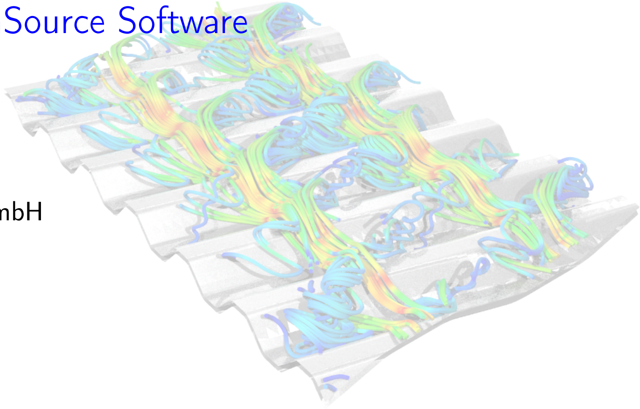
Using OpenSource Software efficiently



Johann Turnow
silentdynamics GmbH

2017-03-21

Contents

Motivation

InsightCAE

Example

Summary

OpenSource Users

What do OpenSource users expect?

- Use OpenSource Software (OSS) out of the box
- Easy to understand / simple handling
- Stable and fast environment!
- Includes every feature
- Validated performance
- Strong community support!
- Works on every platform

What do OpenSource (OpenFOAM) users get?

- Looking for OpenFOAM → four branches?!
- *Which one is the best??*
- Easy to understand?
- *At least for PhD students who have the time....*
- Stable?
- *Try* `rhoSimpleFoam`
- Validated performance?
- *Where is the central database?*
- Strong community support!
- ...
- Works on every platform?
- *Try to compile OpenFOAM on the HLRN!*

What do OpenSource (OpenFOAM) users need?

- Adressing numerical simulations (FEM/CFD) we need the full package
- Starting from:
  - Geometry generation / modification
  - Handle complex geometries
  - Trustful and fast meshing
  - Stable and fast simulation
  - Efficient post processing
  - Report generation (automated?)

What do OpenSource (OpenFOAM) users do?

- They do the whole dance!
  - Combine different openSource Tools
  - Geometry: FreeCAD / Blender / MeshLAB ...
  - Meshing: OpenFOAM / Netgen / Gmsh
  - Solving: OpenFOAM / Code Aster / Code Saturn
  - PostProcessing: Paraview / Ensight / python
- Of course, working process is possible
- But is it really efficient?

What do OpenSource (OpenFOAM) users require?

- Using CAE (CFD and/or FEM) productively for design tasks
  - involves repeated analysis of numerous similar variants
  - quick and efficient, with minimum pre/post processing effort
  - accurate, following a best-practice
  - safe, without need to repeat things because of user errors!
  - a thorough documentation of every analysis is needed
  - to review trends
  - backtrace errors

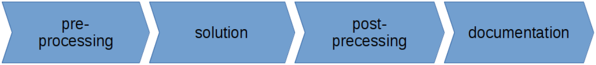
Can we achieve the requirements using OSS?

- Yes we can, because:

- OSS has an open architecture with many possibilities for automation
- many independent software tools for similar tasks are available but with different strengths and weaknesses
- need to combine and support multiple tools
- one quickly ends up in complicated workflows
- ⇒automation can hide complexity of the workflow

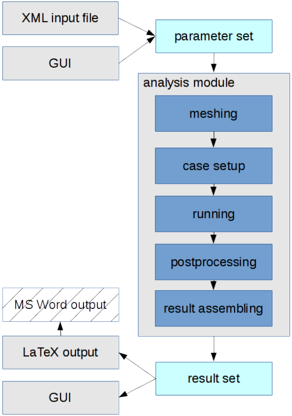
What is the idea/aim of “InsightCAE”?

- Conduct an “analysis” as much automated as possible



- Take a minimum of necessary parameters which need to be changed
- Standardization / best practice / testsuites for a given analysis/task
- Bundle addons, extensions and interfaces for all required external software utilities
- Automatic computation of many variants
- Fast case building
- Deployment: provide one installation package for all workflow-related software components

Modules of InsightCAE



Example for CFD workflow

- basic entity: analysis module
  - using a predefined parameter set given by the user
  - contains problem specific algorithms
  - includes all necessary / need steps
  - ends up with a final report

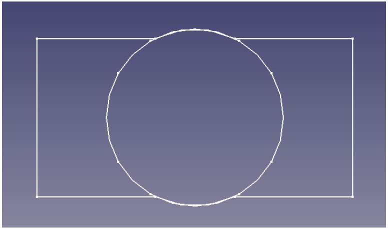
- Now we have an software that combines all the steps of different software modules for an efficient computation

Rectangular to circle duct

silentdynamics

Let's try InsightCAE for an classical example

- Task: Generate a duct that combines a rectangular with circle section
- Goals: uniform outflow, low pressure loss
- Using FreeCAD to sketch the inlet and outlet boundaries



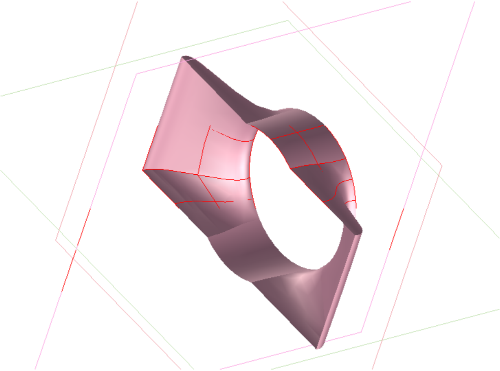
- So far so good
- Let's start the dance.

