SAND2008-7488P

Test Driven Development

Trilinos Users Group, Developer Day

October 23, 2008

Todd Coffey
Computational Mathematics & Applications
Roscoe Bartlett
Uncertainty Quantification & Optimization

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.

What are Unit Tests?

- **A unit test is a *fast* test that *localizes* errors**
  "Working Effectively with Legacy Codes"
- **Unit tests verify correctness of individual functions (units) in a code**

- **Integration tests verify interfaces between units**
- **System tests verify the code end-to-end**

Sandia National Laboratories

Benefits of Unit Tests

Code Coverage

Safeguard code against future edits

Localizes coding errors

Hardened Solvers

Code gets better every time you work on it

Feels good

Quick verification of edits

Shorter Development Time

Sandia National Laboratories

What is Test Driven Development?

1. Write (unit) tests before you write code
2. Verify test fails
3. Write code to satisfy test
4. Refactor code

That's the essence of it.

Benefits of TDD

Design better code

Define work completion

Find defects early

Tests get written

Detect unintended changes

Write testable code

Focus on refactoring

See code from interface

Foo

Teuchos Unit Test Harness

- **Macros that make writing unit tests very easy**

```
#include "Teuchos_UnitTestHarness.hpp”
TEUCHOS_UNIT_TEST( group_name, test_name ) { … }
```

- **Driver code which runs all the tests**

```
#include "Teuchos_UnitTestRepository.hpp”
#include "Teuchos_GlobalMPISession.hpp”
int main( int argc, char* argv[] ) {
  Teuchos::GlobalMPISession mpiSession(&argc, &argv);
  return Teuchos::UnitTestRepository::runUnitTestsFromMain(argc,
argv);
}
```

- **Makefile defines which unit tests get pulled in**

Teuchos TEST options:

TEUCHOS_ECHO(statement)

TEUCHOS_TEST_EQUALITY[_CONST](a, b)

TEUCHOS_TEST_INEQUALITY[_CONST](a, b)

TEUCHOS_TEST_FLOATING_EQUALITY(a, b, tol)

TEUCHOS_TEST_ITER_EQUALITY(it_a, it_b)

TEUCHOS_TEST_ARRAY_ELE_EQUALITY(A, i, val)

TEUCHOS_TEST_MATRIX_ELE_FLOATING_EQUALITY
(A, i, j, tol)

TEUCHOS_TEST_MATRIX_ELE_EQUALITY(A, i, j, val)

TEUCHOS_TEST_COMPARE(a, <=, b)

TEUCHOS_TEST_THROW(statement, std::logic_error)

TEUCHOS_TEST_NOTHROW(statement)

Hello World Unit Test

```
#include "Teuchos_UnitTestHarness.hpp”
std::string hello() {
//  std::string hw = “Hello World”;
return hw;
}
TEUCHOS_UNIT_TEST( HelloWorld, test0) {
   std::string hw = hello();
TEST_EQUALITY_CONST( hw, “Hello World” );
}
```

Xyce Example

- Harmonic Balance (HB) Algorithm (based on MPDE algorithm)
- Builder is a factory for vectors and matrices
- HB uses product vectors
- HB uses Real Equivalent Form for FFT
- Verify product vectors are correct size

Write Unit Test

```
RCP<N_HB_Builder>
createHBBuilder(int numBlocks, int numSolutionVars, int numStateVars)  {
  RCP<N_HB_Buidler> hbBuilder;
  return builder;
}
TEUCHOS_UNIT_TEST( HB_Builder,  createVector )  {
  RCP<N_HB_Builder> hbBuilder = createHBBuilder( 10, 3, 5 );
  TEST_EQUALITY_CONST( is_null(hbBuilder), false );
  RCP<N_LAS_Vector> vec = hbBuilder->createVector(0.0);
  RCP<N_LAS_BlockVector> bvec = rcp_dynamic_cast<N_LAS_BlockVector>(vec,false);
  TEST_EQUALITY_CONST( Teuchos::is_null(bvec), false );
  TEST_EQUALITY_CONST( bvec->blockCount(), 3 );
  TEST_EQUALITY_CONST( bvec->blockSize(), 20 );
}
```

Run Test

```
$ ./XyceDemoUnitTest --show-test-details=ALL
Teuchos::GlobalMPISession::GlobalMPISession(): started serial run

***
*** Unit test suite ...
***

Sorting tests by group name then by test name …
Running unit tests …

0. HB_Builder_createVector_UnitTest ...
  is_null(hbBuilder) = 1 == 0 : failed
Segmentation fault
```

Fill Dependencies

```
RCP<N_HB_Builder>
createHBBuilder(int numBlocks, int numSolutionVars, int numStateVars)  {
  N_IO_CmdParse cmdLine;
  RCP<N_MPDE_Manager> mpdeMgrRCPtr = rcp(new N_MPDE_Manager(cmdLine));
  N_MPDE_Discretization::Type type = N_MPDE_Discretization::Backward;
  int order = 1;
  RCP<N_MPDE_Discretization> discRCPtr = rcp(new N_MPDE_Discretization(type,order) );
  bool warpMPDEFlag = false;
  RCP<N_HB_Buidler>
    hbBuilder = rcp(new N_HB_Builder( mpdeMgrRCPtr, numBlocks, discRCPtr, warpMPDEFlag));
  return builder;
}
TEUCHOS_UNIT_TEST( HB_Builder,  createVector )  { … }
```

