker and To, 2008] for logger data or VTK [Schroeder et al., 2006] for graphical representation of data. The minimum requirement for input data is the boundary of the model region as well as a DEM. Additional data can include water bodies, boreholes, observation sites, etc.

2. Verifying the data: Each data set is automatically visualised in 3D space. Loading all relevant input data sets allows to detect inconsistencies between data sets. This includes general problems such as differing projections but also details such as wrong offsets for boreholes, overlapping stratigraphic layers and many more. The framework also provides a range of visualisation-techniques and -algorithms to support users in this process (see [Rink et al., 2011] for details).

3. Creating a 3D surface model: The model is based on the DEM but additional information (borehole locations, river courses, etc.) can be added via a simple dialogue. The framework uses a default parameter set for mesh generation that usually gives adequate results. Experienced users can set the parametrisation specific to their needs.

4. Adding subsurface information: Based on borehole information or on DEMs of subsurface layers the surface model can be extruded into a full 3D model. We don’t offer functionality such as kringing in our framework because it is available in specialised tools that are necessary for providing the input data.

5. Assigning properties to the various material groups of the model: As of this writing this is not fully implemented. While the simulation algorithms make use of material
parameters, setup via GUI is not yet implemented and respective files are currently created manually.

6. Assigning boundary- and initial conditions: Conditions can either be assigned to geometrical objects or directly to nodes of the model. A dialogue offers selection of various available process types as well as associated parameters (see fig. 1). Conditions are then separately visualised (see [Rink et al., 2012a] for details).

Obviously, this workflow can be different when existing models are imported from other software such as GMS or FEFLOW. Likewise, it is possible to describe workflows for each step of the instructions given above (e.g. calculation of boundary conditions from raster data) as well as for other applications of the framework (such as estimation of mesh quality, calculation of contour lines from 3D raster data, etc.).

2.2 Interface Design

The ability to describe workflows is based on the fact that most of the functionality of the Data Explorer is encapsulated into modules with defined interfaces for input and output.

We differentiate between two types of interfaces:

1. Software interfaces: Defined modules with a given set of input and output parameters.

2. Dialogues: Parts of the GUI that allow users to specify parameters before accessing certain functionality. As human-computer-interfaces these could be classified as a subset of the first group.

With our framework being implemented in C++ we make extensive use of the object-oriented software paradigm, including concepts such as abstraction, modularity, inheritance, etc.[Stroustrup, 1997]. Specifically, we encapsulate functionality into exchangeable modules with a fixed set of input and output parameters. As OpenGeoSys is a complex programme, this is necessary for modifications or replacements of functionality, the addition of multiple algorithms for obtaining a certain result and the general maintainability of the whole project. Some objects of the programme have similar functionality and are thus derived from 'base classes', offering the same methods using different implementations (i.e. Polymorphism). While a detailed explanation of these concepts is outside the scope of this paper, we will offer a simple example: Functionality to load a mesh from a file requires the filename as input and outputs the mesh as a result. The user interface need not differentiate between OpenGeoSys meshes or meshes from other software such as GMSH, FEFLOW, TetGen or the Groundwater Modeling System as the different implementation in the respective interface objects will handle the details. Similar functionality is implemented for the use of different process types for the simulation, various distribution types for boundary conditions or the structure of different geometric objects.

The implementation for the second type of interfaces, i.e. dialogues, is similar: At each point of the workflow only functionality whose application makes sense at that moment can be accessed (e.g. surface mapping is not available without a DEM, specific visualisation-algorithms can only be applied to certain types of data, etc.). Furthermore, error messages will notify users in case of possible problems. However, one should keep in mind that this is not a commercial product but a specialised software and as such only a means to an end for researchers. To provide the nec-
Figure 2: Infrastructure of the build server and management system.

necessary functionality within a reasonable time frame and to avoid unnecessary er-
ners, we make use of existing software libraries such as Qt (http://qt.nokia.com) for GUI development, VTK (http://www.vtk.org) for 3D visualisation or PETSc (http://www.mcs.anl.gov/petsc/) for solving numerical systems.

It is worth noting that all of these software libraries and tools as well as the manage-
ment utilities introduced in the next section are open-source projects and available at no charge. Likewise, our own framework is freely available for scientific use with plans to go open-source in the near future.

