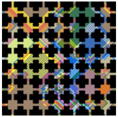


A Methodology for Multiscale-Multiscience modeling and simulations

Bastien Chopard

Computer Science Department
University of Geneva, Switzerland

ICES workshop, Geneva, Nov. 22, 2011

Introduction

- Modeling and simulation are central to modern science
- There is a need to develop new and better numerical approaches
- For instance the Cellular Automata (CA) and Lattice Boltzmann (LB) approaches have been successful alternatives to standard computational methods

CA and LB methods

- a discrete mathematical abstraction of reality
- The macroscopic behavior depends very little on the details of the microscopic interactions.
- Only "symmetries" or conservation laws survive.
- Consider a fictitious world, particularly easy to simulate on a (parallel) computer with the desired macroscopic behavior.

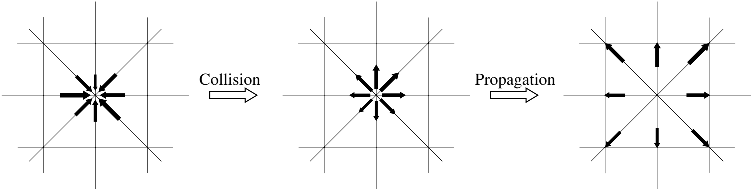
From hydrodynamics PDE

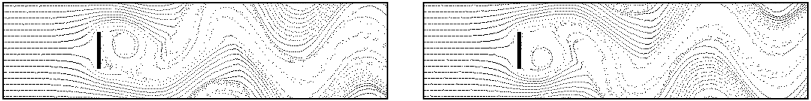
$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u}$$

$\boxed{\textbf{phenomena} \rightarrow \textbf{PDE} \rightarrow \textbf{discretization} \rightarrow \textbf{computer solution}}$

...to virtual fluids

$\boxed{\textbf{phenomena} \rightarrow \textbf{computer model}}$



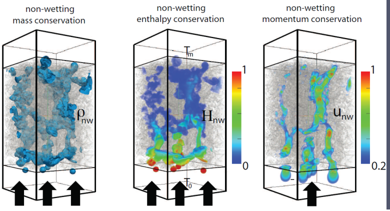


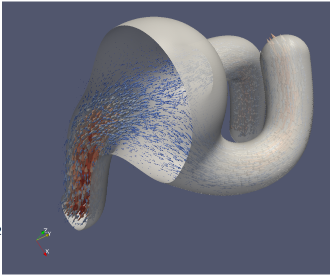
LB simulations

- Simple, flexible, intuitive, efficient
- Palabos software[1] (Jonas Latt)
  - Free, Open source software (`http://www.lbmethod.org/palabos`)
  - Python interface or full C++
  - complex multi-physics, complex data-structures
  - Offer a wide range of models, boundary conditions, dynamics
  - Can handle large scale parallel simulations
  - Automatic high performance parallelization. Scales well up to thousands of cores

1. `http://www.lbmethods.org/palabos`

Examples of LB simulations





Multiscale, multiscience modeling framework

- Propose a modeling and simulation framework for multiscale, multisciences complex systems

Multiscale, multiscience modeling framework

- Propose a modeling and simulation framework for multiscale, multisciences complex systems
  - Theorectical concepts : Complex Automata (CxA)

Multiscale, multiscience modeling framework

- Propose a modeling and simulation framework for multiscale, multisciences complex systems
  - Theorectical concepts : Complex Automata (CxA)
  - Software environment : The MUSCLE library

Multiscale, multiscience modeling framework

- Propose a modeling and simulation framework for multiscale, multisciences complex systems
  - Theorectical concepts : Complex Automata (CxA)
  - Software environment : The MUSCLE library
  - Validation application : In-stent restenosis

Multiscale, multiscience modeling framework

- Propose a modeling and simulation framework for multiscale, multisciences complex systems
  - Theorectical concepts : Complex Automata (CxA)
  - Software environment : The MUSCLE library
  - Validation application : In-stent restenosis
  - A Multiscale Modeling Language : MML

Multiscale, multiscience modeling framework

- Propose a modeling and simulation framework for multiscale, multisciences complex systems
  - Theorectical concepts : Complex Automata (CxA)
  - Software environment : The MUSCLE library
  - Validation application : In-stent restenosis
  - A Multiscale Modeling Language : MML
- The follow up : the MAPPER EU project

The COAST Project

- A. Hoekstra, A. Caiazzo, E. Lorenz, U. Amsterdam (Netherlands)
- R. Hose, P. Lawford, D. Evans, J. Gunn, U. Sheffield (UK)
- B. Chopard, J-L. Falcone, B. Stahl, U. Geneva (Switzerland)
- M. Krafczyk, Y. Hegwald, TU Brauschweig (Germany)
- J. Bernsdorf, D. Wang, NEC (Germany)

- Joris Borgdorf, U. Amsterdam (Netherlands)

Motivations

- Very few methodological papers in the literature.

Motivations

- Very few methodological papers in the literature.
- Multiscale strategies are usually entangled with applications.

