THERMAL SHOCK SIMULATION IN A VALVE

Methodologies for multi-year complex simulations with several contributors

Jerome Ferrari - 09/12/2016
And D.Hersant, JP.Mathieu, S.Meunier, JF.Rit

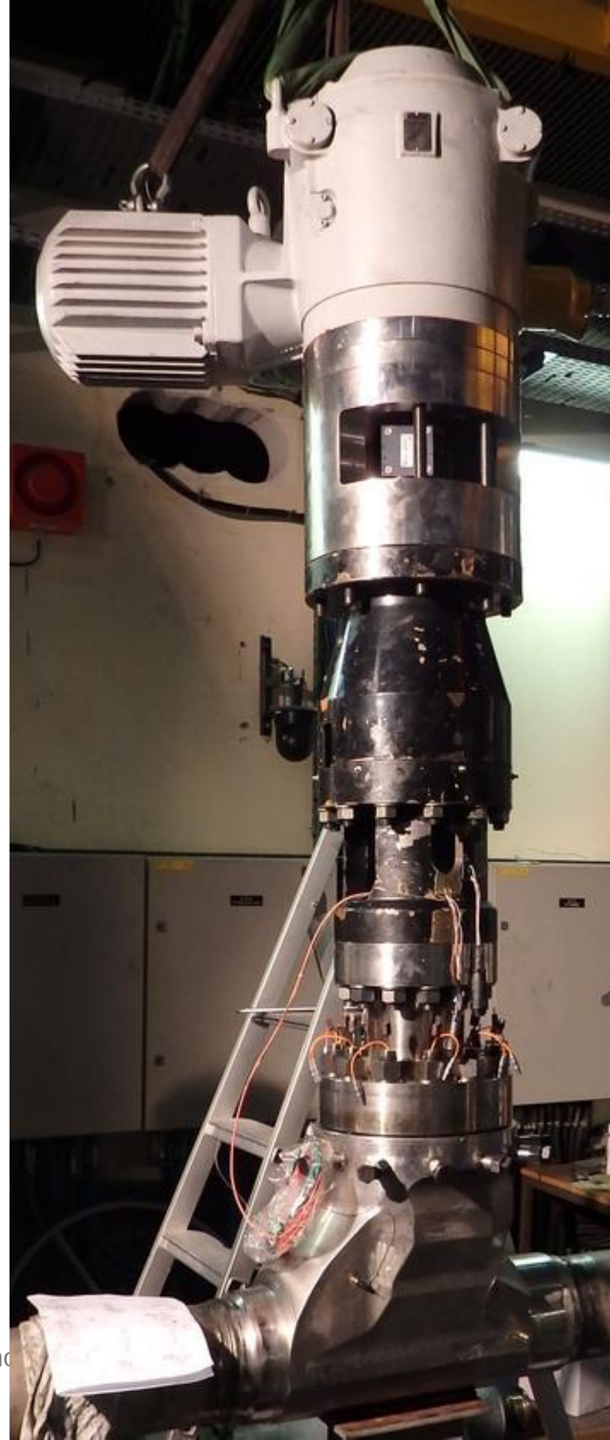
A LONG-TERM SIMULATION

- **Objectives**
  - Perform instrumented thermal shocks in a valve (~qualification test)
  - Perform a simulation of those thermal shocks

- **Key dates:**
  - 2012 : negociations with Velan
  - 07 june 2013 : Partnership signed between Velan and EDF R&D
  - 2015 : tests performed
  - 2015 : first simulation performed
    - 5 engineers involved + 1 trainee
  - 2016 : simulation improvements after experimental comparisons
    - 3 engineers involved + 1 trainee
  - 2017 : study 1 : « influence of reduced fluid velocity »
    - 3 engineers involved
  - 2018 : study 2 : « influence of valve size variations »
    - ??
  - 2019 : study 3 : « development of a simplified method for valve simulation »

