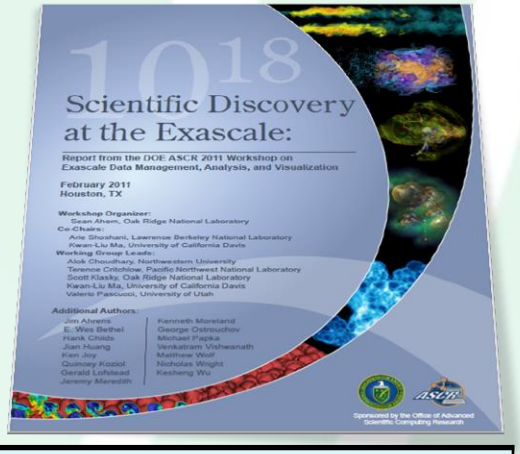
Extreme Scale *In Situ* Analysis with ParaView Catalyst

Joachim Pouderoux
Julien Jomier
Kitware SAS

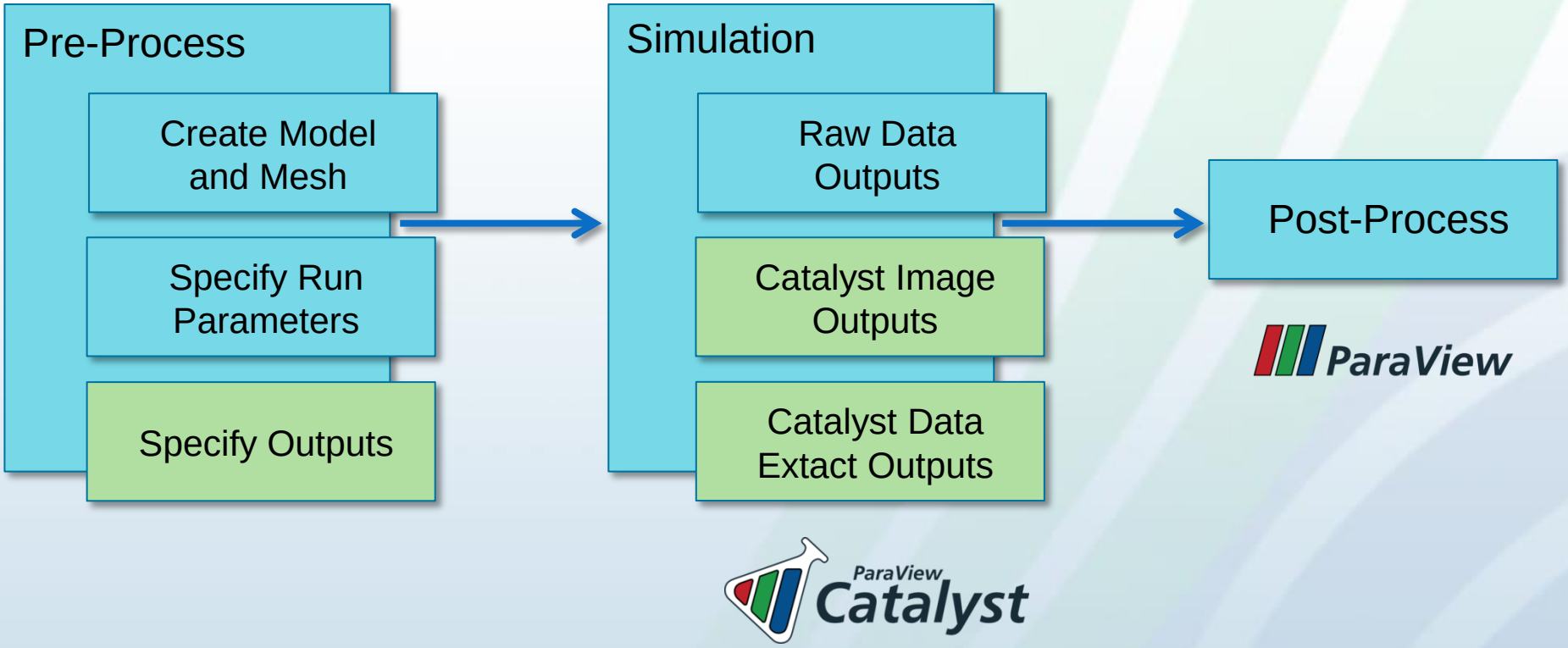
Andrew Bauer
Berk Geveci
Kitware Inc

Why *In Situ*?



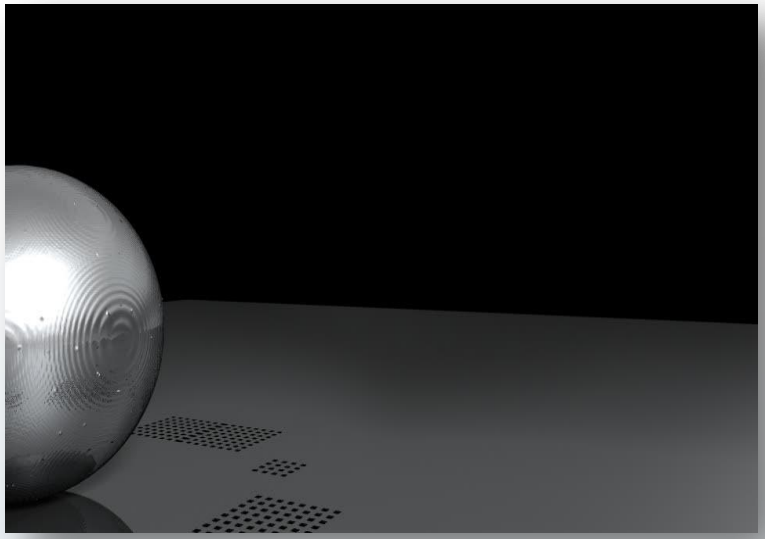
System Parameter	2011	"2018"		Factor Change
System peak	2 PF	1 EF		500
Power	6 MW	≤20 MW		3
System Memory	0.3 PB	32-64 PB		100-200
Node Performance	0.125 TF	1 TF	10 TF	8-80
Node Concurrency	12	1,000	10,000	83-830
Network BW	1.5 GB/s	100 GB/s	1,000 GB/s	66-660
System Size (nodes)	18,700	1M	100k	50
Total Concurrency	225 K	10 B	100 B	40k-400k
Storage Capacity	15 PB	300-1,000 PB		20-67
I/O BW	0.2 TB/s	20-60 TB/s		100-300

