E-Commerce BookStore Application

Dogan Altunbay
Damla Arifoglu
Hilal Zitouni

Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- Discussion
- Conclusion
- Demo

Overview

- **Introduction & Motivation**
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- Discussion
- Conclusion
- Demo

Motivation

- Increasing modularity, reuse, and maintainability using AOSD.
  - On an E-commerce Bookstore application
- E-commerce Application Concerns:
  - Authentication & Authorization
  - Shipment
  - Null Check – DB
  - Exception Handling (a general concern)

Overview

- Introduction & Motivation
- **Crosscutting Concerns**
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- Discussion
- Conclusion
- Demo

Crosscutting Concerns: E-commerce Applications

- E-commerce Applications :
  - Online applications to sell/buy products.
  - Shipment to all over the world
- Increase in web usage :
  - Increase in e-commerce application usages
    - As User demands change and improve
      - System must improve and therefore change

Croscutting Concerns : E-commerce Applications

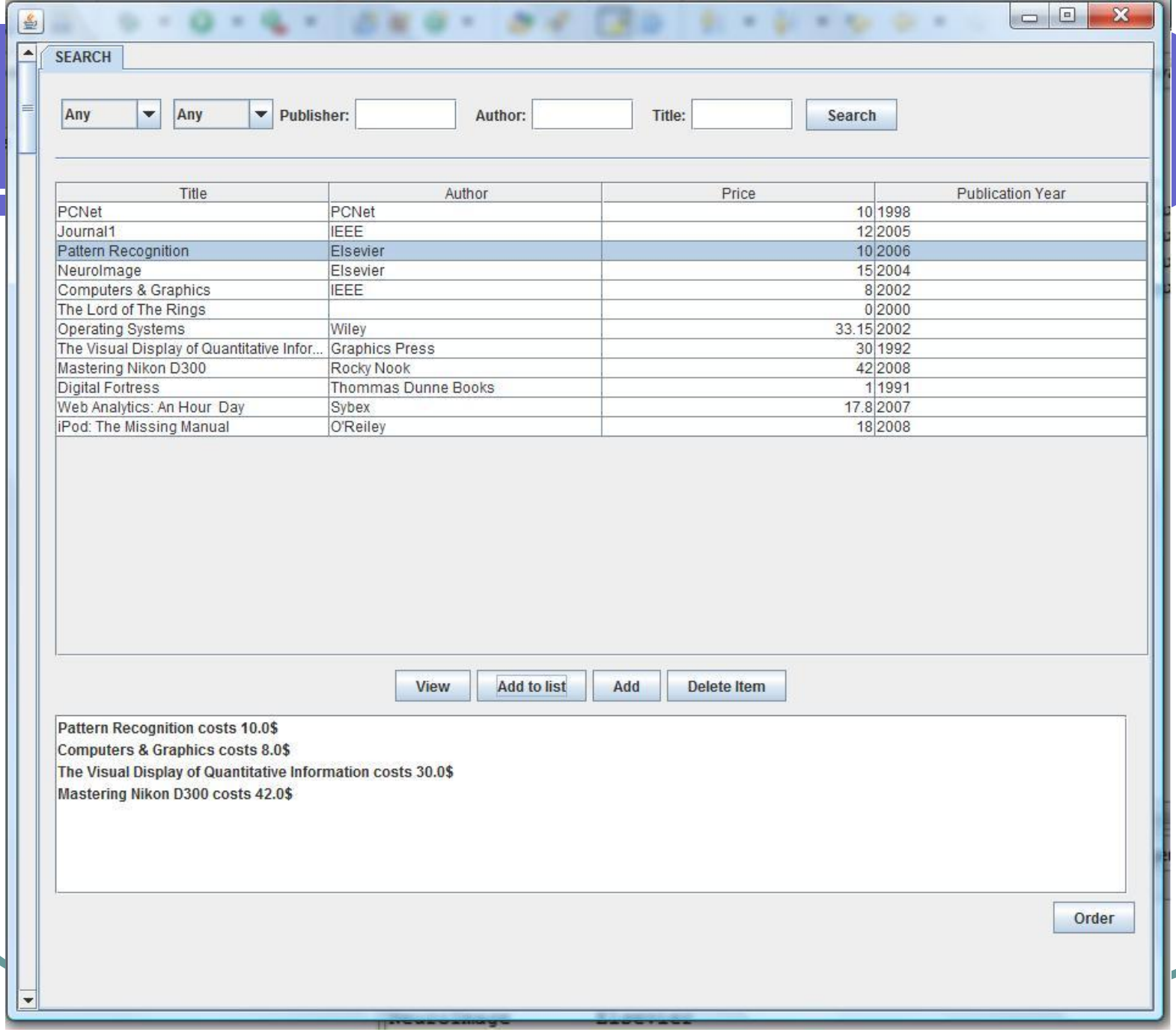
- *Authorization & Authentication*
  - *Authentication :*
    - *User identity check*
    - *Security concerns*
  - *Authorization :*
    - *deciding on the permissions of a user over the system*
      - User types for different accesses.

