Flexible Architecture for Analysis of Water Control Structures

Chendi Cao, Mitchell L. Neilsen
Department of Computer Science, Kansas State University
Manhattan, KS, 66506, USA
{caocd, neilsen}@ksu.edu

Abstract

Scientific computing involves the development, analysis, and execution of computational algorithms to solve mathematical problems from science and engineering. Effective implementation of these algorithms on high-performance computers requires knowledge and techniques from mathematics, statistics, and several areas in computer science including parallel computing, compilers, and theory. Although hydrologic field engineers have a wide array of research tools to use, they face the problem that all of these tools are loosely coupled.

In this paper, we propose a new framework to solve these problems. The framework is scalable and extensible, and it can be used to execute jobs either locally and on a cluster as needed. A canonical hydrograph format is defined using eXtensible Markup Language (XML) format to exchanging flow information between disparate software packages, and translators are developed to convert between formats.

Keywords: High performance computing, hydrology, parallel processing, scientific computing, XML.

1 Introduction

Most flood routing of water through water control structures (dams) before the middle 1960s was computed manually. Then, routing software on computers began to replace manual methods. In the early 1990s, approximately one hundred sites were selected for more in-depth evaluation and data collection, and data analysis. Additional tests were conducted in the USDA-ARS outdoor Hydraulic Engineering Research Unit (HERU) Laboratory near Stillwater, Oklahoma. Kansas State University cooperated with USDA to develop software for water control structure design and analysis. Advances in parallel computing capability have enabled more advanced tools to be developed to benefit all types of hydraulic science research.

The existing model is able to simulate the result of the erosion model, but it is computed in limited computing resource locally. For the output, transformation and sharing the result with other hydraulics software is extremely painful. We enable this new flexible architecture to extend existing dam safety analysis model to handle large simulation jobs and create a compatibility scheme to transfer data seamlessly. In order to use scientific computing to develop and solve mathematical problems from science and engineering in this domain, we have constructed a flexible architecture with three main parts: First, the ability to analyze dams by adding uncertainty analysis and parameter study analysis to existing dam safety analysis tools [9]. This allows engineers to determine which input parameters are most important without the daunting task of executing the model many times while manually modifying the input parameters.

In some cases, the analysis cannot be conducted in a timely fashion using only local computing resources. For this, we propose a new architecture that allows preliminary runs to be executed locally, while more computationally intensive runs can be automatically shipped off to run in the cloud or on a supercomputing cluster. Finally, different tools use different formats to represent hydrographs. This paper defines a new canonical format for hydrographs. The new format can be used to store and exchange data between disparate models and enable the pipelining of processing between different tools. The new framework consists of two parts:

- An Extensible Computing Framework is used to generate hydrographs and flow properties based on physical input from WinDAM input files created by the user. Computational fluid dynamics simulations can be executed locally on a users laptop, in the cloud, or on a supercomputing cluster such as Beocat.
- Canonical XML Hydrograph Translators are developed to convert data to/from the newly defined XML data format for hydrographs.

This paper first summarizes related work in Section 2 and then discusses the implementation and result of our new framework in Section 3. Section 4 presents conclusions and describes directions for future work.

2 Related Work

Windows^{TM} Dam Analysis Modules (WinDAM) were developed by USDA and Kansas State University[15]. WinDAM is a collection of modular software components that can be used to design and analyze the performance of earthen dams. BREACH, developed by the National Weather Service, can be used to simulate hydraulic flows over and through a dam, and estimate the breach potential for a given storm or inflow hydrograph[5]. Dakota is developed by Sandia National Laboratories for the design and analysis of computer experiments, including parameter studies, uncertainty analysis, sensitivity analysis, etc[1].

WinDAM/BREACH + Dakota is a framework by Neilsen and Cao to extend the capabilities of both. It is designed to integrate the simulation models in WinDAM and BREACH with the uncertainty quantification, sensitivity analysis, and parameter studies capabilities in Dakota.

