Deltares
Enabling Delta Life

Adding HEC-HMS models to FEWS

Adding HEC-HMS models to FEWS

Jan Verkade
Jan Talsma

1202849-000

Title
Adding HEC-HMS models to FEWS

Client	**Project**	**Reference**	**Pages**
Eastern Nile Technical Regional Office	1202849-000	1202849-000-ZWS-0038-	27

Keywords
HEC-HMS, FEWS

Summary

This manual describes how an existing HEC-HMS schematisation can be transferred to the FEWS environment to be used in real-time forecasting.

References
Place references here

Version	Date	Author	Initials	Review	Initials	Approval	Initials
1.0	07dec2010	Jan Verkade		Willem van Verseveld		Jaap Kwadijk	
2.0	08 Oct2013	Jan Talsma					
2.1	07Nov2013	Jan Talsma					

State
draft
This is a draft report, intended for discussion purposes only. No part of this report may be relied upon by either principals or third parties.

Contents

Appendices

1 Introduction

This manual describes how an existing HEC-HSM model can be added to a FEWS configuration. It is assumed that the HEC-HMS model runs in the HEC-HMS GUI without any errors. From there the several steps to include the model in FEWS are explained.

The manual starts with an overview of the HEC-HMS environment.

The latest version of this document can be found on the Deltares FEWS documentation wiki pages: http://public.deltares.nl/display/FEWSDOC

2 HEC-HMS environment

2.1 Description of Hydrologic Modeling System[1]

The Hydrologic Modeling System (HEC-HMS) is designed to simulate the precipitation-runoff processes of dendritic watershed systems. It is designed to be applicable in a wide range of geographic areas for solving the widest possible range of problems. This includes large river basin water supply and flood hydrology, and small urban or natural watershed runoff. Hydrographs produced by the program are used directly or in conjunction with other software for studies of water availability, urban drainage, flow forecasting, future urbanization impact, reservoir spillway design, flood damage reduction, floodplain regulation, and systems operation.

The program is a generalized modeling system capable of representing many different watersheds. A model of the watershed is constructed by separating the hydrologic cycle into manageable pieces and constructing boundaries around the watershed of interest. Any mass or energy flux in the cycle can then be represented with a mathematical model. In most cases, several model choices are available for representing each flux. Each mathematical model included in the program is suitable in different environments and under different conditions. Making the correct choice requires knowledge of the watershed, the goals of the hydrologic study, and engineering judgment.

The program features a completely integrated work environment including a database, data entry utilities, computation engine, and results reporting tools. A graphical user interface allows the seamless movement between the different parts of the program. Program functionality and appearance are the same across all supported platforms.

Excellent program documentation is provided by HEC at the HEC-HMS website: http://www.hec.usace.army.mil/software/hec-hms/documentation.html

2.2 HEC-HMS program

The HEC-HMS program and documentation may be downloaded from the HEC website: http://www.hec.usace.army.mil/software/hec-hms/. After installation, the program offers a user interface in which a model schematisation may be created.

2.3 HEC-HMS schematisation files

The HEC-HMS schematisation and model management files for a simple model may include:

```
|   Control1_update.control
|   Control2_forecast.control
|   Run1_historical.log
|   Run2_forecast.log
|   UpperBlueNile.access
|   UpperBlueNile.basin
|   UpperBlueNile.dsc
|   UpperBlueNile.dss
|   UpperBlueNile.gage
|   UpperBlueNile.hms
|   UpperBlueNile.log
```

[1] *This paragraph is taken from http://www.hec.usace.army.mil/software/hec-hms/*

```
|   UpperBlueNile.out
|   UpperBlueNile.run
|   UpperBlueNile_Meteo.met
|   UpperBlueNIle.pdata
|
+---basinStates
|       UpperBlueNile.stateIndex
|       UpperBlueNile_hist_state.state
|
+---maps
\---optimizer
```

2.3.1 .hms file

This file is the main project file, the one to be opened via the HEC_HMS GUI interface. It lists the components of which the model schematisation and model management consist. The list contains:

- Project, definition of the main DSS file linked to the model (where all information is stored if not differently specified);
- Precipitation, definition of internal and external meteorological inputs or settings per basin (definition of external gauges, monthly climatological evaporation, weights for spatial distribution of rain gauges etc…);
- Basin, definition of the schematisation of the basins;
- Control (might be more than one), definition of the different control files;

```
Project: UpperBlueNile
     Description:
     Version: 3.4
     DSS File Name: UpperBlueNile.dss
End:

Precipitation: UpperBlueNile_Meteo
     FileName: UpperBlueNile_Meteo.met
     Description:
End:

Basin: UpperBlueNile
     FileName: UpperBlueNile.basin
     Description:
End:

Control: Control1_update
     FileName: Control1_update.control
     Description:
End:

Control: Control2_forecast
     FileName: Control2_forecast.control
     Description:
End:
```

```
Control: Control3_TMP
     FileName: Control3_TMP.control
     Description:
End:

Control: Control4_TMP
     FileName: Control4_TMP.control
     Description:
End:
```

2.3.2 .run file

This file defines the different configured runs (can be more than one).

```
Run: Run1
     Default Description: Yes
     Log File: Run1.log
     DSS File: UpperBlueNile.dss
     Basin: UpperBlueNile
     Precip: UpperBlueNile_Meteo
     Control: Control1
     Save State Name: UpperBlueNile_hist_state
     Save State At End of Run: Yes
End:

Run: Run2
     Default Description: Yes
     Log File: Run1.log
     DSS File: UpperBlueNile.dss
     Basin: UpperBlueNile
     Precip: UpperBlueNile_Meteo
     Control: Control2
     Save State Name: UpperBlueNile_hist_state
     Save State At End of Run: Yes
End:
```

2.3.3 .Control files

This file defines the control features of the different configured runs (can be more than one). With control features is meant:

- Start and end date/time of the simulation
- Time step of the simulation

```
Control: Control1
     Last Modified Date: 17 November 2010
     Last Modified Time: 09:43:55
     Start Date: 13 July 2010
     Start Time: 09:00
     End Date: 27 July 2010
     End Time: 09:00
     Time Interval:  1440
```

2.3.4 .dss and .dsc file

The .dss file contains all the data handled by the model. Data can be divided in different types:

- Input timeseries (external forcing)
- Output timeseries (simulation results)
- Tables (paired data) representing non time-dependent information of the model (model features like unit hydrographs, cross-sections etc…)

Contents of this file may be viewed (and manipulated) using a separate program: HEC-DSSVue[2].

In such viewer it is also possible to distinguish between the two above mentioned type of data (timeseries VS paired data). Timeseries are generally linked to a model run, (shown in Part F of the record line), while paired data is generally referred as TABLE.

The .dsc file lists timeseries contained in the .dss file:

```
     HECDSS Complete Catalog of Record Pathnames in File C:/Documents
and Settings/verkade/Desktop/temp_FEWS_Sudan/_tmp/UpperBlueNile.dss

     Catalog Created on Nov 29, 2010, 14:56
     Number of Records:0000014                   DSS
     Sort Order: ABCFED

  Ref.
 Number   Tag           Record Pathname

0000001          //FLOW_GAGE1/FLOW/01JAN1990/1DAY/GAGE/
0000002          //FLOW_GAGE1/FLOW/01JAN1991/1DAY/GAGE/
0000003          //FLOW_GAGE1/FLOW/01JAN1992/1DAY/GAGE/
0000004          //FLOW_GAGE1/FLOW/01JAN1993/1DAY/GAGE/
0000005          //FLOW_GAGE1/FLOW/01JAN1994/1DAY/GAGE/
0000006          //FLOW_GAGE1/FLOW/01JAN1995/1DAY/GAGE/
0000007          //FLOW_GAGE1/FLOW/01JAN1996/1DAY/GAGE/
0000008          //PRECIP_GAGE1/PRECIP-INC/01JAN1990/1DAY/GAGE/
0000009          //PRECIP_GAGE1/PRECIP-INC/01JAN1991/1DAY/GAGE/
0000010          //PRECIP_GAGE1/PRECIP-INC/01JAN1992/1DAY/GAGE/
0000011          //PRECIP_GAGE1/PRECIP-INC/01JAN1993/1DAY/GAGE/
0000012          //PRECIP_GAGE1/PRECIP-INC/01JAN1994/1DAY/GAGE/
0000013          //PRECIP_GAGE1/PRECIP-INC/01JAN1995/1DAY/GAGE/
0000014          //PRECIP_GAGE1/PRECIP-INC/01JAN1996/1DAY/GAGE/
```

2.3.5 .met file

The .met file specifies the meteorological data that will be used to force the model per basin. In the following example there is the definition of the external gage to use in the modelled basin (Gage: Precip_Gage1) and monthly evaporation and evapotranspiration.

