VizSchema - Visualization Interface for Scientific Data

Svetlana Shasharina, John Cary,
Seth Veitzer, Paul Hamill,
Scott Kruger, Marc Durant, David Alexander

Tech-X Corporation
Boulder, CO
Work funded by US DOE and Tech-X

Tech-X is small-business company
www.txcorp.com

- **Funded in 1994, Boulder, Colorado, USA**
- **60 employees (physicists, mathematicians, computer scientists and developers)**
- **DOE, DOD, NASA funding and commercial revenue**
- **Applications**
  - Accelerator physics
  - Fusion modeling
  - Nanotechnology
- **Computer science**
  - Grids
  - Data Distribution Service
  - CORBA
  - Multicore computing
  - Semantic Web
  - Visualization

Boulder is different from Algarve :-)

Outline

- Motivation
- VizSchema principles and some details
- Examples of visualization
- Conclusions and future directions

Scientific data and visualization tools are heterogeneous

- **Multiple data formats are in use (HDF5, NetCDF, MDSPlus, custom formats)**
- **Multiple viz tools are in use (VisIt, VTK, IDL, AVS/Express etc)**
- **Settling on one format (HDF5) is not enough:**
  - HDF5 consists of groups and datasets adorned by attributes (metadata), hierarchical
  - One can organize them in any order and not use metadata at all
  - One cannot know what each dataset and group means and how to interpret data
    - What is supposed to be visualized?
    - How to match data to its mesh?
    - How one interprets N-dimensional cube of data?
    - How does a viz tool get what it wants (data ordered as 1d arrays, C-ordering, hints how to build a mesh…)?
- **Need metadata to teach viz tools to recognize and build what they need**

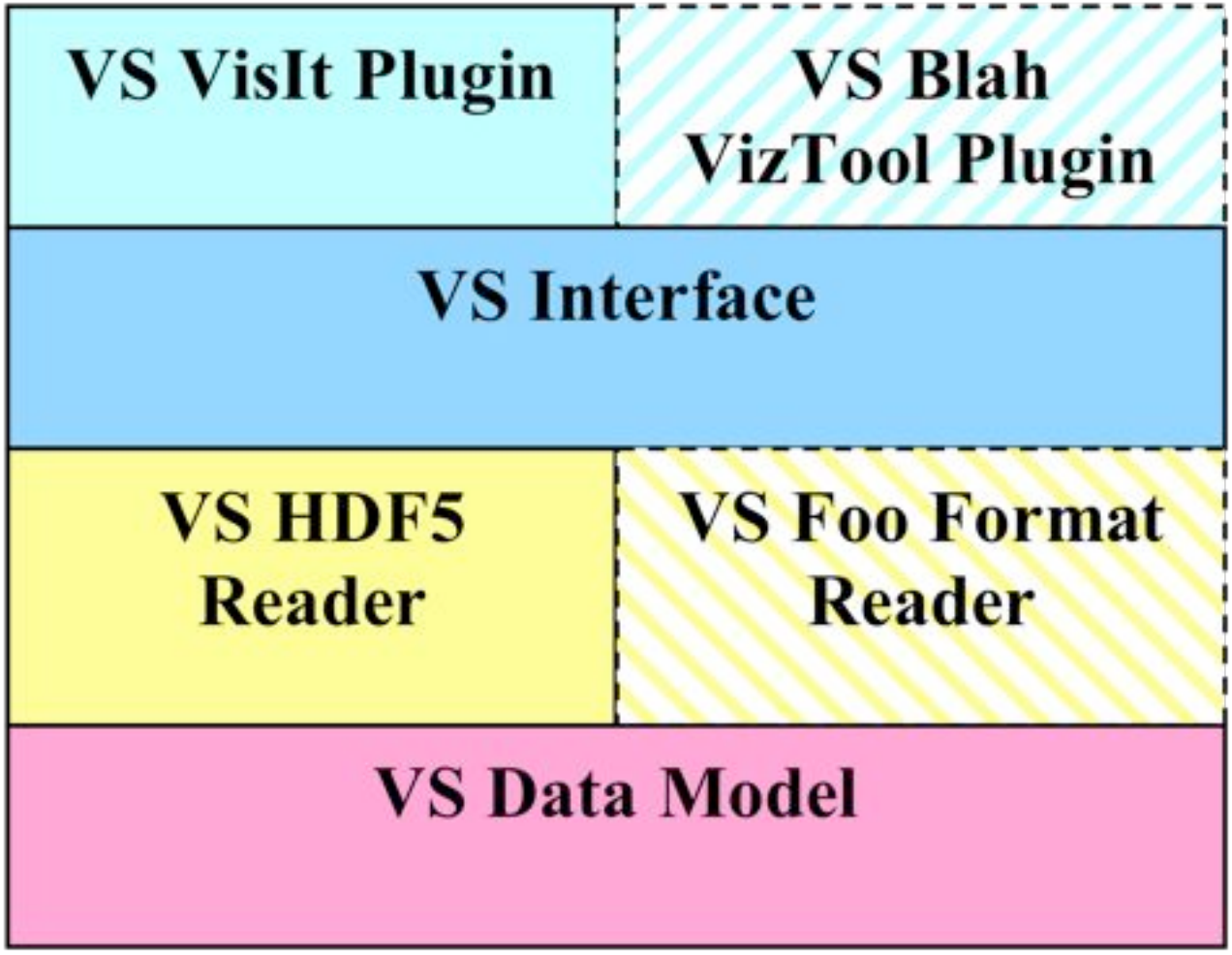
External and internal metadata have their uses

