Physics Modeling of The Dam-Break Problem (One-Dimensional)

Abstract

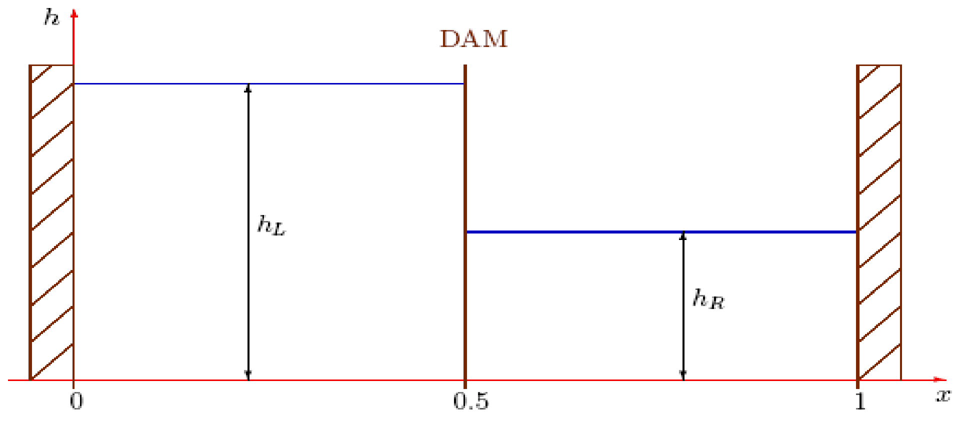
In modeling dam-break floods in natural channels, the practicing engineer must decide whether to use a one-dimensional (1D) or a two-dimensional (2D) numerical model. Here in the examination of what might be considered a 2D problem, it is clearly illustrated that a 1D formulation provides an excellent solution. The solution is based on a formulation of the St.-Venant equations developed for rectangular channels of varying widths, ...

DamBreak++ Physics Simulator

We are going to give an example of how to program a physics simulator using our programming environment or framework. The physics simulator that we are going to program will simulate the wave propagation (nonlinear flow with shocks) in the so-called dam break problem in a horizontal channel, which is a classical problem in fluid mechanics and is motivated by numerous applications in environment and industrial processes. This classical test case is considered a benchmark for comparison of the performance of numerical schemes specially designed for discontinuous transient flow. Although defined by the system of homogeneous shallow water equations for the ideal case of a flat and frictionless channel of unit width and rectangular cross section (known analytical solution), it is widely considered a standard test case for validation of schemes.

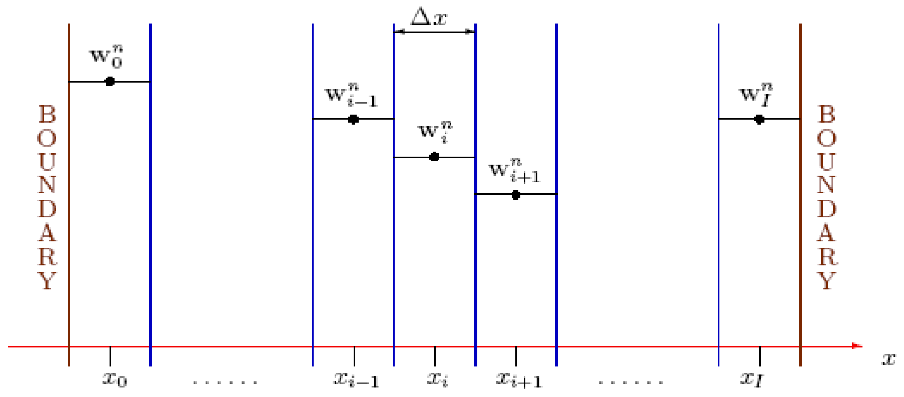
Problem description (initial condition)

A rectangular column of water, in hydrostatic equilibrium, is confined between two walls. At the beginning of the calculation the right wall is removed, and the water is allowed to flow out to the horizontal wall. The initial setup of the dam-break problem is shown in the figure below.