[2] http://www.hec.usace.army.mil/software/hec-dssvue/

```
Meteorology: UpperBlueNile_Meteo
     Last Modified Date: 17 November 2010
     Last Modified Time: 08:51:24
     Version: 3.4
     Unit System: Metric
     Precipitation Method: Specified Average
     Radiation Method: None
     Snowmelt Method: None
     Evapotranspiration Method: Monthly Evaporation
     Use Basin Model: UpperBlueNile
     Use Basin Model: UpperBlueNile
End:

Precip Method Parameters: Specified Average
     Allow Depth Override: No
     Set Missing Data to Zero: No
End:

Subbasin: UpperBlueNile_basin
     Gage: Precip_Gage1

     Begin Et:
     Pan Evaporation: 3.79
     Pan Evaporation: 4.19
     Pan Evaporation: 4.67
     Pan Evaporation: 4.65
     Pan Evaporation: 4.09
     Pan Evaporation: 3.42
     Pan Evaporation: 2.72
     Pan Evaporation: 2.71
     Pan Evaporation: 3.26
     Pan Evaporation: 3.87
     Pan Evaporation: 3.6
     Pan Evaporation: 3.57
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     Evapotranspiration Coefficient: 0.7
     End Et:
End:
```

2.3.6 .basin file

The .basin file contains both the actual model schematisation and the model parameters:

```
Basin: UpperBlueNile
     Last Modified Date: 17 November 2010
     Last Modified Time: 08:51:24
     Version: 3.4
     Unit System: Metric
     Missing Flow To Zero: No
     Enable Flow Ratio: No
     Allow Blending: No
     Compute Local Flow At Junctions: No

     Enable Sediment Routing: No

     Enable Quality Routing: No
End:

Subbasin: UpperBlueNile_basin
     Canvas X: 2099658.788656069
     Canvas Y: 672543.2787210983
     Area: 176572
     Observed Hydrograph Gage: Flow_Gage1
     Downstream: EdDeim_station

     Canopy: SMA
     Initial Canopy Storage Percent: 50
     Canopy Maximum Storage: 1
     End Canopy:

     Surface: SMA
     Initial Surface Storage Percent: 80
     Surface Maximum Storage: 1
     End Surface:

     LossRate: Soil Moisture Account
     Percent Impervious Area: 0.0
     Initial Soil Storage Percent: 50
     Initial Gw1 Storage Percent: 80
     Initial Gw2 Storage Percent: 80
     Soil Maximum Infiltration: 0.6
     Soil Maximum Storage: 2
     Soil Tension Storage: 0.5
     Soil Maximum Percolation: 0.6
     Groundwater 1 Maximum Storage: 55
     Groundwater 1 Storage Coefficient: 7000
     Groundwater 1 Maximum Percolation: 0.8
     Groundwater 2 Maximum Storage: 50
     Groundwater 2 Storage Coefficient: 6000
     Groundwater 2 Maximum Percolation: 0.6
     Transform: Clark
     Time of Concentration: 160
     Storage Coefficient: 650

     Baseflow: SMA Groundwater
     Sma Groundwater Transform: SMA Linear Reservoir
     Sma Groundwater Layer: 1
     Number Groundwater Reservoirs: 6
     Groundwater Storage Coefficient: 500
     Initial Flow: 0
     Sma Groundwater Layer: 2
     Number Groundwater Reservoirs: 6
     Groundwater Storage Coefficient: 500
     Initial Flow: 0
End:

Junction: EdDeim_station
     Canvas X: 1637797.5169797686
     Canvas Y: 695971.0243858383
     Observed Hydrograph Gage: Flow_Gage1
End:

Basin Schematic Properties:
     Last View N: 5000.0
     Last View S: -5000.0
     Last View W: -5000.0
     Last View E: 5000.0
     Maximum View N: 900127.09375
     Maximum View S: 321127.09375
     Maximum View W: 1588806.125
     Maximum View E: 2159806.25
     Extent Method: Elements
     Buffer: 0
     Draw Icons: Yes
     Draw Icon Labels: Yes
     Draw Map Objects: No
     Draw Gridlines: No
     Draw Flow Direction: No
     Fix Element Locations: No
     Fix Hydrologic Order: No
End:
```

2.3.7 .gage file

This file gives some meta-data on the stations for which observed data exists and is used for running the model.

The last line “Pathname” is of special importance because it specifies the path in the DSS where the data is located:

```
Gage Manager: UpperBlueNile
     Version: 3.4
End:

Gage: Flow_Gage1
     Last Modified Date: 17 November 2010
     Last Modified Time: 09:43:55
     Units System: SI
     Reference Height Units: Meters
     Reference Height: 10.0
     Gage Type: Flow
     Units: M3/S
     Data Type: INST-VAL
     Local to Project: NO
     Start Time: 1 January 1990, 00:00
     End Time: 1 January 1992, 00:00
     Start Time: 1 January 1990, 00:00
     End Time: 31 December 1996, 00:00
     Start Time: 2 January 1990, 00:00
     End Time: 1 January 1992, 00:00
     Start Time: 1 January 1992, 00:00
     End Time: 31 December 1996, 00:00
     Start Time: 2 January 1992, 00:00
     End Time: 31 December 1996, 00:00
     DSS File: UpperBlueNile.dss
     Pathname: //FLOW_GAGE1/FLOW/01JAN1990 - 30DEC1996/1DAY/GAGE/
End:

Gage: Precip_Gage1
     Last Modified Date: 17 November 2010
     Last Modified Time: 09:44:01
     Units System: SI
     Reference Height Units: Meters
     Reference Height: 10.0
     Gage Type: Precipitation
     Precipitation Type: Incremental
     Units: MM
     Data Type: PER-CUM
     Local to Project: NO
     Start Time: 1 January 1990, 00:00
     End Time: 31 December 1996, 00:00
     DSS File: UpperBlueNile_precip_forcing.dss
     Pathname:           //PRECIP_GAGE1/PRECIP-INC/01JAN1990           -
30DEC1996/1DAY/GAGE/
End:
```

2.3.8 .pdata file

This file gives some meta-data on the paired data defined in the model.

Also in this case the last line “Pathname” is of special importance because it specifies the path in the DSS where the data is located:

```
Paired Data Manager UpperBlueNile
     Version: 3.4
End:

Pattern: UpperBlueNile_basin
     Data Type: Unit Hydrograph
     Last Modified Date: 17 November 2010
     Last Modified Time: 08:51:24
     Units: M3/S
     Use External DSS File: YES
     DSS File: UpperBlueNile.dss
     Pathname://UPPERBLUENILE_BASIN/FLOW-UNITGRAPH/TS-
PATTERN/3HOUR/TABLE/
End:
```

2.3.9 .log and .out files

This files contain logging and output information of the run (thay do not contain model results)

2.3.10 .state file

This file contains the initial conditions to be used in the run, if the run has been configured to start with initial conditions.

The same file is produced by HEC-HMS periodically or at the end of the run, if the model has been configured to do so.

2.3.11 .stateIndex file

This file is always associated to the .state file and gives a snapshot of the state, giving the time associated to the state.