- **Some application have outputs with the same structure**
  - External metadata can be used as only one description file per an application (find mesh in dataset /foo/blah, for example)
  - An XML is a natural choice for mapping real data to visualization concepts
  - But
    - it is an extra dependency on XML tools
    - Some application teams do not “like” XML :-(
- **Some application changes the structure of outputs depending on simulation**
  - Cannot count on one XML instance, so we do not count on XML file at all-> One needs to have correct metadata in data itself
  - This was our choice as we worked with such capricious applications

Vizschema's path

- Define what is needed for visualization (full at any moment and minimal)
- Started with XML but it barfed (no static descriptions and no acceptance from our application scientists)
- Switched to defining HDF5 markup as metadata
- Create viz-tool-agnostic but format specific data readers (get metadata and data needed for viz). We did C++ HDF5 reader.
- Implement modules in viz tools using the readers API. We did VisIt plugin.

VizSchema offers flexibility and extensibility



Standardization is hard to impose, but works in collaborative projects

- **SciDAC - Scientific Discovery through Advanced Computing**
  - DOE program (about 10 years old)
  - Mixes multiple physics, applied math and computer scientists groups in each domain-specific project
  - Each project is 5 years old and has ~$2M/year funding
- **Such scope allows and imposes standardization**
- **Examples:**
  - FACETS (Framework Application for Core Edge Transport Simulations)
  - COMPASS (accelerator physics)
  - VACET (Visualization and Analysis CET: VisIt tool)
  - Tech-X is involved in FACETS and COMPASS and produce VizSchema

VizSchema Principles

- **User can name all datasets and groups arbitrarily**
- **Minimal markup**
- **Use of only attributes for VS markup**
  - All such attributes start with Vs
  - If an attribute refers to a named node, one can use a short of fully qualified name (path from the top)
    - The entity in the closest node will be used
- **Attributes can be created**
  - at I/O (using C or Fortran HDF5 API) or
  - added after the file was created
    - PyTables are very nice:

```
h5file = tables.openFile(fileName, mode='a')
dataSet = h5file.getNode("/" + dataSetName)
dataSet.attrs.vsType = "variable"
h5file.close()
```

VS data classification

- **"Variable": data to be visualized and using mesh defined external:**
  - One needs to have an attribute to point to the used mesh
  - Electric and magnetic fields in PIC simulations
- **"VariableWithMesh": data mixes the data to be visualized with the spatial information mixed in**
  - Particles (x, y, z, p_x, p_y, p_z) or (x, p_x, y, p_y, z_pz)
  - One need attributes to help extract mesh from this mix
- **"Mesh": spatial information**
  - Needs attributes helping to build all points from whatever is given
- **By default, all data is single domain (SD)**
- **Can be multiple domains (MD) but needs more metadata to stitch all together**

Variables live on meshes externally defined

- **Variables live in datasets:**

```
Dataset "E" {
        Att vsType = "variable"
        Att vsMesh = "mycartgrid
        DATASPACE (n0, n2, n3, 3)
        Att vsCentering = "zonal" //nodal is default
}//3-component var on 3d mesh, E_0, E_1 and E_3 defined
```

- **One can also specify ordering using vsIndexOrder attribute:**

```
"compMinorC" [ix][iy][iz][ic] - default
"compMinorF" [iz][iy][ix][ic]
"compMajorC" [ic][ix][iy][iz] – not supported yet
"compMajorF" [ic][iz][iy][ix] – not supported yet
```

Component minor the component index appears last;
C – the first index is slowest

Meshes metadata depend on meshes kind

```
Group "mycartgrid" {
  Att vsType = "mesh"
  Att vsKind = "uniform"
  Att vsStartCell = [0, 0, 0]
  Att vsNumCells = [200, 200, 104]
  Att vsLowerBounds = [-2.5, -2.5, -1.3]
  Att vsUpperBounds = [2.5, 2.5, 1.3]
}
Dataset "mesh3dstruct" {
  Att vsType = "mesh"
  Att vsKind = "structured"
  DATASPACE (n0,n1,n2,3)
}
```

