Proceedings
of the 4th International Modelica Conference,
Hamburg, March 7-8, 2005,
Gerhard Schmitz (editor)

K. Berg, K. Nyström
Linköping University, Sweden
Hydrological modeling in Modelica
pp. 149-154

Paper presented at the 4th International Modelica Conference, March 7-8, 2005,
Hamburg University of Technology, Hamburg-Harburg, Germany,
organized by The Modelica Association and the Department of Thermodynamics, Hamburg University of Technology

Hydrological modeling in Modelica

Karin Berg Kaj Nyström
Linköping University
Linköping, Sweden
kb@lysator.liu.se kajny@ida.liu.se

Abstract

Hydrological modeling is an area where modeling has been used for a very long time. Applications range from forecasts for the hydro power industry, public safety, agriculture and environmental monitoring. Still, to the best of our knowledge, Modelica has been very little used in hydrological modeling so far. In this paper, we aim to show that the Modelica language is well suited for hydrological modeling and also to outline a possible future development of libraries in order to further facilitate hydrological modeling and coupling of hydrological models to other types of models in Modelica.

A Modelica implementation of the hydrological HBV model is compared with the original Fortran model. The main advantages of using Modelica as modeling language are more readable and re-usable code and better abstraction. The disadvantage is longer execution times compared to the Fortran model.

The HBV model is a quite simple model mathematically. It would be useful to investigate the behaviour of more complex hydrological models as well in order to see whether we can find the same advantages of using Modelica as modeling language in that respect as we have in the case with the HBV model.

Keywords: hydrology, modeling, HBV, Modelica, runoff simulation

1 Introduction

Even though specialised modeling languages have matured over the years, Modelica perhaps being one of the best examples, most hydrological models are still written in Fortran. This, we believe, hampers the development of hydrological modeling. Especially since considerable knowledge of computer science in general and Fortran programming in particular is not something that every hydrologist possess. Clearly, hydrological modeling needs better tools in order to facilitate future model development. The goal of this paper is to present Modelica as an alternative modeling language for hydrological applications and to investigate if a direct translation to Modelica of a hydrological model is actually easier to understand than the corresponding Fortran model.

Hydrological models used today have a wide range of applications including decision support for different business purposes, for example energy trading and farming, hydrological forecasts and warnings and other public safety applications. In addition, they are also often coupled with for example climatological, meteorological, chemical and/or biological models.

As an example of a hydrological model we have chosen a version of the HBV model [4]. HBV is a widely used model throughout the world primarily for hydrological forecasting and runoff simulation, for example as a tool when designing dams for the hydro power industry, but the HBV model has been applied to many other areas as well. Among recent applications are modeling and simulation of nutrient transport in large catchments and simulations of the potential effects of climate change on water flows, water quality, particle transport and biochemical processes in the water.

2 Model description

The HBV model was developed at the Swedish Meteorological and Hydrological Institute (SMHI) in the 1970's [4]. HBV/PULSE, which is used in this study, is a similar model developed from the HBV model in the 1980's as a consequence of the need to study acidification and substance transport. The two models have very similar structure but the HBV/PULSE model is slightly less complex when only hydrology is simulated. HBV and similar models are often described as semi-distributed conceptual models. They have some spatial resolution since they handle systems of catchments, lakes and rivers, but within catchments the spatial resolution is limited.

It is conceptual in the sense that it does not use detailed physical laws of nature in the calculations but rather simple equations which are consistent with the current hydrological knowledge. The model consists mainly of three parts: snow, soil moisture and runoff response functions. These steps are calculated for each land use type in every catchment and there is also possibility to divide the catchments further into sub-catchments. In the HBV model, but not in the HBV/PULSE model, it is also possible to divide the catchments and sub-catchments into altitude zones.

There are also some routines for weighting and correction of input data and evaluation of model performance. Output data from the forest, field and lake areas is weighted The present model is based on an HBV/PULSE model [2] used for research purposes.

Driving variables in the model are daily temperature and precipitation measurements and monthly averages of potential evapotranspiration. The number of parameters in the model vary slightly depending on the current version and application area. The model used in this project has 34 parameters. Both driving variables and parameters are stored in text files.