The framework user interface shown in Figure 1, allows users to specify a range of input parameters. The framework also allows Dakota to drive the computational model and change the model inputs and perform the different analysis.

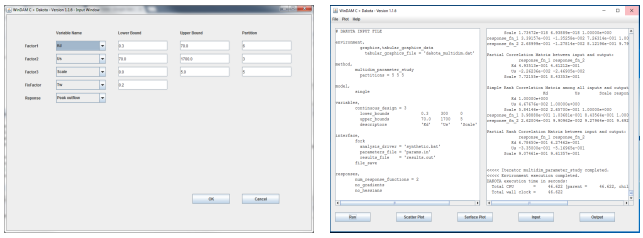

Figure 1: WinDAM/BREACH+Dakota GUI

For the given example, the resulting output shown in Figure 2 confirms a strong positive correlation between erodibility Kd and peak discharge and a small negative correlation between Undrained Sheer Strength Us and maximum discharge[9].

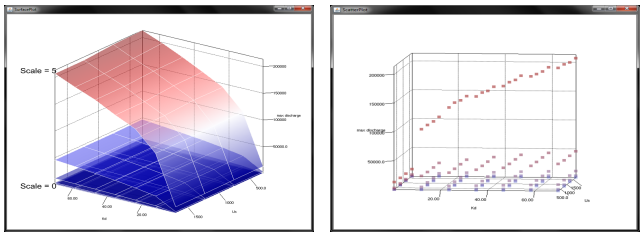

Figure 2: Surface and Scatter Plot

OpenFOAM is a C++ toolbox for the development of customized numerical solvers, and pre-/post-processing utilities for the solution of computational fluid dynamics (CFD)[10]. Beocat is Kansas State University High-Performance Computing (HPC) cluster[3]. Preliminary analysis can be conducted with a limited number of grid elements, and a more detailed analysis can be conducted on a high-performance computing cluster. We automate the process and evaluate the performance of OpenFOAM on a local laptop or on a cluster, such as Beocat. Computer simulation reproduces the behavior of a system using the mathematical model so that computer simulation becomes a useful tool to examine and compare with the experimental results. We build this extensible computing framework to compare and improve existing WinDAM erosion models use of OpenFOAM[7].

The SITES Water Resource Site Analysis computer program analyzes the hydrology and hydraulics for designs typical of NRCS dams and ponds[12]. The Computer Program for Project Formulation Hydrology (WinTR-20) is a single event watershed scale runoff and routing model[18]. eXtensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable[4]. XML is widely used for data storage, data transfer and backup tools for industry. It can also be used as a lingua franca between simulations, remote data sources, and components[2]. Unfortunately, there is no standard XML format for hydrograph definition, which is inconvenient for data storage and transfer in disparate hydraulics software. We proposed a new definition for hydrograph using XML schema XSD file and also developed a translator for existing hydraulics software.

3 Implementation

The new extensible computing framework allows computational fluid dynamics models for OpenFOAM to model the hydraulic flow over dams and through auxiliary spillways on the local machine or on the cluster. OpenFOAM has extensive features and packages for solving complex fluid flows. The computing power for a single machine is limited, high-performance computing (HPC) plays a significant role in the scientific computing for its capability of parallel computing for the large simulation job.

First, the important step is constructing the geometry of the dam and the water flow environment. STL is stereolithography file format and we use it to specify the dam cross-section. SALOME is an open-source software that provides a generic platform for Pre- and Post-Processing for numerical simulation[11]. To define the geometry of the dam, we put all specs of a dam in STL file and edit it in SALOME. SALOME provide

the variety of handy features, it can create geometry, generate the mesh and view geometry result using ParaView inside SALOME generic platform.

