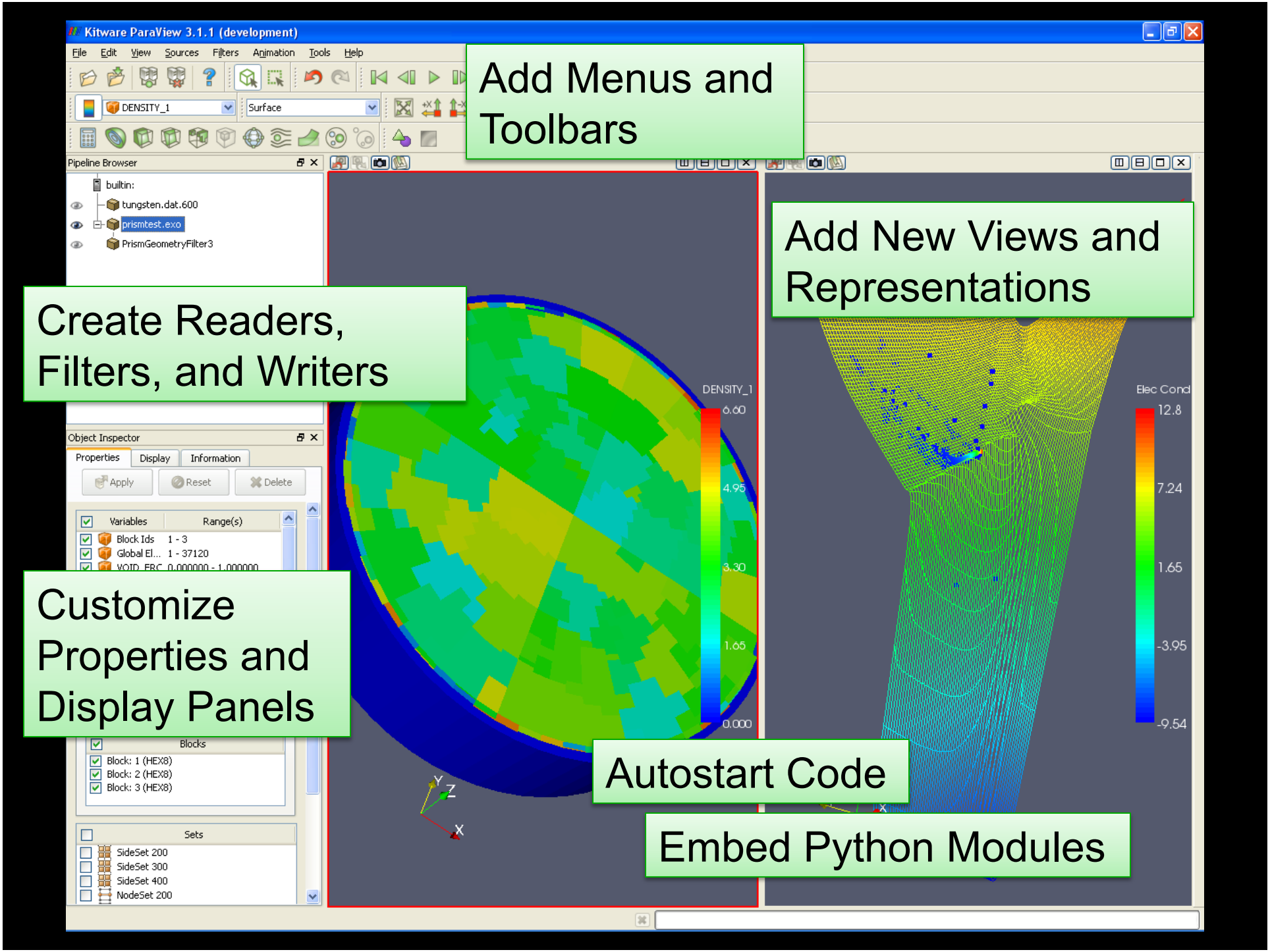
Customizing ParaView with Plugins

IEEE Vis ParaView Tutorial

October 13, 2009

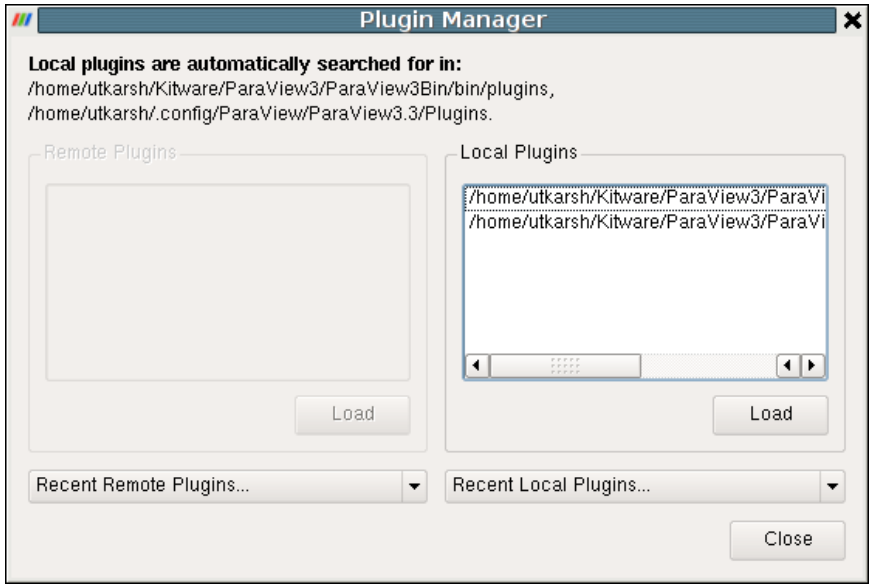
Kenneth Moreland
Sandia National Laboratories

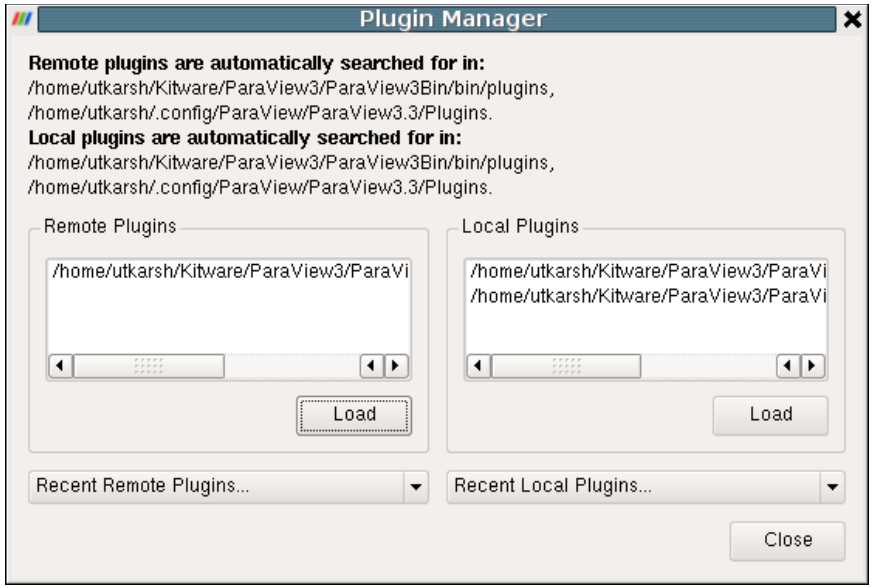
Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.