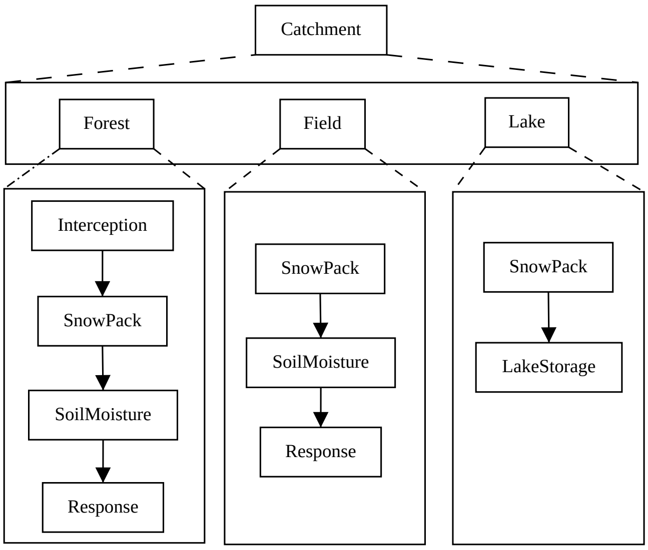

Figure 1: Component hierarchy in the Modelica implementation of the HBV model: Catchments consists of several land use components which in turn consists of different basic HBV components.

3 Model implementation in Modelica

Existing libraries such as Fluid, Species, WasteWater [5], HylibLight (light version of HyLib [1]), QSSFluidFlow [3] were considered for the implementation of HBV, but we found them not very well suited for this application. Since the existing hydraulics libraries are developed for different engineering purposes they include for example pressure, conductance, geometry of pipes and vessels and other parameters which are not known and/or not meaningful to use on the scale on which the HBV model is operating.

Instead, we have created some very simple general components for water storage and transport where the only variable is the water volume itself. There are both some general base classes and HBV specific components and file reading functions.

The HBV model components can be divided into three levels as shown in figure 1. It was possible to transfer some parts of the model to equation form but the main part is written as algorithm statements due to special cases such as if-statements without any else-clause or with different number of assignments in different parts of the statement.

3.1 General components and interfaces

In hydrological applications, water volume is sometimes expressed in the unit mm. Water in the HBV model is entirely expressed in mm and conversion to other units is only performed when all calculations have been made. This may seem strange but is a consequence of precipitation measurements in dm^3/m^2, which can be reduced to simply mm. Because of this, a HydrologyVol unit, connectors and storages for volumes expressed in mm have been created (figure 2). A snow storage was also created, which stores water in both frozen and liquid form. Connectors, icons and variables for stored volumes are declared in the general components.

Discrete variables have been used since the first aim was to make a quick translation of the HBV model from Fortran to Modelica and this was most easily accomplished using discrete variables rather than continuous.

Finally, there are base classes for sources and an infinite sink model. The sink model is not storing anything at the moment, but it can easily be modified to do so. Sources are available with outlets of one or two types. In this application they are used for sources with both snow and water outlets, providing input for the snow storages described above.

```
package Units
  type HydrologyVol =
    Real(final quantity="HydrologyVol",
         final unit="mm");
end Units;
package Interface
  connector HydrologyFlow
    import Hydrology;
    Hydrology.Units.HydrologyVol q;
  end HydrologyFlow;
  connector QoutD =
         discrete output HydrologyFlow;
  connector QinD =
         discrete input HydrologyFlow;
end Interface;
package Components
  partial model Storage
    import Hydrology.Units.HydrologyVol;
    import Hydrology.Interface.QinD;
    import Hydrology.Interface.QoutD;
    discrete HydrologyVol w(min=0);
    QinD qIn1;
    QoutD qOut1;
  end Storage;
  partial model SnowStorage
    import Hydrology.Units.HydrologyVol;
    import Hydrology.Interface.QinD;
    import Hydrology.Interface.QoutD;
    Interface.QoutD qOut1;
    Interface.QinD qIn1;
    Interface.QinD qIn2;
    discrete HydrologyVol s(min=0);
    discrete HydrologyVol w(min=0);
  end SnowStorage;
end Components;
```

Figure 2: Unit declaration and examples of connectors and storages using mm as volume unit

3.2 HBV basic components

Most of the HBV model equations are found in the basic HBV components Interception, SnowPack, SoilMoisture, Response and LakeStorage. One difference versus the Fortran model is that Interception and SnowPack are separated instead of treated together in a common snow routine. This makes the model structure more clear and it also made it possible to use fewer parameters. All basic components except Response are implemented using algorithm statements. The Response model is much smaller than the others and there were no problems with expressing it entirely in equation form (figure 3).

