COM 1205 Project Description
Winter 2003
Testing Addendum

Prepared by Karl Lieberherr and John Sung

Modified by Pengcheng Wu

Testing the Basket Example

This addendum to the COM1205 Project Description describes an addition to the testing procedure. This describes directions on testing the TraversalJ on something beside itself.

The Location of files

The basket example is in ~wupc/com1205/w03/proj/BasketMain.java. You should copy the files to a directory under your project directory. An Example of TraversalJ directory tree

```
Project Dir
gen
classes
interf
  CreateClassGraph.java & AspectJTraversal.java
basket
  BasketMain.java & t2.trv
traversal
  Generated traversal code for Basket
```

Generating the traversal for basket

To generate the traversal code for Basket Main.java, we need to compile the code with TraversalJ, run the code to generate the traversal , which should be like **t2.java**, then recompile the BasketMain.java with the generated traversal code. The procedures outlined below describe how to accomplish these 3 steps for testing TraversalJ.

Compiling with TraversalJ

- cd basket
- ajc ../gen/*.java ../interf/*.java *.java

Generating the Traversal

- cd basket
- java -classpath .:${CLASSPATH} BasketMain -d traversal t2.trv

Compiling with the traversal code

- Add code within the BasketMain.java to start the traversal
- cd basket/traversal
- ajc ../*.java *.java

Running the Basket Example with the generated traversal code

- java -classpath basket/traversal:${CLASSPATH} BasketMain

Windows Considerations

- All of the paths have to be with \ instead of /
- for the classpath option, ";%classpath" instead of ":${CLASSPATH}"

BasketMain.java

```
class Basket {
    Basket(Fruit _f, Pencil _p) { f = _f; p = _p; }
  Fruit f;
  Pencil p;
}
class Fruit {
    Fruit(Weight _w) { w = _w; }
    Weight w;

}
class Orange extends Fruit {
    Orange(Color _c) { super(null); c=_c;}
    Orange(Color _c, Weight _w) { super(_w); c = _c;}
  Color c;
}
class Pencil {}
class Color {
    Color(String _s) { s = _s;}
  String s;
}
class Weight{
    Weight(int _i) { i = _i;}
  int i;
}

class BasketMain {
  static public void main(String args[]) throws Exception {
      Basket b = new Basket(new Orange(new Color("orange"),
                            new Weight(5)),
                  new Pencil()
                  );
      b.t2();
  }
}
```

t2.trv

```
      // traversals for basket
     aspect t2 {^M
        declare strategy :toAll: "from * bypassing {-> ,tail, to
     * } to *";
         declare traversal: t2: intersect(toAll, “from Basket to
     Fruit”;
      }
```

t2.java - Traversal Generated by TraversalJ

```
// This file was generated by DAJ from t2.trv.

import edu.neu.ccs.demeter.*;

public aspect t2 {
  void Basket.t2(java.util.BitSet[] tokens) {
    { java.util.BitSet[] newTokens = { new java.util.BitSet() };
      if (tokens[0].get(0)) {
        newTokens[0].set(0);
      }
      if (!newTokens[0].isEmpty())
        if (f != null)
          t2_crossing_f(newTokens);
    }
  }
  void Basket.t2_crossing_f(java.util.BitSet[] tokens) {
    this.f.t2(tokens);
  }
  void Fruit.t2(java.util.BitSet[] tokens) {
  }
  void Orange.t2(java.util.BitSet[] tokens) {
  }
  before(java.util.BitSet[] tokens): call(void t2*(..)) &&
args(.., tokens) {
    System.out.println(thisJoinPoint + " " +
java.util.Arrays.asList(tokens));
  }
  void Basket.t2() {
    java.util.BitSet[] tokens = { new java.util.BitSet() };
    tokens[0].set(0);
    t2(tokens);
  }
} // t2
```

Output from running BasketMain with generated traversal

```
     \basket\trav>java -classpath .;%classpath% BasketMain
     call(void Basket.t2(BitSet[])) [{0}]
     call(void Basket.t2_crossing_f(BitSet[])) [{0}]
     call(void Fruit.t2(BitSet[])) [{0}]
```