Key Concepts of our application

This phenomenon is best described by the equations of St-Venant (one-dimensional). We need to solve a set of equations by numerical method, conservative finite difference method. The problem consists of evaluating a discrete equation on each cell, these equations are in a conservative form, which means that the variation of state variable inside a cell depends only on the fluxes (in/out).



Discretization

Main concept of discretization (transformation from continuum to discrete space).

- **Global discretization** concept can be characterized by three features:
  - The domain of the problem is represented by a collection of simple domains, called cell. In the 1-D case the domain [a,b] of the equation is discretized as a series of points x_i, i=0,..., N,N-1 where the solution "w" of the equation is discretized as
  - Over the cell, the physical process is approximated by functions of desired type (polynomials or otherwise), and an algebraic equation relating physical quantities at selective points, called nodes, of the element are developed
  - The discretized equations are constructed over each cell

- **Space discretization concept**

Space discretization consists of setting up a mesh or a grid by which the continuum of space is replaced by a finite number of points where the numerical values of the variables will have to be determined.

- **Time Discretization concept**

The time discretization can be categorized into 2 types: Explicit/implicit time stepping. In the case of explicit scheme unknown variable at the current time level depends on the previous time level. Implicit schemes need to solve a linear/non-linear system of the type of Ax=b. More than one set of variables are unknown at the same time level.

- **Discrete representation concept**

Once the mesh has been defined the equations can be discretized, leading to the transformation of the differential or integral equations to discrete algebraic operations involving the values of the unknowns at the mesh points. The basis of all numerical methods consists of this transformation of the physical equations into an algebraic, linear, or non-linear, systems of equations.

Requirements Definition

We want this system to be flexible, especially regarding the following features:

- **It shall be possible to change the numerical method we use**

We may have several varied reasons for wanting to change the numerical method. The numerical method yields wrong answers for the problem. This includes unstable solutions, solutions that are mathematically correct but physically incorrect, or solutions that are not accurate enough. Another reason for changing the method is efficiency.

- **It shall be easy to construct a program that solves the Dam-Break Problem**

We stress the ability to build a physical algorithm quickly by using already developed components, by writing a new algorithm and adding it to the simulation. Moreover, we require that the system should be easily extendible. It is not enough if we can solve a predefined set of problems with a predefined set of solution methods. A user shall be able to describe recent problems, or new solution methods, and add these to the system. This is particularly useful if, for instance, the user develops a new numerical method and wants to compare this method against other methods.

- **The system shall be flexible. Components shall be changeable**

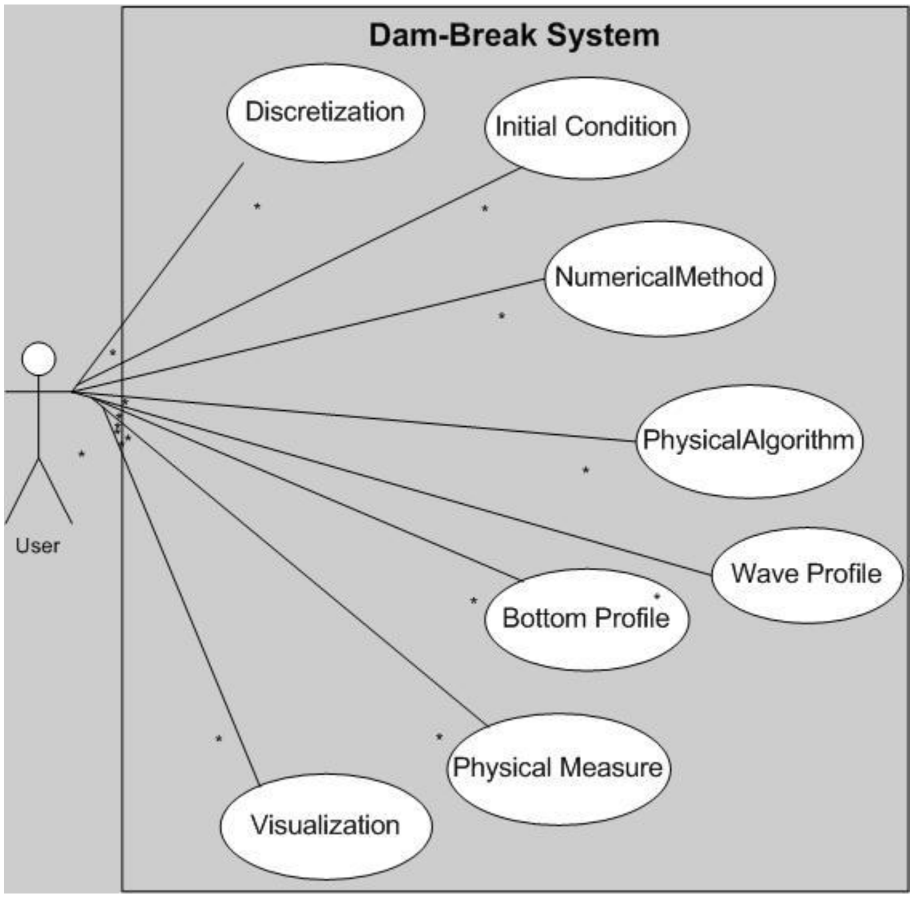
All pieces of the physical algorithm (solve numerically the PDE) shall easily exchangeable. Particularly, it is easy to switch between different implementations of the explicit integrator, or to switch between a flux algorithm (or numerical scheme). When validating numerical schemes, it would be nice to mix different alternatives and compare.

