

Integration of Demeter and AspectJ (DAJ)

John J. Sung

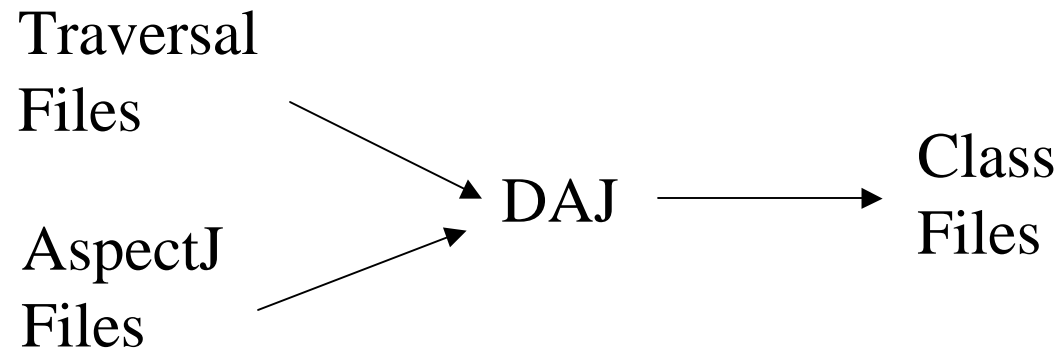
Outline

- Purpose of DAJ
- Syntax of Traversals
- Basket Example
- Development Process
- Conclusion

Purpose of DAJ

- Integration of Demeter ideas with AspectJ
 - Add notion of class graphs, traversals, visitors
- Minimize coupling with AspectJ compiler
 - Create an extension to AspectJ
 - Preprocess the files to generate AspectJ code
 - Use ajc as the back end of the compilation process

Overall Compilation Process



Traversal Files

- Files with .trv extension
- Contains Traversal Specifications
- Preprocessed to generate AspectJ code

Syntax of Traversals

- **ClassGraph Specification**
 - `ClassGraph cgvar;`
 - `ClassGraph cgvar = new ClassGraph(cg, “strategy”);`
- **Traversal Specification**
 - `declare traversal tvar : “strategy”;`
 - `declare traversal tvar(cgvar) : “strategy”;`

Traversal Aspects

- Traversal Aspect Specification
 - aspect aspectName {
 - class graph and/or traversal declarations
 - }

Traversal Syntax Example

```
aspect MyTraversal {
```

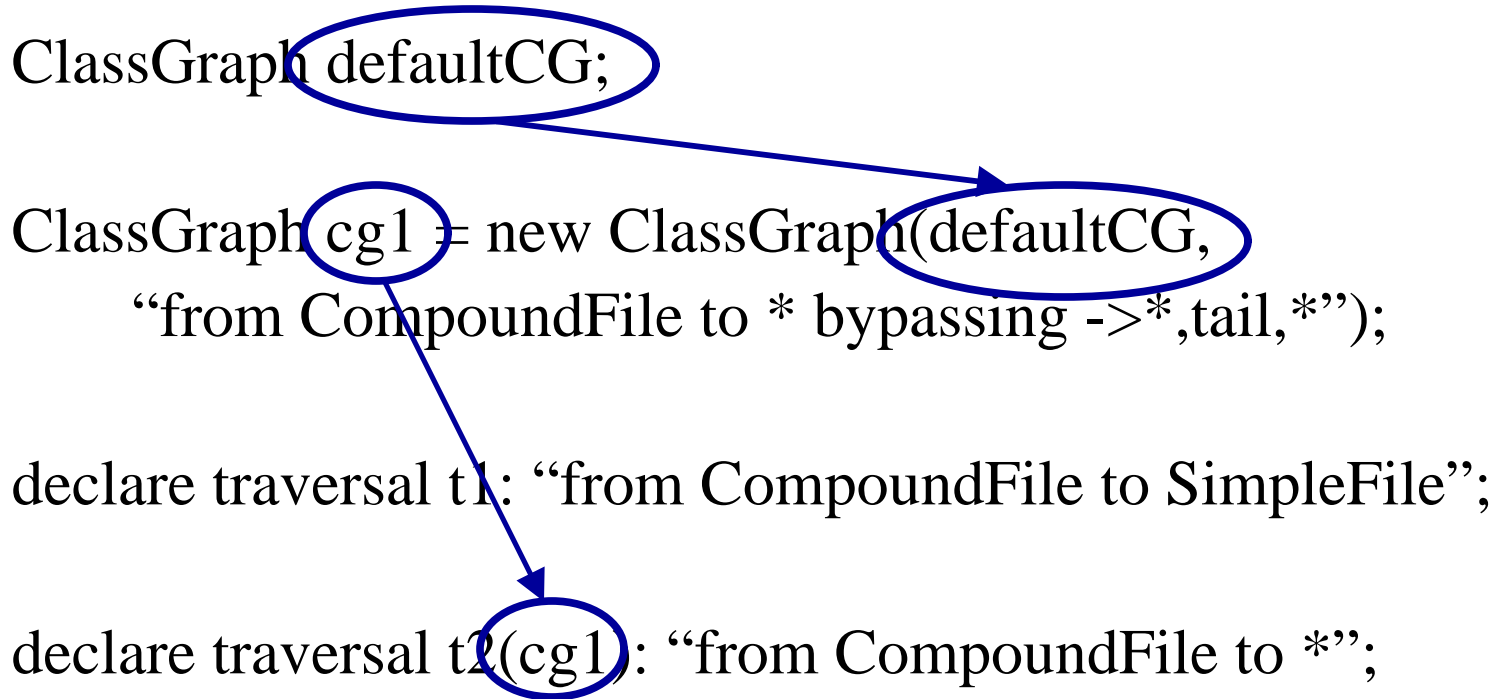
```
ClassGraph defaultCG;
```

```
ClassGraph cg1 = new ClassGraph(defaultCG,  
    "from CompoundFile to * bypassing ->*,tail,*");
```

