How to set up a Sobek RURAL Model in FEWS

How to set up a Sobek-RURAL Model in FEWS

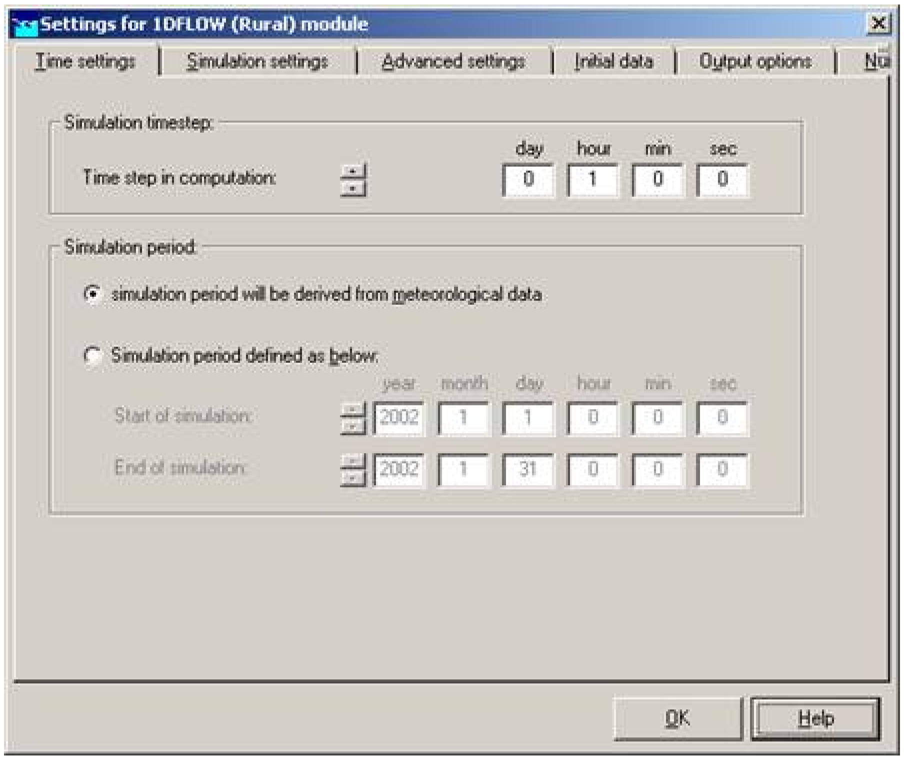
Needed:

- installation of SOBEK (incl license) in the root of the same partition of your hard disk (e.g. c: or d:) as where FEWS is run.
- SOBEK project (*.lit folder) in /Modules/SOBEK/[application name]/ *.lit
- SOBEK adapter: /Modules/SOBEK/[application name]/ SOBEK_Adapter.xml
- general adapter file: Config/ModuleConfigFiles/SOBEK_Forecast and SOBEK_Historical

Settings inside the SOBEK case

Simulation period and calculation time step
The simulation period needs to be triggered by the meteorological data. When running SOBEK from the User Interface there is a meteo-event file used. This file is called *.bui and is located by default in the SOBEK folder \SOBEK211\FIXED. The meteo event has a time step which will influence the calculation time step in the case. It is recommended to make this time step in the meteo event file larger than the desired calculation time step, to avoid SOBEK resetting your time step.

When running SOBEK under FEWS there is also a meteo event file needed. This file (e.g. Rainfall.xml) is exported by the general adapter-file. So even if there is no rainfall needed for running the SOBEK model, it needs to be exported for determining the meteo event and thus the simulation period.



Simulation timestep:

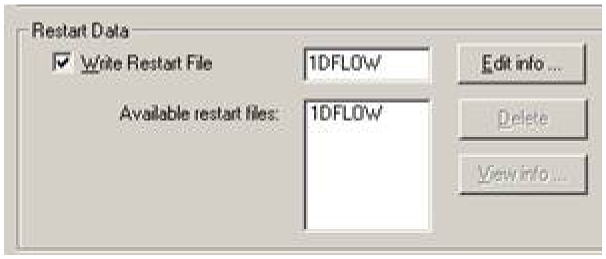
	day	hour	min	sec
Time step in computation:	0	1	0	0

Simulation period:

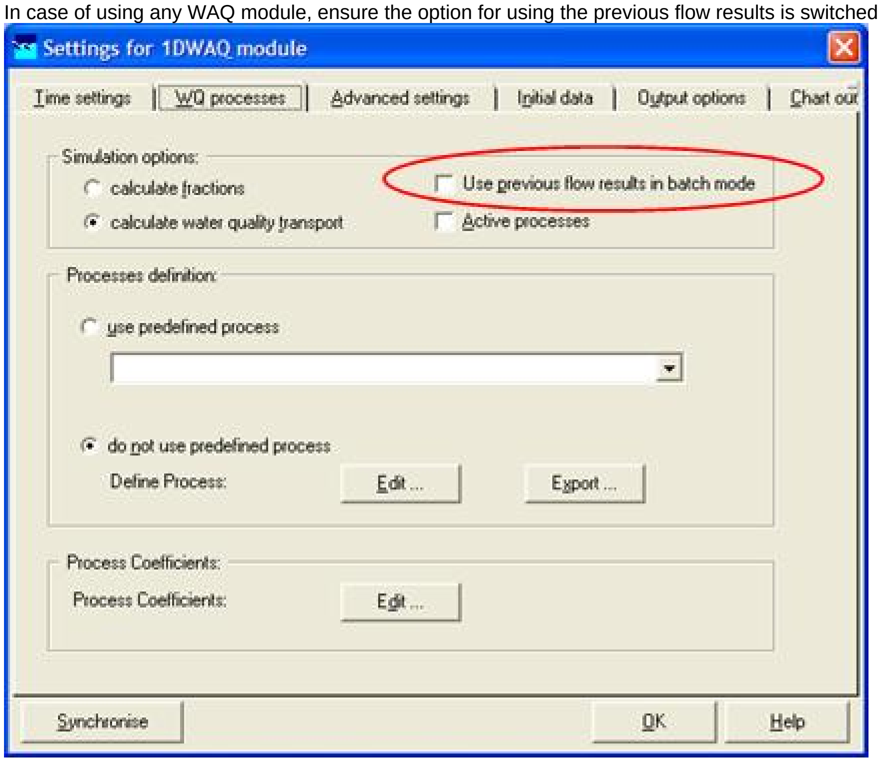
(● simulation period will be derived from meteorological data) ○ Simulation period defined as below:

	year	month	day	hour	min	sec
Start of simulation:	2002	1	1	0	0	0
End of simulation:	2002	1	31	0	0	0

Model state files
FEWS propagates the model state of the SOBEK model with the historical runs. Every next run is started from the last known model state, a warm state. If the model state is disregarded this means a cold state start is made. In both cases the model state is read from so called 'restart-files'. The SOBEK case needs to be instructed to write and read the same restart-files in the settings under the tab 'initial data'. You will receive a warning that the restart files have the same name and will be overwritten.
In the project folder (xxxx.lit) the restart files 1DFLOW.RDA, 1DFLOW.RDF and 1DFLOW.RDS can be found in the case, the folder starts with a number. In the folder xxxx.lit/FIXED the 2D restart files 2DFLOW.FDS and 2DFLOW.FLS can be found. These restart files all need to be copied into the ColdStateFiles/SOBEK_Historical default.xml.


Restart Data
[✓] Write Restart File: 1DFLOW
Available restart files: 1DFLOW

Water Quality calculation
In case of using any WAQ module, ensure the option for using the previous flow results is switched off in the SETTINGS task.

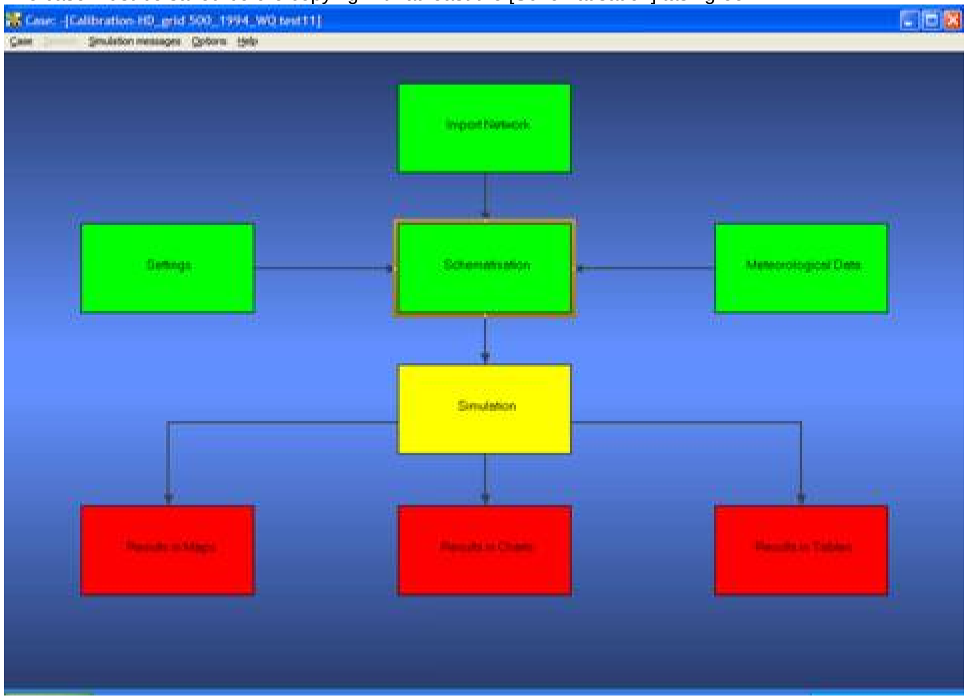

Settings for 1DWAQ module

Simulation options:
( ) calculate fractions
(•) calculate water quality transport
[ ] Use previous flow results in batch mode
[ ] Active processes
Processes definition:
( ) use predefined process
(•) do not use predefined process
Define Process:
Process Coefficients:
Process Coefficients:

Copying the Project folder
Before copying the project folder from the SOBEK program to the FEWS Modules directory, you can reduce the size of the complete case directory by removing the original results

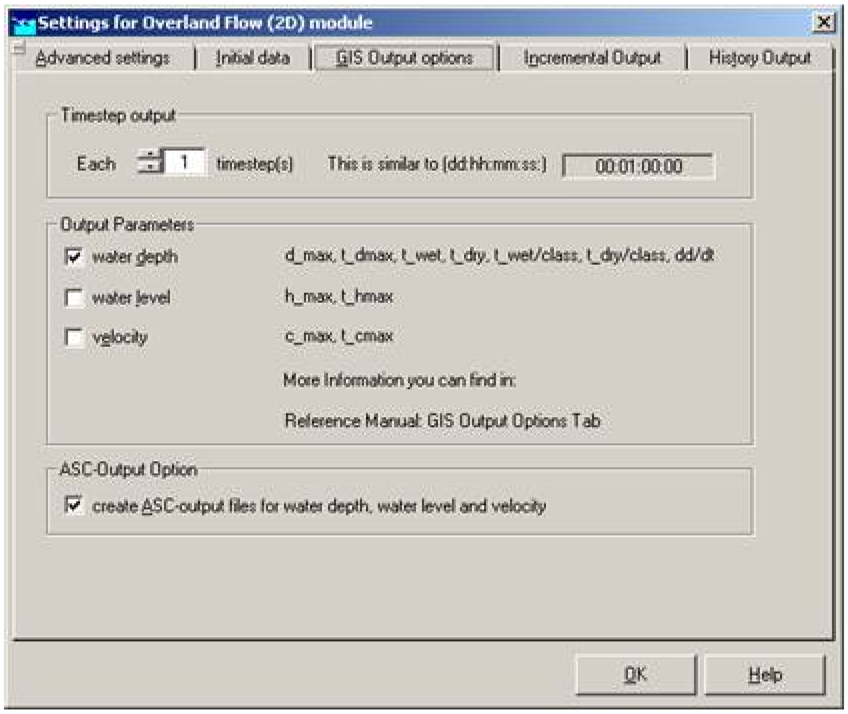
1. run the simulation task
2. cancel the simulation once started
3. press [Show permanent] button on error message box
4. delete all next files in work directory of the case
    a. out;.his;.map;.stu;.pst;.hia;.log;.prn;.hia;.inc;.asc;.mon;.lst
    b. 2DAREA.DAT
    c. 2DFLOW.DAT
    d. 2DSOBEK.CHZ
    e. 2DVELOC.DAT
    f. 2DVOLUME.DAT
    g. 2DPOINT.DAT
5. remove rubish data from the project directory
    a. boundary subdir
    b. data subdir
    c. initial S subdir
    d. Network subdir
    e. Many files in fixed subdir
    f. Many files in Bathymetry subdir

The case must be saved before copying with at least the [Schematisation] task green.



Schematisation
Note that the boundaries, laterals and controllers which data is being updated by FEWS should already been defined as time tables, both for Flow and WAQ modules!**

Output
To make sure the amount of output produced by SOBEK is no more than needed, it is important to uncheck all unnecessary output creation. If only water depth is needed in 2D then only have this option checked.



Timestep output

Each 1 timestep(s) This is similar to [dd:hh:mm:ss:] 00:01:00:00

Output Parameters

- [✓] water depth — d_max, t_dmax, t_wet, t_dry, t_wet/class, t_dry/class, dd/dt
- [ ] water level — h_max, t_hmax
- [ ] velocity — c_max, t_cmax

More Information you can find in:

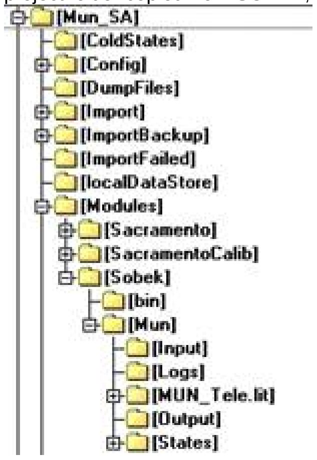
Reference Manual: GIS Output Options Tab

ASC-Output Option

- [✓] create ASC-output files for water depth, water level and velocity

Settings inside the SOBEK module directory

Inside the module directory SOBEK needs to have a /bin folder with *SBKbatch.exe* and *SBKbatch.ini* . Inside the other directory the folders Inputs, Logs, project folder copied from SOBEK, Output and States are needed. In the root of this directory the SOBEK_Adapter.xml is found.



```
[Mun_SA]
  [ColdStates]
  [Config]
  [DumpFiles]
  [Import]
  [ImportBackup]
  [ImportFailed]
  [localDataStore]
  [Modules]
    [Sacramento]
    [SacramentoCalib]
    [Sobek]
      [bin]
      [Mun]
        [Input]
        [Logs]
        [MUN_Tele.lit]
        [Output]
        [States]
```

SOBEK_Adapter.xml

In this file the adapter for SOBEK is configured.