- **It shall be easy to extend the system with new components**

The flux algorithm could be implemented with a Godunov-type scheme or with an ENO flux extrapolation. We need to set a high-level abstraction which identify (represent) the key concepts of the application and let the specific (detail) to subclass;

Use Case Diagram

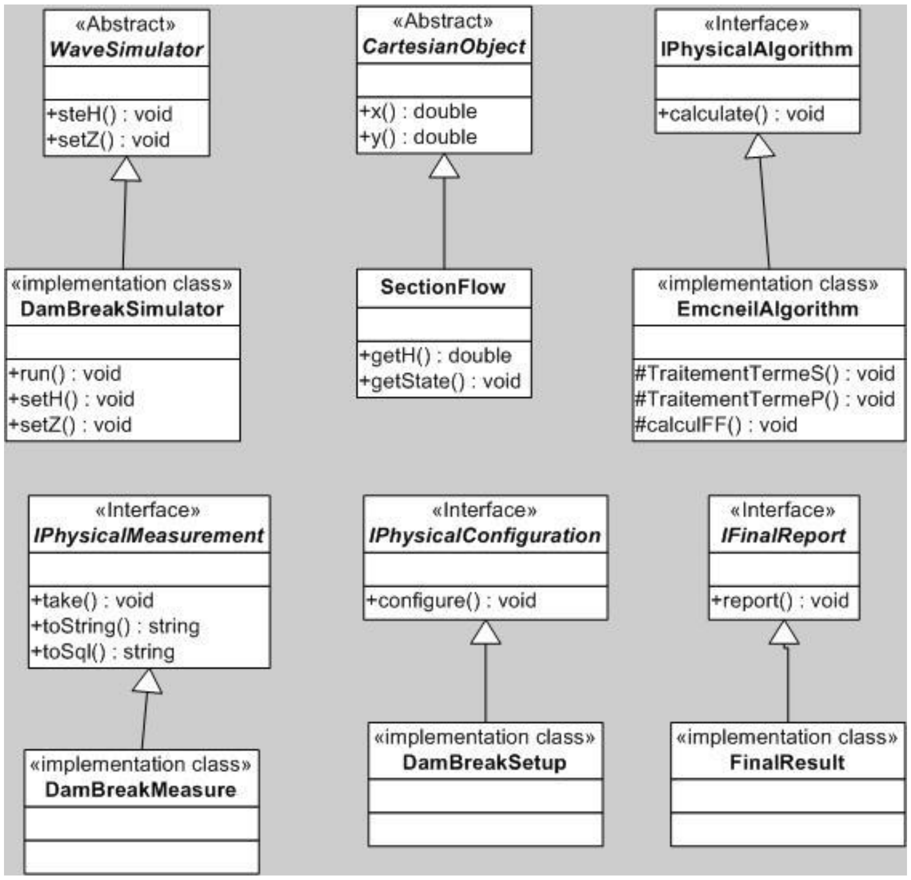
There are several things that should be tracked during the simulation. The simulation should record and report the wave's position, velocity, and acceleration. It should also calculate the wave's kinetic, potential, and total energies and report them. Lastly, the simulation should report the maximum height achieved by the wave.



