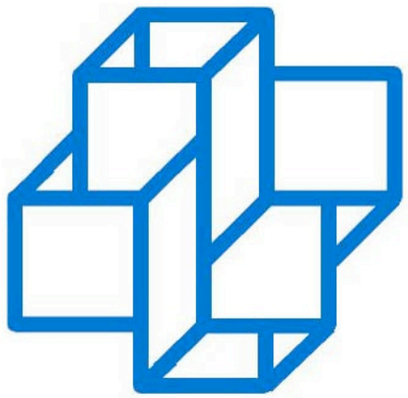
On the reengineering of the SPiNMe environment for supporting the Multiscale Hybrid-Mixed Methods

Antônio Tadeu A. Gomes
Diego Paredes
Frederico Teixeira
Frédéric Valentin

atagomes@lncc.br

http://www.lncc.br/sinapad/SPiNMe
http://www.facebook.com/sinapad



Laboratório Nacional de Computação Científica

Laboratório Nacional de Computação Científica

$$-\epsilon\Delta u + \alpha \cdot \nabla u + \sigma u = f$$

Approximate solution

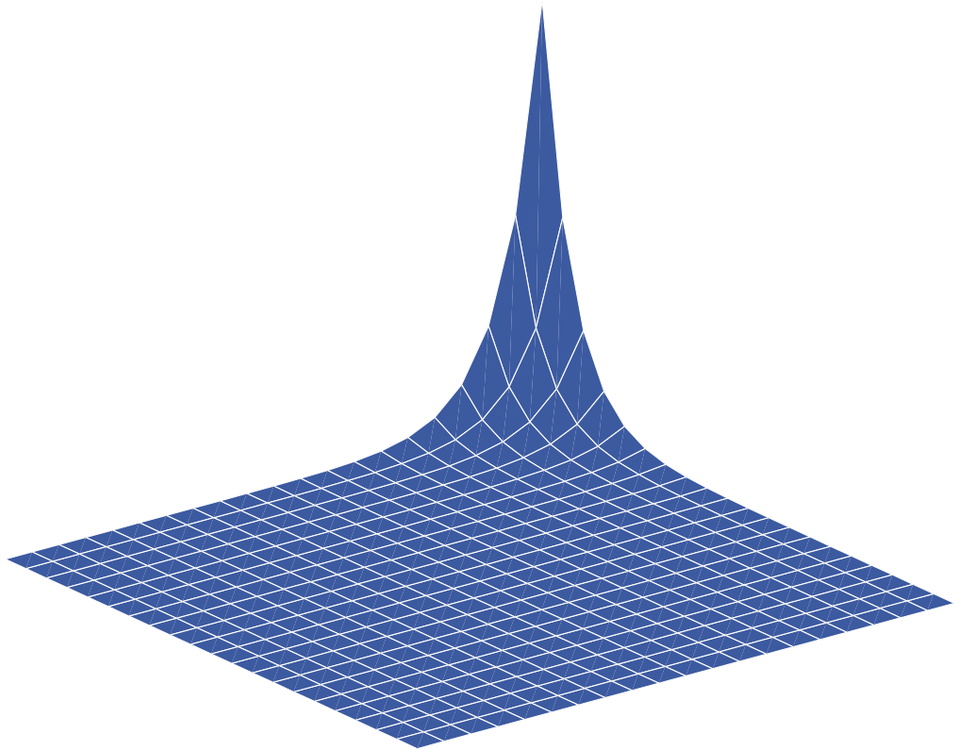
$$\mathbf{A}\mathbf{u} = \mathbf{f}$$

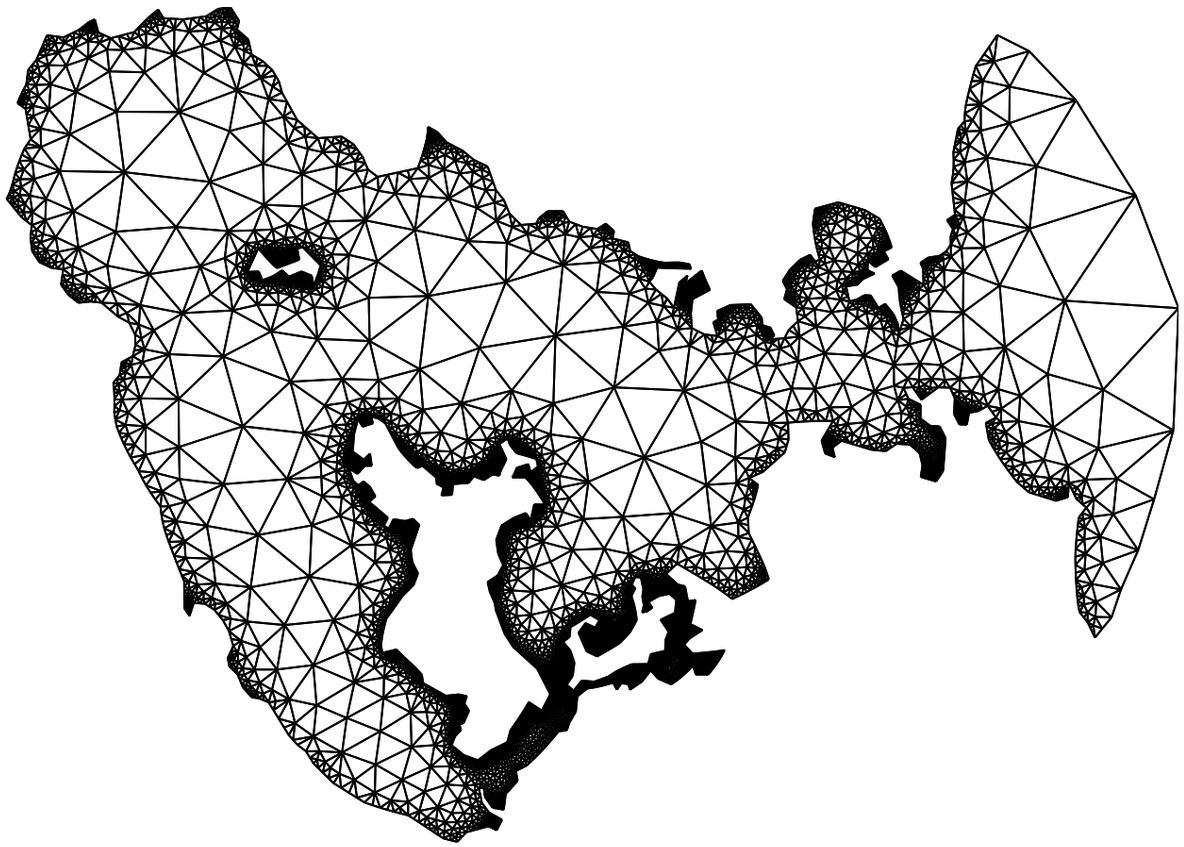
- Accuracy
- Performance
- Flexibility
- Mathematical theory

