Reuseware Tutorial 1:

Weaving in Aspect-Oriented Modeling with Reuseware for different Modelling Languages

Reuseware Principles

- Reuseware is...
  - A language independent modularisation approach [1]
  - A framework: Reuseware Composition Framework [2]
- Common concepts for different composition systems for arbitrary languages
  - Easy specification of new composition techniques and porting of techniques from one language to another
  - Reuse of composition tooling
  - Tailor tooling for composition system and language
- Support features of aspect-oriented systems
  - Support homogeneous cross-cuts (and quantification)
  - Support heterogeneous cross-cuts

[1] On Language-Independent Model Modularisation,Transactions on Aspect-Oriented Development, 2008
[2] http://reuseware.org

Reuseware Principles - Core Concepts

- Model Fragments
  - (Partial) models that may contain variation points
  - Offer a ***Composition Interface***
  - ***Composition Interface*** consists of ***Ports***
  - ***Ports*** point at elements of the model fragment that can be accessed for composition
  - One ***Port*** can point at several elements at arbitrary places in the model fragment (heterogeneous crosscut)
  - Similar Ports can be joined to one ***HomogeneousPort*** (homogeneous crosscut)
- Composition Programs
  - Define ***composition links*** between Ports
  - Are executed to produce a composed model where model fragments are woven at the elements pointed out by the linked Ports

Reuseware Principles - Core Concepts

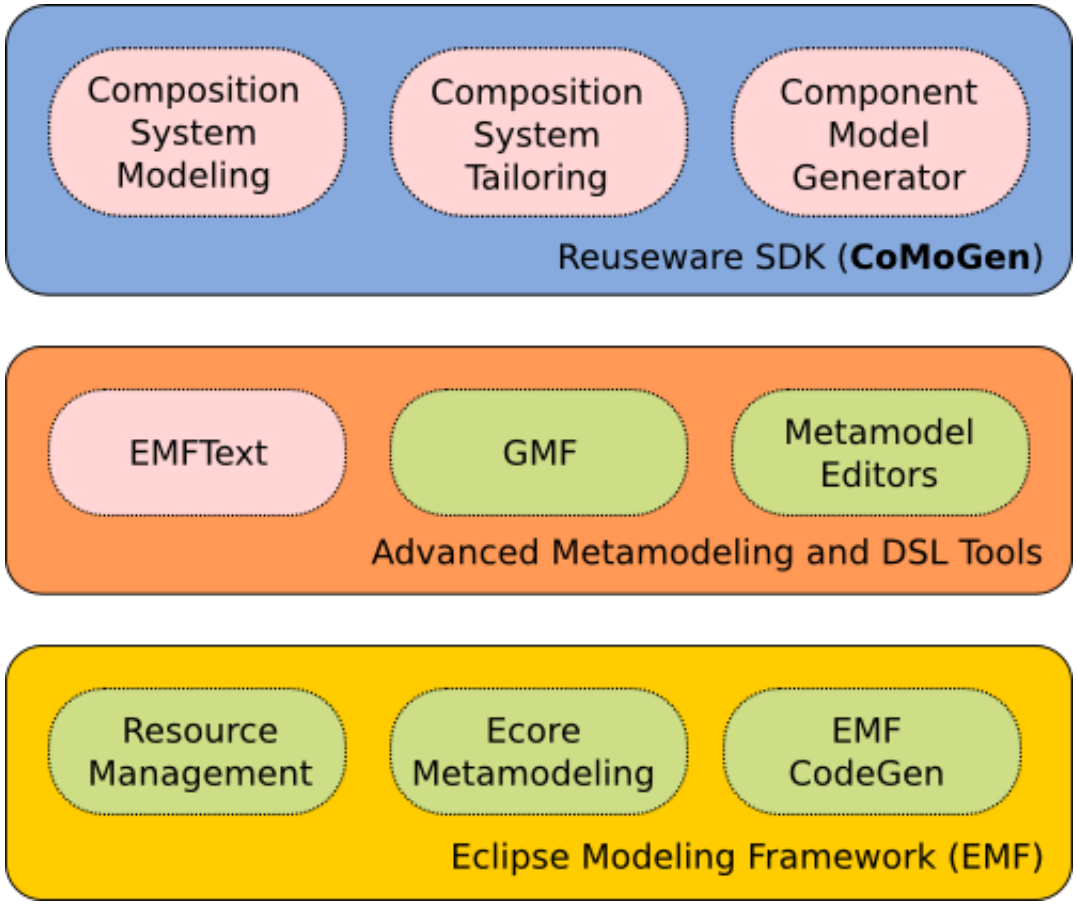
- Composition Systems
  - Define modularisation concepts (e.g., Modules, Packages, Aspects)
  - Define relations between modularisation concepts (e.g, an aspect relates to a core)
- Reuse extensions (for a particular language)
  - Define how modularization concepts defined in a composition system are realized in a concrete language
  - Define which ports are related to which model elements of a model fragment

Reuseware Composition Framework

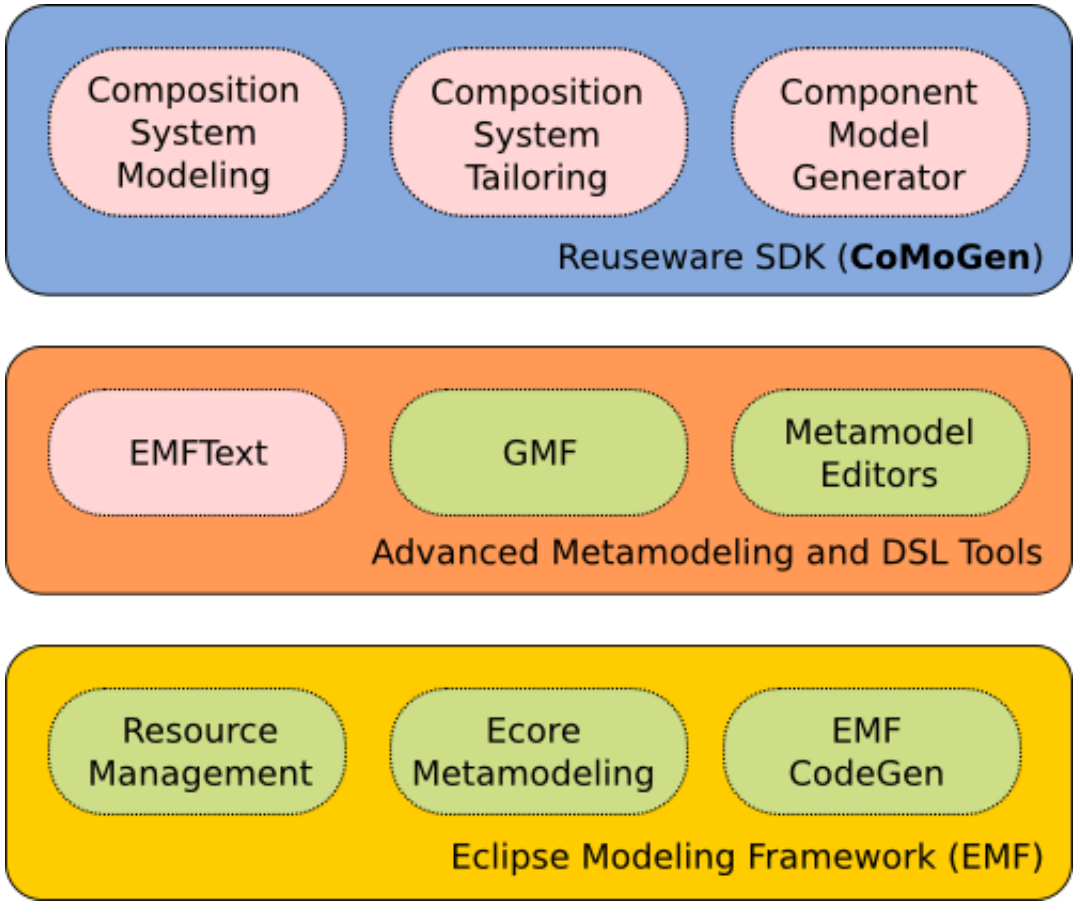
reuseware composition framework

- CoMoGen (Reuseware SDK)
  - Enables developers to define new composition systems
  - Addition to other language engineering (metamodelling /DSL) tools to define modularisation aspect of a language
- CoCoNut (Reuseware Runtime)
  - Provides language independent composition engine
  - Provides language independent component repository
  - Provides language independent composition program editor
  - Composition systems defined with CoMoGen plug into CoCoNut and tailor the above functionality

CoMoGen (Reuseware SDK)



CoMoGen (Reuseware SDK)

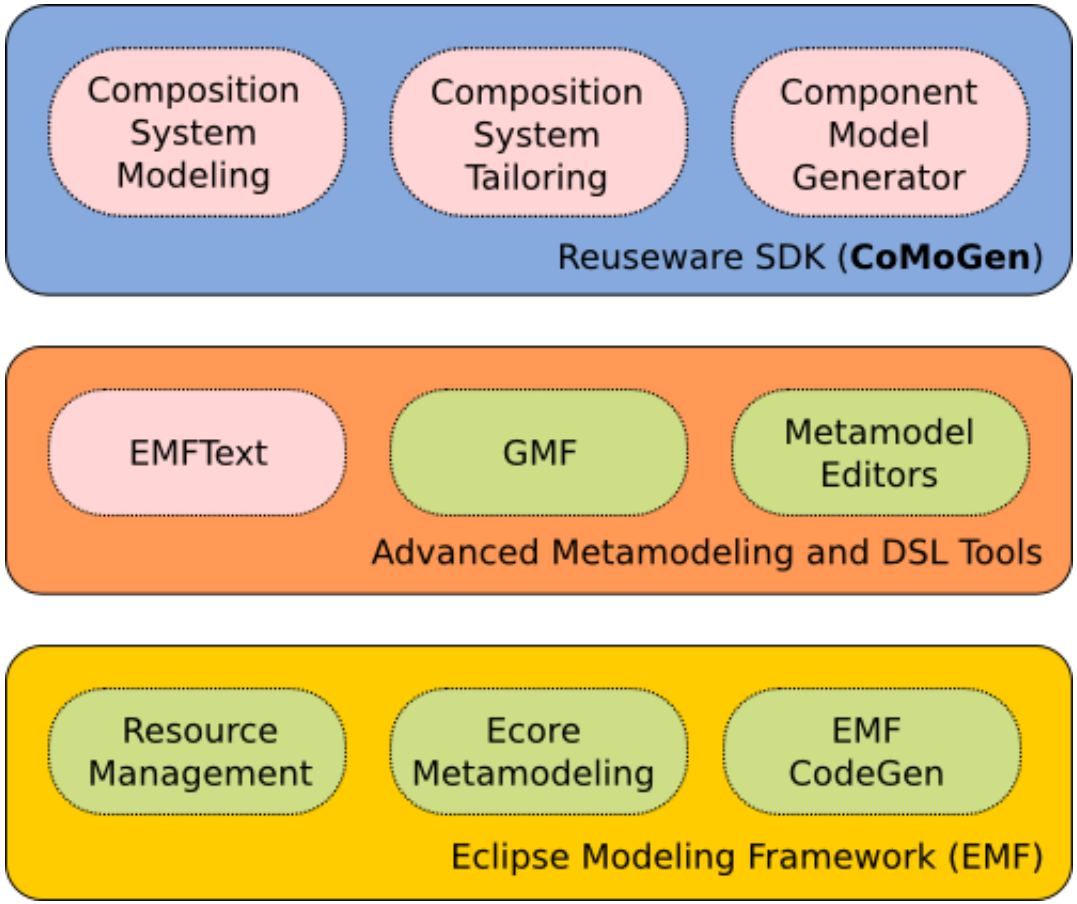


Reuseware builds on EMF and works together with other EMF-based tools

CoMoGen (Reuseware SDK)