Class Diagram

The abstractions presented above are now expressed in the UML language called class diagram. Central to the UML notation is the concept of class data. A class is an abstract, user-defined description of a type of data. It identifies the attributes of the data and the operations that can be performed on instances (i.e., objects) of the data. This modeling technique allows description of a system using the same terminology as the corresponding real-world objects and their associated characteristics.

.



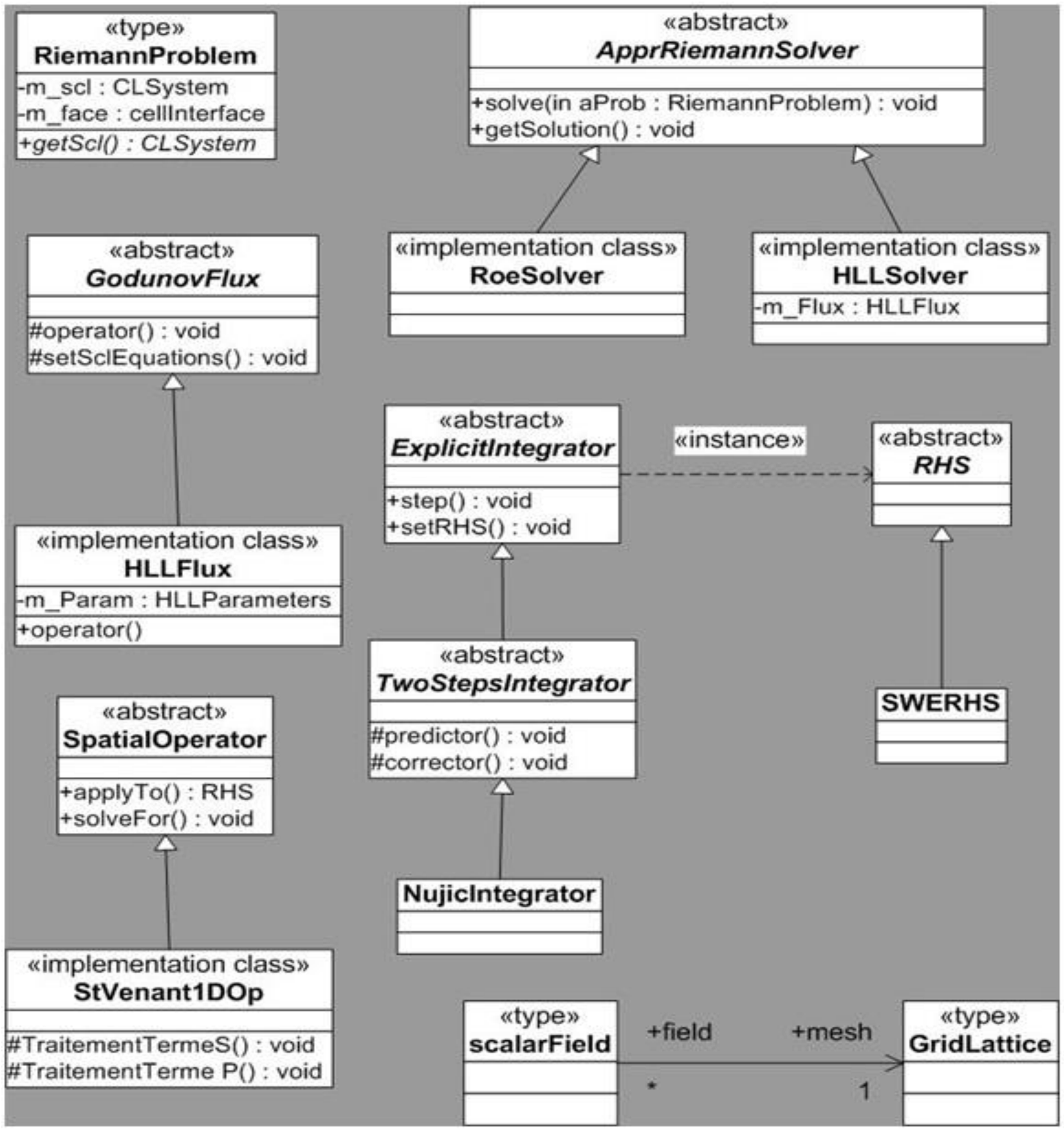
.