Variables with mesh contain their coordinates: two ways to go


```
Dataset ”vpelectrons" {
  Att vsType = "variableWithMesh”
  Att vsNumSpatialDims = 3
}
Dataset “synelectrons” {
  Att vsType = “variableWithMesh
  Att vsSpatialIndeces = “0,2,4”
} // Not implemented yet
```

One can create derived variables

- If prime variable E with 3 components is defined somewhere, one can define new scalars

```
Group anygroupname {
 Att vsType = "variableDefinition"
 Att vsDefinition = "elecEnergyDensity = 0.5*8.854e-12*(E_0*E_0
  + E_1*E_1 + E_2*E_2)"
}
```

- If prime variable e (var with mesh) is defined somewhere

```
Group anyname {
  Att vsType = "variableDefinition"
  Att vsDefinition = "velocity" = {e_3, e_4, e_5}
}
```

- One can pass a string using VisIt expressions rules

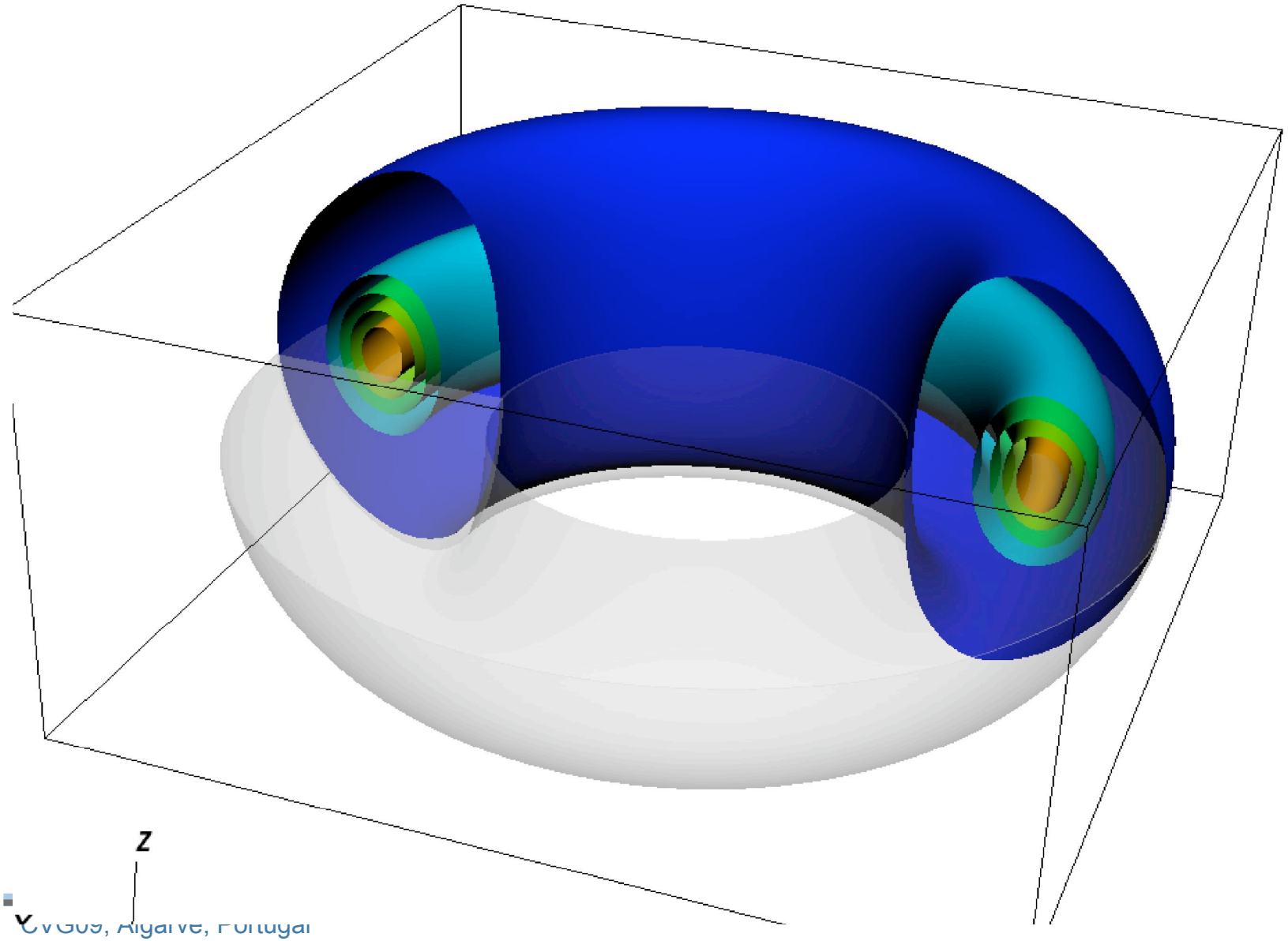
One can have variables defined on multiple domains


```
Dataset "privMesh" {
  Att vsType = "mesh"
  Att vsKind = "structured"
  Att vsMD = "edgeMesh"
}
Dataset "solMesh" {
  Att vsType = "mesh"
  Att vsKind = "structured"
  Att vsMD = "edgeMesh"
}
Dataset "psiPriv" {
  Att vsType = "variable"
  Att vsMesh = "privMesh"
  Att vsMD = "psi"
}
Dataset "psiSol" {
  Att vsType = "variable"
  Att vsMesh = "solMesh"
  Att vsMD = "psi"
}
```