Metamodeling tools can be used to define a language and tools for the language (examples are shown here)

Reuseware builds on EMF and works together with other EMF-based tools

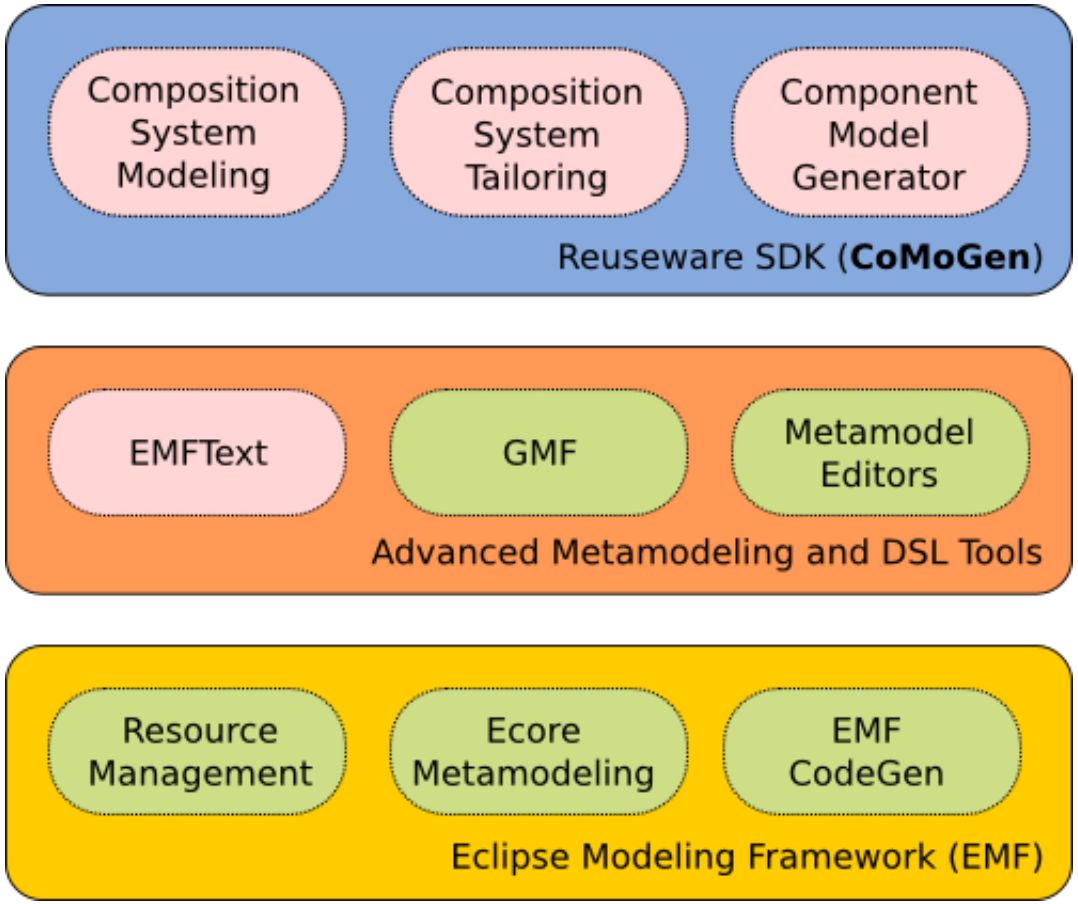


CoMoGen (Reuseware SDK)

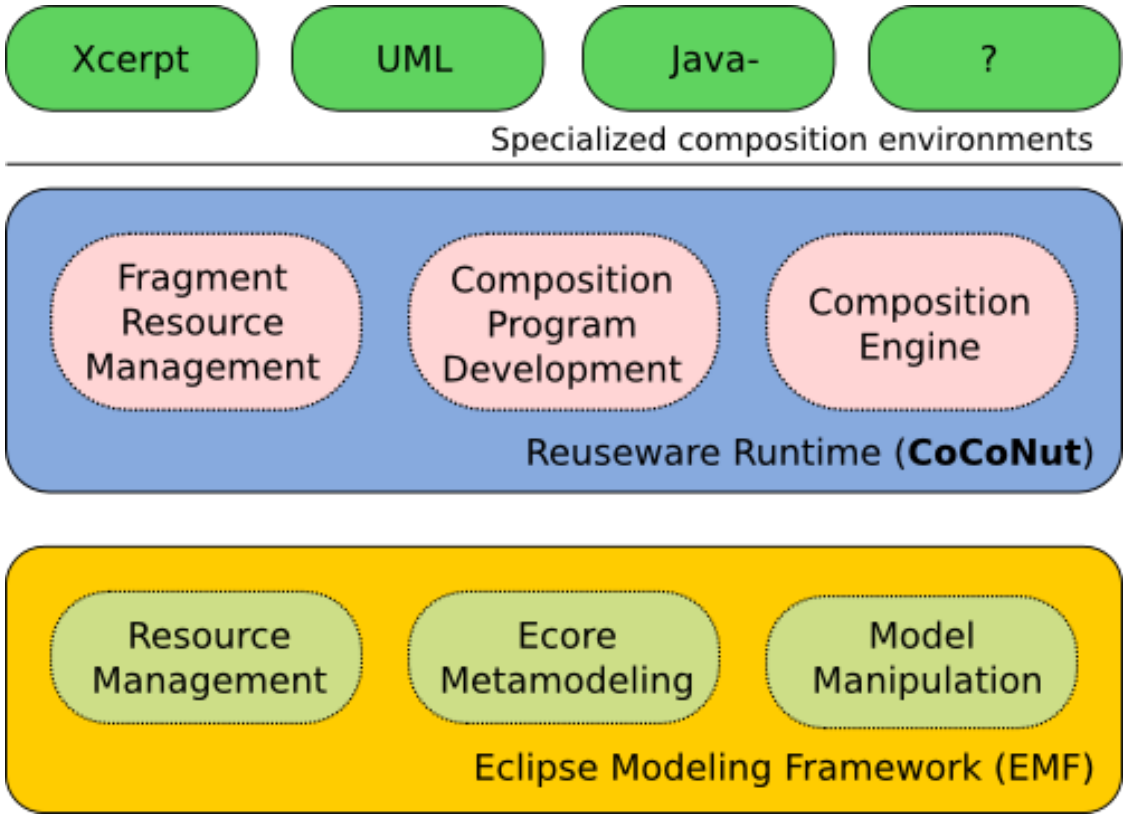
With CoMoGen, model composition systems can be modelled based on a prior defined metamodel

Metamodeling tools can be used to define a language and tools for the language (examples are shown here)

Reuseware builds on EMF and works together with other EMF-based tools



CoCoNut (Reuseware Runtime)

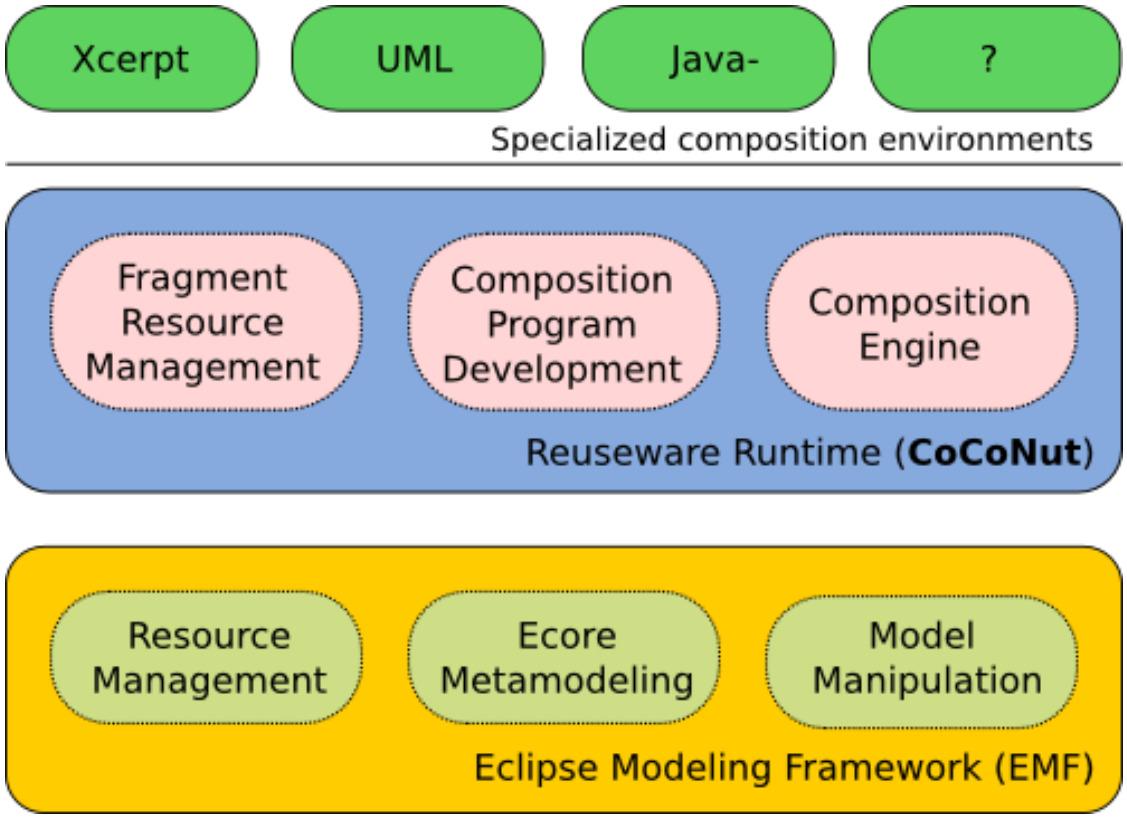


Reuseware builds on EMF and works together with other EMF-based tools

CoCoNut (Reuseware Runtime)

The three language-independent features of CoCoNut can be used with every composition system

