Applications of the new simplified approach to model obstacles: weirs, culverts and virtual ponds

Sven Smolders[1]; Sébastien E. Bourban[2]
sven.smolders@mow.vlaanderen.be
[1] Flanders Hydraulics, dept. of Mobility and Public Works, Flemish Government, Berchemlei 115, 2140 Antwerp, Belgium
[2] EDF R&D LNHE / LHSV, 6, Quai Watier, 78400 CHATOU, France

***Abstract* – A new simplified approach has been implemented in the openTELEMAC system to model most types of hydraulic structures like weirs, barriers, bridges, culverts, storage ponds, etc. In the code, these structures are all considered to be obstacles of flow; hence, the name of the new main Fortran module is obstacles.f. This newly added code wants to generalise the way hydraulic structures are handled and is based on the abstraction of "obstacle." This approach also enhances more unified and simplified user input files that have basically the same structure for every type of obstacle.**

The openTELEMAC system's often simple validation examples are an excellent starting point for end users to explore new code functionalities. This work explores, however, the possibilities of applying the new obstacle code to more complicated, real-life case studies. Weirs, culverts, and the use of virtual ponds are looked into. Weirs are connected to virtual ponds and their crest elevation is controlled by the upstream water level or by a user-given time series. For culverts, the existing TELEMAC-3D validation example "Bergenmeersen" is now also available in 2D. For both 2D and 3D, a version is made to work with the new obstacle code. The 2D version also works with the flood control area and overflow dike being replaced by a virtual pond and weir, respectively.

Keywords: Obstacles, weir, culvert, virtual pond, Dender river, Kakhovka dam break, Bergenmeersen.

I. INTRODUCTION

In 2023, the new simplified approach for implementing hydraulic structures or obstacles in TELEMAC-2D and -3D was presented by [1]. After a year of further development in the Searobin branch, the code is ready to be introduced in the next openTELEMAC release, i.e. V9P0r0. In the new approach, the geometrical information of any obstacle is automatically connected to the information of the model mesh, like node numbers. Connection between obstacles is also made automatically. Furthermore, the physical and operational characteristics of any obstacle are defined within a single file. Even if there are more than one or many different kinds of obstacles, like weirs and culverts. The code includes simple examples of (a) porous and solid barrier blocking; (b) a weir; (c) a storage pond which can be connected to one or more obstacles; (d) a control location which can be used to automate the operation of an obstacle; and (e) a culvert, for which the flow tunnels through the length of the obstacle. Where [1] presented the working principles, introduced the approach and gave information on the 5 examples included within the implementation, this paper focuses on the practical application of the new simplified approach to model obstacles and tries to give some more real life examples on how to use the new approach. These examples were/are also being used to further debug the code and prepare it to be ready for the next openTELEMAC software version release.

All applications included at this moment in the code, i.e. weirs, culverts, virtual ponds and control locations, are used in the examples in this paper. Only the "solid barrier" obstacle is not used. Section II introduces a simplified version of the weirs example and varies the different configurations of the mesh. Section III describes a 2D model of a section of the Dender river in Belgium where three weirs regulate the water level. The crest elevation of these weirs is controlled by their upstream water level. Section IV shows a model of the Kakhovka dam break in Ukraine. The dam is represented as a weir from which the crest elevation was controlled by a time series. The breaking of the dam is modelled by a significant lowering of the crest elevation. Section V introduces culverts. The existing "Bergenmeersen" testcase of TELEMAC-3D is also applied to 2D and is combined with the implementation of a virtual pond and a weir to replace flood control area itself and the overflow dike, respectively.

All examples in this paper focus on the application of the new simplified approach to model obstacles. Calibration of the models or obstacle sizes and/or operations is not shown here. The obstacle code is still under development and in debugging stage, to which this paper contributed. Therefore, the final implementation of the code into the official release of the openTELEMAC system might contain slight differences to what is shown here.

II. WEIRS: SIMPLIFIED TESTCASE

A. Simple testcase in TELEMAC-2D and TELEMAC-3D

When a user wants to become acquainted with a new functionality of code in the openTELEMAC system, the validation examples are a good place to start. The Dender River case presented in the next section contains three weirs. The weirs were operational in the V8P5 version of the openTELEMAC system, but not with the new obstacle code. To determine whether it was possible to use the obstacle code for this model, tests were first performed on a simplified version of the weirs testcase already provided in the openTELEMAC system, which was also adapted for use with obstacles as described in [1].

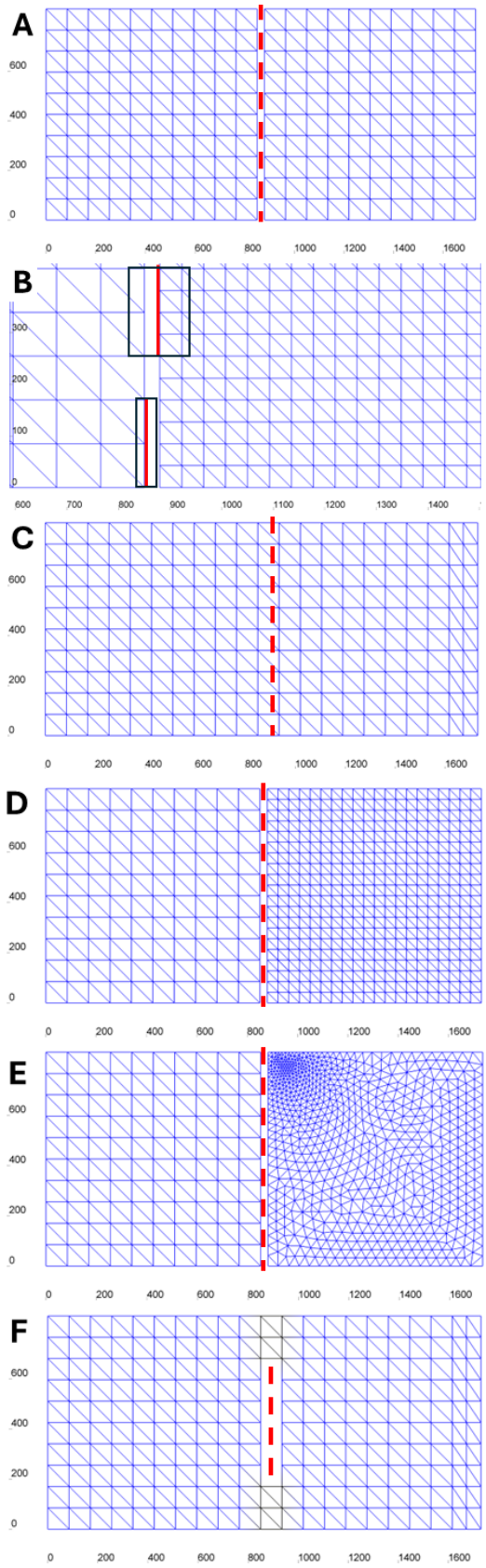

Figure 1. Simplified weir test case: different setups for the mesh and weir

The testcase was reduced to just two ponds connected by a single weir, as opposed to four ponds connected by three weirs. The two ponds have a structured mesh and are depicted in Figure 1A. The red dashed line represents the connecting weir. The nodes upstream and downstream of the weir are neatly aligned, and both ponds have the same mesh resolution. Water enters the model through a discharge boundary on the left side. A constant water level is established as the boundary condition on the right side of the right pond. Water can flow over the weir and into the right pond, leaving the model domain on the right side, once the left pond is filled to the crest elevation of the connecting weir.

There are just two keywords that need to be added to the steering file. The weir's geometrical information is contained in a file in the i2s Blue Kenue format, which can be accessed by using the keyword FILE OF GEOMETRIES FOR THE OBSTACLES. A file of this type is shown in Figure 18. The first number of the line indicates by how many points the obstacle is represented. A weir is described by only two points. The next number in the same line describes the width of the obstacle. In the simple weir testcase the width must be greater than the distance between the two ponds. The width is split equally over the up- and downstream side of the weir, so if it isn’t exactly in the centre, the largest distance between it and a pond must be doubled and entered as the weir width in the geometry file. Examples of this are given in Figure 1B. The weir's name, such as "weir 1", appears after the width number. The x and y coordinates of the two points that characterise the weir obstacle are provided in the following two lines.