For a simple trapezoidal dam cross section, each dam STL file contains 18 triangular facets. Each triangular facets is defined by 3 vertexes and 1 direction. For this experimental design, we used real dam model and applied all the physical parameters in 2D geometry. Therefore, all the length and mesh element in the Z direction has been set to one meter long. Though we constructed the model in 2D geometry, default layout dimension is 3D in OpenFOAM. Water velocity in X direction has been recorded, and Y-axis is the gravity direction.

For the construction of our water flow environment, SALOME has been widely used to generate the geometry and separate the water flow section and the air section in the geometry. The inlet height of water flow for our model has been changed for matching a real dam model. The Background mesh has been generated by the ideasUnvToFoam solver. ideasUnvToFoam is a format mesh conversion function which transfers output file I-Deas unv format from SALOME to blockMesh format for OpenFOAM. Then, snappyHexMesh is the solver to choose which is a tool used to refine the mesh and generate more mesh elements on the border. After running extrudeMesh by extrudeMeshDict, the 2D patch will remove the dam cross section and generate a dam shape in water flow simulation model. In the future, we plan to apply 3D geometry in our simulation model.

After the geometry model is fixed, the water flow model must be specified. The reservoir capacity is specified as an initial condition in setFieldsDict. For the water flow, there are two possible types of flow in the pipe: laminar flow or turbulent flow. For this analysis, we specify turbulent flow, use the kEpsilon and RASmodel which stands for standard k-epsilon turbulence model for incompressible and compressible flows including rapid distortion theory (RDT) based compression term [10].

For the water flow data, a real field example is applied to our OpenFOAM simulation model. To input all the data in OpenFOAM, an ASCII text file dataTable.txt is formed to store the information.

As shown in Listing 3, the left table is the WinDAM output for the hydrograph. The right table shows the ASCII text file for dataTable.txt, the first column is representing the time and the time interval for this 02-SDHstability.WDC example is 0.063 hour. The second column has three values in one bracket, and it shows the different velocity in X, Y and Z direction for simulation purpose.

```
time(hr)  discharge(cfs)  time(s)  velocity(m/s)
0         1.35            ((0      (0.00095 0 0))
0.063     4.54            (0.063   (0.00321 0 0))
0.126     11.32           (0.126   (0.00801 0 0))
0.189     23.72           (0.189   (0.01679 0 0))
.         .               .        .          . .
8.757     0.5             (8.757   (0.00035 0 0)));
```

Figure 3: Input hydrograph data.

The conversion equation we applied here:

$$Velocity = \frac{Discharge}{(W \times D)} \tag{1}$$

Where W is Width and D is Depth.

In our analysis, $W = 1m$, $D = 4m$. For the units of measurement, we converted them from US system to imperial system. In order to apply to OpenFOAM model properly, a scale factor has been added. The velocity for our model is only considering in the X direction because our model is constructed in 2D geometry and the water inlet is designed in the X direction. That is also the reason why we set to value 0 for Y and Z direction in Figure 3.

To achieve the hydrograph data in a different aspect, the sample data from two different regions in our simulation model has been selected. One hydrograph data came from the top of the dam, and another hydrograph data came from the bottom of the dam shown in Figure 4. In Figure 4, there are two different lines in the graph, the red line represents the measure of area (m^2), and the brown line represents velocity (m/s).

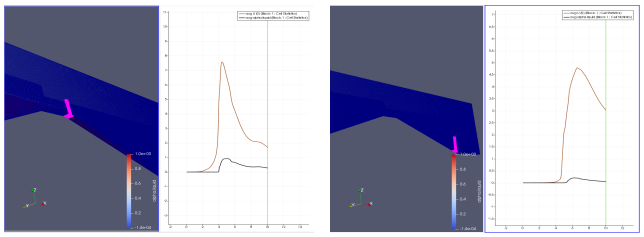

Figure 4: Dam Region Simulation Result, Top(Left), Bottom(Right).