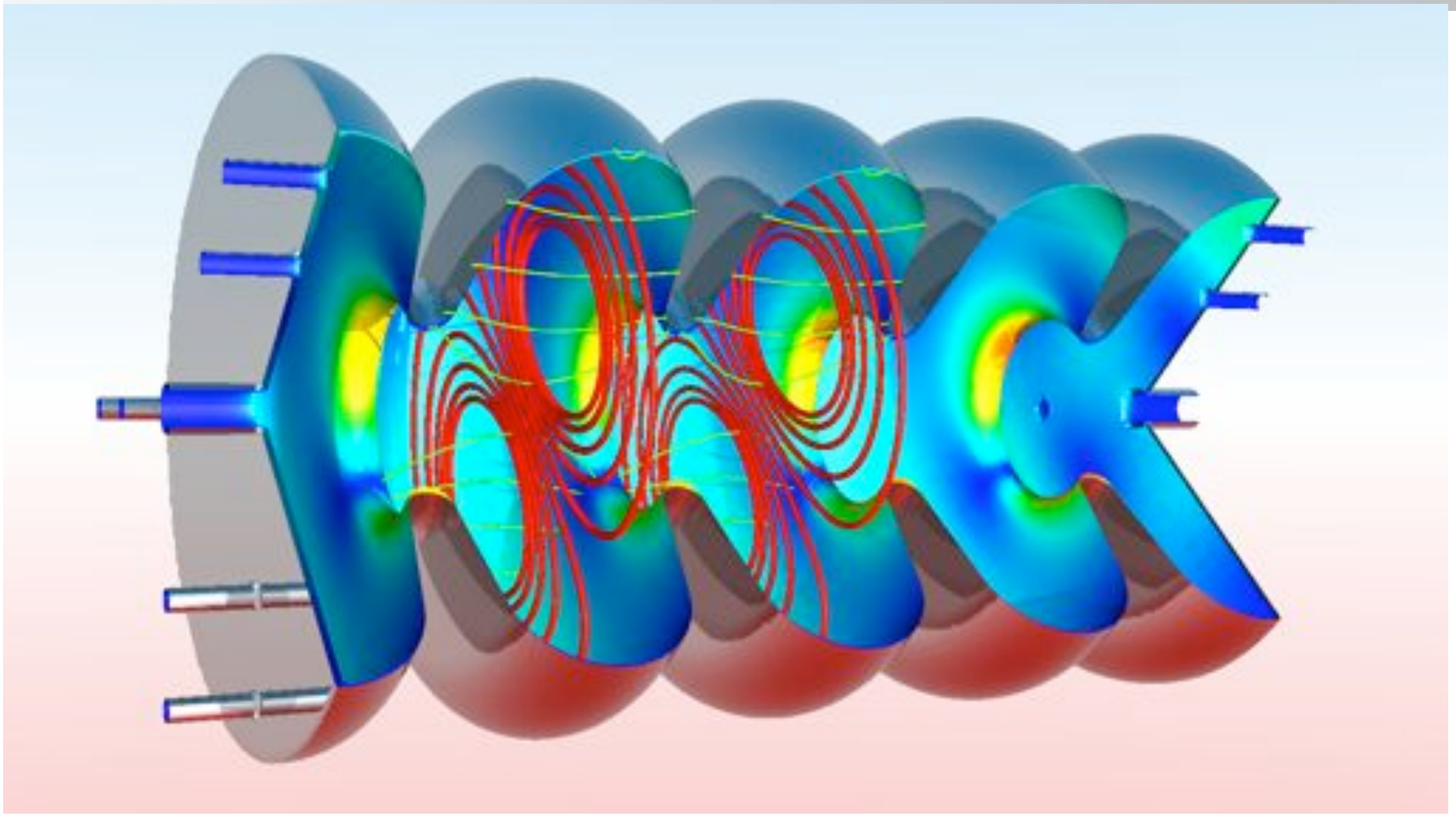
VizSchema is used in many codes

- NIMROD
- FACETS
- UEDGE
- VORPAL
