International Environmental Modelling and Software Society (iEMSs)
2012 International Congress on Environmental Modelling and Software
Managing Resources of a Limited Planet, Sixth Biennial Meeting, Leipzig, Germany
R. Seppelt, A.A. Voinov, S. Lange, D. Bankamp (Eds.)
http://www.iemss.org/society/index.php/iemss-2012-proceedings

Design and implementation of the land surface model NaturalEnvironment within the generic framework OpenDanubia for integrative, distributed environmental modelling

Daniel Waldmann[1,2], Markus Muerth[1], Wolfram Mauser[1]
[1] Dept. of Geography, University of Munich (LMU),Germany
[2] Leibniz Supercomputing Centre (LRZ), Garching, Germany
daniel.waldmann@lrz.de

Abstract: The project GLOWA-Danube (http://www.glowa-danube.de) aimed at investigating the manifold consequences of Global Change on regional water resources in the Upper Danube Basin. In order to achieve this task, an interdisciplinary, university-based network of experts developed the integrative Decision Support System OpenDanubia (OD). The common base for implementing and coupling the various scientific model components is a generic framework, which provides the coordination of the coupled models that run in parallel exchanging iteratively data via their interfaces. The OD framework takes care of technical aspects, such as ordered data exchange between sub-models, data aggregation, data output, model parallelization and data distribution over the network, which means that model developers do not have to be concerned about complexities evolving from coupling their models.
Within this framework the sub-model *NaturalEnvironment*, representing a land surface model, was developed and implemented. The object-oriented design of this sub-model facilitates a plain, logical representation of the actual physical processes simulated by the sub-model. Physical processes to be modelled are organized in naturally ordered, exchangeable lists that are executed on each spatial computation unit for each modelling time step, depending on their land cover. The type of land cover to be simulated on each freely defined spatial unit is distinguished by one of the three types *aquatic*, *terrestrial* and *glacier*. Additionally, the type *terrestrial* is influenced by dynamic land use changes which can be triggered e.g. by the socio-economic OD sub-model *Farming*.
This paper presents the basic design of the open source (GPL'ed) OD framework and highlights the implementation of the sub-model *NaturalEnvironment* within this framework, as well as its interactions with other components included in OD.

***Keywords**: land surface model, framework, object-oriented, open source*

1 INTRODUCTION

1.1 General design of OpenDanubia

The modelling framework OpenDanubia was developed within the project GLOWA-Danube (http://www.glowa-danube.de) and integrates the distributed simulation models of all socio-economical and natural science disciplines taking part in the project. Its purpose is to support the analysis of water-related global change

scenarios in the Upper Danube Basin [Ludwig et al. 2003]. OpenDanubia provides the basis for dynamical coupling of a number of different simulation models and guarantees an ordered sequence of the computation cycles of the sub-models and their data exchange. A detailed description of the framework is beyond the scope of this paper. Nevertheless, in order to supply the reader with the necessary basics, a brief description of OpenDanubia follows. For more details the reader is referred to Hennicker et al. [2010].

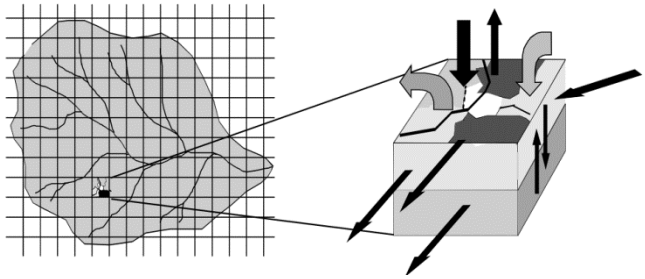

Figure 1: Schematic view of the proxel concept used in OpenDanubia. The arrows exemplary indicate the flows into and out of a proxel.