In Figure 5, the scatter plot demonstrated the result for WinDAM and OpenFOAM simulation. The OpenFOAM simulation is based on the 10-second frame and 0.05-time interval. The blue line represents the inflow, orange line shows the total outflow for the WinDAM and the gray line shows total discharge for the top of the dam which is generated by OpenFOAM simulation. For WinDAM assumption, the total outflow is calculated near the top of the dam. Therefore, the result for

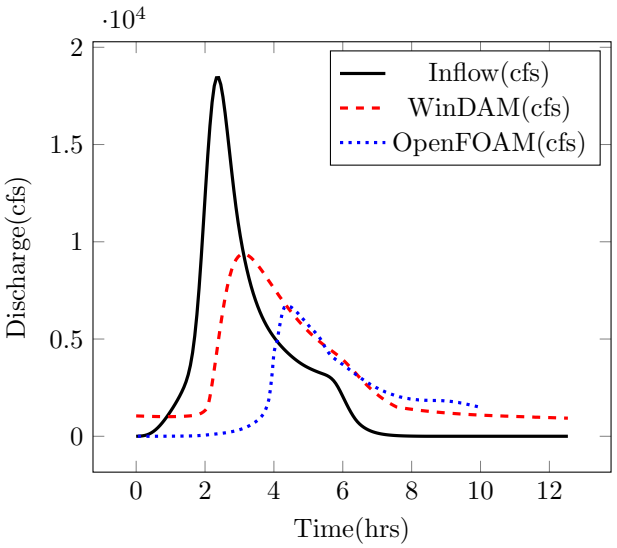

Figure 5: Hydrograph Comparison Plot.

top of dam generated in OpenFOAM is selected to compare with WinDAM. The hydrograph for these two different results should have the same amount of area in the graph because the amount of water goes through the dam are expected to be similar. The hydrograph for WinDAM has the higher peak of the water volume than the OpenFOAM. For the OpenFOAM hydrograph, it shifts to the right compared to the WinDAM hydrograph. The turbulence occurs at the end of the hydrograph shows that there is friction along the edge of the dam. Therefore, the total water volume is close for these two hydrographs. The reason why OpenFOAM hydrograph has the two-second delay is that for the OpenFOAM simulation, the position of inlet water is built far away from the dam. It takes more time to reach the dam compared to WinDAM. WinDAM assume that the water inlet is fairly close to the dam. For the overall comparison, both outflow hydrographs have similar pattern and shape.

After done the preliminary experiment, we apply our model on Beocat. Beocat is the High-Performance Computing (HPC) cluster at Kansas State University, it is running under CentOS and coordinated by job scheduler system called Slurm.

The sample code:

```
#!/bin/bash
#SBATCH --job-name=opfile1_4C32G_0.0001_f2
#SBATCH --mem-per-cpu=32G
# Memory per core, use --mem= for memory per node
#SBATCH --time=02:00:00
# Use the form DD-HH:MM:SS
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=4
#SBATCH --mail-user=caocd@ksu.edu
#SBATCH --mail-type=ALL
# same as =BEGIN,FAIL,END
singularity exec /opt/beocat/containers/
openfoam-v1712.img /bin/bash <<EOF
. /opt/OpenFOAM/setImage_v1712.sh
.....
OpenFOAM code
.....
EOF
```

Listing 1: Slurm job bash code.

As shown above, in Listing 1, start with /bin/bash and OpenFOAM code is adding inside the $SBATCH$ code segment. One single job can request 32 GB of memory per CPU, request 4 core at a time and total running time is 2 hours. Since the OpenFOAM module is installed inside a container on Beocat, the number of nodes is limited to only one. For this running time simulation job, we select two flow function,

$$\begin{aligned} f_1 &= 0.5 + 0.5sin(0.2\pi t). \\ f_2 &= 4. \end{aligned} \tag{2}$$

f is the velocity of water flow in X direction. t is the time in seconds.