Kitware ParaView 3.1.1 (development)
File Edit View Sources Filters Animation Tools Help
DENSITY_1
Surface
Pipeline Browser
builtin:
tungsten.dat.600
prismtest.exo
PrismGeometryFilter3
Object Inspector
Properties Display Information
Apply Reset Delete
Variables Range(s)
Block Ids 1 - 3
Global El... 1 - 37120
Blocks
Block: 1 (HEX8)
Block: 2 (HEX8)
Block: 3 (HEX8)
Sets
SideSet 200
SideSet 300
SideSet 400
NodeSet 200
DENSITY_1
6.60
4.95
3.30
1.65
0.000
Elec Cond
12.8
7.24
1.65
-3.95
-9.54
Add Menus and Toolbars
Create Readers, Filters, and Writers
Add New Views and Representations
Customize Properties and Display Panels
Autostart Code
Embed Python Modules

Loading Plugins

- **GUI Plugin Manager (Tools→Manage Plugins/Extensons).**





- **PV_PLUGIN_PATH environment variable.**
- **Recognized locations (see top of Plugin Manager).**

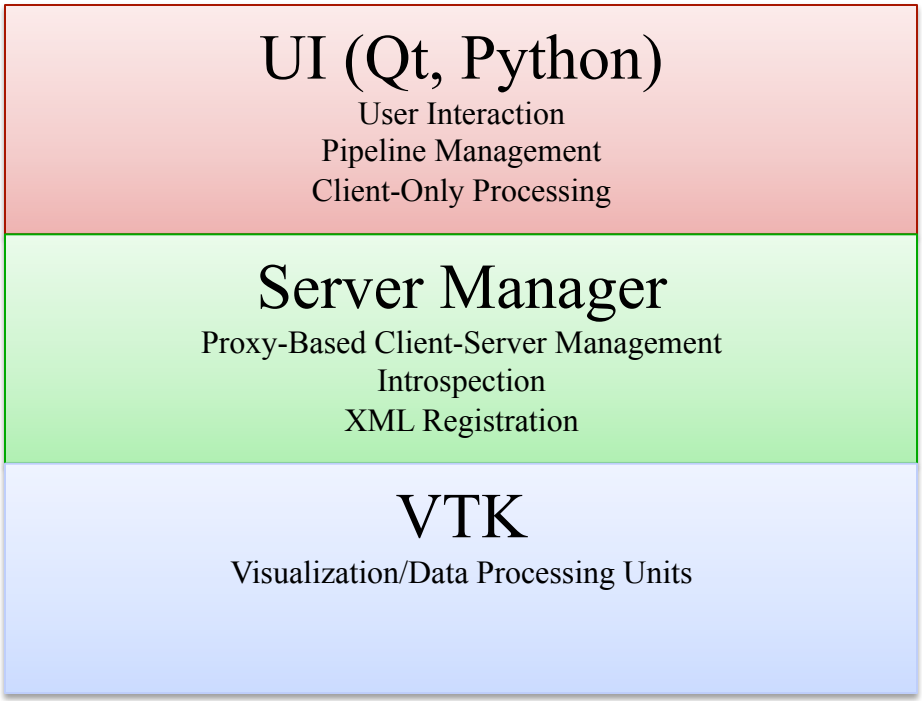
Sandia National Laboratories

Where to Go for Help

- Documentation on the Wiki
  - http://www.paraview.org/Wiki/Plugin_HowTo
  - www.paraview.org → Help→ Wiki → Plugins
- MIRARCO plugin wizard
  - http://pluginwizard.mirarco.org/
- Examples
  - ParaView3/Examples/Plugins
  - ParaView3/Plugins

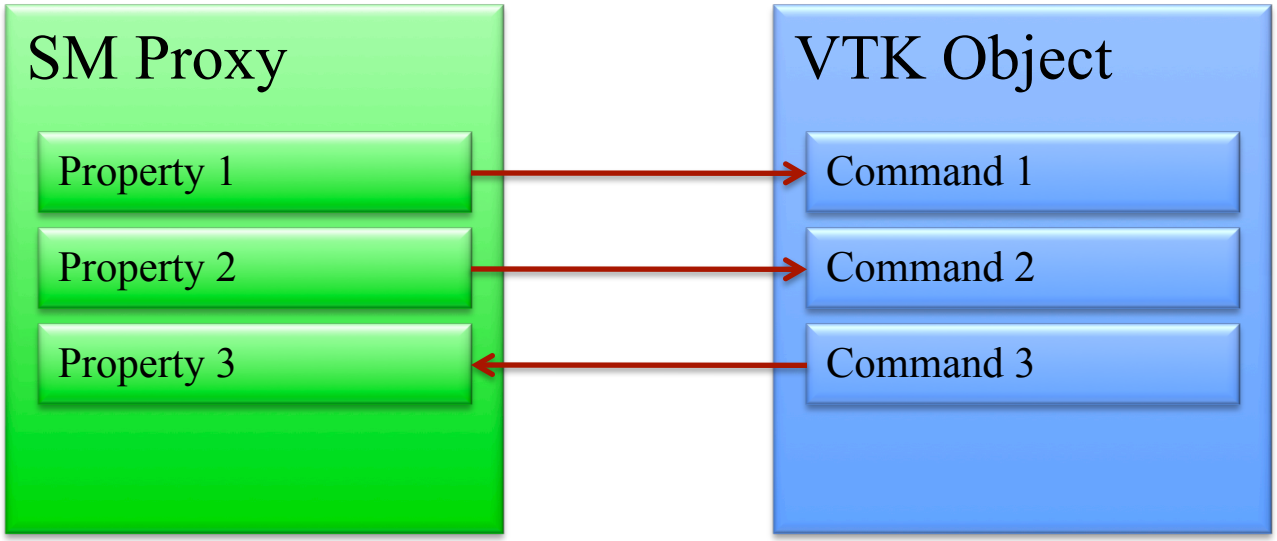
Sandia National Laboratories

ParaView Application Layers



Sandia National Laboratories

Server Manager Proxy Objects



Client Only

Any, all, or none of client, render server nodes, and data server nodes.

Root Tag

Named Group

Filter Proxy Definition

```
<ServerManagerConfiguration>
  <ProxyGroup name="filters">

    <SourceProxy name="OBBDicer" class="vtkOBBDicer" label="OBB Dicer">
      <Documentation
        short_help="Break dataset into pieces."
        long_help="Define pieces for a dataset and annotate in a field.">
        This filter uses a tree of oriented bounding boxes to partition the
        space in which a data set is defined.  A field is then added to the
        dataset that defines which partition each point is contained in.
      </Documentation>

      <InputProperty name="Input" ...

      <IntVectorProperty name="DiceMode" ...

      <IntVectorProperty name="NumberOfPointsPerPiece" ...

      <IntVectorProperty name="NumberOfPieces" ...

      <IntVectorProperty name="MemoryLimit" ...

    </SourceProxy> <!-- OBBDicer -->

  </ProxyGroup>
</ServerManagerConfiguration>
```

Property tags inside proxy tag.