OpenDanubia is a raster-based model. The catchment to be modelled, in our case the 77,000 km^2 Upper Danube Basin, consists of raster cells, so-called proxels (process pixels). On the basis of such a proxel all processes represented by a sub-model are computed (Figure 1). This means that each sub-model has its own type of proxels, characterised by its own attributes and processes. Within the project GLOWA-Danube the spatial resolution of all sub-models is fixed to 1x1 km^2, while for the computation of sub-scale processes proxels can be divided into so-called subproxels based on land cover fraction. Yet, for all interactions between proxels, subproxel fluxes are aggregated. The temporal computation time step varies between 1 hour and 1 month, depending on the dynamics on the modelled processes. Data exchange between different components (or sub-models) is defined by interfaces (Figure 2). For reasons of a clear, logical structuring, the sub-models are packaged into components (e.g. the *Landsurface* component consists of the *SNT* (Soil Nitrogen Transformation), *Biological* and *NaturalEnvironment* sub-models) and the data exchange may occur on component or sub-model basis. Thus the exchange is encapsulated into more generalized variables, such as soil trafficability (in 5 levels from good to bad) between *Landsurface* and *Actor*s and more detailed variables, for example plant nitrogen uptake per soil layer (in kg m^{-1} s^{-1}) between *Biological* and *SNT*.

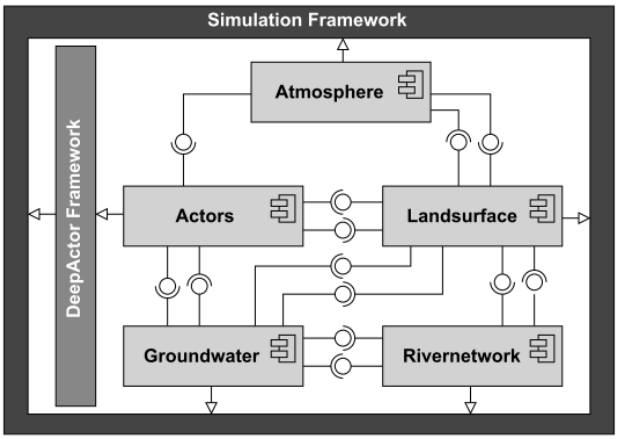

Figure 2: System architecture of OpenDanubia [Hennicker et al. 2010].

1.2 Framework specifications relevant for *NaturalEnvironment*

Figure 3 shows the basic setup of a sub-model with its proxel types within the framework. Attributes commonly used by the sub-models, such as elevation, coordinates or land use, are provided by an abstract proxel representation. In addition to these inherited attributes, more specific properties as well as the algorithms of a model, are defined in the concrete implementation of the proxel type of a sub-model. The invocation of the sub-models' equations to be computed on each proxel as well as the computation order of the sub-models and finally their data exchange are coordinated by the framework.

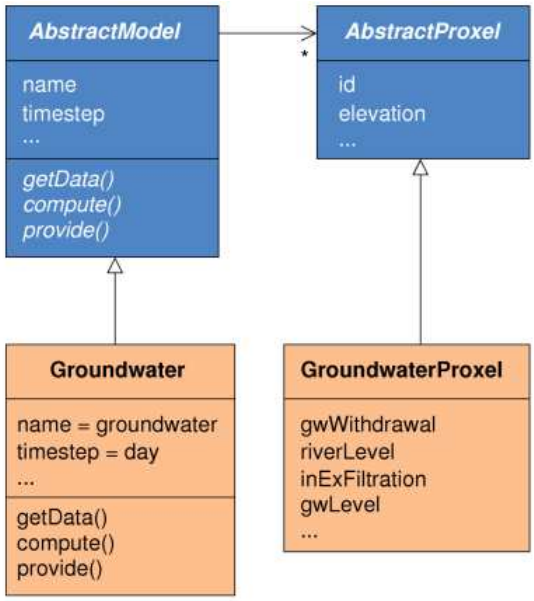

Figure 3: General concept of abstract and concrete implementations of a sub-model and its proxel type within OD.

In order to introduce a new sub-model within the OD framework, plug-points are provided that have to be implemented by the model developer. These guide the control flow of the sub-model(s) and also provide mechanisms for data exchange, thus allowing the model developer to concentrate on the scientific algorithm instead of dealing with such technical issues. In order to retrieve the required data from another sub-model, a model developer requests during the *getData()* step proxel tables provided by some other model. This data can be used in the *compute()* cycle to calculate the outputs of the proxel itself, which in turn *provide()* the necessary input data for further sub-models within the model network (Figure 4).