Run the simulation based on different cores and different deltaT. deltaT is the time stamp of the simulation. In OpenFOAM contolDict file, we turn off the time stamp auto adjustment. The summary table as shown in Table 1.The simulation is performed under the circumstance that end time is 4 second, write interval is 0.05 second and the unit for runtime is in hours. For example, first case, use f_1 flow function, $delaT$ = 0.0001, 1 Core represents that one 32GB memory CPU requested, therefore the simulation time case 1 is 4 hours 54 minutes and 59 seconds. By reviewing the result, roughly 1 core performs 2 times faster than 1 core, 4 core performs 2 times faster than 2 core, 8 core performs 2 times faster than 4 core. The running time is getting smaller when you increase the timestamp deltaT.

N	F	deltaT	1 Core	2 Core	4 Core	8 Core
1	f_1	0.0001	4:54:59	2:33:16	1:29:36	0:50:11
2	f_1	0.0002	2:38:29	1:17:46	0:52:56	0:28:19
3	f_1	0.0005	1:18:25	0:26:47	0:17:15	0:12:44
4	f_1	0.0010	0:46:08	0:21:36	0:09:26	0:07:25
5	f_2	0.0001	4:36:29	2:32:46	1:28:09	0:48:56
6	f_2	0.0002	2:43:03	1:12:17	0:52:29	0:27:23
7	f_2	0.0005	1:07:25	0:30:42	0:16:57	0:12:29
8	f_2	0.0010	0:34:40	0:22:44	0:09:16	0:07:09

Table 1: Run Time Table for OpenFOAM simulation.

For the different deltaT, the Figure 6 shows the speedup for the various number of cores. 8 core performs approximately 5 times faster than 1 core, 4 core performs almost 3 times faster than 1 core and 2

core can achieve 2 times faster than 1 core. It shows that the bigger deltaT we have, the less speedup we can achieve.

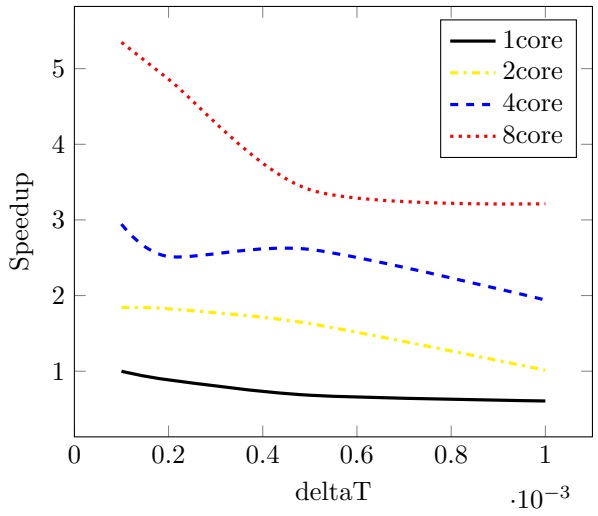

Figure 6: Performance comparison plot for different number of cores.

4 Canonical Hydrograph Translators

eXtensible Markup Language (XML) emphasize simplicity, generality, and usability. For the implementation of this canonical XML hydrograph translator, the new XML definition for hydrograph has to be defined. A new XML schema XSD file is created for storing all the information for hydrograph. The development is staged and now we are mainly focusing on the WinTR-20 and SITES and make our XML compile compatible with this two software. Here is the XML schema XSD file defined for hydrograph.

```
<?xml version="1.0"?>
<xs:schema name = "Hydrograph_XML_Schema"
elementFormDefault="qualified">
<xs:element name="run">
<xs:complexType>
<xs:sequence>
<xs:element name="Software" type="xs:string"/>
<xs:element name="ExampleName" type="xs:string"/>
<xs:element name="ExecutionTime" type="xs:string"/>
<xs:element name="Hydrograph" type="xs:string"/>
<xs:complexType>
<xs:sequence>
<xs:element name="StartTime" type="xs:time" units= "hours"/>
<xs:element name="EndTime" type="xs:time" units= "hours"/>
<xs:element name="TimeInterval"
type="xs:time"units= "hours" />
<xs:element name="DrainageArea"
type="xs:string" units= "squareMiles" />
.....
.....
</xs:schema>
```

Listing 2: Hydrograph XML Schema.