The name of the weir, “weir 1” connects the geometry data of the obstacle to the control data for the obstacle. The control data is given in the FILE OF SETTINGS FOR THE OBSTACLES. This is the second and last keyword/file that is needed in the steering file to operate all obstacles in the model. An example is shown in Figure 19. The physical characteristics of the obstacles are set in this control file using keywords similar to those found in the steering file. The keyword NUMBER OF OBSTACLES DEFINED AS WEIRS or NUMBER OF LINKAGES DEFINED AS WEIRS must be used to indicate the number of weirs in the basic weir testcase. Next, a few keywords must be provided for each obstacle, in this case, a weir. The weir's crest elevation is determined by CREST ELEVATION. The weir's length is given by the keyword OPEN WIDTH, while its discharge coefficient is set by DISCHARGE COEFFICIENT. The discharge over the weir is calculated using the latter. The coordinates of the two points in the geometry file, that describe the location of the weir, do not determine the length.

ELEMENTS MASKED BY USER is another keyword that can be added to the steering file. An obstacle can be drawn over an existing mesh and does not require a space between mesh segments, as shown in Figure 1C. In this case the weir will possibly mask some of the elements in the mesh. This also occurs when the weir is located between two ponds, but the mesh resolutions on both sides differ, as shown in Figures 1D and E. Unfortunately, TELEMAC does not handle this masking well, leading to instabilities and errors. As a result, masking should be avoided at this stage of development to

ensure a smooth simulation. This makes it difficult for the end user to set a good weir width, especially when both sides of the weir have a very different mesh resolution. A solution to this problem is being investigated, and it requires the code to automatically select the necessary nodes on both sides of the given weir while taking mesh resolution into account.

Finally, a weir was tested as an obstacle within a hole in the mesh, as shown in Figure 1F. Prior to the implementation of the obstacles code, weirs had to be modelled as a hole in the mesh with aligned mesh nodes on both sides. The Dender river model in the following section contained three of these types of weirs, and testcase 1E investigated whether the obstacle code could simply replace the old method of modelling weirs without modifying the original mesh. And it works fine.

All test cases were run in TELEMAC-2D and -3D to assist with code debugging. Figure 2 shows an example of such an encountered bug. Increasing the mesh resolution in the receiving pond on the left resulted in a structured increase in tracer values at uneven nodes. For the 3D versions of the testcases, when the mesh resolution or shape changed, the time step had to be reduced for stability reasons. The simulations did not crash due to instabilities, but they did cause strange behaviour in the result files.

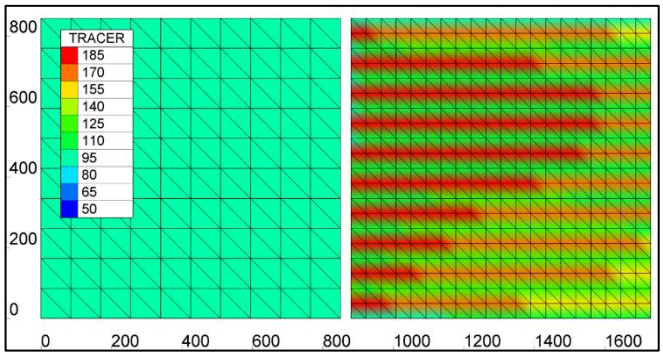

Figure 2. Example of a bug in tracer handling when the mesh resolution downstream the weir was doubled.

With the experience gained from these simple testcases and some bugs resolved, the code was ready for the first more complex real-world testcase.

III. WEIRS: CREST LEVEL CONTROLLED BY UPSTREAM WATER LEVEL, THE DENDER RIVER CASE

A. *Dender river*

The Dender River is a tributary of the Scheldt Estuary in Belgium. The river is 60 kilometres long, and a weir and sluice complex prevents Scheldt tides from entering it. Numerous weirs and sluices can be found along the river. The Dender river valley is known to be prone to flooding. The last major flooding event occurred in 2010. This event marked the start of a completely new flood mitigation strategy for the Dender. The plan includes rebuilding the river's weir and sluice structures. The weirs were originally built in 1869 and are still functional today.

The weirs are made of thick wooden beams that must be manually hoisted out of the construction. Figure 3 depicts the complex of two weirs and one sluice at Geraardsbergen. The picture was taken looking upstream. The weir on the right, located beneath a square building, is known as the large weir or "weir 1" in the Telemac model. Figure 4 shows a schematic overview of this weir. The weir is 4.45 meters long and made up of six wooden beams of varying heights. From the lowest to the highest beam, the heights are 0.78 m, 0.63 m, 0.97 m, 0.82 m, 0.25 m, and 0.25 m, respectively. The bottom elevation is 13.30 m TAW (the Belgian vertical reference level, which corresponds to mean sea level at low water on the Belgian coast), while the maximum crest elevation is 17.00 m TAW. Figure 4 shows a closer look at the smaller weir ("weir 2" in the Telemac model) on the left side of Figure 3. This weir is made up of four parts, each measuring 1.45 meters in length. Each part is made up of two wooden plates that are the same height, or 1.20 m. The bottom of the weir is located at 14.45 m TAW. The crest elevation is adjusted by manually hoisting wooden beams or plates from these weirs. The desired water level 300 meters upstream of the weirs is set to 16.85 m TAW.

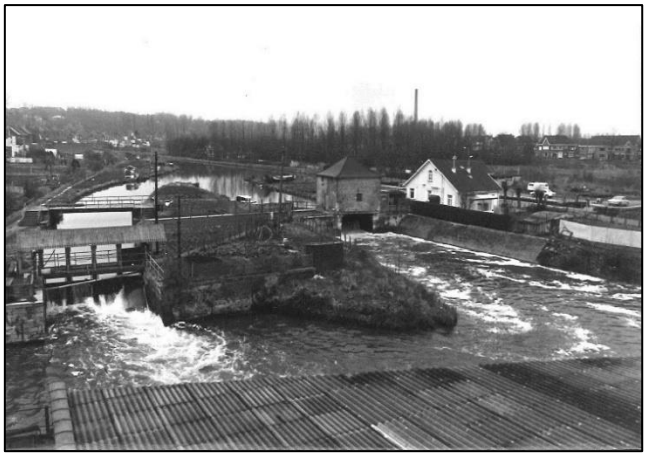

Figure 3. Weirs on the Dender at Geraardsbergen: left the small weir (weir 2 in the model) and right under the square house is the big weir (weir 1 in the model). (picture taken in 01/01/1978 source: [2])

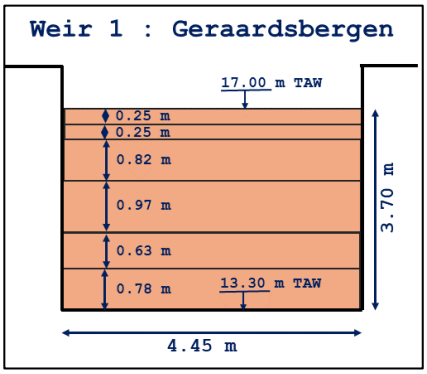

Figure 4. Schematic representation of weir 1, the larger weir at Geraardsbergen.

The third weir is located downstream at Idegem and is also located underneath a building. It consists of two parts with a length of 3.95 m and 4.45 m. Both parts have five wooden

beams to control the crest elevation. The beam at the bottom has a height of 1.2 m and the four other beams above have all the same height of 0.70 m. The weir's bottom elevation is 11.59 m TAW and the maximum crest elevation is 15.59 m TAW. The desired water level 600 m upstream from this weir is 15.23 m TAW.

