Efficiently exploit HPC resources in scientific analysis and visualization with ParaView

Nicolas Vuaille @FOSDEM23, CC-BY-NC-ND

kitware

About me Nicolas Vuaille

- C++ developer
- Free software enthusiast
- Employed by Kitware Europe
  - FOSS code contributions
  - Community interactions
  - Mainly on ParaView
- nicolas.vuaille@kitware.com

kitware

About ParaView

An **open-source** application and architecture for display and analysis of massive **scientific** datasets.

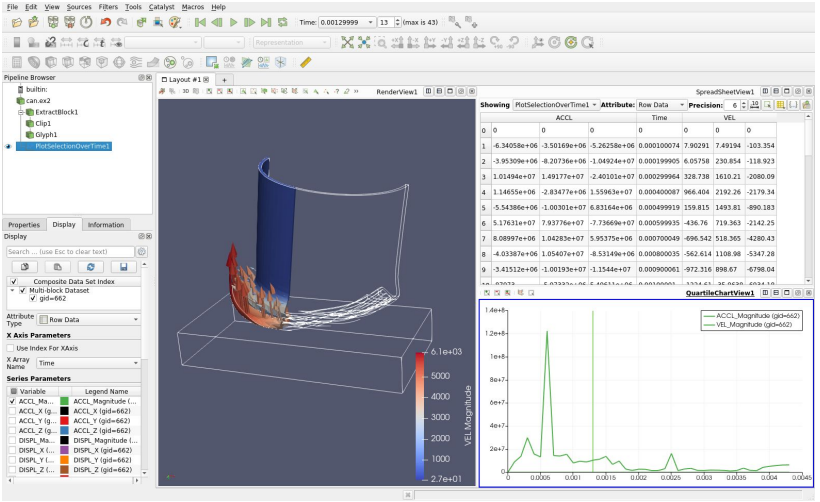
- Scientific data analysis
- Open-source (BSD-3-clause)
- Community (gitlab, discourse)
  - https://gitlab.kitware.com/paraview/paraview/-/blob/master/CONTRIBUTING.md
- Supported by Kitware (VTK, CMake)
  - Dev contracts, commercial support, courses

kitware

What's for ?

An open-source application and architecture for **display** and **analysis** of massive **scientific datasets**.

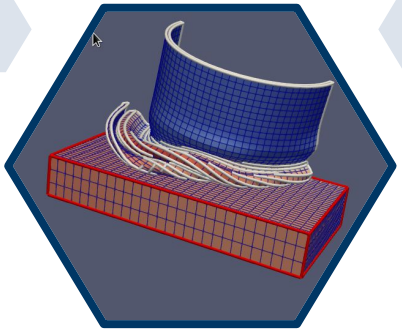
- Understand simulation output
  - 3D visualization
    - along with charts and more
  - Data Processing
    - filtering, data extraction
  - Realistic rendering
    - also make your comm' with real data



kitware

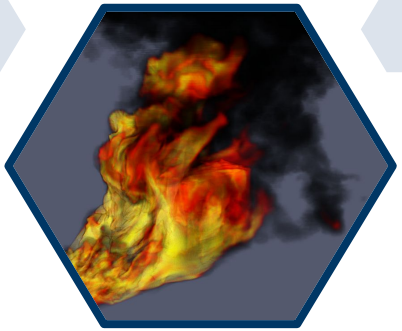
Features / Application Domains

Fluid Dynamic



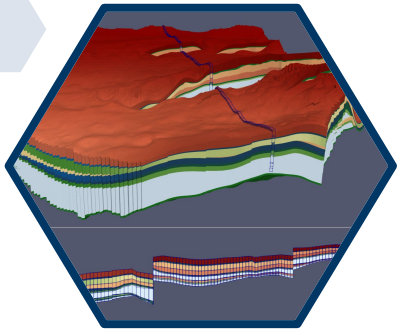
Structural Analysis

Medical



Particles

Astrophysic



Geoscience

kitware

How ?

An open-source **application** and **architecture** for display and analysis of massive scientific datasets.