3 Running the HEC-HMS model in the Delft-FEWS environment

It is assumed that there is a working stand alone HEC-HMS schematisation. This means that a model has been delivered which can be opened with no errors from the HEC-HMS GUI interface, and can run the configured runs using data from a .dss.

This schematisation and its corresponding model run files consist of a number of "human readable" files (text files). These files may have to be adapted in order for the model to run in the Delft-FEWS environment.

3.1 Required changes to the HEC-HMS files

In order to be able to run the HEC-HMS model within the Delft-FEWS environment, some changes to the HMS model management files have to be made.

There are three different kinds of changes needed in order to run the HEC-HMS model under Delft-FEWS:

1 *Source of forcing data,* changes to be applied to .gage file
2 *Source of paired data,* changes to be applied to .pdata file
3 *Number of controls,* changes to be applied to .hms file
4 *Number of runs and state handling*, changes to be applied to .run file

3.1.1 Source of forcing data

Forcing data can be managed from the GUI via the "Time Series data" filter or directly on the .gage file.

A stand alone model generally runs with forcing data (rainfall and/or discharges) present in the .dss file. In such situation forcing data is stored in the same .dss file as forcing data and results.

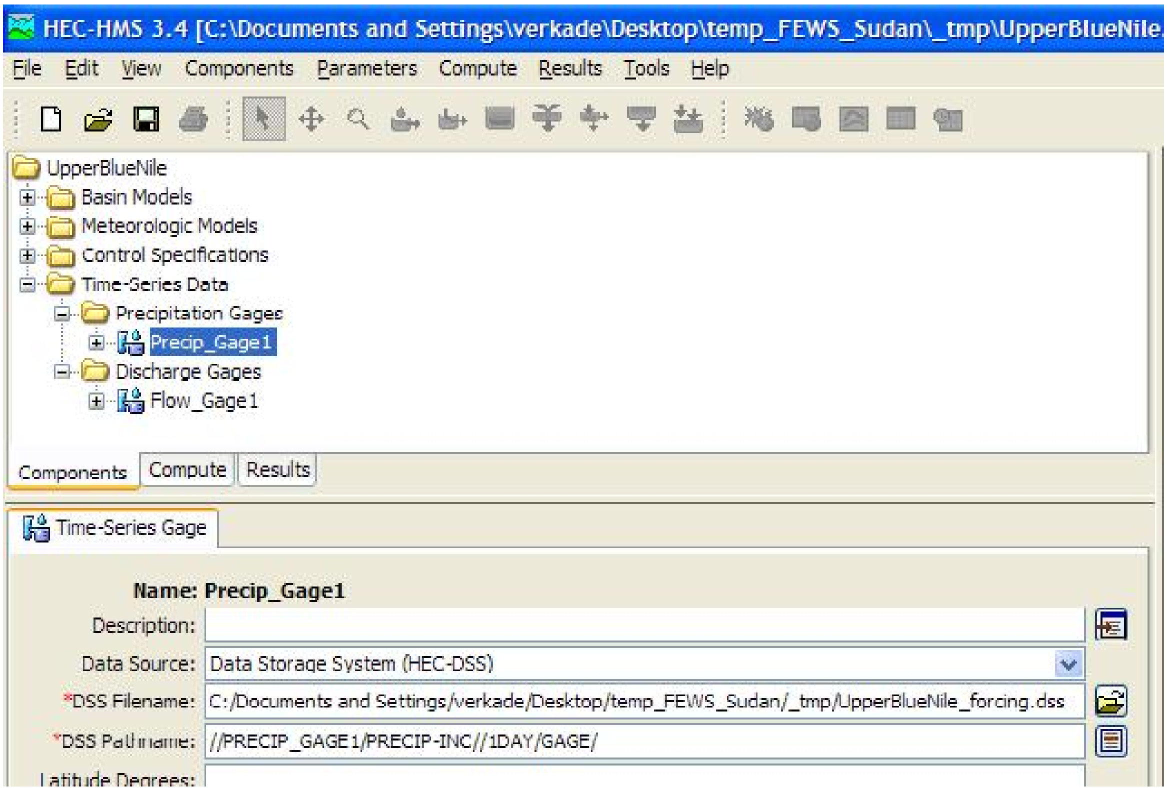
When running the model from Delft-FEWS, the adapter prepares a .dss file with the forcing timeseries from Delft-FEWS. For this reason the model needs to read forcing data strictly from the .dss Delf-FEWS prepares.

In order to manage the forcing data, changes might be needed in the .gage file:

1. If the data source is not a .dss file, it has to be set from the field "Data Source" by setting "Data Storage System HEC DSS".
   The handiest way is to change it via the GUI, since it will automatically refer to the model .dss file, adding its path and the internal path of the timeseries.
2. It is recommended – *though not strictly necessary* – that the forcing data is stored in a different .dss file than the model simulation results (highlighted in yellow). The reason for this is that forcing data is exported from FEWS; if stored in the same .dss-file as the simulation results, this will (marginally) slow down conversion of that .dss-file to a PI-timeseries file which will subsequently be imported back into FEWS. This can be changed in two different ways
   a) From the GUI interface as shown in Figure 3.1 and Figure 3.2, by editing the name of the .dss in the field "DSS Filename" (the DSS pathname has to be left as it is)

b) Directly in the .gage file by editing the field “DSS File” (highlighted in yellow).

In the shown example only the rainfall is given as external forcing from Delft-FEWS (data from Delft-FEWS stored in UpperBlueNile_Forcing.dss).



HEC-HMS 3.4 [C:\Documents and Settings\verkade\Desktop\temp_FEWS_Sudan_tmp\UpperBlueNile.
File Edit View Components Parameters Compute Results Tools Help
UpperBlueNile
Basin Models
Meteorologic Models
Control Specifications
Time-Series Data
Precipitation Gages
Precip_Gage1
Discharge Gages
Flow_Gage1
Components Compute Results
Time-Series Gage

Name: Precip_Gage1

Description:

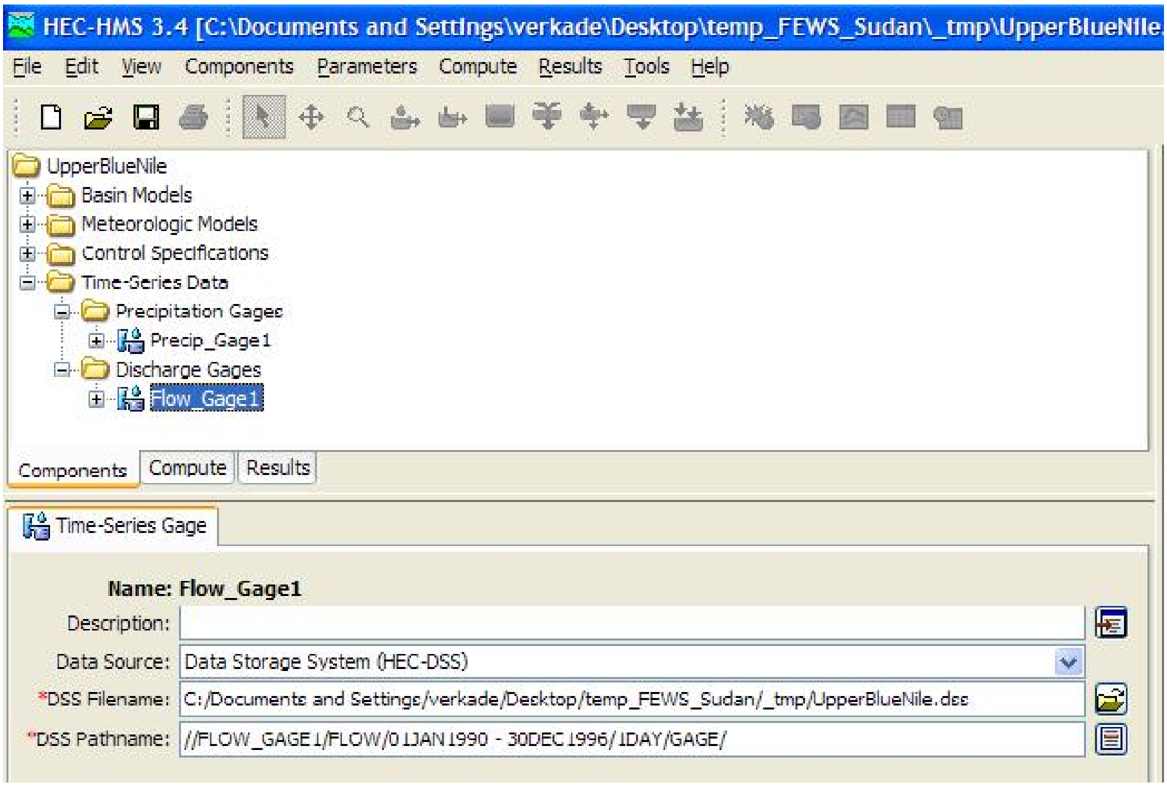
Data Source: Data Storage System (HEC-DSS)