Sandia National Laboratories

```
<ServerManagerConfiguration>
  <ProxyGroup name="filters">

    <SourceProxy name="OBBDicer" ...

      <InputProperty name="Input" command="SetInputConnection">
        <ProxyGroupDomain name="groups">
          <Group name="sources" />
          <Group name="filters" />
        </ProxyGroupDomain>
        <DataTypeDomain name="input_type">
          <DataType value="vtkDataSet" />
        </DataTypeDomain>
      </InputProperty>

      <IntVectorProperty name="DiceMode" ...

      <IntVectorProperty name="NumberOfPointsPerPiece" ...

      <IntVectorProperty name="NumberOfPieces" ...

      <IntVectorProperty name="MemoryLimit" ...
```

vtkAlgorithmMethod

Groups from which inputs can come.

Data Types the Input can have.

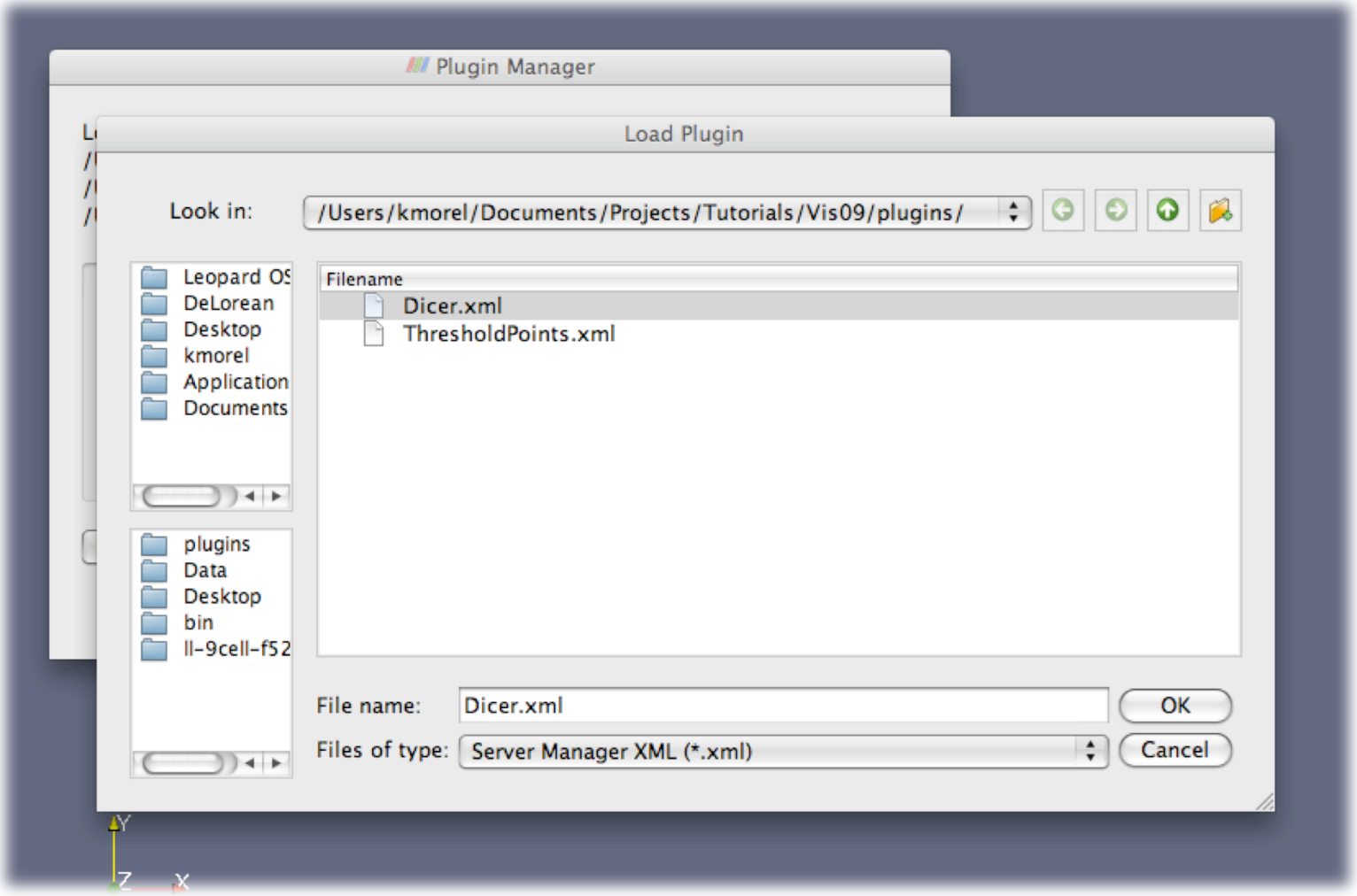
Sandia National Laboratories

```
<ServerManagerConfiguration>
  <ProxyGroup name="filters">

    <SourceProxy name="OBBDicer" ...

      <InputProperty name="Input" ...

      <IntVectorProperty name="DiceMode" command="SetDiceMode"
                         number_of_elements="1"
                         default_values="0">
        <EnumerationDomain name="enum">
          <Entry value="0" text="Number of Points" />
          <Entry value="1" text="Specified Number" />
          <Entry value="2" text="Memory Limit" />
        </EnumerationDomain>
        <Documentation>
          Specify the method to determine how many pieces the data should
          be broken into.
        </Documentation>
      </IntVectorProperty>

      <IntVectorProperty name="NumberOfPointsPerPiece"
                         command="SetNumberOfPointsPerPiece"
                         number_of_elements="1"
                         default_values="5000">
        <IntRangeDomain name="range" min="1000" />
        <Documentation>
          The maximum number of points to have in each piece.  Only valid
          if the mode is set to Number of Points.
        </Documentation>
      </IntVectorProperty>

      <IntVectorProperty name="NumberOfPieces" ...

      <IntVectorProperty name="MemoryLimit" ...
```