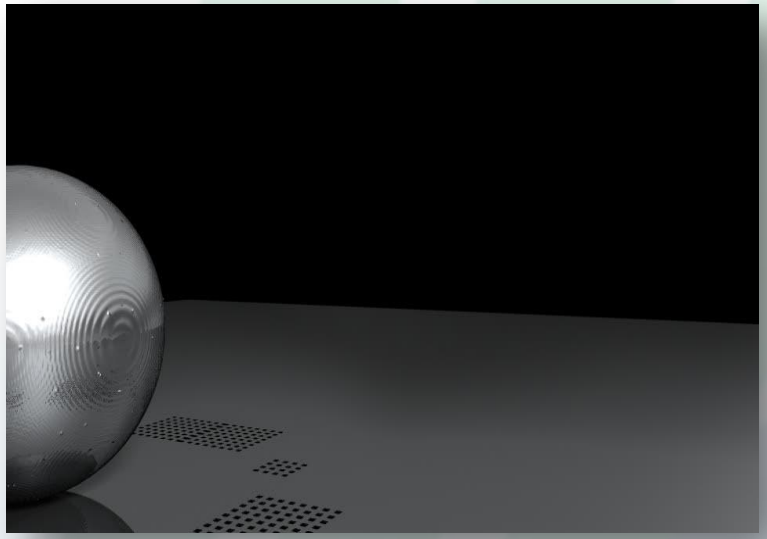
Simulation Workflow



Kitware

Access to More Data





Post-processing

In situ processing

CTH* simulation with roughly equal data stored at simulation time

Reflections and shadows added in post-processing for both examples

*A multi-material, large deformation, strong shock wave, solid mechanics code

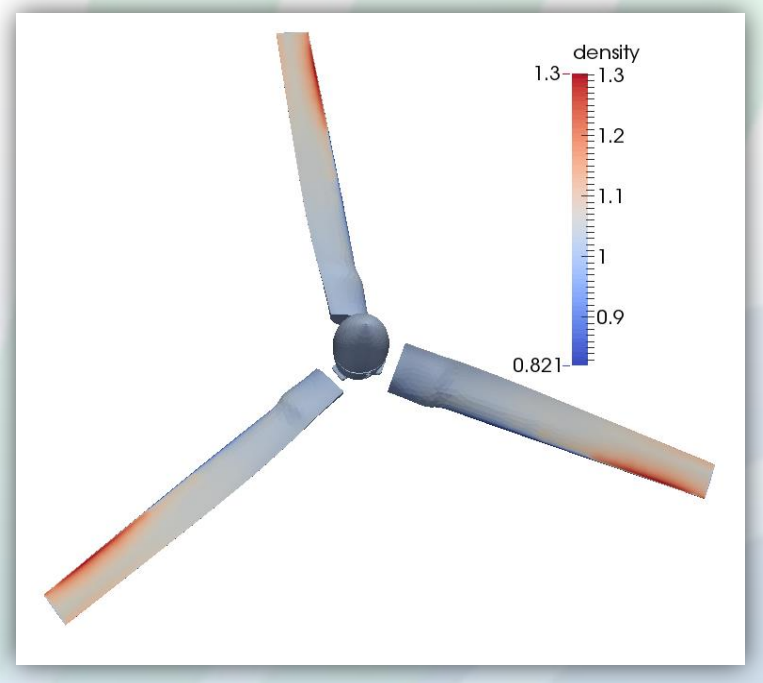
Sandia National Laboratories

Kitware

Reduced File Size

Rotorcraft simulation output size for a single time step and 32 MPI rank run

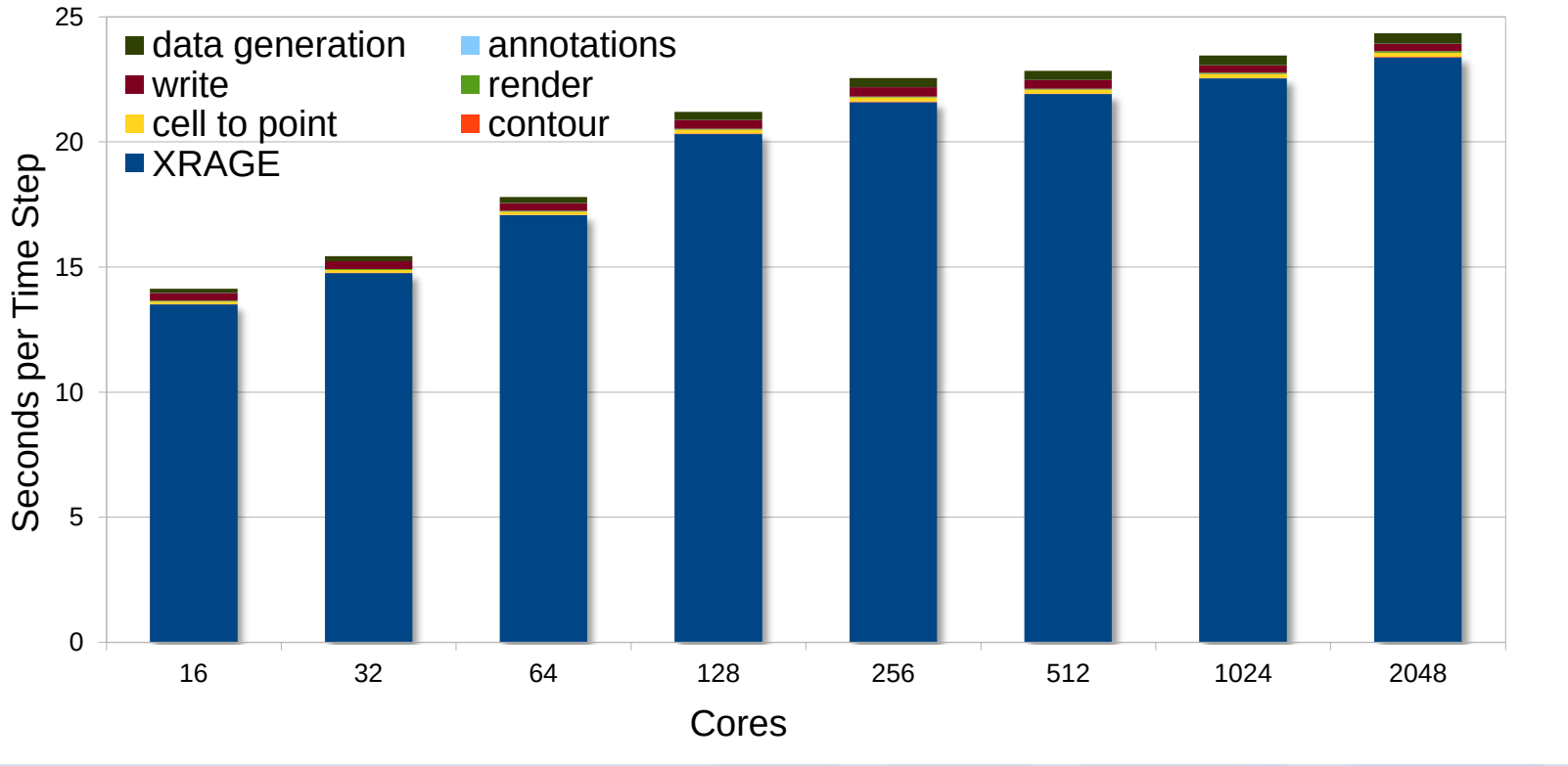
- Full data set – 448 MB
- Surface of blades – 2.8 MB
- Image – 71 KB



