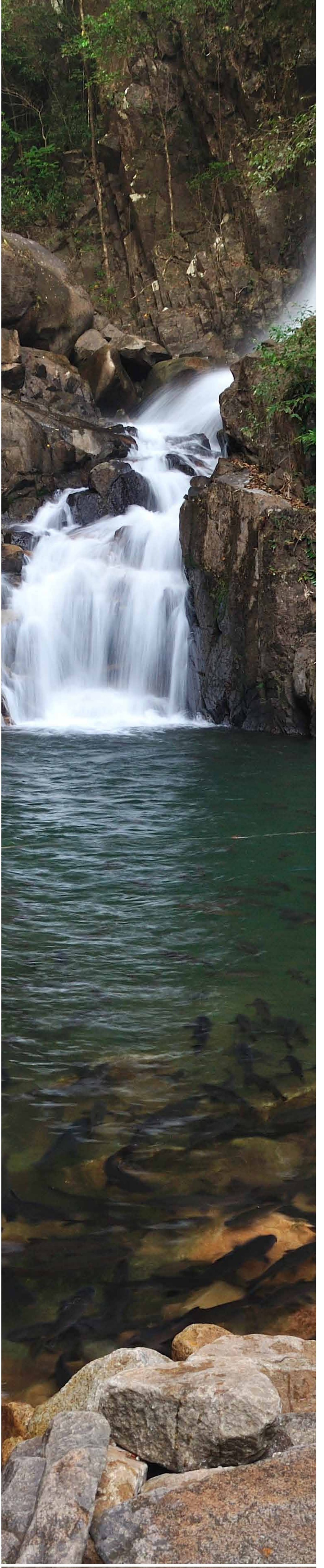
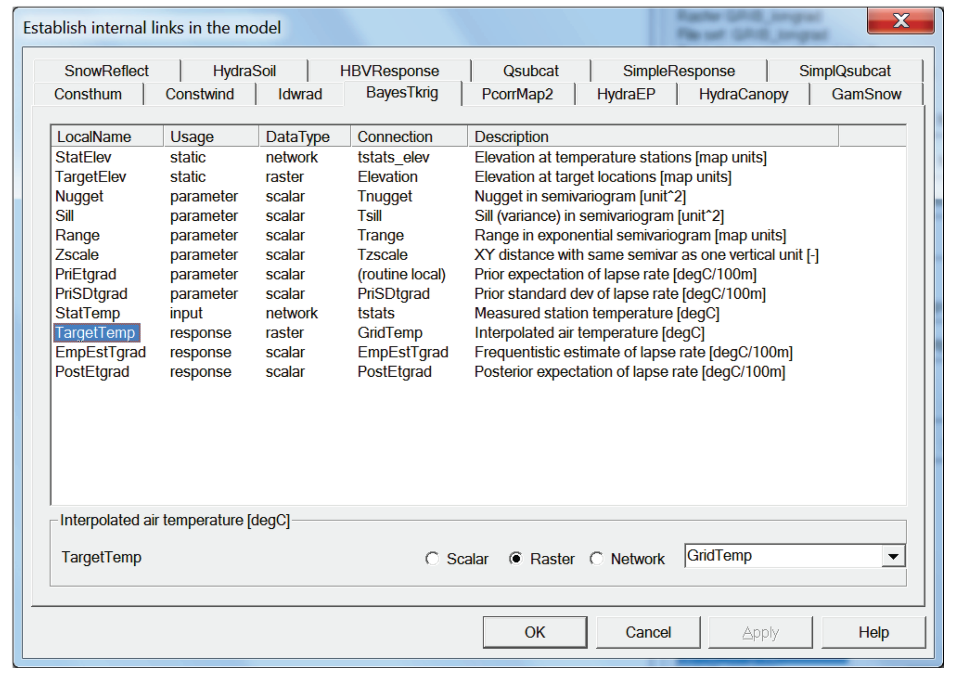
The Open Source model framework ENKI



What is ENKI?