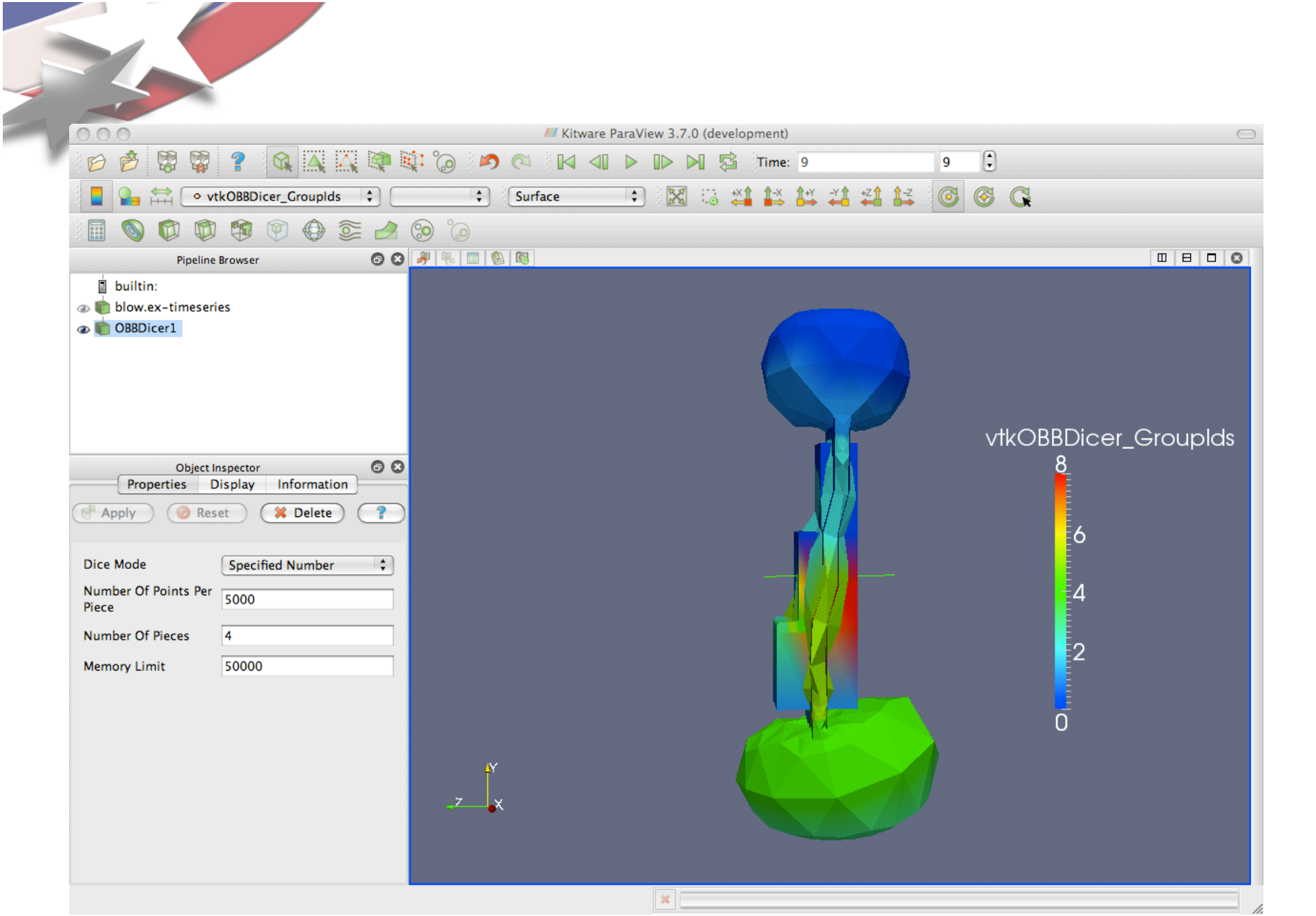
Numerical Properties are Vectors (scalars are size 1)