General

The file describes in the 'general' section which project to use and which case to use inside the project.

variable	**required**	**default value**	**meaning**	**example**
version	yes		Version of Batch Data Structure	2
projectDir	yes		SOBEK project directory	BosRegge.lit
caseName	yes		Case name as in SOBEK project	Basismodel voor voorspelinstrument 101105, WL 20060404
showTablesFlipped	no	FALSE	Option to show all the input and output tables in the user interface flipped (true=horizontal, false = vertical)	TRUE
resultsHisFile	no		Output filename	Output\Output.his
dataPath	no		Path where all kind of info files are stored	Output
resultsSaved	no	FALSE	If the results have been saved and are availalbe	TRUE
parInfoFile	no		Filename of file with parameter information	Output\Output_par.csv
storeWorkDir	no	FALSE	Option to save the complete working directory	FALSE
workDirStored	no	FALSE	Option to determine if working directory has been saved	FALSE
storeOutputInOneFile	no	FALSE	Option to save all the output in one output file, which is required for FEWS	TRUE
storeFEWSState	no	FALSE	Option to save state information, which is required for FEWS	TRUE
fewsDiagnosticsFile	no		Filename of FEWS diagnostics file	Logs\Diagnostics.xml
clearWorkingDirectories	no	TRUE	Option to clear working directory after running the simulation	FALSE
skipTimeSteps	no	0	Number of timesteps to be skipped when converted FEWS ASCII Meteo file to Bui file.	0
useXMLBinFileType	no	FALSE	Option to write Delft-FEWS PI files with binary data section. Improves performence very much, but bad to check the contents of the PI files. Only use this if everything is going right.....	TRUE
useParsenUpdateMode	yes	FALSE	Option to let the PARSEN of the Flow modules only update the time series in the NEFIS files. Might improve performence very much, but does not pick up any other updated data from the MDB Files	FALSE
parsenMinimumTimeSeriesLength	yes	1000	If useParsenUpdateMode is TRUE, it is required to define the number of timesteps to generate in the time series tables. Value must be higher than zero.	1000
parsenSourceFiles	yes		If useParsenUpdateMode is TRUE, the source files which should be updated by the PARSEN must be defined here.	Defaults\sobek.*

NB: The elements parsenMinimumTimeSeriesLength, parsenSourceFiles are only required in case useParsenUpdateMode is used. So use either all three, or none of them. In case you have useParsenUpdateMode = TRUE, the PARSEN (SOBEK 1D Flow pre-processor) will not completely generate the NEFIS files sobek.mda and sobek.mdf, but will only update the time series in the file. This improves the performance for large networks.

postProcessingActivities

In the 'postProcessingActivities' the 2D map outputs from SOBEK are moved Output folder and the restart files written to the States folder. Please note that whether they are read by FEWS depends on the generalAdapter file.

parameters

In 'parameters' it is described which timeSeries need to be translated to which SOBEK files and how they should be interpreted.
Notice that updating of time series in SOBEK files should be defined as follows, where the **description** is very important:

- comment: has no meaning, can be any string just for your reference
- description: Description of sub parameter, previously used as Sobek object ID
- sobekObjectId, like laternal node ID, boundary node ID, controller ID
  If not defined the description element is used (as it was in the older days)
- locationID : is the location ID as used in the PI XML time series file
- parameterID : is the parameter ID as used in the PI XML time series file
- type: type of data or time series. See the table below to select the type of parameter to select.

Please note that the rainfall, evaporation, wind, wqmeteo time series should be all in one separate single XML file, so one for all rainfall series, one for all evaporation series and one for all wind series (speed and direction both in one file).
Rainfall series are always converted independent on the parameterID, where all locationIDs are converted to meteo stations. Evaporation works the same.
Regarding wind series, the adapter looks for the following time series:

- there should be at least two time series, one containing the wind speed and one with the wind direction
- if a time series parameter ID contains the string "speed" (not case sensitive), this series is used as a wind speed series
- if a time series parameter ID contains the string "dir" (not case sensitive), this series is used as a wind direction series
- the location IDs are used to identify different wind fields
- the first defined wind field (locationID) is always used as global wind field.
- in case more than one different wind fields (location IDs) all series are also written as additional wind fields, next to the global wind field.
- The resulting wind file is always converted to the corresponding WQ Wind file (*.qwt), but only containing the global wind field.

