Combining Expressive Pointcut Language and Instance Level Aspect Weaving

Eos: A prototype for .NET framework.

Overview

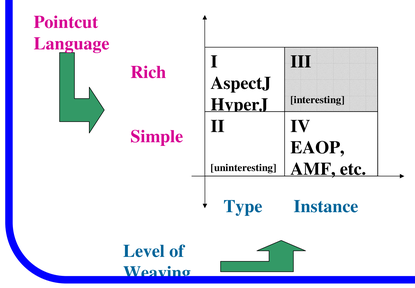
Problem: Instance level weaving and rich pointcut language exist as mutually exclusive features in current aspect-oriented languages and approaches.

Objective: To analyze the feasibility and utility of AO languages having both elements.

Current Status: Eos, our prototype for .NET framework provides instance and type level weaving with a reasonably rich pointcut language.

Future Work and Issues: Richer constructs to specify which instance to weave, join point Encapsulation, richer pointcut language, concurrency, cross language aspect weaving.

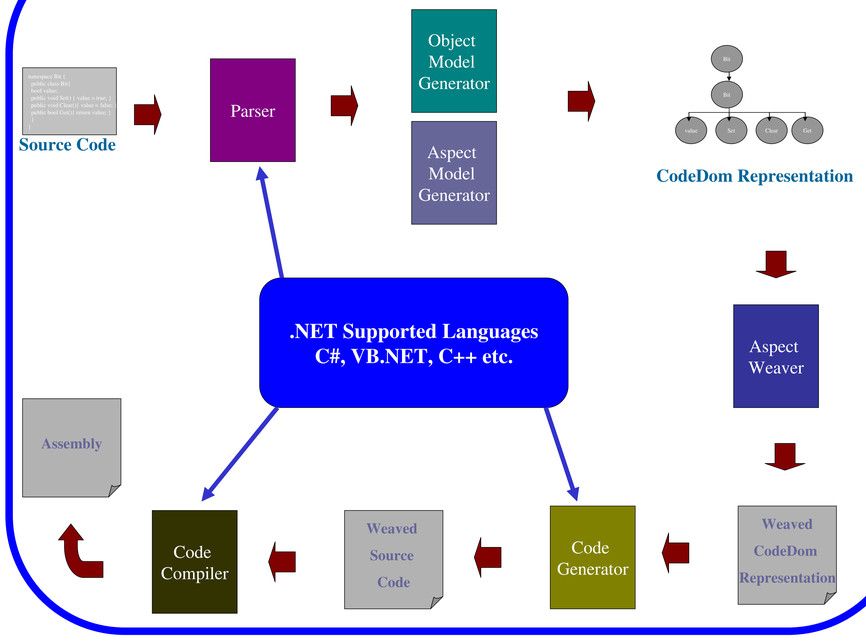
Motivation



Approach

- Provides constructs to specify level of weaving.
- Utilizes developer's hint in the form of pointcut expressions to optimize instance level aspect weaving.
- Provides a reasonably rich pointcut language.

Eos Architecture and Working



Example

Behavioral ER model of a Bijection

```
namespace SET {
public class Set : System.Collections.CollectionBase
{
    // If  this is a unique element, insert it and return true else return false.
    public bool Insert(Element element){ … }
    // If the element is in the set, remove it and return true, else return false.
    public bool Remove(Element element){ … }
   // If the element is in the set, return it, else return null.
    public Element Retrieve(string Name){ … }
}
}
```

```
public instancelevel aspect Bijection  /// BIJECTION
{
    bool busy; Set A; Set B;

    pointcut CallInsert():call(public Set.Insert());
    pointcut CallRemove():call(public Set.Remove());

    public void Relate(Set A, Set B){
        addObject(A); addObject(B);
            this.A = A; this.B = B;
    }

    after():CallInsert(){
        if(!busy && (bool) thisJoinPoint.getReturnValue()) {
            busy = true;
        /// Bijection Logic: Ensures bijection between the sets.
        …
        busy = false;
        }
    }

    after():CallRemove(){
        if(!busy && (bool) thisJoinPoint.getReturnValue()) {
            busy = true;
            /// Bijection Logic: Ensures bijection between the sets.
            …
            busy = false;
            }
    }
} /* End Bijection */ } /* End namespace SET */
```

Weave these instances.

Pointcut expression acts as hint to the weaver

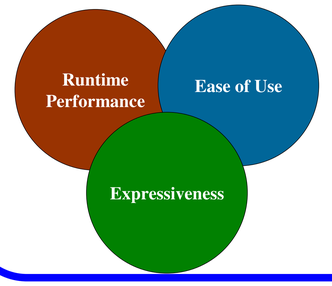
Advice do not need any code to distinguish between instances

Constructs	Current Status	Next Milestone
Join points available	Field Get and Set, Method Call and Execution	Exceptions, Object Initialization, Destruction, etc.
Instance level weaving	Aspects	Pointcuts, Advices.
Quantifying instances	addObject, removeObject	Explicit Constructs

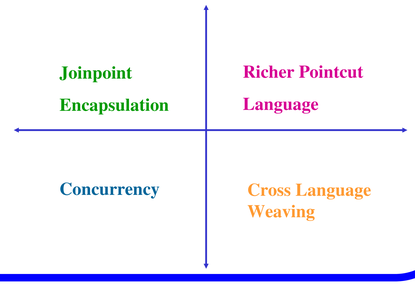
Limitations

- Limited join point model.
- Not able to represent all constructs in C# due to the limitations of CodeDom representation.
- Very primitive quantification mechanism for instance level aspect weaving.

Evaluation



Future Work and Issues



University of Virginia

Department of Computer Science

Hridesh Rajan and Kevin Sullivan

http://www.cs.virginia.edu/~eos

This work was supported in part by the National Science Foundation under grant ITR-0086003.