Reuseware builds on EMF and works together with other EMF-based tools

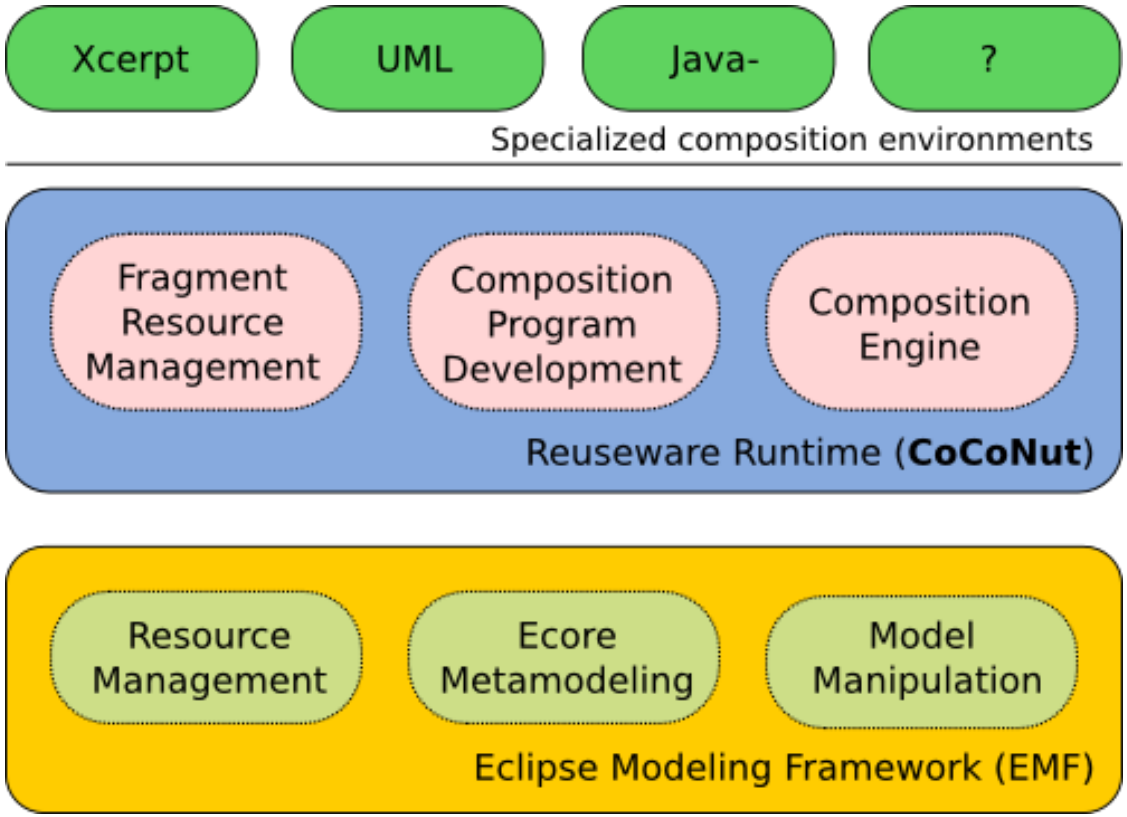


CoCoNut (Reuseware Runtime)

Specific composition systems defined with CoMoGen plug into CoCoNut

The three language-independent features of CoCoNut can be used with every composition system

Reuseware builds on EMF and works together with other EMF-based tools

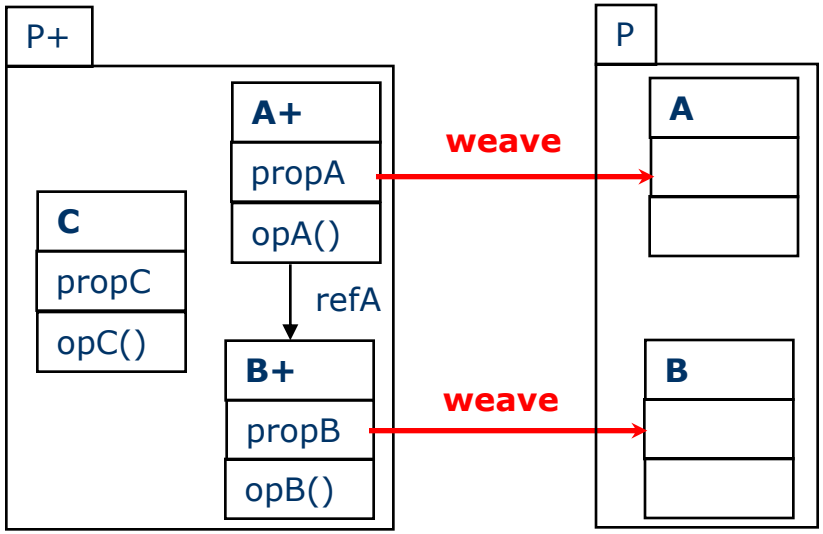


Reuseware Use Case: Class Weaving

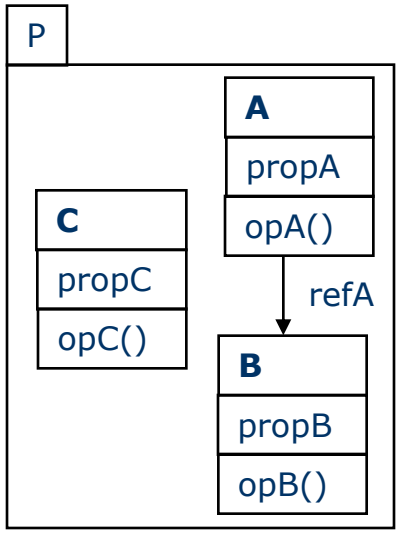
- Demonstrates CoMoGen Features
  - How to define a concrete composition system
  - How to use the composition system with different language
- Demonstrates CoCoNut Features
  - Using the composition program editor
  - Using the model fragment repository
  - Using the composition engine

Reuseware Use Case: Class Weaving

- Based on Class Concept found in many languages
  - A Class usually has operations, properties and references to other classes and is contained in a package
- Weaving Class A+ (Advice) into Class A means:
  - All operations, properties and references of A+ are added to A
  - if any of the new elements of A points at a class B+ which is woven into another class B, the pointers are redirected to B
  - Classes that are not source of a weaving should be copied completely



Model fragments + weaving description



Woven (composed) Model

Reuseware Use Case: ClassWeaving Composition System

- A composition system defines
  - Fragment roles
    - Role a model fragment plays in the modularisation (e.g., aspect or core)
    - Fragment roles collaborate through associations between ports
  - Static ports
    - Defined for one fragment role
    - Each fragment playing the role has to offer the port
  - Dynamic ports
    - Defined for one fragment role
    - Each fragment playing the role can offer several of these ports
  - Contribution Associations
    - Defines that two ports are related
    - Executing a composition link between the two ports will trigger the copying of model elements
  - Configuration Associations
    - Defines that two ports are related
    - Executing a composition link between the two ports will NOT trigger the copying of model elements

Reuseware Use Case: ClassWeaving Composition System

```
compositionsystem ClassWeaving {

  fragment role Core {
    static port Container;
    dynamic port Classes;
  }

  fragment role Aspect {
    static port Content;
    dynamic port Advices;
  }


  contribution Aspect.Content --> Core.Container;

  contribution Aspect.Advice --> Core.Class;
}
```

A **Core** acts as container for additional content (**Container**); it contains **Classes** which should be individually accessible for extension (therefore the number of ports is dynamic - it depends on the number of existing classes)

Reuseware Use Case: ClassWeaving Composition System

```
compositionsystem ClassWeaving {

  fragment role Core {
    static port Container;
    dynamic port Classes;
  }

  fragment role Aspect {
    static port Content;
    dynamic port Advices;
  }


  contribution Aspect.Content --> Core.Container;

  contribution Aspect.Advice --> Core.Class;
}
```

A **Core** acts as container for additional content (**Container**); it contains **Classes** which should be individually accessible for extension (therefore the number of ports is dynamic - it depends on the number of existing classes)

An **Aspect** offers additional **Content**; it contains **Advices** as extensions for classes which should be individually accessible

Reuseware Use Case: ClassWeaving Composition System

```
compositionsystem ClassWeaving {

  fragment role Core {
    static port Container;
    dynamic port Classes;
  }

  fragment role Aspect {
    static port Content;
    dynamic port Advices;
  }

  contribution Aspect.Content --> Core.Container;

  contribution Aspect.Advice --> Core.Class;
}
```

A **Core** acts as container for additional content (**Container**); it contains **Classes** which should be individually accessible for extension (therefore the number of ports is dynamic - it depends on the number of existing classes)

An **Aspect** offers additional **Content**; it contains **Advices** as extensions for classes which should be individually accessible

A **Content** contributes to a **Container**

Reuseware Use Case: ClassWeaving Composition System

```
compositionsystem ClassWeaving {

  fragment role Core {
    static port Container;
    dynamic port Classes;
  }

  fragment role Aspect {
    static port Content;
    dynamic port Advices;
  }


  contribution Aspect.Content --> Core.Container;

  contribution Aspect.Advice --> Core.Class;
}
```

A **Core** acts as container for additional content (**Container**); it contains **Classes** which should be individually accessible for extension (therefore the number of ports is dynamic - it depends on the number of existing classes)

An **Aspect** offers additional **Content**; it contains **Advices** as extensions for classes which should be individually accessible

A **Content** contributes to a **Container**

An **Advice** contributes to a **Class**

Reuseware Use Case: ClassWeaving Reuse Extensions

- A Reuse Extension defines
  - How a composition interface define by a fragment role (which is defined in a composition system) is linked to the content of a model fragment
  - Each port links to a set of model elements treated as:
    - **Prototype**: Element that can be copied with its contained elements
    - **Anchor**: Element that can be referenced by other elements
    - **Hook**: Variation point where Prototypes can be put
    - **Slot**: Variation point where Anchors can be put
- For ClassWeaving we define
  - A reuse extension for Ecore
  - A reuse extension for UML

Reuseware Use Case: ClassWeaving for Ecore

```
reuseextension ClassWeavingEcore
implements ClassWeaving
epackages <http://www.eclipse.org/emf/2002/Ecore>
rootclass EPackage {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      EPackage.eClassifiers is hook {}
    }
    port Class {
      EClass.eOperations is hook {
        port expr = $name$
      }
      EClass.eStructuralFeatures is hook {
        port expr = $name$
      }
      EClass is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      EPackage.eClassifiers is prototype {}
    }
    port Advice {
      EClass.eOperations is prototype {
        port expr = $name$
      }
      EClass.eStructuralFeatures is prototype {
        port expr = $name$
      }
      EClass is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **Ecore** language (using the URI of the Ecore metamodel)

Reuseware Use Case: ClassWeaving for Ecore

```
reuseextension ClassWeavingEcore
implements ClassWeaving
epackages <http://www.eclipse.org/emf/2002/Ecore>
rootclass EPackage {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      EPackage.eClassifiers is hook {}
    }
    port Class {
      EClass.eOperations is hook {
        port expr = $name$
      }
      EClass.eStructuralFeatures is hook {
        port expr = $name$
      }
      EClass is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      EPackage.eClassifiers is prototype {}
    }
    port Advice {
      EClass.eOperations is prototype {
        port expr = $name$
      }
      EClass.eStructuralFeatures is prototype {
        port expr = $name$
      }
      EClass is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **Ecore** language (using the URI of the Ecore metamodel)