*DSS Filename: C:/Documents and Settings/verkade/Desktop/temp_FEWS_Sudan/_tmp/UpperBlueNile_forcing.dss

*DSS Pathname: //PRECIP_GAGE1/PRECIP-INC//1DAY/GAGE/

Latitude Degrees:

Figure 3.1 “Gage” elements in the HEC-HMS user interface



HEC-HMS 3.4 [C:\Documents and Settings\verkade\Desktop\temp_FEWS_Sudan_tmp\UpperBlueNile.

- UpperBlueNile
  - Basin Models
  - Meteorologic Models
  - Control Specifications
  - Time-Series Data
    - Precipitation Gages
      - Precip_Gage1
    - Discharge Gages
      - Flow_Gage1

Name: Flow_Gage1

Description:

Data Source: Data Storage System (HEC-DSS)

*DSS Filename: C:/Documents and Settings/verkade/Desktop/temp_FEWS_Sudan/_tmp/UpperBlueNile.dss

*DSS Pathname: //FLOW_GAGE1/FLOW/01JAN1990 - 30DEC1996/1DAY/GAGE/

Figure 3.2 “Gage” elements in the HEC-HMS user interface

3. It is recommended to check the data types (highlighted in purple) in order to clarify the format of data which has to be exported from Delft-FEWS. Four data types are possible:
   a) PER-CUM (PRECIP-INC): Period Cumulative, typically used for incremental precipitation (mm/Δt)
   b) INST-CUM (PRECIP-CUM): Instantaneous Cumulative, typically used for cumulative precipitation (mm)
   c) INST-VAL (FLOW), Instantaneous Value, typically used for instantaneous discharges (timestep less than 24h)
   d) PER-AVG (FLOW), Period Average, typically used for average discharges (timestep longer than 24h)

   The definition of the correct data type (as for the export of the correct data type from Delft-FEWS) is very important for the interpretation of the exported values from the model and their eventual interpolation. Figure 3.3 shows interpolation techniques per type:

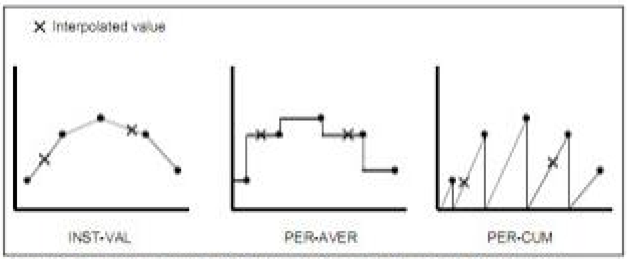


Figure 7.10 Interpolation of "INST-VAL", "PER-AVER" and "PER-CUM" Data

Figure 3.3 Interpolation of “INST-VAL”, “PER-AVER”, “PER-CUM”

Figure 3.4 shows the difference between PER-CUM (PRECIP-INC) and INST-CUM (PRECIP-CUM) data type for precipitation:

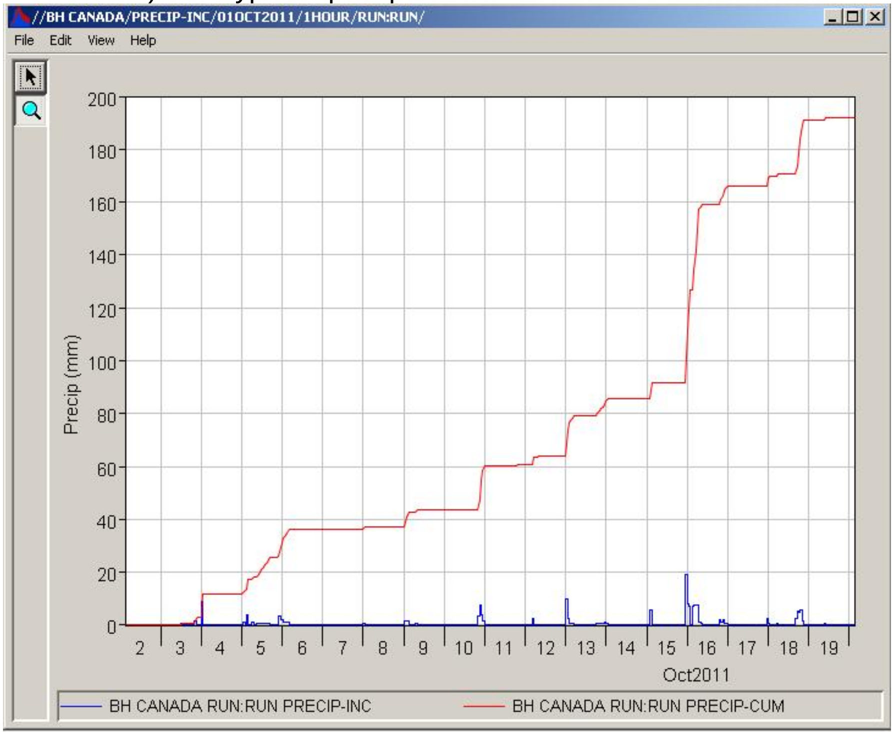


Figure 3.4 Precipitation data, PER-CUM VS INST-CUM

```
Gage Manager: UpperBlueNile
     Version: 3.4
End:

Gage: Flow_Gage1
     Last Modified Date: 17 November 2010
     Last Modified Time: 09:43:55
     Units System: SI
     Reference Height Units: Meters
     Reference Height: 10.0
     Gage Type: Flow
     Units: M3/S
     Data Type: INST-VAL
     Local to Project: NO
     Start Time: 1 January 1990, 00:00
     End Time: 1 January 1992, 00:00
     Start Time: 1 January 1990, 00:00
     End Time: 31 December 1996, 00:00
     Start Time: 2 January 1990, 00:00
     End Time: 1 January 1992, 00:00
     Start Time: 1 January 1992, 00:00
     End Time: 31 December 1996, 00:00
     Start Time: 2 January 1992, 00:00
     End Time: 31 December 1996, 00:00
     DSS File: UpperBlueNile.dss
     Pathname: //FLOW_GAGE1/FLOW/01JAN1990 - 30DEC1996/1DAY/GAGE/
End:

Gage: Precip_Gage1
     Last Modified Date: 17 November 2010
     Last Modified Time: 09:44:01
     Units System: SI
     Reference Height Units: Meters
     Reference Height: 10.0
     Gage Type: Precipitation
     Precipitation Type: Incremental
     Units: MM
     Data Type: PER-CUM
     Local to Project: NO
     Start Time: 1 January 1990, 00:00
     End Time: 31 December 1996, 00:00
     DSS File: UpperBlueNile_precip_forcing.dss
     Pathname:           //PRECIP_GAGE1/PRECIP-INC/01JAN1990           -
30DEC1996/1DAY/GAGE/
End:
```

3.1.2 Source of paired data

paired data can be managed from the GUI via the “Paired Data” filter or directly on the .pdata file.

A stand alone model generally runs with paired data (cross sections/ hydrographs etc…) present in the .dss file. In such situation paired data is stored in the same .dss file as forcing data and results.

