

# How (the import of BAW data into) Delft3D has been made OpenMI compliant

Integrated Modelling Workshop  
at the BAW in Hamburg  
31<sup>st</sup> October 2008



Adri Mourits, [adri.mourits@deltares.nl](mailto:adri.mourits@deltares.nl)  
Peter Schade, [peter.schade@baw.de](mailto:peter.schade@baw.de)  
Günther Lang, [guenther.lang@baw.de](mailto:guenther.lang@baw.de)



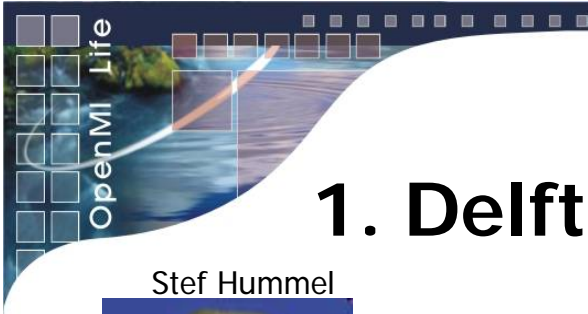
# Overview

1. Introduction
2. How the "database" GEIWrapper has been made OpenMI compliant  
data flow, code examples
3. How Delft3D-FLOW has been made OpenMI compliant  
single domain, multi domain
4. Perspectives  
Deltares, BAW



# 1. Introduction





# 1. Delft Hydraulics



# Deltares OpenMI



1 Jan 2008

Stef Hummel



OpenMI experts



Peter Gijbers

Appl. comm. expert



Irv Elshoff

Delft3D expert



Bert Jagers

OpenMI applications



Jaco Stout

Delft3D maintainers



Adri Mourits

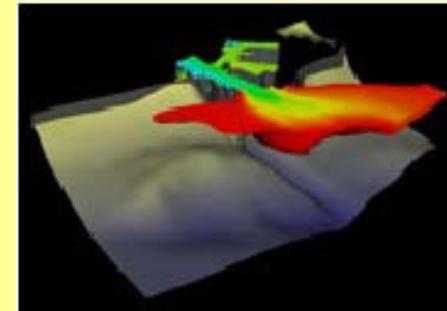
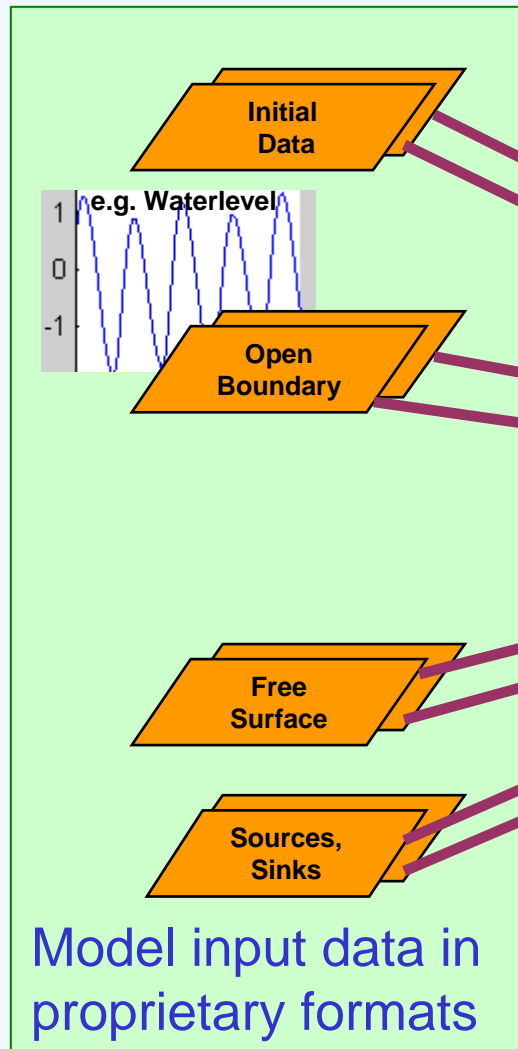