Finite Element Methods

Laboratório Nacional de Computação Científica

- Geometric flexibility
- Amenability to adaptations

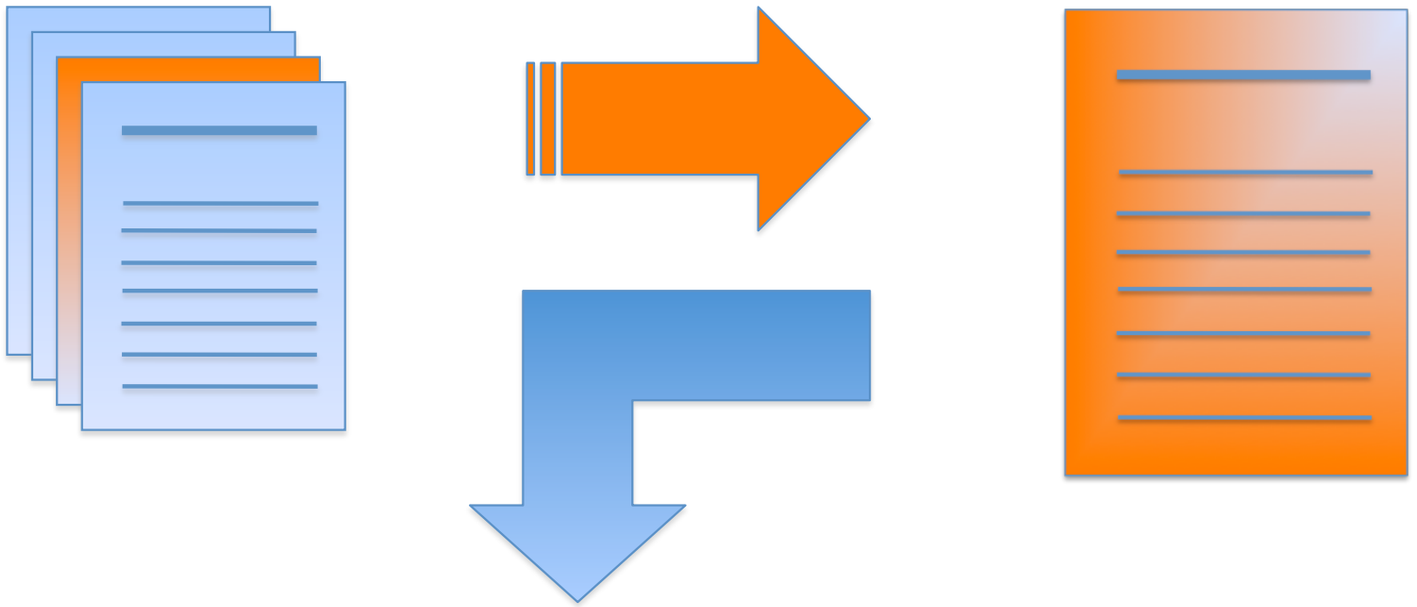




New methods...

Laboratório Nacional de Computação Científica

Digital libraries of "non-executable" papers



Gotcha!!!

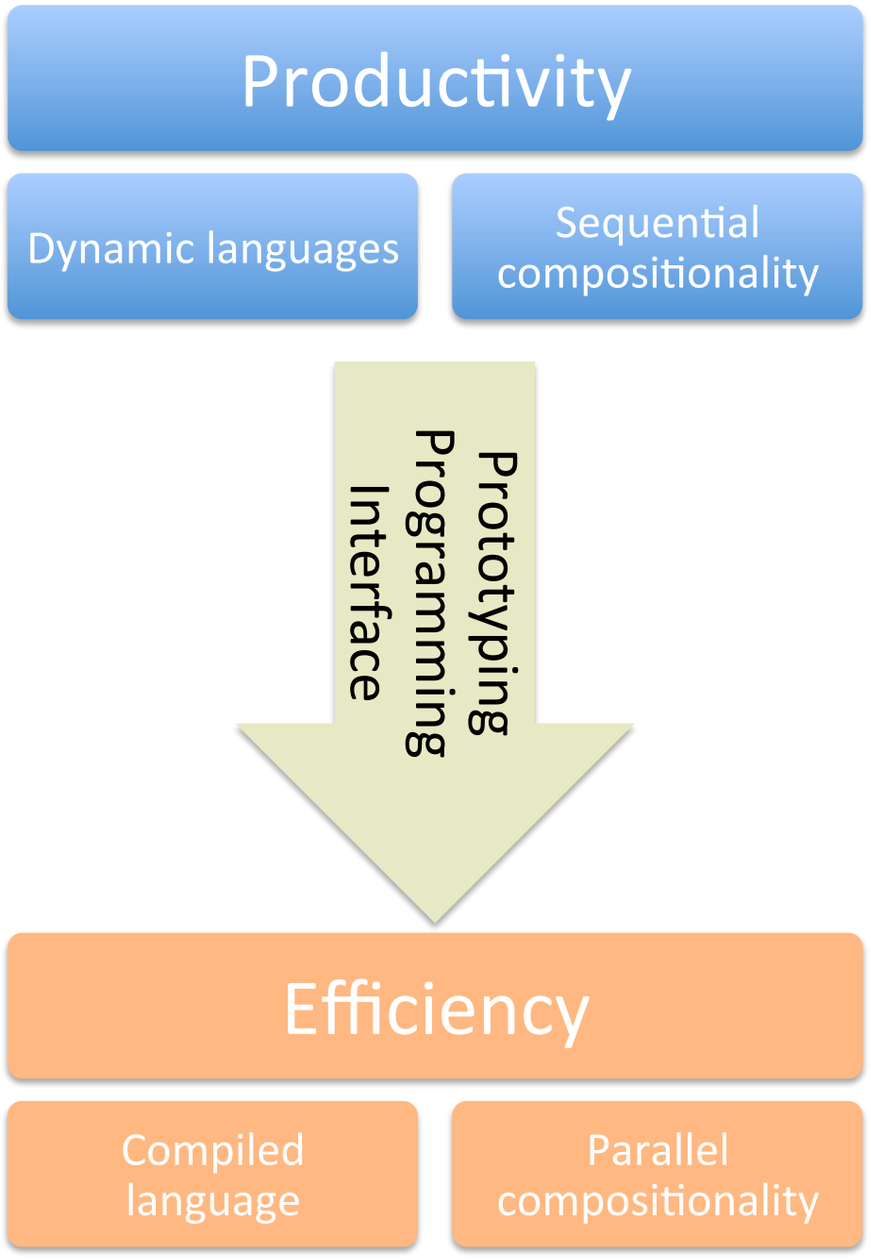
$$a(u,v) + \sum_K \frac{1}{2} \int_{\partial K} \gamma h_{\partial K} [\nabla u \cdot n][\nabla v \cdot n] \, ds = (f, v)$$

- Do I need **brand new code**?
- Will my implementation be **useful in the future**?
- Can I **deploy at runtime**?
- How can I **compare** against other method implementations?

Scientific software: technical complexity

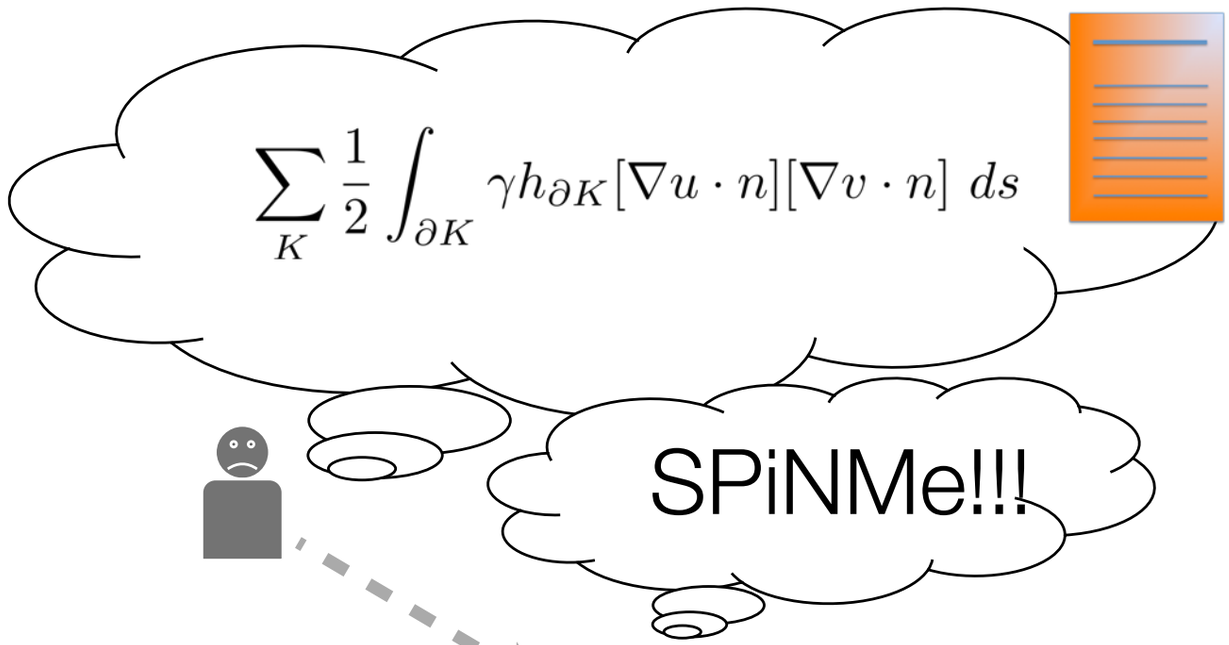
Laboratório Nacional de Computação Científica