B. Dender river model

The model domain is shown in Figure 6. The mesh contains 55355 nodes. To estimate the extent of flooding, the river valley's floodplain is incorporated into the model domain. The mesh resolution varies from 1.5 m in the river to more than 60 m on the floodplains. For the river the bathymetry data from 2012 was used and for the floodplain a digital elevation model with data from 2014 [3] was used. For this exercise the Manning bottom friction coefficient in the river was set to 0.02 $s/m^{1/3}$ and on the floodplains to 0.03 $s/m^{1/3}$. Figure 6 shows the location of the weirs with the holes in the mesh and the location of two control points upstream. The latter will be used to automate the crest elevation of the weirs. The model has an upstream discharge boundary and a downstream water level boundary. The data for both boundaries is given in Figure 7. The first nine days have normal discharge and water level data. After nine days the data show the flood event that lasts around 10 days.

Figure 5. Detailed view on the wooden boards of the small weir at Geraardsbergen. Chains are attached to hoist the boards. (picture taken 29/04/2016)

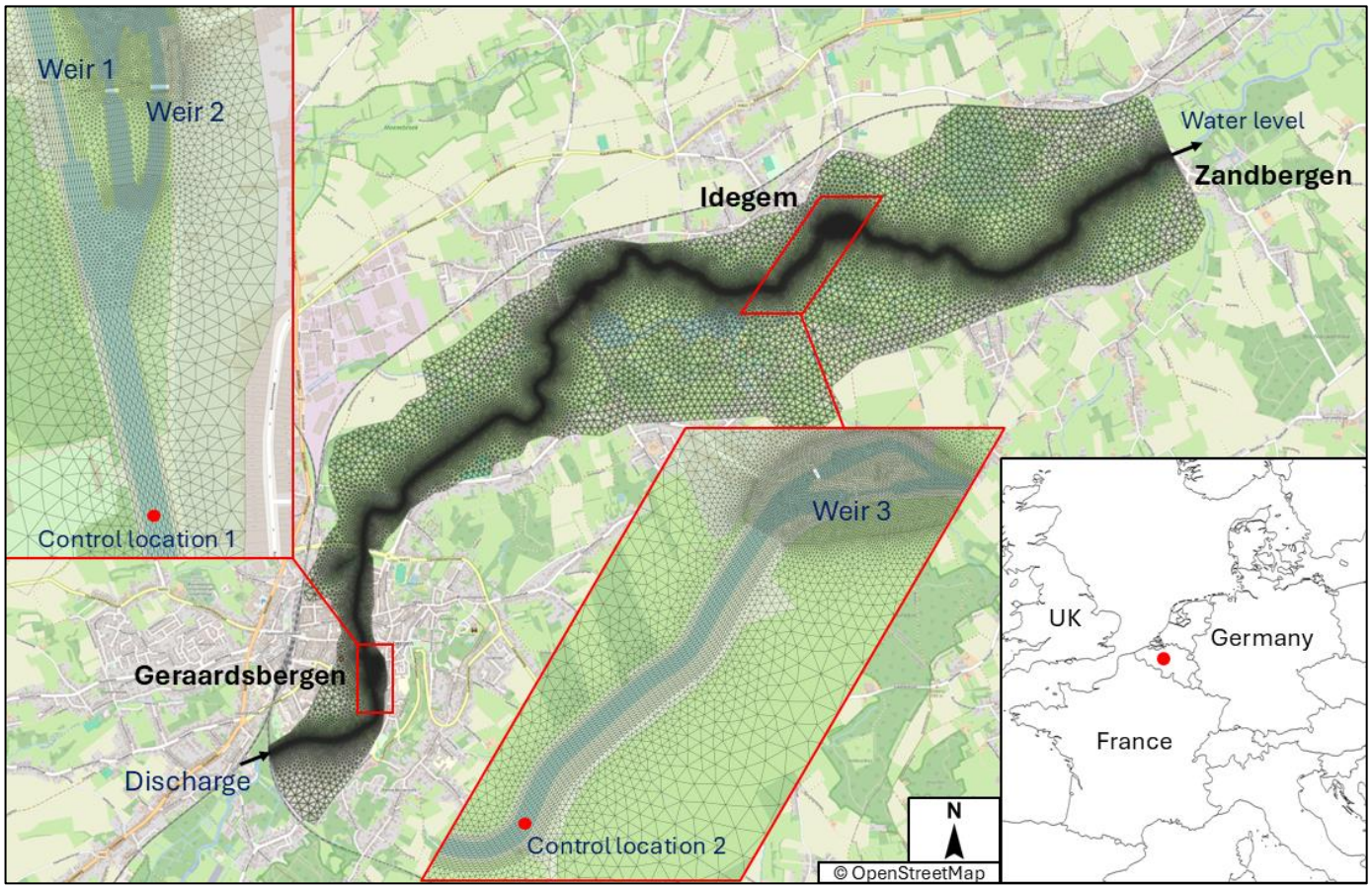

Figure 6. Model mesh of section of the Dender river between Geraardsbergen and Zandbergen. The mesh hold 3 weirs: 2 located upstream at Geraardsbergen and 1 downstream at Idegem. The Dender river is located west from Brussels in Belgium. (Coordinate system Lambert Belge 1972)

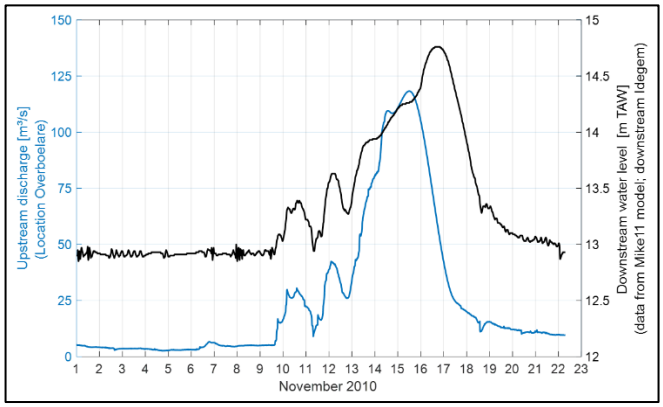

Figure 7. Boundary conditions for the Dender river model: upstream discharge in blue and downstream water level in black.

The time step of the original model was 6 seconds. With the use of obstacles this time step was reduced to 2 seconds for stability reasons. In Telemac-2D, the initial model completed a 20-day simulation in approximately one hour and ten minutes. By reducing the time step and requiring a lot more iterations, for the same 20 days run the simulation time increased to 25 hours and seven minutes. It was beyond the scope of this exercise to determine what caused the significant increase in runtime, but it is a concern when using the obstacle code.

C. Control of crest elevation by upstream water level

To control the crest level of the weir based on the water level of a point upstream, the obstacle code includes control points for this purpose. Unfortunately, at the time of this exercise, this section of the code still contained a bug. Without the bug, the user would have to add two control points to both the geometries and control files. An example of this will be given in the next section. The water level from these control points appears in the user_wei_controls.f subroutine. This subroutine allows the user to program how the water level will affect the weir's crest level (WEI%YS). Because of the bug, the water level at the control points could not be passed to the user subroutine at the time of this exercise. To work around this bug two nodes were selected manually (node numbers chosen from the mesh) and the call to the subroutine user_wei_controls.f was made from the telemac2d.F subroutine directly somewhere within the time loop, passing the water depth and bottom elevation (to calculate the free surface elevation) of the two chosen nodes to the user subroutine. The user_wei_controls.f subroutine as it was used in this exercise is given in Figure 8. Without the bug, this subroutine would be called automatically from the obstacles module through the get_obs_flowrate.f subroutine, which uses the control locations. Therefore, the user_wei_controls.f subroutine shown in Figure 8 can be used with a few minor modifications to work as intended once the bug is fixed.