A core can be extended with new classes by extending the **eClassifiers** reference of the **EPackage** metaclass

Reuseware Use Case: ClassWeaving for Ecore

```
reuseextension ClassWeavingEcore
implements ClassWeaving
epackages <http://www.eclipse.org/emf/2002/Ecore>
rootclass EPackage {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      EPackage.eClassifiers is hook {}
    }
    port Class {
      EClass.eOperations is hook {
        port expr = $name$
      }
      EClass.eStructuralFeatures is hook {
        port expr = $name$
      }
      EClass is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      EPackage.eClassifiers is prototype {}
    }
    port Advice {
      EClass.eOperations is prototype {
        port expr = $name$
      }
      EClass.eStructuralFeatures is prototype {
        port expr = $name$
      }
      EClass is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **Ecore** language (using the URI of the Ecore metamodel)

A core can be extended with new classes by extending the **eClassifiers** reference of the **EPackage** metaclass

Extending a class means extending the **eOperations** and **eStructuralFeatrues** references of an **EClass**; An **EClass** itself will be referenced (anchor) as replacement for advices; Each **EClass** and its references are accessible through a port identified by the class name

Reuseware Use Case: ClassWeaving for Ecore

```
reuseextension ClassWeavingEcore
implements ClassWeaving
epackages <http://www.eclipse.org/emf/2002/Ecore>
rootclass EPackage {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      EPackage.eClassifiers is hook {}
    }
    port Class {
      EClass.eOperations is hook {
        port expr = $name$
      }
      EClass.eStructuralFeatures is hook {
        port expr = $name$
      }
      EClass is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      EPackage.eClassifiers is prototype {}
    }
    port Advice {
      EClass.eOperations is prototype {
        port expr = $name$
      }
      EClass.eStructuralFeatures is prototype {
        port expr = $name$
      }
      EClass is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **Ecore** language (using the URI of the Ecore metamodel)

A core can be extended with new classes by extending the **eClassifiers** reference of the **EPackage** metaclass

Extending a class means extending the **eOperations** and **eStructuralFeatrues** references of an **EClass**; An **EClass** itself will be referenced (anchor) as replacement for advices; Each **EClass** and its references are accessible through a port identified by the class name

The **eClassifiers** reference of the **EPackage** metaclass defines the content of an aspect

Reuseware Use Case: ClassWeaving for Ecore

```
reuseextension ClassWeavingEcore
implements ClassWeaving
epackages <http://www.eclipse.org/emf/2002/Ecore>
rootclass EPackage {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      EPackage.eClassifiers is hook {}
    }
    port Class {
      EClass.eOperations is hook {
        port expr = $name$
      }
      EClass.eStructuralFeatures is hook {
        port expr = $name$
      }
      EClass is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      EPackage.eClassifiers is prototype {}
    }
    port Advice {
      EClass.eOperations is prototype {
        port expr = $name$
      }
      EClass.eStructuralFeatures is prototype {
        port expr = $name$
      }
      EClass is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **Ecore** language (using the URI of the Ecore metamodel)

A core can be extended with new classes by extending the **eClassifiers** reference of the **EPackage** metaclass

Extending a class means extending the **eOperations** and **eStructuralFeatrues** references of an **EClass**; An **EClass** itself will be referenced (anchor) as replacement for advices; Each **EClass** and its references are accessible through a port identified by the class name

The **eClassifiers** reference of the **EPackage** metaclass defines the content of an aspect

An advice is modelled as an instance of **EClass**; The **eOperations** and **eStructuralFeatrues** references are exported; The **EClass** itself is to be replaced (slot); Each advice **EClass** and its references are accessible through a port identified by the class name

Reuseware Use Case: ClassWeaving for UML

```
reuseextension ClassWeavingUML
implements ClassWeaving
for <http://www.eclipse.org/uml2/2.1.0/UML>
rootclass Model {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      Package.packagedElement is hook {}
    }
    port Class {
      Class.ownedOperation is hook {
        port expr = $name$
      }
      Class.ownedAttribute is hook {
        port expr = $name$
      }
      Class is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      Package.packagedElement is prototype {}
    }
    port Advice {
      Class.ownedOperation is prototype {
        port expr = $name$
      }
      Class.ownedAttribute is prototype {
        port expr = $name$
      }
      Class is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **UML** language (using the URI of the UML metamodel)

Reuseware Use Case: ClassWeaving for UML

```
reuseextension ClassWeavingUML
implements ClassWeaving
for <http://www.eclipse.org/uml2/2.1.0/UML>
rootclass Model {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      Package.packagedElement is hook {}
    }
    port Class {
      Class.ownedOperation is hook {
        port expr = $name$
      }
      Class.ownedAttribute is hook {
        port expr = $name$
      }
      Class is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      Package.packagedElement is prototype {}
    }
    port Advice {
      Class.ownedOperation is prototype {
        port expr = $name$
      }
      Class.ownedAttribute is prototype {
        port expr = $name$
      }
      Class is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **UML** language (using the URI of the UML metamodel)

In UML, the packagedElement reference contains **Classes** and **Associations**...

Reuseware Use Case: ClassWeaving for UML

```
reuseextension ClassWeavingUML
implements ClassWeaving
for <http://www.eclipse.org/uml2/2.1.0/UML>
rootclass Model {
 fragment role Core if $not name.startsWith('advice')$ {
    port Container {
      Package.packagedElement is hook {}
    }
    port Class {
      Class.ownedOperation is hook {
        port expr = $name$
      }
      Class.ownedAttribute is hook {
        port expr = $name$
      }
      Class is anchor {
        port expr = $name$
      }
    }
  }