- A GUI (click on buttons)
- A python wrapping (run scripts)
- A framework (plugins, custom apps, ...)
- Based on **Visualization ToolKit** (VTK)

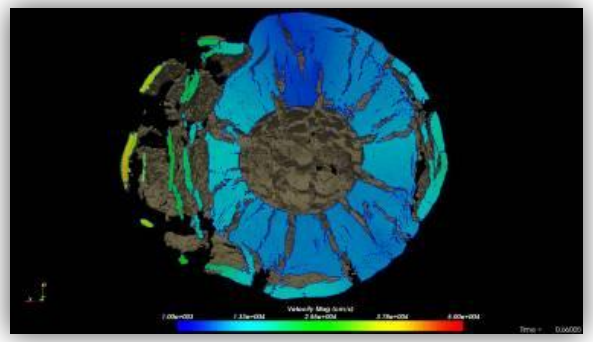
kitware

Which hardware to run it?

An open-source application and **architecture** for display and analysis of massive scientific datasets.

- From classical **desktop**
  - try out official binaries
    https://www.paraview.org/download/
- To largest **supercomputers**
  - large selection of build options (python, data distribution, parallelisation, rendering ...)

1 billion cell asteroid detonation simulation



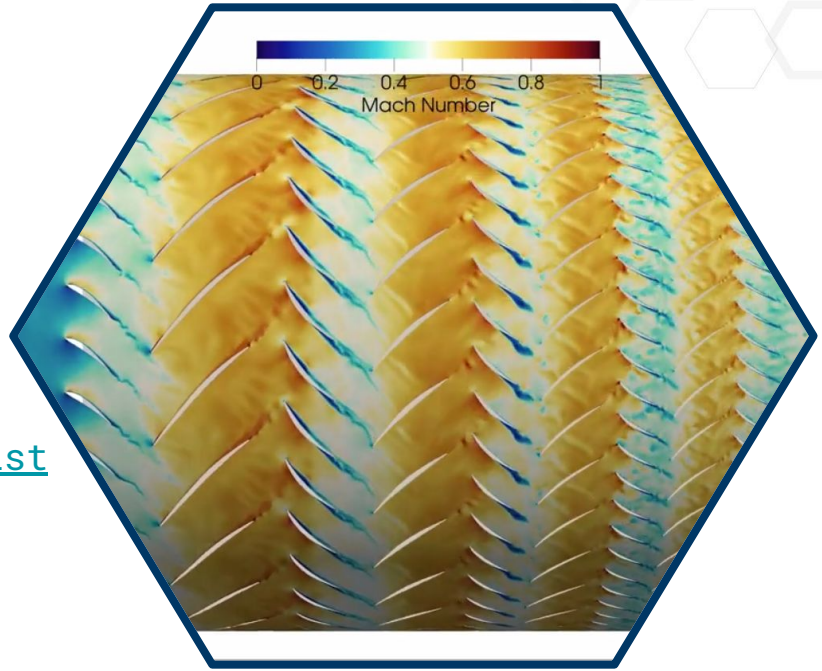
Source: Sandia National Lab

kitware

Usage

- Research and Industry
- Widely adopted at SuperComputing Sciviz contest:

  https://invidious.fdn.fr/playlist?list=PLyZk_jpQ4X_pQAUzmUG17DBQnrlN2zIyE



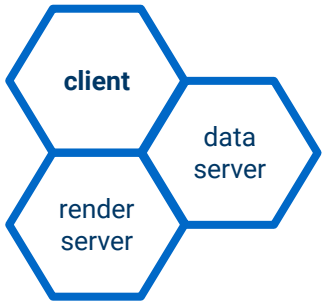
DLR Rig250 Compressor

kitware

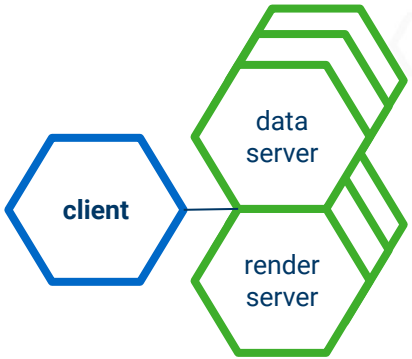
How ParaView efficiently exploit HPC resources ?