Every 30 minutes, regardless of the time step, the user_wei_controls.f subroutine determines if the water level in the control points deviates more than 0.05 meters from the desired water level upstream the weirs. It also verifies the weirs' current crest level. If this crest level is not equal to the bottom level and the water level upstream the weir needs to be lowered, the crest level of the weir is lowered with the specific value of the beam height for that weir. For simplicity, the beam heights for the small weir upstream (weir 2 in Geraardsbergen) and the downstream weir (weir 3 at Idegem) in the Dender river model were all set to the same value. For the big weir upstream (weir 1 at Geraardsbergen), schematically shown in Figure 4, a vector was created using the precise beam heights. The listing file keeps track of the crest level of and the discharge over each weir in the model.

```
!                  ****************************
                   SUBROUTINE USER_WEI_CONTROLS
!                  ****************************
     &( WEI,J,AT,LT,ENTET,H,ZF,I1,I2)
!
!***********************************************************************
! BIEF V8P4
!***********************************************************************
!
!brief    User subroutine to control the operation of a weir structure
!+          WEI%OLAW      = TYPE OF HYDRAULIC LAW
!+          WEI%GEO%WIDTH = OPEN WIDTH
!+          WEI%YS        = CREST ELEVATION
!+          WEI%DLAW      = LAW OF DISCHARGE COEFFICIENT
!+          WEI%ALPHA     = DISCHARGE COEFFICIENT
!+          WEI%YP        = HEIGHT OF THE WEIR CREST
!+          WEI%THETA     = ANGLE OF V-SHAPED WEIR OPENING
!+          WEI%GAMMA     = ANGLE OF TILTATION
!+          WEI%ALAX      = TIME RELAXATION FACTOR
!+          WEI%NLAX      = NUMBER OF RELAXATION STEPS
!
!history  S.E. BOURBAN (EDF R&D), M.S. TURNBULL (HRW)
!+        20/06/2023!+        V8P4!+        Initial implementation.
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!|   WEI   |<->| STRUCTURAL DEFINITION OF A WEIR
!|   GRAV  |-->| GRAVITY
!|   AT    |-->| CURRENT TIME IN SECOND SINCE THE START OF TIME
!|   LT    |-->| CURRENT TIME ITERATION NUMBER FROM THE START
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE DECLARATIONS_SPECIAL
      USE OBS_GEOMETRY, ONLY : OBJ_VARIABLES,S_SMOOTH, T_SMOOTH
      USE OBS_WEIR
      USE OBS_CTRLLOC
      USE OBS_CONTROLS
      USE INTERFACE_PARALLEL, ONLY : P_MAX,P_MIN
      USE BIEF_DEF, ONLY : NCSIZE
!
      IMPLICIT NONE
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      TYPE(OBJ_WEIR), INTENT(INOUT) :: WEI
      INTEGER, INTENT(IN) :: LT,I1,I2,J
      LOGICAL, INTENT(IN) :: ENTET
      DOUBLE PRECISION, INTENT(IN) :: AT,H(*),ZF(*)
!
      INTEGER T_INT,LTI,N,K,KW
!
      DOUBLE PRECISION YS_MAX1,YS_MIN1,S1,S_CTR1,S_DS,TS
      DOUBLE PRECISION, DIMENSION(6) :: D_YS1
      DOUBLE PRECISION, DIMENSION(7) :: YS1
      DOUBLE PRECISION YS_MAX2,YS_MIN2,D_YS2,S2,S_CTR2
      DOUBLE PRECISION YS_MAX3,YS_MIN3,D_YS3,S3,S_CTR3
!
      INTRINSIC INT
```

```
      IF(I1.GT.0) THEN
        S1=H(I1)+ZF(I1)
      ELSE
        S1=0.D0
      ENDIF
      IF(I2.GT.0) THEN
        S3=H(I2)+ZF(I2)
      ELSE
        S3=0.D0
      ENDIF
      IF(NCSIZE.GT.1) THEN    ! In case of parallel
        S1=P_MAX(S1)+P_MIN(S1)
        S3=P_MAX(S3)+P_MIN(S3)
      ENDIF
      S2=S1  ! Weir 2 is controlled by the same location as weir 1
!
      YS_MAX1=17.0D0
      YS_MIN1=13.30D0
      YS_MAX2=17.06D0
      YS_MIN2=14.66D0
      YS_MAX3=15.59D0
      YS_MIN3=11.59D0
!     weir 1 can be lowered in different steps
      D_YS1=[0.25D0,0.25D0,0.82D0,0.97D0,0.63D0,0.78D0]
      D_YS2=0.3D0
      D_YS3=0.8D0
!     weir 1 is special as it has unequal steps to open or close
      YS1=[17.0D0,16.75D0,16.50D0,15.68D0,14.71D0,14.08D0,13.3D0]
!     The desired controlled water level upstream is
      S_CTR1=16.85D0
      S_CTR2=16.85D0
      S_CTR3=15.23D0
!     Allowed range on the desired water level upstream is
      S_DS=0.05D0
!     Comparing water level upstream only every 1800 seconds
      T_INT=1800
!     check how many time steps needed to reach interval time
      TS=AT/LT
      IF (T_INT/TS-(INT(T_INT/TS)).GT.0) THEN
        LTI=INT(T_INT/TS)+1  ! LTI is interval in time between checks for
      ELSE                   ! desired water level
        LTI=INT(T_INT/TS)
      ENDIF
!     check time interval
      IF ((LT-(INT(LT/LTI)*LTI)).LT.1) THEN
      IF( ENTET ) THEN
        WRITE(LU,*) "water level and crest elevation check "
        WRITE(LU,*) "water level 300 m upstream WEIR 1 = ",S1
        WRITE(LU,*) "water level 600 m upstream WEIR 3 = ",S3
      ENDIF
!     Check if upstream water level (S) is below or above the
!     desired range(S_CTR-S_DS, S_CTR+S_DS)
!     and check if weir crest elevation is fully open or closed
!     for weir 1 we need to know at which crest level we are
!     to know how much needs to be added or removed
```

```
      IF (J.EQ.1) THEN   ! Weir 1
        DO K=1,7
          IF ((WEI%YS-YS1(K)).LT.0.01D0) THEN
            KW=K
          ENDIF
        ENDDO
        IF (S1.GT.(S_CTR1+S_DS).AND.WEI%YS.GT.YS_MIN1) THEN !lower crest level
          WEI%YS=WEI%YS-D_YS1(KW)
          IF( ENTET ) THEN
            WRITE(LU,*) "USER CREST ELEVATION WEIR 1 SET TO ",WEI%YS
          ENDIF
        ENDIF
        IF (S1.LT.(S_CTR1-S_DS).AND.WEI%YS.LT.YS_MAX1) THEN  ! increase
          WEI%YS=WEI%YS+D_YS1(KW-1)
          IF( ENTET ) THEN
            WRITE(LU,*) "USER CREST ELEVATION WEIR 1 SET TO ",WEI%YS
          ENDIF
        ENDIF
      ENDIF
      IF (J.EQ.2) THEN    ! weir2
!     check if weir 2 crest elevation needs to be lowered
        IF (S2.GT.(S_CTR2+S_DS).AND.WEI%YS.GT.YS_MIN2) THEN
          WEI%YS=WEI%YS-D_YS2
          IF( ENTET ) THEN
            WRITE(LU,*) "USER CREST ELEVATION WEIR 2 SET TO ",WEI%YS
          ENDIF
        ENDIF
!       check if weir 2 crest elevation needs to be increased
        IF (S2.LT.(S_CTR2-S_DS).AND.WEI%YS.LT.YS_MAX2) THEN
          WEI%YS=WEI%YS+D_YS2
          IF( ENTET ) THEN
            WRITE(LU,*) "USER CREST ELEVATION WEIR 2 SET TO ",WEI%YS
          ENDIF
        ENDIF
      ENDIF
      IF (J.EQ.3) THEN    ! weir 3
!     check if weir 3 crest elevation needs to be lowered
        IF (S3.GT.(S_CTR3+S_DS).AND.WEI%YS.GT.YS_MIN3) THEN
          WEI%YS=WEI%YS-D_YS3
          IF( ENTET ) THEN
            WRITE(LU,*) "USER CREST ELEVATION WEIR 3 SET TO ",WEI%YS
          ENDIF
        ENDIF
!       check if weir 3 crest elevation needs to be increased
        IF (S3.LT.(S_CTR3-S_DS).AND.WEI%YS.LT.YS_MAX3) THEN
          WEI%YS=WEI%YS+D_YS3
          IF( ENTET ) THEN
            WRITE(LU,*) "USER CREST ELEVATION WEIR 3 SET TO ",WEI%YS
          ENDIF
        ENDIF
      ENDIF
    ENDIF
!-----------------------------------------------------------------------!
    RETURN
    END
```

Figure 8. Automatic control of the crest elevation using two control locations programmed into the user_wei_control.f subroutine.