When running the model from Delft-FEWS, the adapter prepares a .dss file with the forcing timeseries from Delft-FEWS. This .dss file *will overwrite any existing .dss file with the same name*. For this reason it is strictly necessary to export paired data from the model .dss provided in the stand alone setup to another .dss file containing only paired data.

In order to extract safely paired data it is possible to use DSSVue:

1. Open the model .dss file (containing stand alone timeseries and paired data)
2. Select the paired data needed for the model
3. Go to “Edit” and “Copy to” and select the .dss file where paired data need to be copied (a newly made .dss file)

Once paired data has been isolated in one .dss file, in order to manage them, changes are needed in the .pdata file:

1. If the data source is not a .dss file, it has to be set from the field “Data Source” by setting “Data Storage System HEC DSS”. The handiest way is to change it via the GUI.
2. In this case the “DSS filename” is the file name of the newly made .dss file with paired data. This can be changed in two different ways
    c) From the GUI interface by editing the name of the .dss in the field “DSS Filename”
    d) Directly in the .pdata file by editing the field “DSS File” (highlighted in yellow).

```
Paired Data Manager UpperBlueNile
     Version: 3.4
End:

Pattern: UpperBlueNile_basin
     Data Type: Unit Hydrograph
     Last Modified Date: 17 November 2010
     Last Modified Time: 08:51:24
     Units: M3/S
     Use External DSS File: YES
     DSS File: UpperBlueNile_model.dss
     Pathname://UPPERBLUENILE_BASIN/FLOW-UNITGRAPH/TS-
PATTERN/3HOUR/TABLE/
End:
```

3.1.3 Number of Controls

The .hms file should contain reference to a single “Control” element only. This may be set either through the user interface (Figure 3.5) or, manually, in the .hms-file.

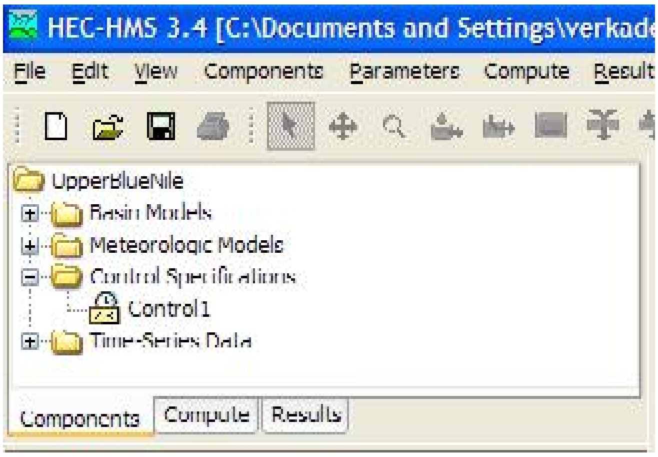


Figure 3.5 “Control” elements in the HEC-HMS user interface

```
Project: UpperBlueNile
     Description:
     Version: 3.4
     DSS File Name: UpperBlueNile.dss
End:

Precipitation: UpperBlueNile_Meteo
     FileName: UpperBlueNile_Meteo.met
     Description:
End:

Basin: UpperBlueNile
     FileName: UpperBlueNile.basin
     Description:
End:

Control: Control1
     FileName: Control1.control
     Description:
End:
```

3.1.4 Number of Runs

The .run file should contain reference to a single “Run” element only. This may be set either through the user interface (Figure 3.6Figure 3.5) or, manually, in the .run-file.

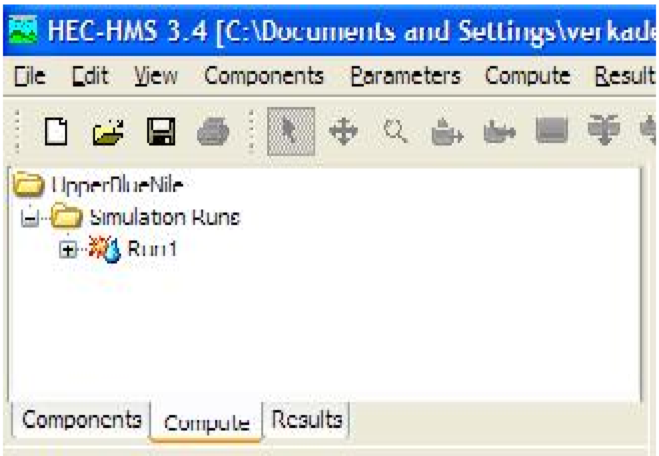

HEC-HMS 3.4 [C:\Documents and Settings\verkade
File Edit View Components Parameters Compute Result
UpperBlueNile
Simulation Runs
Run1
Components Compute Results

Figure 3.6 “Simulation Run” elements in the HEC-HMS user interface

3.1.5 State handling, continuous VS event mode

The HEC-HMS adapter is able to run the models both in continuous and in event mode:

1. Continuous mode: every simulation starts from an initial state which has been made by a previous run and saves a state at the end of the simulation. This is the mode used in operational forecasting.
2. Event mode: the model runs with no initial state for a defined and finite meteorological event.

Different configuration settings are needed in order to run the HMS model in the two modes:

3.1.5.1 Event mode

In order to run the HEC-HMS model in event mode the following setting are needed:

- In the .run file of the HEC-HMS model no lines defining “Start State Name” and “Save State Name”. This can be done directly deleting the two lines in the .run file or via the GUI in the run definition, Figure 3.7 and Figure 3.8.
- In the General adapter no “exportStateActivity” and “importStateActivity”. Remember to define a “start” tag of the relative view period of the exported timeseries.

In case of running in event mode the following info will be visualized in the log: *INFO - No "Start State Name" present in D:\.. \Model.run. Starting without start state.*

3.1.5.2 Continuous mode

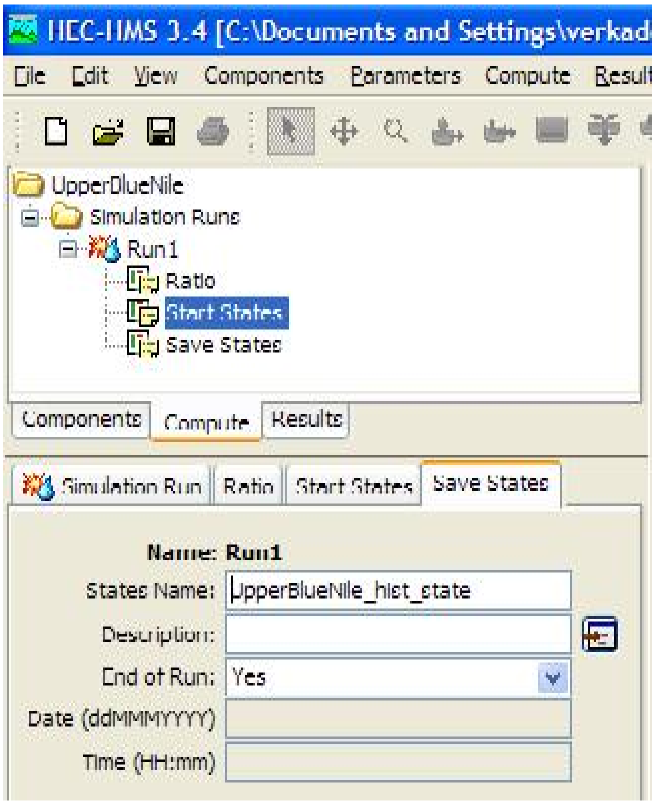
When used in operational forecasting, the “start state” file should be the last saved state from a previous run. It is therefore important to save a state at the end of the run, and declare a start state. It is not strictly needed to name the two with the same name, the HEC-HMS adapter will read the .run file and manage the state handling.

However, before changing the .run options, it is needed to create a state to be used as cold state (if not already existing). This can be done by Figure 3.7 and Figure 3.8:

1. Define a “State Name” in the “Save States” window. At first it is suggested to create a state with the name of the input restart file (UpperBlueNile_hist_state).
2. Select “Yes” in the field “End of Run”.
3. Run the model and create a state which can be used as cold state
4. Change “State Name” in the “Save States” window to the output state name (UpperBlueNile_out_state).
5. Select the previously made start state (UpperBlueNile_hist_state) in the window “start state”.