Example of wind:
There are 6 time series in the file wind.xml

series	locationId	parameterId
1	Lauwersoog	Wind.Speed
2	Delfzijl	Wind.Speed
3	Harlingen	Wind.Speed
4	Lauwersoog	Wind.Direction
5	Delfzijl	SpeedWind.Direction
6	Harlingen	Wind.Direction

These series will be interpreted as:

field	wind speed	wind direction
global, using Lauwersoog (the first station)	1, (Lauwersoog, Wind.Speed)	4, (Lauwersoog, Wind.Direction)
additional field for again Lauwersoog	1, (Lauwersoog, Wind.Speed)	4, (Lauwersoog, Wind.Direction)
additional field for Delfzijl	2, (Delfzijl, Wind.Speed)	5, (Delfzijl, Wind.Direction)
additional field for Harlingen	3, (Harlingen, Wind.Speed)	6, (Harlingen, Wind.Direction)

Another example of wind:

There are 2 time series in the file wind.xml

series	locationId	parameterId
1	Lauwersoog	Wind.Speed
2	Lauwersoog	Wind.Direction

These series will be interpreted as:

field	wind speed	wind direction
global, using Lauwersoog (the first station)	1, (Lauwersoog, Wind.Speed)	2, (Lauwersoog, Wind.Direction)

In this example no additional wind fields are written.

parameter types

type	description
undefined	not allowed as type
rainfallfile	converts full contents of the file into RR rainfall file. All series should have same time stamps. File name will be *.bui and will be written in cmtwork. Case registry will be updated, so the file will be used in the computation
rrrestart	rr restartfile (state file)
flowrestart	flow restart (state file)
flow_hc_boundary	H boundary with constant value
flow_ht_boundary	H boundary with timeSeries
flow_qc_boundary	Q boundary with constant value
flow_qt_boundary	Q boundary with timeSeries

flow_htref_boundary	H boundary with timeSeries from ref.table
flow_qtref_boundary	Q boundary with timeSeries from ref.table
windfile	wind, should be a single PI file
flow_latndc	lateral flow on connection node, constant value
flow_latndt	lateral flow on connection node, time series
flow_latndref	lateral flow on connection node, time series from ref.table
flow_latbrc	lateral flow on node at branch, constant value
flow_latbrt	lateral flow on node at branch, time series
flow_latbrref	lateral flow on node at branch, time series from ref.table
flow_latdic	diffuse lateral flow on branch, constant value
flow_latdit	diffuse lateral flow on branch, time series
flow_latdiref	diffuse lateral flow on branch, time series from ref.table
flsboundline_hc	H line boundary on grid, constant value
flsboundline_ht	H line boundary on grid, time series
flsboundline_qc	Q line boundary on grid, constant value
flsboundline_qt	Q line boundary on grid, time series
flsboundnode_hc	H boundary node on grid, constant value
flsboundnode_ht	H boundary node on grid, time series
flsboundnode_qc	Q boundary node on grid, constant value
flsboundnode_qt	Q boundary node on grid, time series
flsinitpoint	initial condition point at grid
rr_up_seepdef	rainfall-runoff seepage definition ID, (no timeSeries!)
rr_up_stordef	rainfall-runoff storage definition ID, (no timeSeries!)
rr_up_alfadef	rainfall-runoff alfa drainage coeff. definition ID, (no timeSeries!)
rr_up_infdef	rainfall-runoff infiltratie definition ID (no timeSeries!)
rr_up_initdef	rainfall-runoff initial definition ID (no timeSeries!)
rr_up_ernstdef	rainfall-runoff ernst drainage coeff. definition ID, (no timeSeries!)
rr_insodemandtable	rainfall-runoff industrial demand time series, like a lateral flow. Note: a dummy salt concentration of 0 mg/l will be used!
rr_insodischargetable	rainfall-runoff insdustrial discharge time series, like a lateral flow
evapfile	converts full contents of the file into RR evaporation file. All series should have same time stamps. File name will be *.evp and will be written in cmtwork. Case registry will be updated, so the file will be used in the computation
rtc_par3	RTC, parameter3 record ID
rtc_rule	RTC, rule record ID
rtc_maxf	RTC, maxf record ID
rtc_engd	RTC, engd record ID