- Long time-to-market
- "Crosscutting error" proneness
  - Software architecture!!
- Mitigation strategy
  - Stratification of productivity and efficiency
  - Productivity through rapid prototyping



SPiNMe

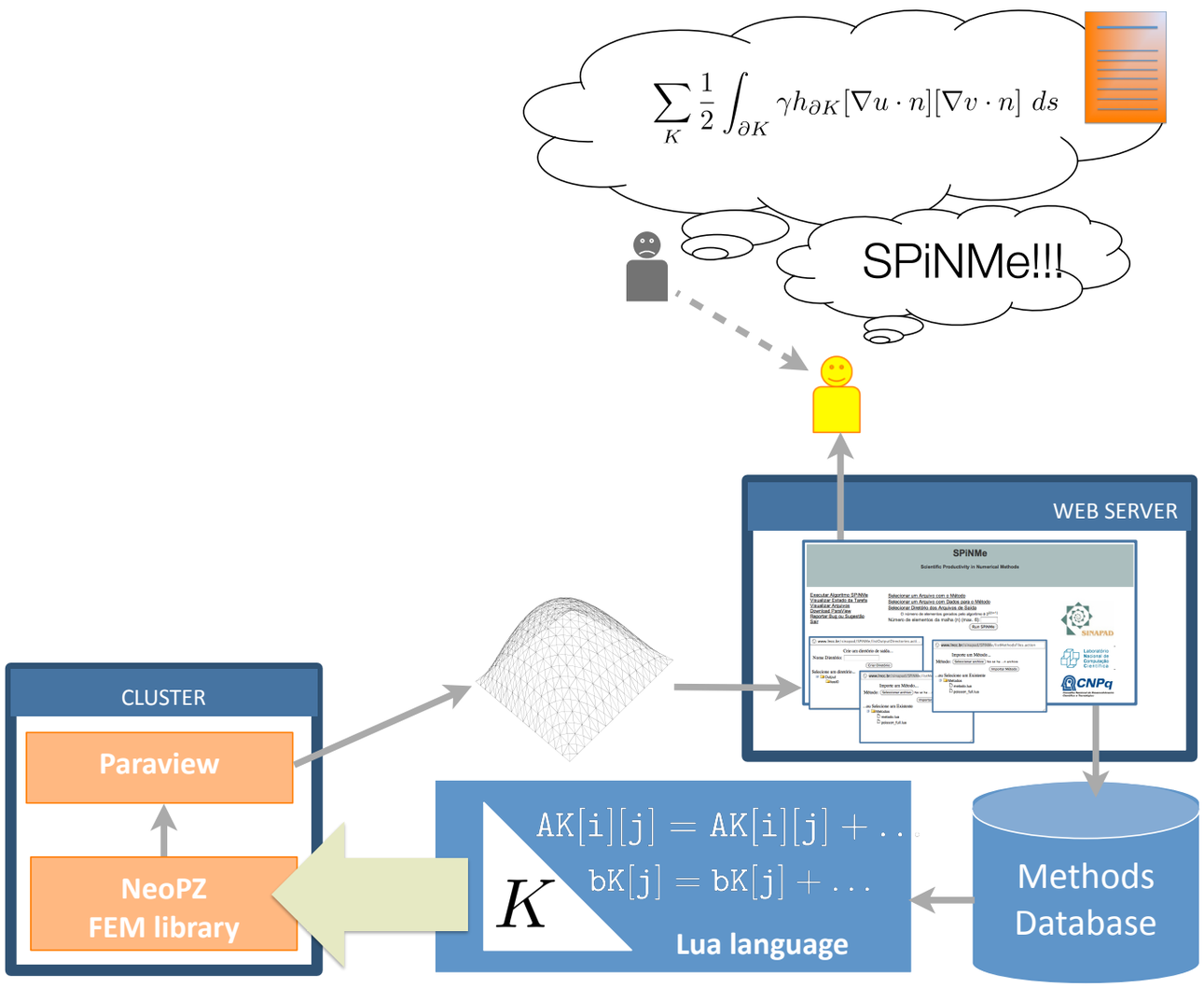
- Rapid prototyping of FE methods
  - Local programming
  - Good performance
- Perpetuation/reproduction of methods
  - Concept of "*executable paper*"



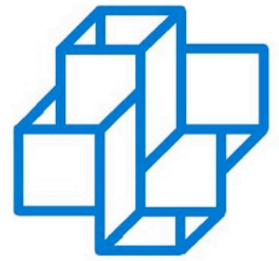
What has been done so far?

Laboratório Nacional de Computação Científica

- Stationary
- Scalar
- Transient
- Vectorial
- Meshes: gmesh, tetgen
- GPU awareness



Code snippets



Laboratório Nacional de Computação Científica

- General problem: convection-reaction-diffusion...

$$\begin{cases} u_t - \nabla \cdot (\epsilon \nabla u) + \alpha \cdot \nabla u + \sigma u = f, & \text{in } \Omega, \\ u = g, & \text{on } \partial\Omega_D, \\ (\epsilon \nabla u) \cdot \eta = h, & \text{on } \partial\Omega_N, \\ u(x, y, 0) = u_0(x, y), & \text{in } \Omega. \end{cases}$$

$$\Omega = [0.1]^d \subset \mathcal{R}^d, d = 2, 3 \text{ and } \partial\Omega = \partial\Omega_D \cap \partial\Omega_N$$

- ...whose variational form depends on the method...

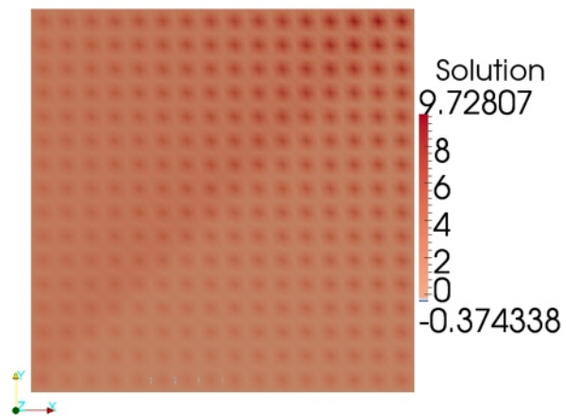
Code snippets

- Stationary problem - Continuous Galerkin Methods

Find $u \in H^1(\Omega), u = g$ on $\partial\Omega$, such that:

$$(\epsilon\nabla u, \nabla v) + (\alpha \cdot \nabla u, v) + (\sigma u, v) = (f, v), \quad \forall v \in H_0^1(\Omega).$$

```
1  StatARD = { Dimension = 2,
2    FreeNodes = {
3      [1] = {
4        Diffusion = { Epsilon = { {0.0001, 0.0},
5                                  {0.0, 0.0001} } },
6        Reaction = { Sigma = 0.01 },
7        Advection = { Alpha = { 0.5, 0.5 }, },
8        SourceSink = { F = 1.0 },
9      }
10   }, -- End of FreeNodes
11   ConstrainedNodes = {
12     [14] = { PhyEntity = 1,
13         Dirichlet = { G = 1.0, }, },
14     [15] = { PhyEntity = 1,
15         Dirichlet = { G = 0.0, }, },
16   }, -- End of ConstrainedNodes
17 }
```



Without stabilization term

Code snippets

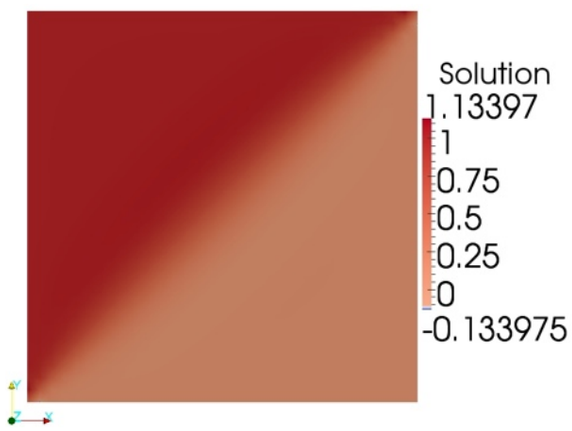
Laboratório Nacional de Computação Científica