Client Server Architecture

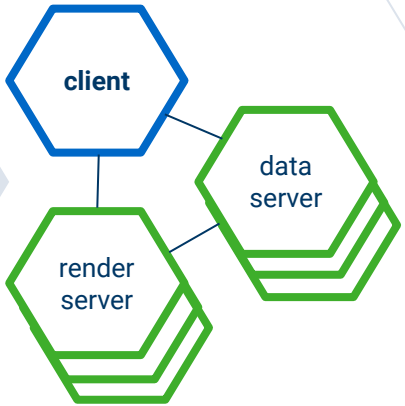
Built-in
paraview



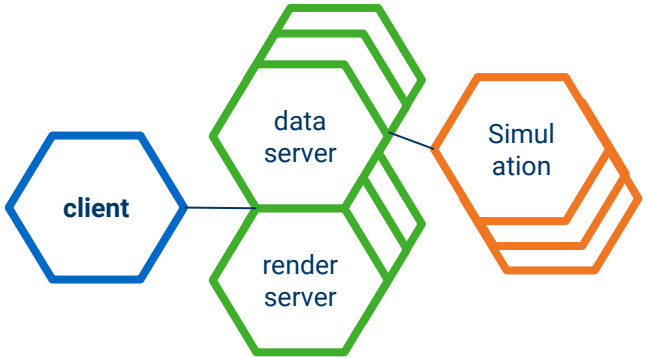
Distributed
pvserver



Graphic Nodes
data/render server



In Situ
catalyst

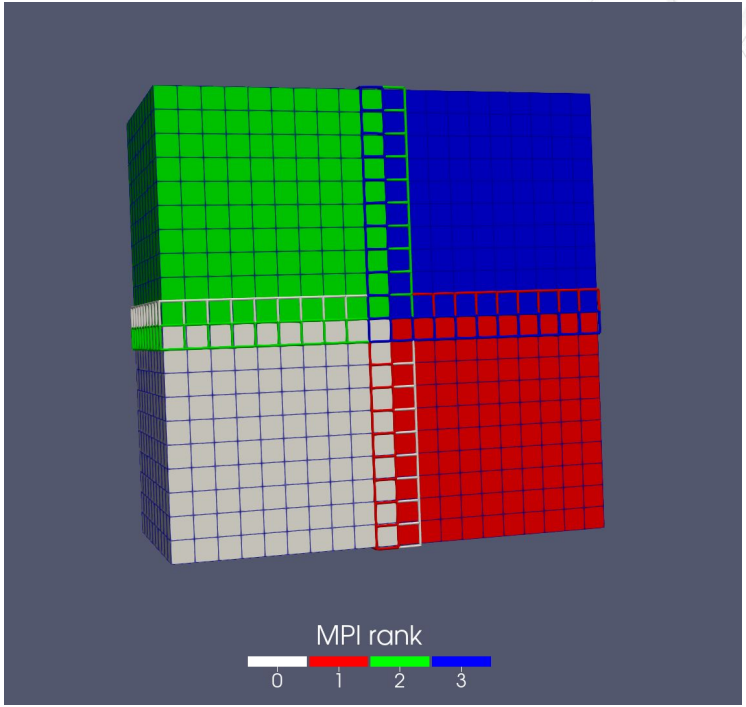


kitware

Data Distribution Analysis

- Based on **MPI** standard
- Readers distribute data over ranks
  - load balanced analysis
- Filters support Ghost Cells
  - when neighborhood info is needed
- Filters can redistribute data
  - ensure load balancing