HPCMP CREATE-AV™ Helios (Army AFDD/AMRDEC) simulation

Kitware

Small Run-Time Overhead

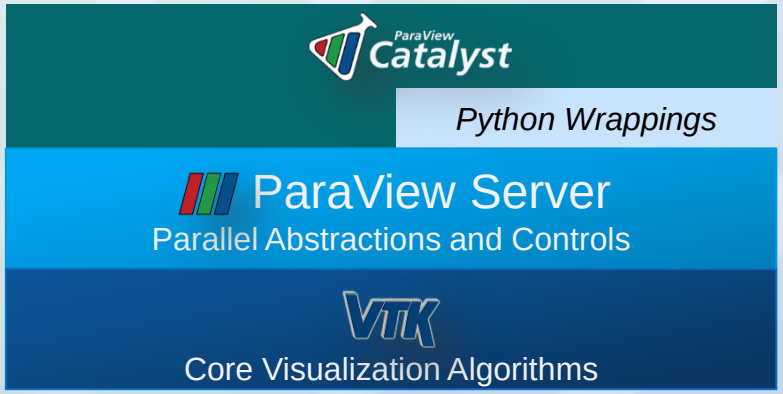


XRAGE (LANL) simulation

Kitware

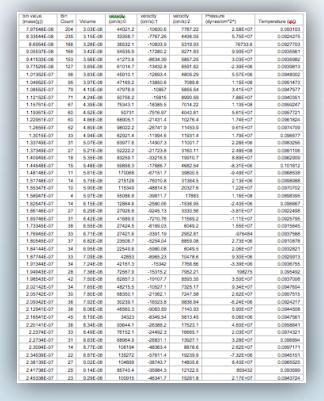
What is ParaView Catalyst?

- A set of *in situ* data analysis and visualization capabilities developed in response to current and near future data analysis challenges
  - Light-weight version of the ParaView server library that is designed to be directly embedded into parallel simulation codes
  - Available since 2010, open source – comes with ParaView
- Brings all ParaView pipeline mechanics to the simulation code
  - Data processing through filters
  - Data writers
  - Rendering and compositing
  - C/C++/Fortran & Python examples
  - Editions: only needed features in Catalyst library

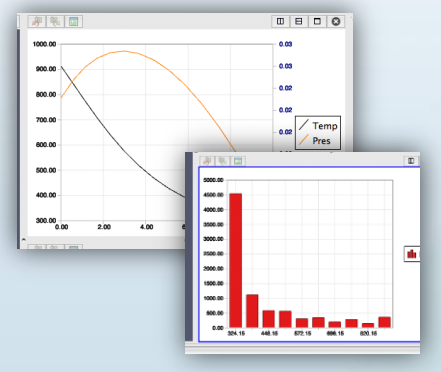


Kitware

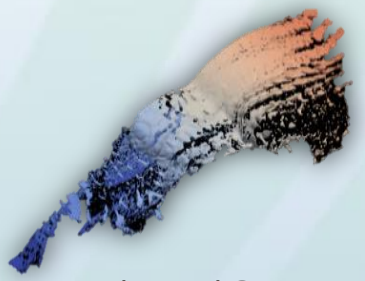
Catalyst Outputs



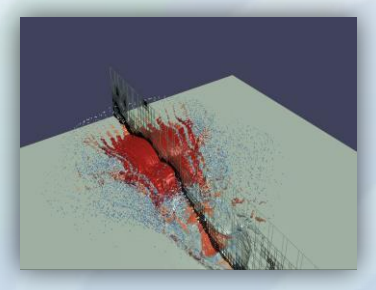
Statistics



Series Data



Polygonal Output
with Field Data



Rendered Images

Kitware

Requirements

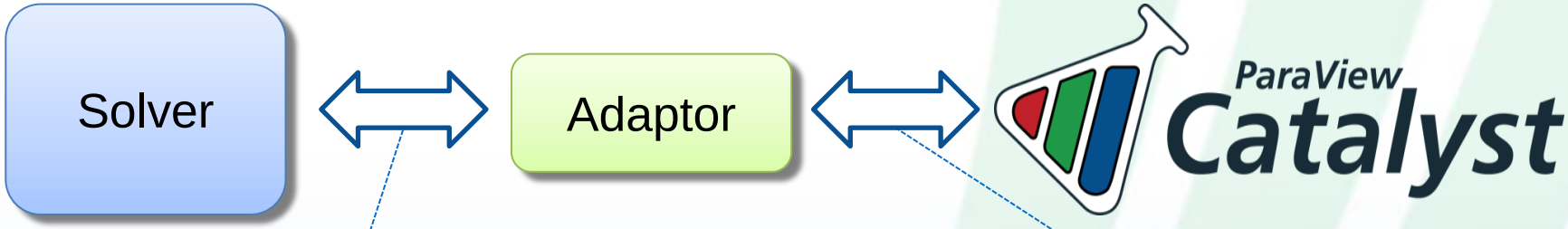
Simulation Users

- Knowledge of ParaView as a post-processing/analysis tool
  - Basic interaction with GUI Catalyst script generator plugin
  - Incremental knowledge increase to use the *in situ* tools from basic ParaView use
- Programming knowledge can be useful to extend the tools

Simulation Developers

- Pass necessary simulation data to Catalyst
- Need sufficient knowledge of both codes
  - VTK for grids and field data
  - ParaView Catalyst libraries

Interactions



- Typically only 3 calls between simulation code and adaptor
  - `Initialize()`
    - MPI communicator (optional)
    - Add analysis scripts
  - `CoProcess()`
    - Does the work (potentially)
  - `Finalize()`
- Information provided by solver to adaptor
  - Time, time step, force output
  - Grids and fields

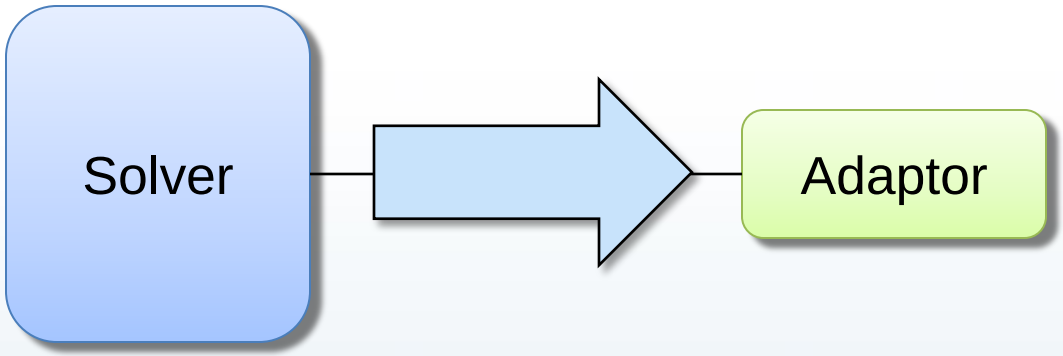
- Information provided by adaptor
  - Pipelines to execute
  - Time, time step, force output
  - Grid and fields when needed
  - MPI communicator
- Information provided by Catalyst
  - If co-processing needs to be done
  - What grids and fields are needed
- User data can be shared both ways