Frank Platzek

## 2. How the “database“ GEIWrapper has been made OpenMI compliant



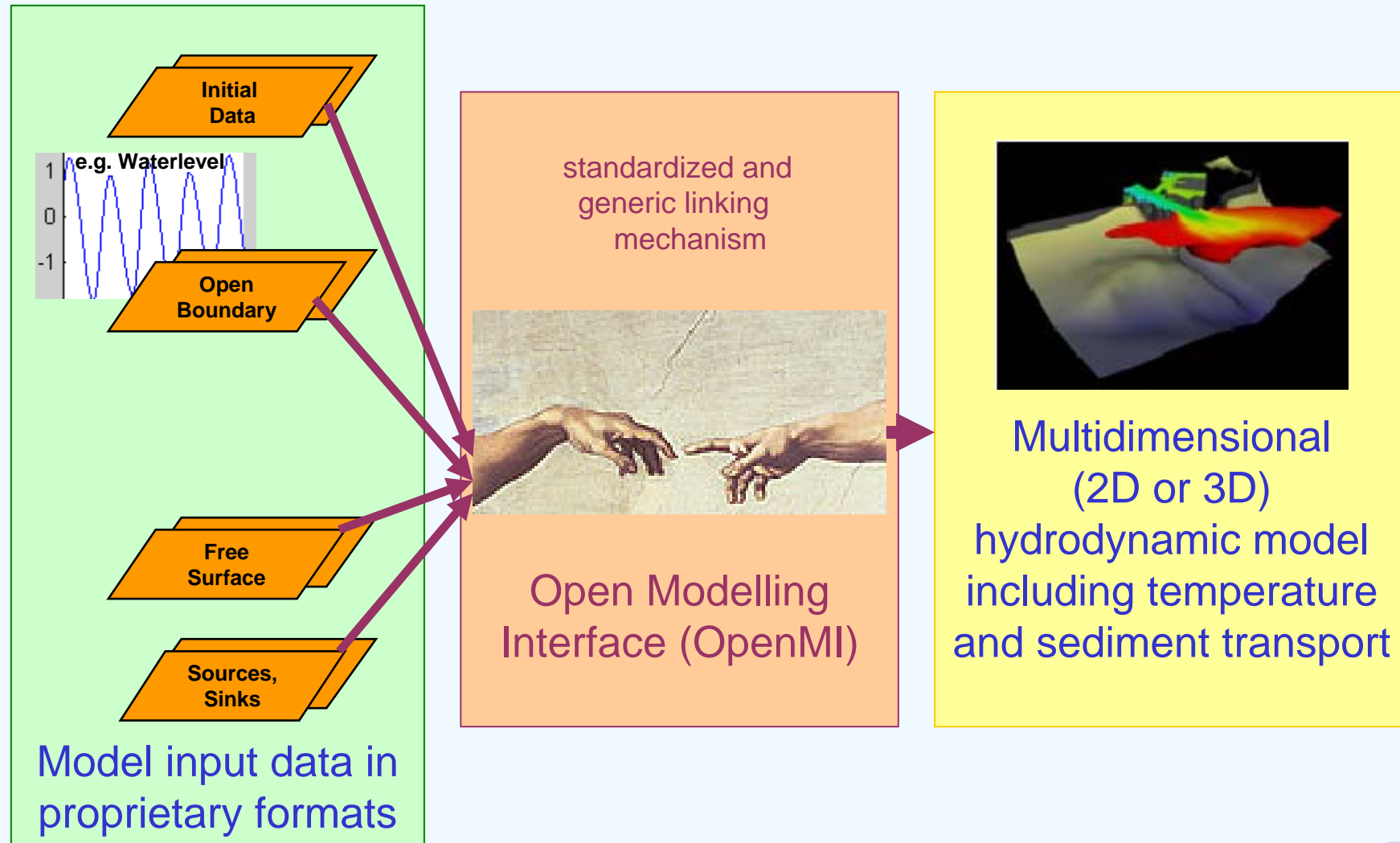
## 2. How would you integrate a model?



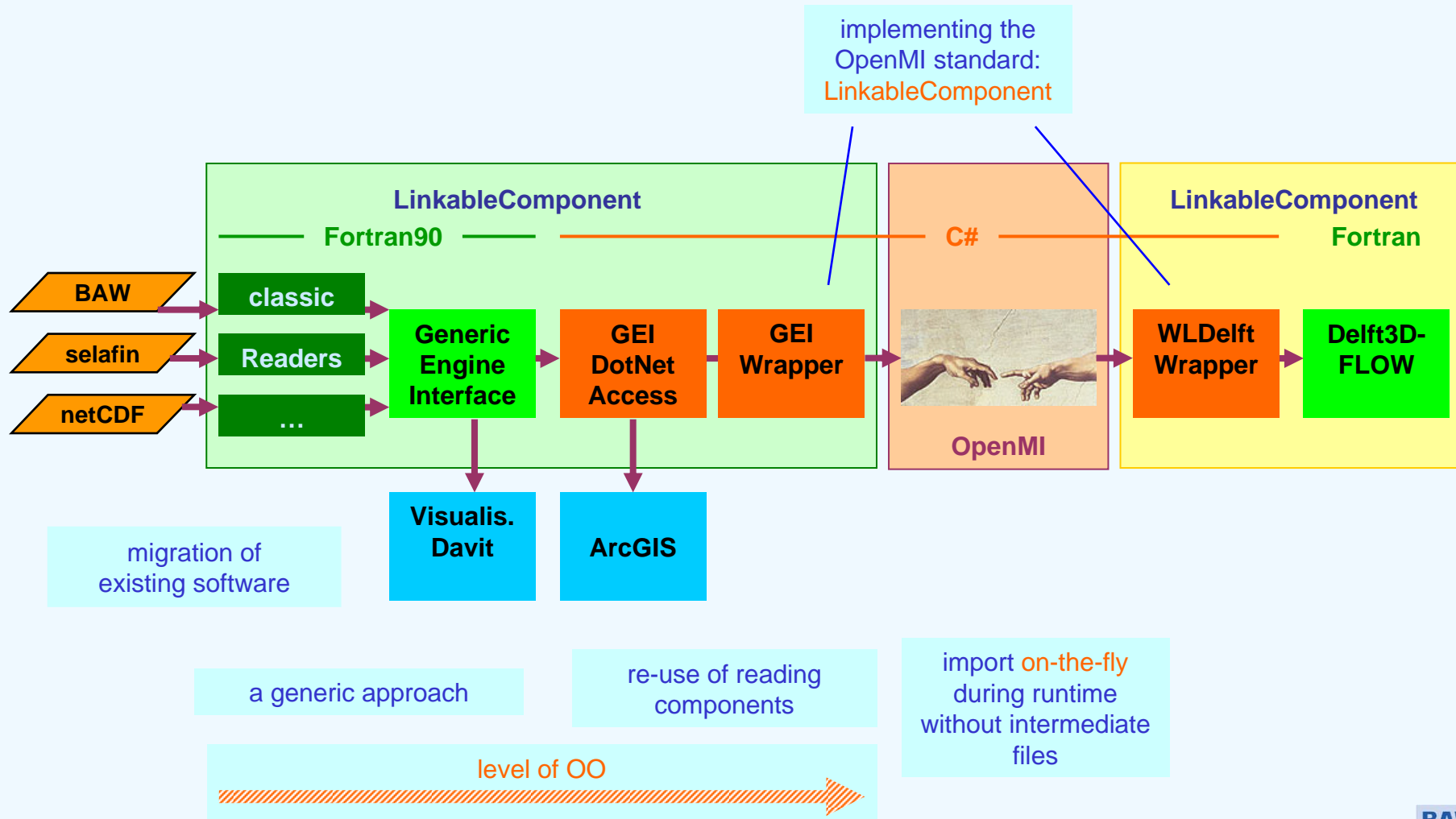
Multidimensional  
(2D or 3D)  
hydrodynamic model  
including temperature  
and sediment transport