flow_timecontroller_tbl	1D flow structure time controller, time series
flow_intervalcontrollersetpoint_tbl	1D flow structure interval controller setpoint, time series
flow_controllerpidsetpoint_tbl	1D flow structure PID setpoint, time series
1dwq_fraction_concentration_tbl	1d waq substance concentration for fraction type, time series
1dwq_boundary_concentration_tbl	1d waq substance concentration for local boundary, time series
2dwq_fraction_concentration_tbl	2d waq substance concentration for fraction type, time series
2dwq_boundary_concentration_tbl	2d waq substance concentration for local boundary, time series
fls_restart	fls restart / state
1dwq_restart	1d waq restart / state
2dwq_restart	2d waq restart / state
1d_flow_dam	1d flow weir object, representing a obstruction in the river
1d_flow_dambreak_hc_boundary, 1d_flow_dambreak_connectionnode	do exactly the same: create a breach in the dike
rr_temperature	temperature for the RR module, should be a single PI file
1d_flow_lateral_dat_file	Converts the complete contents of a PI xml file into a 1D flow lateral.dat file. The PI locationID should be used to map to the lateral IDs (any type). The parameter (and qualifier) is not used. All series in the XML should exist as definition with a time series table in the SOBEK lateral .dat file. Vice versa is not needed, as SbkBatch only updates the definitions.
rr_boundarytable	rainfall-runoff boundary node, water level time table. Note: a dummy salt concentration of 0 mg/l will be used!
1d_flow_boundary_dat_file	Converts the complete contents of a PI xml file into a 1D flow Boundary.Dat file. The PI locationID should be used to map to the boundary IDs. The parameter (and qualifier) is not used. All series in the XML should exist as definition with a time series table in the SOBEK boundary.dat file. Vice versa is not needed, as SbkBatch only updates the definitions.
1d_flow_control_def_file	Converts the complete contents of a PI xml file into a 1D flow control.def file. The PI locationID should be used to map to the control def IDs (of time controller type). The parameter (and qualifier) is not used. All series in the XML should exist as definition with a time series table in the SOBEK control.def file. Vice versa is not needed, as SbkBatch only updates the definitions.
1d_flow_profile_def	changes the reference to a cross section definition in profile.dat. The sobekObjectId should refer to the record in the profile.dat, while the classFile(s) refer to the records in the profile.def. Typically used in batching runs with varying cross sections per run.
greenhouse_water_usage_file	Convert the complete contents of a PI xml file into the \sobek\fixed\3b\KasGebrData.Dat file, which is referred to in the sbkbatch.ini key [General]KasGebrDataFile=.... In delft_3b.ini [Options]NewFormatKasdata should be -1 and greenhouse node definitions should be setup to using this new format (which is not supported by the GUI). All series in the XML are converted to update existing definitions in the KasGebrData.Dat. Note that this file is not register to in the case registry ,so updating this file may affect other cases too. Only definition with single column tables are supported. The PI locationID should be used to map to the kasgebr IDs . The parameter (and qualifier) is not used. All series in the XML should exist as definition with a time series table in the SOBEK kasgebr file. Vice versa is not needed, as SbkBatch only updates the definitions.
rr_runoff_rnf	Converts the complete contents of a PI xml file into a *.rnf file. Can only be used in case also a rainfallfile is used, so sbkbatch can write it to the same base file name as the RR module uses that.
rr_openwater_target_level_timetable	Converts one individual timeseries into a OW_T table in openwate.tbl

rr_openwater_target_level_timetable_file	Converts a complete PI xml (so all timeseries) into a OW_T tables in openwate.tbl, based on locationid
rr_timecontroller_tbl_file	Converts a complete PI xml (so all timeseries) into a INST tables in struct3b.tbl, based on locationid
1d_flow_trigger_def_file	Converts a complete PI xml (so all timeseries) into a CNTL tables in control.tbl, based on locationid
wq_meteofile	Converts a complete PI xml (so all timeseries) into a Delwaq format file (based on parameterid, locationid is ignored). This file is called wqmeteo.mwq and is included in the delwaq wq computation
1dwq_boundary_concentration_timetbl_file	Converts a complete PI xml (so all timeseries) into ..\work\boundwq.dat, using location and parameter IDs. Note that (unlike with other complete file conversions, like 1d_flow_control_def_file etc) the file is fully converted only and not updated! That means that definitions which are not in the PI xml file, not will be converted.
2dwq_boundary_concentration_timetbl_file	Converts a complete PI xml (so all timeseries) into ..\work\2DBNDWQ.DAT, using location and parameter IDs. Note that (unlike with other complete file conversions, like 1d_flow_control_def_file etc) the file is fully converted only and not updated! That means that definitions which are not in the PI xml file, not will be converted.

Flow_dam and Flow_dambreak should be triggered by time series with non-missing values. The flow_dam should have timeseries where the location id is equal to the reachsegment id. The adapter creates the weir/dam at this segment. The time series values represent the height of the weir. This is automatically added to the bed level to get a correct crest level of the weir.

The dambreaks can be created at all node types that are calculation points (fixed and regular calculation points, connection nodes and linkage nodes). The time series that trigger a dambreak should contain by default missing values, but a -1 for a dambreak at the left side, a +1 for a dambreak at the right side. The adapter creates a new reach to represents the dambreak. At this reach a weir structure is placed, with a crest level that will be lowered at the start of the dambreak.

Notice that using dams or dam breaks will make that the restart files do not fit anymore with the network and will be neglected.

The flow_lateral_dat_file has some special features. This routine reads the network.cn file to know the type of the laterals (FLNO, FLBR, FLDI) and then creates a complete new lateral.dat file, based on the series PI-XML file. In case a time series is not in the XML file, the original definition is re-used. However, this method is not very efficient so this whole flow_lateral_dat_file routine is most efficient in case you convert ALL the laterals. The locationID is used as the required lateral id. The parameterId is not used, although you can use it to parse the interpolation option to the definition. Therefore include the string "block" in to the parameterId, so the routine knows that the block interpolation option should be used. Default interpolation option is linear. It is of course (in operational systems) not possible to define a periodicity.