 fragment role Aspect if $name.startsWith('advice')$ {
    port Content {
      Package.packagedElement is prototype {}
    }
    port Advice {
      Class.ownedOperation is prototype {
        port expr = $name$
      }
      Class.ownedAttribute is prototype {
        port expr = $name$
      }
      Class is slot {
        port expr = $name$
      }
    }
  }
}
```

The **ClassWeaving** composition system is implemented for the **UML** language (using the URI of the UML metamodel)

In UML, the packagedElement reference contains **Classes** and **Associations**...

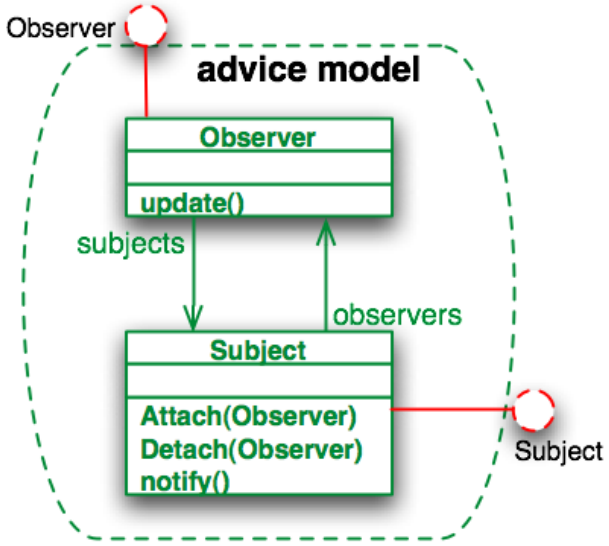
...the class contains only **ownedAttributes** (and no references or associaitons)

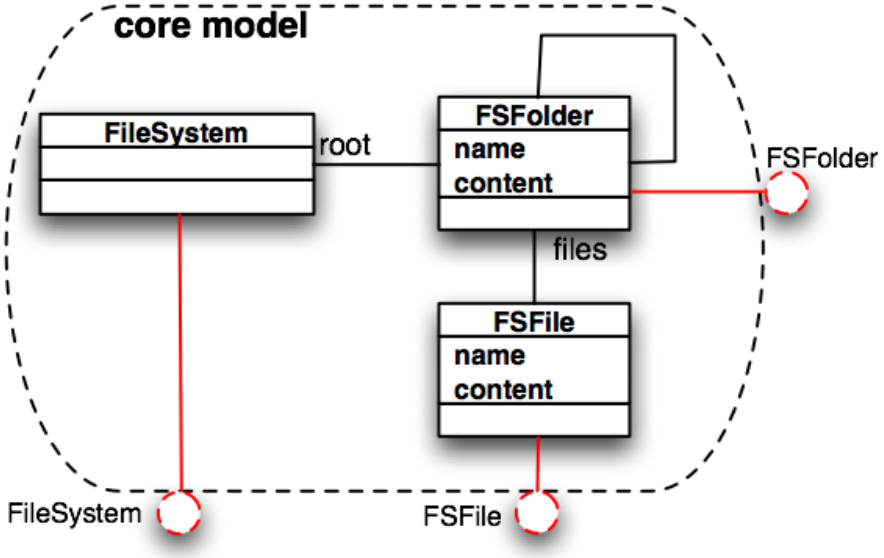
Reuseware Use Case: Using Ecore ClassWeaving

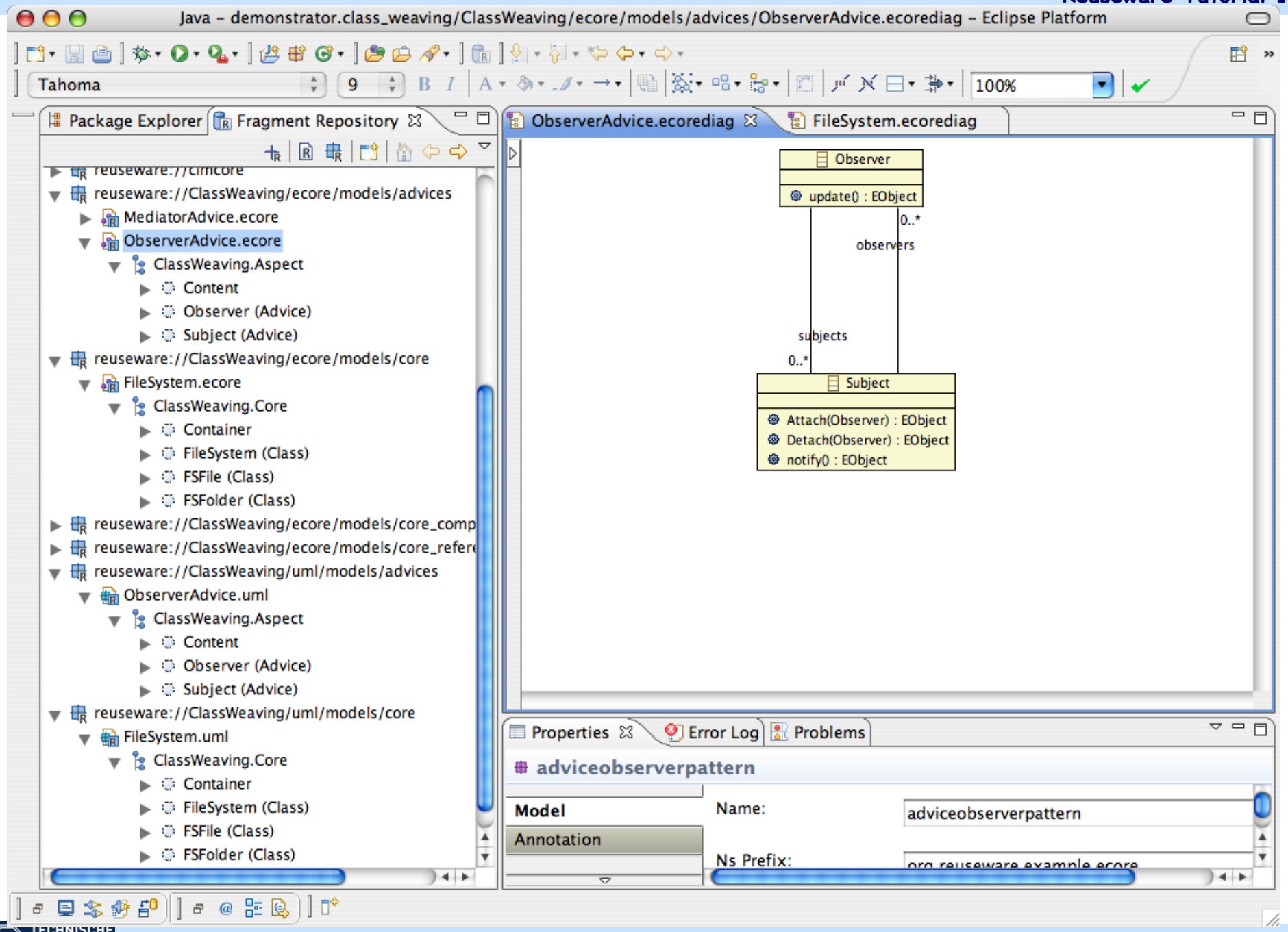
- Activate in CoConut:
  - ClassWeaving composition system
  - ClassWeaving for Ecore reuse extendsion
- Activation through
  - Eclipse plugin extension point
  - Dynamically at runtime
- Enables use of
  - Fragment repository
  - Composition program editor
  - Composition engine

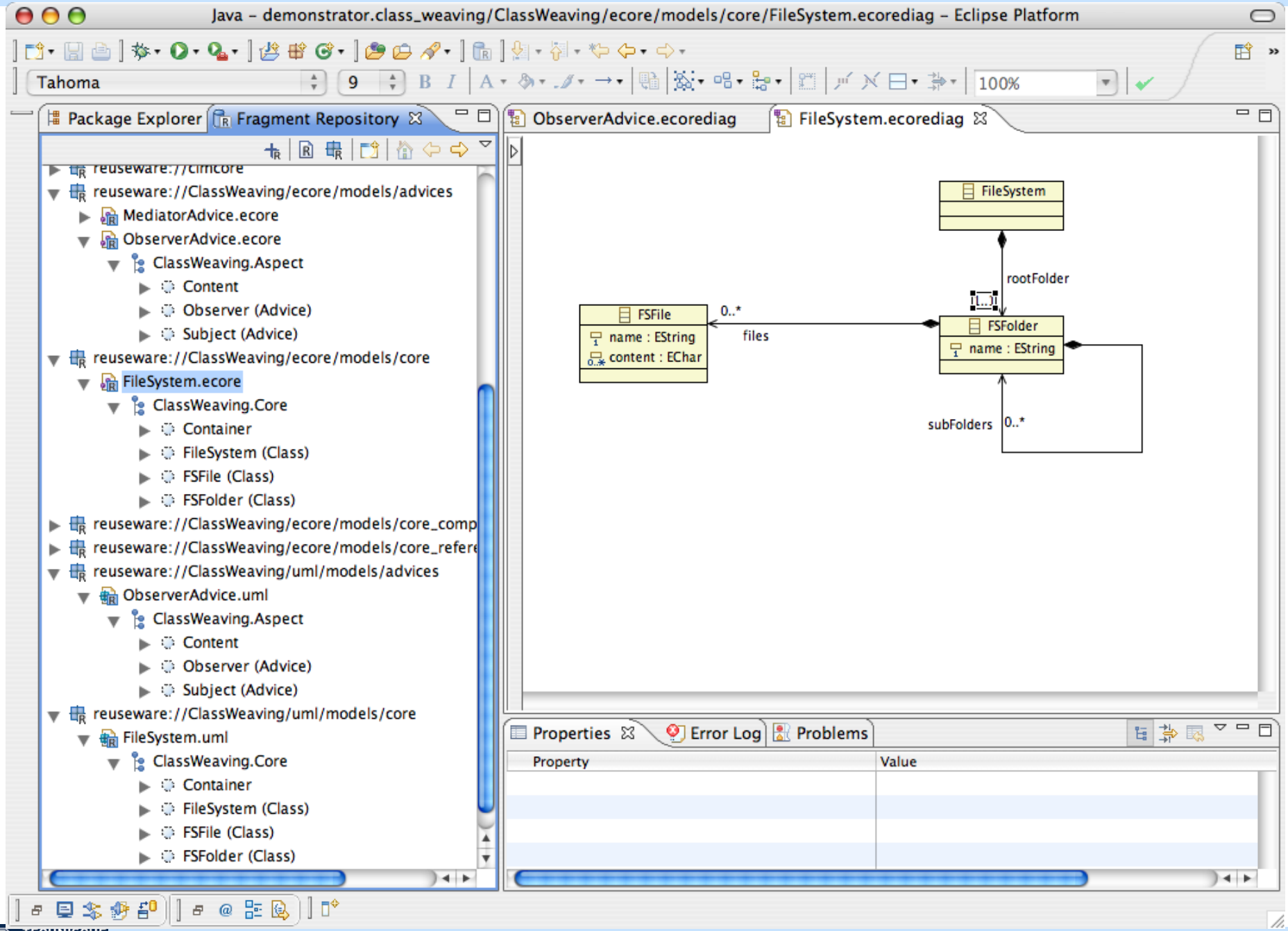
Reuseware Use Case: Example Observer

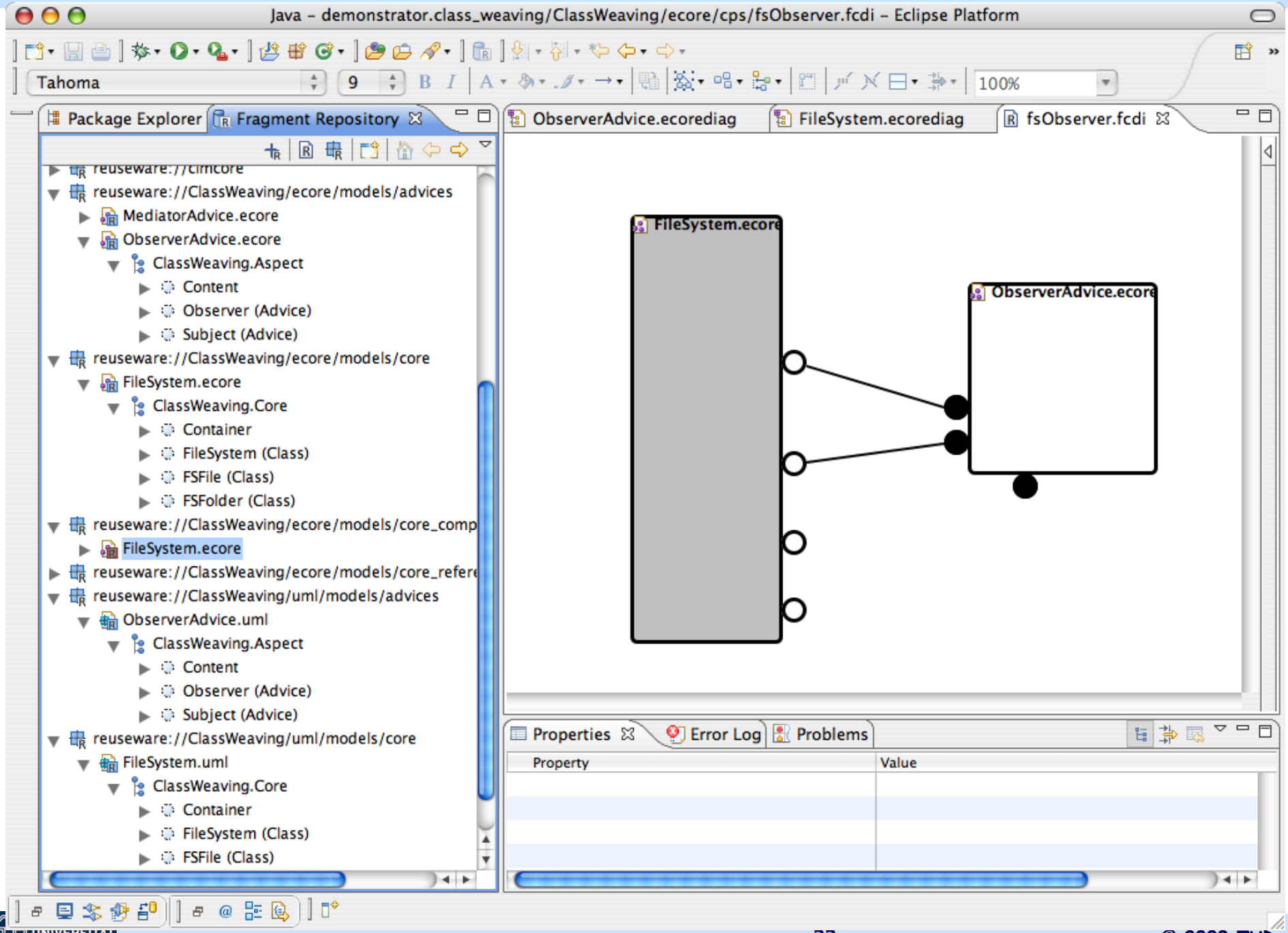
- Weaving observer functionality into a model of a file system



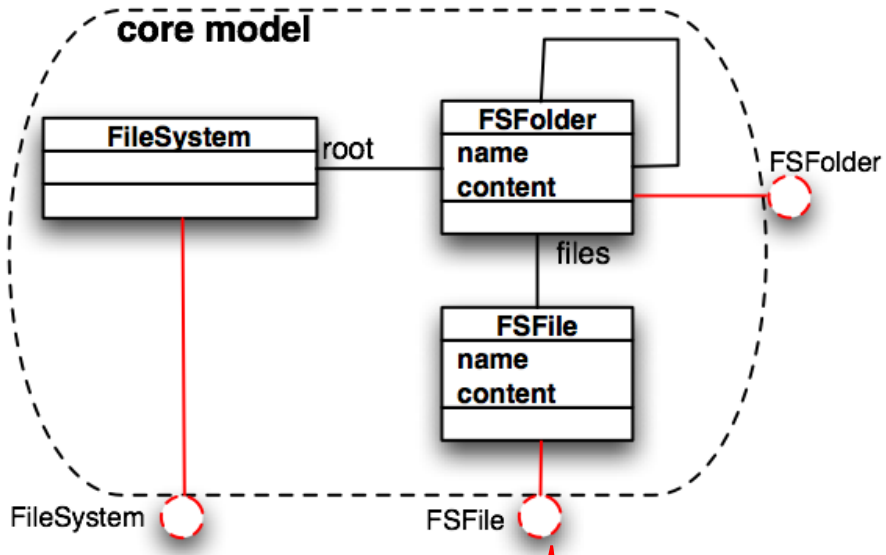


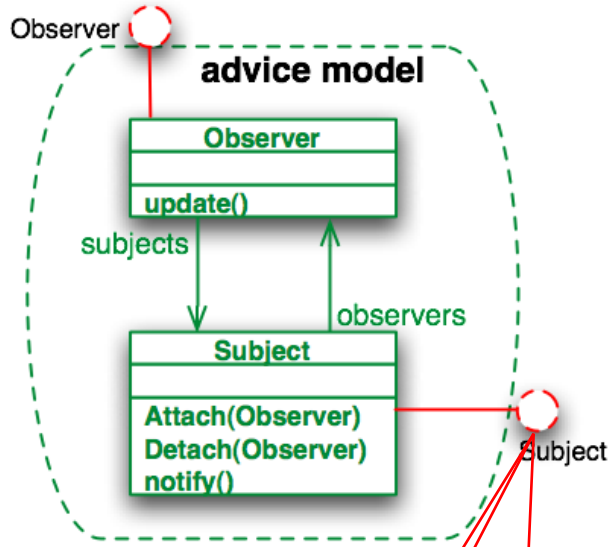

Java – demonstrator.class_weaving/ClassWeaving/ecore/models/advices/ObserverAdvice.ecorediag – Eclipse Platform
Tahoma
9
100%
Package Explorer
Fragment Repository
reuseware://cimcore
reuseware://ClassWeaving/ecore/models/advices