Croscutting Concerns : E-commerce Applications

- Adding New Functionalities:
  - With the increase in demand, user needs will change
    - Thus, so do system.
  - Previous system has a common type of user
    - New Version : User Types
      - Guest User, Registered User, Admin User

Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - **BookStore Application**
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- Discussion
- Conclusion
- Demo


SEARCH
Any
Any
Publisher:
Author:
Title:
Search
Title
Author
Price
Publication Year
PCNet
PCNet
10
1998
Journal1
IEEE
12
2005
Pattern Recognition
Elsevier
10
2006
NeuroImage
Elsevier
15
2004
Computers & Graphics
IEEE
8
2002
The Lord of The Rings
0
2000
Operating Systems
Wiley
33.15
2002
The Visual Display of Quantitative Infor...
Graphics Press
30
1992
Mastering Nikon D300
Rocky Nook
42
2008
Digital Fortress
Thommas Dunne Books
1
1991
Web Analytics: An Hour Day
Sybex
17.8
2007
iPod: The Missing Manual
O'Reiley
18
2008
View
Add to list
Add
Delete Item
Pattern Recognition costs 10.0$
Computers & Graphics costs 8.0$
The Visual Display of Quantitative Information costs 30.0$
Mastering Nikon D300 costs 42.0$
Order

Example Case : BookStore Application

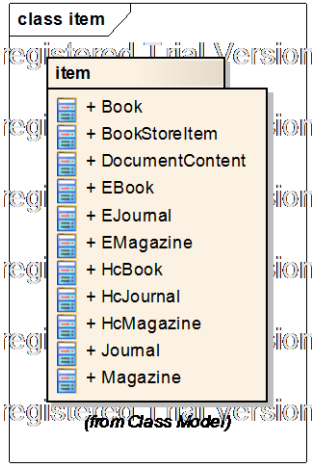
- users will search through the document database of the Book Store,
  - view and/or order documents.
  - shipment to the desired destination
    - payment policies will change according to the country

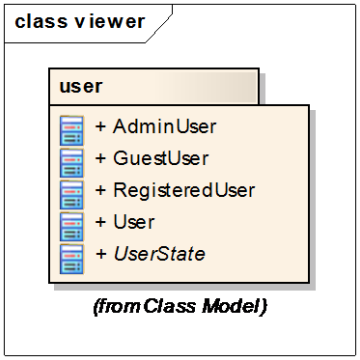
Example Case : BookStore Application

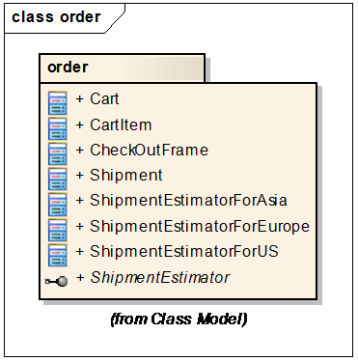
- ***Object Oriented vs. Aspect Oriented***
  - Our previous system: do not have the authentication for different user types.
  - HOWEVER
    - The non-registered users should not be able to order any selected document.
  - Our current system: provide the users different types of authentications.