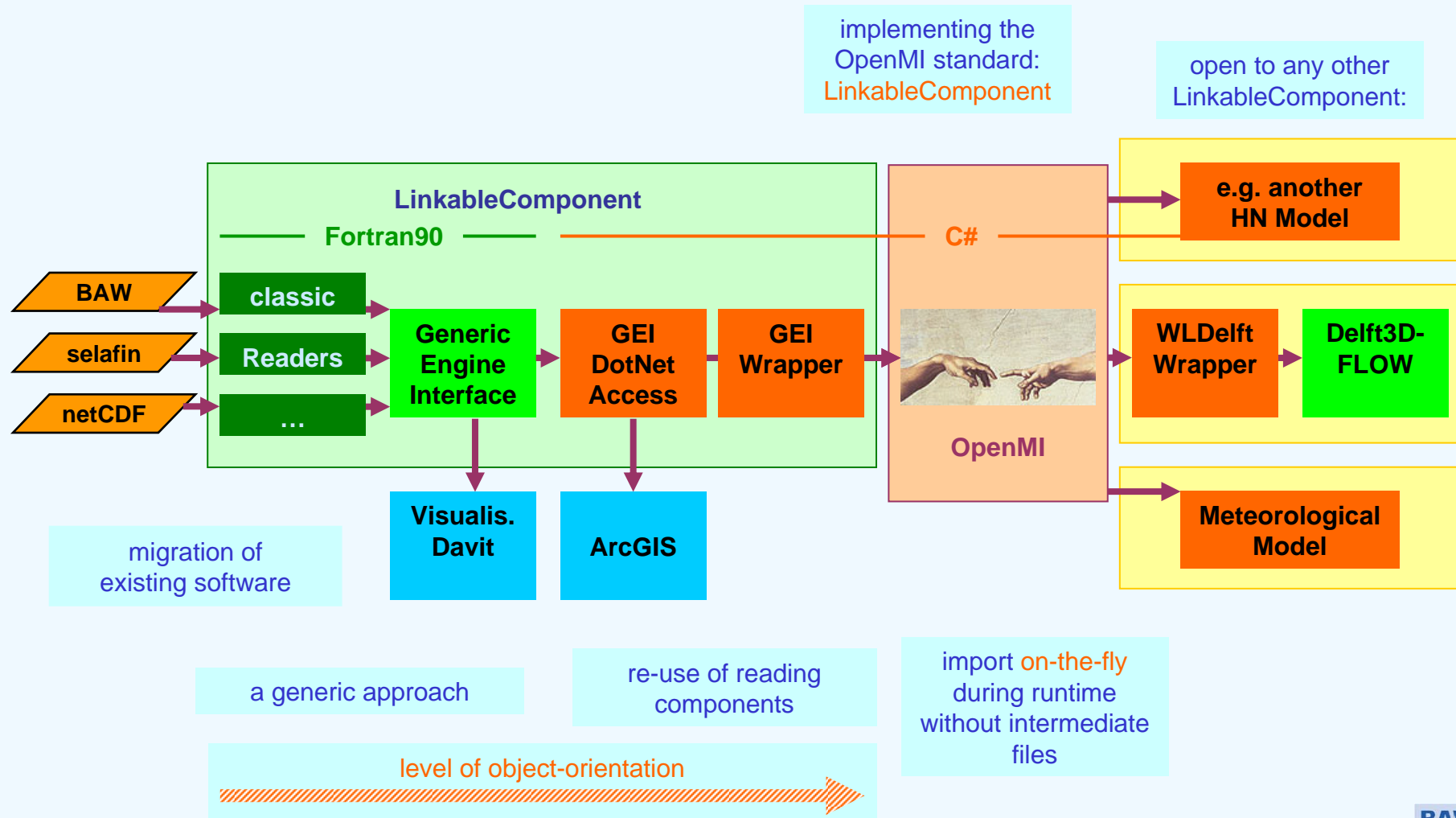
## 2. How would you integrate a model?



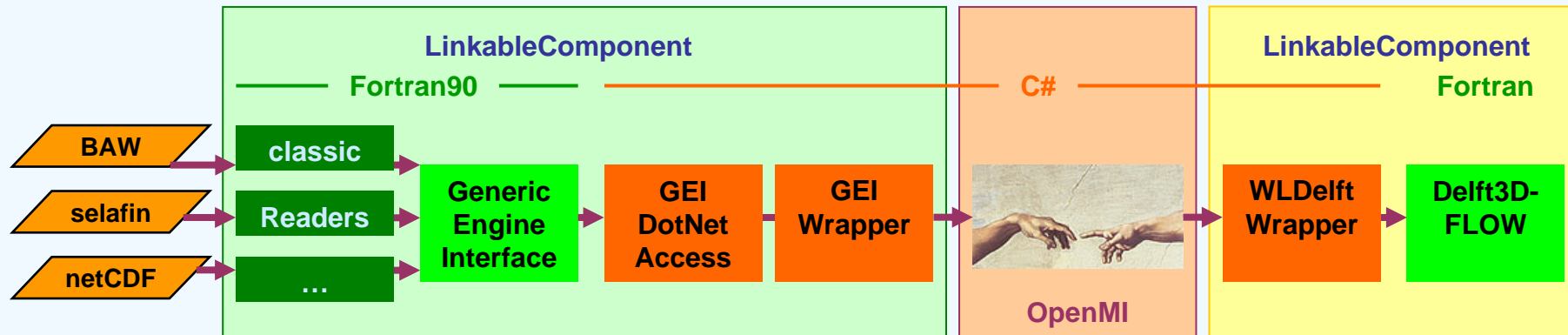
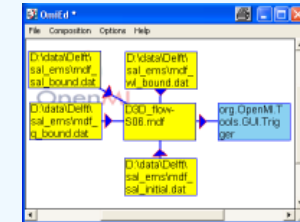
# 2. OpenMI Compliant Import