In listing 2, the XML schema XSD file contains data like software name, example name, execution time, hydrograph name, start time, end time and time interval, etc.

SITES mark each line by number to distinguish the format and function for that line. From the top, the first line 1SJ1 SITEHYDG is the title for the project and followed by the execution date, the second line contains project name and several parameters. Starting from line 5, discharge data are stored in the column. The key elements have been extracted from this output file like the first column in line 5: time interval when the flood happened, the second column in line 5: discharge value in CFS, and etc.WinTR-20 output file uses totally different format to store the data compare to SITES. WinTR-20 stores all the information in rows, that means, all information is queued in rows format.

Definition for hydrograph for storing both SITES and WinTR-20 hydrograph information has been developed, and XML hydrograph translator is constructed in Python to convert output files from SITES and WinTR-20 to the XML files. The SITES generate output hydrograph file in *.DHY format and WinTR-20 generates output hydrograph file in *.hyd format.

```
caocd@E2227EW10:~$ python compiler.py
Usage: compiler.py [options] input output

Options:
  -h, --help        show this help message and exit
  -s, --SITES       convert SITES *.DHY file
  -w, --WinTR-20    convert WinTR-20 *.hyd file
```

Listing 3: Canonical XML Hydrograph Translators

The XML hydrograph translators are been developed in Python. In Listing 3, the help option has been shown. SITES and WinTR-20 options are created by Python argparse package. The user can select -s (–SITES) or -w (–WinTR-20) to choose which XML file they want to convert. Also, User can specify the input and output file.

```
1SJ1    SITEHYDG04/09/201810:12:23
2 156B4 W1 0     1  1   0   0      7.50      7.50   6 0   1 1   0
3   1   603.43   603.43     0.00   615.00   1    0.00    36.00     0.00
4 1264    0     0.0000   0.00  34.57   34.57   0.00
5     0.0000      0.00       0.00       0.00
5     0.1620      0.06       0.00       0.00
5     0.3240      0.25       0.00       0.00
5     0.4860      0.62       0.00       0.00
.     ......      .....      .....      .....
5   204.6080    34.57        0.00       0.00
```

```
<?xml version= "1.0" encoding= "UTF+-8"+?>
<Run>
<Software>SITES</Software>
<ExampleName>1SJ1</ExampleName>
<ExecutionTime>SITEHYDG04/09/201810:12:23</ExecutionTime>
<Hydrograph name="Principal Spillway Hydrograph Inflow" >
<StartTime units= "hours" value="0.0000"/>
<EndTime units= "hours" value="204.6080"/>
<TimeInterval units= "hours" value="0.1620"/>
<DrainageArea units= "squareMiles" value="0"/>
<data name = "Inflow" units="CFS">
<value>0.00</value>
<value>0.06</value>
<value>0.25</value>
...
<value>34.57</value>
<value>34.57</value>
</data>
</Hydrograph>
```

Figure 7: SITES samplejob1a.DHY(left) and samplejob1a.XML(right)

Figure 7 is the result of applying SITES samplejob1a.DHY. to XML hydrograph translator, and the samplejob1a.XML is the result output file. Most of the data stored in samplejob1a.DHY. can be transfer and stored in samplejob1a.XML. For example, ExampleName: 1SJ1, StartTime: 0.0000, EndTime: 204.6080

and TimeInterval: 0.1620. These parameters have been stored based on the schema XSD file format.

