Building Simulation Applications using ORCAN

Typical components of a simulation application:

- Geometry (CAD)
- Volume and surface meshes
- Mesh generator
- Discretization of various differential equations
- Solvers
- Visualization of Geometry/Mesh/Result
- (Graphical) user interface, steering
- Import/Export to and from other applications
- Coupling with other applications

a typical, simple use case:

- get a geometry
- convert to surface mesh
- generate volume mesh from surface mesh
- assign material (properties, problems to be solved), sources, boundary conditions
- discretize (set up matrix)
- solve
- visualize results

Geometry

Creation:

- complex cases -> CAD, import from IGS, Step, ...
- testing, case studies, research: fast creation of relatively simple, parameterized models

Shape Healing:

- subdivision of space into non-overlapping, closed volumes ("waterproofness")
- non-overlapping, unique faces separating volumes

Surface tesselation (input for grid generators, visualization tools)

- variable accuracy (deflection)

How to get a geometry ?

- ocs::GeometryRef geom=ocs::Geometry::New()

- if it can read e.g. IGS: load a file

```
if(geom.I.IGESReaderPtr) {
    geom.I.IGESReader->Read(infile);
}
```

- or use an appropriate GeometryReader, which can work with the Interfaces offered by the geometry reference:

```
ocs::GeometryReaderRef  reader=ocs::GeometryReader::New()
reader.SetInput(infile)
reader.SetOutput(geom)
reader.Execute()
```

from geometry to surface mesh

Mesh Interfaces:

- Create/delete vertices, faces, elements
- assign attributes to vertices/faces/elements
- Query topology, hierarchy

from Geometry to SurfMesh: SurfMeshGen

```
ocs::SurfMeshGenRef smeshgen=ocs::SurfMeshGen::New()

ocs::SurfMeshRef smesh=ocs::SurfMesh::New()

smeshgen.SetInput(geom)

smeshgen.SetOutput(smesh)

smeshgen.Execute()
```

Information passed to SurfMesh: PatchID, LeftID, RightID

from surface mesh to volume mesh

VolMeshGen

```
ocs::VolMeshGenRef generator=ocs::VolMeshGen::New()

ocs::VolMeshRef volmesh=ocs::VolMesh::New()

generator.SetInput(smesh)

generator.SetOutput(volmesh)

   ... set parameters ...

generator.Execute()
```

- all currently used Generators assign VolumeID to elements
  -> assign e.g. materials, source terms
- PatchID is assigned to element faces belonging to a common physical surface
  -> assignment of boundary conditions
- LeftID/RightID: distinction between inner and outer boundaries

If the VolMeshGenerator does not assign Patch/Left/RightID:

use a MeshCoupling

```
ocs::MeshCouplingRef coupler=ocs::MeshCoupling::New();

coupler.I.CouplingExtractAllPtr-> ExtractSurfMeshFromVolFacesCopyAll
( mVolMesh,mVolumeSurfMesh,mVolumeFaces,mVolVertIDs);

coupler.I.SurfMeshInterpolatorPtr->SetInputMeshes(mGeomSurfMesh,
mVolumeSurfMesh);

coupler.I.SurfMeshInterpolatorPtr->MapFrom1To2("PatchID",'e')
```

Association between volumes and properties (materials)

- VolumeBroker
- Material Database
- Material

set up an equation system, and solve it

PDEDiscretizer interfaces

LESSolver interfaces

```
ocs::PDEDiscretizerRef discretizer=ocs::PDEDiscretizer::New("FemHeat")

ocs::LESSolverRef solver=ocs::LESSolver::New()

set parameters via PropertyMap ...

discretizer.I.LESCreate->Init(volmesh,solver)

discretizer.I.LESCreate->MakeMatrixEntries()

solver.I.Solve->SolveLES()

discretizer.I.Solution->TransferToMesh()
```

When iterating nonlinear problems ->UpdateMatrixEntries()

Coupling, e.g. of thermal conduction with a radiation model ?