The data on the crest levels of the three weirs as well as the water level of the two control points is extracted from the listing file. Figure 9 shows in the top panel the water levels in the control locations.

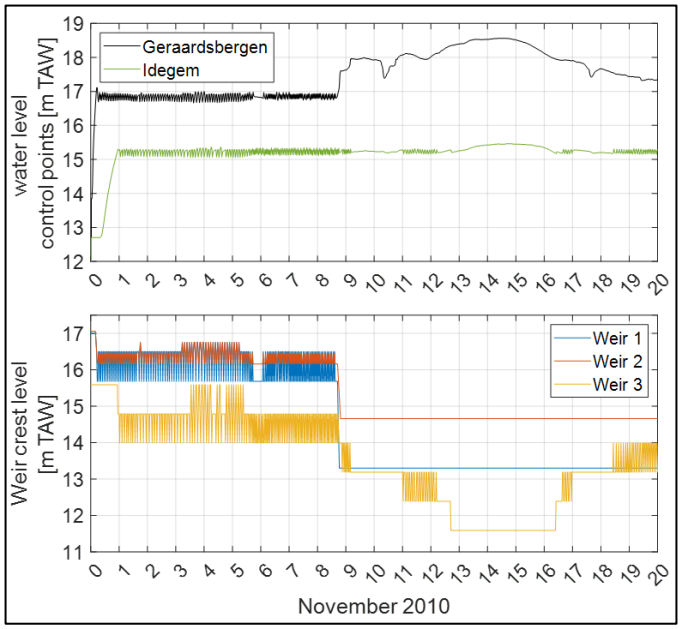

Figure 9. Modelled water level upstream of the weirs 1 and 2 (Geraardsbergen) and weir 3 (Idegem) in the top panel; and in the bottom panel the crest elevation of the three weirs.

After a spin up time of 1 day for the model, the water level at the Geraardsbergen control point remained close to the desired water level of 16.85 m TAW. The lower panel in Figure 9 shows, however, that in order to keep the water level relatively constant, the crest level must be adjusted several times per day. Given the heavy wooden beams and the manual labour to hoist them out of the construction, the automation applied in the model is not a feasible in reality for these old weirs. It demonstrates, however, what could be achieved with automated weir control. After nine days in the simulation, the large flood enters and lowers the crest level of the three weirs to the bottom level. Opening the two weirs upstream is insufficient to maintain the desired water level, resulting in the Dender valley flooding in 2010. However, in this simulation, the downstream weir appears to be maintaining the upstream water level8 at the desired elevation, probably because a large amount of water is diverted out of the river near the Geraardsbergen weirs.

The new approach to modelling weirs, the standardised input files, and all of the built-in features, such as the control location, make it much easier for end users to apply and give them much more control options than they had before.

IV. Weirs: Crest Level Controlled by Time Series, the Kakhovka Dam Break

A user-supplied time series can also be used to control a weir's crest level. This example uses a broad crested weir to model (in TELEMAC-2D) the break in the Nova Kakhovka dam in Ukraine. The crest elevation abruptly changes from eighteen meters above mean sea level to four meters above mean sea level. The Nova Kakhovka dam broke on June 6, 2023, at 03:00, creating a 450 m-wide opening [4].

Figure 10 depicts the model domain, which includes a portion of the Black Sea to illustrate the impact of the massive inflow of freshwater. The resolution of the mesh, which has 255725 nodes, ranges from 50 m near the dam and in the Djnepr river, to 500 m near the mesh's edges. The Manning bottom friction coefficient was set to 0.02 $s/m^{1/3}$ for the river and Black Sea area and to 0.05 $s/m^{1/3}$ for all other areas. The model has a constant water level (0 m above sea level) boundary (red line in Figure 10) at the Black Sea. The original model contained a small reservoir upstream of the dam. This reservoir was controlled by a water level boundary, controlled by a time series of measured water level data from the Kakhovka reservoir. The simulation began with the dam already broken and the opening represented in the bathymetry. The initial water level of the reservoir was set, using the INITIAL CONDITIONS = SPECIAL keyword in the steering file and the user_condin_h.f subroutine to set the water level in the small reservoir to 17.42 meter above sea level (= the last registered water level in the Kakhovka reservoir before the dam break [4]). The water level boundary on the small reservoir ensured that the amount of water escaping the reservoir corresponded to the volume of the actual reservoir. There was no bathymetry data for the Kakhovka reservoir, which was the reason for using a small reservoir with water level boundary. The topography data for the land comes from satellite data (SRTM) [5] and the bathymetry from the black sea comes from the EMODnet database [6]. Salinity was added as a tracer. For simplicity the salinity in the Black Sea was set constant to 14, a value based on data from [7].

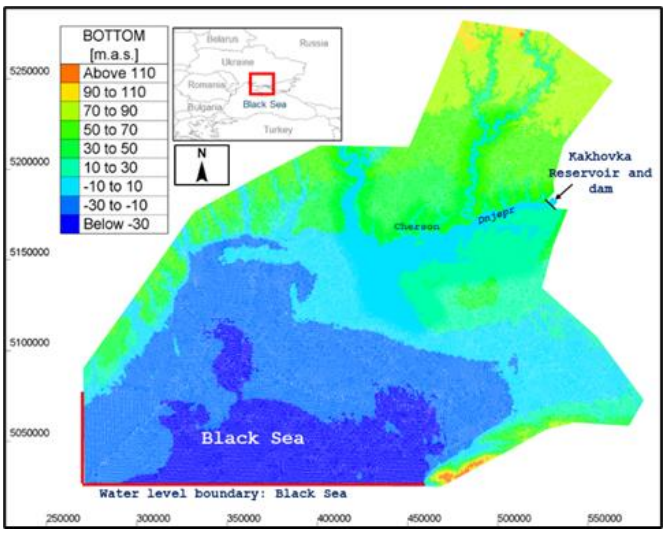

Figure 10. Model domain with bottom elevation and boundary conditions for the Kakhovka dam break model.