Adaptor Overview

Adaptor

- Creates VTK data objects representing simulation data
  - Deep copy, Shallow copy – Zero copy API
- Creates Catalyst pipelines
  - Information on how to process VTK data objects to get desired output
- Typical adaptors
  - Higher level interfaces to Catalyst to simplify vtkCPProcessor: vtkCPAdaptorAPI, CAdaptorAPI (wrapped in Fortran)
  - http://www.paraview.org/files/catalyst/docs/ParaViewCatalystUsersGuide_v2.pdf

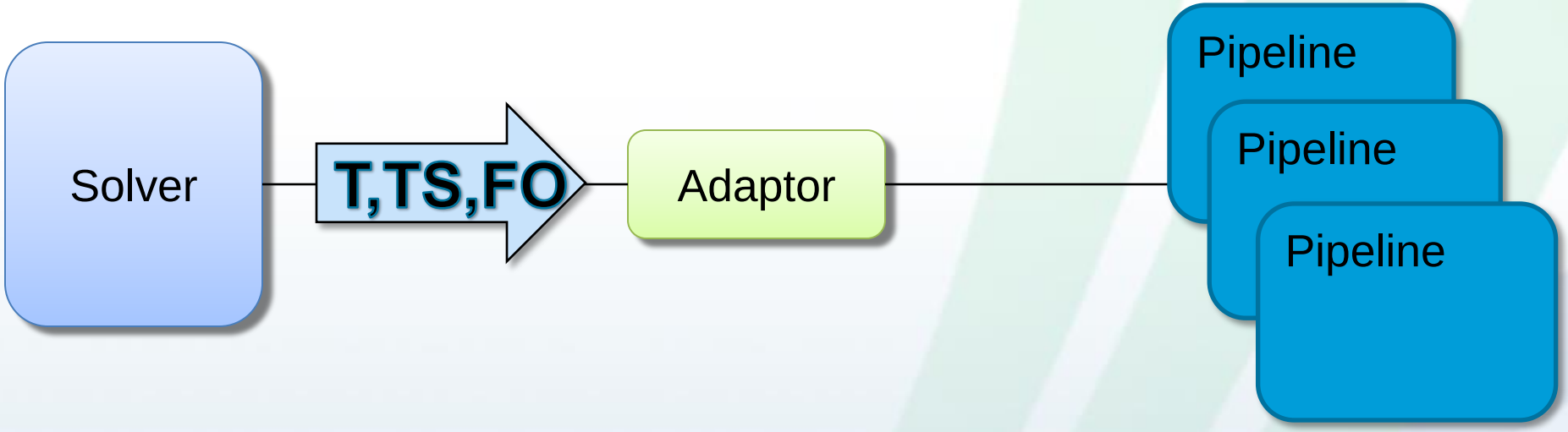
Information Flow



Initialization

- Information for creating pipelines

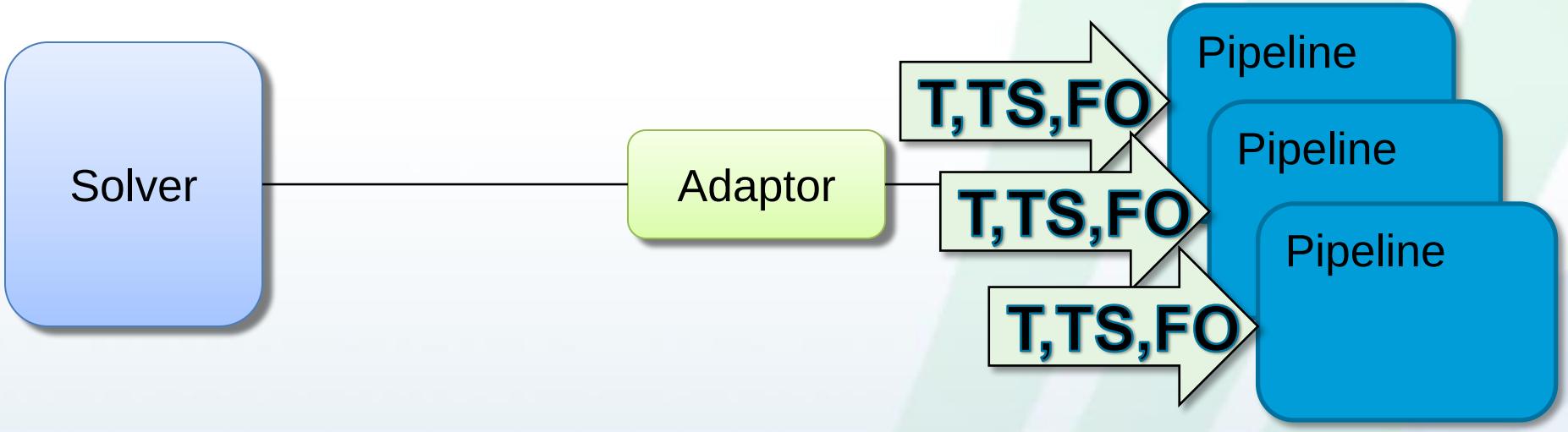
Information Flow



After simulation completes time step update

- Time, Time Step, Force Output flag
- Information for creating grid and field information