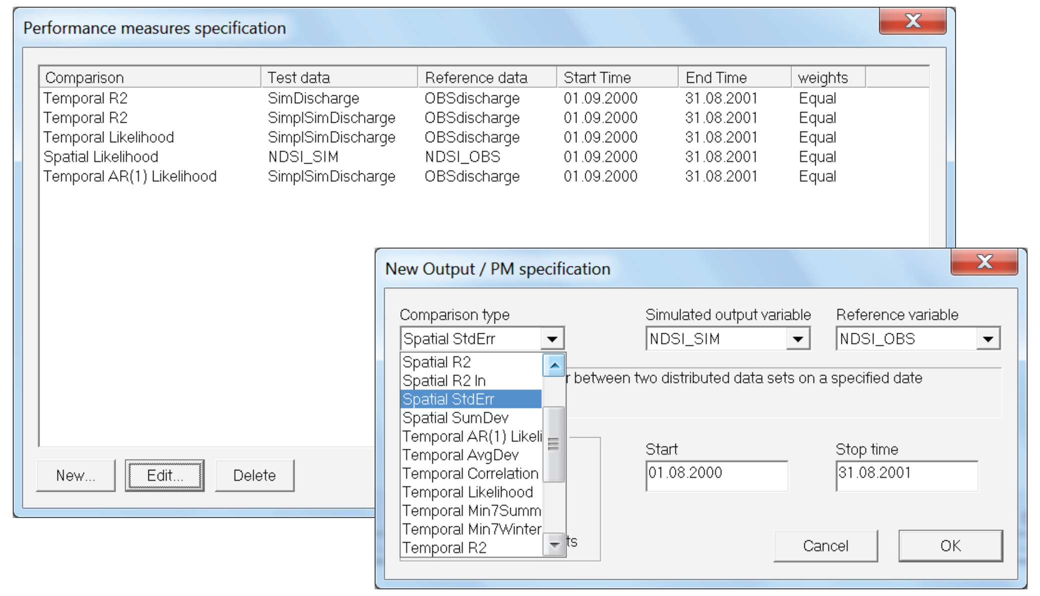
ENKI is a modular framework for implementing hydrological or other environmental models. Both lumped and distributed models are supported. ENKI builds a model from a set of user-defined subroutines, which operate on GIS data within a spatial region.

Also providing calibration and evaluation functionality, ENKI makes it easy for model developers to implement and test single routines and various model compositions in a fixed framework. ENKI is now released as open source under GNU LGPL.



The ENKI framework recognises the number, types, and names of each subroutine variable. The framework then exposes the variables to the user within the proper context, ensuring that:

- The model is completely and consistently set up
- Distributed maps coincide spatially where necessary
- Time series exist for input variables
- State variables are initialised for the correct date/time
- GIS data sets exist for static map data



Contacts:
Sjur Kolberg, SINTEF
sjur.kolberg@sintef.no
Oddbjørn Bruland, Statkraft
oddbjorn.bruland@statkraft.no

SINTEF

Statkraft

Who is ENKI for?

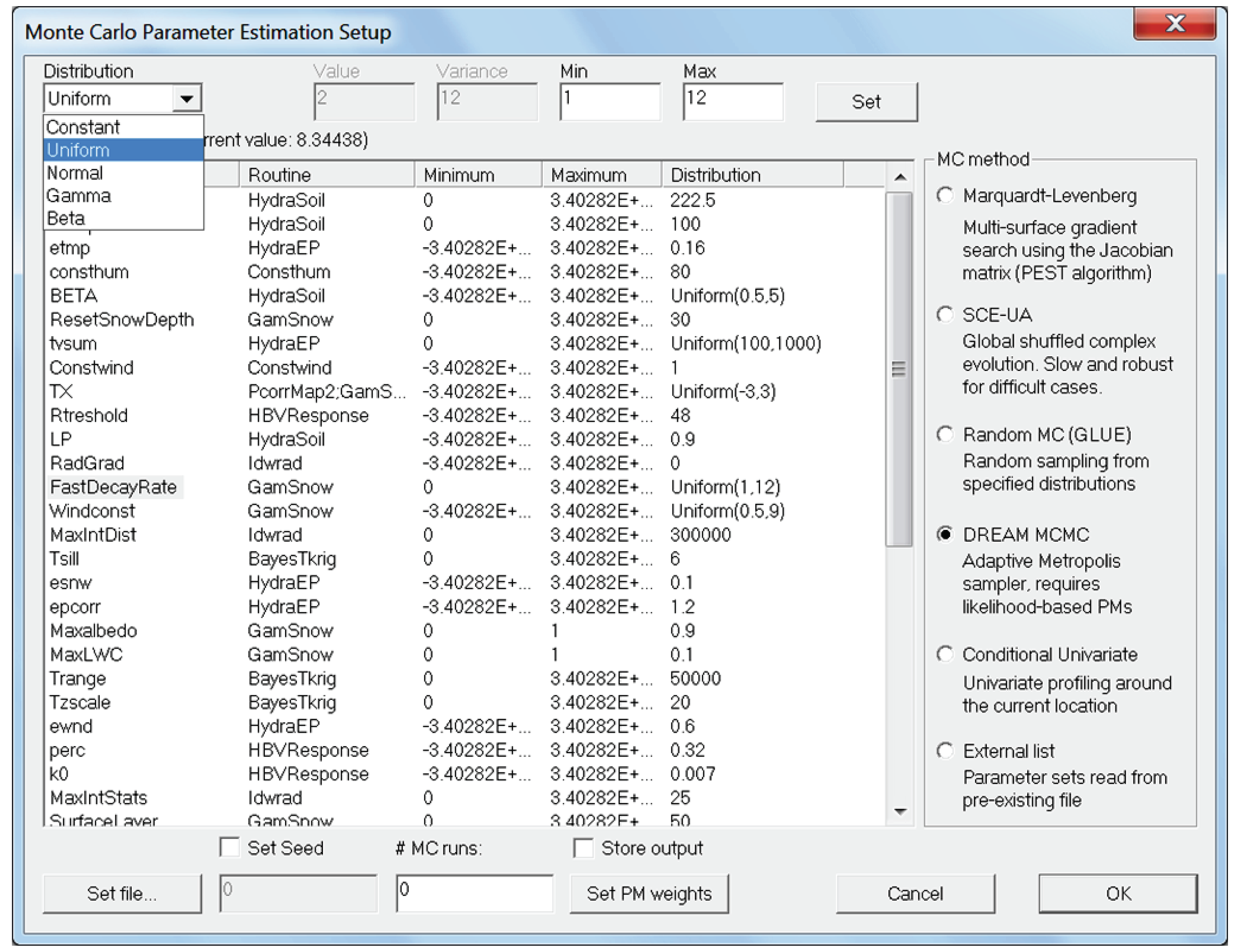
ENKI offers functionality for three different levels of involvement in model construction:

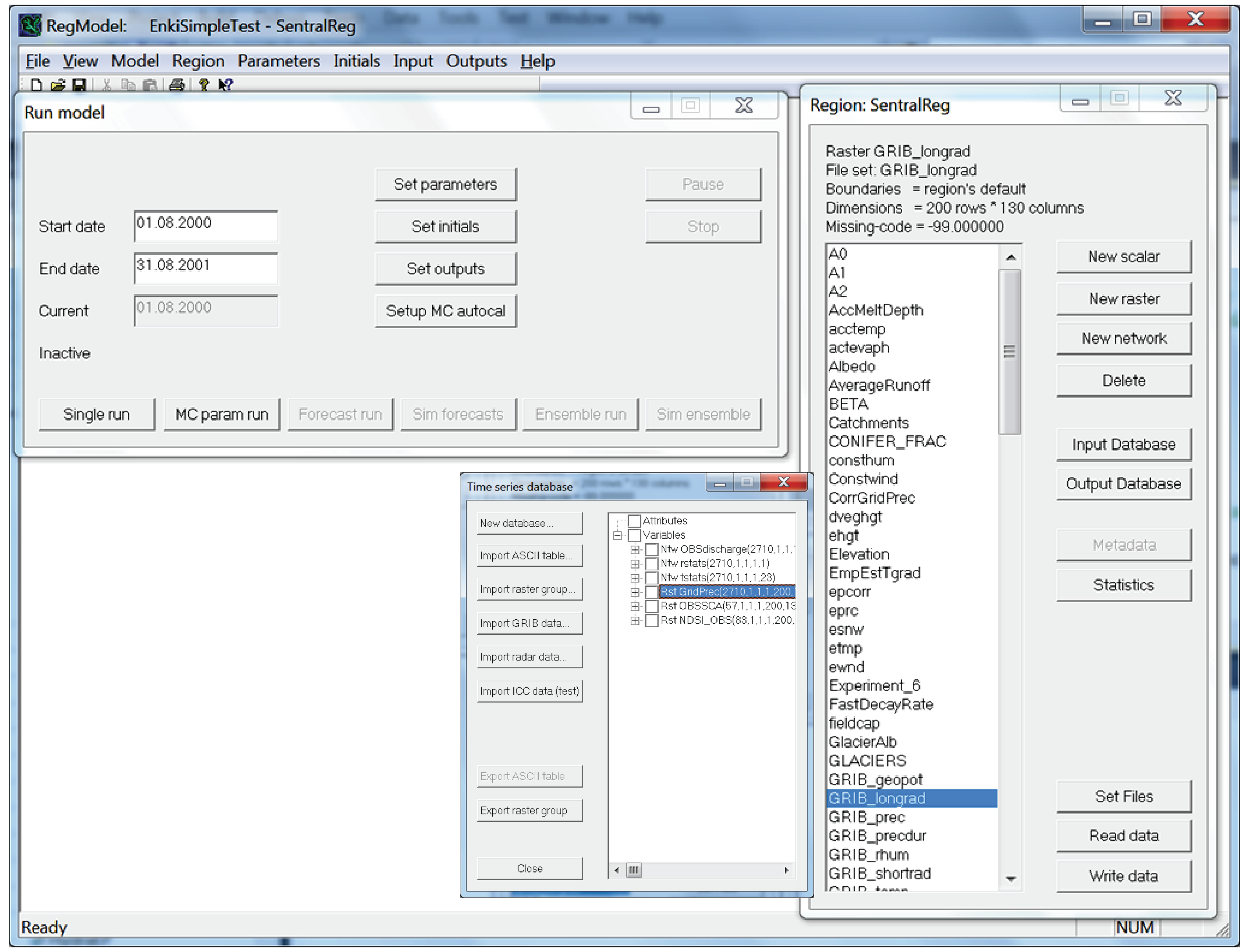
Model application:
Run and evaluate pre-built models for any response using several objective functions, choose search algorithm for calibration, and analyse uncertainty arising from input errors or parameter equifinality.