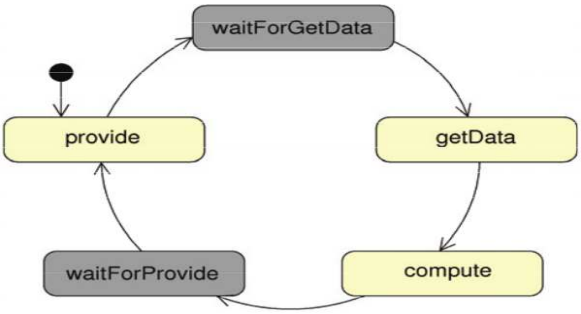

Figure 4: Plug-points of the model cycle

Since the project GLOWA-Danube is also interested in investigating future impacts of land use changes, the framework furthermore provides a basic mechanism to deal with dynamic land use changes. Whilst a spatial resolution of 1 km² is

sufficient for assessing the water cycle on a regional scale for most components, this is not true for all of them. The *Farming* sub-model of the *Actors* component deals with agricultural productivity on the basis of administrative districts. Drawing reliable conclusions from modelled agricultural statistics and estimating produced crop yield on a district base requires to model the cultivated area in more detail than one square kilometre. Furthermore, to evaluate the impacts of future land use changes in the agricultural sector, triggered by climatic as well as changes in the water cycle, it is essential to provide a modelling mechanism for simulating dynamic land use changes at runtime. Therefore the concept of sub-scale modelling was introduced in OD. Land cover changes can be initiated by sub-models during runtime by sending a land use change request to the framework. The framework keeps track of the land cover changes and the respective fractions of each subproxel and notifies the other sub-models within OpenDanubia of the latest changes. Both the subscale modelling approach and the dynamic land use changes are basically included in the framework, yet the concrete implementation of those features has to be realised by the affected sub-models itself.

This paper aims at describing the technical implementation of these features within the scientific land surface sub-model *NaturalEnvironment*. Thus, after a brief introduction in the modelling concept of the sub-model *NaturalEnvironment*, the following paragraphs describe the implementation of these design extensions within *NaturalEnvironment*.

2 ACTUAL SUB-MODEL DESIGN

The sub-model *NaturalEnvironment* summarises matter and energy fluxes occurring on the inanimate land surface layer above ground as well as in the pedosphere layer. This mainly refers to the transfer, storage and balancing of water and radiation/energy to and from those layers (a more detailed view of the processes considered is given in section 2.1, cf. Tab. 1). *NaturalEnvironment* is mostly based on the hydrological land surface model PROMET [Mauser and Bach 2009] including surface and soil energy processes [Muerth and Mauser 2012]. Since the focus of this paper is the description of the modelling concept, the reader is referred to the mentioned paper for further scientific model details.

NaturalEnvironment relies on an intensive data exchange with the other *Landsurface* sub-models of OD, Biological (treating the vegetation) and SNT (simulating soil nitrogen transfer). Furthermore it receives meteorological input data from *Atmosphere*, and interacts with *Groundwater* and *Rivernetwork* (see Figure 2). Of course, there is also an interaction with the *Actor* sub-models, which model the anthropogenic influences in the catchment.

2.1 Proxel types and process lists

According to the abstract proxel concept of the OD framework, the sub-model *NaturalEnvironment* operates on *NaturalEnvironmentProxel*s. Basically such a proxel may contain up to 10 subproxels (as a convention within GLOWA-Danube) of the following types:

- *AquaticSubProxel*: represents water bodies
- *GlacierSubProxel*: represents glaciated areas
- *TerrestrialSubProxel*: represents natural and anthropogenic used areas

In the current implementation OD is capable of modelling 5 different land cover types, such as water (represented by an *AquaticSubProxel*), forest or agriculture (represented by a *TerrestrialSubProxel*). These in turn are separated into general land use types (e.g. arable land and grass land), which are again divided into subtypes like maize or rapeseed.

The three subproxel types are all inherited from the superclass *NaturalEnvironmentSubProxel*, which defines common properties for all subtypes, such as variables for water or energy storage. The subtypes themselves extend the common type according to their specific requirements, e.g. define the thermal conductivities of their ground layers. Furthermore, the actual physical processes to be modelled by *NaturalEnvironment*, the *process lists*, are coupled to certain types of subproxels (cf. Table 1). The common base for all modelled processes (i.e. the process list) is the abstract type *AbstractProcessingUnit*, which defines a general computation cycle processed by each *NaturalEnvironmentSubProxel* (or derived subtype). Developers implementing new modelling processes must derive a new process from this type refining the process by including the model equations into it.