- MODAVE
- PolySwift++

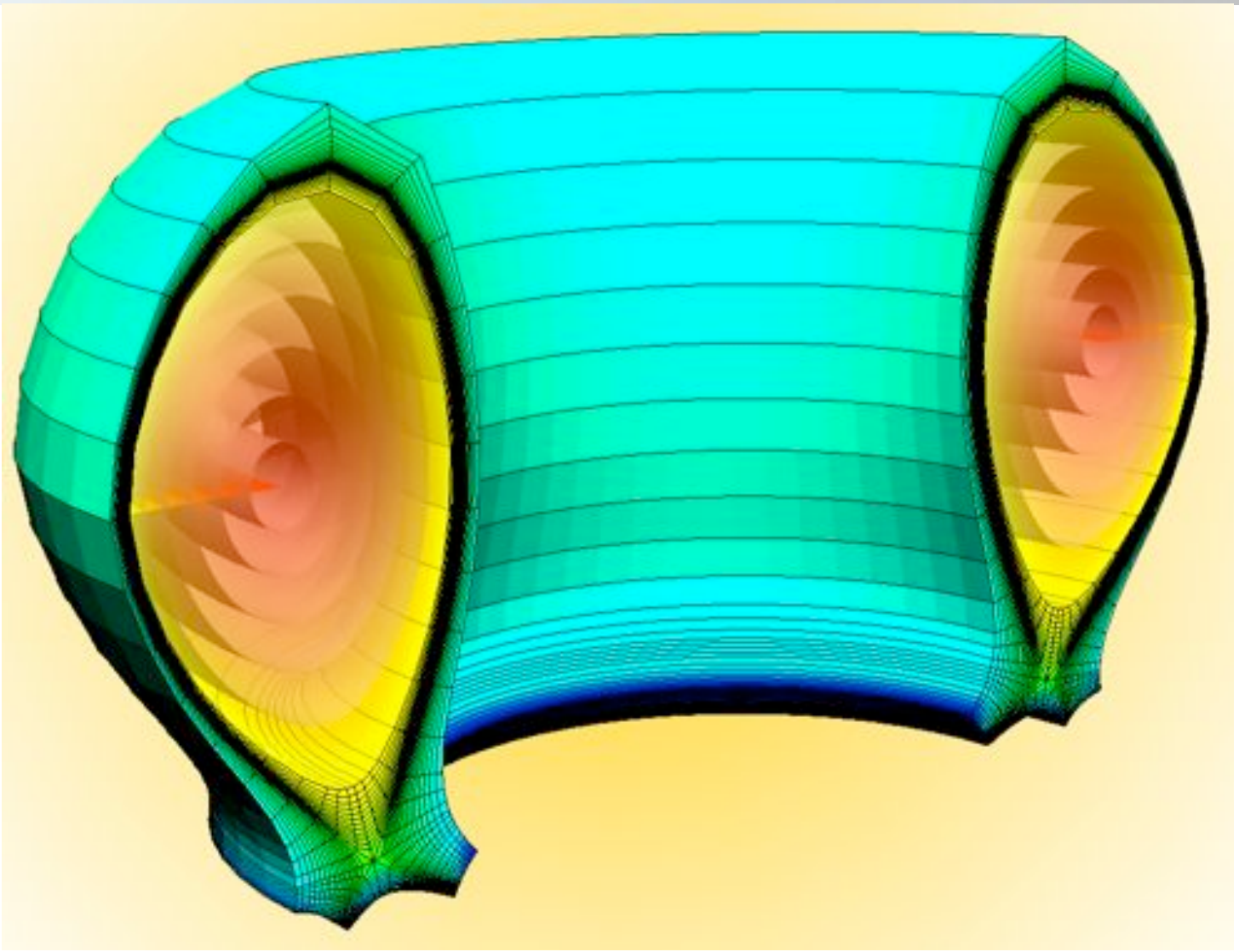
NIMROD



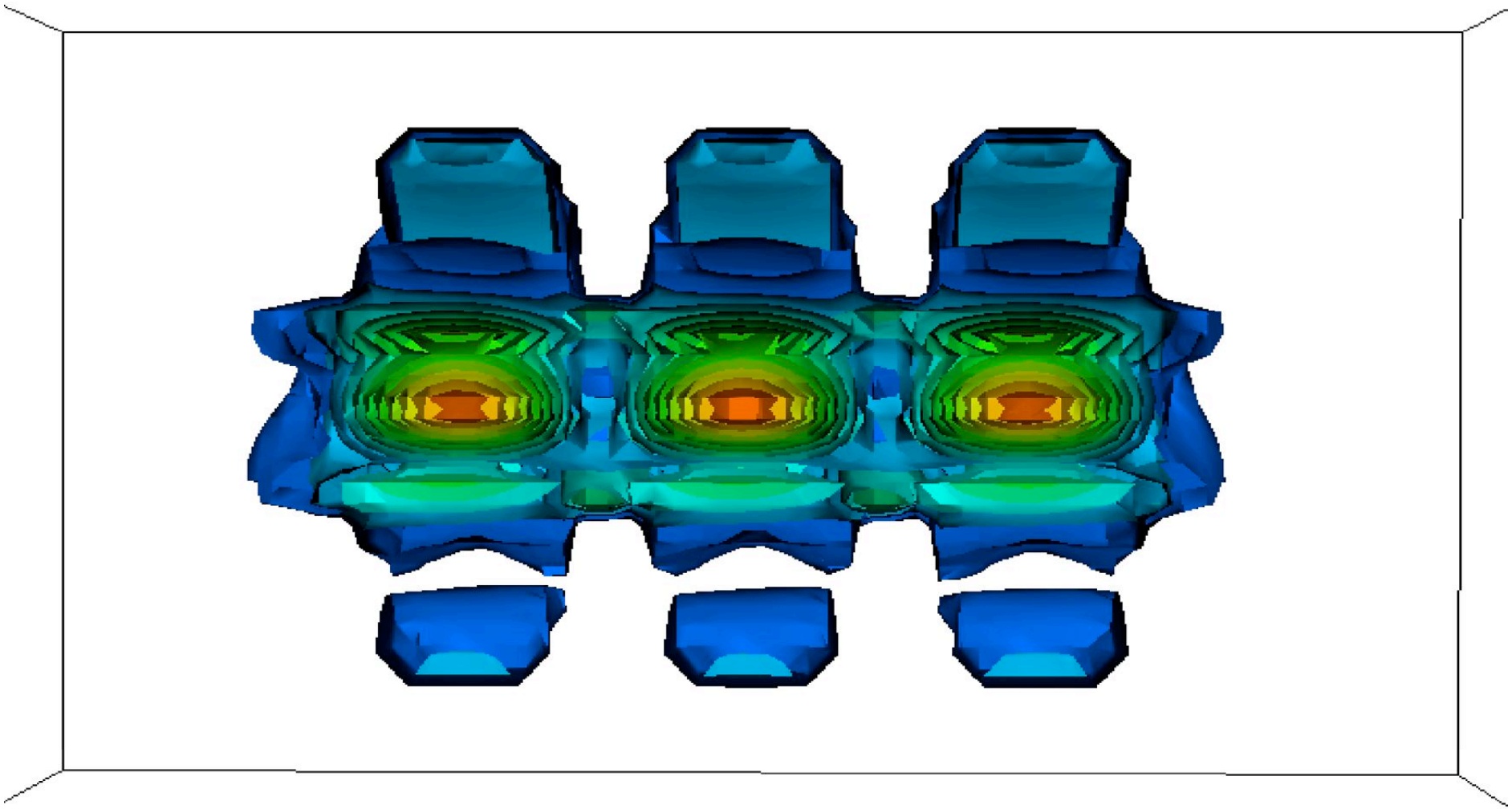