The four interfaces shown here are the ones that each developer intending to use the Framework will have to provide implementations for.

The following steps outline how to develop a simulation using the Framework

The names of each interface are self-explanatory. The **IPhysicalConfiguration** class implementation is meant to set up the initial conditions of the simulation. In the sample application it sets the sections initial water elevation and discharge. The Physics Simulator uses the **IPhysicalAlgorithm** class implementation to numerically integrate the equations governing the simulation. In the sample application the Runge-Kutta algorithm is implemented for the Shallow-water equations. The **IPhysicalMeasurement** class implementation dictates what physical quantities will be periodically measured. The **IFinalReport** is provided so that summary statistics and final calculations can be reported to the user in a plain text format.

Other types of the **DamBreak++** library is presented in the diagram below.



scalarField/GridLattice class (finite difference discretization) Represent the grid (discretization parameters finite difference). This class represents the spatial discretization key concept described above. It holds the parameters of the spatial discretization that will be used by the scalar field, which strongly depend on the discretization.

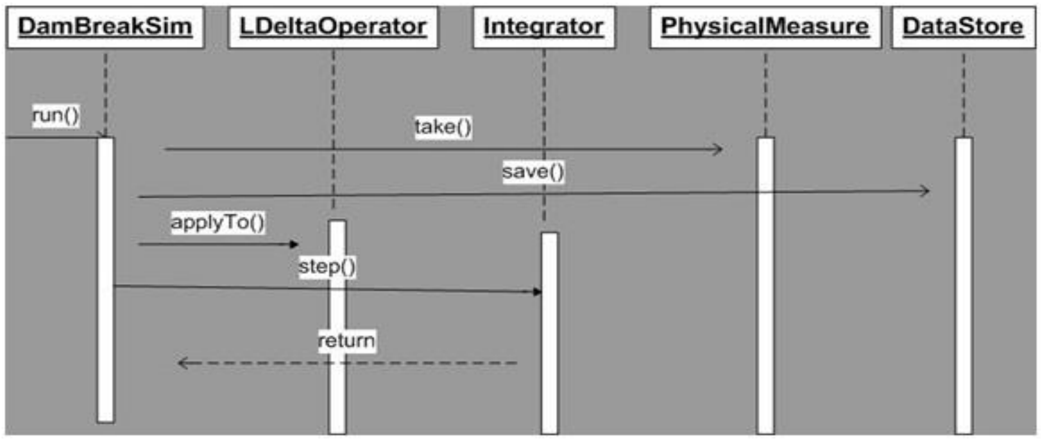
ExplicitIntegrator Numerical integration (explicit time stepping). Also, time stepping can be achieved by different algorithm, Runge-Kutta of 2nd order, … up to many orders approximation (Two-steps family).

Riemann Solver

In the problem that we are interested in, we need to approximate the numerical flux at cell-interface by some algorithm, for example, we can choose to solve it by a Riemann solver, or we can use some extrapolation method (ENO flux extrapolation). Numerical algorithm to solve the problem taking account the physics, numerical treatment of each tem of the equation

Sequence Diagram

Sequence diagram belongs to the behavior diagram. It shows the calls that are made to the objects. The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur. One of the primary uses of sequence diagrams is in the transition from requirements expressed as use cases to the next and more formal level of refinement.



References
"A C++ Differential Equations Solver Using Object-Oriented Numeric", J. Belanger

Abstract
Over the last few years, we have been migrating from a small library of numerical code originally written in C to C++. In this report, we present the mathematical abstractions used and how object-oriented programming techniques are applied for scientific software design. Finally, implementations details are provided including the relationship between data structure. The result is tight, readable code that is easy to maintain and extend. Example with Shallow water equations is drawn from our prototype C++ based environment.