Radiation interfaces

MeshCoupling interfaces

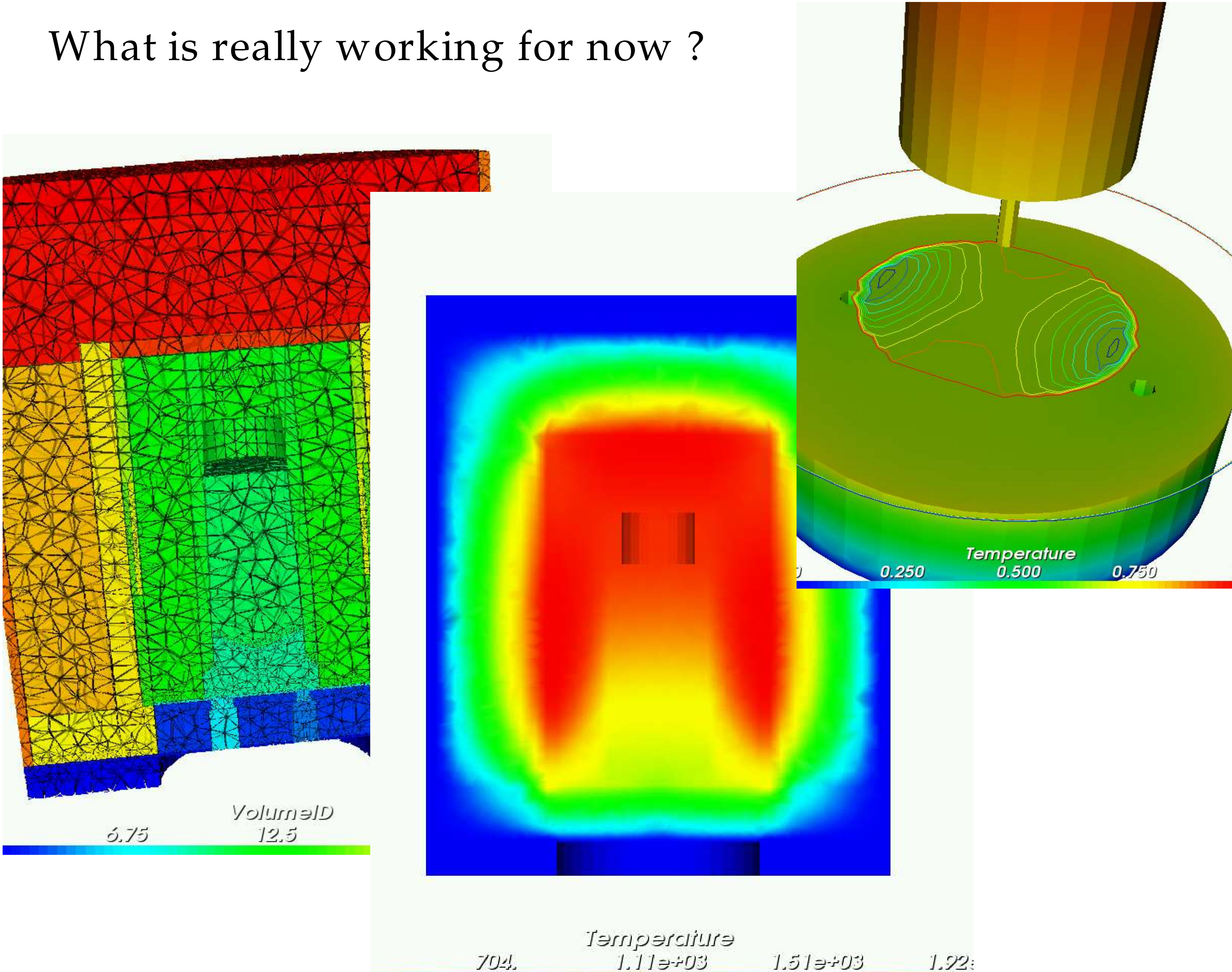
RadiationConductionCoupling interfaces

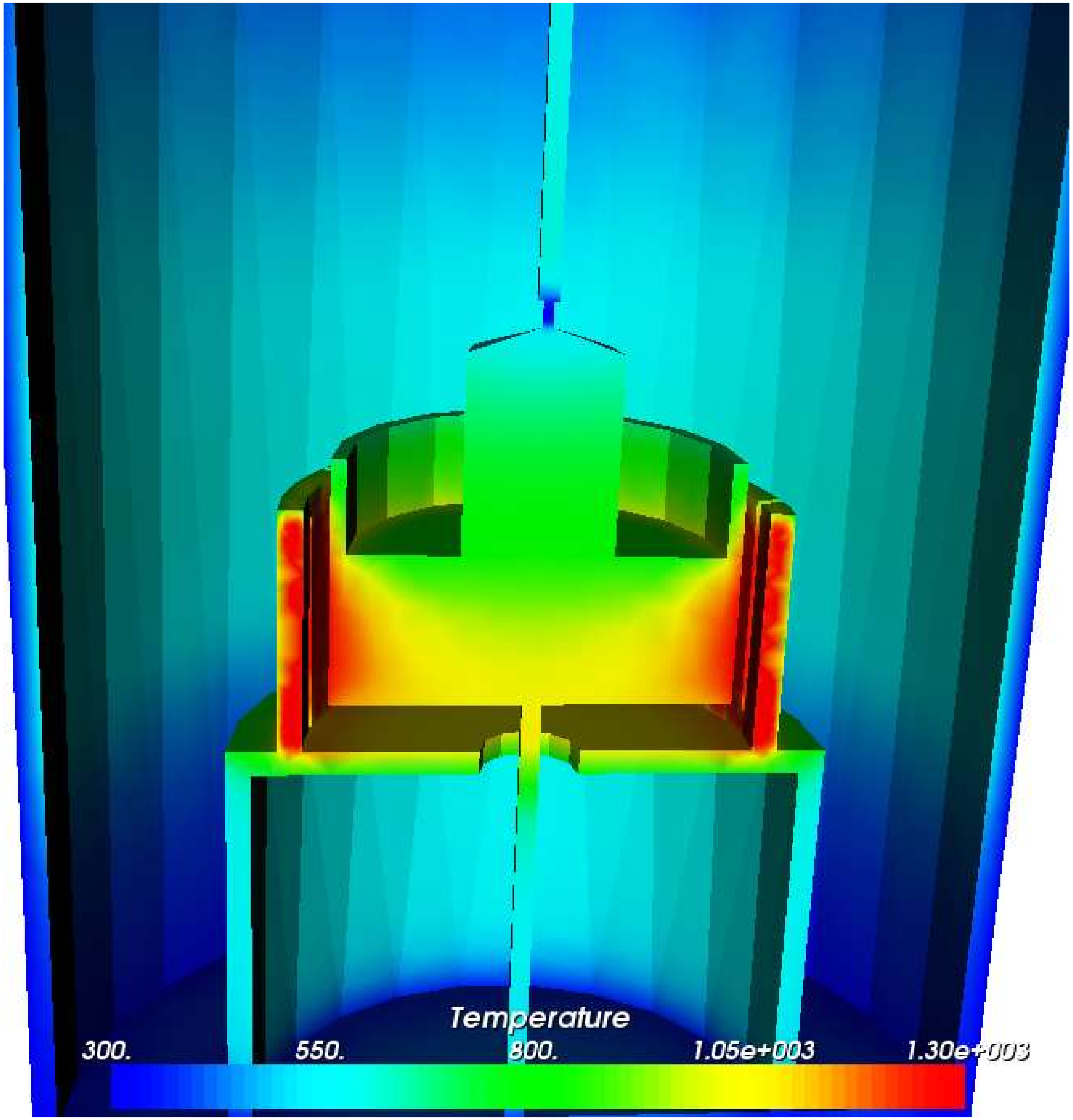
```
ocs::RadiationConductionCoupling
coupler=ocs::RadiationConductionCoupling::New()

coupler.I.Couple->SetVolMesh(vmesh)
coupler.I.Couple->SetConduction(discretizer)
coupler.I.Couple->SetRadiation(radiation)

coupler.I.Couple->Init()

coupler.I.Couple->Iterate()
```

Visualization:

```
// get output context
mOutputContext = ocs::OutputContext::New();

if( !mOutputContext ) {
    OCERROR( "could not create OutputContext instance");
    return;
}


mVisualization = mOutputContext.I.DefaultPtr->GetVisualization();
if( !mVisualization ) {
    OCERROR( "can not get active visualization from context");

    return;
}


mVisualization.I.MeshVisSinglePtr->SetInput(volmesh)
```

What is really working for now ?




Temperature
300.
550.
800.
1.05e+003
1.30e+003

Next steps: Create an application ...

Code Coupling

codes computing different domains: e.g. Heat conduction and radiation coupled with a 3D Navier-Stokes solver (Sthamas3D)

either by

- simple exchange of boundary conditions: unstable, slow, but simple
- use of a common matrix (coupling equations from interpolation component): most efficient, not always possible
- "Partitioned but strongly coupled iteration schemes for nonlinear fluid-structure interaction": H.G.Matthies, J. Steindorf, Computers and Structures 80 (2002)

  2 Subsystems with their own solvers, these are used to obtain approximations for the derivatives of coupling matrix, which can be used for a Block-Newton method to solve the coupled system

the big dream: bring things together in a flexible way ...

... OpenCASCADE, Tetgen, Gmsh,Laspack, Netgen, Petsc, LSS-Fem, Vtk, ...

OpenFOAM

Overture

GetDP

GetFem

create some dynamically configurable application, to select components, display and allow to modify their parameters,...