# 2. OpenMI Compliant Import



# 2. Linking People and Organisations



G. Lang, J. Jürges, A. Plüß,  
U. Schiller, S. Spohr,  
R. Schubert, U. Vierfuß



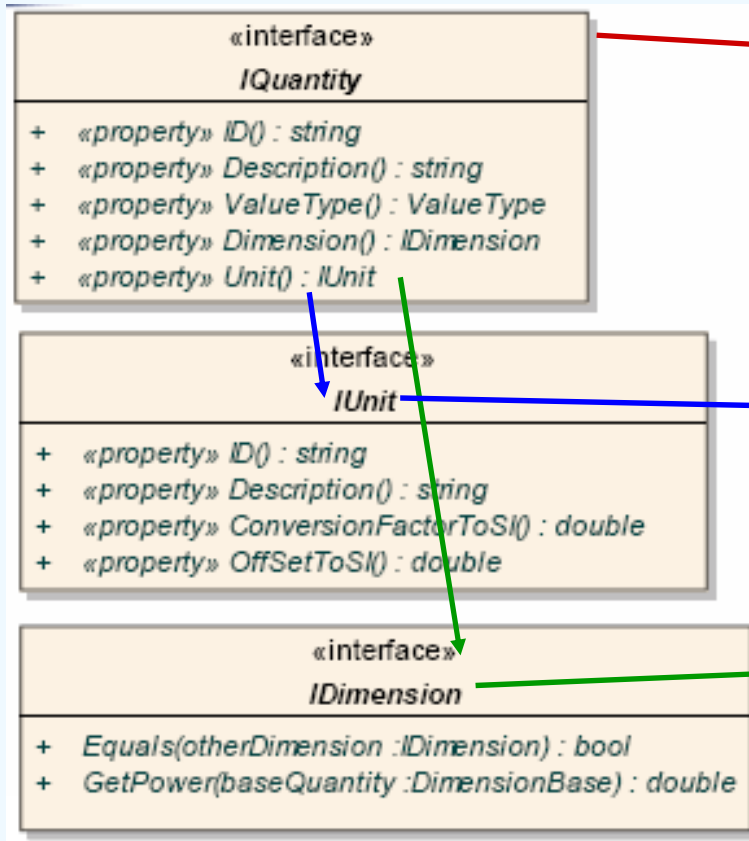
A. Mourits, R. Brinkman,  
G. Donchyts, B. Jagers,  
D. Levelt



# 2. Core of the GEI\*: Base Modules

Specifications in the OpenMI interface standard

Image of the standard in Fortran 90 modules: datatypes / methods



```

TYPE , PUBLIC :: t_omi_quant
  PRIVATE
  CHARACTER (LEN=c_len_id):: id
  CHARACTER (LEN=c_len_de):: description
  INTEGER :: valuetype
  TYPE (t_omi_dim) :: dimension
  TYPE (t_omi_unit) :: unit
END TYPE t_omi_quant
  
```

```

TYPE , PUBLIC :: t_omi_unit
  PRIVATE
  CHARACTER (LEN=c_len_id) :: id
  CHARACTER (LEN=c_len_de) :: description
  REAL (KIND=Double) :: convfactosi
  REAL (KIND=Double) :: offsettosi
END TYPE t_omi_unit
  
```

```

TYPE , PUBLIC :: t_omi_dim
  PRIVATE
  REAL (KIND=Double) :: length
  REAL (KIND=Double) :: mass
  REAL (KIND=Double) :: time
  REAL (KIND=Double) :: electriccurrent
  REAL (KIND=Double) :: temperature
  REAL (KIND=Double) :: amountofsubstance
  REAL (KIND=Double) :: luminousintensity
  REAL (KIND=Double) :: currency
END TYPE t_omi_dim
  
```

GEI\* = Generic Engine Interface



# 2. Spatial Interpolation in C# Code

Available mapping methods in  
OATC.OpenMI.SDK.Spatial

Use of mapping methods  
in GEIWrapper

Target Source	● Point	— Line	⋈ PolyLine	△ Polygon
● Point	✓	✓	✓	✓
— Line	✓	✗	✗	✓
⋈ PolyLine	✓	✗	✗	✓
△ Polygon	✓	✓	✓	✓

### Initialisation once:

```
sourceElementSet.Type = "XYPoint"
targetElementSet.Type = "XYPoint"
```

```
mapper.Initialise( "Nearest" ,
                  sourceElementSet,
                  targetElementSet);
```

### Computation for each timestep:

```
targetValues =
(ScalarSet) mapper.MapValues (sourceValues);
```



# 2. Pros and Cons ...

## ... for Developers

- ... but there are **guidelines** and **training courses**;
- OpenMI is a **mature** and **versatile** standard.
- The **migration** is **much easier**, if the classic components already have **defined interfaces**.
- for integrating further models / formats: **re-use of components** due to **generic approach** and **OpenMI Standard**.