Once the cold state is made (till point), the other operations can be done manually in the .run file.

Note that it is not possible anymore to run the model at this point, because the initial state has been made at a timestep which is at the end of the simulation. The model as it is now need to be saved and implemented in Delft-FEWS.



Name: Run1

States Name: UpperBlueNile_hist_state

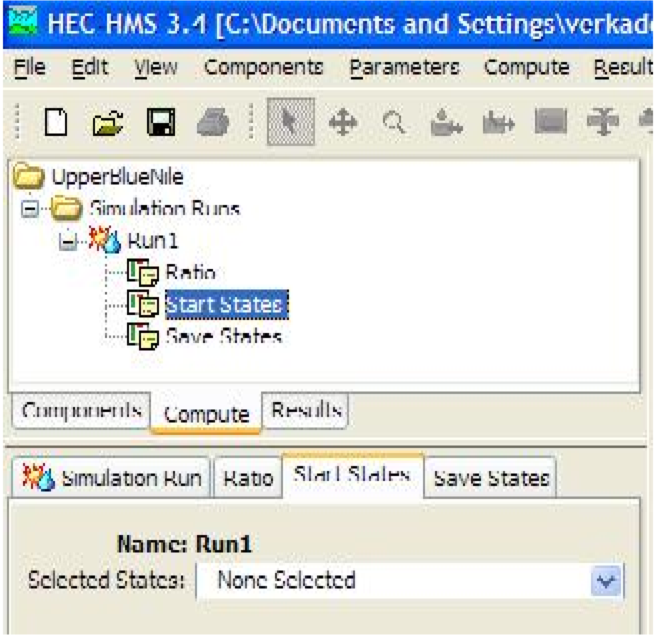
Description:

End of Run: Yes

Date (ddMMMYYYY)

Time (HH:mm)

Figure 3.7 Setting of “Save state” in the HEC-HMS user interface



Name: Run1

Selected States: None Selected

Figure 3.8 Setting of “Start state” in the HEC-HMS user interface

```
Run: Run1
     Default Description: Yes
     Log File: Run1.log
     DSS File: UpperBlueNile.dss
     Basin: UpperBlueNile
     Precip: UpperBlueNile_Meteo
     Control: Control1
     Save State Name: UpperBlueNile_hist_state
     Save State At End of Run: Yes
     Start State Name: UpperBlueNile_out_state
End:
```

NB: Following the explained procedure, the Delft-FEWS HEC-HMS run will handle three different .dss file:

1. UpperBlueNile_forcing.dss, created by the General Adapter of Delft-FEWS and declared in the .gage file at each timeseries exported from Delft-FEWS.
2. UpperBlueNile_model.dss, created by the user with paired data and placed together with the model files. This is declared in the .pdata file at each paired data needed from the model.
3. UpperBlueNile.dss, created by the HEC_HMS model, with model results and declared in the .run file. This will be used by Delft-FEWS to import model results.

3.2 Running a HEC-HMS simulation run through a script file

HEC-HMS simulations may be started outside of the user interface, using a script. This is described in Chapter 2 of the User Manual[3].

The script is called using the following command line; the .cmd file is included in the program installation folder:

```
hec-hms.cmd -s "..\UpperBlueNile.script"
```

The script includes a reference to the .hms-file, its location (line 2) and to the simulation run that needs to be computed (line 3):

```
from hms.model.JythonHms import *
OpenProject("UpperBlueNile", "..\\UpperBlueNile")
Compute("Run1")
Exit(1)
```

[3] United States Army Corps of Engineers, Hydrologic Engineering Centre (2010), Hydrologic Modeling System, User's Manual version 3.5, United States: Davis, available from: http://www.hec.usace.army.mil/software/hec-hms/documentation.html

4 Running an HEC-HMS model in Delft-FEWS

After implementing the above descripted changes, adding an HEC-HMS model to a Delft-FEWS configuration includes the following steps:

1. Make a new Coldstate directory for the HEC-HMS model and zip the newly made cold state `/Config/ColdStateFiles`
2. Copy the HEC-HMS files from the HEC-HMS work directory to the FEWS environment, zipped in `/Config/ModuleDataSetFiles`
3. Make a new General Adapter file in the `/Config/ModuleConfigFiles`
4. Add new HEC-HMS workflows
5. Register new files and location in the `/Config/RegionConfigFiles`

The steps will be explained in more detail in the following paragraphs.

4.1 Copy the HEC-HMS cold state to the FEWS environment

1 Copy the .state and files to a .zip file that has the filename: <moduleInstanceId> + “default.zip” (e.g. `UpperBlueNile_HEC-HMS_historical default.zip`)

2 Copy this .zip-file to the `%REGION_HOME%\ColdStates` folder

4.2 Copy the HEC-HMS model files to the FEWS environment

1. Create a new folder in the `%REGION_HOME%/Modules` folder, e.g.: `f:\FEWS_Sudan\FEWS_Sudan\Modules\hechms\`
2. Create 3 sub-folders:
   a) A bin folder with the HEC-HMS program (copied from the installation folder)
   b) A bin folder with the adapter
   c) A folder containing the model files, e.g. `UpperBlueNile`. See below example.

```
%REGION_HOME%/Modules
+---hechms
    +---bin
    +---binAdapter
    +---UpperBlueNile
        +---basinStates
        +---maps
        +---optimizer
        +---piInputTimeSeries
        +---piOutputTimeSeries
```

3. Zip HMS model files to `/Config/ModuleConfigFiles`. These files include (also see section 2.3):
   a. .basin
   b. .control
   c. .gage
   d. .hms

    e. .met
    f. .run
    g. .pdata (if available)
4. Zip the .script file (section 3.2) to the same folder.

4.3 Create a "General Adapter" file

The General Adapter is part of DELFT-FEWS. It is responsible for the data exchange with the models and for executing the models and their adapters, see Figure 4.1. The General Adapter module can be configured to carry out a sequence of five types of tasks;

a. Startup Activities. These activities are run prior to a module run and any export import of data. The activities defined are generally used to remove files from previous runs that may implicate the current run.
b. Export Activities. These activities defined all items to be exported through the published interface XML formats to the external module, prior to the module or the module adapters being initialized.
c. Execute Activities. The execute activities define the external executables or Java classes to be run. Tracking of diagnostics from these external activities is included in this section.
d. Import Activities: These activities define all items to be imported following successful completion of the module run.
e. Shutdown Activities. These activities are run following completion of all other activities. The activities defined are generally used to remove files that are no longer required.

The general part administrates the needed files for the general adapter in terms of file names and locations.