2.3 Automated Testing

The development and maintenance of OpenGeoSys is a joint effort of a number of Eu-
ropean universities and research institutions. The development team consists of about half a dozen core developers and a large number of users, many of whom have also added features over time. In addition, the comprehensive pre- and postprocessing tool GINA has been developed for geotechnical applications (such as geological waste depo-
sition) [Kunz et al., 2011]. Responsibilities between core developers are divided between the core functions of the framework (such as optimisation of data structures, GUI & visu-
alisation, maintenance & automated testing, hydrological or chemical simulations, etc.). Users or part-time developers may add to any part of the programme in coordination with the core team.

This requires an extensive infrastructure to ensure the availability of the latest version of the software on all supported operating systems, the proper integration of new features and the guarantee of repeatable and correct results for all implemented functionality.

To manage the contributions of multiple authors, we use GIT (http://git-scm.com/), a version control system that allows for the parallel development, merging of
source codes and management of new features. In addition, we use Jenkins-CI
(http://jenkins-ci.org/) as a continuous integration application. Upon any submitted
change to the source code via GIT, a number of operations and tests are automatically
started by this build-server. These processes include:

- Compiling the source code for each supported operating system (OS), i.e. Windows
  32 and 64bit, Linux, Mac OS.
- Providing a downloadable version to our download site for each OS
- Testing functionality of features of the programme, e.g. generation of meshes from
  geometry, conversion of input formats, etc.
- Simulation of over 100 test cases to ensure robust results of all the implemented
  solvers.

Should any of the above steps fail, the respective developer is notified and can solve
the problem as soon as possible. In addition, the build server will display warnings for
potential problems in the source code, check how often implemented functionality is ac-
tually used (code coverage) and automatically create a documentation for the software
based on developer-comments. Results of all of all of the above steps are accessible via
a web-frontend provided by Jenkins-CI.

The complete infrastructure containing version control, build server and testing function-
ality along with further components that could not be addressed in the scope of this paper
is depicted in figure 2.

3 Case Study

As an example we present a case study for one of the model regions of WESS (Water
& Earth System Science). The project is concerned with the interdisciplinary research
in the fields of water and solute fluxes at the catchment scale as a function of climate
and land use changes. One of the study areas is the catchment of the river Ammer, a
tributary to the Neckar in south-west Germany. The catchment has a size of 180 km²
and is mainly fed by groundwater from the karstic and fractured aquifers. The Ammer river
has a length of 25 km.

By applying the workflow given in section 2.1 we imported the available input data into
the framework. This includes a DEM, the boundary of the model region, four groundwater
production wells, over 100 boreholes in the region and the river network consisting of the
Ammer itself as well as its tributaries, the Kaesbach and the Kochart (see fig. 3a). All
this information is then included into the surface mesh depicted in fig. 3b. Based on the
borehole data (fig. 3c) and a digitised map of the layering, subsurface layers have been
constructed and incorporated into a 3D subsurface model of the model region (fig. 3d).

For the subsequent simulation of groundwater flow, boundary conditions have been as-
signed to the model (see fig. 4a and 4b) at the three rivers (large dots) and at four
groundwater production wells (blue cylinders). Furthermore, a boundary condition based
on groundwater recharge has been set on the surface of the model region (small dots).
While final results of that simulation are not yet available, a time step from a preliminary
simulation of a steady state model is depicted in figure 4c to demonstrate the basic visu-
alisation principle. In addition, results can be visualised in combination with any of input
data sets or modelling parameters and the visualisation of results can be enhanced by
visualisation techniques such as calculating iso-surfaces of the result (e.g. groundwater
head) or selecting a subset of the result by setting thresholds.
4 CONCLUSIONS

We presented a framework for the generation and visualisation of hydrological models. By combining functionality typically found in geographic information systems, CAD systems and simulation software and providing the required interfaces for data import we allow scientists to easily set up hydrological models with the possibility of subsequent modifications or refinements. This allows the testing of hypotheses and the iterative refinement of model structure and parameters.

While the framework will be continuously extended in the future by adding refined functionality for model generation and data visualisation, we would like to particularly extend the functionality concerning user-guided workflow definition as well as data verification. Based on the case studies where the framework has been applied as well as the data that has been integrated into the corresponding models, we would estimate these as the most beneficial topics for future applications.

Acknowledgements

WESS is supported by a grant from the Ministry of Science, Research and Arts of Baden-Württemberg (AZ Zu 33-721.3-2) and the Helmholtz Center for Environmental Research,
(a) Boundary conditions in 2D  (b) Boundary conditions in 3D  (c) Preliminary result

Figure 4: Visualisation of modelling data: boundary conditions from above (left) and zoomed in (centre), as well as preliminary results (right).

Leipzig (UFZ). The research was also supported by the TERENO initiative (Terrestrial Environmental Observatories: http://www.tereno.net).

REFERENCES