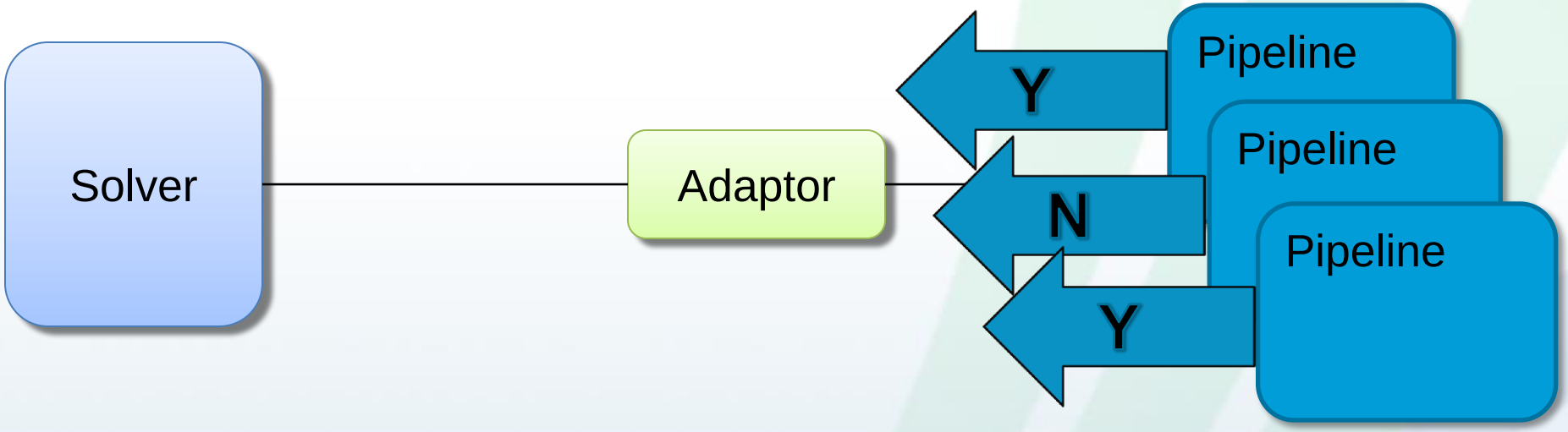
Information Flow



After simulation completes time step update

- Time, Time Step, Force Output flag are passed to each pipeline

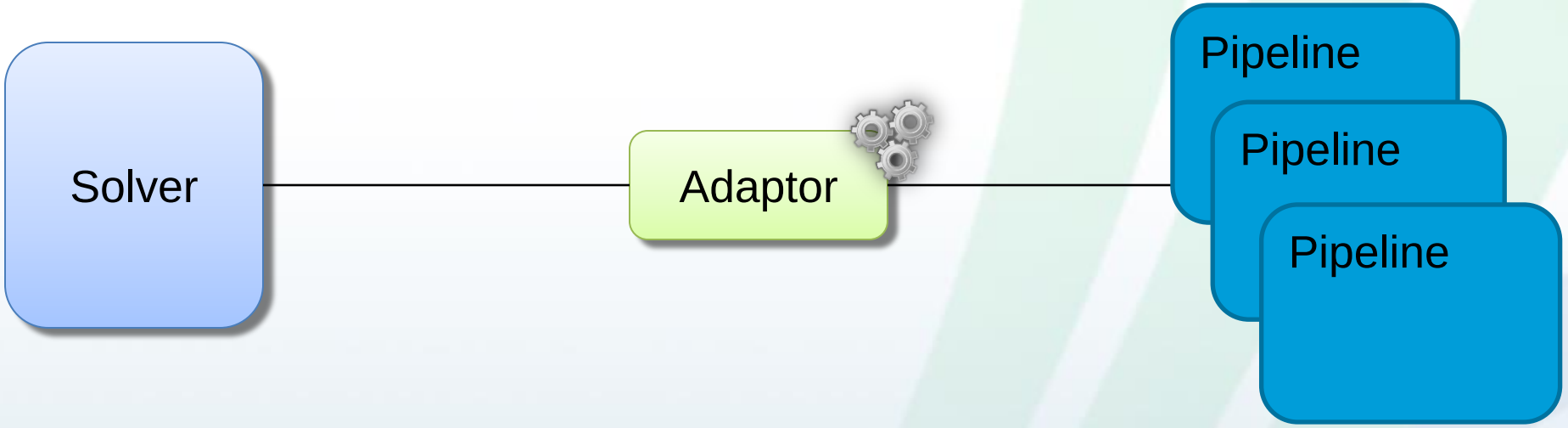
Information Flow



After simulation completes time step update

- Pipelines return if they need to be executed/updated

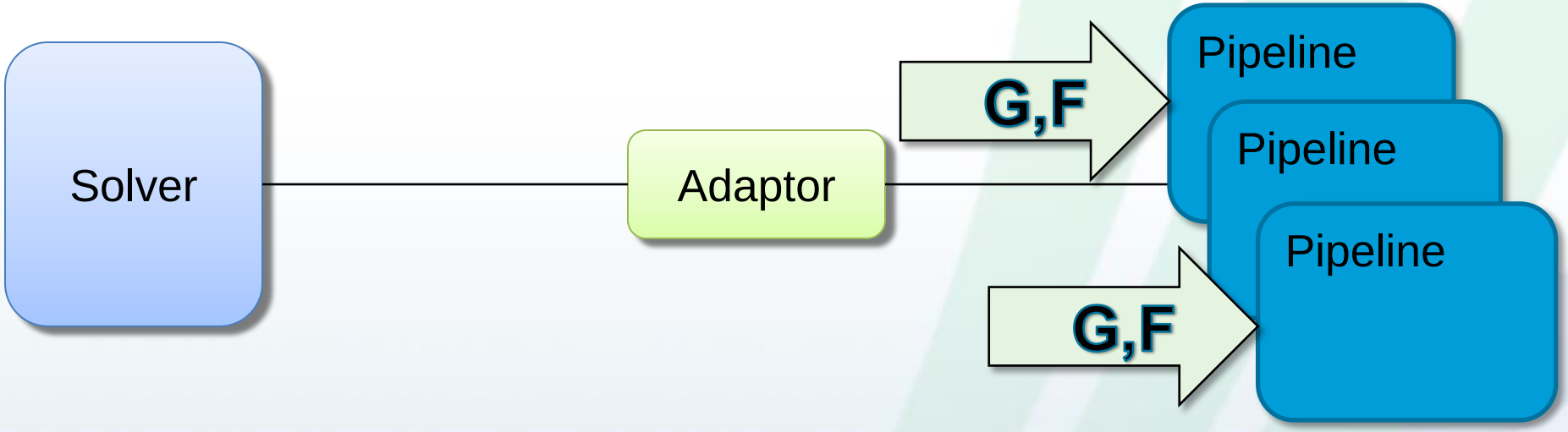
Information Flow



After simulation completes time step update

- If any pipeline needs to be executed
  - Adaptor populates VTK objects that represent grids and fields in simulation output

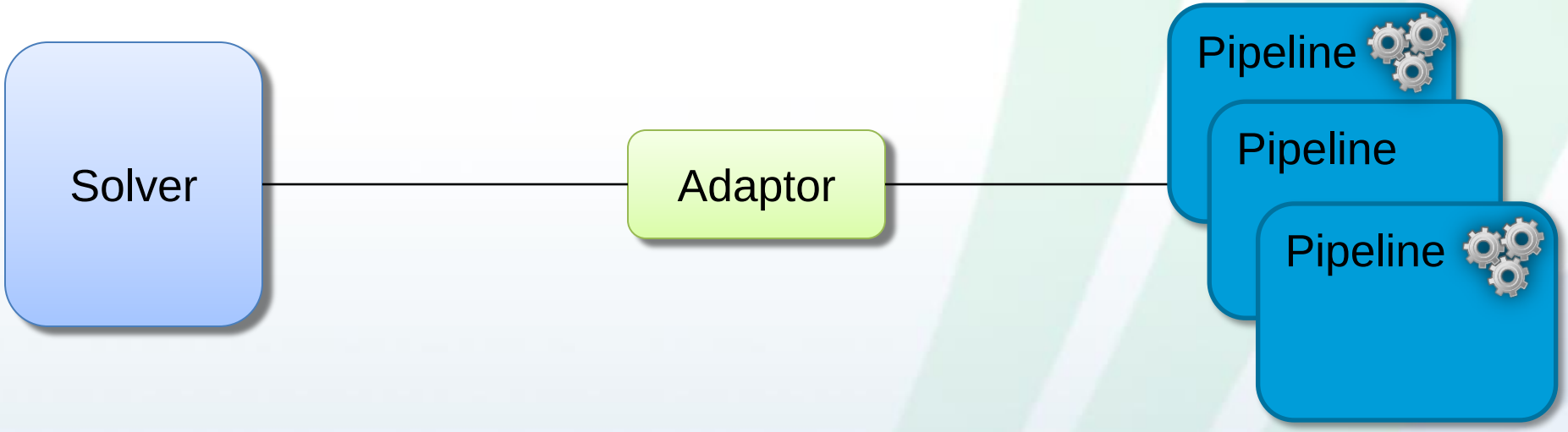
Information Flow



After simulation completes time step update

- Pass VTK data object representing Grids and Fields to pipelines that need to execute/update