At model initialisation, a user-defined list of the processes to be calculated for each time step of the model run (i.e. the process list) is read. This means, that all of the implemented processes can be enabled or disabled (of course only if their outputs are not directly required by other sub-models), thus providing high flexibility for exchanging predefined algorithms implemented as a processing unit. Table 1 briefly describes the implemented processes and lists their priorities as defined by the process developers. These assigned priorities define the order in which the processes are executed by each subproxel. The calculation process is as follows:

1. The framework calls the *computeProxel* method of the proxel which first calculates common proxel variables and then iterates over all of its currently assigned subproxels, calling their compute method.
2. The *compute* method of the subproxel iterates over its defined process list calling the *compute* method of each *AbstractProcessingUnit.*
3. In the *compute* method of the actual process the scientific algorithms are processed which calculate the new states of the subproxel and update its variables accordingly.

Within the concrete process implementation the process developer has access to all subproxel variables (e.g. current soil water storage) of the concrete subproxel type the process is running on, as well as to the proxel variables, such as precipitation or air pressure. Furthermore the developer can declare local variables within the process, which are private to the process and thus cannot be accidentally modified from other processes. Thereby the physical process and its state is clearly encapsulated and thematically separated.

Table 1: The complete list of processes for all subproxel types in OD.

Process	Priority	SubProxel type	Comment
ResetProperties	1100	All	Used for resetting global proxel variables
VegetationProperties	1050	TerrestrialSubProxel	Sets vegetation related properties, e.g. wind speed in canopy
SurfaceProperties	1000	All	Sets dynamic surface properties depending on land cover
SnowProperties	900	All	Calculates snow properties, e.g. actual snow albedo
SoilProperties	850	TerrestrialSubProxel	Pre-calculates dynamic soil thermal and hydraulic properties
SurfaceWaterDrivers	500	All	Sets water related drivers of land surface processes
VegetationRadiationProcesses	100	TerrestrialSubProxel	Calculates radiation distribution and absorption within canopy
EnergyBalanceProperties	0	All	Sets incoming radiation depending on actual cover
TerrestrialEvaporationProcesses	-40	TerrestrialSubProxel	Calculates evaporation on TerrestrialSubProxels
AquaticEvaporationProcesses	-50	AquaticSubProxel	Calculates evaporation from water bodies and streams
SnowEnergyBalanceProcesses	-54	All	Solves the energy balance on snow covered surfaces
GlacierEnergyBalanceProcesses	-55	GlacierSubProxel	Solves the energy balance on GlacierSubProxels
VegetationEnergyBalanceProcesses	-56	TerrestrialSubProxel	Solves the energy balance on actually vegetated SubProxels
SoilEnergyBalanceProcesses	-57	TerrestrialSubProxel	Solves the energy balance for the soil layers
AquaticEnergyBalanceProcesses	-58	AquaticSubProxel	Solves the energy balance on water bodies
SoilTemperatureProcesses	-60	TerrestrialSubProxel	Computes the temperature of all soil layers
SnowWaterProcesses	-100	All	Determines snow water related variables, such as snow melt
SoilWaterProcesses	-400	TerrestrialSubProxel	Calculates soil water variables like actual soil moisture
GlacierWaterProcesses	-500	GlacierSubProxel	Calculates e.g. ice melt and runoff from glaciers
ErosionProcesses	-800	TerrestrialSubProxel	Calculates erosion on terrestrial proxels
BalanceProcesses	-1000	All	Checks water and energy balance for all subproxel types

2.2 Subscale modelling and dynamic land use changes

In order to provide the possibility to model certain regions (in our case the agricultural areas) in a finer resolution than the standard 1 km², the concept of subscale modelling was introduced in OD. The general concept of subscale modelling is the allocation of an arbitrary number of subproxels within each proxel. Instead of modelling each proxel at coarse resolution, the subproxels are modelled at a custom, finer resolution. To this point, this concept is essentially the same as operating the whole model on a finer resolution, with two crucial exceptions:

(a) the exact locations of the subproxels within the proxel they belong to are not defined, and
(b) the properties of the subproxels may be inherited from the proxel they belong to, instead of defining their own.