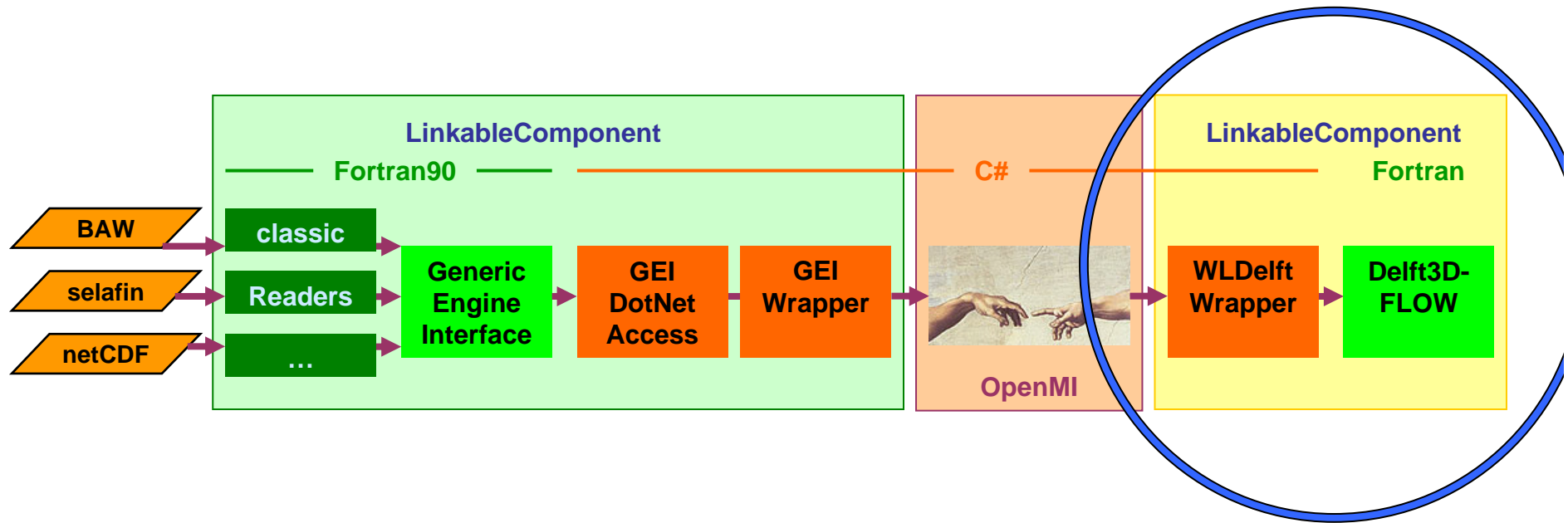
- making a model Open compliant **costs an effort**, too, ...



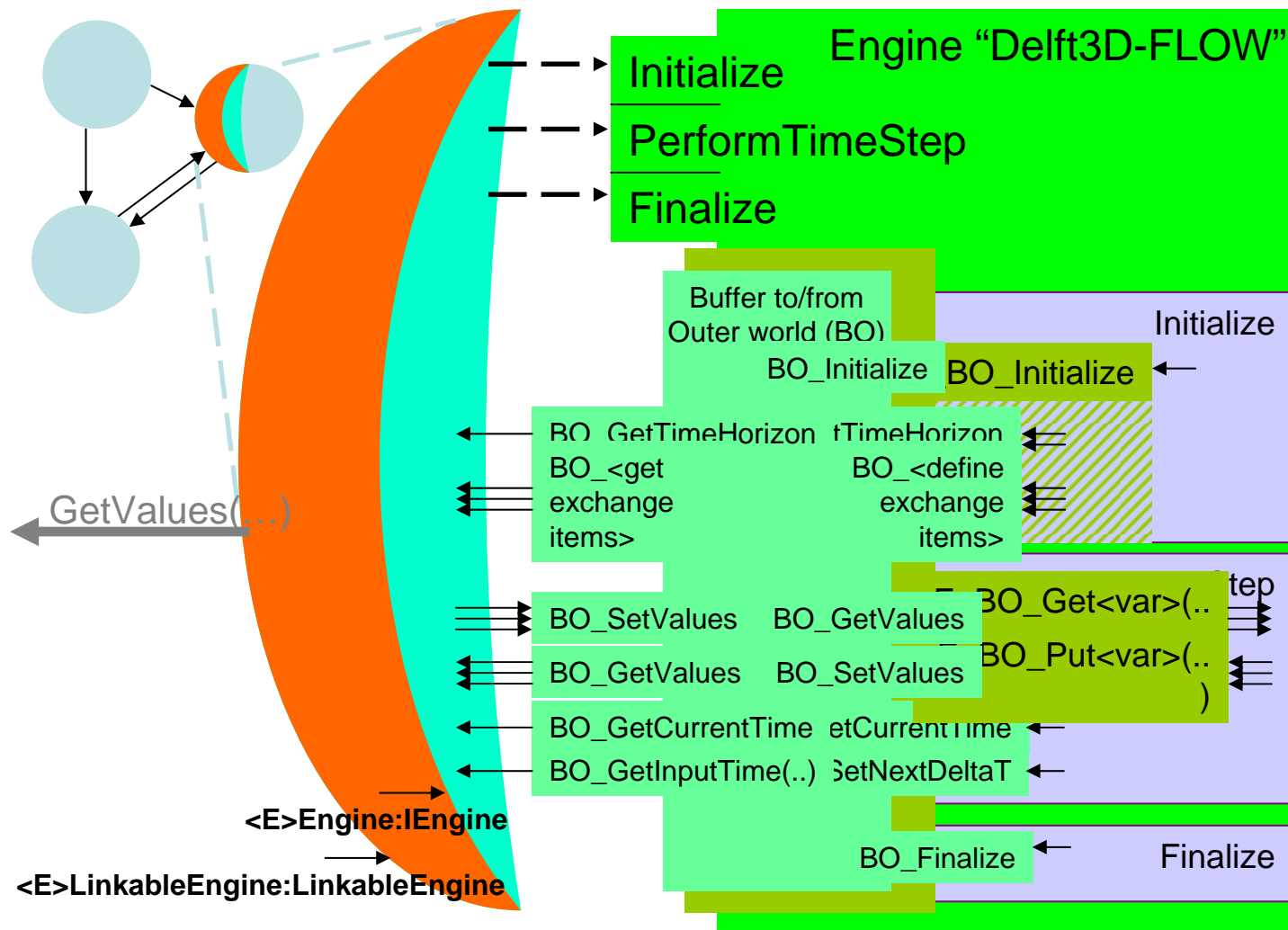
# 3. How Delft3D-FLOW has been made OpenMI compliant