Motivations

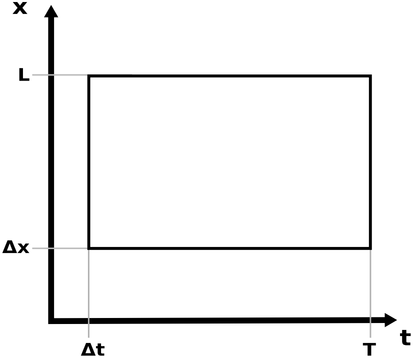
- Very few methodological papers in the literature.
- Multiscale strategies are usually entangled with applications.
- Can we develop a framework that help the design and deployment of complex multiscale-multiscience applications ?

From a multiscale systems to many single-scale systems :

Let us consider a system of size L evolving over a time T.
Computation with space and time discretization Δx and Δt

Resolved spatial scales : $\Delta x < \xi < L$ and
Resolved temporal scales : $\Delta t < \tau < T$

Scale Map

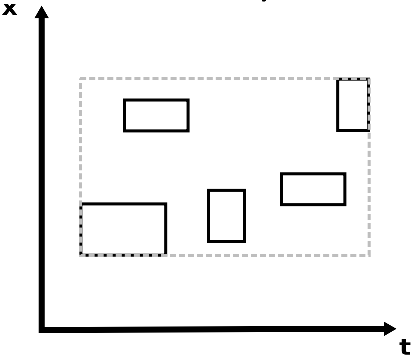


From a multiscale systems to many single-scale systems :

Let us consider a system of size L evolving over a time T.
Computation with space and time discretization Δx and Δt

Resolved spatial scales : $\Delta x < \xi < L$ and
Resolved temporal scales : $\Delta t < \tau < T$

Scale Map

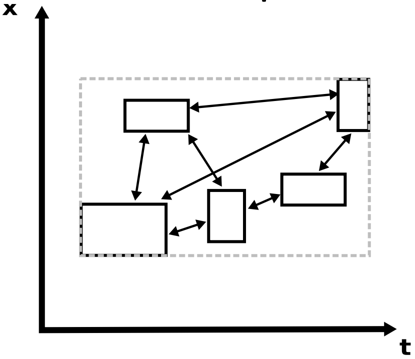


From a multiscale systems to many single-scale systems :

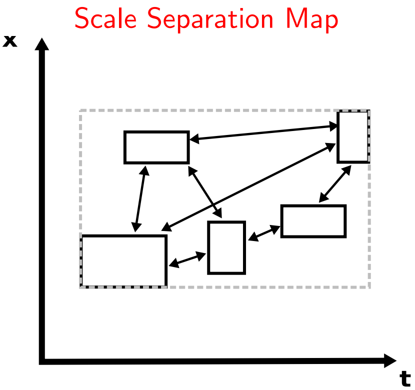
Let us consider a system of size L evolving over a time T.
Computation with space and time discretization Δx and Δt

Resolved spatial scales : $\Delta x < \xi < L$ and
Resolved temporal scales : $\Delta t < \tau < T$

Scale Map



From a multiscale systems to many single-scale systems :



- Submodels
- Smart Conduits

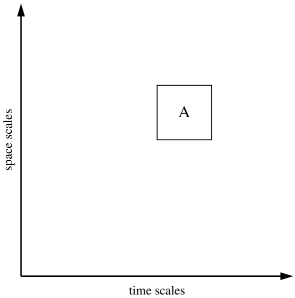
Complex Automata (CxA)

- A CxA is a set of coupled (single-scale) submodels
- **Lattice Boltzmann (LB)**, **cellular automata (CA)** models and **Finite Difference (FD)** schemes, and also **particle models**, ...
- They can be decribed with the same generic **execution loop**
- Submodels should not know about the rest of the system : they are autonomous components
- Only the **smart conduits** know about the properties of the submodels they connect.

A. G. Hoekstra, A. Caiazzo, E. Lorenz, J.-L. Falcone, and B. Chopard. *Complex Automata : multi-scale Modeling with coupled Cellular Automata*, in **Modelling Complex Systems by Cellular Automata**, chapter 3, Springer Verlag, 2010.

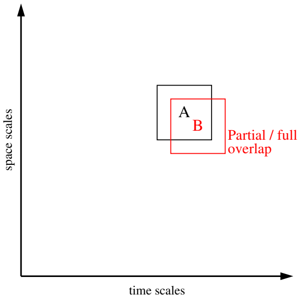
I. Relation between the scales

- The Scale Separation Map (SSM) specifies the relation between the sub-models in five regions :.



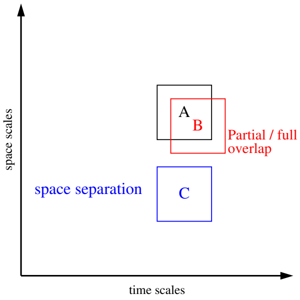
I. Relation between the scales

- The Scale Separation Map (SSM) specifies the relation between the sub-models in five regions :.



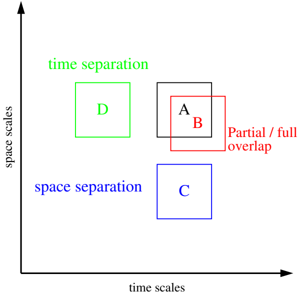
I. Relation between the scales

- The Scale Separation Map (SSM) specifies the relation between the sub-models in five regions :.