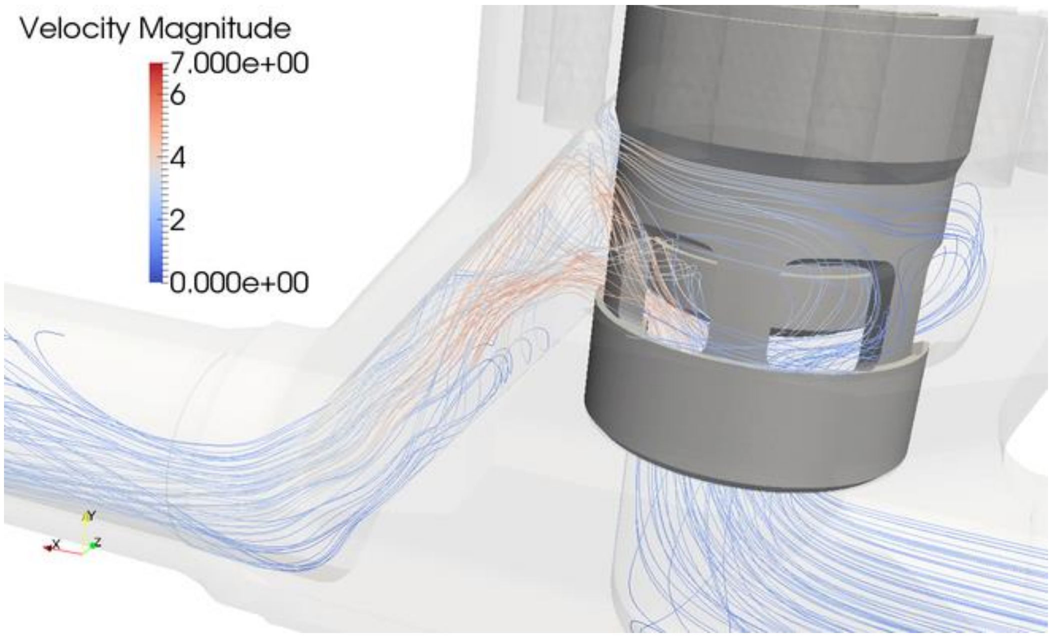
EXPERIMENT

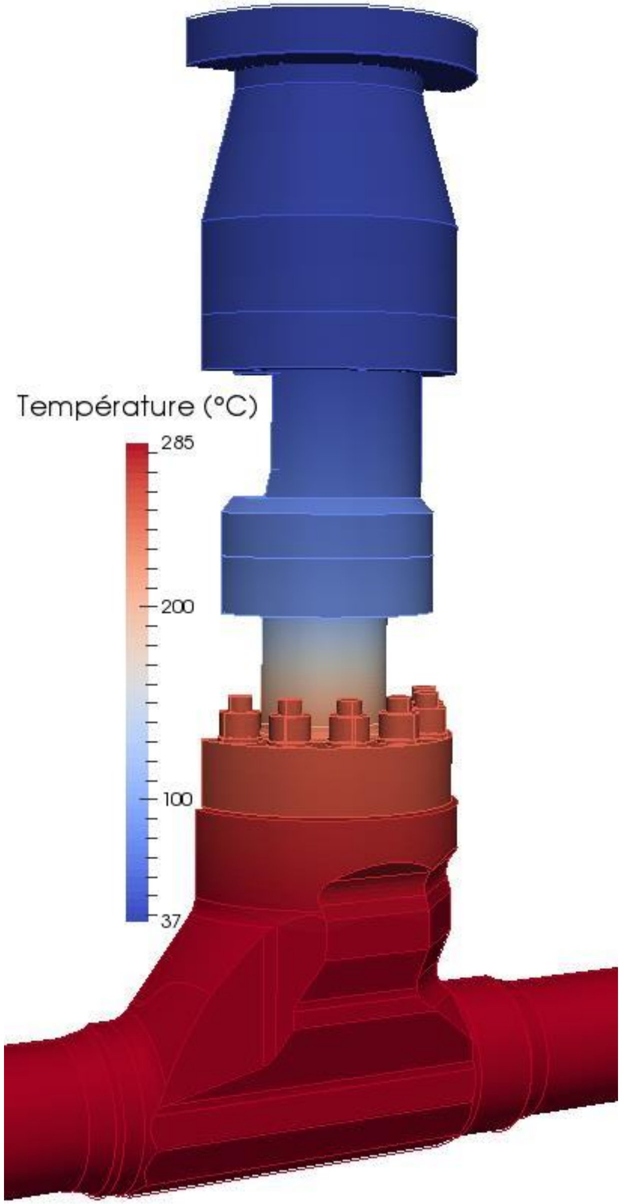
- **14 thermal shocks**
  - 14 × (cold + hot)
- **37 thermocouples**
- **12 strain gauges on the twelve studs of the body-bonnet flange**
- **Residual strain measurements**