Create geometrie

- How to generate the stl surfaces for applying snappy?
- Use `iscad` from InsightCAE to build up the model
  - `iscad`
  - Use the console to import the FreeCAD sketches like:
    ```
    xsec_kf=Sketch(YZ, "sketches.fcstd", 'xsec_k');
    xsec_k=Wire(xsec_kf?alledges);
    xsec_l=Wire(Sketch(YZ, "sketches.fcstd",
    'xsec_l')?alledges);
    ```
- Extrude the sketches
  - `housing= Sweep(xsec_k, xsec_l);`

Create geometrie - Here we are

- Rebuild the 3D model using `iscad`

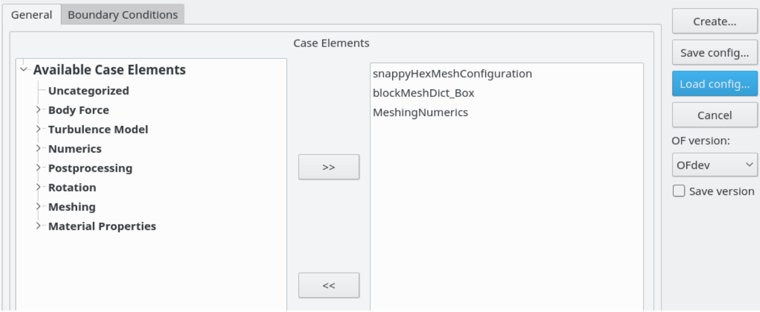


- Give me the stl file:
  `exportSTL("housing.stlb", 1e-2) « housing;`

Rectangular to circle duct

Mesh setup

- Generate the mesh setup for `snappyHexMesh`
- Use `isofCaseBuilder`



snappyHexMeshConfiguration
blockMeshDict_Box
MeshingNumerics

OF version: OFdev

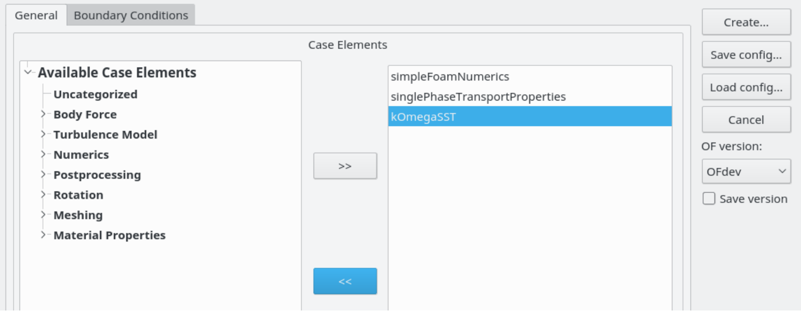
□ Save version

- Give me the state file for the mesh setup
- Save as `mesh.iscb`

Rectangular to circle duct

Case setup

- Generate the case setup
- Use `isofCaseBuilder`



simpleFoamNumerics
singlePhaseTransportProperties
kOmegaSST

OF version: OFdev

□ Save version

- Give me the state file for the case setup
- Save as `case.iscb`

Rectangular to circle duct

Running the case

- We are almost done
- Now easy automatization `run.sh`

```
# !/bin/bash
isofCaseBuilder -b mesh.iscb &&
blockMesh &&
decomposePar &&
mpirun -np 10 snappyHexMesh -overwrite -parallel &&
isofCaseBuilder -b case.iscb &&
mpirun -np 10 simpleFoam -parallel &&
reconstructPar -latestTime &&
rm -rf processor*
```

Rectangular to circle duct

Automated workflow

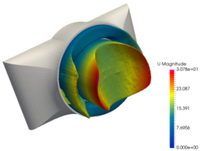
1. Change your sketches / Extrusion
2. Run the `run.sh` file
3. Start `paraview`

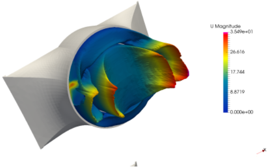
- That is not enough!
- PostProcessing still takes time
- Start `paraview` → generate layouts → save the state file
- Apply our python script: `isPV.py -b state.pvsm`
- Gives us the pictures as png file for every paraview layout!

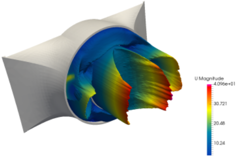
Rectangular to circle duct

Automated workflow - here we go

- Version 1,2,3, ...







Let’s summarize the dance

- Efficient computations are possible using OSS
- InsightCAE connects different OSS using predefined interfaces
- Standardized simulations / reducting of time consuming user mistakes
- Quality is ensured
- We get a really fast workflow!

End

Thank you very much!

Johann Turnow
Email: johann.turnow@silentdynamics.de
Tel.: +49 381 36 76 84 11

silentdynamics GmbH
http://silentdynamics.de

```
$ sudo add-apt-repository
http://downloads.silentdynamics.de/ubuntu
$ sudo apt-key adv -recv-key -keyserver keys.gnupg.net
79F5CBA4
$ sudo apt-get update
$ sudo apt-get install insightcae-base
```