VORPAL



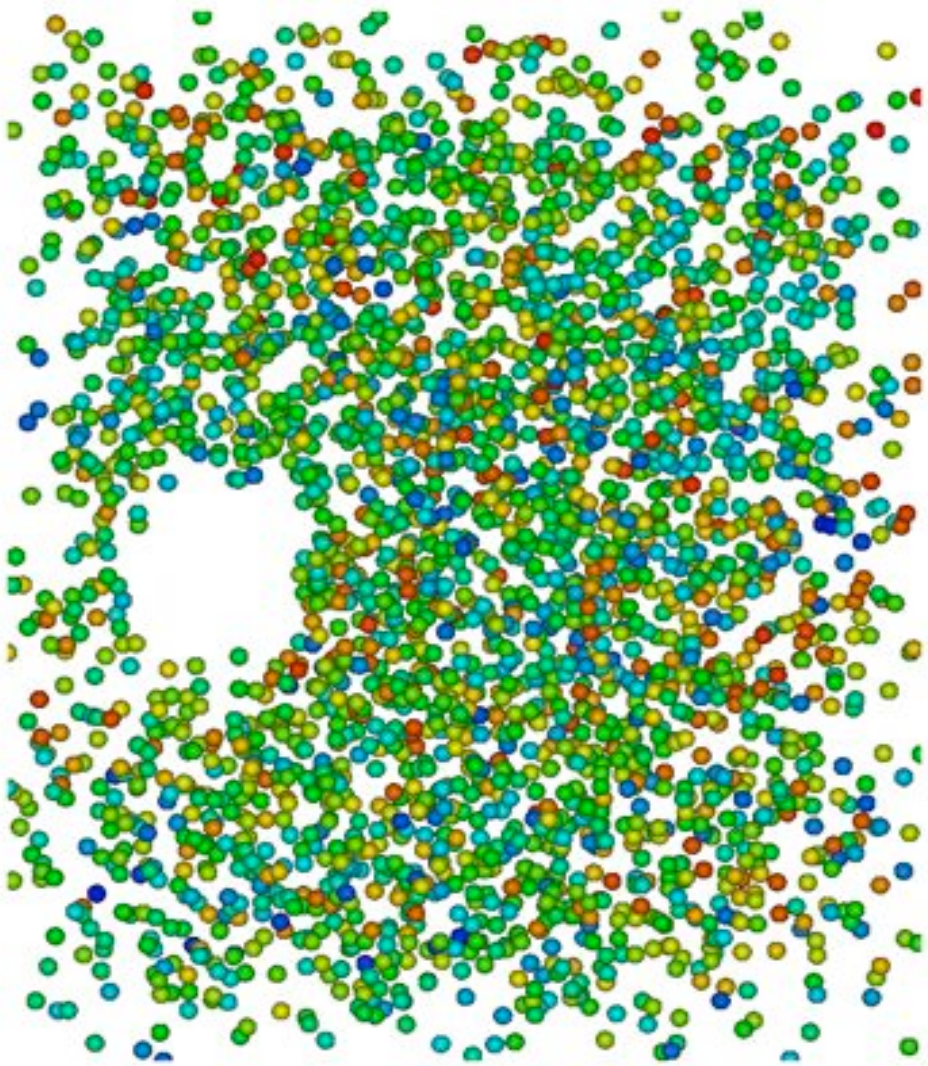
Core and edge data can be visualized in one viz (FACETS)