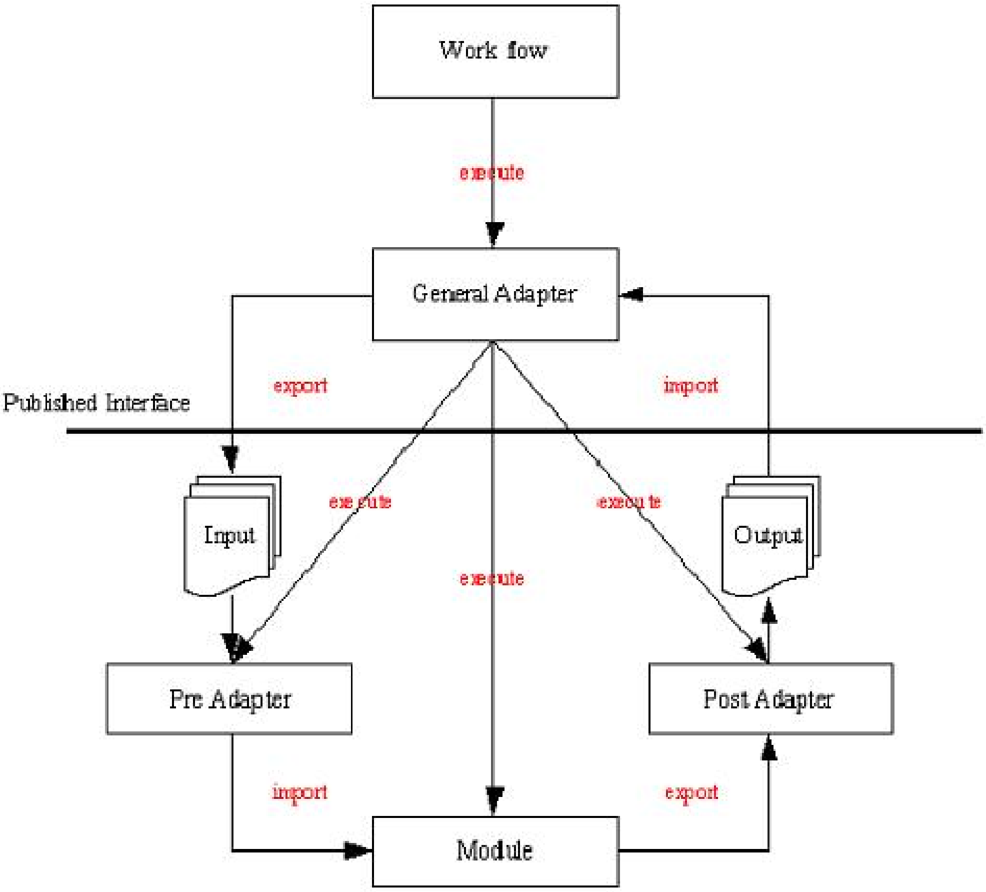


Figure 4.1 Schematic overview of General Adapter tasks

The general adapter file is stored in the `/Config/ModuleConfigFiles/<segment>/` directory. An example is shown in the Appendix. Some of the sections are discussed in more detail below.

4.3.1 "General" section

Make sure that the `%ROOT_DIR%` refers to the newly created module folder and that the `%WORK_DIR%` refers to the bin folder that includes the HEC-HMS program files, specifically the `hec-hms.cmd` file. Other elements in the General section should be referenced relative to these items.

```
<general>
        <rootDir>%REGION_HOME%/Modules/hechms</rootDir>
        <workDir>%ROOT_DIR%/bin</workDir>
        <exportDir>%ROOT_DIR%/UpperBlueNile/piInputTimeSeries</exportDir>
        <exportIdMap>IdExportHECHMS</exportIdMap>
        <importDir>%ROOT_DIR%/UpperBlueNile/piOutputTimeSeries</importDir>
        <importIdMap>IdImportHECHMS</importIdMap>
        <dumpFileDir>$GA_DUMPFILEDIR$</dumpFileDir>
        <dumpDir>%ROOT_DIR%/UpperBlueNile</dumpDir>
        <diagnosticFile>%ROOT_DIR%/UpperBlueNile/piDiagnostic.xml</diagnosticFile>
        <missVal>-901.0</missVal>
```

```
                <convertDatum>true</convertDatum>
        </general>
```

4.3.2 exportStateActivities

In the state location section should be important to place the name of the state file corresponds with the .state files handled in input and output in HEC-HMS, eg. `UpperBlueNile_hist_state.state` and `UpperBlueNile_out_state.state`.

Nevertheless this is a needed section for the adapter schema, it is actually not used from the adapter who directly reads in the HEC-HMS .run file the names of the input and output state files.

```
<exportActivities>
        <!-- Export state (warm state)-->
        <exportStateActivity>

                <moduleInstanceId>UpperBlueNile_HEC-HMS_historical</moduleInstanceId>

                <stateExportDir>%ROOT_DIR%/UpperBlueNile/basinStates</stateExportDir>

                <stateConfigFile>%ROOT_DIR%/UpperBlueNile/State.xml</stateConfigFile>

                <stateLocations type="file">
                     <stateLocation>
        <readLocation>UpperBlueNile_hist_state.state</readLocation>
        <writeLocation>UpperBlueNile_hist_state.state</writeLocation>
                     </stateLocation>
                </stateLocations>

         <stateSelection>
             <warmState>
                <stateSearchPeriod unit=”hour” start=”-240” end=”-1”/>
             </warmState>
          </stateSelection>
…
</exportActivities>
```

4.3.3 exportTimeseriesActivities

timeseries listed in the .gage file need to be exported from Delft-FEWS. It is important to make sure that the exporting time step is the same as required by the .gage file.

```
<exportActivities>
…
      <exportTimeSeriesActivity>
            <exportFile>%ROOT_DIR%/UpperBlueNile/piInputTimeSeries/input,xml</exportFile>
            <timeSeriesSet>
                  …
            </timeSeriesSet>
      </exportTimeSeriesActivity>
…
</exportActivities>
```

4.3.4 exportRunFileActivities

Exporting the run file is fundamental because the adapter retrieves the run information (start/end etc…) from this file.

```
<exportRunFileActivity>
       <exportFile>%ROOT_DIR%/work/RunInfo.xml</exportFile>
       <properties>
              <string value="%ROOT_DIR%/work/UpperBlueNile.hms" key="hmsProjectFile"/>
       </properties>
</exportRunFileActivity>
```

4.3.5 executeActivities

Three activities should be defined:

1. the pre-adapter, it is important to address the right bin in the argument
2. the actual model run
3. the post-adapter, it is important to address the right bin in the argument

By post and pre adapter it is very important to address the right bin in the command and the RunInfo.xml as argument.

See below example for what commands and what arguments should be used.

```
<executeActivities>
       <executeActivity>
           <command>
                     <className>nl.wldelft.fews.adapter.hec.HmsPreAdapter</className>
                     <binDir>%ROOT_DIR%/binAdapter</binDir>
              </command>

              <arguments>
                      <argument>%ROOT_DIR%/work/RunInfo.xml</argument>
              </arguments>

              <timeOut>120000</timeOut>
       </executeActivity>

       <executeActivity>
              <command>
                     <executable>%WORK_DIR%/HEC-HMS.cmd</executable>
              </command>

              <arguments>
                     <argument>-s</argument>
                     <argument>%ROOT_DIR%/UpperBlueNile/UpperBlueNile.script </argument>
              </arguments>

              <timeOut>120000</timeOut>
       </executeActivity>

       <executeActivity>
              <command>
                     <className>nl.wldelft.fews.adapter.hec.HmsPostAdapter</className>
                      <binDir>%ROOT_DIR%/binAdapter</binDir>
              </command>

              <arguments>
                      <argument>%ROOT_DIR%/work/RunInfo.xml</argument>
              </arguments>

              <timeOut>120000</timeOut>
       </executeActivity>
</executeActivities>
```

The .script file that is referenced in the second activity looks as follows:

```
from hms.model.JythonHms import *
OpenProject("UpperBlueNile", "../UpperBlueNile/")
```

```
Compute("Run1")
Exit(1)
```

As this file is run from the UpperBlueNile (model) folder, the second part on line 2 references its own folder.

4.3.6 importActivities

If a state needs to be imported, the location should be as follows:

```
<importActivities>
      <!-- Import state -->
      <importStateActivity>
        <stateConfigFile>%ROOT_DIR%/UpperBlueNile/piOutputState.xml</stateConfigFile>
      </importStateActivity>
      <!-- Import HMS results-->
      <importTimeSeriesActivity>
             <importFile>%ROOT_DIR%/UpperBlueNile/piInputTimeSeries/input,xml</exportFile>
             <timeSeriesSet>
                   …
             </timeSeriesSet>
      </importTimeSeriesActivity>
</importActivities>
```

Add the desired output time series in the importTimeSeriesActivity. Since the location and parameters in HEC-HMS are different than in Delft-FEWS, these should be mapped in the IdImportHECHMS file, see section 4.4

An example of a general adapter for executing a HEC-HMS model is shown in the Appendix. This is an UpdateStates file. Also a Forecast file has to be created. Main difference is that the Forecast file does not have an importStateActivity. Furthermore, all time series types are forecasting instead of historical and use a relative view period with addition *endOverrulable="true*". Different is also the warm state search period, its end should be set to 0 (for a Forecast file) instead of -1 (UpdateStates). The UpdateStates search period is set to -1 because the forecast length for UpdateStates would be 0 and FEWS will crash if a state at T0 is used because the simulation time is empty.