The original model was modified to use the obstacles approach: the small reservoir is replaced by a virtual pond. A weir is added between the virtual pond and the mesh. The geometry for both the virtual pond and the weir are added to the FILE OF GEOMETRIES FOR THE OBSTACLES (not shown here). All parameters for the pond and weir were set in the control file or FILE OF SETTINGS FOR THE OBSTACLES, see Figure 11. First, the total number of each type of obstacle must be specified. The NUMBER OF LINKAGES DEFINED AS WEIRS is set to one. The NUMBER OF STORAGE PONDS is set to one, and the NUMBER OF CONTROL LOCATIONS is set to one. The latter is only added to invoke a call to the user_wei_controls.f subroutine. This subroutine links the actual crest level of the weir (WEI%YS) in the model for each time step to a user-defined time series of crest elevations. Therefore, the control location needs no geometry in this case nor other parameters. Through the obstacle's name, the parameters of an obstacle set in the control file are connected to the parameters of the same obstacle in the geometry file. For the weir in this case an extra keyword is given to link it to the storage pond or virtual pond: NAME OF THE PAIRED STORAGE POND : POND 1. The keyword TYPE OF HYDRAULIC LAW is set to 2 to indicate a broad crested weir is used. The default value is 1, which is a sharp crested weir. The keyword OPEN WIDTH of the weir is important to set the length of the weir as this value will be taken to calculate the total discharge over the weir. The geometry of the weir is not used to calculate the length of the weir. The same for the pond: the geometry of the pond does not determine its volume nor surface area. Instead a user defined H – V relation is given in the control file. For the Kakhovka reservoir the total volume and responding water level were taken from [8]. The relationship is linear, so two points are sufficient. In the steering file an extra keyword is added: TEMPORAL CONTROL FILE FOR THE OBSTACLES, which refers to the file containing the time series data for crest level control. This file is shown in Figure 12. The simulation starts at 00:00hr June 6the 2023. After three hours the dam breaks, lowering the crest level instantly from eighteen meters to four meters above mean sea level.

```
#
#   TEST THE IMPLEMENTATION OF THE MODULE: OBSTACLES
#_____________________________________________________
#
  NUMBER OF LINKAGES DEFINED AS WEIRS :      1
  NUMBER OF CONTROL LOCATIONS         :      1
  NUMBER OF STORAGE PONDS             :      1
#
  NAME OF THE WEIR    :    WEIR 1
   TYPE OF HYDRAULIC LAW = 2
   CREST ELEVATION       = 18.5
   DISCHARGE COEFFICIENT = 0.4
   OPEN WIDTH            = 450.0
   NAME OF THE PAIRED STORAGE POND : POND 1
#
 NAME OF THE POND : POND 1
   INITIAL WATER LEVEL           = 17.42
   INITIAL TRACER CONCENTRATION = 0.5
   H : 0.0; 17.42
   V : 0.0; 37540100000
#
 NAME OF THE CONTROL LOCATION : CTRL 1
#_____________________________________________________
#
```

Figure 11. Control file for the weir that mimics the Kakhovka dam break (FILE OF SETTINGS FOR THE OBSTACLES)

```
#  Crest levels for the Kakhovka Dam.
#  YS = crest level weir
TIME      YS
S         M
0         18.5
10800     18.5
10800     4.0
2592000   4.0
```

Figure 12. Temporal control file for the crest elevation of the weir that mimics the Kakhovka dam break (TEMPORAL CONTROL FILE FOR THE OBSTACLES)

The user_wei_controls.f subroutine is called automatically because a control location has been added. Only two lines need to be active in this subroutine: a call to the get_obs_controls.f subroutine, which reads the TEMPORAL CONTROL FILE FOR THE OBSTACLES and returns the user-defined crest level (YS) for each time step. The second line designates the called crest level (YS) as the new crest level of the weir: WEI%YS = YS. The bottom panel in Figure 13 shows the change in crest level of the weir. The top panel in Figure 13 shows the water level in the virtual pond and compares it with the water level in the small reservoir of the original model and with the observed water level (data from [4]). The discharge over the weir/dam is also shown. The water level in the pond/reservoir is dependent on the discharge over the weir. The weir represented in the bathymetry of the original model will differ from the weir defined as an obstacle, as well as from the actual opening in the broken dam. Calibration and testing with the size of the weir could improve this.

Finally, Figure 14 shows the water depths four days after the dam break and gives an impression of the extent of the flood. The same kind of figure is given in [4], and the extent of the flood is similar. The time step of this model was 1 s. It was the same in the original model and in the obstacle version. Both simulations ran without stability issues, but the obstacle version needed two hours longer, i.e. 5 hours and 33 minutes for a simulation of 6 days on 48 processors. Efficiency and speed testing were not part of this exercise, and thus it remains just an observation.

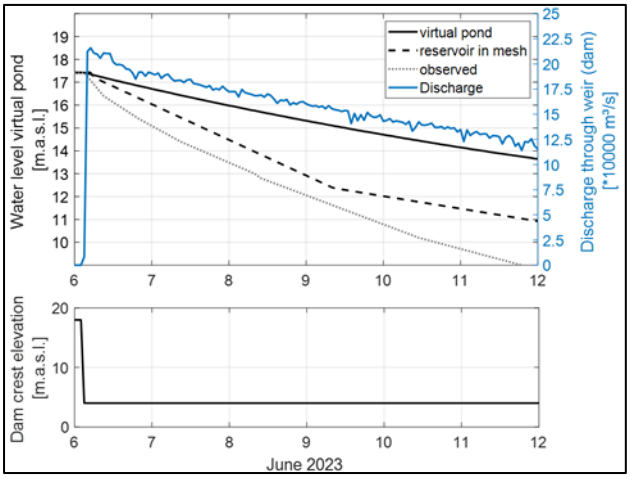

Figure 13. In the top panel the water level inside the virtual pond and the discharge through the dam break, i.e. the weir; in the bottom panel the weir (dam) crest elevation.

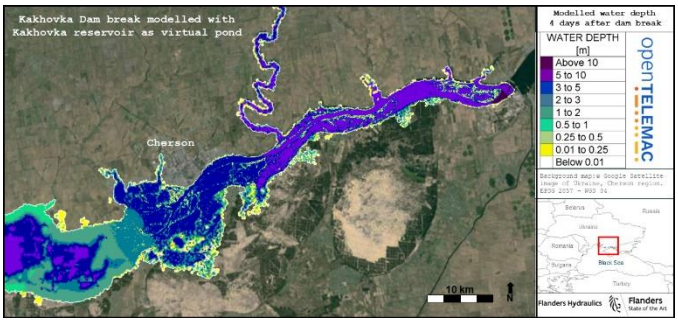

Figure 14. Modelled water depths four days after the dam break.

V. CULVERTS: BERGENMEERSEN TESTCASE

A. *Original TELEMAC-3D validation example*

The name Bergenmeersen refers to an existing validation example in TELEMAC-3D. The name originates from a flood control area (FCA) with controlled reduced tide (CRT) in the Scheldt estuary (Belgium). A FCA is an area adjacent to the Scheldt estuary but separated by a dike with a lower crest level. When storm surges enter the estuary, excess storm water can flow over these "overflow" dikes into the FCA, lowering the water level in the estuary. Some of these areas are designed to shelter tidal habitat, and culverts allow water to enter and leave these areas with each tide. Because their ground level is low (to provide ample storage space for storm water), the culverts only allow a limited amount of water to enter, preventing the area from becoming completely submerged, hence the name reduced tide. There are separate culverts for letting water in and out. [9] contains more detailed information on this area, as well as the design of the culverts and how they are modelled. [10] provides more information on how they work in storm conditions.

The model domain is shown in Figure 15. The overflow dike between the FCA and the Scheldt estuary is visualised with a black dashed line. There are six inlet culverts and 9 outlet culverts. This testcase validates the use of a Bodhaine + Carlier approach to model culverts (see [9]). The default way culverts are modelled in TELEMAC is based only on the Carlier formulations. The original code uses subroutines lecbus.f and buse.f and they work the same in 2D as in 3D. The testcase will be used to introduce also the Bodhaine + Carlier approach into the obstacles code. To test it also in 2D, a 2D version of the testcase was made. Because the culverts use the same subroutines in 2D as in 3D, only a few specific keywords needed to be changed to get the 2D version. And to get it more stable the time step was reduced from 3 to 1 s. The results for 2D were exactly the same as the results for 3D (not shown).