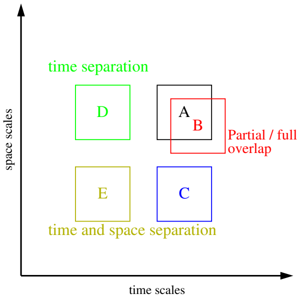
I. Relation between the scales

- The Scale Separation Map (SSM) specifies the relation between the sub-models in five regions :.



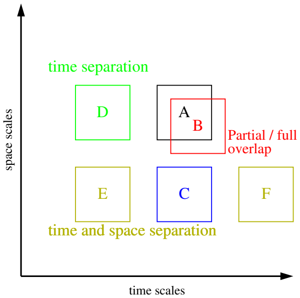
I. Relation between the scales

- The Scale Separation Map (SSM) specifies the relation between the sub-models in five regions :.



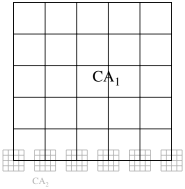
I. Relation between the scales

- The Scale Separation Map (SSM) specifies the relation between the sub-models in five regions :.
- There is more than the standard micro-macro relation and more than than the "bi-scale" modeling



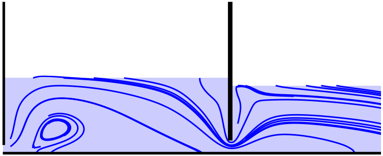
II. Relation between computational domains

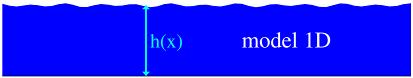
single-Domain (sD)



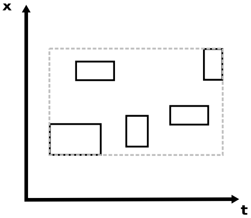
(Example : advection-diffusion, suspension flows)

multi-Domain (mD)





III Generic "Submodel Execution Loop"

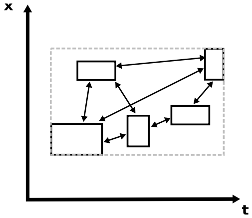


- f_{init} is for initialization
- S is for one iteration of the Solver
- B is to specify the Boundaries
- O_i is for Intermediate Observation
- O_f is for Final Observation

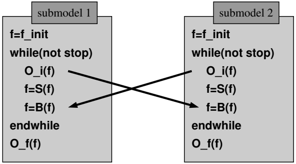
submodel

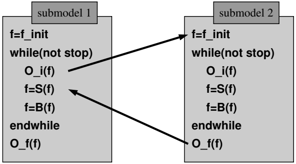
```
f=f_init
while(not stop)
    O_i(f)
    f=S(f)
    f=B(f)
endwhile
O_f(f)
```

IV. Coupling Templates



- One has several operators in the submodel execution loop
- O_i, O_f as origin
- f_{init}, B and S as possible destinations





Example : Coral growth

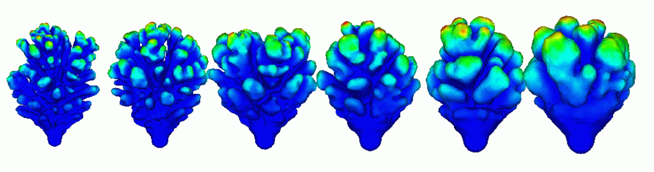
FLOW

```
D := D_init
f := f_init
while not EndCondition
  D := U(D)
  f := B(f)
  f := C(f)
  f := P(f)
  O_i[f]
end
O_f[f]
```

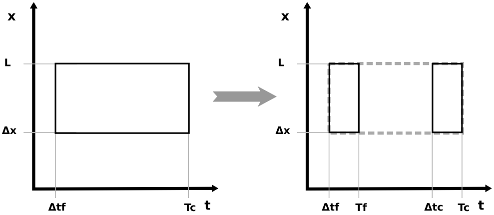
CORAL

```
D := D_init
f := f_init
while not EndCondition
  D := U(D)
  f := B(f)
  f := C(f)
  f := P(f)
  O_i[f]
end
O_f[f]
```

Coral grows due to nutrient brought by water flow



Coupling Speedup : Coral growth



Fluid is computed for

$$\frac{T_f}{\Delta t_f}\frac{T_c}{\Delta t_c} \quad \text{iterations instead of} \quad \frac{T_c}{\Delta t_f}$$

Speedup :

$$S = \frac{\Delta t_c}{T_f} >> 1$$

Classification of problems

- relation in the Scale Separation Map
- single-Domain (sD) or multi-Domain (mD) relation
- coupling templates

		TIME			
		overlap		separation	
SPACE	overlap	snow transport advection-diffusion ... O_i to S **single domain**	Fluid-Structure Grid transition ... O_i to B **multi domain**	Forest-Savannah-Fire ... O_i to f_init O_f to S **single domain**	Coral Growth ... O_i to f_init O_f to B **multi domain**
	separation	Algae-Water ... O_i to S **single domain**	Wave propagation ... O_i to B **multi domain**	Suspension O_i to f_init O_f to S **single domain**	Bio-Physics Tissue-Fluid O_i to f_init O_f to B **multi domain**