Considering (a) this means that lateral flows between subproxels cannot be modelled (since the location, i.e. the computation order of the subproxels is not defined), but only between proxels. Within GLOWA-Danube this fully permissible, since lateral flows only occur in the component *Rivernetwork* which operates on the standard 1 km² resolution and does not recognise subproxels. So since the majority of ODs' sub-models only computes on the proxel level, an upscaling of subproxel states and storages has to be executed by the subscale sub-models before passing the computed results to the regular proxels.
The consequence of (b) is that the actual resolution of all subproxel properties which are inherited from the proxel, is of course limited to the resolution of the proxel. In the context of OD this means, that common proxel properties, e.g. elevation, are initialised with a raster dataset with a 1 km spatial resolution. So these also represent the inherited properties of the subproxel. Therefore, subproxels are only sensitive to the subscale information they are able to store or represent (the land use type and the state and storage changes it introduces), but cannot achieve a higher general spatial model accuracy or reliability.

Considering (a) and (b) together, the subproxel concept provides a handy way for changing land use dynamically at model runtime:

(a) means that subproxels can be removed from or added to a proxel without memorizing locations and (re)calculating such consequences as flow directions
(b) means that an arbitrary number of subproxels of arbitrary size may be generated on a proxel, since only proxel properties and no additional static information is required for defining the subproxel.

The technical implementation of a land use change within NaturalEnvironment works as follows:

1. The land use changing sub-model (in our case *Farming*) triggers the transition from an existing land use to the desired new land use by requesting the change at the framework.
2. The framework keeps track of the land use change by updating the subproxel list which contains the land uses and corresponding areas.
3. Before the actual computations of a *NaturalEnvironmentProxel* occur, the proxel queries the framework if the land use has changed. If this is true the proxel instantiates a new subproxel (if necessary) and initializes its process list and storage variables accordingly. Then it forwards the land use change to the affected *NaturalEnvironmentSubProxel*.
4. The subproxel in turn propagates the change information to its actual process list, i.e. each instance of *AbstractProcessingUnit*.
5. In the concrete implementation of the process, it is up to the process developer to deal with the land use change and update the state and storage variables of the process as well as the affected subproxel properties. This guarantees that the modification is supervised by the expert responsible for the modelling process.

2.3 Model outputs

The outputs of ODs models are spatio-temporal datasets. They can generally be distinguished in two categories:

(a) data stacks: these contain the spatially distributed data of each model output variable for the whole catchment. Optionally the data can be temporally aggregated during runtime before output. Figure 5 shows an example output of annual evapotranspiration.

(b) point lists: arbitrary model variables can be output as lists on a proxel or subproxel basis.

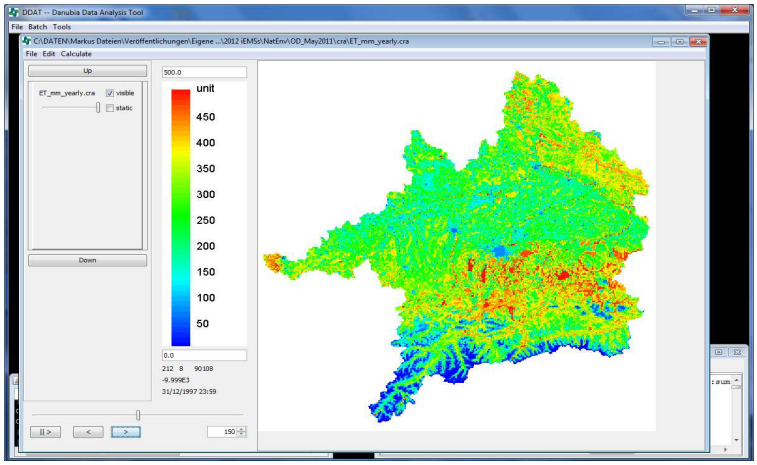

Figure 5: Annual ET sum of the whole catchment for an arbitrary year (visualized with the viewer and analysis tool that is also freely available from www.opendanubia.glowa-danube.de).

An example of outputs of a proxel with different agricultural land uses is presented in Figure 6. The evapotranspiration (ET) of two different land uses (A and B) is calculated in millimetres per subproxel land cover. The actual proxel ET is then the area weighted average of both subproxel ET values. During late April land use A is removed and immediately afterwards land use C is added, yet with a larger fraction than A had before. The effect on the total (proxel) ET is immediately obvious one day later. It has to be noted that depending on the actual timing of land use changes, vegetation parameters are taken from predefined tables for initialisation. In general, plants are dynamically resolved in the *Biological* sub-model.