5 Conclusions

In conclusion, a new framework is developed in three different parts. First, WinDAM/BREACH+Dakota framework has been constructed as a visualization tool for the hydraulic engineer. Second, OpenFOAM extensible simulation framework is built and hydraulic engineer can use it on the local machine and on the cluster. Third, canonical XML hydrograph translators are composed for exchanging hydrograph information between different software, and an XML schema XSD file to storing all the hydrograph information. For the future plan, the 2D dam model needs to be reorganized to 3D model, and apply the WinDAM erosion model on top of 3D dam model. In our plan, the updated model can reproduce the dam breach situation automatically based on the erosion model, and estimate how the discharge will affect downstream economics.

The new model needs to interactively update the geometry based on resulting erosion to the hydrograph model and then add back to the erosion model by recalculating corresponding STL file. The related research project has been found: Modeling Scour Depth at Quay Walls due to Thrusters by Van Den Brink [16]. Van Den Brink develop a model which occurs erosion due to the thrusters, we can learn from that project and apply to our framework. For the XML hydrograph translator, the translator can be extended to accept the distinct format of hydrograph output, for example, HEC- HMS to XML file. Also, the translator can be translated back to original file format by only using the XML file.

References

[1] Adams, B.M., et al. Dakota: A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.3 User's Manual. *Sandia Technical Report.*SAND2014-4633, July 2014. Updated May 12, 2017 (Ver. 6.6).

[2] Andresen, D., Neilsen, M., Singh, G., Kalita, P. DHARMA: A grid-enabled domain-specific metaware for hydrology. Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems, 15(2), 791796.

[3] Beocat. https://support.beocat.ksu.edu/ BeocatDocs/index.php/Main_Page/

[4] Extensible Markup Language (XML). https://www.w3schools.com/xml/xml_whatis.asp

[5] Fread, D.L. BREACH: An erosion model for earthen dam failures. National Weather Service, Office of Hydrology, Silver Spring, MD.

[6] Mark J Fenbers. HydroGen (H2). A Software Suite for Extracting Data and Generating Hydrographs for the Web.

[7] Neilsen, M. L. Computational Fluid Dynamics Models for WinDAM.

[8] Neilsen, M.L. Global sensitivity analysis of dam erosion models. *In Proceedings of the 10th International Conference on Scientific Computing* Paper No. CSC-3502, July 22-25, 2013.

[9] Neilsen, M.L., Cao,C. Coupled dam safety analysis using BREACH and WinDAM *In Proc. of the 15th Intl. Conf. on Scientific Computing*Las Vegas, NV, July 17-20, 2017.

[10] OpenFOAM. - software and documentation retrieved from the www.openfoam.org, 2014.

[11] SALOME. http://www.salome-platform.org/

[12] SITES. https://www.nrcs.usda.gov/wps/portal/nrcs/detailfull/national/water/manage/hydrology/?cid=stelprdb1042517.

[13] Tobias Holzmann. Hydraulic Jump of a 2D River. https://holzmann-cfd.de/openfoam/openfoam-tutorials/meshing-running/hydraulic-jump-of-a-2d-river.

[14] Tejral, R.D., Hunt, S.L. Comparing process-based breach models for earthen embankments subjected to internal erosion. *In Proc. of the Joint Federal Interagency Sedimentation and Hydrologic Modeling Conference* Apr. 19-23, 2015, Reno, NV. 10 p.

[15] Temple, D.M., Hanson, G.J., Neilsen, M.L. WinDAM – Analysis of overtopped earth embankment dams. *In Proc. of the ASABE Annual Conference* Paper Number 062105, 2006.

[16] Van Den Brink. Modeling Scour Depth at Quay Walls due to Thrusters.

[17] Visser, K., Tejral, R.D., Neilsen, M.L. WinDAM C earthen embankment internal erosion analysis software. *In Proc. of the Joint Federal Interagency Sedimentation and Hydrologic Modeling Conference* Apr. 19-23, 2015, Reno, NV. 10 p.

[18] WinTR-20. https://www.nrcs.usda.gov/wps/portal/nrcs/detailfull/null/?cid=stelprdb1042793.