- Implementation of a new contribution - Stabilized Methods

Find $u \in H^1(\Omega), u = g$ on $\partial\Omega$, such that:

$$(\epsilon\nabla u, \nabla v) + (\alpha \cdot \nabla u, v) + (\sigma u, v) + B(u, v) = (f, v) + F(v) \quad \forall v \in H_0^1(\Omega).$$

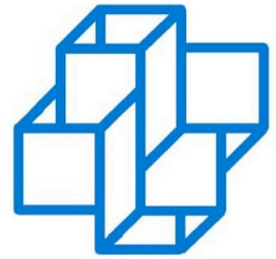
```
1  unusual_physics = { Dimension = 2,
2    FreeNodes = {
3      [1] = {
4        ... -- Same as before
5        unusual_term = {}, -- New term!!!
6      }
7    }, -- End of FreeNodes
8    ConstrainedNodes = {...}, -- End of ConstrainedNodes
9  }
10 unusual_physics.FreeNodes[1].unusual_term = { -- New term specs!
11    Nu = unusual_physics.FreeNodes[1].Diffusion.Epsilon[1][1],
12   Sigma = unusual_physics.FreeNodes[1].Reaction.Sigma,
13   A = unusual_physics.FreeNodes[1].Advection.Alpha,
14   H = 1/256, -- mesh dependent!!!!
15   f = unusual_physics.FreeNodes[1].SourceSink.F,
16 }
```



With stabilization term

- Where is “B” and “F”???

Code snippets



Laboratório Nacional de Computação Científica

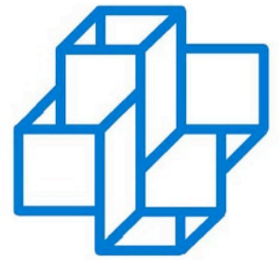
$$a(u_h, v_h) + B(u_h, v_h) = (f, v_h) + F(v_h)$$

```
local function tau_K() ... -- aux function omitted for brevity

function unusual_term.asmFreeNodes(data, weight, ek, ef)
 local x, phi, dphi = data:getX(), data:getPhi(), data:getDPhix()
 local r = phi.rows

 for i=0,r-1 do
  ef[i][0] = ef[i][0] + weight * f * phi[i][0] -
    N * weight * f * tau_K(x[0],x[1]) *
    (Sigma * phi[i][0] - A[0] * dphi[0][i] - A[1] * dphi[1][i])
  for j=0,r-1 do
   ek[i][j] = ek[i][j] - N * weight *
    (-Sigma * Sigma * phi[j][0] * phi[i][0] * tau_K(x[0],x[1]) +
      Sigma * (A[0] * dphi[0][j] + A[1] * dphi[1][j] ) * tau_K(x[0],x[1]) * phi[i][0] -
      Sigma * phi[j][0] * tau_K(x[0],x[1]) * (A[0] * dphi[0][i] + A[1] * dphi[1][i]) -
      (A[0] * dphi[0][j] + A[1] * dphi[1][j]) *
      tau_K(x[0],x[1]) * (A[0] * dphi[0][i] + A[1] * dphi[1][i]) )
  end
 end
end
```

```
unusual_physics = { Dimension = 2,
  FreeNodes = {
    [1] = {
      ... -- Same as before
      unusual_term = {}, -- New term!!!
    }
  }, -- End of FreeNodes
  ConstrainedNodes = {...}, -- End of ConstrainedNodes
}
unusual_physics.FreeNodes[1].unusual_term = { -- New term specs!
   Nu = unusual_physics.FreeNodes[1].Diffusion.Epsilon[1][1],
  Sigma = unusual_physics.FreeNodes[1].Reaction.Sigma,
  A = unusual_physics.FreeNodes[1].Advection.Alpha,
  H = 1/256, -- mesh dependent!!!!
  f = unusual_physics.FreeNodes[1].SourceSink.F,
}
```

Franca, L., Valentin, F.: *On an Improved Unusual Stabilized Finite Element Method for the Advective-Reactive-Diffusive Equation.* Computer Methods in Applied Mechanics and Engineering, v. 190 p. 1785-1800. (2000)

Code snippets



Laboratório Nacional de Computação Científica

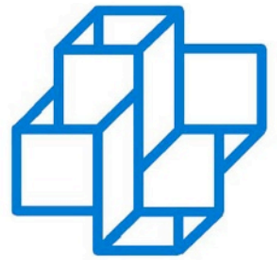
- Transient problem, first order in time - Continuous Galerkin Methods

$$\text{Find } u : [0,T] \longrightarrow H_0^1(\Omega), u(x,t) = g(x), \text{ on } \partial\Omega, \text{ such that:}$$

$$\begin{cases} \left(u_t, v\right) + \left(\epsilon \nabla u, \nabla v\right) + \left(\alpha \cdot \nabla u, v\right) + \left(\sigma u, v\right) = \left(f, v\right), \quad \forall v \in H_0^1(\Omega). \\ \\ u(x,y,0) = u_0(x,y), \quad \text{in } \Omega. \end{cases}$$

```
TranADR = { Dimension = 2,
  FreeNodes = {
    [1] = {
      Transient = { Coeff = 1.0 },
      Advection = { Alpha = {1.0, 1.0} },
      Diffusion = { Epsilon = { {1.0, 0.0}, {0.0, 1.0} } },
      Reaction = { Sigma = 1.0 },
      SourceSink = { F = function(x,y,z,t)
        return {2*math.pi^2*math.exp(-t)*math.sin(math.pi*x)*math.sin(math.pi*y)
          + math.pi*math.exp(-t)*(math.cos(math.pi*x)*math.sin(math.pi*y)
          + math.sin(math.pi*x)*math.cos(math.pi*y))} end, }, },
        }, -- End of FreeNodes
        ConstrainedNodes = {
                [1] = {
        PhyEntity = 1,
        Dirichlet = { G = function(x,y,z,t) return 0.0 end, }, },
        }, -- End of ConstrainedNodes
}
```

Code snippets



Laboratório Nacional de Computação Científica

- Where are the other simulation parameters? (Interpolation order, solver, matrix data structure, initial condition, exact solution...)

- Going on with the example in the previous slide...

```
1  Precision = 1
2  InterpolationOrder = 4
3  StructuralMatrix = NONSYM_FULL
4  Solver = DIRECT_LU
5  TimeStep = 1/80
6  TimeIterations = 80
7  ThetaMethod = 0.5
8  ExactSolution = function(x,y,z,t)
9    return {math.exp(-t)*math.sin(math.pi*x)*math.sin(math.pi*y)} end
10 InitialCondition = function(x,y,z,t)
11   return {math.sin(math.pi*x)*math.sin(math.pi*y)} end
```

Code snippets

Laboratório Nacional de Computação Científica

- Transient problem, second order in time - Continuous Galerkin Methods

$$\text{Find } u : [0,T] \longrightarrow H_0^1(\Omega), u(x,y,t) = g(x,y), \text{ on } \partial\Omega, \text{ such that:}$$

$$\begin{cases} \left(u_{tt}, v\right) + \left(\epsilon \nabla u, \nabla v\right) = \left(f, v\right), \quad \forall v \in H_0^1(\Omega). \\ u(x,y,0) = u_0(x,y), \quad \text{in } \Omega. \\ u_t(x,y,0) = u_1(x,y), \quad \text{in } \Omega. \end{cases}$$

```
SecTranD_Dir = { Dimension = 2,
  FreeNodes = {
    [1] = {
      TransientSecond = { Coeff = 1.0 },
      Diffusion = { Epsilon = { {1.0, 0.0}, {0.0, 1.0} } },
      SourceSink = { F = function(x,y,z,t) return
        2*t*math.pi^2*math.sin(math.pi*x)*math.sin(math.pi*y)
        end, }, }, }, -- End of FreeNodes
  ConstrainedNodes = {
    [1] = {
      PhyEntity = 1,
      Dirichlet = { G = function(x,y,z,t) return 0.0 end, }, }
  }, -- End of ConstrainedNodes
}
```

```
1  Precision = 1
2  InterpolationOrder = 3
3  IntegrationRuleOrder = 8
4  StructuralMatrix = BAND
5  Solver = LU
6  TimeStep = 1/40
7  TimeIterations = 400
8  ExactSolution = function(x,y,z,t)
9    return t*math.sin(math.pi*x)*math.sin(math.pi*y) end
10 ExactSolutionPrime = function(x,y,z,t)
11   return math.sin(math.pi*x)*math.sin(math.pi*y) end
12 InitialCondition =
13   function(x,y,z,t) return 0.0 end
14 InitialConditionPrime =
15   function(x,y,z,t) return 0.0 end
```