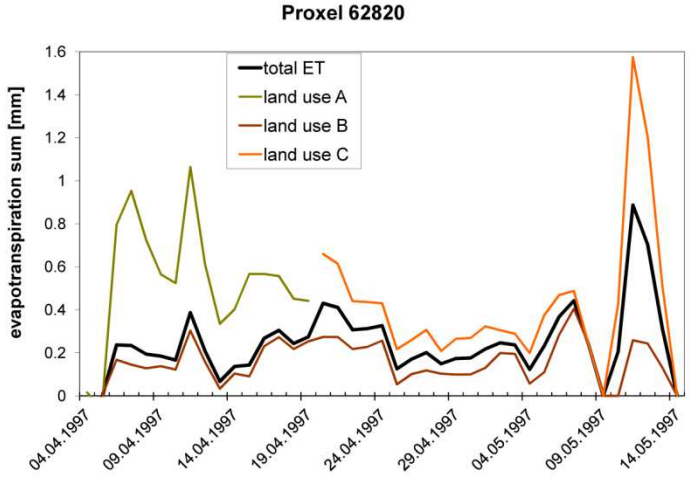

Figure 6: Daily ET of subproxel land use types and their weighted proxel average.

4 CONCLUSION

OD provides a scalable, flexible modelling framework. Its sub-models can easily be distributed within existing computing infrastructures. Depending on the number of sub-models employed, these can be executed on desktop machines, a cluster or in a cloud. OD simplifies the development of new models especially in larger project groups, since its intense usage of interfaces for data exchange allows for clear delineation of scientific competences. Thus, the algorithms of specific sub-models are clearly separated and exact descriptions of the data exchanged can be defined at the interface level. This facilitates avoiding misconceptions regarding data exchange and prevents illegal access to private variables of the sub-models.

The object-oriented design of *NaturalEnvironment*, especially the utilisation of process lists for encapsulating scientific algorithms makes it a valuable tool for development and testing of new model algorithms. The abstract model concept with its automated calling of plug-points allows scientists to fully concentrate on the scientific algorithms instead of dealing with administrative issues like control flow.
The processes implemented in *NaturalEnvironment* can be easily switched on and off for different model runs, which makes it easy to compare the impact of certain equations on the results. This can be used for improvement of overall model performance as well as for educational purposes and community model building.

NaturalEnvironment can be considered to be highly portable to other regions than the Upper Danube basin, since (a) it mainly uses physically-oriented scientific equations instead of region-specific empirical formulae and (b) the processes encapsulating the algorithms can be easily exchanged or replaced. When porting to other (larger) regions, the concept of subscale modelling allows for an optimal tuning between computation time and spatial resolution, since proxels may be defined generally with a coarse resolution, but the resolution can be increased in areas where required by using sub-proxels. Since the sub-proxels are not implemented statically, it is possible to react on changing (environmental) conditions during runtime and modify the land use on the subproxels or add and remove those arbitrarily. This is an essential functionality for reliably modelling future changes in the water cycle.

ACKNOWLEDGMENTS

GLOWA-Danube was funded by the German Ministry of Education and Research (BMBF), the State of Bavaria and the State of Baden-Württemberg. Special thanks for the collaborative work go to Tim Reichenau, Roland Barthel and Tatjana Krimly.

REFERENCES

Hennicker, R., S. Bauer, S. Janisch, and M. Ludwig, A Generic Framework for Multi-Disciplinary Environmental Modelling, 2010 International Congress on Environmental Modelling and Software, Ottawa, Canada, 2010.

Ludwig, R., W. Mauser, S. Niemeyer, A. Colgan, R. Stolz, H. Escher-Vetter, M. Kuhn, M. Reichstein, J. Tenhunen, A. Kraus, M. Ludwig, M. Barth and R. Hennicker, Web-based modeling of water, energy and matter fluxes to support decision making in mesoscale catchments - the integrative perspective of GLOWA-Danube, *Physics and Chemistry of the Earth*, 28, 621-634, 2003.

Mauser, W. and H. Bach, PROMET - Large scale distributed hydrological modelling to study the impact of climate change on the water flows of mountain watersheds, *Journal of Hydrology*, 376, 362-377, 2009.

Muerth, M. and W. Mauser, Rigorous evaluation of a soil heat transfer model for mesoscale climate change impact studies, *Environmental Modelling and Software*, 35, 149-162, 2012.