Domains limit values and help GUI build widgets

Sandia National Laboratories

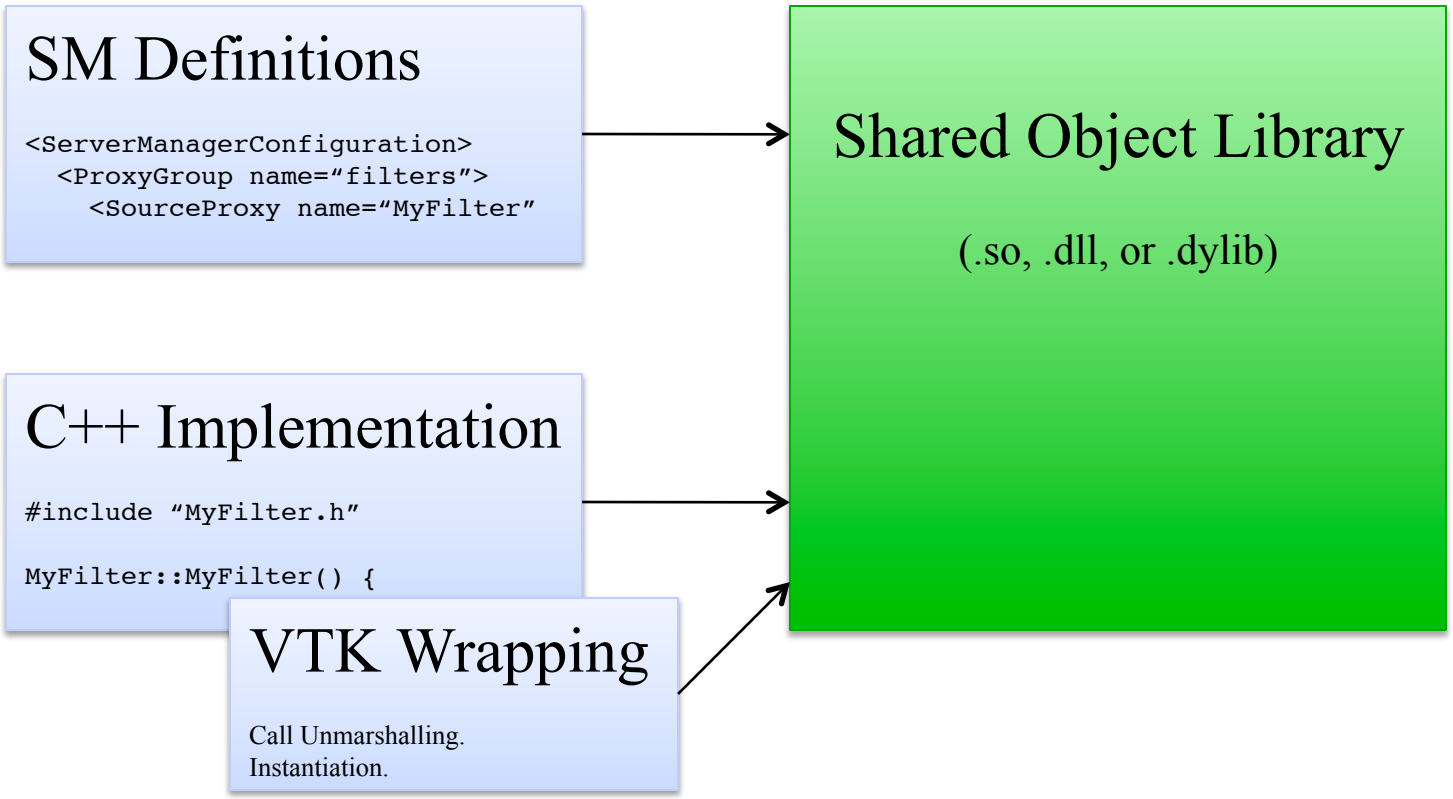


Sandia National Laboratories



Sandia National Laboratories

Bundling SM XML with Implementation



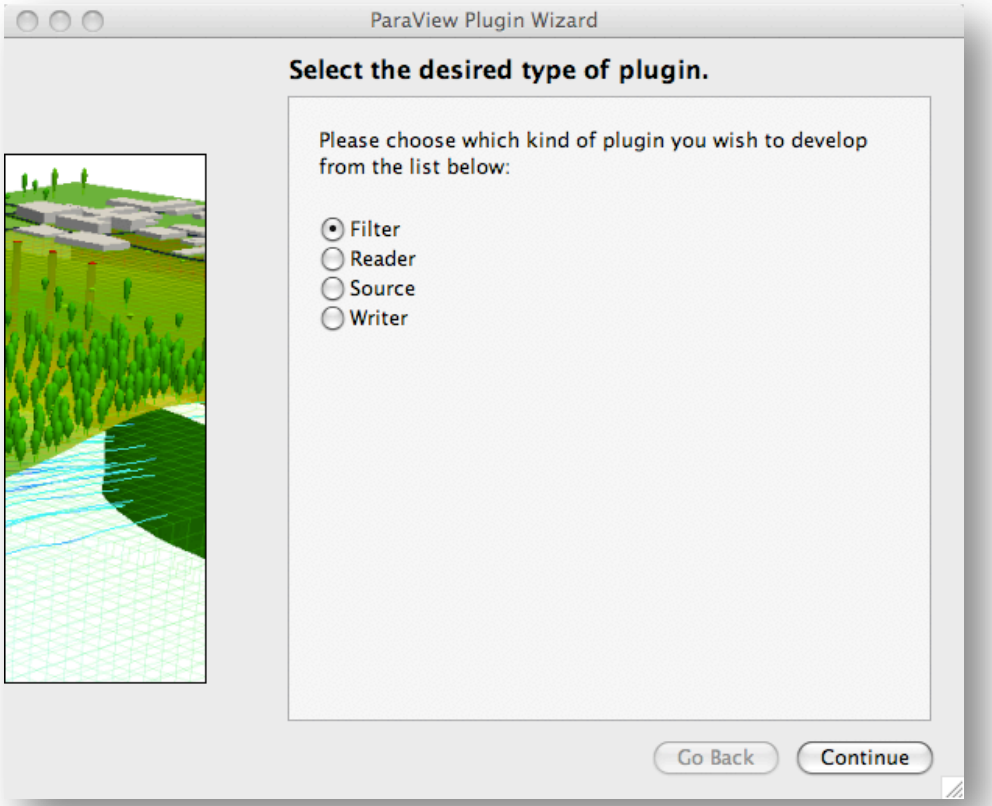
Bundling Plugins with CMake

```
PROJECT(MyFilter)

FIND_PACKAGE(ParaView REQUIRED)
INCLUDE(${PARAVIEW_USE_FILE})

ADD_PARAVIEW_PLUGIN(MyFilter "1.0"
  SERVER_MANAGER_XML MyFilter.xml
  SERVER_MANAGER_SOURCES vtkMyFilter.cxx
  )
```

Sandia National Laboratories

Using MIRARCO's Plugin Wizard

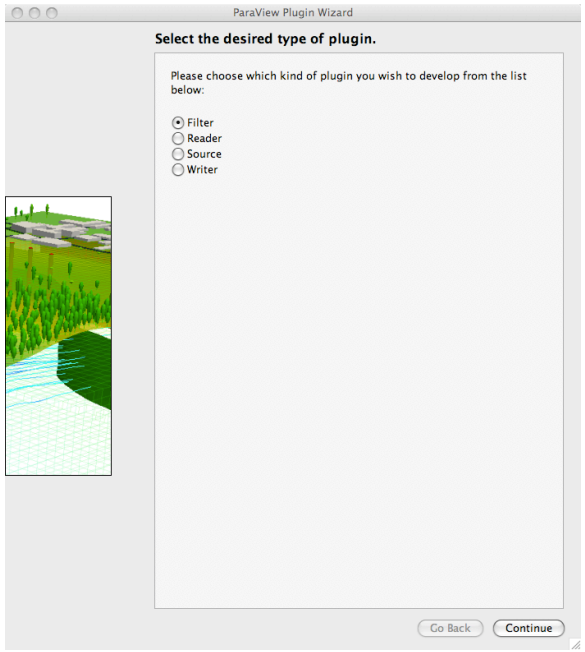
- http://pluginwizard.mirarco.org/

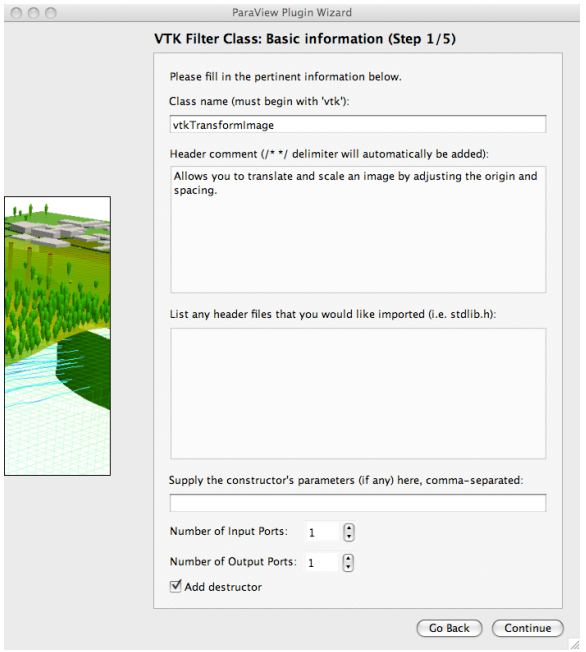


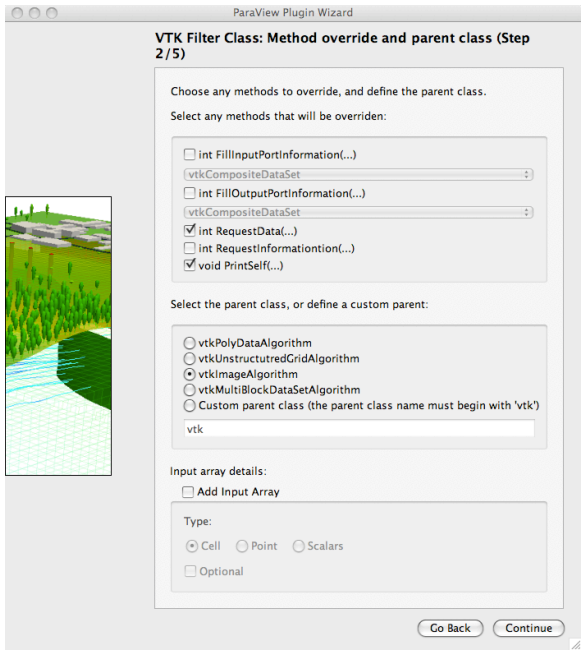
Designed by Matthew Ansell, Matthew Livingstone, and Robert Maynard

Sandia National Laboratories

Create a vtkTransformImage Filter







Sandia National Laboratories

Implementation: Add Properties to SM XML

```
<SourceProxy name="TransformImage" class="vtkTransformImage">

  <InputProperty
    name="Input"
    command="SetInputConnection">
    <ProxyGroupDomain name="groups">
      <Group name="sources"/>
      <Group name="filters"/>
    </ProxyGroupDomain>
    <DataTypeDomain name="input_type">
      <DataType value="vtkImageData"/>
    </DataTypeDomain>
  </InputProperty>

  <DoubleVectorProperty name="Translate" command="SetTranslate"
                        number_of_elements="3" default_values="0 0 0">
  </DoubleVectorProperty>
  <DoubleVectorProperty name="Scale" command="SetScale"
                        number_of_elements="3" default_values="1 1 1">
  </DoubleVectorProperty>

</SourceProxy>
```