Code snippets

Laboratório Nacional de Computação Científica

- Stationary problem - Mixed Methods

Find $\mathbf{u} \in H(div, \Omega)$ and $p \in L^2(\Omega)$ such that

$$\begin{cases} \left(\epsilon^{-1}\mathbf{u}, \mathbf{v}\right) + \left(p, \nabla \cdot \mathbf{v}\right) = 0, \forall \mathbf{v} \in H(div, \Omega). \\ \left(q, \nabla \cdot \mathbf{u}\right) = \left(f, q\right), \forall q \in L^2(\Omega). \end{cases}$$

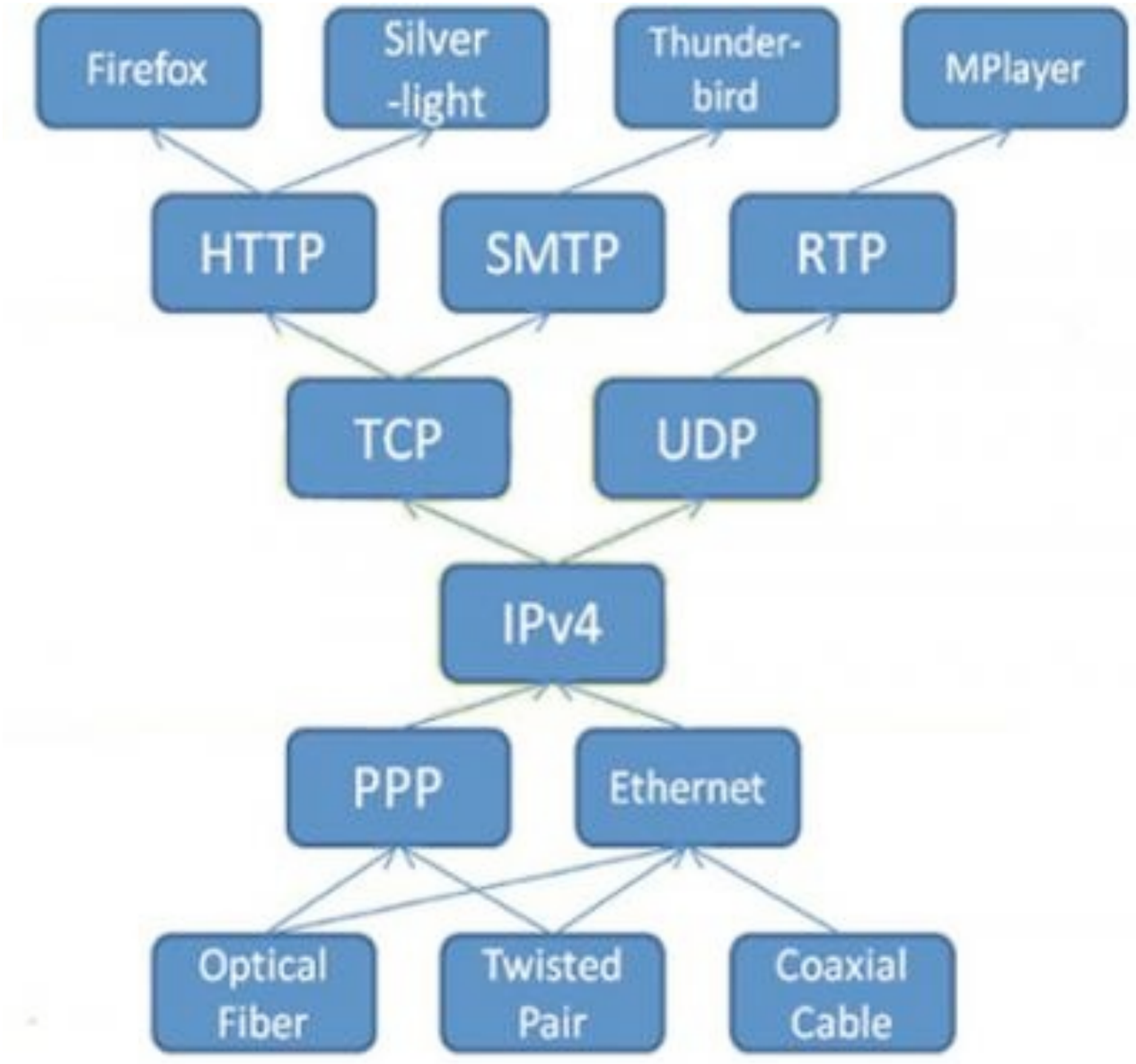
```
Mix_Dir = {
  Dimension = 2,
  FreeNodes = {
    [1] = {
      MixedDiffusion = { Epsilon = { {1.0, 0.0}, {0.0, 1.0} } },
      SourceSink = { F = function(x,y,z,t)
        return 2 * math.pi^2 * math.sin(math.pi*x) *
                   math.sin(math.pi*y) end, },
    }, -- End of [1]
  }, -- End of FreeNodes
  ConstrainedNodes = {
    [15] = { PhyEntity = 1,
    Dirichlet = { G = function(x,y,z,t)
                          return math.sin(math.pi*x) *
                                 math.sin(math.pi*y) end, }, },
  }, -- End of ConstrainedNodes
}
```

```
InterpolationOrder = 1
StructuralMatrix = NONSYM_FULL
Solver = LU
ExactPrimalSolution = function(x,y,z,t)
   return math.sin(math.pi*x)*math.sin(math.pi*y) end
-- dual = - Epsilon * grad(primal)
ExactDualSolution = function(x,y,z,t)
   return { -math.pi*math.cos(math.pi*x)*math.sin(math.pi*y),
            -math.pi*math.sin(math.pi*x)*math.cos(math.pi*y) } end
```