SIMULATION

- **3 large computations on super computer**
  - Fluid → thermic → mechanics
  - *Code_Saturne → Code_Aster → Code_Aster*
  - 10h → 2h15 → 9h30



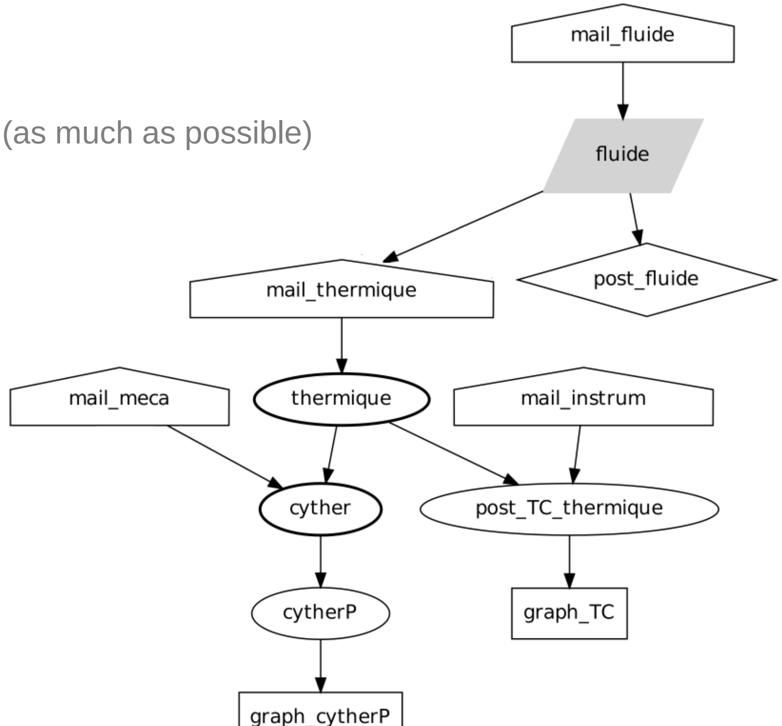


ORGANISATION

« ALL THAT CAN BE SCRIPTED SHALL BE SCRIPTED »

- **Needs : traceability, durability, work sharing and coordination**
  - Someone may leave the projet (from 5 to 3 engineers)
  - Someone may enter the project (ex: trainees)
  - One’s work must be integrated with other’s work
  - Change of simulation software version

- **Intensive use of scripting**
  - Meshes are made with Salome’s Python Interface (as much as possible)
  - Post-treatment and comparison with experiment:
    - A lot of Numpy and Matplotlib
    - A bit of scripted Paraview (from Salome)
    - A bit a manual Paraview (from Salome)
  - Computation stages are automated with Ctest