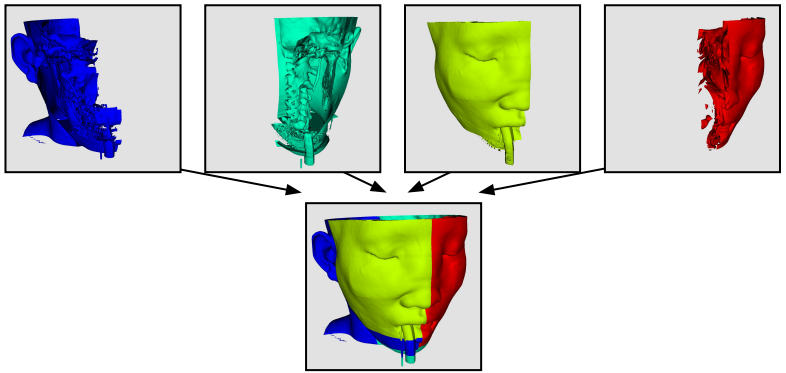
```
$ mpirun -n 4 pvserver
```



Ghost cells in wireframe

Data Distribution Visualization

- **Composition** (Ice-T)
- Dedicated rendering nodes
- Offscreen Rendering
- Multiple GPUs



```
$ mpirun -n 2 ./bin/renderserver --force-offscreen-rendering
```

kitware

Performances Instruction Parallelism

- SMPTools : **CPU** parallelism
  - Enable at build-time
  - Choose backend at runtime (OpenMP /TBB / C++ Threads)
  - Used in many algorithm

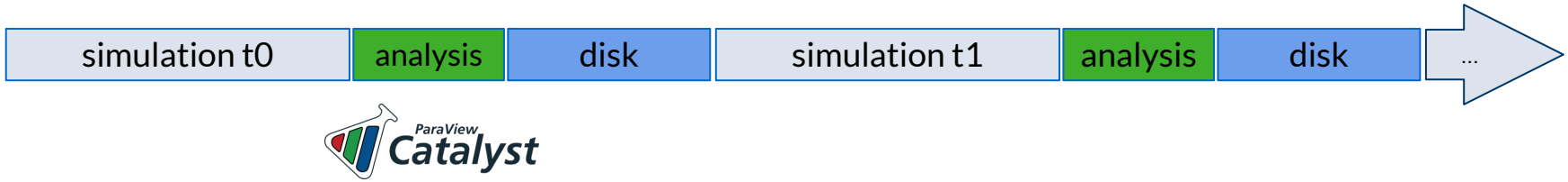
```
$ VTK_SMP_BACKEND_IN_USE=OpenMP ./bin/paraview
```

Performances Instruction Parallelism

- VTKm: **Heterogeneous** System ("Many core")
  - Optional third party
    - `$ cmake -DPARAVIEW_USE_VTKM=ON .`
  - Dedicated filters using it.
  - Backends: CUDA / OpenMP / TBB ...

Performances *In-situ*

- **Concurrent** analysis and visualization tasks during simulation
  - Reduce I/O
  - Increase value of stored data
  - Zero-copy analysis



Catalyst

- Standalone C/C++ API: https://gitlab.kitware.com/paraview/catalyst
- Minimal instrumentation
- ABI - stable
  - choose version at runtime !

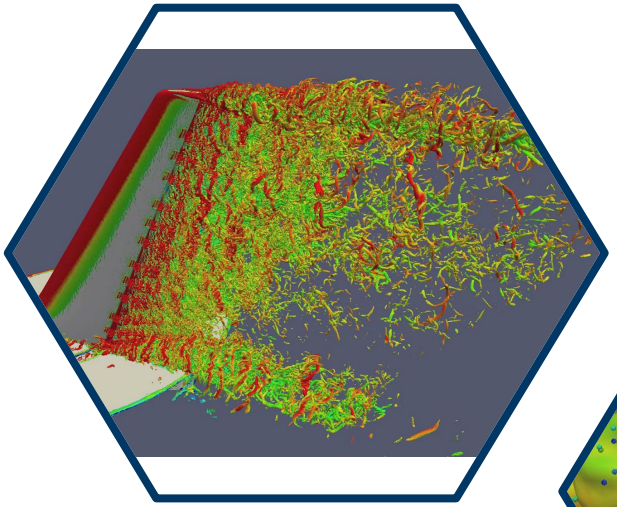
```
CatalystAdaptor::Initialize(argc, argv);
for (auto timeStep : timesteps)
{
  updateFields(timeStep);
  CatalystAdaptor::Execute(timeStep, grid, attributes);
}

CatalystAdaptor::Finalize();
MPI_Finalize();
```