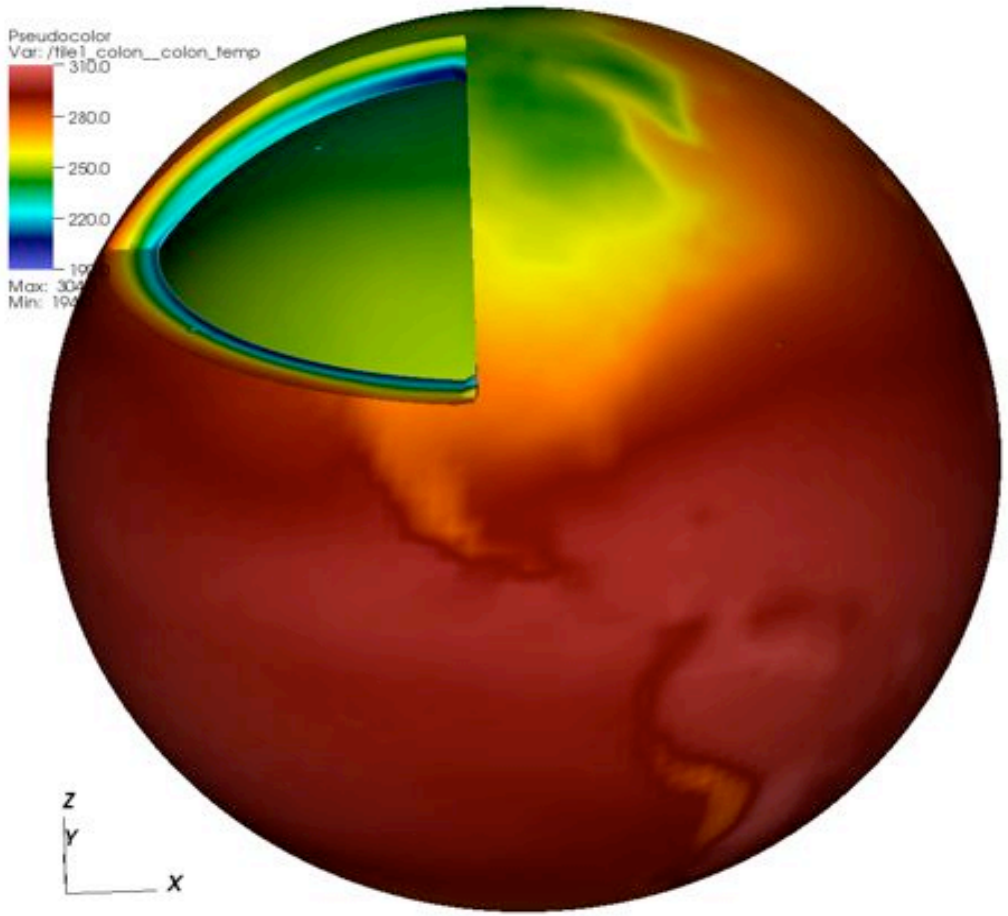
VORPAL Fields



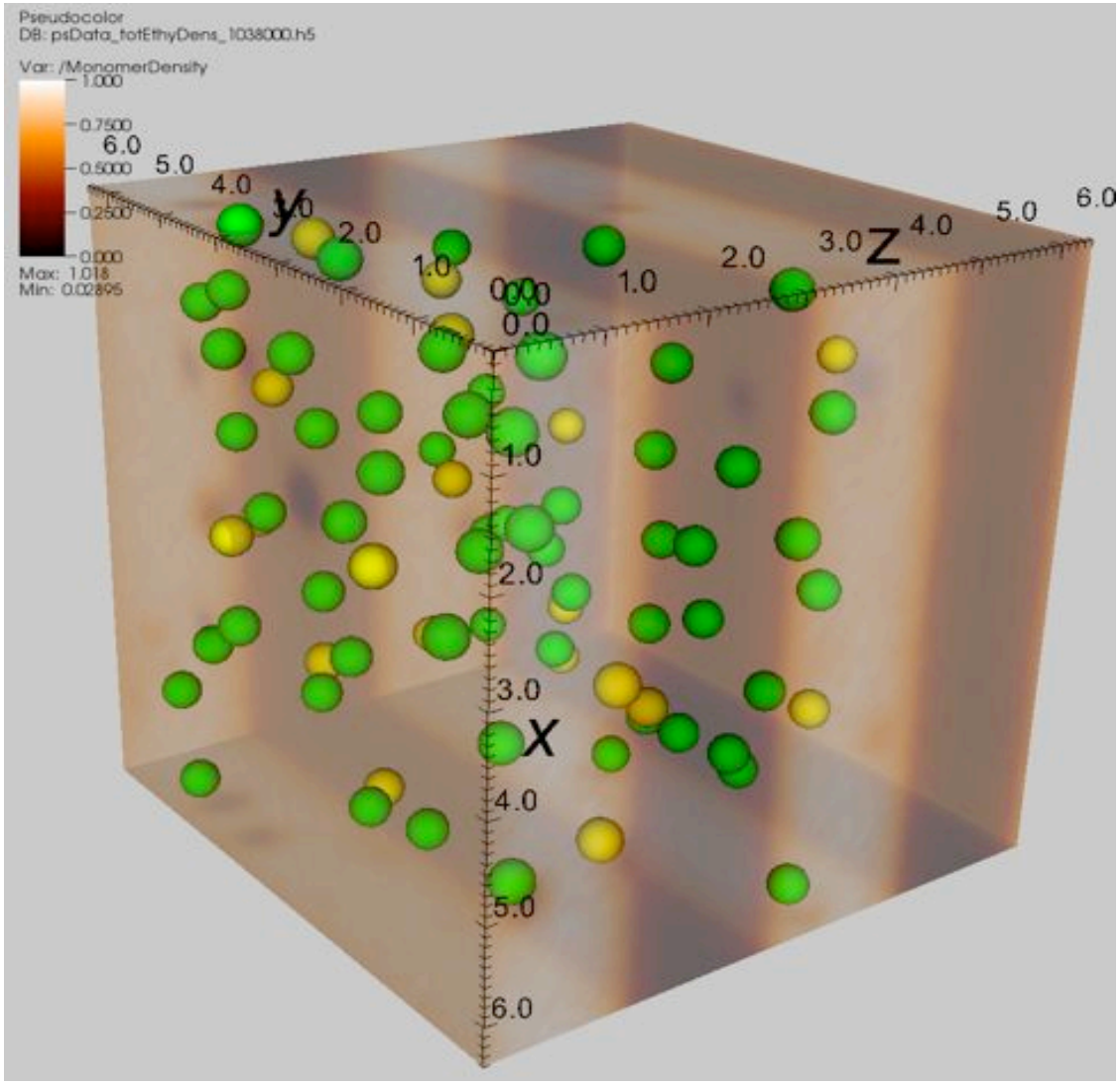
VORPAL's Particles



MODAVE


Pseudocolor
Var: /tile1_colon__colon_temp
310.0
280.0
250.0
220.0
Z
Y
X

PolySwift++


Pseudocolor
DB: psData_totEthyDens_1038000.h5
Var: /MonomerDensity
1.000
0.7500
0.5000
0.2500
0.000
Max: 1.018
Min: 0.02895
Y
Z
X
0.0
1.0
2.0
3.0
4.0
5.0
6.0

Conclusions

- **Standardization is possible**
  - Common goals
  - Value added
  - Agreement to accommodate new requirements
  - Centralized development was key, though
  - Regression tests is a must (we diff png files)
- **Standardization allowed combining data from different applications in one viz (see movie)**
- **Development of tools of adding markup after files were generated was very useful**
  - Some file are expensive to regenerate

Future directions

- **Parallelization of the VS VisIt plugin**
- **Extending to other formats**
  - NetCDF
  - MDSplus
  - More meshes and standardization of kinds and attributes
- **Subselection for very large data**

Pointers

- **Wiki:**
  - https://ice.txcorp.com/trac/vizschema/wiki/WikiStart
- **Dependencies:**
  - VisIt (1.11.2)
  - HDF5 (1.8.2)
- **Email questions and requests for the code to: sveta@txcorp.com**