Relation between the scales separation and the coupling templates

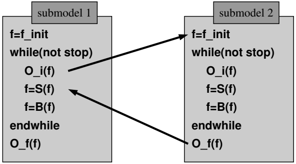
We consider two submodels, X and Y with **single-domain** (sD) relation

name	coupling	temporal scale relation
interact	$O_i^X \rightarrow S^Y$	overlap
call	$O_i^X \rightarrow f_{init}^Y$	X larger than Y
realease	$O_f^Y \rightarrow S^X$	Y smaller than X
dispatch	$O_f^X \rightarrow f_{init}^Y$	any

When the relation between computational domains is **multi-domain**, change $S \rightarrow B$

Thus, the relation in the SSM determines the workflow

CxA Execution Model



- Submodels are autonomous processes
- Asynchronous communication through the conduits :
  - Data is written to the conduit as soon as ready.
  - Submodels read the data they need from the conduits (wait if needed).
- Only local interactions are necessary : parallelization is possible and natural
- Propagation of the termination condition

Send-Receive through the conduits

Example of the Coral submodel :

```
  while not EndConditions
     DomainConduit.send(D)
     f := B(f)
     velocityMap := VelocityConduit.receive()
     f :=  S(f,velocityMap)
  end
DomainConduit.stop()
myStop()
```

The COAST software environment : MUSCLE

- Jade (Java Agent based lightweight middleware) as a platform to build the coupling software.
- Allows us to couple submodels (and legacy codes in C, Fortran).
- A "Jade coordinator" is used to setup the system then goes away,
- Low overhead.
- Predefined parametrized conduits
- Public release in Jan. 2009 [2]

2. http ://muscle.berlios.de

Can we do math with this approach ?

Mathematical formulation of couplings

CxA operators P and C can be used to express coupling strategies

- Time splitting
- Coarse graining
- Amplification
- ...

and estimate errors

Time splitting

Assume we have a sD problem with the following SEL

$$P_{\Delta t} C_{\Delta t} = P_{\Delta t} C_{\Delta t}^{(1)} C_{\Delta t}^{(2)}$$

Then if $C_{\Delta t}^{(1)}$ acts at a longer time scale than $C_{\Delta t}^{(2)}$ we may want to approximate

$$[P_{\Delta t} C_{\Delta t}]^M \approx P_{M\Delta t} C_{M\Delta t}^{(1)} [C_{\Delta t}^{(2)}]^M$$

Coarse graining

This strategy consists in expressing a sD problem as

$$[P_{\Delta x} C_{\Delta x}]^n \approx \Gamma^{-1} [P_{2\Delta x} C_{2\Delta x}]^{n/2} \Gamma$$

where Γ is a projection operator (implemented in the smart conduit)

Amplification

We consider a process acting at low intensity but for a long time, in a time periodic environment. For instance a growth process in a pulsatile flow.
We have two coupled (mD) processes which are iterated $n >> 1$ times

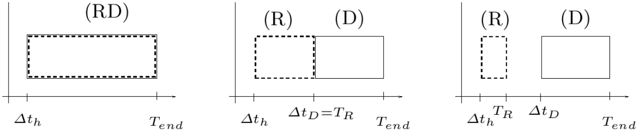
$$[P^{(1)}C^{(1)}]^n \qquad \text{and} \qquad [P^{(2)}C^{(2)}(k)]^n$$

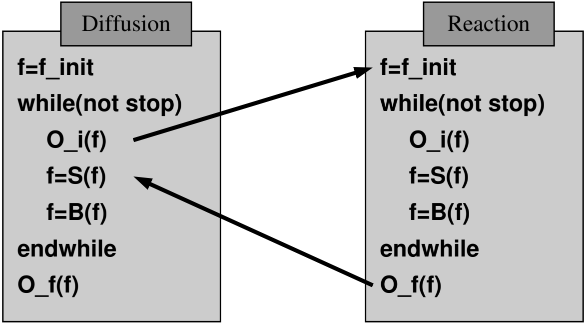
where k expresses the intensity of process $C^{(2)}$.
If the period of process $C^{(1)}$ is $m << n$, we can approximate the above evolution as

$$[P^{(1)}C^{(1)}]^m \qquad \text{and} \qquad [P^{(2)}C^{(2)}(k')]^m$$

with $k' = (n/m)k$, for a linear process.

Reaction-Diffusion with time splitting





$$\partial_t \rho = d \partial_{xx} \rho + \kappa(\rho_\lambda - \rho), \qquad (1)$$

We assume a fast reaction i.e. $\|\kappa\| \gg \|d\|$ (in some units).

The LB model in CxA language

$$f(t + \Delta t_R) = P[I + D(\tau_R) + R(\kappa)]f(t) \qquad (2)$$

$D(\tau_R)$ the diffusion collision operator at scale Δt_R, $R(\kappa)$ the reaction collision operator, I, the identity and P the propagation. It can be time-split as

$$f(t + \Delta t_D) = P[I + D(\tau_D)][I + R(\kappa)]^{\Delta t_D/\Delta t_R} f(t) \qquad (3)$$

The error E of this time splitting can be computed analytically

A. Caiazzo, J-L. Falcone, B. Chopard and A. G. Hoekstra, *Asymptotic analysis of Complex Automata models for reaction-diffusion systems*, Applied Numerical Mathematics 59 pp. 2023–2034 (2009)

