Composition

Prasun Dewan
Department of Computer Science
University of North Carolina
CB 3175 Sitterson Hall
Chapel Hill, NC  27599-3175
dewan@cs.unc.edu
http://www.cs.unc.edu/~dewan

Class Decomposition

- Model
  - Semantics State
  - Manipulates State
- Controller
- BarView
  - Displays state

Reuse

- Model
  - Semantics State
  - Manipulates State
- Controller
  - Displays state
- TextView

Inheritance-based Composition

- Model
  - Controller
  - BarView
  - IS-A
- Controller
  - BarView
- TextView
Inheritance-based Composition

Model

TextoView

Controller

Delegation & MVC

Model

Observerable/Listener

Controller

BarView

Has-A

gVal

Editing View

State

Observer/Listener

Push Button

Increment

TextView

Draw Bar

Editing View

State

Observer/Listener

Push Button

Increment

TextView

Draw Bar

Multilevel MVC

Model

Controller

TextView

ScreenView

Multilevel MVC

Model

Controller

TextView

ScreenView

Multilevel MVC

Model

Controller

TextView

PrintView

Multilevel MVC

Model

Controller

TextView

Printer Format
Observer but Not View

- Model
- Model Replica
- Spread. Cell
- Dependent Cell
- Mouse
- Controller
- Lateral Observer
- “Callback”
- Higher-Level Observer

Delegate but Not Observer

- Delegator
- Stack
- push
- displayInt
- IntStack
- push
- displayStack
- Vector

Delegate but Not Observer

- Observer received “notification” about state change in Observable
- Notification is Observable-Specific not Observer-specific.
- Observer does not “depend” on Observable to do its tasks.

Delegation Vs Inheritance

Inheritance-based Composition

- Model
- Controller
- BarView

Delegation-Based Composition

- Model
- Controller
- BarView
Reuse Model & View

- Model
- MController
- TextView

Delegation Vs Inheritance

- Arbitrary class can be replaced

Dynamic View Switching

- Increment
- Reset
- Clear

Dynamic Configuration in Delegation

- Model
- Controller
- TextView
- setModel(Model)
- addListener(TextView)
- removeListener(BarView)

Compile-time binding in Inheritance

- Model
- Controller
- BarView
Delegation Vs Inheritance

- Arbitrary class can be replaced
- Dynamic component replacement
- IS-A compile time
- HAS-A runtime

Multiple Views

Multiple Instantiation in Delegation

Multiple Subclasses in Inheritance

Multiple Instantiation in Delegation

Delegation Vs Inheritance

- Arbitrary class can be replaced
- Dynamic component replacement
- IS-A compile time
- HAS-A runtime
- Multiple Component Instantiation
**Distribution in Delegation**

- Model
- Controller
- BarView

**Distribution in Delegation**

- Model
- Controller
- BarView

**Distribution in Inheritance**

- Model
- Controller
- BarView

**Proxies in Delegation**

- Model
- Controller
- BarView

**Delegation Vs Inheritance**

- Arbitrary class can be replaced
- Dynamic component replacement
  - IS-A compile time
  - HAS-A runtime
- Multiple Component Instantiation
- Distribution
- Proxies

**Delegation Vs Inheritance**

- Connections made by component users
- O(N^2) connections explicitly made
- Connections not visualized

- Connections specified by component definers
- O(n) connections explicitly made (components define an environment)
- Links Visualized
Delegation Vs Inheritance
- Connections remade with dynamic links
- Distribution requires more work and objects
- Ditto with Proxies

Delegation vs Inheritance
- No dynamism
- No distribution
- No proxies

Overcoming Delegation Drawbacks
- Connections made by component users
- Connections specified by component definers
- External agent makes connection at instantiation time based on specification

External Generic Agent

Facets

Registering
Auto Connection

Model

Generic Composer

Specific Composer

Controller

TextView

addListener()

reg(c,m,v)

AC(c,m,v)

registerFacet(<FacetName>, <FacetType>, <Method>, arity)

registerFacet(listener, Listener, addListener, many)

registerFacet(changeable, Changeable, setModel, single)

Specifying Facets

Facet Specification API

Server-API Approach

Model

Generic Composer

Controller

TextView

setModel()

addListener()

setModel()

Connection Visualization
Basic Connection Patterns

```java
public void set<FacetName> (<FacetType>)

public <FacetType> get<FacetName> ()

public void add<FacetName> (<FacetType>)

public Enumeration enumerate<FacetName> ()

public void remove<FacetName> (<FacetType>)

public <FacetType> <FacetName>
```

Some Advanced Patterns

```java
public class Controller implements AutoConnectAll

public String[] getAutoConnectFacets()

public String getOpposite<FacetName>()

public Object getReferenceFor<FacetName>()
```

Facet Description API

```java
public void setOppositeFacet (<Object>, <SourceFacet>, <DestFacet>)
```

General Connection

```text
connect (a, u, b, v)
```

Direct Connection

```text
connect (a, u, b, v)
```

Proxy-based Connection

```text
proxy (p, a, w, x, v, b)
```
Interposition

connect (a, u, b, v)

1. connect (A, U, B, V) -> proxy(P, A, U, V, U, V, B);
   interpose (P); ProxyConnect (P, A, U, V, U, V, B)

2. connect (A, U, B, V) -> DirectConnect (A, U, B, V)

Right Facet Translation

connect (a, u, b, v)

4. connect (A, U, B, V) -> proxy(P, A, U, X, V, B);
   ProxyConnect (P, A, U, W, X, V, B)

General Facet Translation

connect (a, u, b, v)

6. connect (A, U, B, V) -> proxy(P, A, U, W, X, V, B);
   ProxyConnect (P, A, U, W, X, V, B)

Type Translation: Adapter

connect (a, u, b, v)

3. connect (A, U, B, V) -> proxy(P, A, U, V, U, V, B);
   ProxyConnect (P, A, U, U, V, B)

Left Facet Translation

connect (a, u, b, v)

5. connect (A, U, B, V) -> proxy(P, A, U, W, U, V, B);
   ProxyConnect (P, A, U, W, X, V, B)

Delegation Vs Inheritance

- Connections made
  by component
  users

- O(N^2) connections
  explicitly made

- Connections not
  visualized

- Connections specified
  by component definers

- O(n) connections
  explicitly made
  (components define
  an environment)

- Links Visualized
Manual Vs Assisted Delegation

- Connections made by component users
- O(N^2) connections explicitly made
- Connections not visualized
- Connections specified by component definers/registrars
- 1 - O(N^2) connections explicitly made (components define an environment)
- Links Visualized

Delegation vs Inheritance

- Flexibility of Delegation => Less Automation
- Provide incremental path
- Some disadvantages not inherent and can be partially overcome