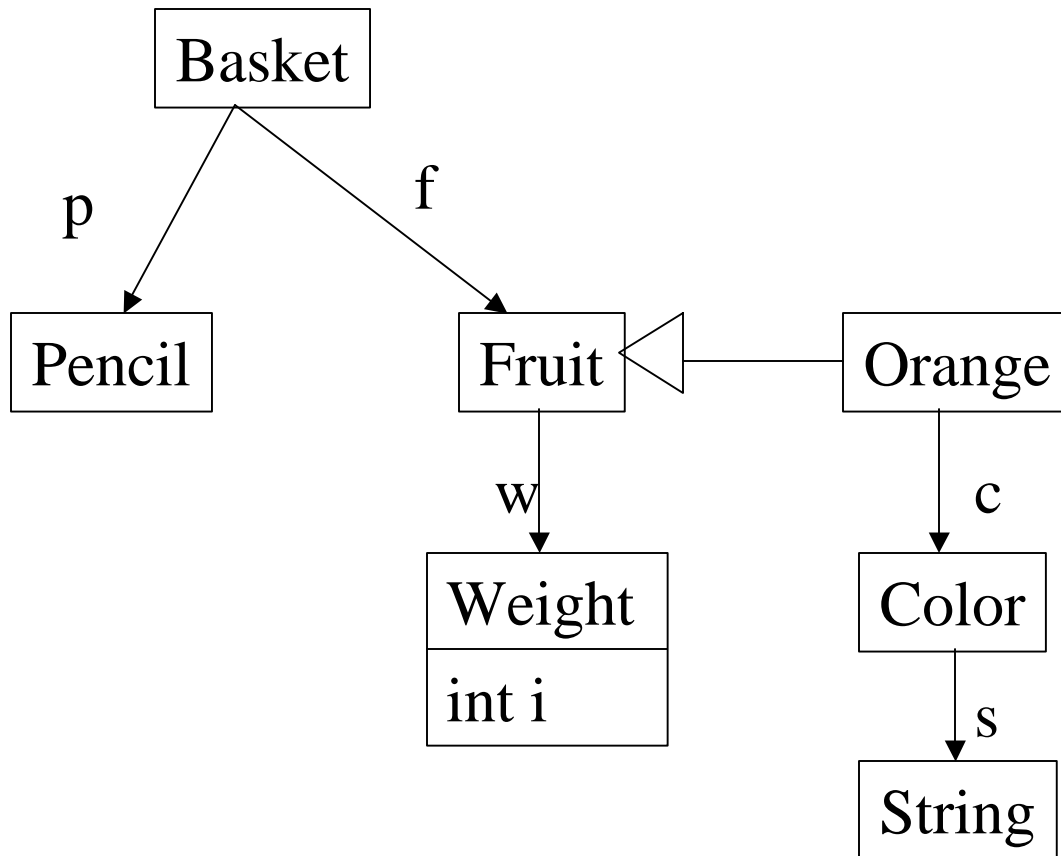
```
declare traversal t1: "from CompoundFile to SimpleFile";
```

```
declare traversal t2(cg1): "from CompoundFile to *";
```

```
}
```