Temporal scales

The time scales are such that

$$\Delta t_R < \tau_R = 1/\kappa \ll \Delta t_D < \tau_D = 1/(\lambda^2 d)$$

thus, the actual scale separation is

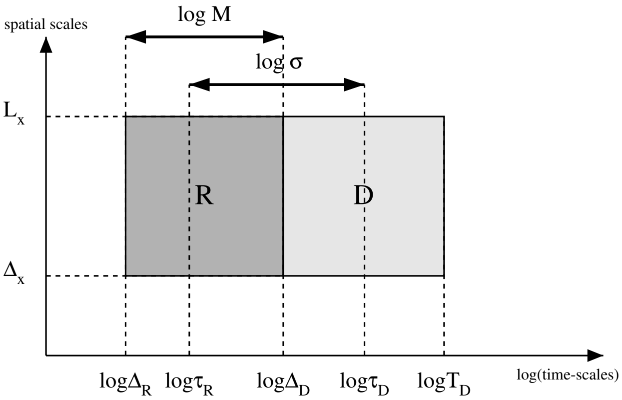
$$\sigma = \frac{\tau_D}{\tau_R} = \frac{\kappa}{\lambda^2 d}$$

whereas, the enforced scale separation is

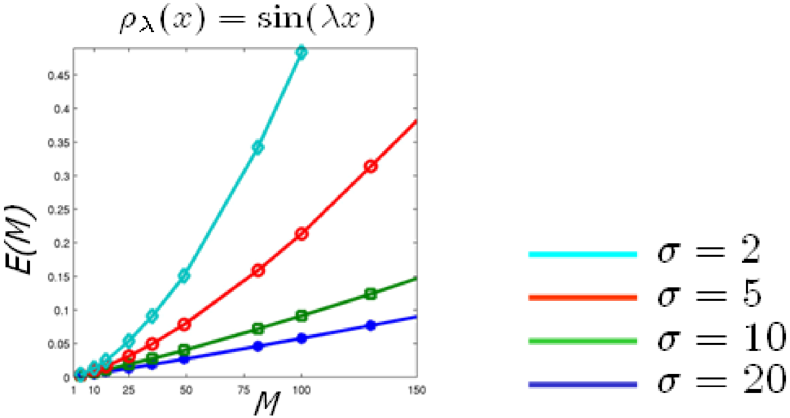
$$M = \frac{\Delta t_D}{\Delta t_R}$$

Scales separation

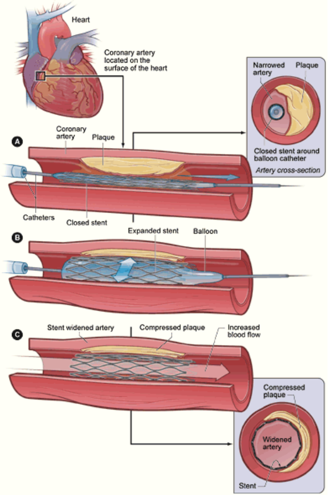
$$\log M = \log \Delta t_D - \log \Delta t_R \qquad \log \sigma = \log \tau_D - \log \tau_r$$



Time-splitting error versus scale separation

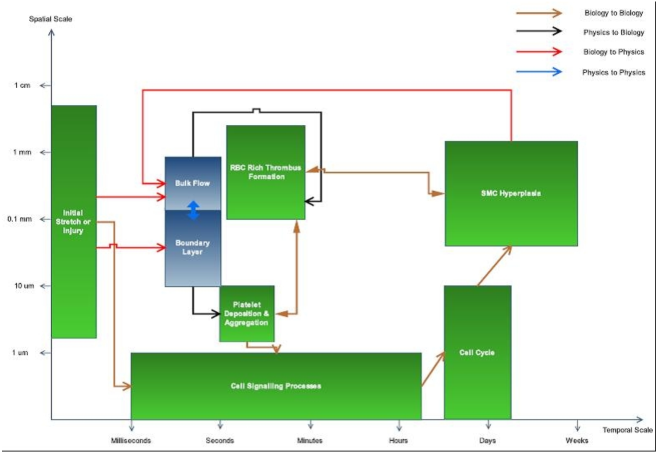


Biomedical application : in-stent restenosis



- Coronary heart disease (CHD) remains the most common cause of death in the UK, being responsible for approximately 105,000 deaths in 2004 (BHF Stats, 2006).
- In 2005, 94% of 70,142 UK procedures involved the deployment of a stent (BCIS Stats, 2006).

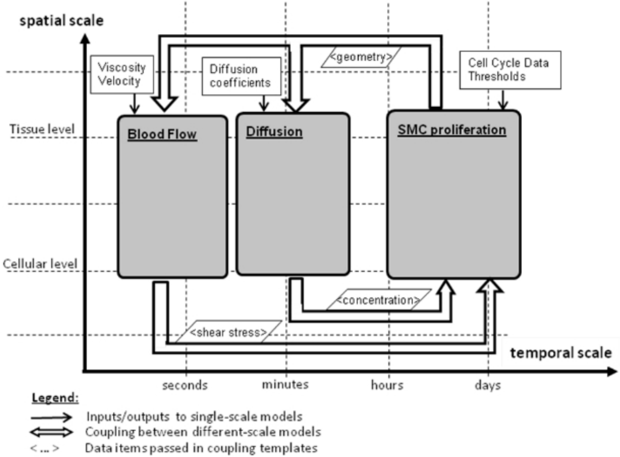
Restenosis : the full Scale Separation Map