Laboratório Nacional de Computação Científica

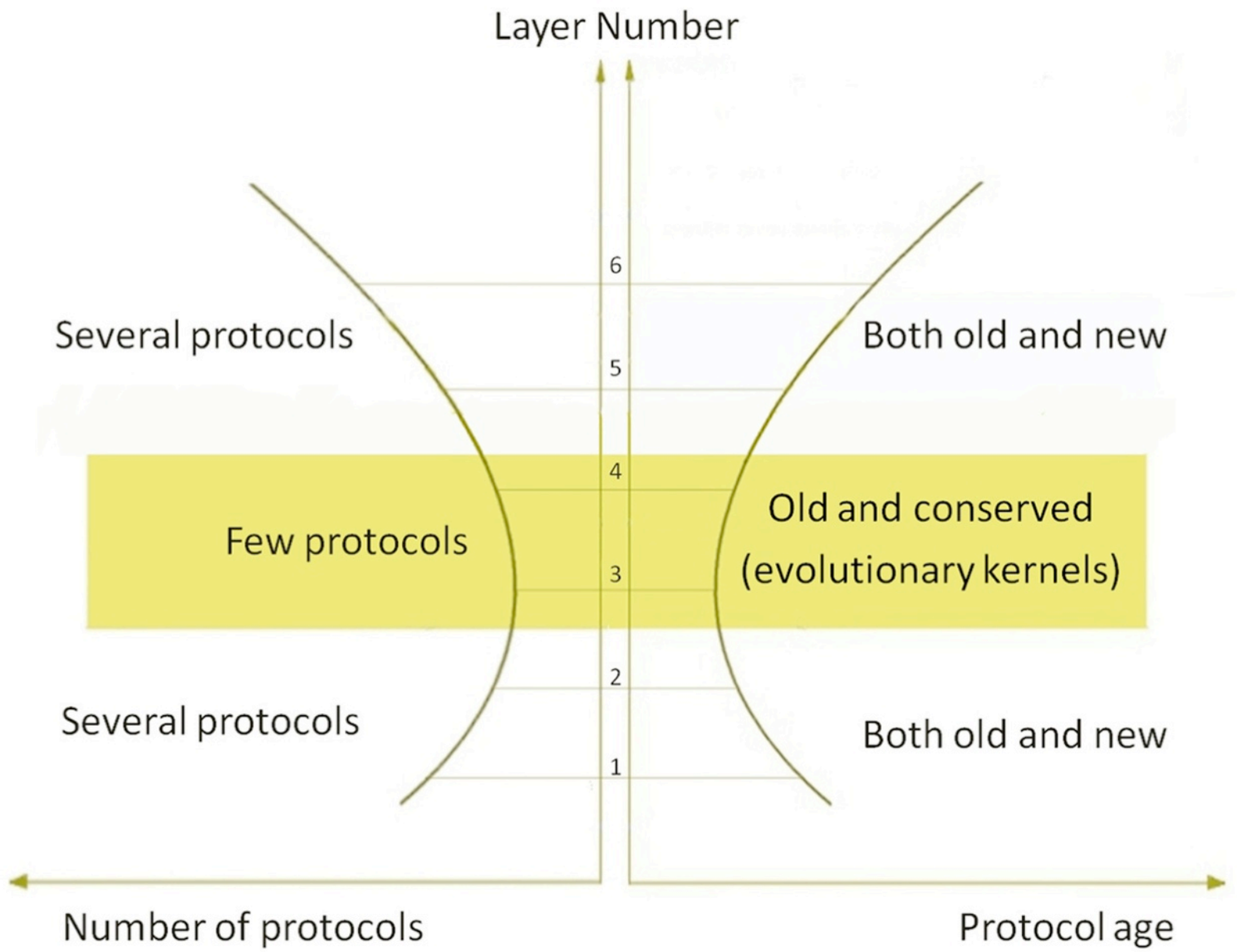
Difficulties

- Design/evolution of PPI
  - Resemblance with Internet's hourglass model



Difficulties

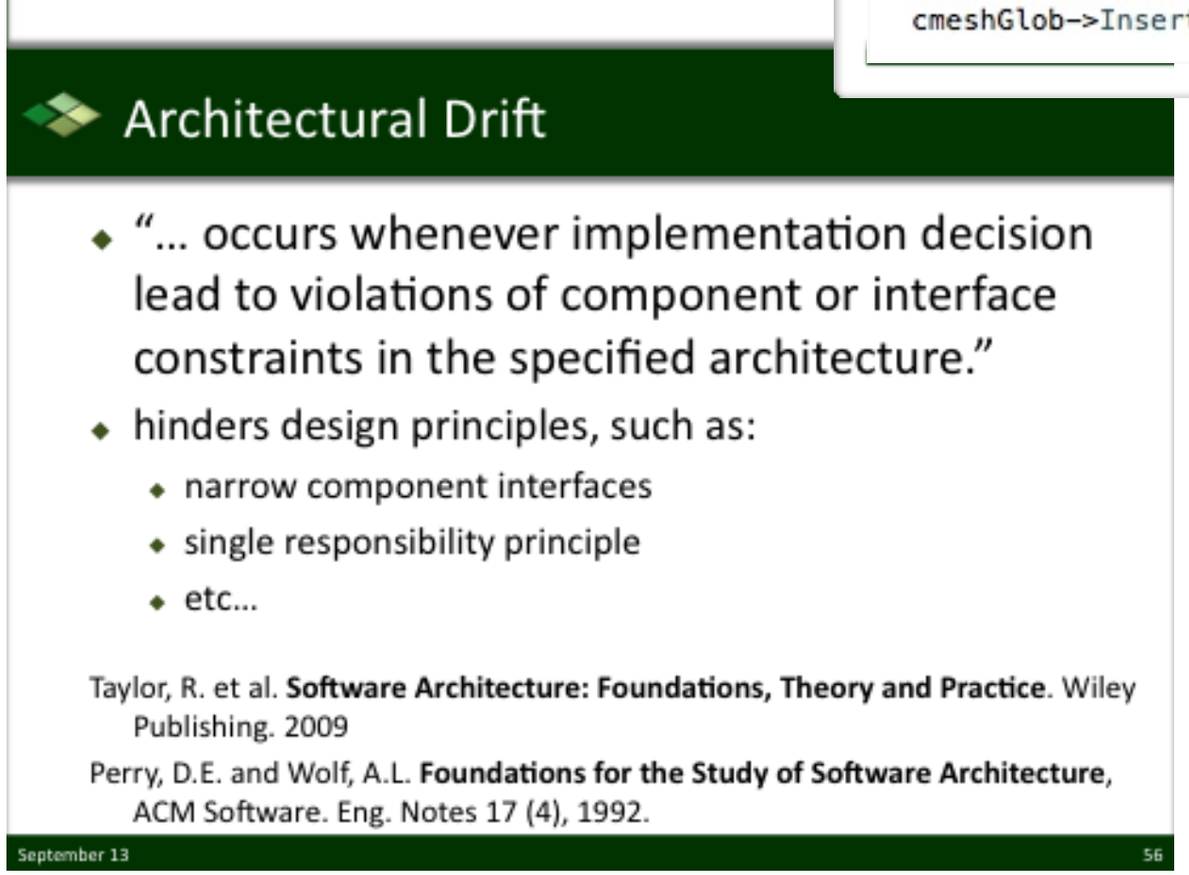
- Design/evolution of PPI
  - Resemblance with Internet's hourglass model
- Is ossification a problem?



Difficulties

- Implementing the specific family of MHM methods
  - NeoPZ's architectural degradation (drift)
  - Not only...

```
gmesh->ResetReference();
TPZCompMesh* cmeshGlob = new TPZCompMesh(gmesh);

// The MHM method needs one part to be computed in the global mesh
// and another one to be computed locally in each element.
// In this example, the part computed globally is a constant for each element
// that's why pOrder=0.
cmeshGlob->SetDefaultOrder(pOrder);
cmeshGlob->SetDimModel(2);
cmeshGlob->SetAllCreateFunctionsDiscontinuous();

// This material is fake because our MHM implementation uses the Multiphysics
// feature of NeoPZ, and it's in this feature that we specify the material/contribute()
    method
TPZMaterial* matGlob = new TPZMatPoisson3d( matIdGlob, 2);  // fake...
TPZAutoPointer<TPZMaterial> mat1(matGlob);
cmeshGlob->InsertMaterialObject( matGlob );
```

