Aspect-Oriented Programming for Web Controller Layer

Keiji Hokamura, Kyushu Institute of Technology
Naoyasu Ubayashi, Kyushu Institute of Technology
Shin Nakajima, National Institute of Informatics
Akihito Iwai, DENSO CORPORATION
Today’s talk

AOP [Kiczales1997]
- Pointcut: specifies "where"
- Crosscutting concerns (CCC) (e.g. logging)
- Aspect: module of CCC

Our approach
- AOWP (Web-specific AOP mechanism)
  - Web protocol-based pointcuts (request, pflow, etc.)
  - Web protocol-based aspect instantiation

- AspectJ-based AOP
  - Insufficient support for Web-specific CCC
  - Program-execution-based pointcut (call, etc.)
Outline

• Motivation
• AOWP
• Practical example
• Conclusion & Future work
Motivation
Example:

**PC sale on EC-site**

- Prices in the business store and the education store are different.
- To use the education store, users have to access the confirmation page first and read explanation of the store.
Example:

PC sale on EC-site

- However, some users access the education store without the access to the confirmation page.

A pointcut for the CCC captures the page requests if the users did not access the confirmation page.

forward to the confirmation page ➡ crosscutting
Modularize CCC with AspectJ-based AOP

Pointcut

save history of the access to the confirmation page

ConHistory.accessed();

add to the original code

Education store

Confirmation page

EduNoteAction

+ void doAction()

EduWSAction

+ void doAction()

A page request to education store fires a doAction method if the classes have a ‘Edu-’ prefix

• In AspectJ-based AOP, the pointcut captures program execution points which relate to the page requests

Programmers are required to deeply understand the original code

call (void Edu*.doAction()) && if (!ConHistory.isAccessed())
Our approach

<table>
<thead>
<tr>
<th>Web protocol-based aspects</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOWP</td>
</tr>
<tr>
<td>page request</td>
</tr>
<tr>
<td>page request flow</td>
</tr>
<tr>
<td>per Web application</td>
</tr>
<tr>
<td>per session</td>
</tr>
<tr>
<td>pre page request</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AspectJ-based AOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>method call</td>
</tr>
<tr>
<td>field reference and set</td>
</tr>
<tr>
<td>control flow</td>
</tr>
<tr>
<td>etc.</td>
</tr>
<tr>
<td>singleton</td>
</tr>
<tr>
<td>per object</td>
</tr>
<tr>
<td>per control flow</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>based on program execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>points captured by pointcuts</td>
</tr>
<tr>
<td>aspect instantiation</td>
</tr>
</tbody>
</table>

- AspectJ [Kiczals2001]
- AOPHP [Stamey2005]
- phpAspect [phpaspect.org]
Modularize CCC with AOWP

Pointcut

- Captures page requests which do not come via the confirmation page
  ⬤ Does not depend on the original code

request URL: /eduConf.php

request URL: /edu/ws.php

request URL: /edu/notebook.php
AOWP
Pointcut designators and aspect instantiation in AOWP

- per-session aspect
  - Client (Ann)
    - cookie data
    - request history (time)
  - Client (Bob)
    - cookie data
    - request history (time)
- context information
  - form data
  - per-request aspect
  - per-application aspect
  - Ann's session data
  - Bob's session data

page request history per user (used by the pflow pointcut and etc.)

points captured by the request pointcut
pflow pointcut

- **Pflow** captures page requests based on the page history (if multiple requests in history are used, the *pagehistory* pointcut is used)
- These pointcuts are based on **trace-based AOP** [Douence2004, Walker2004, Allan2005], but **AOWP** is different in that the history is managed **per user**
AOWP/PHP

• A proof-of concept implementation in PHP
• Class-based AOP framework
• Aspects are described as subclass of AOWP_Aspect class
• Pointcuts and advice are described in the constructor of the aspect class
Aspect description in AOWP/PHP

class EduAccessConAspect extends AOWP_Aspect {

    public function __construct() {
        $requestPC = new AOWP_RequestPointcut('edu/.*');
        $pflowPC = new AOWP_PFlowPointcut('eduConf.php');
        $pflowPC->Not();
        $requestPC->addAnd($pflowPC);