Restenosis : Scale Separation Map

- A 3-submodel simplification (time separation is achieved)

D Evans, PV Lawford, J Gunn, D Walker, DR Hose, RH Smallwood, B Chopard, M Krafczyk, J Bernsdorf, A Hoekstra. *The Application of Multi-Scale Modelling to the Process of Development and Prevention of Stenosis in a Stented Coronary Artery.* **Phil. Trans. R. Soc. A** 366, pp. 3343–3360, 2008



Wrapping things together in the software environement

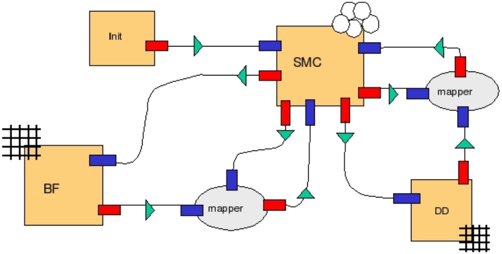
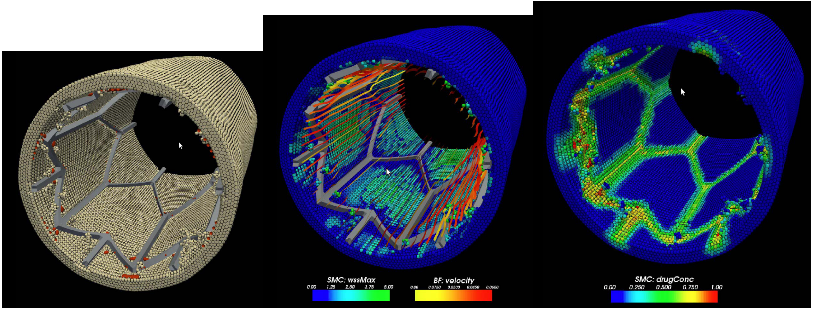


Figure 16: Example of Connection Scheme for a CxA coupling BF, SMC and DD (see technical deliverable 3.2). For each single scale model, it is indicated whether it is based on a lattice or on agent. Mappers are used to map different inputs onto the time dependent domain of cells.

Runs on a distributed infrastructure (MAPPER project) more than one output.

3D model



SMC: wssMax
0.00 1.25 2.50 3.75 5.00
BF: velocity
0.00 0.0150 0.0300 0.0450 0.0600
SMC: drugConc
0.00 0.250 0.500 0.750 1.00

MML : a Multiscale Modeling Language

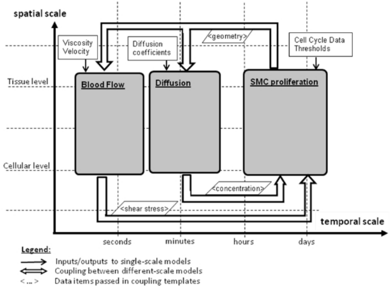
- the SSM turned out to be very powerful to design applications
- Formalize the CxA ideas into a language : high level representation of a complex multiscale application
- Allows scientists with different backgrounds and geographical locations to better co-develop a multiscale application
- Provide blueprints of a complex multiscale application that can be further augmented by other groups
- Standard for publication

J-L Falcone, B. Chopard and A. Hoekstra, MML : towards a Multiscale Modeling Language,

Procedia Computer Science 1 :11, 819-826, 2010

Main ingredients

- Sub-models
- Spatial and temporal scales
- Computational domain relation
- Coupling templates
- Conduits



We want to represent these features on a **descriptive language**

xMML

- XML-like language
- Easy grammar for the user
- Full description language
- From application description to “glue-code” production and scheduling

xMML example

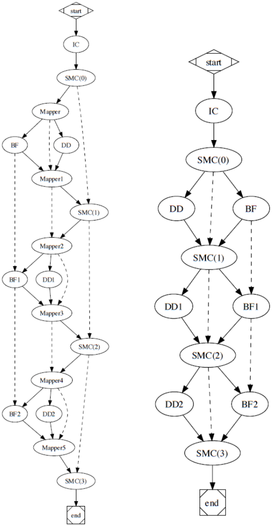
```
<model id="suspensionFlow">
  <description>
    Flow with a suspension of particles. The conentration
    of particles affect locally the flow viscosity and the
    particles are advected by the flow.
  </description>
  <submodel id="flow">
    <spacescale dimension="2" dx="1 mm" lx="10 cm" ly="30 cm" />
    <spacescale dt="1 ms" t="1 min" />
    <ports>
      <in id="concentration" operator="C" dt="1 ms" dx="1 mm" />
      <out id="velocity" operator="Oi" dt="10 ms" dx="1 mm" />
    </ports>
  </submodel>
```

xMML example continued

```
  <submodel id="advectionDiffusion">
    <spacescale dimension="2" dx="1 mm" lx="10 cm" ly="30 cm" />
    <spacescale dt="10 ms" t="1 min" />
    <ports>
      <in id="velocity" operator="C" dt="10 ms" dx="1 mm" />
      <out id="concentration" operator="Oi" dt="10 ms" dx="1 mm" />
    </ports>
  </submodel>

  <coupling from="flow.velocity" to="advectionDiffusion.velocity" />
  <coupling from="advectionDiffusion.concentration" to="flow.concentration">
    <filter kind="timeInterpolation" />
  </coupling>
</model>
```