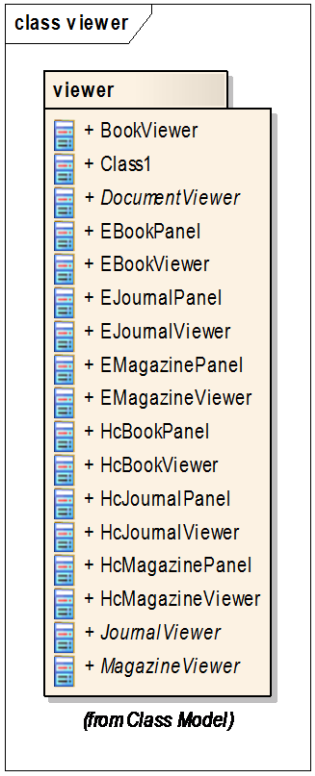
Example Case : BookStore Application

- **Overall Package System :**









ITEM, USER, ORDER, VIEWER

Overview

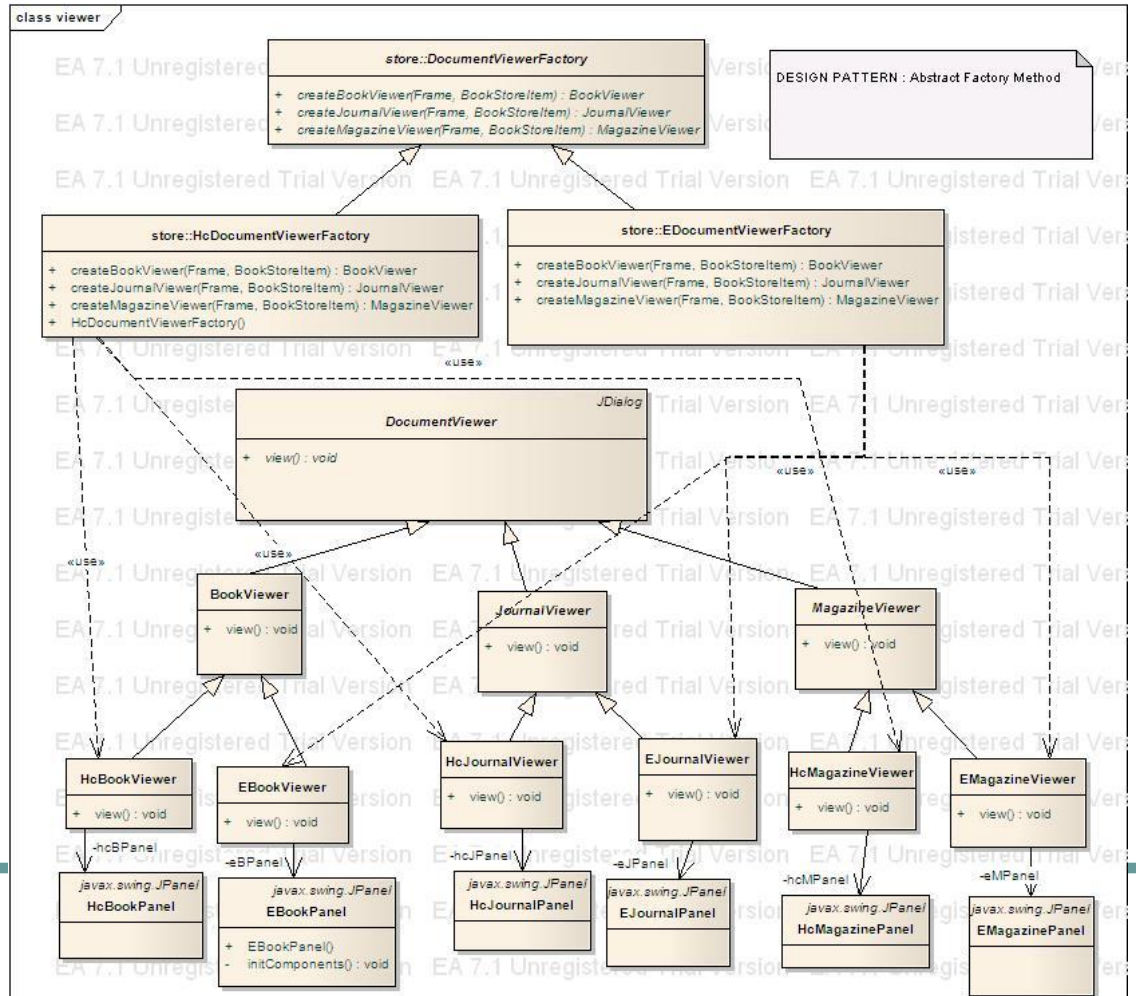
- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - **Design Patterns**
- Aspect Oriented Programming
- Related Work
- Discussion
- Conclusion
- Demo