        $advice = new AOWP_BeforeAdvice();
        $advice->setPointcut($requestPC);
        $advice->setAdviceBody('forwardToCon', $this);
        $this->addAdvice($advice);
    }

    public function forwardToCon() {
        forward('eduConf.php');
    }

}

Definition of the pointcut request('edu/.*') && !pflow('eduConf.php')

Definition of the advice
AOWP/PHP’s pointcut designators

<table>
<thead>
<tr>
<th>Designator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>Select page requests</td>
</tr>
<tr>
<td>pflow</td>
<td>Select based on the flow of page requests</td>
</tr>
<tr>
<td>accessnum</td>
<td>Select based on the number of access users</td>
</tr>
<tr>
<td>functioncall</td>
<td>Select points of a function call</td>
</tr>
<tr>
<td>globalvariableget</td>
<td>Select points of reading global variable</td>
</tr>
<tr>
<td>globalvariableset</td>
<td>Select points of writing global variable</td>
</tr>
<tr>
<td>scriptexecution</td>
<td>Select points of a script file execution</td>
</tr>
<tr>
<td>methodcall</td>
<td>Select points of a method call</td>
</tr>
<tr>
<td>if</td>
<td>Select if a specified condition is true</td>
</tr>
</tbody>
</table>

Aspects which use only Web-protocol-based pointcut designators may be applicable to multiple web applications.
Practical example
Login aspect

Blog system

• Execute login authentication before page requests
• The aspect is instantiated per user (per session)
• The aspect is applicable to multiple web applications

Wiki

Aspect (per session)

Field
loginedID = null

Pointcut
request(‘.*’) && !if(loginedID == null)

Advice
Authenticate a user with form data.

Success: write the user’s loginID to the loginedID field
Failure: forward to login page
Future work & Conclusion
Conclusion

- AOWP: Web-specific AOP mechanism
  - Web-protocol-based pointcut designators
  - Web-protocol-based aspect instantiation

- We can modularize the Web-specific crosscutting concerns as Web-protocol-based aspects
Future work

- More case studies and evaluation
- Self-* systems implemented with AOWP
  - Self-* systems [Kramer2007] automatically reconcile themselves with changes
  - They are suitable for continuous systems and open systems (e.g. web applications)
  - Monitor aspect + AOP-based evolution
Thank you
Spair slides
Weaving for Web-specific pointcut

- The Web-specific events are related to the points at the PHP source code.
  - Page requests: execution the PHP file
  - Read form data: read $_GET or $_POST
  - Read/write cookie data: read/write $_COOKIE
  - Read/write session data: read/write $_SESSION
Web-specific aspect instantiation

- Aspect instances are automatically created
  - Where is the instance created or destroyed?
- In current AOP
  - issingleton, perthis, pertarget, percflow, percflowbelow
- In web applications
  - perrequest: Cut across a request
  - persession: Cut across a user session
  - perapplication: Cut across the entire web application
Web-specific trace-based pointcut

Decide applying aspects based on the history of the program execution

• In Web application
  - The trace is managed per a user session
  - The web-specific trace-based pointcut
  - pagehistory, sessionhistory, cookiehistory, etc.
How AspectJ-based AOP handle page requests

To select the page requests to pages for educational people

we should specify the point in the original code related to the requests

function genNoteAction() {
    // View the detail of
    // notebook PC
}
genNoteAction();

function eduNoteAction() {
    // View the detail of
    // notebook PC
}
eduNoteAction();

function eduWSAction() {
    // View the detail of
    // work station PC
}
eduWSAction();

To select the page requests to pages for general people

function eduNoteAction() {
    // View the detail of
    // notebook PC
}
eduNoteAction();
Web-specific aspect instantiation (in new ver.)

Aspect instances are automatically created.
- Where is the instance created or destroyed?

Aspect instantiation in AOWP
- per request
- per session
- per application
Implementation

- Weaver implemented in PHP
- All the original code is translated into ASTs (implemented with the extended PEAR PHP_Parser)
- Aspects are woven as modifying the ASTs
- The woven ASTs are translated into the woven code
Weaving of login aspect

<table>
<thead>
<tr>
<th>AOWP-IDE</th>
<th>a number of files</th>
<th>line codes</th>
<th>weaving time</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOWP Home</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project Path</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aspect Path</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Folder Name</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AOWP-IDE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| PukiWiki       |                   |            |              |
| WordPress      |                   |            |              |