Information Flow



After simulation completes time step update

- Pipelines execute and output desired information

Catalyst Pipelines

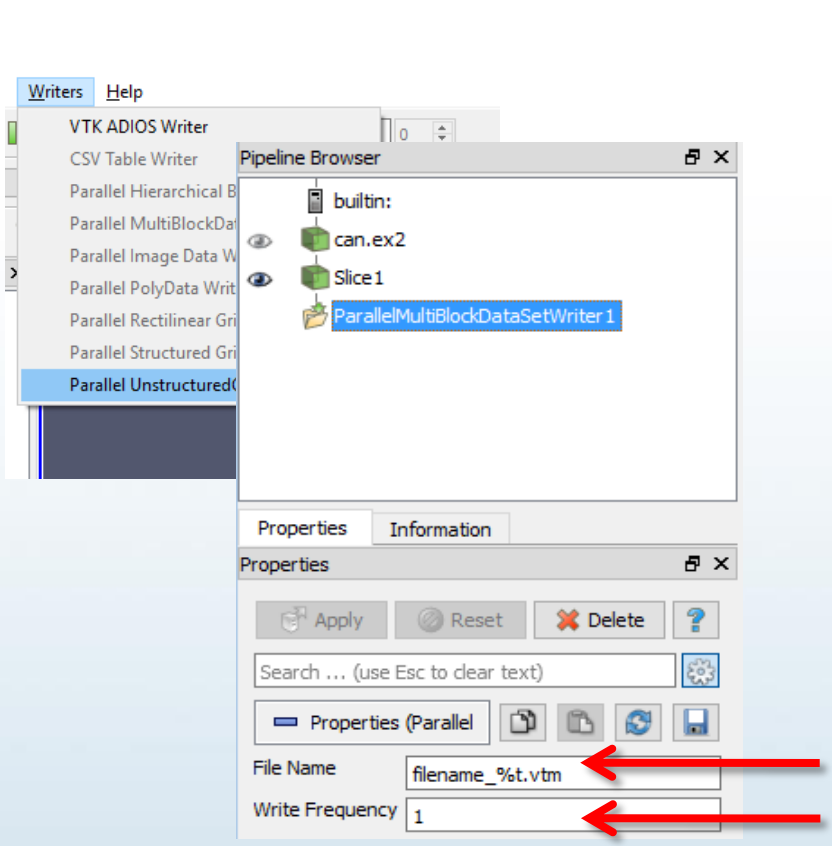
- Hard-coded pipelines
  - Users don't need ParaView knowledge in order to use Catalyst, just how to specify Catalyst output through simulation input deck
  - Developers provide input deck options to specify hard-coded pipelines
  - Can be done in Python or C++

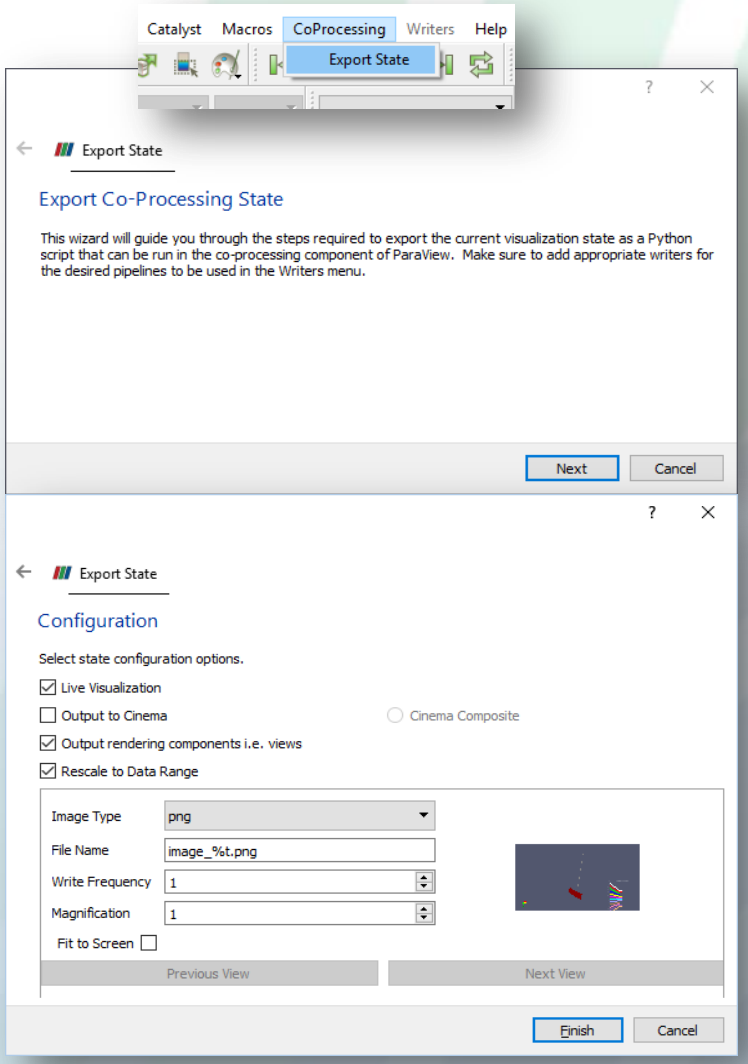
- Python scripts generated by the ParaView plugin
  - Offer better control to the user
  - Catalyst User's Guide for detailed instructions
    http://www.paraview.org/files/catalyst/docs/ParaViewCatalystUsersGuide_v2.pdf
  - Python script description
    https://blog.kitware.com/anatomy-of-a-paraview-catalyst-python-script/

Preparing *In Situ* Processing Pipelines

- Create ParaView pipelines with GUI and export script for Catalyst
  - Load “CatalystScriptGeneratorPlugin”
  - Start with a representative dataset from the simulation (eg. Step 0 on rank 0)
  - Create analysis and visualization pipelines
  - Specify extra pipeline information to tell what to output during simulation run
    - Add in data extract writers
    - Create screenshots to output
    - Both require file name and write frequency

