TSF to Playa transition

Kevin Long

February 5, 2011

1 Various name changes

Most of the name changes can be automated using the script `changeNames.sh`. For example, to change the names of all files in the current directory, run

```
[PATH-TO-TRILINOS]/packages/Sundance/Playa/maintenance/changeNames.sh *.cpp *.hpp
```

Some manual changes may be needed, particularly those described in 1.2.

1.1 Bye bye TSF

This script will change the namespace name `TSFExtended` to `Playa` throughout, and will change include directives such as

```
#include "TSFVectorDecl.hpp"
```

to

```
#include "PlayaVectorDecl.hpp"
```

1.2 Class namespace changes

Classes Printable, Tabs, Handleable, and Handle have all been moved from the Sundance namespace to the Playa namespace. Likewise, the names of their header files have been changed from SundanceTabs.hpp to PlayaTabs.hpp, etc. Much of this renaming can be done by the script, but because most files using, *e.g.*, `Tabs` had assumed the namespace Sundance, you may need to do one of the following edits manually:

1. Add whichever of

   ```
   using Playa::Tabs;
   using Playa::Out;
   using Playa::Handle;
   using Playa::Handleable;
   ```

   are needed in each file.
2. Change `Tabs`, `Out`, `Handle`, and `Handleable` to `Playa::Tabs`, etc.
3. Add `using namespace Playa;` to each source file affected (not recommended)

1.2.1 NOX sub-namespaces

The namespaces NOX::TSF or NOX::Playa should be changed to NOX::NOXPlaya.

1.3 Header file name changes

The header files

```
NOX_Playa_Group.H
NOX_Playa_Vector.H
NOX_Playa.H
NOX_StatusTest_SafeCombo.H
NOX_Playa_StatusTestBuilder.H
PlayaNOXSolver.H
```

have had their extensions changed from `.H` to `.hpp`.

Change

```
#include "PlayaBlockVectorSpaceDecl.hpp"
#include "PlayaBlockVectorSpaceImpl.hpp"
```

to

```
#include "PlayaDefaultBlockVectorSpaceDecl.hpp"
#include "PlayaDefaultBlockVectorSpaceImpl.hpp"
```

1.4 Function name changes

Old	New
VectorSpace<S>::lowestLocallyOwnedIndex()	VectorSpace<S>::baseGlobalNaturalIndex()
productSpace(...)	blockSpace(...)
ObjectWithVerbosity::setVerbosity()	ObjectWithVerbosity::setVerb()

2 Element access

2.1 Bracket operators

The vector bracket operators expect local indices.

2.2 get/setElement functions

In order to reduce the confusing redundancy in element access functions, the getElement() and setElement() functions have been removed. If you use them, they should be replaced as follows:

- Access involving local elements only should be done through the bracket operators using local indices, supplemented by block iterators if needed. This is sufficient for any elementwise functions.
- Reading global elements should be done through the GhostView interface. A solver developer will almost never need to do this.
- Setting global elements should be done through the LoadableVector interface. A solver developer will almost never need to do this.

2.3 LoadableVector interface

The `LoadableVector` methods expect global indices.

If you want high-performance vector element loading, it's best to dynamic cast the vector's pointer to a LoadableVector pointer before beginning the fill loop. A quick (for the programmer) and dirty (for the computer) shortcut is to use the `loadable` function as follows:

```
Vector<double> x = getSomeVector();
for (int g=low; g<high; g++) loadable(x)->setElement(g, val);
```

This does a dynamic cast in the inner loop and so will be slow, but it's an easy shortcut if you want to keep your code looking clean.

2.4 Ghost (off-processor) elements

Read-only access to selected off-processor elements is done through the `GhostView` interface. The methods of `GhostView` expect global indices.

3 Solver changes

To support old XML solver parameter files, the linear solver builder recognizes solver type name `TSF` as equivalent to solver type name `Playa`.

4 Removed features

1. Support for partitioned BCs has been removed. This had gone untested due to various Thyra limitations, and the need has been superseded by the use of Nitsche BCs. It can be put back in if needed.
2. The `SequentialIterator` object has been removed. It has been superseded by the combination of block iterators and bracket operators on local indices.
3. The TSF GMRES solver has been removed.
4. Thyra interoperability is gone for now. It can be added should the need arise.