Jobs

In the 'jobs' section nothing needs to be changed.
It should be:

```
<jobs>
  <job>
    <jobName>Sobek historical run</jobName>
    <storeDir/>
    <parameter>
      <classIndex>1</classIndex>
    </parameter>
  </job>
</jobs>
```

Output

In the 'output' section the output locations need to be identified. Moreover the SOBEK_Adapter needs to know in which file to read the output and to which PI-file to write the output so that the generalAdapter can read it.

Type of output	corresponding outputfile
water level, depth on nodes	calcpoint
flow on branches, discharge/velocity	reachseg
results at structures, discharge/crestlevels, waterlevels up&down	struc
Total Waterbalance, Flow	qwb
RR open water levels	ow_lvldt

RR unpaved runoff	upflowdt
RTC decision parameters	rtcparal
1D Wq results: .his for History stations, .map for whole area	delwaq.his and delwaq.map
RR boundary flows	bndflodt
Overlandflow History stations, Velocity, Waterlevels/depth	flshis

mapStack

In the 'mapStack' part the 2D maps are put together in a so called map stack. The generalAdapter will import these map stacks as a grid timeSeries.

Example:

```
<mapStack>
    <fileName>output\dm1dmapstack.xml</fileName>
    <fileMask>dm1d????.asc</fileMask>
    <geoDatum>Rijks Driehoekstelsel</geoDatum>
    <locationId>map</locationId>
    <parameterId>InundationDepth</parameterId>
    <timeStep>unit="hour" multiplier="1"</timeStep>
    <module>2dflow</module>
    <mapStackActive>true</mapStackActive>
  </mapStack>
```

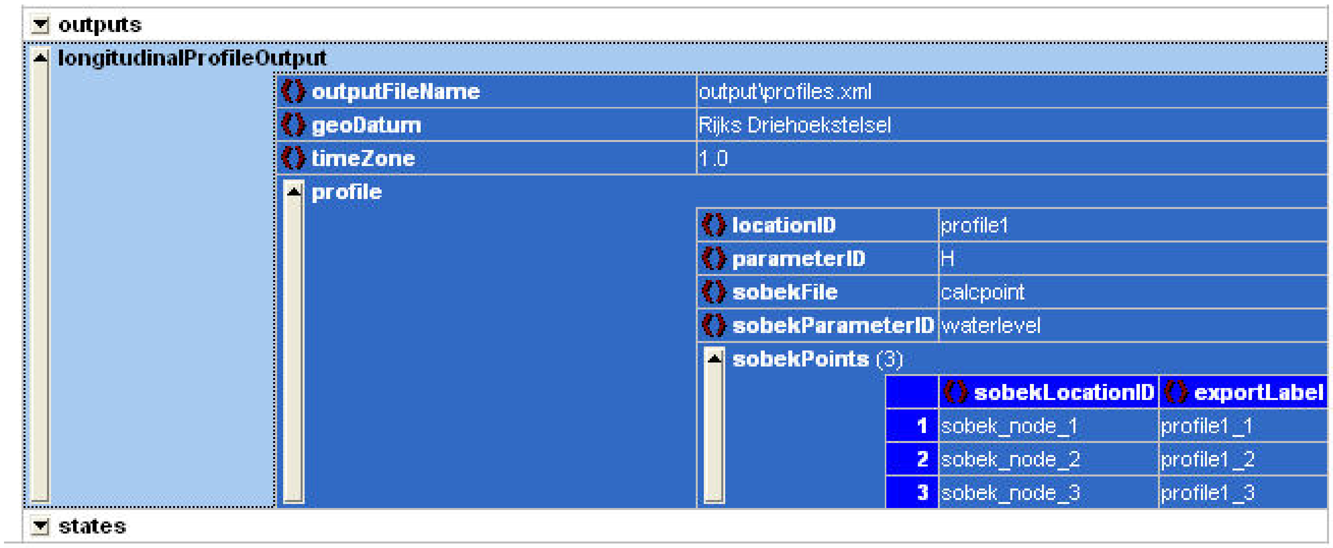
States

The last section in the SOBEK_Adapter.xml file is 'states'. This describes the read locations and writelocations of the model states. This is similar to other external modules.
RR states: * rrr
1D states: *.rda, *.rdf
2D states: *.fls
WQ states: *.res

Example:

```
<states>
  <stateXmlOutputFile>States\Outputstates.xml</stateXmlOutputFile>
  <state>
    <readLocation>States\Input.rda</readLocation>
    <writeLocation>States\Output.rda</writeLocation>
  </state>
  <state>
    <readLocation>States\Input.rdf</readLocation>
    <writeLocation>States\Output.rdf</writeLocation>
  </state>
  <state>
    <readLocation>States\Input.fls</readLocation>
    <writeLocation>States\Output.fls</writeLocation>
  </state>
  <state>
    <readLocation>States\1Dlevels-in.xyz</readLocation>
    <writeLocation>States\1Dlevels-out.xyz</writeLocation>
  </state>
  <state>
    <readLocation>States\Input.rrr</readLocation>
    <writeLocation>States\Output.rrr</writeLocation>
  </state>
</states>
```