Catalyst Plugin Features

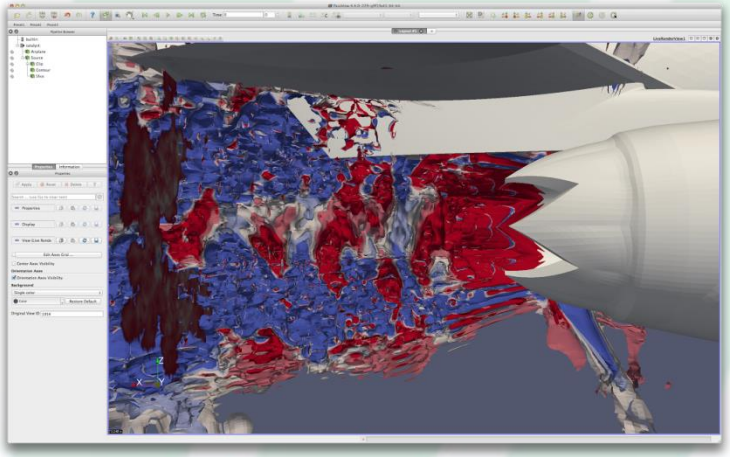


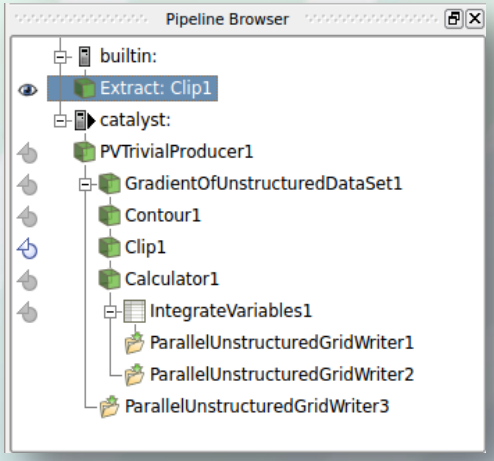


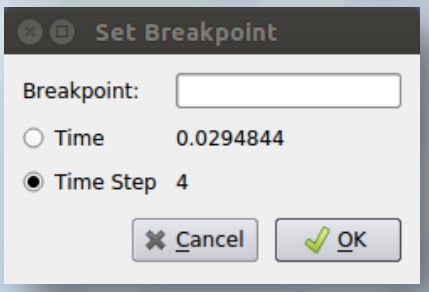
Kitware

Live *In Situ*

- Provides functionality for interacting with simulation data during simulation run
- Connect ParaView client to a running simulation with Live Visualization enabled
- Only transfer requested data from simulation nodes to client/data server nodes - then perform “local” processing
- Debugging features
  - Pause
  - Breakpoints catalyst:



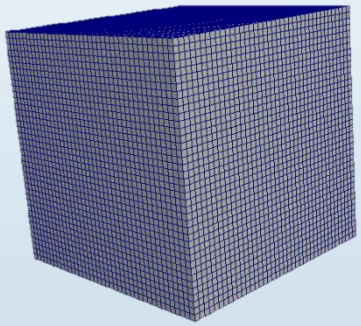




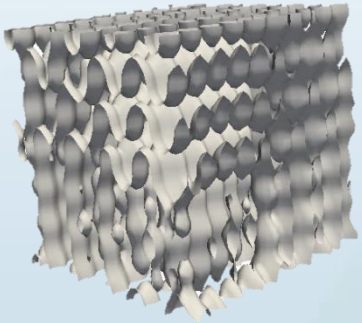
Kitware

In Transit/Hybrid Workflow

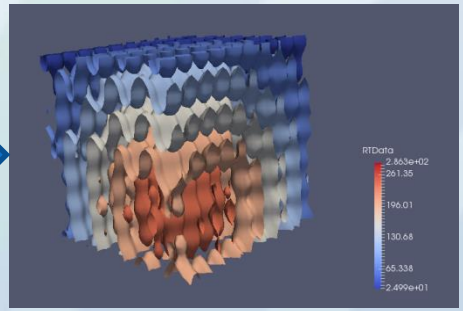
- Some analysis and viz operations may not work efficiently at the same parallelism that the simulation code is run at
  - Solution: run these things on a smaller set of processes somewhere “else”
  - Trying Cori@NERSC Burst Buffer for transport mechanism