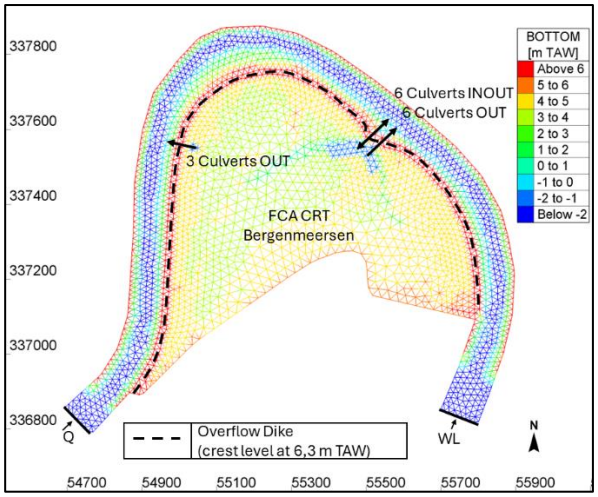

Figure 15. Model domain for the Bergenmeersen testcase for culverts in TELEMAC-3D

B. *Bergenmeersen testcase for obstacles*

To make this testcase work with obstacles, the first step was to add the Bodhaine + Carlier approach for culverts code to the obstacles' culvert module. The code is copied from the buse.f subroutine and inserted into the existing obstacles module, obs_culvert.f. Several additional keywords specific to culverts modelled using the Bodhaine + Carlier approach have been added to the module and can be entered by the user in the control file. Some of the Bergenmeersen culverts only allow flow in one direction. This one-way valve treatment was also added to the obs_culvert.f module.

In Blue Kenue, lines can be drawn to represent each culvert individually. These lines are defined only by their starting (entrance) and ending (exit) points. For culverts that only go one way, the first point in the geometry file (see Figure 18 INLET 1) must be the culvert's entrance point. A geometry file was created for the Bergenmeersen testcase, which included 15 culverts. The physical parameters of each culvert must be specified in the control file (see Figure 19, INLET 1). Because the culvert code in obstacles is a copy of the buse.f subroutine, the file that was read by the lecbus.f subroutine in the original version can be converted into the control file for obstacles using the proper keywords. The keyword TYPE OF HYDRAULIC LAW needs to be set to 2 for the Bodhaine + Carlier approach. The default value here is 1 (only Carlier).

The steering file for the Bergenmeersen testcase's 2D obstacle version is identical to the original 2D version, with the exception of the removal of the culvert keywords, the addition of the geometry and control file for obstacles, and the keyword MAXIMUM NUMBER OF ITERATIONS FOR ADVECTION SCHEMES = 200. The latter was added out of necessity. The

2D version produces good results, but they are not exactly the same as the original version. When the results are plotted, some of them require further investigation because they reveal that there are still some bugs in the code. An example is given in Figure 17 in the bottom panel: the red arrow points at a negative discharge for an outlet culvert. These culverts have a one-way valve, so flow is only restricted to outflow. The negative discharge suggests that flow is entering the FCA through the outlet culverts. At the time of writing, the 3D version still contained some errors when masking elements. This must be resolved first before the culvert results can be simulated in 3D. However, since the 2D version works, two additional possibilities for the new obstacle approach for the Bergenmeersen testcase are shown below.

C. Bergenmeersen testcase: culverts+virtual pond+weirs

The FCA area can be replaced with a virtual pond, and the overflow dike function can be replaced by connecting 16 weirs between the river and the virtual pond (see Figure 16). The culverts remain the same, but they must be paired with the virtual pond in the control file using the keyword: NAME OF THE PAIRED STORAGE POND. By replacing the FCA with a virtual pond, the number of nodes of the mesh is reduced from 2618 to 1374. A significant difference for such a small model. Figure 18 shows part of the geometry file and Figure 19 shows part of the control file. In each of the figures an example of culvert, weir and virtual pond is given. The H-V relationship for the virtual pond was derived using Blue Kenue's bathymetry volume calculation function. It was not yet possible to define a single large weir made up of multiple lines/points, which is why there are 16 weirs in this case.

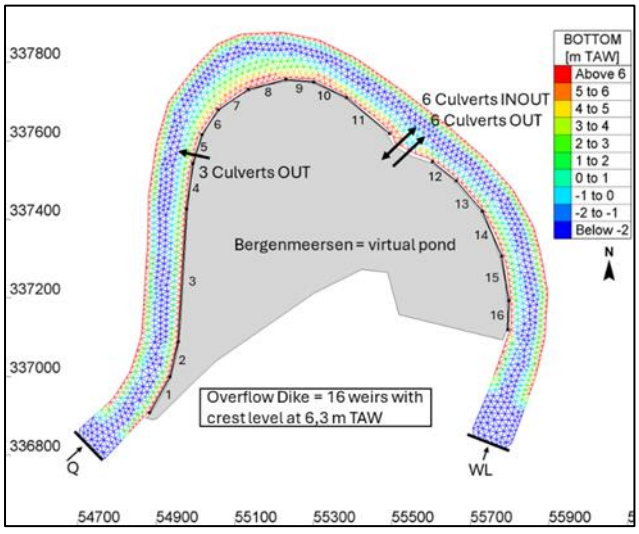

Figure 16. Model domain for Bergenmeersen testcase for obstacles where FCA is replaced by virtual pond and the overflow dike is replace by 16 weirs.

To test the water flooding the overflow dike (here represented by the 16 weirs) the boundary conditions were changed to the storm period of 6th of December 2013, like used in [10]. Figure 20 shows the discharge over the weirs as the water level in the estuary raised above the crest level (6.3 m TAW) and proves the new setup of the testcase is working properly.

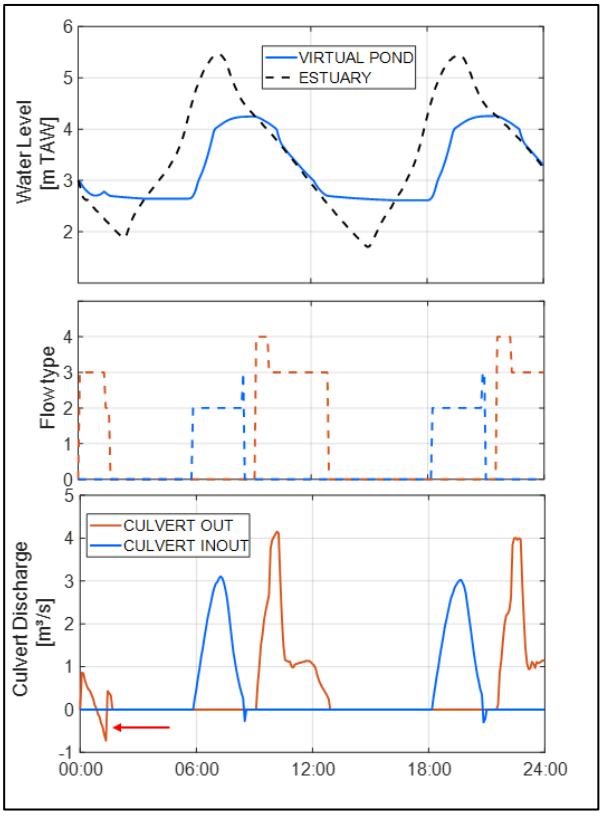

Figure 17. TELEMAC-2D results for Bergenmeersen testcase for obstacles where the FCA is replaced with a virtual pond.

```
##############################################
:FileType i2s  ASCII  EnSim 1.0
:Application  BlueKenue
:Version      3.12.21
:WrittenBy    S.SMOLDERS
:CreationTime 2024/08/24 00:00:00.000
#---------------------------------------------
:Name  obs_bergenmeersen
#
:AttributeCount 1
:AttributeUnits 1 m
:EndHeader
2 30 "WEIR 1"
54883.258021169451000 336907.949447681660000
54935.124043914024000 336999.065433584330000
2 30 "WEIR 2"
54935.124043914024000 336999.065433584330000
54956.150809891558000 337088.779635088460000
2 30 "WEIR 3"
54956.150809891558000 337088.779635088460000
54977.177575869107000 337425.908782928130000
...
42 30 "POND 1"
54873.8827 336894.444
54886.4825 336908.140
54902.9170 336933.339
54936.3337 336993.051
...
2 5 "INLET 1"
55637.6000 337567.290  0
55497.3600 337553.140  0
2 5 "INLET 2"
55627.3000 337578.230  0
```