There are limitations on storage of water and snow intercepted in for example trees, since the trees only can hold a limited amount of water or snow. There is also a limitation on relative water content in the snow, but SoilMoisture and Response have no such limits.

```
model Response
  import Hydrology;
  extends Hydrology.Components.Storage;
  outer parameter Real k;
  outer parameter Real alfa;
equation
 when sample(0,1) then
    w = pre(w) + qIn1.q - qOut1.q;
    qOut1.q = 0.001*k*w^(1 + alfa);
 end when;
end Response;
```

Figure 3: Modelica code for the HBV response function

3.3 HBV land use components

Features common for the three types of land use class in HBV - the icon type and a snow pack variable which need to be accessed from several basic components - are declared in the partial class LandUse. Forest, Field and Lake are extensions of LandUse. Forest is the land use model which contains most parts and equations. It consists of four model parts: Interception, SnowPack, SoilMoisture and Response, and also one sink for containing evaporated water. Field has the same structure as Forest but without Interception since field is defined in the HBV model as land area with no interception. Lake models consists of only two parts - SnowPack and LakeStorage. Contributions from the different land use types are added and given a weight proportional to their area. Calculations for the land use components are only performed if the area is greater than zero.

3.4 HBV catchment component

The largest component in the HBV model is the catchment. Each Catchment consists of one precipitation source P and the three land use components described above. As mentioned in the previous section, there is also a component for adding and weighting the outflow from the three land use components. Since flows are weighted based on the relative area of the land use class, this type of catchment component can be used even if the catchment for example has no lake or consists only of one big lake.

Reading of driving variables and parameters is accomplished using external C functions which are called

from Catchment. Parameters in the model can also be altered by the user between simulations. In order to manage the parameters in an efficient way, all parameters are stored as inner parameters in Catchment. All models inside Catchment consequently have the parameters which are common for the whole catchment, which is almost all parameters, declared as outer.
Daily temperature and monthly maximum evapotranspiration are calculated in the Catchment component and accessed as outer parameters from the catchment parts. Precipitation has its own component P, a source with three snow outlets and three rain outlets which provides input to the land use components. Catchment also keeps track of which day and month it is in order to assure that the right data is delivered to the other model components.

4 Simulation

When setting up an HBV model for a specific area, Catchments are declared with the appropriate parameter files, PTQW files (driving variables and measured flow and lake water levels) and initial values. They are connected to each other directly or through AddFlows components depending on the geography of the area. The test simulation setup in this study consisted of two catchments connected as shown in figure 4. Since the test areas are small (0.44 and 0.43 km^2 respectively), the AddFlows component adds the two outflows without any weighting or distribution of the flows over time. The model was run for 1475 time steps, which corresponds to a time period of a little more than four years. An Euler solver with fixed step was used since calculations only need to be performed once every time step.
A rough estimation of execution times was made in order to make sure that the Modelica model performed reasonably well compared to the Fortran version. Without any optimisations, the Modelica model has approximately five to ten times longer execution time than the Fortran model. The model described in this paper does not have very long execution time in either case, so in this application it is not the most important factor to consider when choosing modeling language.
The added outflow from the two catchments, Qout (figure 5), was compared with the corresponding output from the Fortran model. Qout is the runoff response to precipitation which is used for hydrological forecasts and warnings. Some other important model variables, for example the soil moisture storage which is used for estimating the risk of forest fires, were also compared with Fortran results. The comparisons show that the Modelica model produces the same results as the Fortran version.