Execution graph



(a) Complete data flow (b) Simplified data flow

Multiscale APPlications on European e-infRastructures

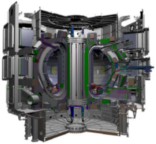
From applications $\rightarrow$ MML $\rightarrow$ computing infrastructure

- Running tightly coupled **Distributed Multiscale Applications** using several supercomputing platforms
- Deploy middleware implementing the CxA-MML-MUSCLE approach on the e-Infrastructure (EGI, PRACE, DEISA)

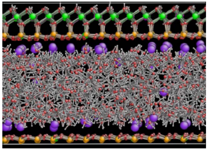
http ://www.mapper-project.eu

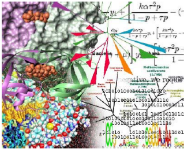
Application portfolio


virtual physiological human


fusion


hydrology

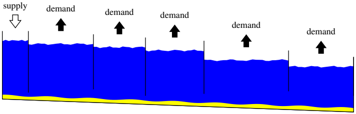

nano material science

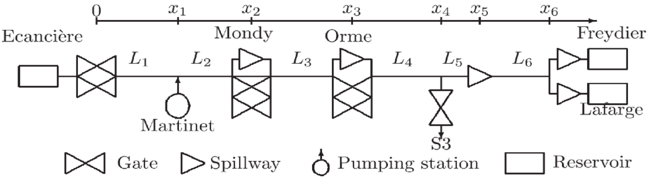

computational biology

- Participants : UvA NL, UCL UK, UU UK, PSNC PL, CYFRONET PL, LMU DE, UNIGE CH, CHALMERS SE, MPG DE

Simulation of irrigation canals

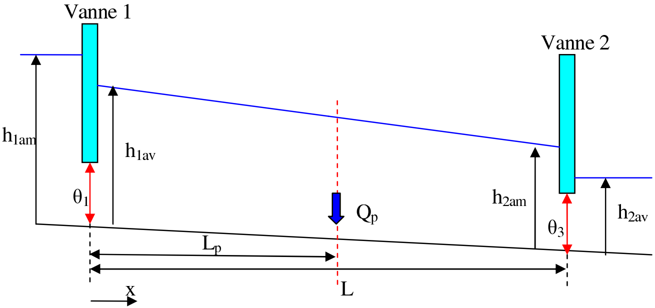
Develop a simulation tools for the optimal management of irrigation canals





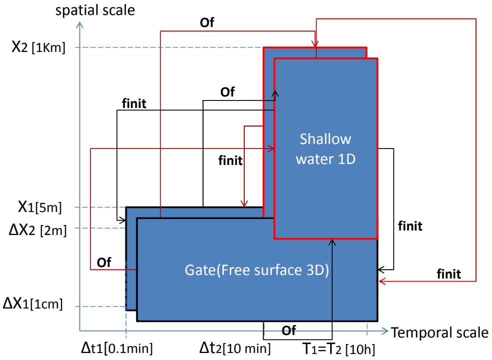
L. Lefèvre, E. Mendes et al. (ESISAR Valence, INP-Grenoble)

Control of canal operation



- Maintain the discharge Q at the downstream gate, for all lateral demands Q_p. Constraint are :
  - Water height : $h_{min} \leq h \leq h_{max}$
  - Gate opening : $\theta_{min} \leq \theta \leq \theta_{max}$
  - Speed of the command : $\dot{\theta}_{min} \leq \dot{\theta} \leq \dot{\theta}_{max}$

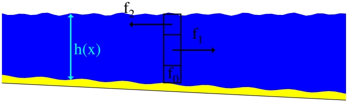
Scale Separation Map and submodels

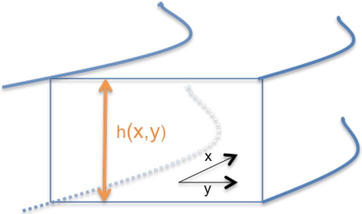


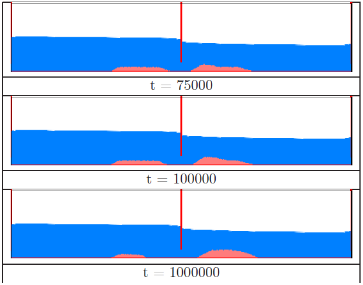
Multiple instance of tightly coupled sub-models

- Shallow water equation : 1D and 2D
- Detailed 3D free-surface model
- Transport and erosion of sediments
- Gates : $Q = \alpha O \sqrt{h_{up} - h_{down}}$
- spill-way,...