448 MB/TS



2.8 MB/TS

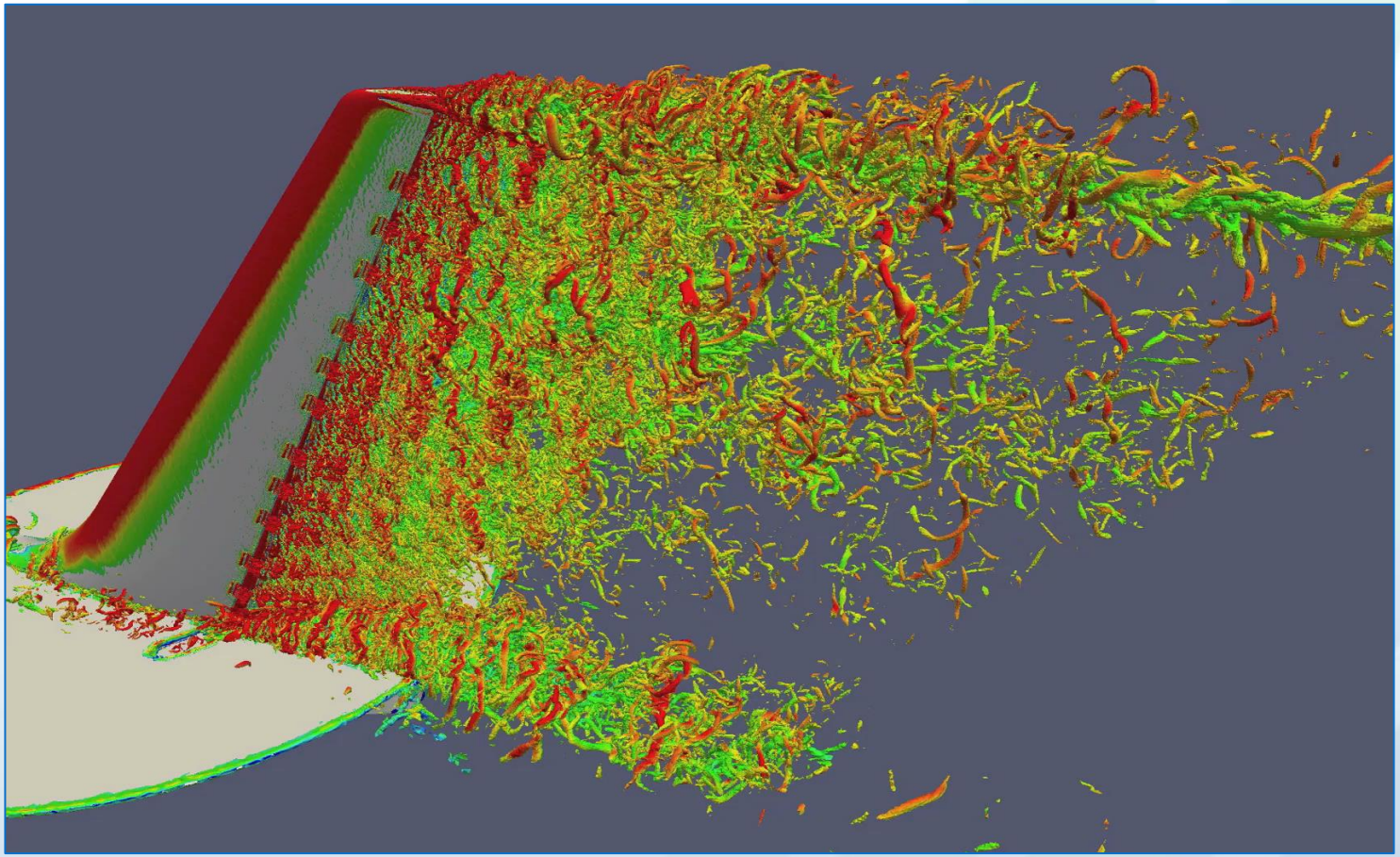


71 KB/TS

*Image size independent of data set size

Kitware

Going to Exascale



PHASTA running with 256K MPI ranks
Performed on MIRA@Argonne (IBM BG/Q) - 2014

Going to Exascale 2

- PHASTA run with 1M MPI ranks on MIRA@Argonne (IBM BG/Q) - 2016
- Catalyst edition for reduced library size
- *In situ* times
  - 1.9 second initialization
  - 5.6 seconds/per slice operation and image output

M. Rasquin, C. Smith, K. Chitale, S. Seol, B. Matthews, J. Martin, O. Sahni, R. Loy, M. Shephard, and K. Jansen, "**Scalable fully implicit finite element flow solver with application to high-fidelity flow control simulations on a realistic wing design**," Computing in Science and Engineering, vol. 16, no. 6, pp. 13–21, 2014.

Future Work

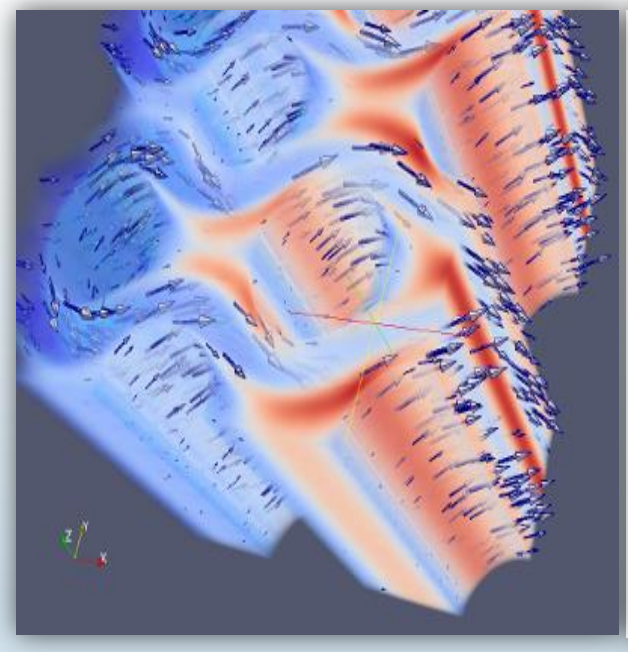
- At scale file IO for data extracts
  - Reduced IO amount but increased complexity due to non-load-balanced and dynamic output size and location
- In transit workflows (Burst Buffers, ADIOS, etc)
- Catalyst editions with even smaller library sizes
- Advanced Cinema options 

- Parallelism appropriate with simulation code's parallelism (e.g. MPI, OpenMP, CUDA, threads, VTK-m, vtkSMPTools, etc.)
- Steering features
- New SENSEI framework – SC16 paper 

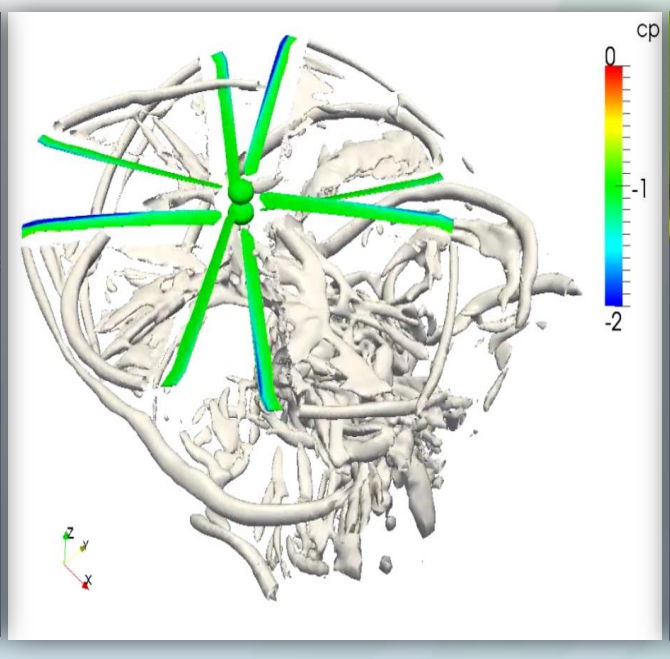
ParaView Catalyst

www.paraview.org/in-situ

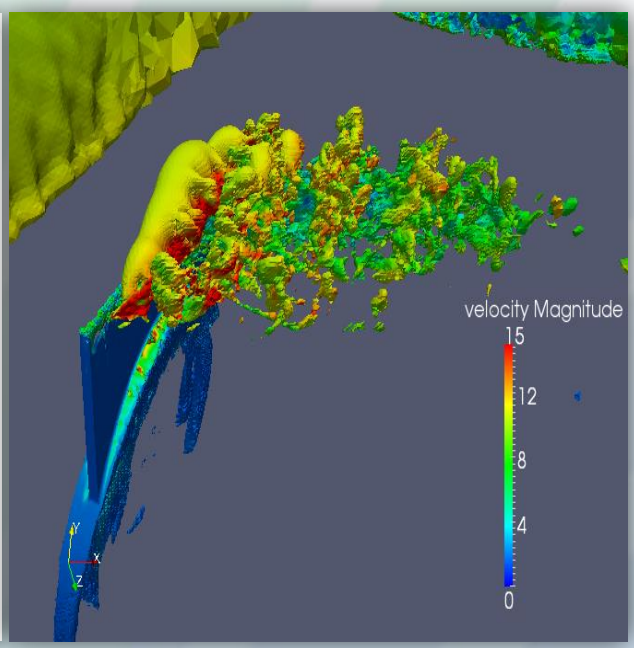
Thanks for your attention



Code Saturne
EDF



Helios
Army Aeroflighdynamics Directorate



Phasta
UC-Boulder

Kitware

Acknowledgments

This work was mainly supported by

Sandia National Laboratories

Los Alamos National Laboratory

Army SBIRs

The ELCI and AVIDO projects

French FSN (“Fond pour la Société Numérique”) cooperative projects that associates academic and industrial partners to design and provide software environment for very high performance computing

Kitware