Basket Example



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 Weight{
    Weight(int _i) { i = _i;}
    int i;
    int get_i() { return i; }
}

class Pencil {}

class Color {
    Color(String _s) { s = _s;}
    String s;
}
```

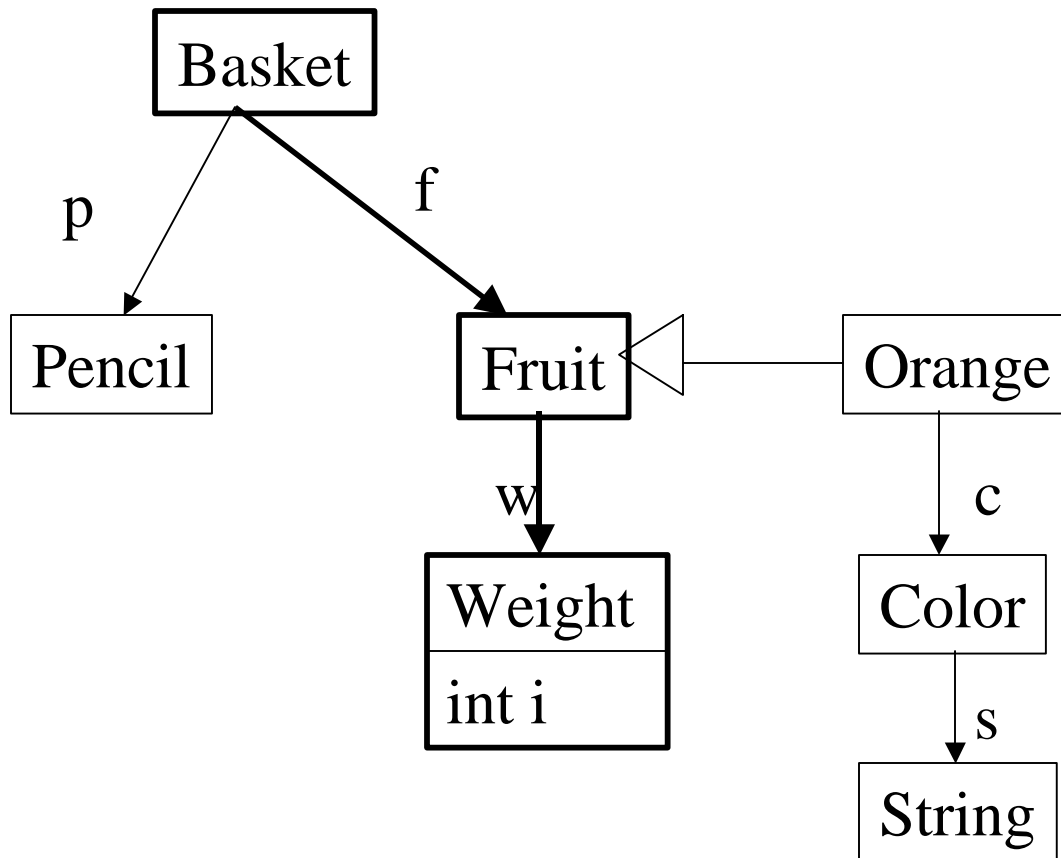
BasketMain.java

```
class BasketMain {  
    static public void main(String args[]) throws Exception {  
        Basket b = new Basket(  
            new Orange(  
                new Color("orange"),  
                new Weight(5),  
                new Pencil());  
        int totalWeight = b.totalWeight();  
        System.out.println("Total weight of basket = " +  
totalWeight);  
    }  
}
```

Traversals

- Count the total weight within the basket
- Traversal Strategy: “From Basket to Weight”
- Visitor: Add up all the values within Weight

Basket Example Traversal Graph



BasketTraversal.trv

```
// traversals for basket
aspect BasketTraversal {
    ClassGraph default;
    ClassGraph myClassGraph = new ClassGraph(default,
        "from Basket to * bypassing {->*,*,java.lang.String }");
    declare traversal t2(myClassGraph) : "from Basket to Weight";
}
```

BasketTraversal.java

```
public aspect BasketTraversal {  
    // traversal t2 : {source: Basket -> target: Weight} with { }  
    public void Basket.t2(){  
        if (f != null) t2_crossing_f();  
    }  
    public void Basket.t2_crossing_f() { f.t2();}  
    public void Fruit.t2(){  
        if (w != null) t2_crossing_w();  
    }  
    public void Fruit.t2_crossing_w() { w.t2();}  
    public void Weight.t2(){  
    }  
}
```