Sandia National Laboratories

Implementation: Add Properties to Header

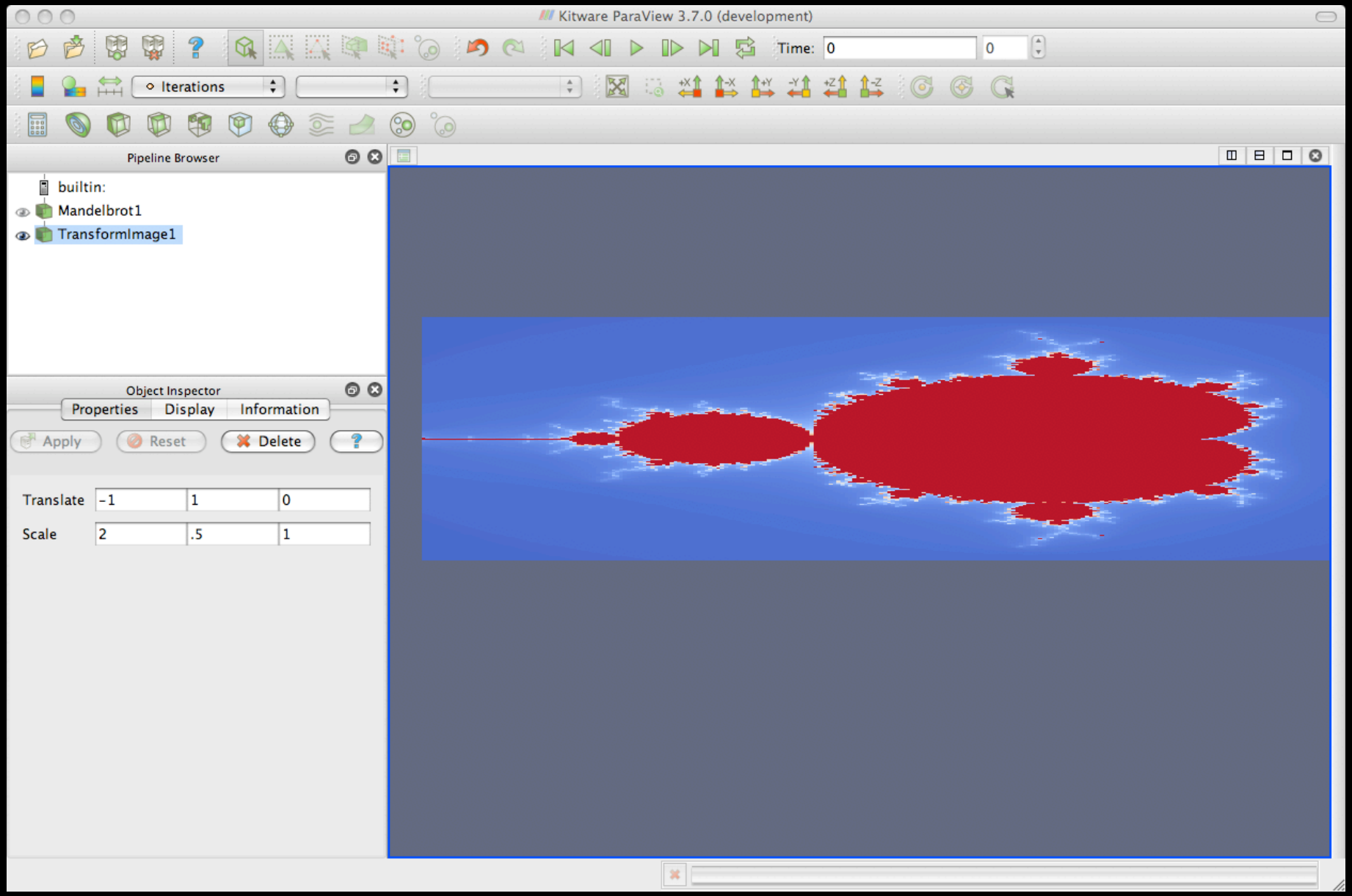
```
public:
  static vtkTransformImage *New();
  vtkTypeRevisionMacro(vtkTransformImage,vtkImageAlgorithm);
  void PrintSelf(ostream& os, vtkIndent indent);

  vtkGetVector3Macro(Translate, double);
  vtkSetVector3Macro(Translate, double);

  vtkGetVector3Macro(Scale, double);
  vtkSetVector3Macro(Scale, double);

protected:
  vtkTransformImage();
  ~vtkTransformImage();

  double Translate[3];
  double Scale[3];

  int RequestData(vtkInformation *, vtkInformationVector **,
                  vtkInformationVector *);
```

