Deprecation of vtkAttributeData and Subclasses

Kitware Inc.

October 10, 2001

Abstract

As of VTK 4.0, vtkAttributeData and it's subclasses (vtkScalars, vtkVectors, vtkNormals, vtkTCoords, vtkTensors) will be deprecated. This does **not** mean that VTK will no longer support attributes. Instead, all attributes and fields will be represented by vtkDataArray or one of it's subclasses. This documents provides information about why there changes were made, when they will become part of VTK and what users and developers can do to modify their code to be compatible with the coming release.

1 Why

One of the important improvements in VTK 4.0 is the uniform handling of attributes and fields. As of next release, it will be much easier to work with field data (see Field Data Changes). One of the changes made involved storing attributes in the field data and treating them as any other field. Now, whenever an attribute (vtkScalars etc...) is added to the point or cell data, the data array (vtkDataArray) which is normally handled under the hood by the attribute data is extracted and added to the field data. This makes manipulating attributes by directly modifying the corresponding arrays possible and very easy. The biggest advantage of this is that the interface of all attributes and fields is now uniform. For example, to modify a tuple of any attribute (or field), one can use the Set/GetTuple() pair, instead of using a different pair (Set/GetScalar, Set/GetVector, Set/GetTensor etc...) for each attribute. Furthermore, most of the code which had to treat attributes specially can now be replaced with code which simply processes all fields the same way.

2 When

The changes mentioned above will be part of the CVS tree as of November 1, 2001 and will be part of the next (4.0) release.

3 Impact on Users and Developers

With the removal of vtkAttributeData, it's subclasses and some related filters, some programs/filters will have to be modified. Most of these changes are relatively simple. For example,

```
vtkVectors* vectors = vtkVectors::New();
vectors->SetNumberOfVectors(numVec);
float vec[3];
for (int i=0; i<numVec; i++)
  {
  // Compute vec
  vectors->SetVector(i, vec);
  }
```

can be replaced by

```
vtkFloatArray* vectors = vtkFloatArray::New();
vectors->SetNumberOfComponents(3);
vectors->SetNumberOfTuples(numVec);
```

```
float vec[3];
for (int i=0; i<numVec; i++)
  {
  // Compute vec
  vectors->SetTuple(i, vec);
  }
```

Some of modifications can be more significant. For example, any code which uses vtkScalars::GetColor() will have to use vtkLookupTable::MapScalars() (or something similar depending on the filter) instead. To help users and developers to migrate their code, we wrote a Perl script which parses C++, Tcl, Python or Java files and tries to generate a list of required changes. This script is part of the package AttributeChanges.zip which can be downloaded at http://public.kitware.com/files/AttributeChanges.zip