MediatorAdvice.ecore
ObserverAdvice.ecore
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/ecore/models/core
FileSystem.ecore
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
FSFolder (Class)
reuseware://ClassWeaving/ecore/models/core_comp
reuseware://ClassWeaving/ecore/models/core_refere
reuseware://ClassWeaving/uml/models/advices
ObserverAdvice.uml
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/uml/models/core
FileSystem.uml
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
FSFolder (Class)
ObserverAdvice.ecorediag
FileSystem.ecorediag
Observer
update() : EObject
0..*
observers
subjects
0..*
Subject
Attach(Observer) : EObject
Detach(Observer) : EObject
notify() : EObject
Properties
Error Log
Problems
adviceobserverpattern
Model
Annotation
Name:
adviceobserverpattern
Ns Prefix:
org.reuseware.example.ecore

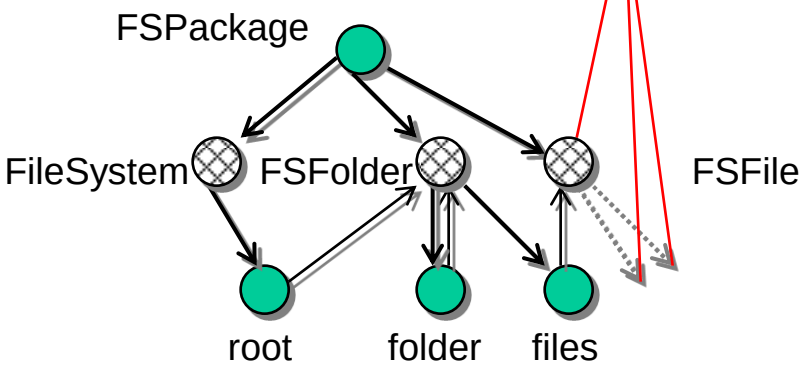

Java – demonstrator.class_weaving/ClassWeaving/ecore/models/core/FileSystem.ecorediag – Eclipse Platform
Package Explorer
Fragment Repository
reuseware://ClassWeaving/ecore/models/advices
MediatorAdvice.ecore
ObserverAdvice.ecore
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/ecore/models/core
FileSystem.ecore
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
FSFolder (Class)
reuseware://ClassWeaving/uml/models/advices
ObserverAdvice.uml
reuseware://ClassWeaving/uml/models/core
FileSystem.uml
ObserverAdvice.ecorediag
FileSystem.ecorediag
FileSystem
rootFolder
FSFile
name : EString
content : EChar
0..*
files
FSFolder
name : EString
subFolders
0..*
Properties
Error Log
Problems
Property
Value