Figure 18. small part of the geometry file defining the weirs, culverts and virtual pond

```
#
#   TEST THE IMPLEMENTATION OF THE MODULE: OBSTACLES
#
#_____________________________________________________________
#
  NUMBER OF OBSTACLES DEFINED AS WEIRS :     16
#
  NAME OF THE WEIR   :    WEIR 1
    CREST ELEVATION       = 6.3
    DISCHARGE COEFFICIENT = 0.4
    OPEN WIDTH            = 118
    NAME OF THE PAIRED STORAGE POND : POND 1
#
  NAME OF THE WEIR   :    WEIR 2
    CREST ELEVATION       = 6.3
    DISCHARGE COEFFICIENT = 0.4
    OPEN WIDTH            = 118
    NAME OF THE PAIRED STORAGE POND : POND 1
#
  NUMBER OF LINKAGES DEFINED AS CULVERTS :     15
#
  NAME OF THE CULVERT   :    INLET 1
    TYPE OF HYDRAULIC LAW             = 2
    SECTIONAL SHAPE                   = 1
    VALVE OPERATION                   = 0
    ELEVATION OF CULVERT BOTTOM       = 4.35 ; 4.2
    OPENING HEIGHTS                   = 1.45 ; 1.8
    OPENING WIDTHS AT THE BASE        = 2.7 ; 2.7
    LENGTH OF THE CULVERT             = 9.5
    HEAD LOSS COEFFICIENT AS ENTRY    = 0.9 ; 0.5
    HEAD LOSS COEFFICIENT AS EXIT     = 1.0 ; 1.0
    HEAD LOSS DUE TO VALVE            = 0
    FACTOR FOR FLOW TYPE 5 OR 6       = 10
    CORRECTION FOR CVALV FLOW TYPE 5 = 0
    CORRECTION FOR CLOES FLOW TYPE 5 = 6
    HEAD LOSS COEFF TRASH SCREEN      = 1
    MANNING COEFFICIENT FOR WALLS     = 0.015
    TIME RELAXATION FACTOR            = 0.
    NUMBER OF RELAXATION STEPS        = 0
    NAME OF THE PAIRED STORAGE POND : POND 1
#
  NUMBER OF STORAGE PONDS :     1
#
  NAME OF THE POND : POND 1
    INITIAL WATER LEVEL          = 3.0
#   INITIAL TRACER CONCENTRATION = 0.0
    H : 0.0; 1.0; 2.0; 3.0; 4.0; 5.0; 6.0; 6.2; 7.0
    V : 75.0; 139.0; 235.0; 410.0; 851.0; 3666.0; 7330.0; 8090.0; 8090.0
#
```

Figure 19. Part of the control file for the Bergenmeersen testcase with culverts (FCA replaced by a virtual pond and overflow dike replaced by 16 weirs).

The original T2D Bergenmeersen testcase ran on 4 processors, simulating 3 days in 467 s. Replacing only the culverts with obstacle culverts the simulation took 443 s. When replacing the FCA with a virtual pond (and thus reducing the number of nodes in the model by half) and culverts the simulation took 247 s, almost halve the simulation time. However, when replacing also the overflow dike with the 16 weirs (including the virtual pond and culverts) the simulation time increased to 2860 s. Again, this is only an observation from a few tests, but it seems the weirs are really slowing down the simulation. It seems like an obstacle-weir related problem and not a general obstacle issue.

VI. CONCLUSIONS

The examples shown in this paper demonstrate the versatility and potential of the new simplified approach to model obstacles in TELEMAC. This approach offers endless possibilities for future developments. The user input files are similar for all types of obstacles. However, the control file can become very long and can take quite some time filling in obstacle physical properties by hand. For only 15 culverts, the control file counted 279 lines. In the SCALDIS model [10], 252 culverts are used to handle flow in and out all FCA's, and that will result in a very long control file and a lot of scrolling.

The last culvert testcases show that for culvert and pond obstacles the simulation time does not seem to be affected at all. The weirs in obstacles however seem to slow down the simulation a lot and this is a concern.

The obstacle approach is a big improvement for user friendliness and future possibilities in the openTELEMAC system and if all goes well, will be available in the next official release V9P0.

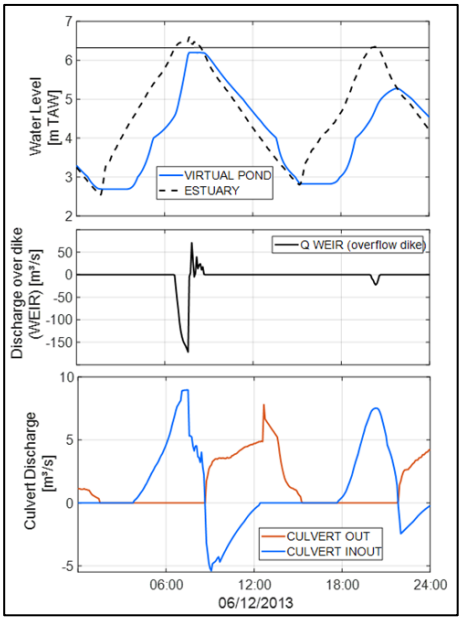

Figure 20. Results for the Bergenmeersen testcase for obstacles with the overflow dike replaced by 16 weirs and the FCA replaced by a virtual pond.

ACKNOWLEDGEMENT

S.S. thanks his wife, Gudrun, for taking care of the children and freeing up time, many evenings and on weekends to complete all the work necessary for this paper.

REFERENCES

[1] Sébastien E. Bourban and Michael S. Turnbull (2023): A simplified approach to modelling all types of obstacles in TELEMAC-2D and 3D. In: Kopmann, R. and Folke, F. (2023) Proceedings of the XXIXth TELEMAC Users Conference. 12-13 October 2023. Karlsruhe, Germany.

[2] Picture copyright available under public licence: Agentschap onroerend erfgoed, beeldbank. https://id.erfgoed.net/afbeeldingen/213733

[3] Digitaal Hoogtemodel Vlaanderen II, DTM, raster, 1m, Agentschap digitaal Vlaanderen, versie 2014.01.

[4] UKCEH & HRW, 2023. A rapid assessment of the immediate environmental impacts of the destruction of the Nova Kakhovka Dam, Ukraine. Report prepared by UK Centre for Ecology & Hydrology and HR Wallingford for the UK Foreign, Commonwealth & Development Office (FCDO) Expert Advisory Call Down Service 2 Lot 4 Rapid Assessment of the Environmental Impacts of the destruction of the Nova Kakhovka Dam, Ukraine. 29 June 2023. 10.5281/zenodo.10462809.

[5] Jarvis A., H.I. Reuter, A. Nelson, E. Guevara, 2008, Hole-filled seamless SRTM data V4, International Centre for Tropical Agriculture (CIAT), available from http://srtm.csi.cgiar.org.

[6] The bathymetric data for the black sea have been derived from the EMODnet Bathymetry portal: http://www.emodnet-bathymetry.eu

[7] Illustration by Philippe Rekacewicz, UNEP/GRID-Arendal: https://www.grida.no/resources/6539 accessed on 10/07/2024

[8] Wikipedia search on volume and water level of the Kakhovka reservoir: https://en.wikipedia.org/wiki/Kakhovka_Reservoir#:~:text=The%20reservoir%20covers%20a%20total,km%20(14%20mi)%20wide. (Accessed 23/08/2022)

[9] Smolders, S., Leroy, A., Teles, M. J., Maximova, T., & Vanlede, J. (2016) Culverts modelling in TELEMAC-2D and TELEMAC-3D. Proceedings of the XXIIIrd TELEMAC-MASCARET User Conference (pp. 21-33).

[10] Smolders, S., João Teles, M., Leroy, A., Maximova, T., Meire, P., & Temmerman, S. (2020). Modeling storm surge attenuation by an integrated nature-based and engineered flood defense system in the Scheldt Estuary (Belgium). Journal of Marine Science and Engineering, 8(1), 27.