Longitudinal profiles



- outputs
  - longitudinalProfileOutput
    - outputFileName: output\profiles.xml
    - geoDatum: Rijks Driehoekstelsel
    - timeZone: 1.0
    - profile
      - locationID: profile1
      - parameterID: H
      - sobekFile: calcpoint
      - sobekParameterID: waterlevel
      - sobekPoints (3)

	sobekLocationID	exportLabel
1	sobek_node_1	profile1_1
2	sobek_node_2	profile1_2
3	sobek_node_3	profile1_3

- states

GeneralAdapter

The generalAdapter is similar to other generalAdapter files. For the exact syntax inside the *executeActivities* part of the file always copy an existing SOBEK generalAdapter. A special point of attention is that there should always be a meteo time series exported to SOBEK. This file will determine the start and end time of the run.

Please make sure the GA run is configured to set the missing value definition in the XML files it exports to -999.99 and not the default NaN. This should be set in general section, please see the example below. If NaN is used as missing value definition the SOBEK adapter will complain about missing value when there is 0.0 precipitation.

```
<general>
  <rootDir>$REGION_HOME$/Modules/Sobek/Singapore</rootDir>
  <workDir>%ROOT_DIR%</workDir>
  <exportDir>%ROOT_DIR%/Input</exportDir>
  <exportIdMap>IdExportSobek</exportIdMap>
  <importDir>%ROOT_DIR%/Output</importDir>
  <importIdMap>IdImportSobekForecast</importIdMap>
  <dumpFileDir>$GA_DUMPFILEDIR$</dumpFileDir>
  <dumpDir>%ROOT_DIR%</dumpDir>
  <diagnosticFile>%ROOT_DIR%/Logs/diagnostics.xml</diagnosticFile>
  <missVal>-999.99</missVal>
  <convertDatum>true</convertDatum>
  <timeZone>
    <timeZoneOffset>+08:00</timeZoneOffset>
  </timeZone>
</general>
```

How to run a model in Ensemble mode?

The default method for running in ensemble mode is that all runs are performed sequentially. There are two alternatives to running the models simulatiously to reduce the overall time:

1. run the model in a directory per ensemble member with default settings (SbkBatch uses the Case Management)
2. run the model in a temporary directory without using the Sobek Case Management.

The first option is the preferred one as it guarantees that all Case Management is correct. However, it requires quite a lot of configuration as all the possible workingdirectories have to be pre-configured. Instead of one, for all the ensemble members a working directory should be prepared. In the General Adapter you should link to these directories by using
<rootDir>$REGION_HOME$/Modules/Sobek/Singapore_%ENSEMBLE_MEMBER_INDEX%</rootDir>.
See also 19 Parallel running of ensemble loops and activities on one forecasting shell instance.

The second option is more advanced but faster and especially useful for small models where the CMT activities are a significant part of the simulation time. Next to that the configuration is much easier as only one temp directory has to be specified.
In the General Adapter you specify only <rootDir>%TEMP_DIR%</rootDir> where the TEMP_DIR will be defined as the system TEMP_DIR (or defined in the global.properties) and an added unique subdirectory. In case your system TEMP_DIR variable is "c:\windows\Temp" the General Adapter will use as TEMP_DIR in each particular instance of the General Adapter (i.e. per ensemble member) TEMP_DIR = "C:\windows\temp\<FSS><*ModuleInstance*><ENSEMBLE_MEMBER_INDEX>_<threadId>."
In this configuration it is quite useful to have no CMT activities anymore. Therefore define in the SBKBATCH.INI the key [General] RunFromTempDirWithoutCMT=-1. Then you need to configure:

- a CMTWORK and WORK directory where all the files are available already
- FNM files in the CMTWORK that has only relative paths or links to files that are not specific to the actual run (like language files)
- use as [General]SimulateCall and [General]SimulateCallRtnFile directly the required program or a batch file that runs multiple programs. For example if you need to run only the RR module, you define:

  ```
  [General}
  SimulateCall=\Sobek212\Programs\3b\sobek_3b.exe sobek_3b.fnm sobek_3b.rtn
  SimulateCallRtnFile=sobek_3b.rtn
  ```

- save the whole setup in a moduleDataSet that will be exported before starting the simulation

Notice that this functionality is available only since SbkBatch.exe version 2.1.0.112

Installation and latest version

The adapter software is installed as part of the SOBEK installation in the Programs folder. However, due to various features extensions, a later version of the SbkBatch.exe (and the corresponding sobekAdvancedBatch.xsd) can be needed. Therefore the latest version of both can be found here:

- sobekAdvancedBatch.xsd
- SbkBatch.exe