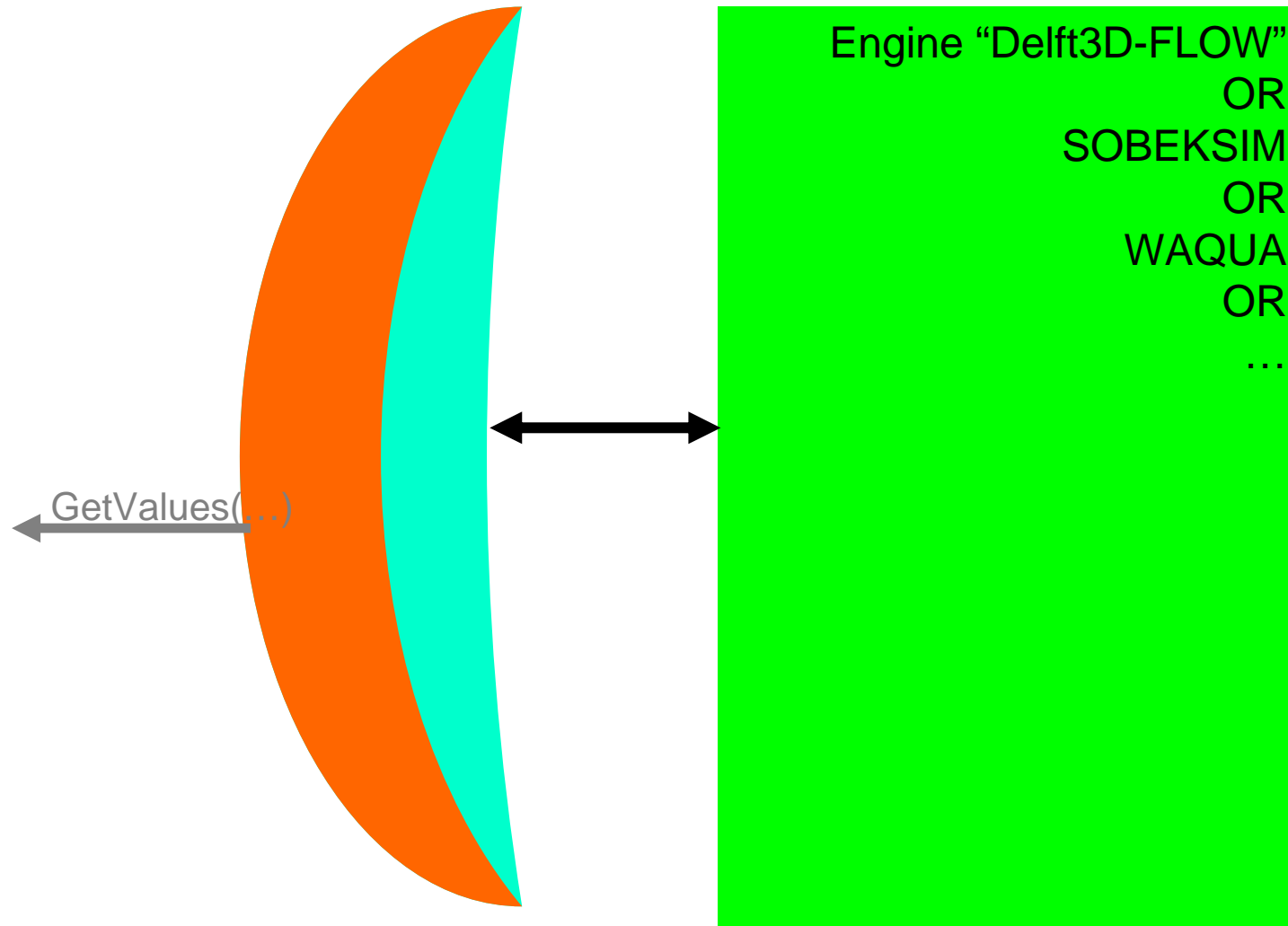
# 3. From GEI to Delft3D-FLOW



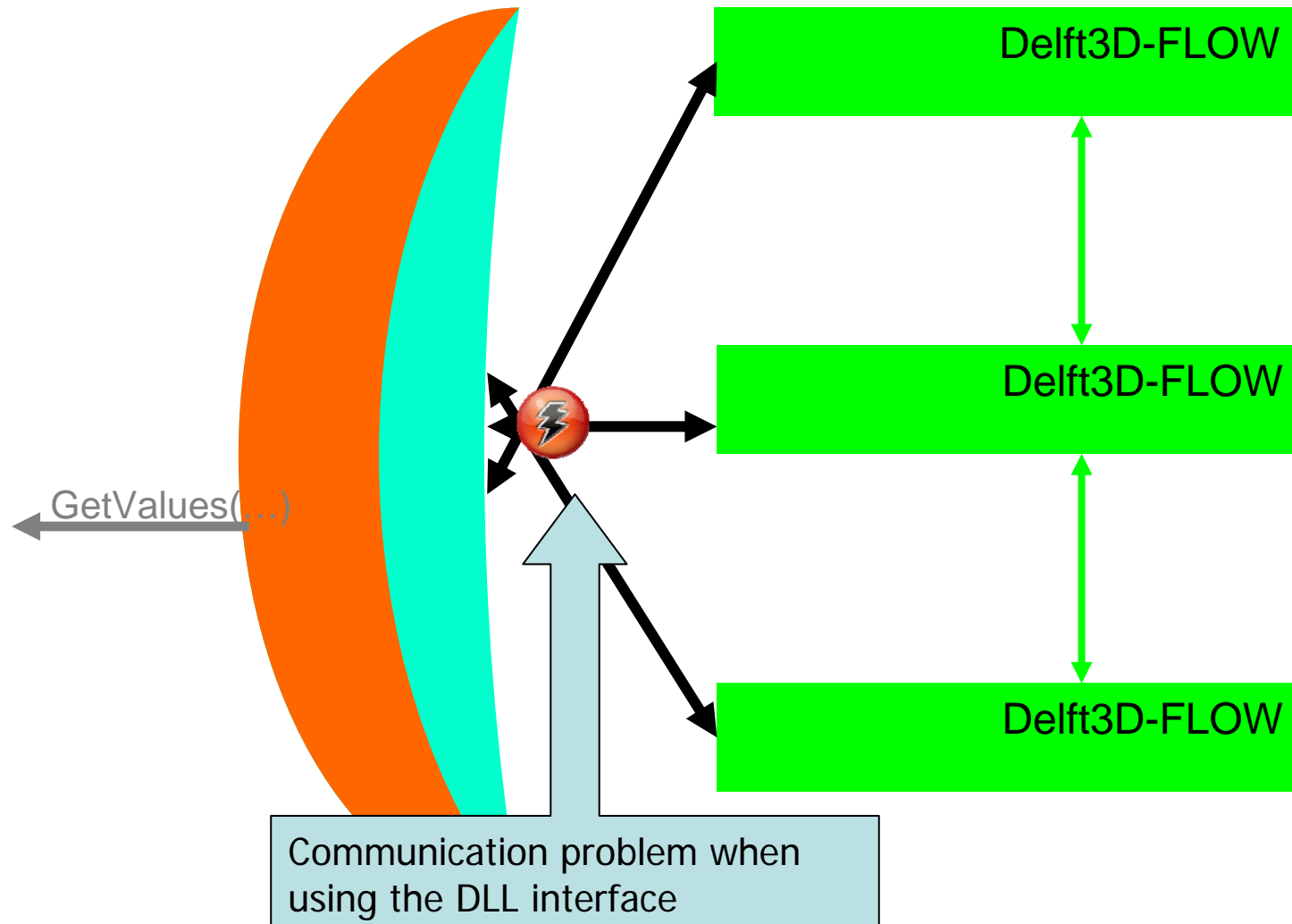
### 3. Delft3D-FLOW, single domain



### 3. One wrapper, different engines

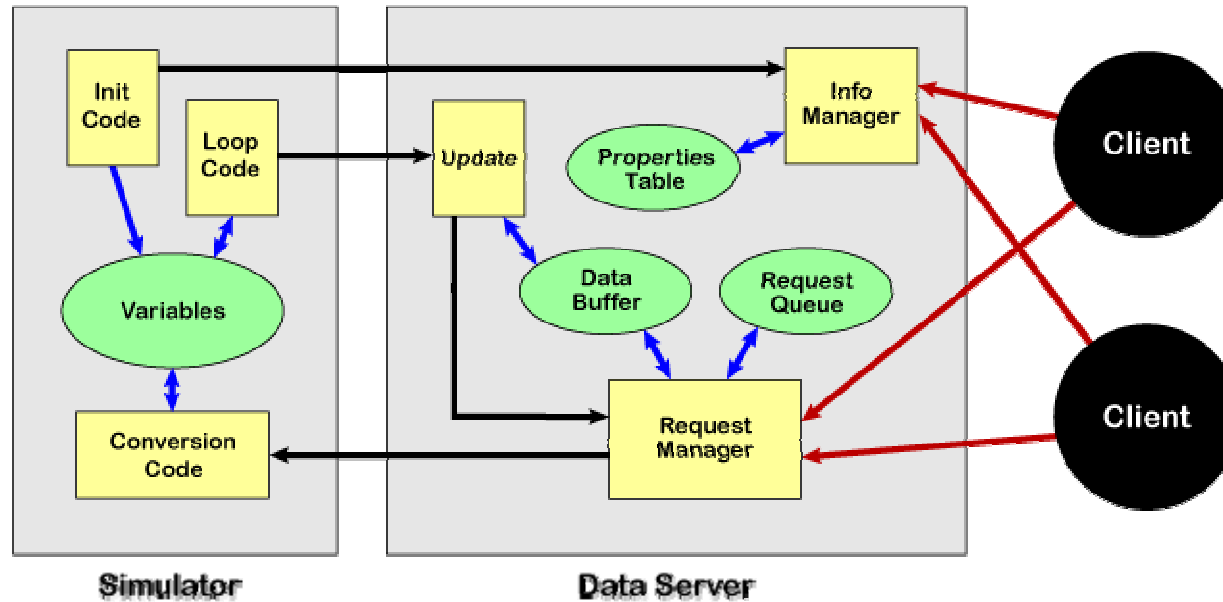


### 3. Delft3D-FLOW, multi domain



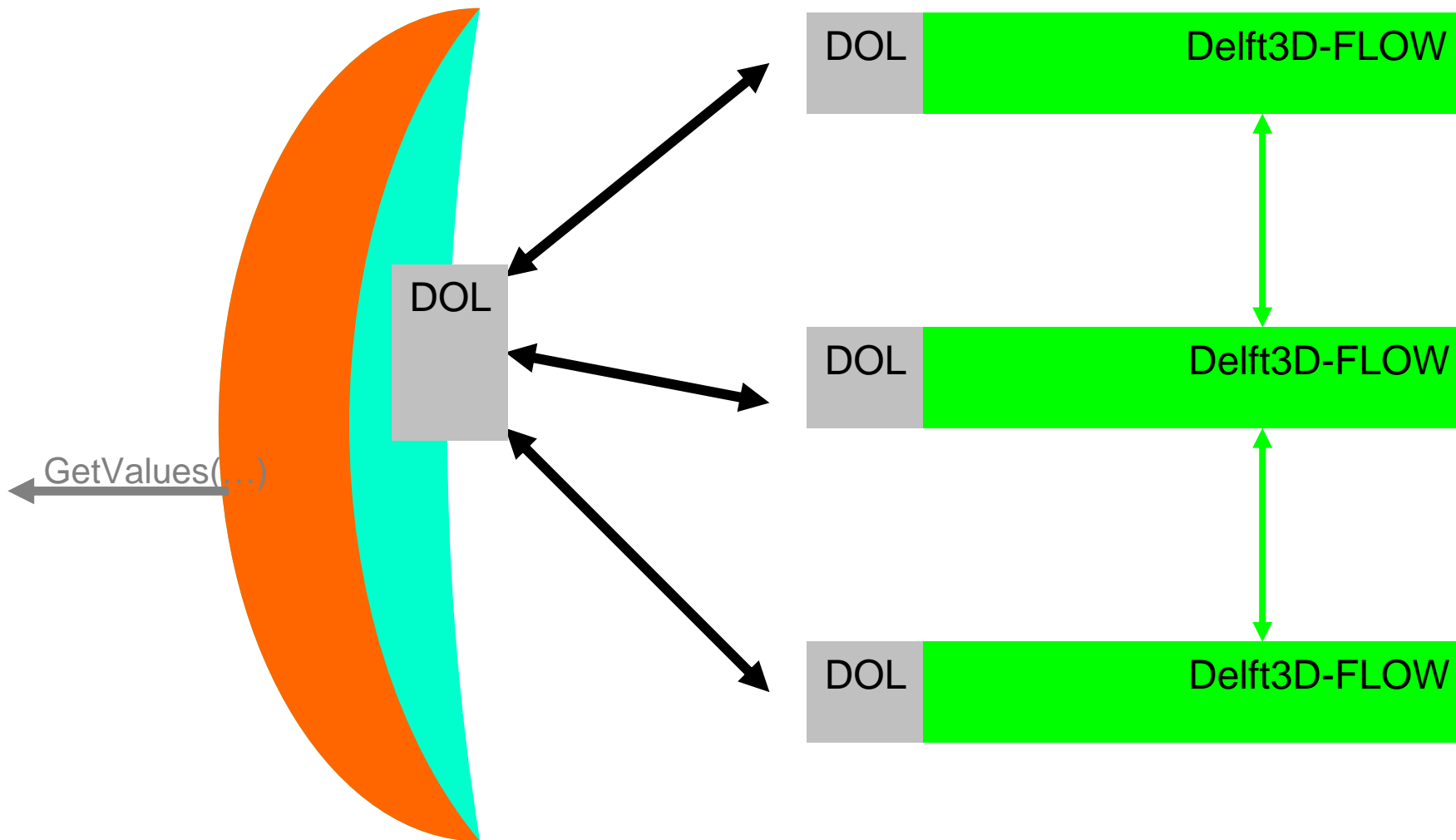
Communication problem when using the DLL interface

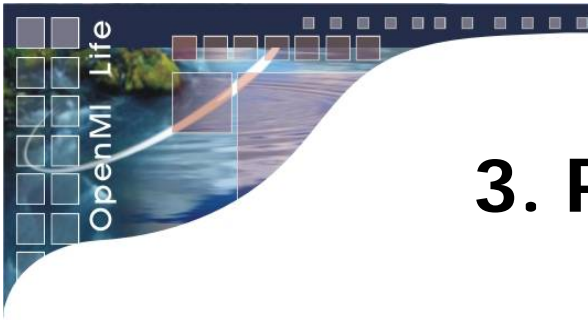
### 3. Solution: DelftOnline (DOL) Developed by Deltares



- Data server in between a simulator (Delft3D-FLOW) and clients requesting data (wrapper, online visualisation, ...)
- Simulator and clients may run on different hosts on the internet
- Data is communicated via JNI