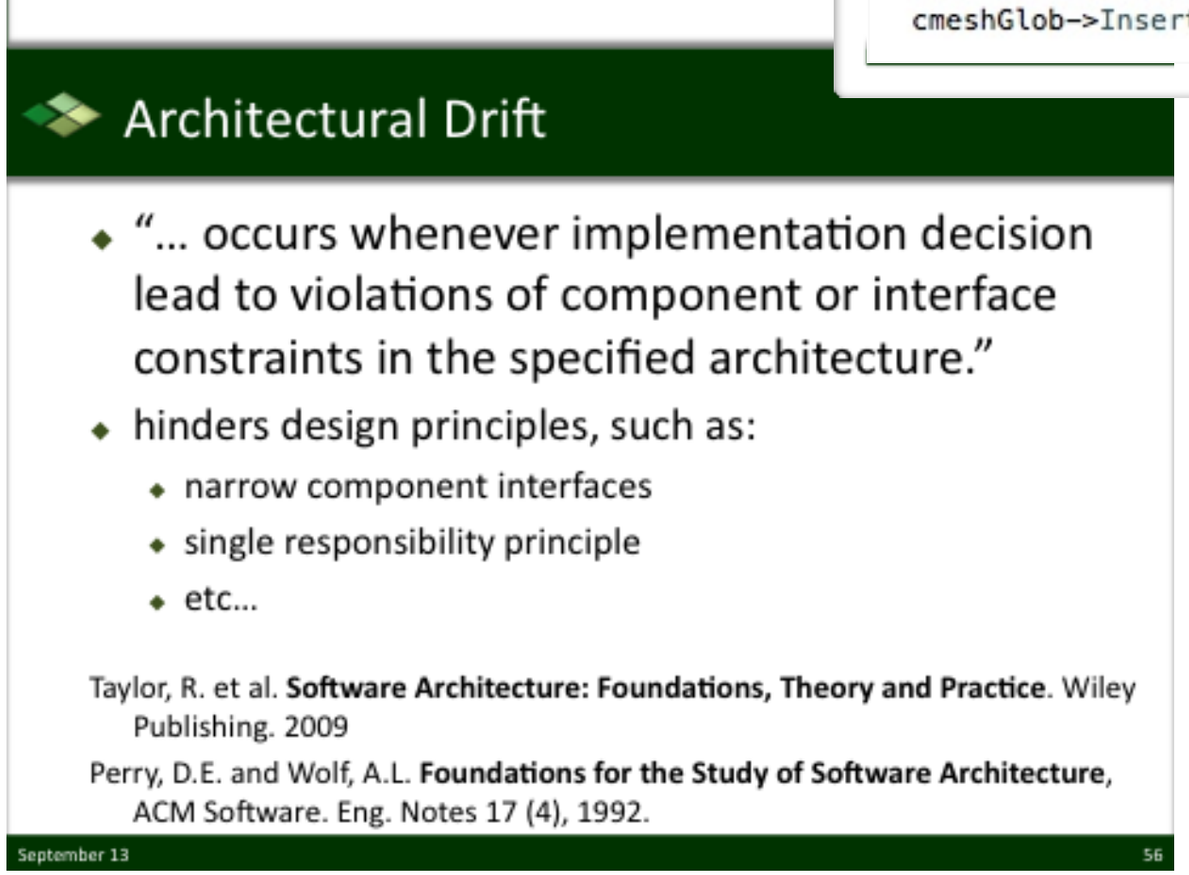


Source: Alessandro Garcia (Brazilian Academy of Science)

Difficulties

- Implementing the specific family of MHM methods
  - NeoPZ's architectural degradation (drift)
  - Not only...

```
gmesh->ResetReference();
TPZCompMesh* cmeshGlob = new TPZCompMesh(gmesh);

// The MHM method needs one part to be computed in the global mesh
// and another one to be computed locally in each element.
// In this example, the part computed globally is a constant for each element
// that's why pOrder=0.
cmeshGlob->SetDefaultOrder(pOrder);
cmeshGlob->SetDimModel(2);
cmeshGlob->SetAllCreateFunctionsDiscontinuous();

// This material is fake because our MHM implementation uses the Multiphysics
// feature of NeoPZ, and it's in this feature that we specify the material/contribute()
    method
TPZMaterial* matGlob = new TPZMatPoisson3d( matIdGlob, 2);  // fake...
TPZAutoPointer<TPZMaterial> mat1(matGlob);
cmeshGlob->InsertMaterialObject( matGlob );
```



Source: Alessandro Garcia (Brazilian Academy of Science)

- But also...

Difficulties

- Implementing the specific family of MHM methods
  - NeoPZ's architectural degradation (drift)
  - Not only...

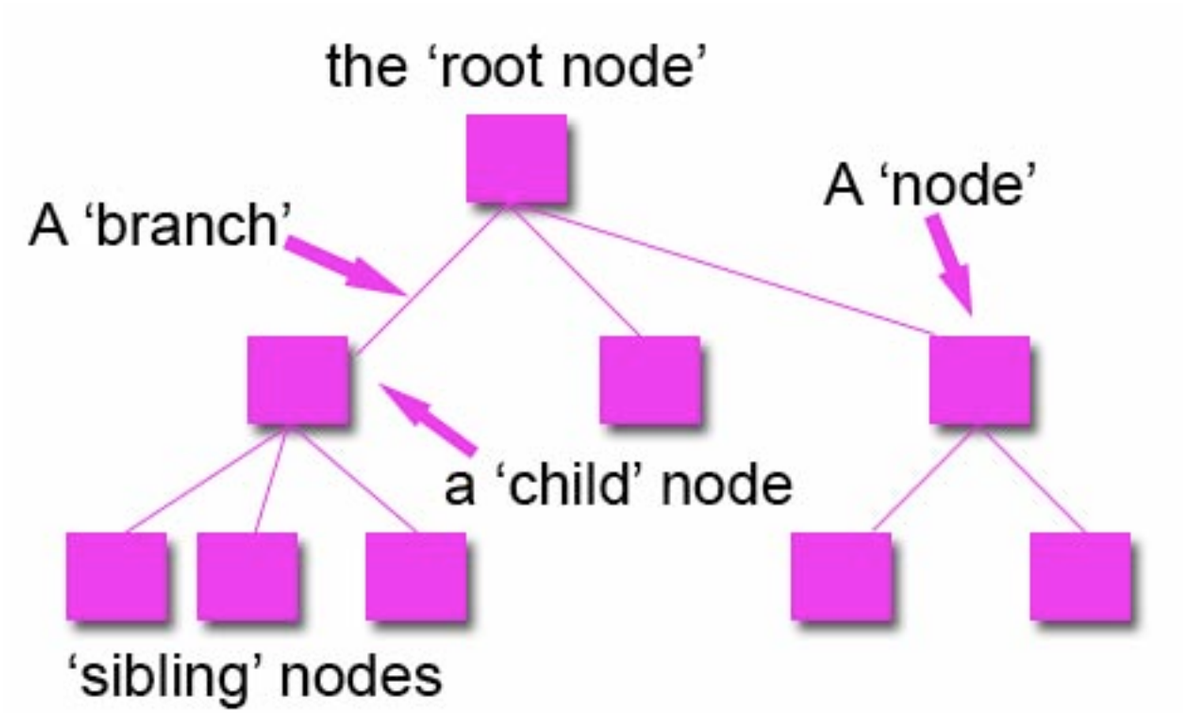
```
gmesh->ResetReference();
TPZCompMesh* cmeshGlob = new TPZCompMesh(gmesh);

// The MHM method needs one part to be computed in the global mesh
// and another one to be computed locally in each element.
// In this example, the part computed globally is a constant for each element
// that's why pOrder=0.
cmeshGlob->SetDefaultOrder(pOrder);
cmeshGlob->SetDimModel(2);
cmeshGlob->SetAllCreateFunctionsDiscontinuous();

// This material is fake because our MHM implementation uses the Multiphysics
// feature of NeoPZ, and it's in this feature that we specify the material/contribute()
    method
TPZMaterial* matGlob = new TPZMatPoisson3d( matIdGlob, 2);  // fake...
TPZAutoPointer<TPZMaterial> mat1(matGlob);
cmeshGlob->InsertMaterialObject( matGlob );
```

Source: Alessandro Garcia (Brazilian Academy of Science)

- But also...
  - Need for stressing the ability of the SPiNMe platform to use different numerical libraries

Difficulties

- Implementing the specific family of MHM methods
    - NeoPZ's architectural degradation (drift)
- Implementation of a new FEM library specifically crafted for exploring the loosely-coupled strategy of MHM methods to solve global and local problems