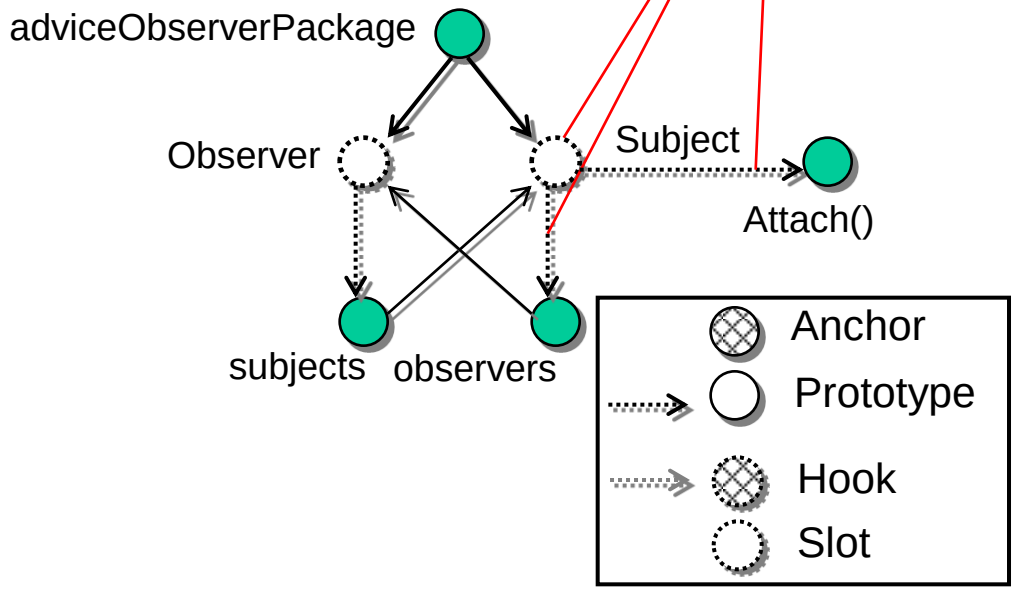

Java – demonstrator.class_weaving/ClassWeaving/ecore/cps/fsObserver.fcdi – Eclipse Platform
Tahoma
9
100%
Package Explorer
Fragment Repository
reuseware://cimcore
reuseware://ClassWeaving/ecore/models/advices
MediatorAdvice.ecore
ObserverAdvice.ecore
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/ecore/models/core
FileSystem.ecore
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
FSFolder (Class)
reuseware://ClassWeaving/ecore/models/core_comp
FileSystem.ecore
reuseware://ClassWeaving/ecore/models/core_refere
reuseware://ClassWeaving/uml/models/advices
ObserverAdvice.uml
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/uml/models/core
FileSystem.uml
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
ObserverAdvice.ecorediag
FileSystem.ecorediag
fsObserver.fcdi
FileSystem.ecore
ObserverAdvice.ecore
Properties
Error Log
Problems
Property
Value

Reuseware Use Case: Example - Observer

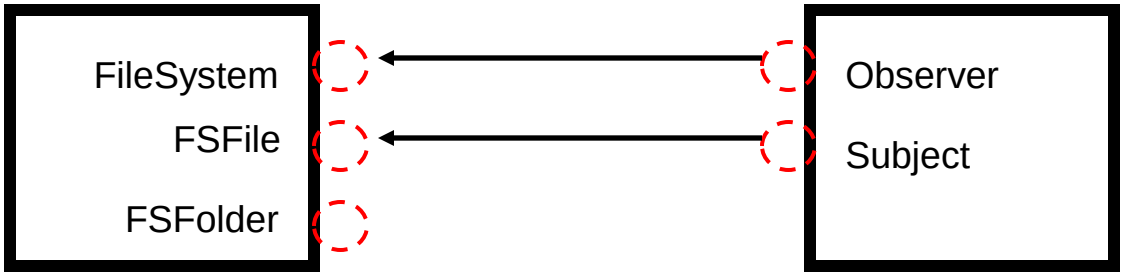


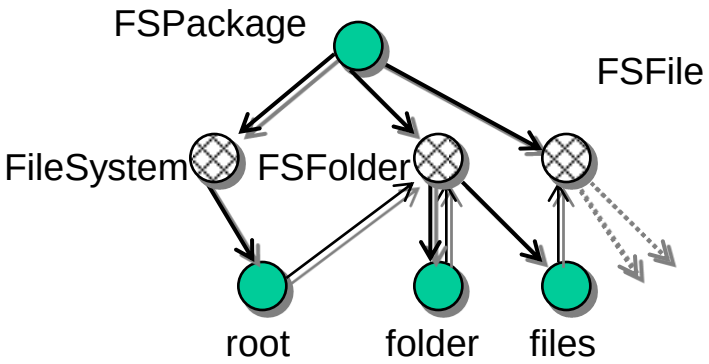


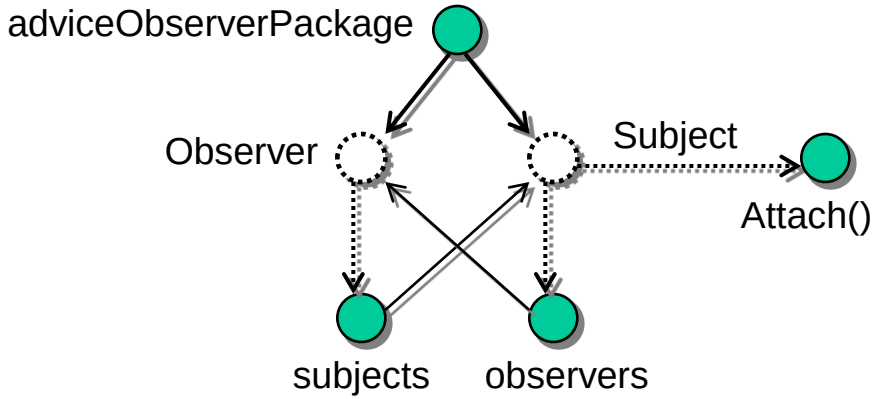


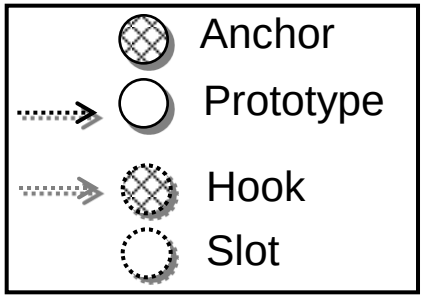


Reuseware Use Case: Example - Observer

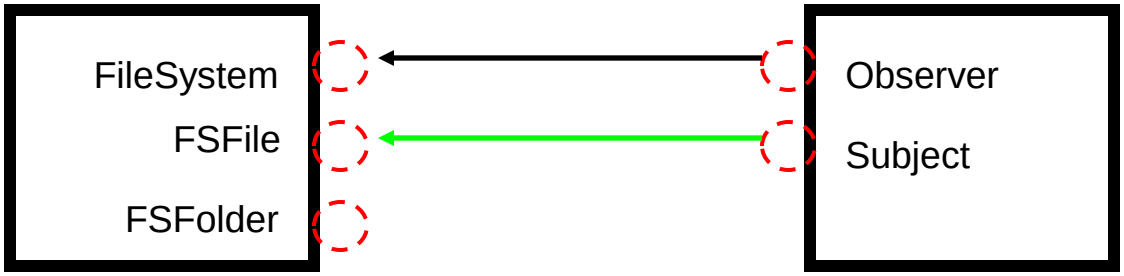


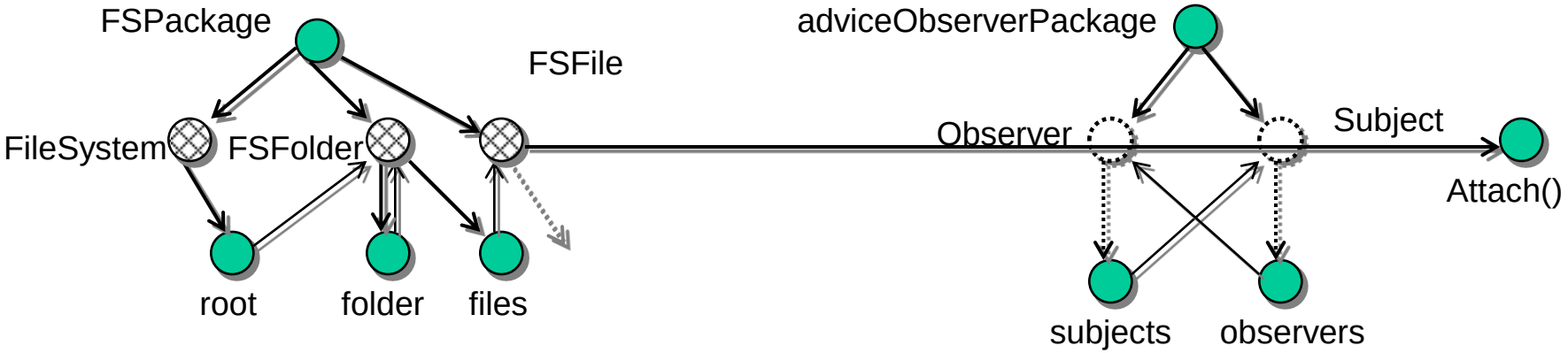


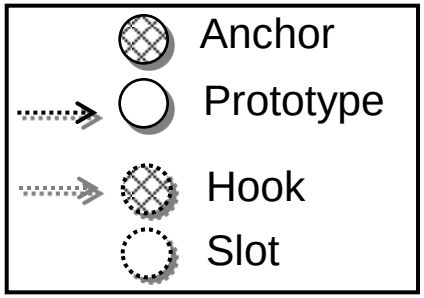




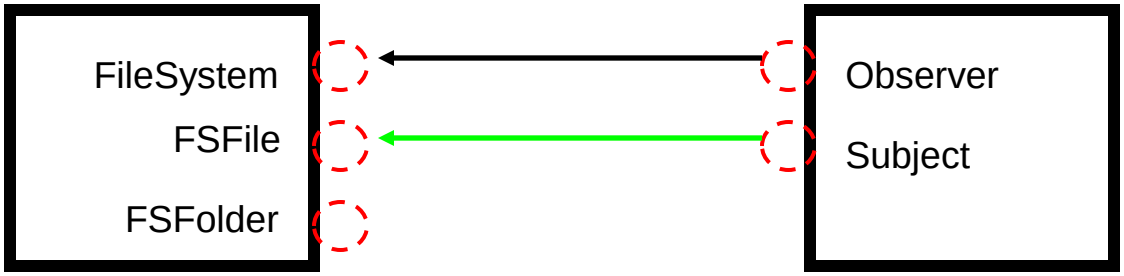
Reuseware Use Case: Example - Observer

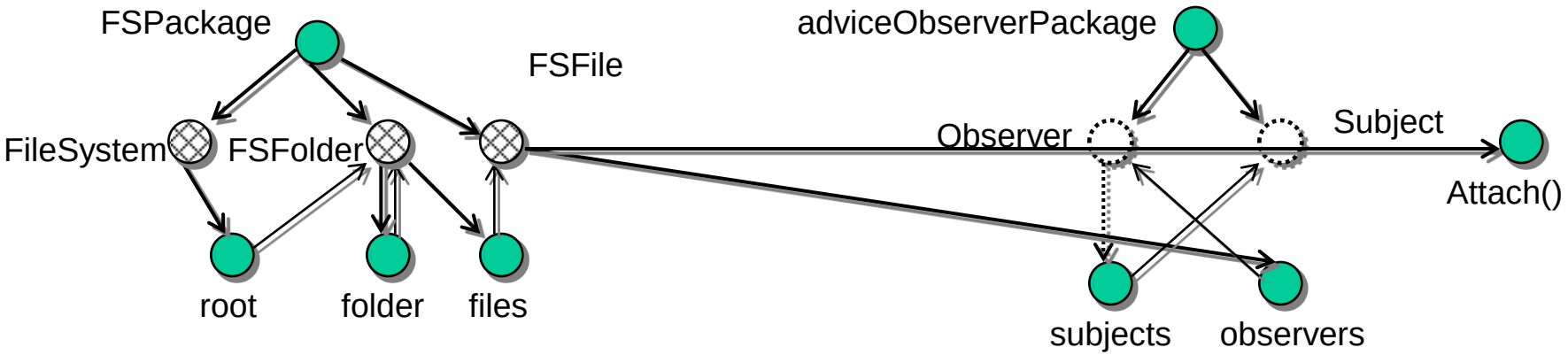


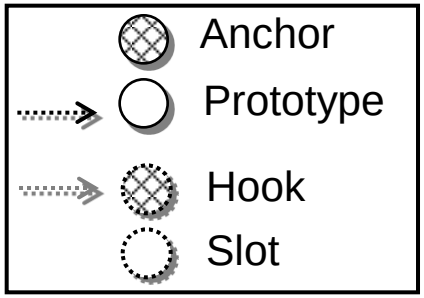




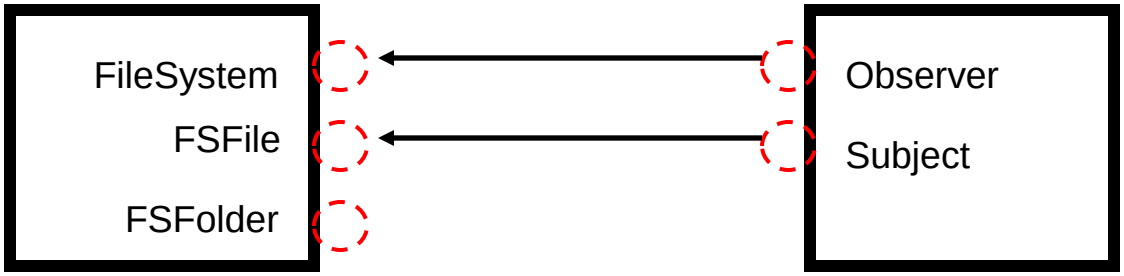
Reuseware Use Case: Example - Observer

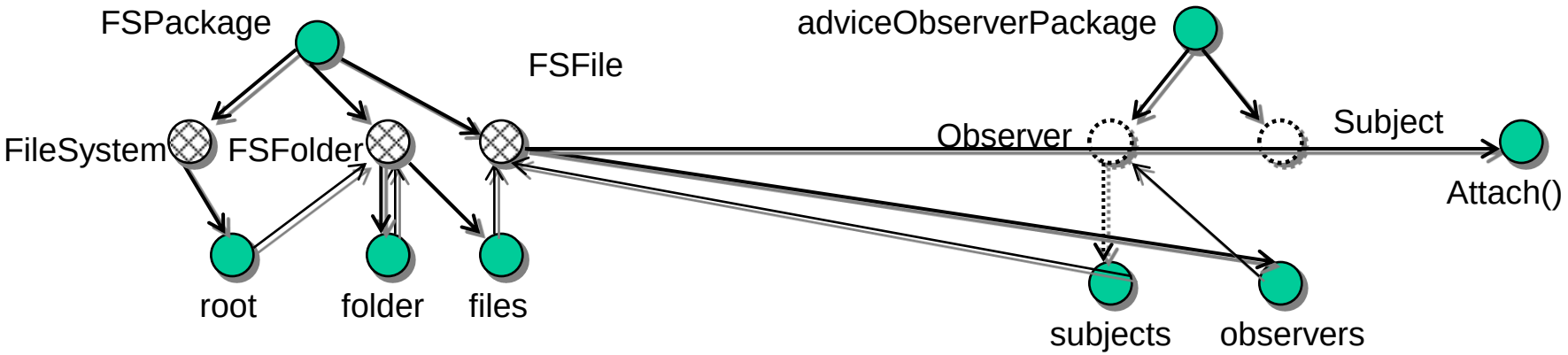