5 Experiences and conclusions

The translation from sequential Fortran code to an object oriented approach was rather easy but it was difficult to also translate the code to equations. The greatest use for Modelica is probably in designing new models and making additions to existing models. Some restructuring of the model was done in this project but more needs to be done in order to take full advantage of the Modelica language.
The main benefit of translating the model into Modelica is more readable and reusable code, which facilitates future model development. Modelica has proven quite easy to work with for a non computer scientist with some background in programming. Debugging, though, could be improved since many of the errors that can be encountered are radically different from those encountered in ’normal’ imperative programming in for example C.
The test example in this study was quite small, with only two catchments. In most cases the model is run over a larger number of catchment. It would therefore be useful to test the HBV implementation on a larger scale as well.
Since hydrological modeling is performed with many different techniques, further development of general hydrological base classes and components is needed including components for for example physically based modeling. It would also be useful to investigate the behaviour of more complex hydrological models as well in order to see whether we can find the same advantages of using Modelica as modeling language in that respect as we have in the case with the HBV model.

```
model HBV
  import Hydrology;
  Hydrology.HBV.Components.SimpleCatchment SimpleCatchment1
      (Forest1(
       SnowPack1(s(start=0.0),w(start=0.0)),
       SoilMoisture1(w(start=100.0)),
       Response1(w(start=10.0))),
      Field1(
       SnowPack1(s(start=0.0),w(start=0.0)),
       SoilMoisture1(w(start=100.0)),
       Response1(w(start=10.0))));
  Hydrology.HBV.Components.StartCatchment StartCatchment1
     (Forest1(
       SnowPack1(s(start=0.0),w(start=0.0)),
       SoilMoisture1(w(start=55.0)),
       Response1(w(start=10.0))),
      Field1(
       SnowPack1(s(start=0.0),w(start=0.0)),
       SoilMoisture1(w(start=55.0)),
       Response1(w(start=10.0))));
  Hydrology.Components.TestSource TestSource1;
  Hydrology.Components.Sink Sink1;
  Hydrology.HBV.Components.Add2Flows Add2Flows1;
equation
  connect(TestSource1.qOut1, SimpleCatchment1.qIn1);
  connect(Add2Flows1.qOut1, Sink1.qIn1);
  connect(SimpleCatchment1.qOut1, Add2Flows1.qIn2);
  connect(StartCatchment1.qOut1, Add2Flows1.qIn1);
end HBV;
```

Figure 4: Modelica code for declaration and connection of the HBV model parts

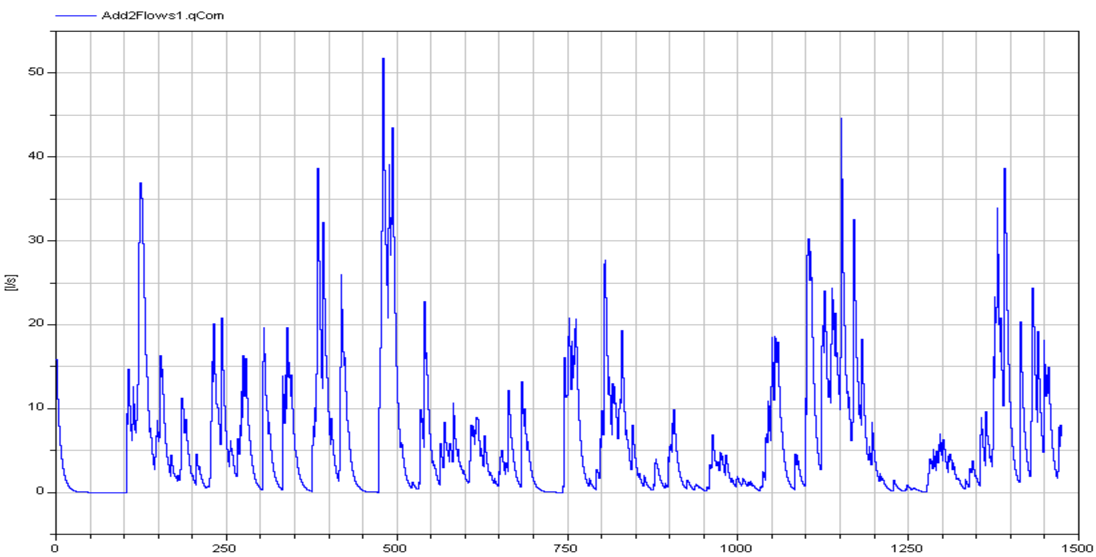

Figure 5: HBV runoff response simulated in Modelica

References

[1] P. Beater. Modeling and digital simulation of hydraulic systems in design and engineering. education using modelica and hylib. In M. Otter, editor, *Modelica workshop 2000 proceedings*, pages 33–44. The Modelica Association, 2000.

[2] B. Carlsson, S. Bergström, M. Brandt, and G. Lindström. PULS-modellen - struktur och tillämpningar (the PULSE model - structure and applications, in Swedish), 1987. SMHI Reports Hydrology No.8.

[3] S.M.O. Fabricius and E. Badreddin. Modelica library for hybrid simulation of mass flow in process plants. In The Modelica Association, editor, *2nd International Modelica Conference proceedings*, pages 225–234. The Modelica Association, 2002.

[4] G. Lindström, B. Johansson, M. Persson, M. Gardelin, and S. Bergström. Development and test of the distributed hbv-96 hydrological model. *Journal of Hydrology*, 201:272–288, 1997.

[5] G. Reichl. Wastewater a library for modelling and simulation of wastewater treatment plants in Modelica. In P. Fritzson, editor, *Proceedings of the 3rd International Modelica Conference*, pages 171–178. The Modelica Association, 2003.