More Dependencies

```
RCP<N_HB_Builder>
createHBBuilder(int numBlocks, int numSolutionVars, int numStateVars)  {
  N_IO_CmdParse cmdLine;
 RCP<N_PDS_Manager> pdsMgrRCPtr;
 bool isSerial;
 bool procFlag;
 pdsMgrRCPtr = rcp(new N_PDS_Manager(isSerial,procFlag));
 cmdLine.registerParallelMgr(pdsMgrRCPtr);
  int iargs = 2;
  char *arg0 = "Xyce";
  char *arg1 = "foo.cir";
  char *cargs[iargs];
  cargs[0] = arg0;
  cargs[1] = arg1;
  cmdLine.parseCommandLine(iargs,cargs);
  RCP<N_MPDE_Manager> mpdeMgrRCPtr = rcp(new N_MPDE_Manager(cmdLine));
  N_MPDE_Discretization::Type type = N_MPDE_Discretization::Backward;
  int order = 1;
  RCP<N_MPDE_Discretization> discRCPtr = rcp(new N_MPDE_Discretization(type,order) );
  bool warpMPDEFlag = false;
  RCP<N_MPDE_Manager> mpdeMgrRCPtr = rcp(new N_MPDE_Manager(cmdLine));
  N_MPDE_Discretization::Type type = N_MPDE_Discretization::Backward;
  int order = 1;
  RCP<N_MPDE_Discretization> discRCPtr = rcp(new N_MPDE_Discretization(type,order) );
  bool warpMPDEFlag = false;
  RCP<N_HB_Buidler>
    hbBuilder = rcp(new N_HB_Builder( mpdeMgrRCPtr, numBlocks, discRCPtr, warpMPDEFlag));
  return builder;
}
TEUCHOS_UNIT_TEST( HB_Builder,  createVector )  { … }
```

Sandia National Laboratories

Segfaults

```
RCP<N_HB_Builder>
createHBBuilder(int numBlocks, int numSolutionVars, int numStateVars)  {
 RCP<N_PDS_Comm> pdsCommRCPtr = rcp(new N_PDS_SerialComm);
 N_IO_CmdParse cmdLine;
 RCP<N_PDS_Manager> pdsMgrRCPtr;
 bool isSerial;
 bool procFlag;
 pdsMgrRCPtr = rcp(new N_PDS_Manager(isSerial,procFlag));
 cmdLine.registerParallelMgr(pdsMgrRCPtr);
  int iargs = 2;
  char *arg0 = "Xyce";
  char *arg1 = "foo.cir";
  char *cargs[iargs];
  cargs[0] = arg0;
  cargs[1] = arg1;
  cmdLine.parseCommandLine(iargs,cargs);
  RCP<N_MPDE_Manager> mpdeMgrRCPtr = rcp(new N_MPDE_Manager(cmdLine));
  N_MPDE_Discretization::Type type = N_MPDE_Discretization::Backward;
  int order = 1;
  RCP<N_MPDE_Discretization> discRCPtr = rcp(new N_MPDE_Discretization(type,order) );
  bool warpMPDEFlag = false;
  RCP<N_MPDE_Manager> mpdeMgrRCPtr = rcp(new N_MPDE_Manager(cmdLine));
  N_MPDE_Discretization::Type type = N_MPDE_Discretization::Backward;
  int order = 1;
  RCP<N_MPDE_Discretization> discRCPtr = rcp(new N_MPDE_Discretization(type,order) );
  bool warpMPDEFlag = false;
  RCP<N_HB_Buidler>
    hbBuilder = rcp(new N_HB_Builder( mpdeMgrRCPtr, numBlocks, discRCPtr, warpMPDEFlag));
 RCP<Epetra_Map> baseMap;
  int numGlobalElements = numSolutionVars;   int IndexBase = 0;
  RCP<Epetra_Comm> comm = rcp(pdsCommRCPtr->petraComm(),false);
  baseMap = rcp(new Epetra_Map(numGlobalElements,IndexBase,*comm));
  hbBuilder->generateMaps(*baseMap);
  // Now we have to call generateHBMaps or it will also segfault on createVector
  hbBuilder->generateHBMaps(*baseMap);
  return builder;
}
```

Sandia National Laboratories

Test passes!

```
$ ./XyceDemoUnitTest --show-test-details=ALL
Teuchos::GlobalMPISession::GlobalMPISession(): started serial run
***
*** Unit test suite ...
***
Sorting tests by group name then by test name …
Running unit tests ...
0. HB_Builder_createVector_UnitTest ...
 is_null(hbBuilder) = 0 == 0 : passed
 Teuchos::is_null(bvec) = 0 == 0 : passed
 bvec->blockCount() = 3 == 3 : passed
 bvec->blockSize() = 20 == 20 : passed
 [Passed]

Summary: total = 1, run = 1, passed = 1, failed = 0

End Result: TEST PASSED
```

Summary

- Unit tests should be fast and localize errors
- Unit tests should be first line of defense
- System tests are too slow and too global for interactive development

- Test Driven Development results in
  - Finding defects earlier
  - Faster development
  - Testable code
  - Unit tested code

Action Items

1. Write “Hello World” program in Teuchos Unit Test Harness
2. Pick a small block of your own code and unit test it.
3. Report feedback on this to me at tscoffe@sandia.gov

Thank you!

Sandia National Laboratories