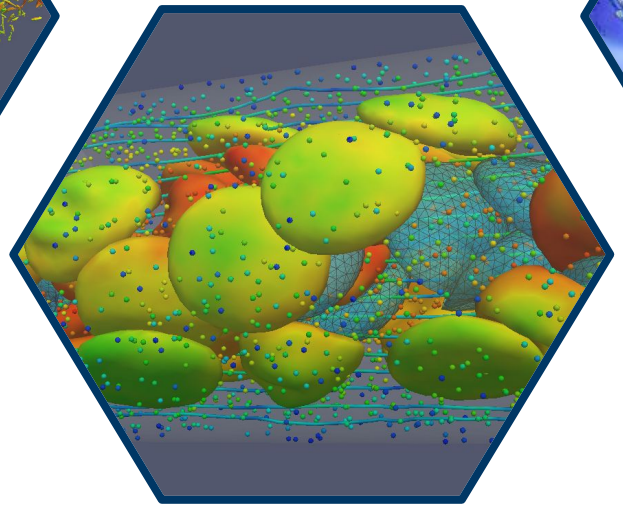
kitware

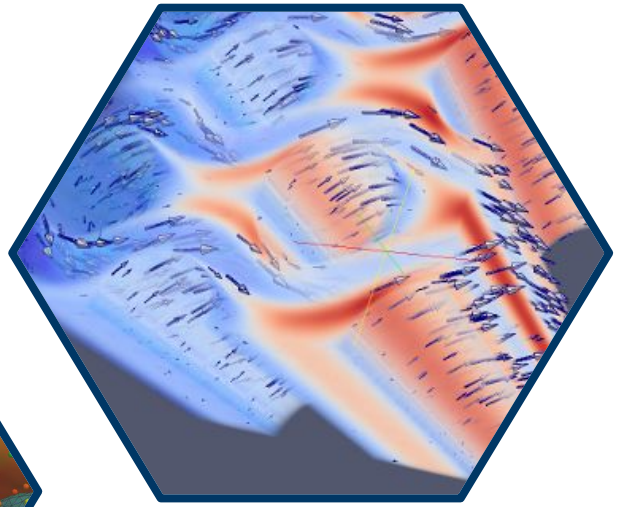
ParaView Catalyst

A set of **in-situ** data **analysis** and **visualization** capabilities, highly configurable



256K MPI ranks on Mira@ANL (BG/Q)





Code Saturne (EDF)

kitware

ParaView Catalyst

A set of in-situ data **analysis** and **visualization** capabilities, highly **configurable**

- ParaView implementation
- Configure with python scripting
  - can be generated from GUI
- Live Visualization from ParaView

kitware

Conclusion

- **Client-server** mode for batch processing
  - python scripting
  - good **scaling with MPI**
- **State-of-the-art** libraries for performance and distribution
  - MPI, VTKm, TBB
- API for **in-situ**

ParaView Application Architecture

User Interface

GUI / VR

Python

Web / jupyter

Catalyst

Custom App

ParaView Server Manager

VTK filter and rendering

Low Level APIs

OpenGL / IceT

MPI / DIY

OpenMP / TBB

VTKm

OpenXR

kitware

Roadmap

- In Transit (Adios2)
- Use of DIY
- Better VTKm integration
- New `ImplicitArrays` in VTK
  - "views" on memory, data compression, etc.

Questions ?

Thanks for attending!

kitware

Resources

- ParaView: https://www.paraview.org
- setup paraview server: https://docs.paraview.org/en/latest/ReferenceManual/parallelDataVisualization.html
- catalyst : https://gitlab.kitware.com/paraview/catalyst
- catalyst adios: https://gitlab.kitware.com/paraview/adioscatalyst
- VTKm: https://m.vtk.org/
- SMPTools: https://www.kitware.com/vtk-shared-memory-parallelism-tools-2021-updates/
- DIY: https://github.com/diatomic/diy
- Implicit arrays: https://www.kitware.com/vtkimplicitarrays-a-new-vtk-framework-for-manipulating-array-like-data/

kitware