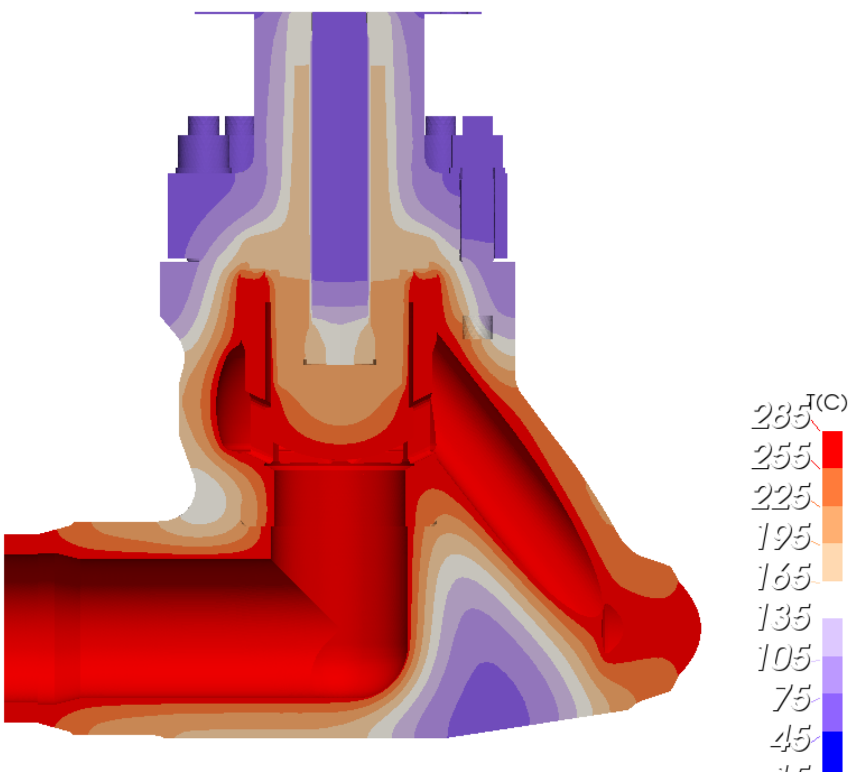
ORGANISATION

« OUR SIMULATION IS A SOFTWARE »

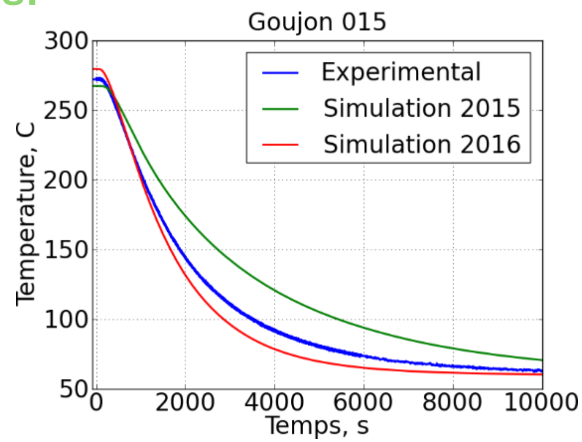
- **A source code repository with Subversion (EDF R&D forge Pleiade)**
  - 2 500 commits : 750 for meshing, 740 Code_Aster, 120 post-treatment, 350 reports
  - 27 000 lines of code : 10 000 for meshing, 8 000 Code_Aster, 9 000 post-treatment
- **A bug tracker with Redmine (EDF R&D forge Pleiade)**
  - 350 tickets since November 2013
  - #1 : « organize the code structure »
  - #350 : « test the new solver »
- **Daily automated tests with Jenkins**
  - If a script has changed
  - On the necessary parts of the simulation
  - « If your work disturbs someone else's work, we'll know it »
- **Continuous improvement**
  - Major versions for key report publication : v1.0 for 2015, v3.0 for 2016
  - Branches : for a trainee's work, to test a new idea

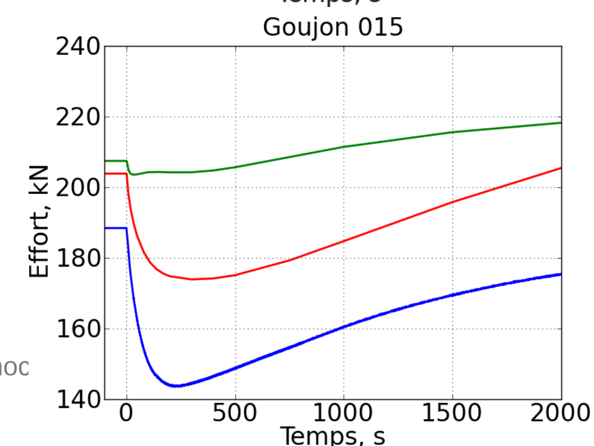
2016 : SIMULATION IMPROVEMENTS AFTER EXPERIMENTAL COMPARISONS

- **At the end of 2015: Qualitative disagreement : no untightening at start of cold shock**
  - Mesh refinement → Fail
  - Studs thread length in simulation → Fail
  - Artificial increase of the heat transfer coefficient between plug and cage → Fail
  - Add heat transfer coefficient between body and cage → **success!**