Example Case : Design Patterns

- Some design patterns used to achieve better solutions for common problems
  - Abstract Factory Design Pattern
  - Strategy Design Pattern
  - State Design Pattern

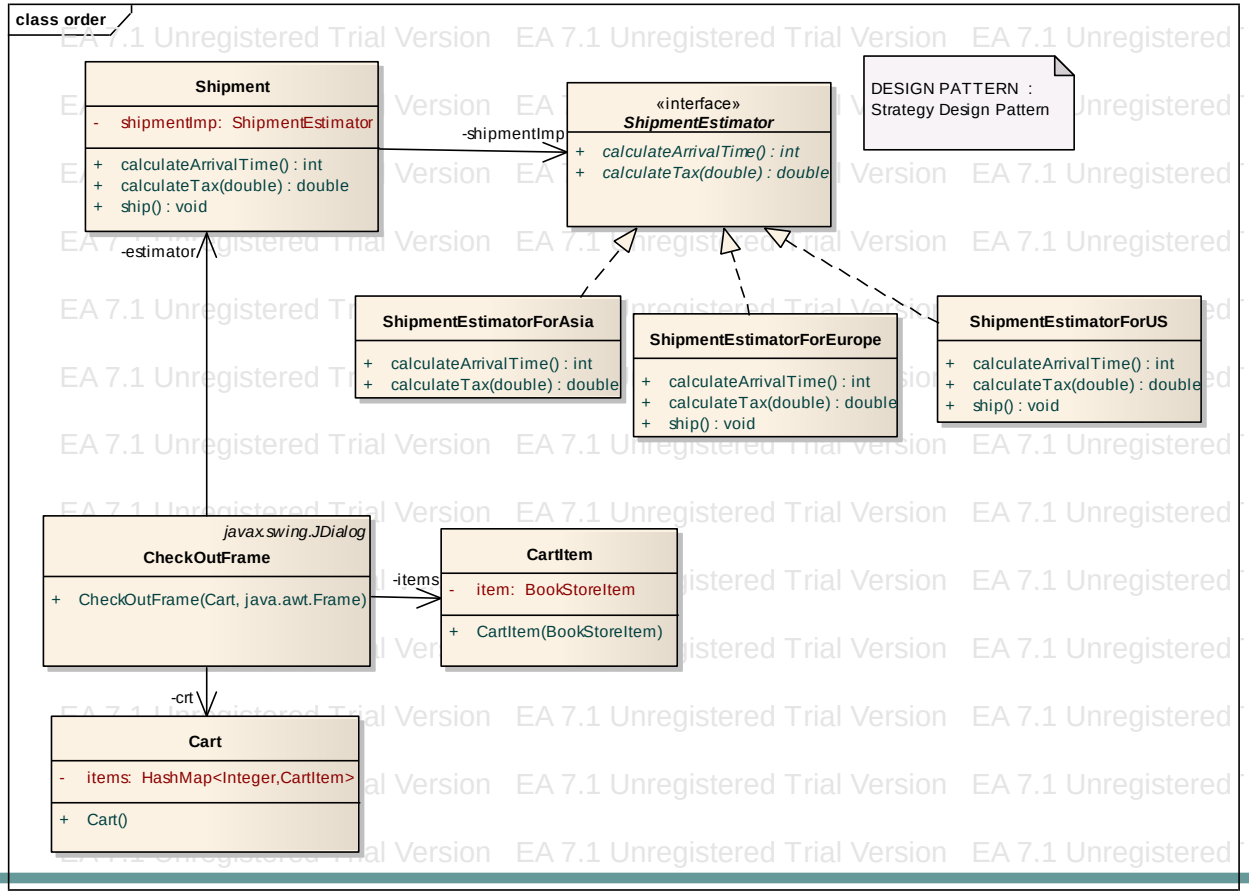
Example Case : Design Patterns

Abstract Factory Design Pattern



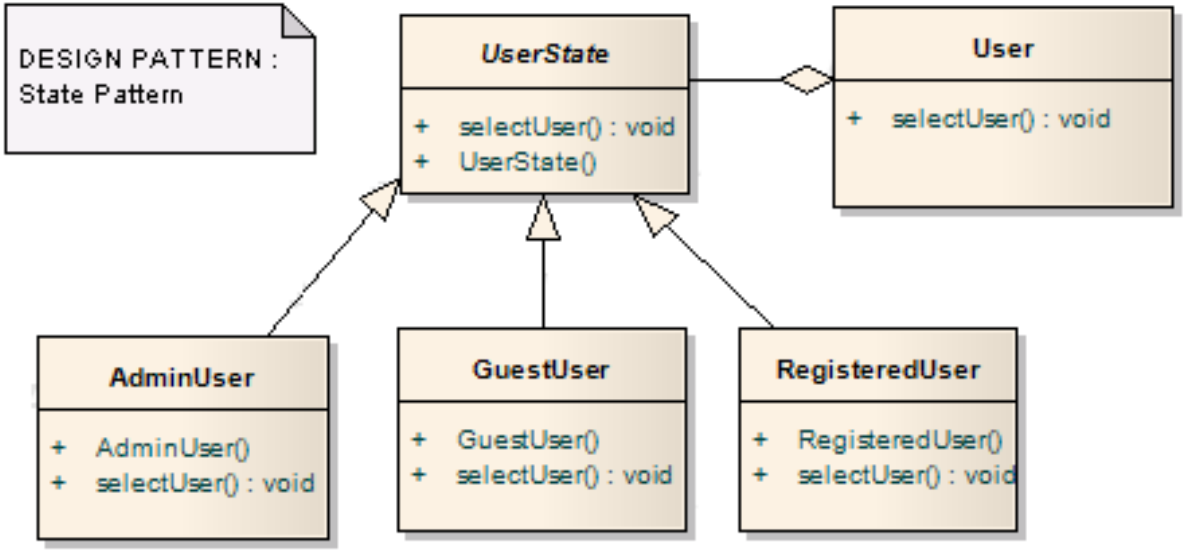
Example Case : Design Patterns

Strategy Design Pattern



Example Case : Design Patterns

State Design Pattern



Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- **Aspect-oriented Programming**
- Related Work
- Discussion
- Conclusion
- Demo

Aspect-oriented Programming

- *Handling Aspects:*
  - JBoss AOP : pure java syntax
  - Weaving :
    - reflection with the help of xml files that define the bindings between join points and advices.

Aspect-oriented Programming

```
1 private void viewDoc(BookStoreItem item){
2
3             if(!this.user.isAuthenticated())
4                         throw new UnsupportedOperationException();
5
6             DocumentViewer viewer = null;
7             .
8             .
9             .
10            viewer= docFactory.createMagazineViewer(this,item);
11
12                        viewer.view();
13 }
```

Figure 5 – Example insertion of authentication related code to legacy code

Aspect-oriented Programming

```
1 private void viewDoc(BookStoreItem item){
2
3            if(!this.user.isAuthenticated())
4                        throw new UnsupportedOperationException();
5
6            DocumentViewer viewer = null;
7            .
8            .
9            .
10           viewer= docFactory.createMagazineViewer(this,item);
11
12                       viewer.view();
13 }
```

Without AOP exception handlers should be added to each and evey necessay code block

Figure 5 – Example insertion of authentication related code to legacy code

Aspect-oriented Programming

```
1 public class AuthenticationAspect{
2         public Object login(MethodInvocation method) throws Throwable{
3                 //before login
4                 UI3View frame = (UI3View) method.getTargetObject();
5                                         new LoginDialog(frame,true).setVisible(true);
6                                         Object result = method.invokeNext();
7                                         //after login
8                                         return result;
9         }
10        public Object checkPermission(MethodInvocation method) throws Throwable{
11                UI3View frame = (UI3View) method.getTargetObject();
12                User usr = ((IUserMixin)frame).getUser();
13                if(usr.hasPermission(method.getMethod().getName()))
14                        return method.invokeNext();
15                else
16                        JOptionPane.showMessageDialog(frame,"You have not permission for "+
17                        method.getMethod().getName(),"Error",JOptionPane.ERROR_MESSAGE);
18                return null;
19        }
}
```