### 3. Delft3D-FLOW, multi domain





### 3. Pros and

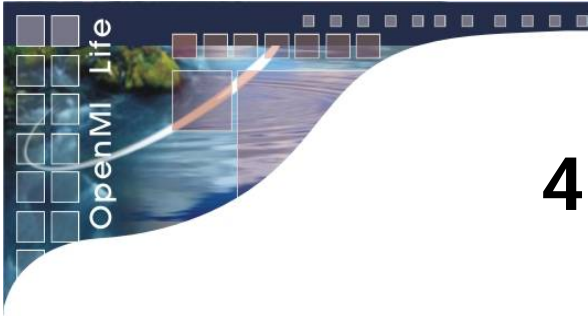


- One wrapper for all kernels (single/multi domain)
- Delft3D may run on a different machine/cluster (user)

- Complex (programmer)  
mixture of Fortran-DLL / java DOL / C# wrapper

# 4. Perspectives





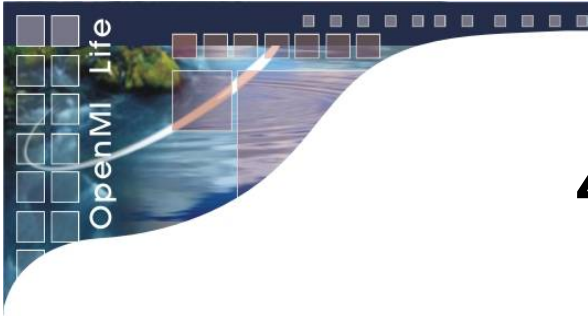
## 4. Deltares realisation 2008



Main Development Stream (**MDS**) in 2008:

- Internal code reorganization
- Modular code philosophy:
  - Independent, re-usable, as small as possible building blocks
  - Usable for PC and HPC
  - Key module: External Communication (**EC-module**) including:
    - Read/write module for time/space related data
    - OpenMI module
    - Interpolation module
    - ...

Delft3D-FLOW\_OpenMI is a separate research version.



## 4. Deltares plans for 2009



- Finish EC-module
- Merge Delft3D-FLOW\_OpenMI into MDS, using the new EC-module
- Unstructured grid

Advantages of Modular code:

- EC-module is easy to extend
- EC-module is easy to be used by other applications

# 4. Perspectives

- **OpenMI 2.0** (OpenMI Association)
- **OpenMI goes Linux** (BAW):  
Mono: .NET on Linux  
  
Wish List:
- **more mapping methods:**  
polar coordinates, interpolation by triangulation
- **parallel computing**
- **more 2D/3D OpenMI compliant models**  
(users and vendors of software)