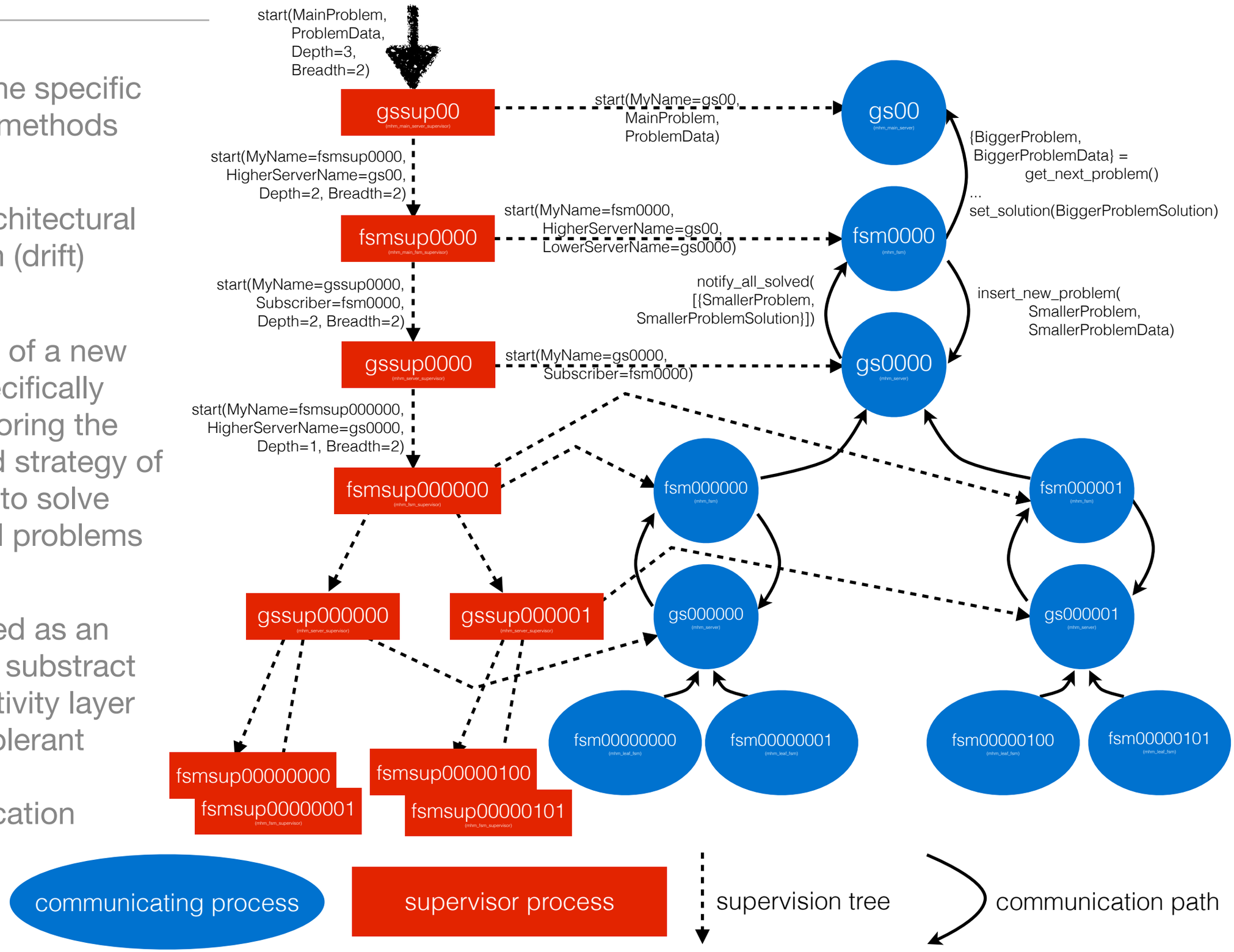


PARTS OF A TREE DATA STRUCTURE

Source: www.teach-ict.com

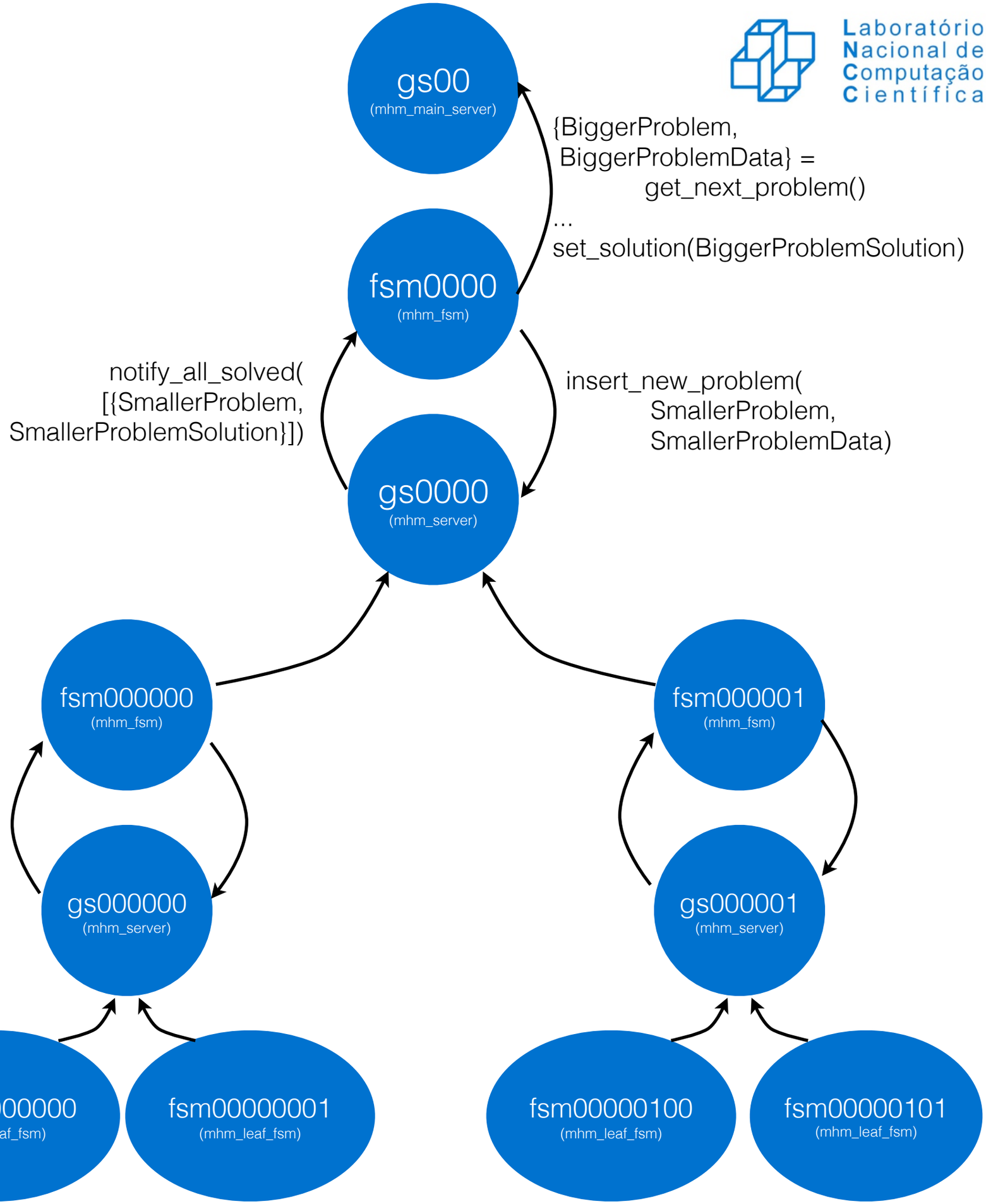
Difficulties

- Implementing the specific family of MHM methods
  - NeoPZ's architectural degradation (drift)
- Implementation of a new FEM library specifically crafted for exploring the loosely-coupled strategy of MHM methods to solve global and local problems
  - Erlang used as an additional substract of productivity layer for fault-tolerant process communication



Overall design

- GS processes:
  - Workload queueing
  - Notification of solved workloads
- FSM processes:
  - Problem splitting and reduction
  - Global-local solving
- Leaf-FSM processes:
  - Local solving



Future work

Laboratório Nacional de Computação Científica

- NeoPZ-related:
  - Nonlinear PDEs
  - Symbolic coding
  - JiT compilation of Lua code

- MHM-related:
  - Variable depth and breadth in subtrees
  - Support for dynamic adaptivity (c.f. Diego's slides!)
- SPiNMe-related:
  - Link to scientific hypothesis database

Thank you!!!
Merci!!!

Antônio Tadeu A. Gomes
Diego Paredes
Frederico Teixeira
Frédéric Valentin

atagomes@lncc.br

http://www.lncc.br/sinapad/SPiNMe
http://www.facebook.com/sinapad

Laboratório Nacional de Computação Científica

CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico

SINAPAD