Using Introduced Members

Figure 6 – Authentication Aspect

Aspect-oriented Programming

```
1 <aop>
2    <pointcut expr="call(public void UI3View->display())" name="show"/>
3     <pointcut expr="execution(private void ui.UI3View->;viewDoc(item.BookStoreItem))"
4     name="view"/>
5    <aspect class="aspect.AuthenticationAspect" scope="PER_VM"/>
6    <bind pointcut="show">
7        <advice aspect="aspect.AuthenticationAspect" name="login"/>
8    </bind>
9    <bind pointcut="view">
10        <advice aspect="aspect.AuthenticationAspect" name="checkPermission"/>

11   </bind>
12   <bind pointcut="execution(public void ui.UI3View->display())">
13        <advice aspect="aspect.AuthenticationAspect" name="login"/>

14   </bind>
15    <introduction class="ui.UI3View">
16        <mixin>
17            <interfaces>user.IUserMixin</interfaces>
18            <class>user.UserMixin</class>
19            <construction>new user.UserMixin()</construction>
20        </mixin>
21    </introduction>
22 </aop>
```

Figure 7 – Sample jboss-aop.xml file.

Aspect-oriented Programming : Exception Handling

- Exception Handlers : OOP
  - Scattered in code.
- implementing all the exception handlers as a separate aspect code
  - increase modularity

Aspect-oriented Programming : Enforcement Aspects:

- Restriction in the invocation of some of the methods from certain classes.
- in order to view the information of the selected document,
  - press the “View” button for a selected document
  - However,
    - CONTROL THAT :
      - *getContent()* → CANNOT BE INVOKED IN, OTHER THAN *EDocumentViewerFactory* class

Aspect-oriented Programming : Null Check

- check for the database problems
- In Database Systems :
  - often possible for a primary field to be null
    - will arise problems for the insertion

Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- **Related Work**
- Discussion
- Conclusion
- Demo

Related Work

- Reina, A. et al [2]
  - developed a web-application using two emerging technologies: AspectJ and Java Server Pages(JSP) and Cacoon-a web framework based on XML.
- Concerns: Security(authentication and exception handling) Pooling, Cashing and Logging Concerns

Related Work

- Sun Microsystems Java Pet Store specification:
  - The famous “Pet Store” AOP application in AspectJ
- Concerns:
  - Aspect_Security
  - aspect_Logging => performs log function call. By this aspect, all code is concentrated in one place, not scattered all over the application

Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- **Discussion**
- Conclusion
- Demo

DISCUSSION:

- make improvements in code
  - with less code generation and more reusability.
- making a small change in the system requires a lot of changes in the whole project.

DISCUSSION:

- Naming & Implementation Standardization
  - Forces programmers to make a standard code generation for a possible later use of AOP
- Writing aspect codes for previously implemented programs will be harder
  - if no standardization

DISCUSSION : AspectJ vs. JBOSS

- AspectJ superiorities :
  - Better documentation and IDE usage.
  - Easier implementation.
    - Difficult for JBOSS :
      - REASON : "reflections" - the message passing procedure- in JBoss
- JBoss superiorities:
  - provide dynamic AOP
- Differences:
  - JBoss Codes written in java files, just pointcuts written in different files, XML files
    - Keyword : “bind” → for binding pointcuts to corresponding places
  - AspectJ Codes written in “*.aj” files

Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- Discussion
- **Conclusion**
- Demo

CONCLUSION:

- Always there will be a need of an improvement to the system.
- Great Design will not save programmers from intervening into code for maintanance.
  - it might be quite difficult to foresee the upcoming technology and its requirements.
- However; with AOP:
  - making changes to the system is easier.
- Especially for concerns like
  - Authentication
  - Authorization
  - Null Check
  - Exception Handling
  - Adding New Items and Functionalities

Overview

- Introduction & Motivation
- Crosscutting Concerns
- Example Case
  - BookStore Application
  - Design Patterns
- Aspect Oriented Programming
- Related Work
- Discussion
- Conclusion
- **Demo**

Thank You For Listening!..

ANY QUESTIONS ?