Sandia National Laboratories

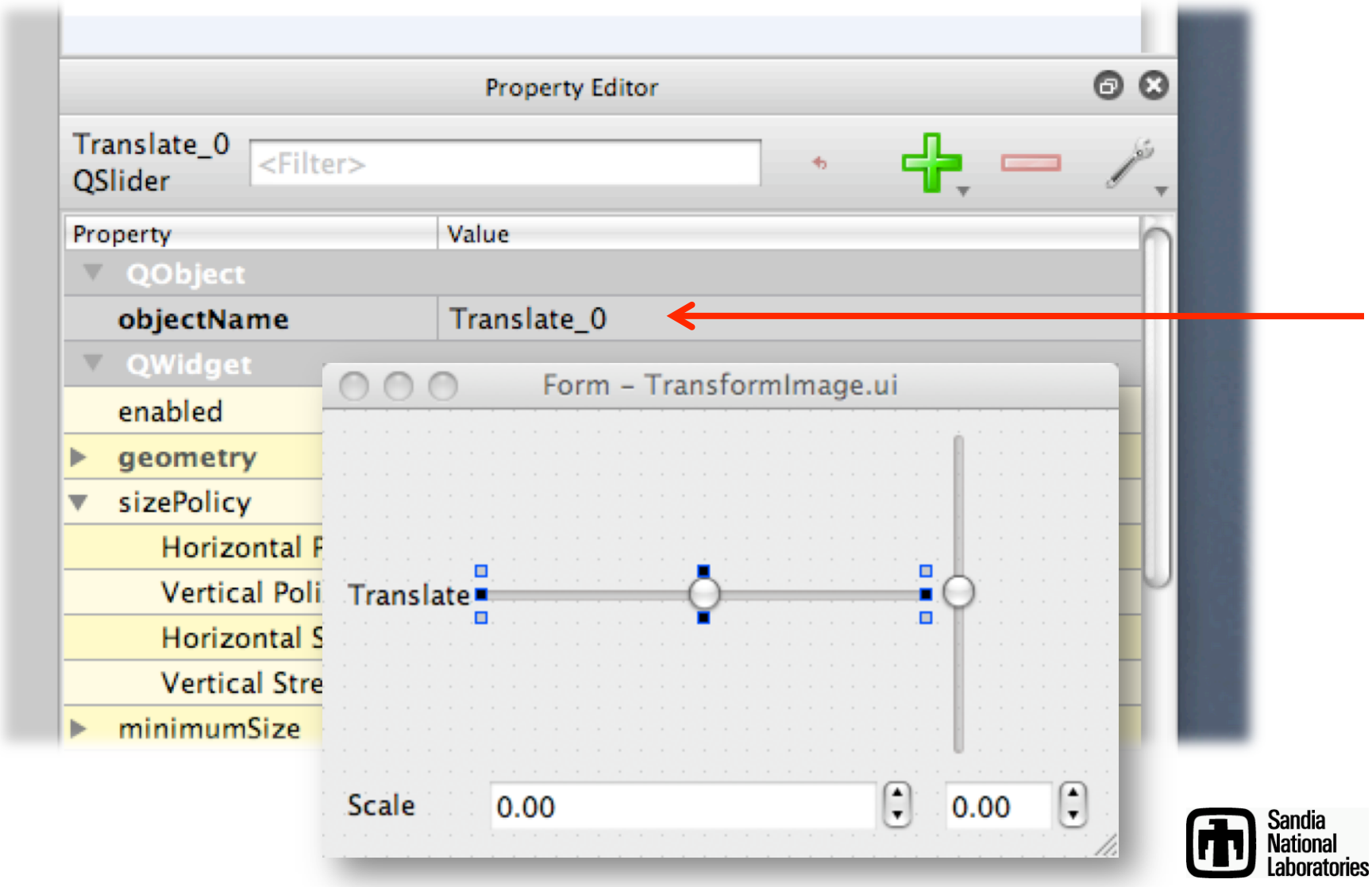
Implementation: Computation

```
int vtkTransformImage::RequestData(vtkInformation *request,
                                   vtkInformationVector **inputVector,
                                   vtkInformationVector *outputVector)
{
  vtkImageData *input = vtkImageData::GetData(inputVector[0]);
  vtkImageData *output = vtkImageData::GetData(outputVector);

  output->CopyStructure(input);
  output->GetPointData()->PassData(input->GetPointData());
  output->GetCellData()->PassData(input->GetCellData());

  double origin[3], spacing[3];
  output->GetOrigin(origin);
  output->GetSpacing(spacing);
  for (int i = 0; i < 3; i++)
    {
    origin[i] += this->Translate[i];
    spacing[i] *= this->Scale[i];
    }
  output->SetOrigin(origin);
  output->SetSpacing(spacing);

  return 1;
}
```

Sandia National Laboratories


Kitware ParaView 3.7.0 (development)
Time:
0
0
Iterations
Pipeline Browser
builtin:
Mandelbrot1
TransformImage1
Object Inspector
Properties
Display
Information
Apply
Reset
Delete
Translate
-1
1
0
Scale
2
.5
1

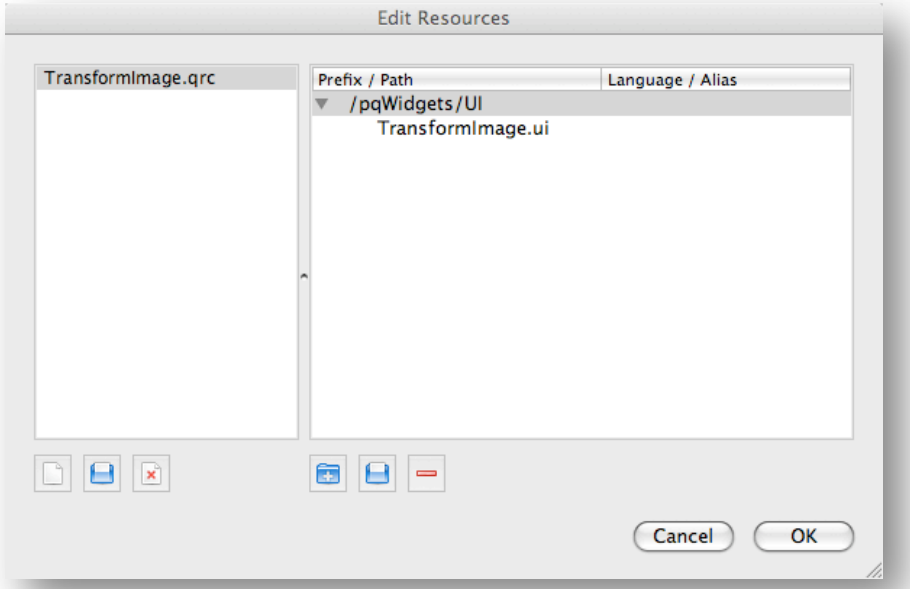
Creating a Custom Object Panel in Qt Designer



Sandia National Laboratories

Linking the Custom Object Panel

- **ParaView will look for a Qt resource file with path :/pqWidgets/UI/*proxyname*.ui**




```
<RCC>
  <qresource prefix="/pqWidgets/UI" >
    <file>TransformImage.ui</file>
  </qresource>
</RCC>
```