```
    public void Orange.t2(){  
        super.t2();  
    }  
    pointcut pointcut_t2() : call(public void t2*());  
    before () : pointcut_t2 () {  
        System.out.println(thisJoinPoint);  
    }  
} // BasketTraversal
```

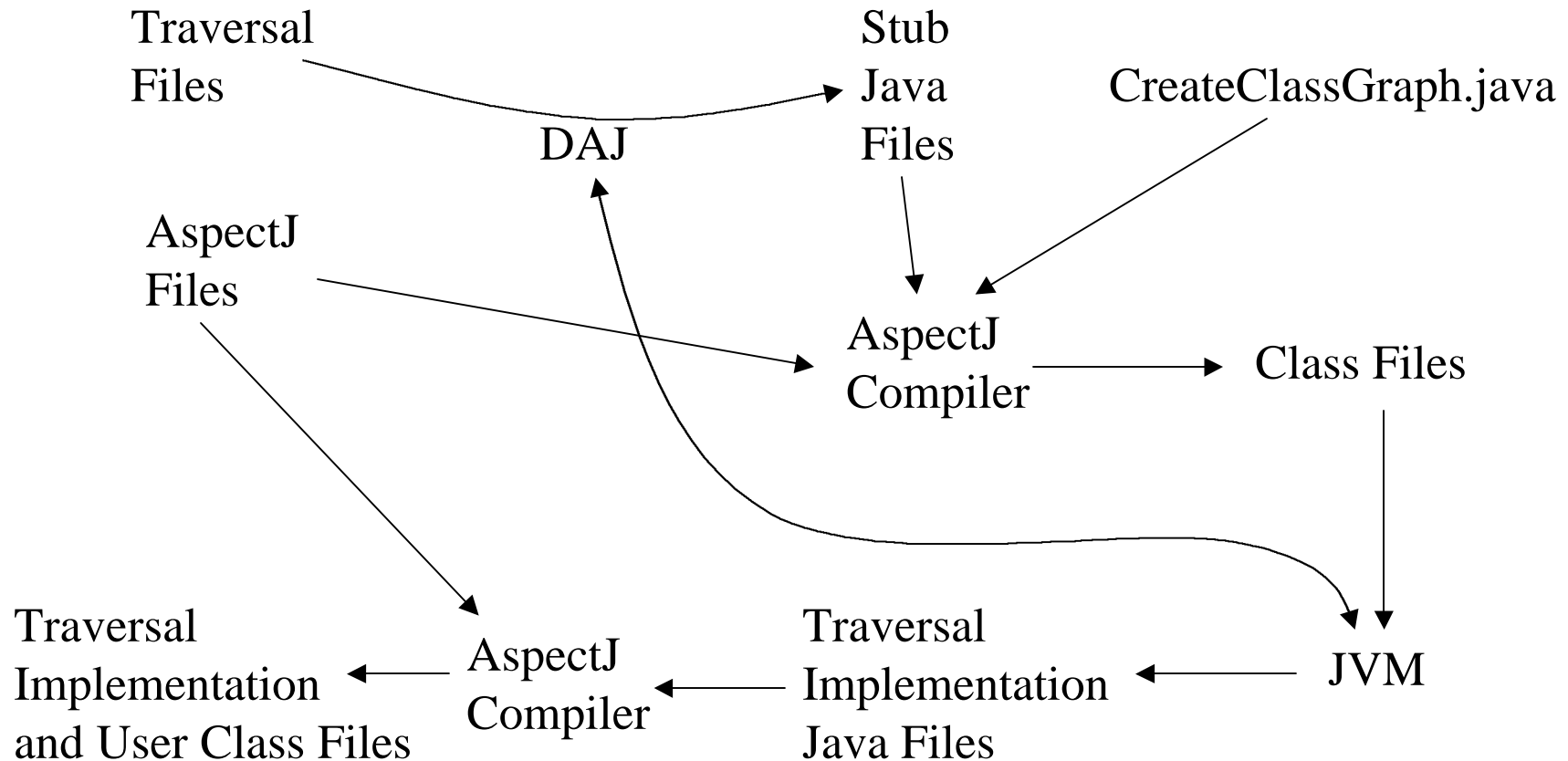
BasketMainCount.java

```
// the aspect for counting the total weight of the basket
aspect BasketMainCount {
    static int returnVal;
    int Basket.totalWeight() {
        returnVal = 0;
        t2();
        return returnVal;
    }
    pointcut t2WeightPC(Weight weight) : call(* *t2*()) && target(weight);
    before(Weight weight) : t2WeightPC(weight) {
        returnVal += weight.get_i();
    }
}
```

Compilation Process

- Parse the .trv files and generate stubs
- Compile user .java, stubs and CreateClassGraph.java
- Run the user code with stubs to generate the traversal code using DJ
- Compile user .java with generated traversals

Compilation Process Detail



Conclusion

- Integrated Demeter style traversals with AspectJ
 - It has syntax similar to DJ and AspectJ
 - Generates traversals
- Web address
 - to be added later