Model analysis: Add or replace subroutines, run multi-model ensembles, switch between calibrated and mapped parameters, and experiment with different distribution schemes without having to write or compile source code.

Routine implementation and testing: Code the core of a new lumped or distributed subroutine, include it in an ENKI model, and let ENKI handle all model administration and interface code.

As ENKI continues to develop as a experimental tool, its core API is also being implemented in Statkraft's forecast system for operational hydropower. This common core, the modular design, and the open source license all facilitate rapid dissemination of new methods into operational use.





Why distributed models?

Mountainous areas exhibit strong gradients in meteorology, topography and land surface properties.

For nonlinear processes, the use of catchment averages in model equations lead to biased results. For catchment sizes of 10^2-10^3 km^2, model errors depend more on heterogeneity and uncertainty in input data, than on inadequate model equations. Spatial distribution allows different response from various parts of the catchments, and emphasises interpolation and downscaling of input data.

Why a regional calibration approach?

Parameter equifinality and poor runoff data in regulated basins encourage the use of several series to reduce the information deficit.

A set of gauged basins is seen as a sample representing the region, enabling estimation of uncertainty also for the ungauged parts.

In Norwegian mountains, regional calibration reduces Nash-Sutcliffe values by 0.05-0.07 compared to catchment specific calibration. Sensitivity analyses emphasise the meteorology-related parameters as the most important.

Operationally, it is easier to maintain a common model for several sub-basins, than to calibrate, feed and update a model for each reservoir. Forecasts are increasingly needed for arbitrary spatial domains; stream intakes, electricity market regions, or river sections with legal flow requirements.

Technology

ENKI is written in C++, and uses a plug-in structure to invoke the subroutines. These are built separately as dynamic-link libraries (DLLs). All subroutines are coded as sub-classes of a generic method class, which is known by the ENKI framework. The subroutine programmer can rely on a few routines being called in specific situations:

- **The constructor** is called when the user includes the method in a model, and informs the ENKI framework about the routine's variable interface.
- **Init**() is called when the model is linked to a specific region, and all the routine's variables are linked to GIS data objects with known spatial extent. Optional.
- **PreProcess**() is called when all parameter values are set, thus for each iteration during auto-calibration. Optional.
- **Respond**() is called for each time step, and implement the process equations.
- **Calc**() may replace Respond() when the routine is purely vertical.

Vertical routines implementing Calc() rely on the framework for spatial looping, and can be used in lumped or distributed models without adaptation. Other routines may combine variables with different geometry.

A minimal ENKI subroutine
The source code of an ENKI routine is highly compact. Below is the source code for a linear reservoir, admittedly with the simplest possible numeric solution.

```
LinearTank::LinearTank()   // Constructor, defining the variable interface
{
   outflow = newmethvar("outflow",true,"methvar","response","Outflow from linear tank");
   storconst = newmethvar("storconst", true,"methvar","parameter","Response tank time constant",0,1);
   inflow = newmethvar("inflow", true,"methvar","input","Inflow to linear tank");
   storage = newmethvar("storage", true,"methvar","state","Response tank storage in mm",0);
}

bool LinearTank::Calc()    // Response function, representing a single linear tank
{
   storage->m_value += inflow->m_value;
   outflow->m_value = storage->m_value * storconst->m_value;
   storage->m_value -= outflow->m_value;
   return true;
}
```

State of software

ENKI is now developing both as a research tool and as a simulation engine for an operational forecast system. Currently it builds under Windows and Visual Studio; efforts to remove these platform and compiler dependency has started.

Recent modifications include a full separation of API and user interface, making it possible to run ENKI from GIS programs and other software environments.

Source code and binaries are available from the authors, released under the GNU LPGL license.