Choc chaud, t=200.0 s (surserrage max)

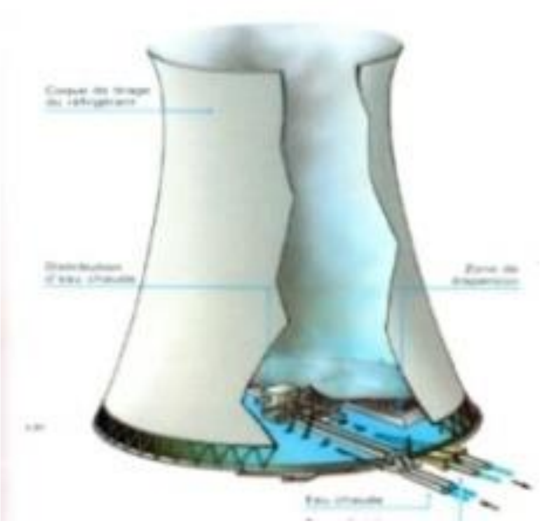




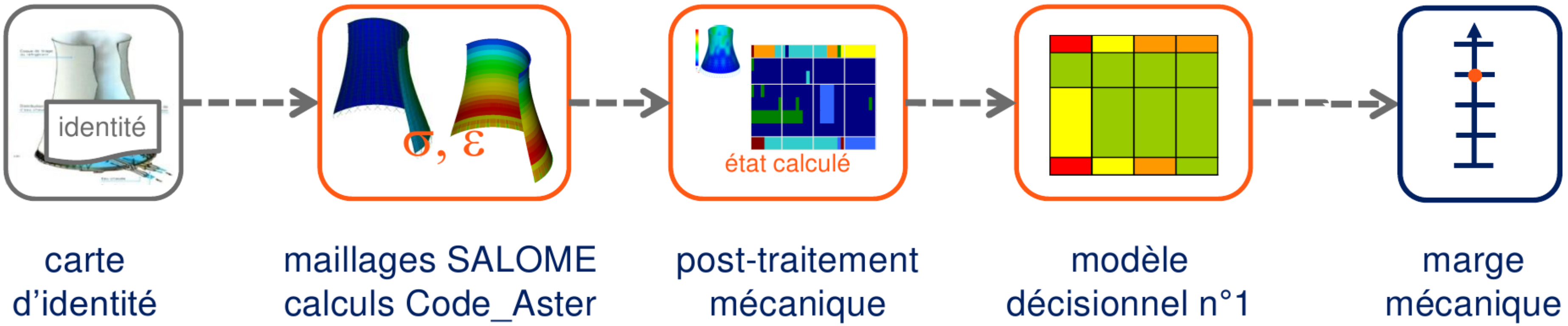
Thermal shoc

OTHER COMPARABLE WORKS

COOLING TOWERS



- **Chronology**
  - 2008 : technical proposal
  - 2012 : first delivery to SEPTEN
  - 2016 : still improving…
- **A business software**
  - No need to learn Code_Aster or Salome
- **10 different developers since the beginning**
  - Between 0 and 5 developers at the same time



ISSUES

- **The investment in the developing structure has to be cost-effective**
  - depends on study size and value
- **The objective is still getting the best simulation results**
  - Not the best software
  - Not the best developing environment

- **Difficulties to change habits**
  - To learn something new : Python, Subversion
- **Reluctances**
  - To comply with other’s formalism
  - To spend more time than otherwise needed
  - To feel tracked and exposed
  - To loose ownership and have his work diluted

CONCLUSIONS

- **Salome's Python interface allows us to integrate Salome in something bigger : multi-year complex simulation work**

- **Which allows us to get traceability, durability, work sharing and coordination via the use of software development techniques**

PERSPECTIVES

- **Simulation objective 2017 : « influence of reduced fluid velocity »**

- **But also, inner improvements**
  - Integrate the fluid simulation into the scripted workflow
    - Thanks to a newly solved « Prims at the wall » bug
  - More automated post-treatment
    - Thanks to a recently added feature