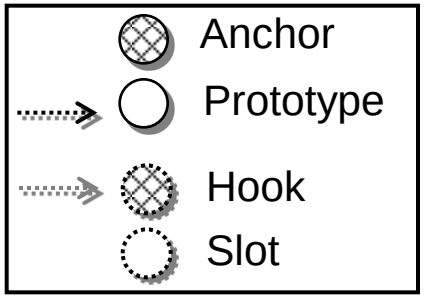




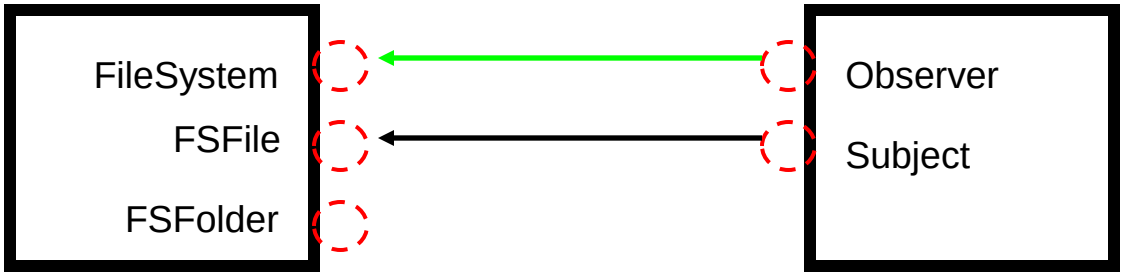
Reuseware Use Case: Example - Observer

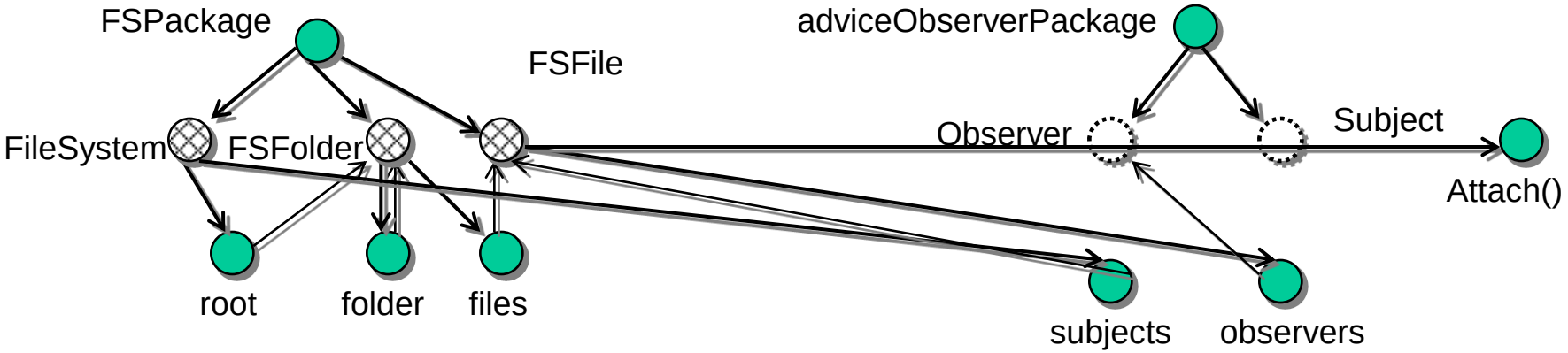


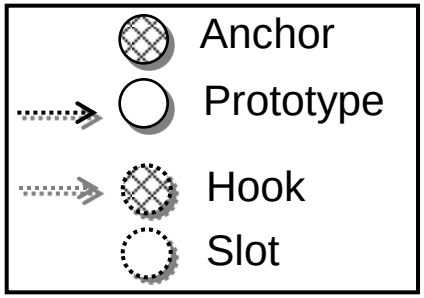




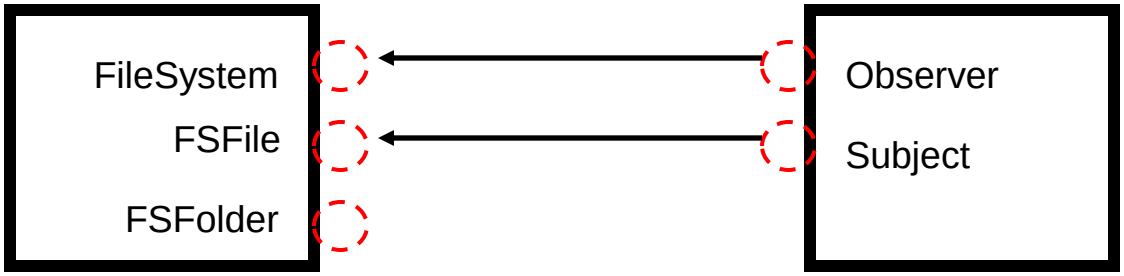
Reuseware Use Case: Example - Observer

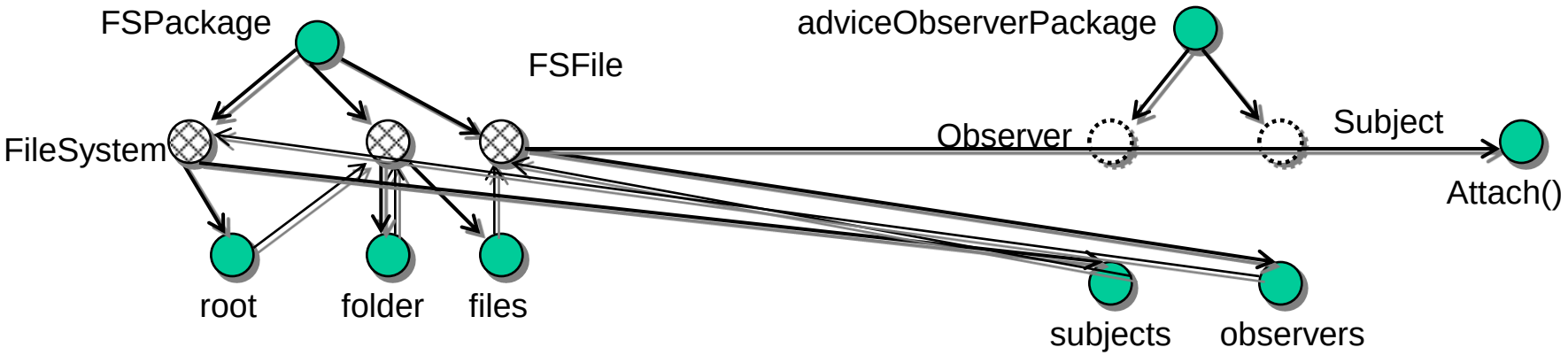


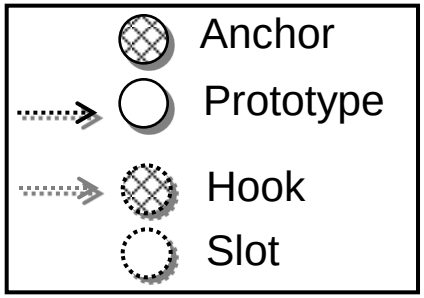


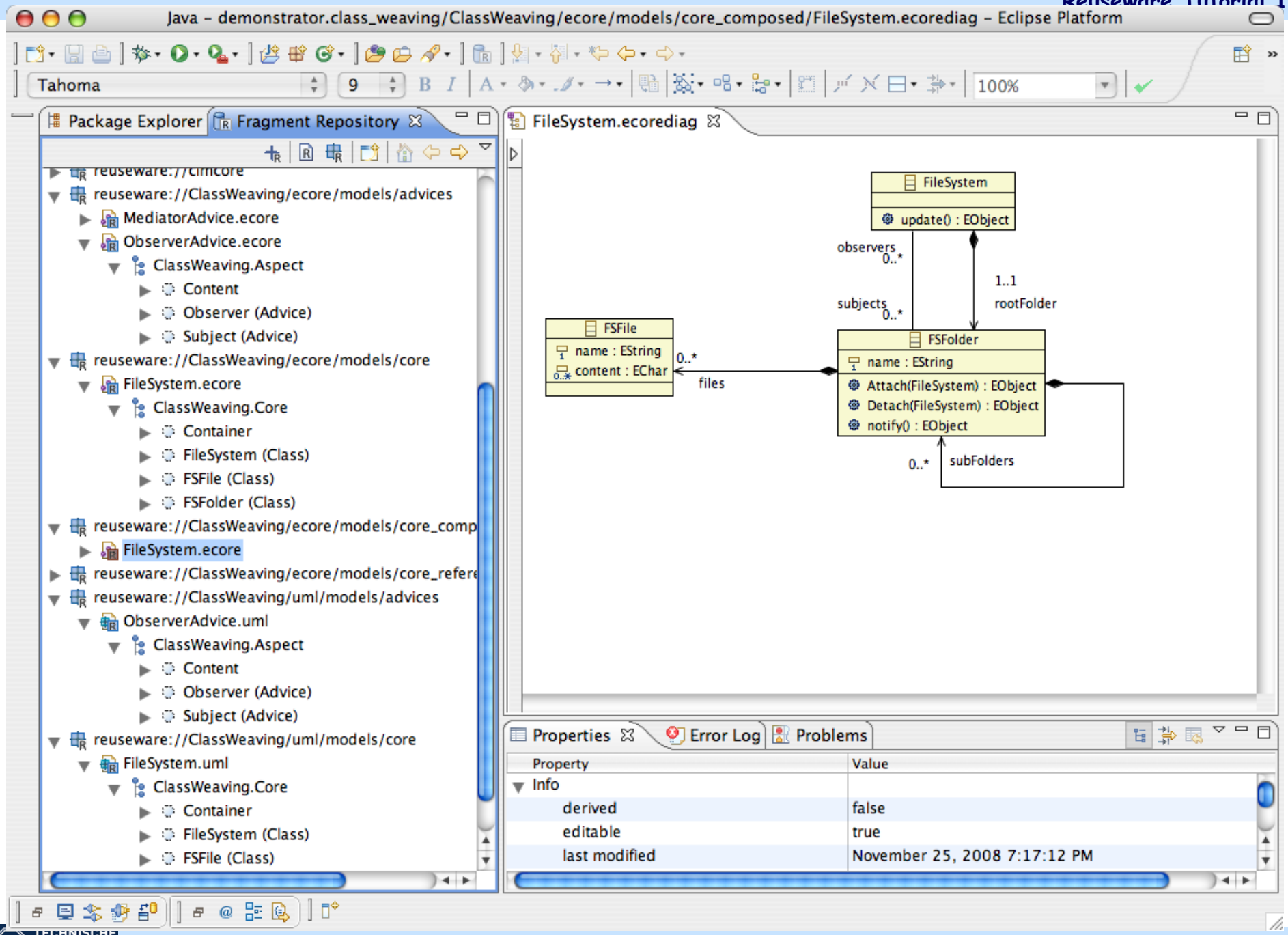


Reuseware Use Case: Example - Observer








Java – demonstrator.class_weaving/ClassWeaving/ecore/models/core_composed/FileSystem.ecorediag – Eclipse Platform
Package Explorer
Fragment Repository
reuseware://ClassWeaving/ecore/models/advices
MediatorAdvice.ecore
ObserverAdvice.ecore
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/ecore/models/core
FileSystem.ecore
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
FSFolder (Class)
reuseware://ClassWeaving/ecore/models/core_comp
FileSystem.ecore
reuseware://ClassWeaving/ecore/models/core_refere
reuseware://ClassWeaving/uml/models/advices
ObserverAdvice.uml
ClassWeaving.Aspect
Content
Observer (Advice)
Subject (Advice)
reuseware://ClassWeaving/uml/models/core
FileSystem.uml
ClassWeaving.Core
Container
FileSystem (Class)
FSFile (Class)
FileSystem.ecorediag
FileSystem
update() : EObject
observers
0..*
subjects
0..*
1..1
rootFolder
FSFile
name : EString
content : EChar
0..*
files
FSFolder
name : EString
Attach(FileSystem) : EObject
Detach(FileSystem) : EObject
notify() : EObject
0..*
subFolders
Properties
Error Log
Problems
Property
Value
Info
derived
false
editable
true
last modified
November 25, 2008 7:17:12 PM