4.4 Create an IdMapFile

It is important to map Delft-FEWS locations and parameter to the correct ones of HEC-HMS. IdMapFiles convert parameter and locations names from those used within HEC-HMS to those used within FEWS and vice versa. It is recommended to create an IdMapFile for both “FEWS to HMS” (export) and “HMS to FEWS” (import). Please note that the “internal” and “external” references should be seen from the viewpoint of FEWS.

Parameter and location id of HEC-HMS can be retrieved from the pathname which defines the location of a timeseries in the .dss (to be found in the .gage file). The id mapping can be created from the pathnames definition (see corresponding colors).

```
Pathname: //PRECIP_GAGE1/PRECIP-INC/01JAN1990 - 30DEC1996/1DAY/GAGE/
<map internalLocation="UpperBlueNile" externalLocation="PRECIP_GAGE1"
internalParameter="P.merged" externalParameter="PRECIP-INC"/>

Pathname: //FLOW_GAGE1/FLOW/01JAN1990 - 30DEC1996/1DAY/GAGE/
<map internalLocation="900012" externalLocation=" FLOW_GAGE1"
```

```
internalParameter="Q.simulated.historical" externalParameter="FLOW"/>
```

4.5 Create a workflow file

The GeneralAdapter module should be run from a workflow file. The workflow should be palced in the the `%REGION_HOME%\Config\WorkflowFiles` folder and registered in `%REGION_HOME%\Config\RegionConfigFiles\WorkflowDescriptors.xml`

4.6 Additional requirements

Requirements that are not described here include usual needed operations by any general adapter:

- Registering moduleInstance files in `ModuleInstanceDescriptors.xml`
- Registering workflow files in `WorkflowDescriptors.xml`
- Ensuring that the parameters are listed in `Parameters.xml` (if used)
- Ensuring that the used locations are listen in `Locations.xml`
- Ensuring the unit conversions are listed in the UnitConversionsFiles folder if applicable.

A Example of General Adapter file

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2009 sp1 (http://www.altova.com) by ICT (Stichting Deltares) -->
<generalAdapterRun xmlns="http://www.wldelft.nl/fews"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.wldelft.nl/fews
http://fews.wldelft.nl/schemas/version1.0/generalAdapterRun.xsd">
<general>
        <rootDir>%REGION_HOME%/Modules/hechms</rootDir>
        <workDir>%ROOT_DIR%/bin</workDir>
        <exportDir>%ROOT_DIR%/UpperBlueNile/piInputTimeSeries</exportDir>
        <exportIdMap>IdExportHECHMS</exportIdMap>
        <importDir>%ROOT_DIR%/UpperBlueNile/piOutputTimeSeries</importDir>
        <importIdMap>IdImportHECHMS</importIdMap>
        <dumpFileDir>$GA_DUMPFILEDIR$</dumpFileDir>
        <dumpDir>%ROOT_DIR%/UpperBlueNile</dumpDir>
        <diagnosticFile>%ROOT_DIR%/UpperBlueNile/piDiagnostic.xml</diagnosticFile>
        <missVal>-901.0</missVal>
        <convertDatum>true</convertDatum>
</general>

<activities>
        <startUpActivities>
               <purgeActivity>
                       <filter>%ROOT_DIR%/UpperBlueNile/piInputTimeSeries/*.*</filter>
               </purgeActivity>
               <purgeActivity>
                       <filter>%ROOT_DIR%/UpperBlueNile/piOutputTimeSeries/*.*</filter>
               </purgeActivity>
               <purgeActivity>
                       <filter>%ROOT_DIR%/UpperBlueNile/*.ds*</filter>
               </purgeActivity>
        </startUpActivities>

        <exportActivities>
               <!-- Export state (warm state)-->
            <exportStateActivity>

               <moduleInstanceId>UpperBlueNile_HEC-HMS_historical</moduleInstanceId>
               <stateExportDir>%ROOT_DIR%/UpperBlueNile/basinStates</stateExportDir>
               <stateConfigFile>%ROOT_DIR%/UpperBlueNile/State.xml</stateConfigFile>
               <stateLocations type="file">
                    <stateLocation>
                           <readLocation>UpperBlueNile_hist_state.state</readLocation>
                           <writeLocation>UpperBlueNile_hist_state.state</writeLocation>
                    </stateLocation>
               </stateLocations>
               <stateSelection>
                    <warmState>
                           <stateSearchPeriod unit=”hour” start=”-240” end=”-1”/>
                   </warmState>
               </stateSelection>
           </exportStateActivity>
           <!-- Export time series -->
        <exportTimeSeriesActivity>
        <description>Export rainfall as input for HecHms model</description>
               <exportFile>FEWS2DSS.xml</exportFile>
               <timeSeriesSets>
                   <timeSeriesSet>
                      <moduleInstanceId>Catchments_Aggregate
                                      _Precipitation_Historical</moduleInstanceId>
                      <valueType>scalar</valueType>
                      <parameterId>P.merged</parameterId>
                      <locationId>UpperBlueNile</locationId>
                      <timeSeriesType>external historical</timeSeriesType>
                      <timeStep times="21:00"/>
                      <relativeViewPeriod unit="day" startOverrulable="true"
                                                                       end="0"/>
                      <readWriteMode>read only</readWriteMode>
                   </timeSeriesSet>
               </timeSeriesSets>
               <checkMissing>true</checkMissing>
        </exportTimeSeriesActivity>
</exportActivities>
<executeActivities>
<executeActivity>
    <command>
              <className>nl.wldelft.fews.adapter.hec.HmsPreAdapter</className>
              <binDir>%ROOT_DIR%/binAdapter</binDir>
       </command>

       <arguments>
               <argument>%ROOT_DIR%/work/RunInfo.xml</argument>
       </arguments>

       <timeOut>120000</timeOut>
</executeActivity>

<executeActivity>
       <command>
              <executable>%WORK_DIR%/HEC-HMS.cmd</executable>
       </command>

       <arguments>
              <argument>-s</argument>
              <argument>%ROOT_DIR%/UpperBlueNile/UpperBlueNile.script </argument>
       </arguments>

       <timeOut>120000</timeOut>
</executeActivity>

<executeActivity>
       <command>
              <className>nl.wldelft.fews.adapter.hec.HmsPostAdapter</className>
               <binDir>%ROOT_DIR%/binAdapter</binDir>
       </command>

       <arguments>
               <argument>%ROOT_DIR%/work/RunInfo.xml</argument>
       </arguments>

       <timeOut>120000</timeOut>
</executeActivity>
</executeActivities>
<importActivities>
```

```
               <!-- Import state -->
               <importStateActivity>
                    <stateConfigFile>%ROOT_DIR%/UpperBlueNile/
                                          piOutputState.xml</stateConfigFile>
               </importStateActivity>
               <!-- Import HMS results-->
               <importTimeSeriesActivity>
                      <description>Import XML file</description>
                      <importFile>simulation.xml</importFile>
                      <timeSeriesSets>
                         <timeSeriesSet>
                         <moduleInstanceId>UpperBlueNile_HEC-
                                                HMS_historical</moduleInstanceId>
                         <valueType>scalar</valueType>
                         <parameterId>Q.simulated.historical</parameterId>
                         <locationId>900012</locationId>
                         <timeSeriesType>simulated historical</timeSeriesType>
                         <timeStep times="21:00"/>
                         <readWriteMode>add originals</readWriteMode>
                         </timeSeriesSet>
                      </timeSeriesSets>
               </importTimeSeriesActivity>
       </importActivities>
</activities>
</generalAdapterRun>
```