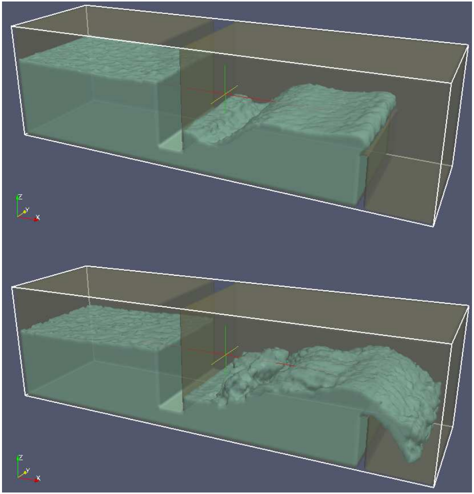
Submodels



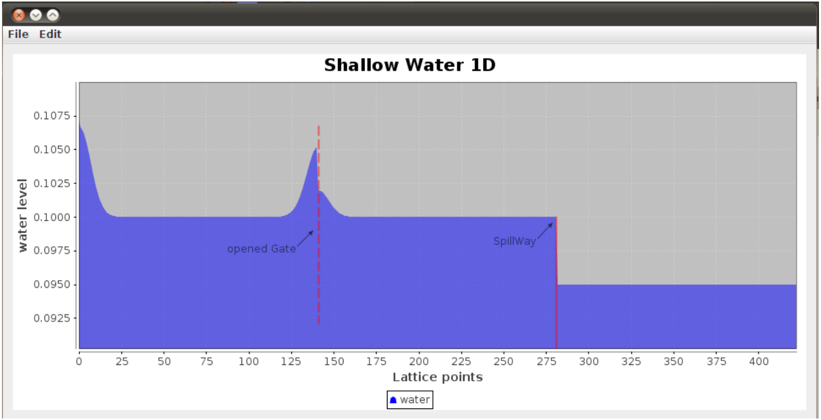




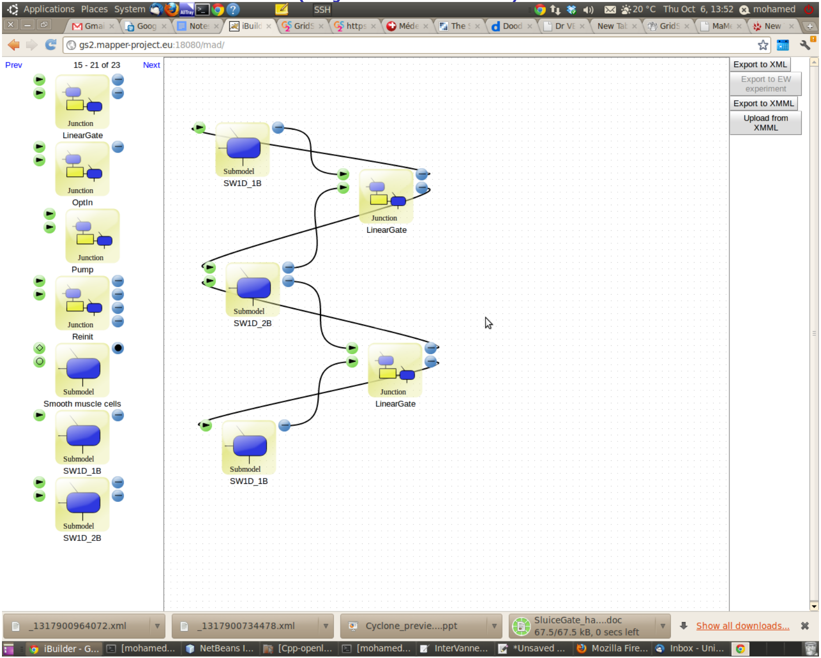
3D, free surface



Coupling 1D SW models (Mohamed Ben Belgacem)



MAD tool (Cyfronet, PL)



Submodel (kernel) and interface to MUSCLE

```
SW1D can1;= new SW1D(L, dx, dt, width, 0.03d);

for (int j = 0; j < nbriteration; j++) {
    can1.collision();
    can1.propagation();

  //Observation: Collects data to send to the Gate
    info = new HashMap<String, Double>();
    info.put("f1", can1.getf1(nx));
    info.put("h", can1.getH()[nx]);
    f_out.send(info);//send the Data to the Gate

  // Boundary: receive the data from the Gate
    double fin = f_in.receive();
    can1.setf2(nx, fin);// update the distribution function

    can1.bounceBack();  // boundary at the left end
      }
```

MUSCLE Coupling script

```
# declare kernels
cxa.add_kernel('SW1D1', 'com.unige.irigcan.kernel.d1.SW1D_1B_kernel')
cxa.add_kernel('SW1D2', 'com.unige.irigcan.kernel.d1.SW1D_2B_kernel')
cxa.add_kernel('SW1D3', 'com.unige.irigcan.kernel.d1.SW1D_1B_kernel')
cxa.add_kernel('Gate', 'com.unige.irigcan.junction.Gate_kernel')
cxa.add_kernel('Spill', 'com.unige.irigcan.junction.Spill_kernel')
```

MUSCLE Coupling script (continued)

```
# configure connection scheme
cs = cxa.cs

cs.attach('SW1D1' => 'Gate') { tie('f_out', 'f1_in')}
cs.attach('SW1D2' => 'Gate') { tie('f_out', 'f2_in')}
cs.attach('Gate' => 'SW1D1') { tie('f1_out', 'f_in')}
cs.attach('Gate' => 'SW1D2') { tie('f2_out', 'f_in')}
#
cs.attach('SW1D2' => 'Spill') { tie('f_out_X', 'f1_in')}
cs.attach('SW1D3' => 'Spill') { tie('f_out', 'f2_in')}
#
cs.attach('Spill' => 'SW1D2') { tie('f1_out', 'f_in_X')}
cs.attach('Spill' => 'SW1D3') { tie('f2_out', 'f_in')}
```

See simulation...

Thank you for your attention