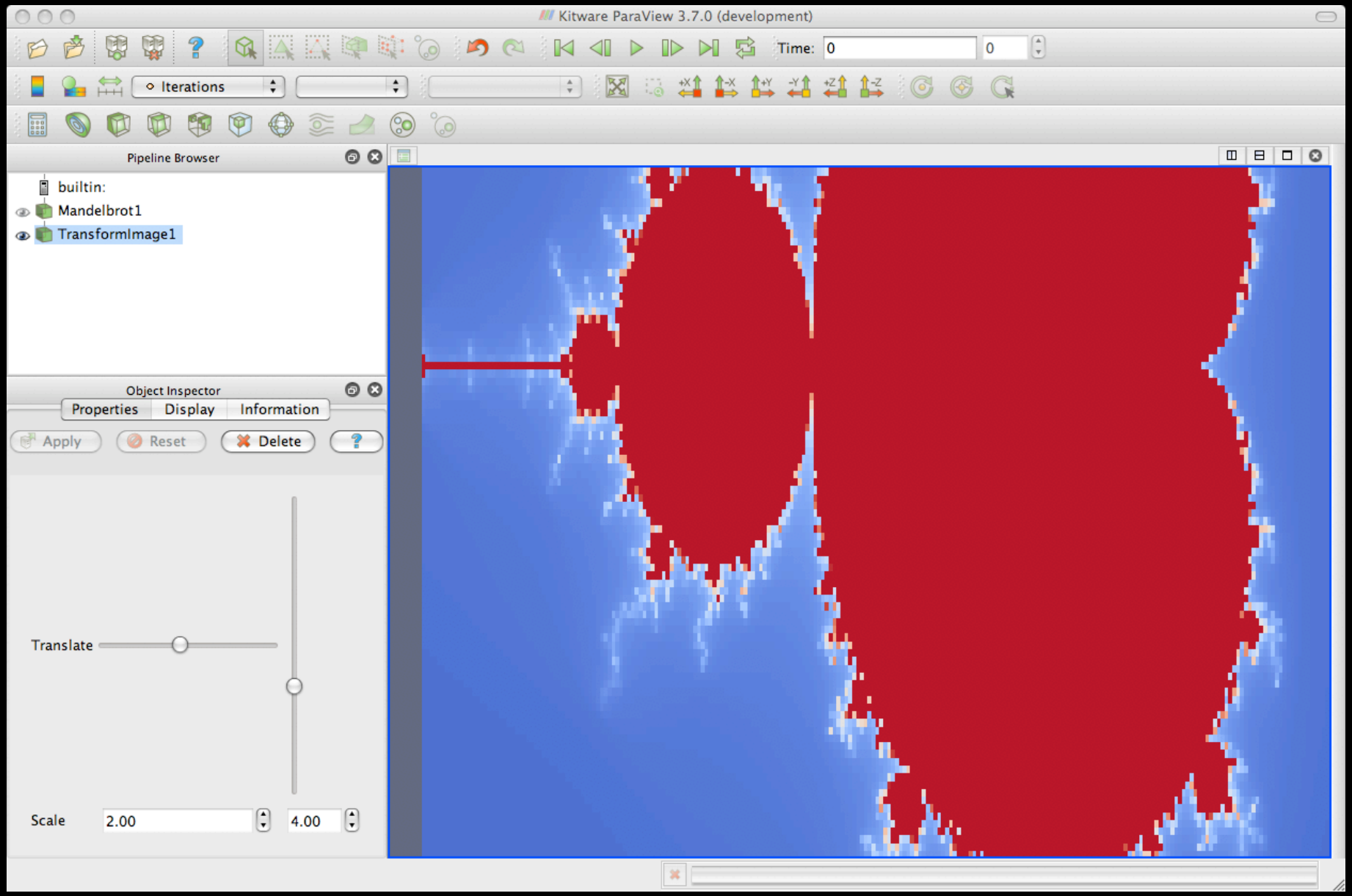
Sandia National Laboratories

Linking the Custom Object Panel

- **Add the resource file to the ADD_PARAVIEW_PLUGIN command.**

```
ADD_PARAVIEW_PLUGIN(TransformImageSMPlugin "1.0"
      SERVER_MANAGER_XML TransformImage.xml
      SERVER_MANAGER_SOURCES vtkTransformImage.cxx
      GUI_RESOURCES TransformImage.qrc
      )
```

Sandia National Laboratories


Kitware ParaView 3.7.0 (development)
Time:
0
0
Iterations
Pipeline Browser
builtin:
Mandelbrot1
TransformImage1
Object Inspector
Properties
Display
Information
Apply
Reset
Delete
Translate
Scale
2.00
4.00

Panel Classes

pqAutoGeneratedObjectPanel Automatically creates widgets based on the properties defined in the server manager proxies. Used internally to create the automatically defined panels. Can be extended if you have small adjustments to make or small features to add.

pqNamedObjectPanel When your widgets have names corresponding to the server manager properties, this class can automatically hook them up. If your initial design comes principally from Qt Designer, this is a good class to extend.

pqObjectPanel Base class for all object panels. Provides little other than a blank widget container. You are responsible for managing Apply/Reset, SM state, and the undo stack.

```
#include "pqTransformImagePanel.h"

#include "ui_TransformImage.h"
class pqTransformImagePanel::pqUI : public
Ui::TransformImage {};

pqTransformImagePanel::pqTransformImagePanel(
                              pqProxy *pxy, QWidget *p)
  : pqNamedObjectPanel(pxy, p)
{
  this->ui = new pqUI;
  this->ui->setupUi(this);

  this->linkServerManagerProperties();
}

pqTransformImagePanel::~pqTransformImagePanel()
{
  delete this->ui;
}
```

```
PROJECT(TransformImage)

FIND_PACKAGE(ParaView REQUIRED)
INCLUDE(${PARAVIEW_USE_FILE})

QT4_WRAP_UI(UI_SRCS TransformImage.ui)
QT4_WRAP_CPP(MOC_SRCS pqTransformImagePanel.h)

ADD_PARAVIEW_OBJECT_PANEL(IFACES IFACE_SRCS
  CLASS_NAME pqTransformImagePanel
  XML_NAME TransformImage XML_GROUP filters
  )

ADD_PARAVIEW_PLUGIN(TransformImageSMPlugin "1.0"
  SERVER_MANAGER_XML TransformImage.xml
  SERVER_MANAGER_SOURCES vtkTransformImage.cxx
  GUI_INTERFACES ${IFACES}
  GUI_SOURCES ${MOC_SRCS} ${UI_SRCS} ${IFACE_SRCS}
    pqTransformImagePanel.cxx
  )
```

Autostart Plugins

- ParaView can automatically launch code when your plugin starts up and when it shuts down.
- Create a QObject class with methods you want called on startup and shutdown.
- Use ADD_PARAVIEW_AUTO_START command to register startup and shutdown methods.

```
#include <QObject>

class pqMyApplicationStarter : public QObject
{
  Q_OBJECT

public:
  pqMyApplicationStarter(QObject* p=0);
  ~pqMyApplicationStarter();

  // Callback for startup.
  void onStartup();

  // Callback for shutdown.
  void onShutdown();
```

```
QT4_WRAP_CPP(MOC_SRCS pqMyApplicationStarter.h)

ADD_PARAVIEW_AUTO_START(IFACES IFACE_SRCS
  CLASS_NAME pqMyApplicationStarter
  STARTUP onStartup
  SHUTDOWN onShutdown
  )

ADD_PARAVIEW_PLUGIN(Autostart "1.0"
  GUI_INTERFACES ${IFACES}
  GUI_SOURCES pqMyApplicationStarter.cxx
  ${MOC_SRCS} ${IFACE_SRCS}
  )
```

Menu/Toolbar Plugins

- Your plugin can create menus and toolbars and add actions.
  - Connect actions to Qt slots to perform arbitrary operations.
- Create a subclass of QActionGroup that fills itself with QActions in its constructor.
  - Hint: You can use Qt Designer to create a “dummy” widget with the QActions you want to return. Create all your QActions with a single setupUi call.
- Use ADD_PARAVIEW_ACTION_GROUP command to register startup and shutdown methods.

Sandia National Laboratories

```
QT4_WRAP_CPP(MOC_SRCS MyActions.h)

ADD_PARAVIEW_ACTION_GROUP(TB_IFACE TB_IFACE_SRCS
  CLASS_NAME MyActions
  GROUP_NAME "ToolBar/MyActions”
  )

ADD_PARAVIEW_ACTION_GROUP(M_IFACE M_IFACE_SRCS
  CLASS_NAME MyActions
  GROUP_NAME "MenuBar/MyActions”
  )

ADD_PARAVIEW_PLUGIN(SourceToolbar "1.0”
  GUI_INTERFACES ${TB_IFACES} ${M_IFACES}
  GUI_SOURCES
    ${MOC_SRCS}
    ${TB_IFACE_SRCS} ${M_FACE_SRCS}
    SourceToolbarActions.cxx
  )
```

What's Next

- More plugin types exist. More on the way.
  - Keep up to date with Wiki documentation.
- Coming soon: Branding.
  - The client applications will be replaced with a generic empty application.
  - Empty application configured to load a set of predefined plugins, called a brand.
  - In the future, you should be able to create new "vertical applications" by writing plugins.
  - Allows you to better share code between vertical applications (brands).