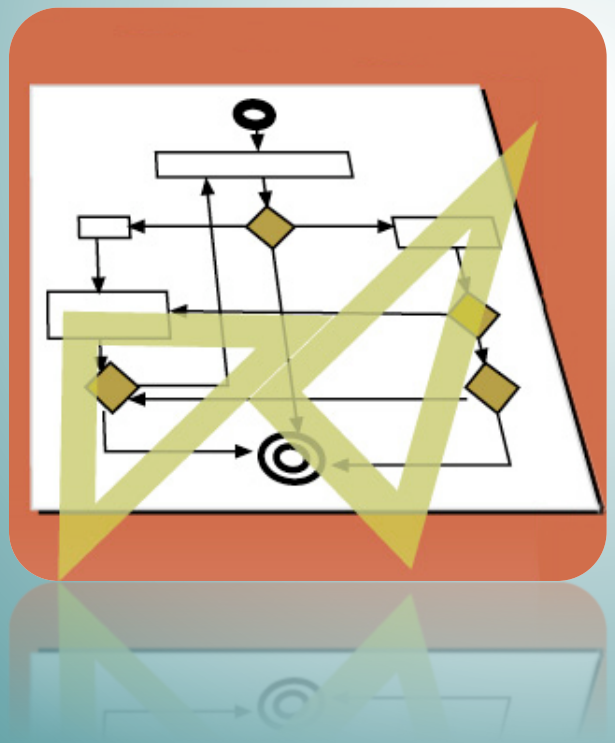
Ingeniería del Software II

Seminario. Programación Orientada a Aspectos



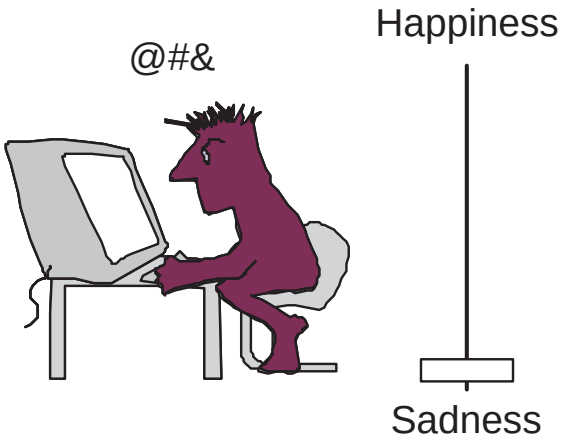
Pablo Sánchez Barreiro

DPTO. DE MATEMÁTICAS, ESTADÍSTICA Y COMPUTACIÓN

p.sanchez@unican.es

Table of Contents

Crosscutting Concerns Drawbacks

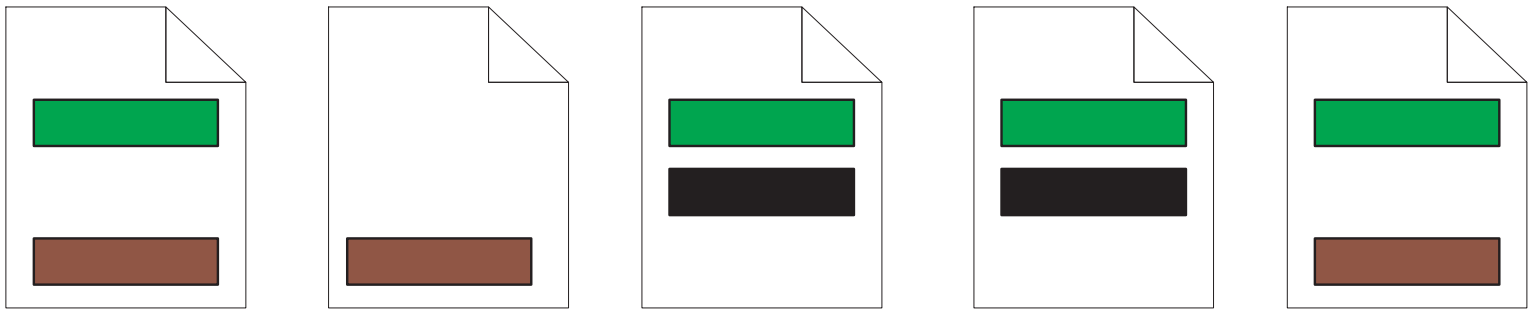


Base Functionality

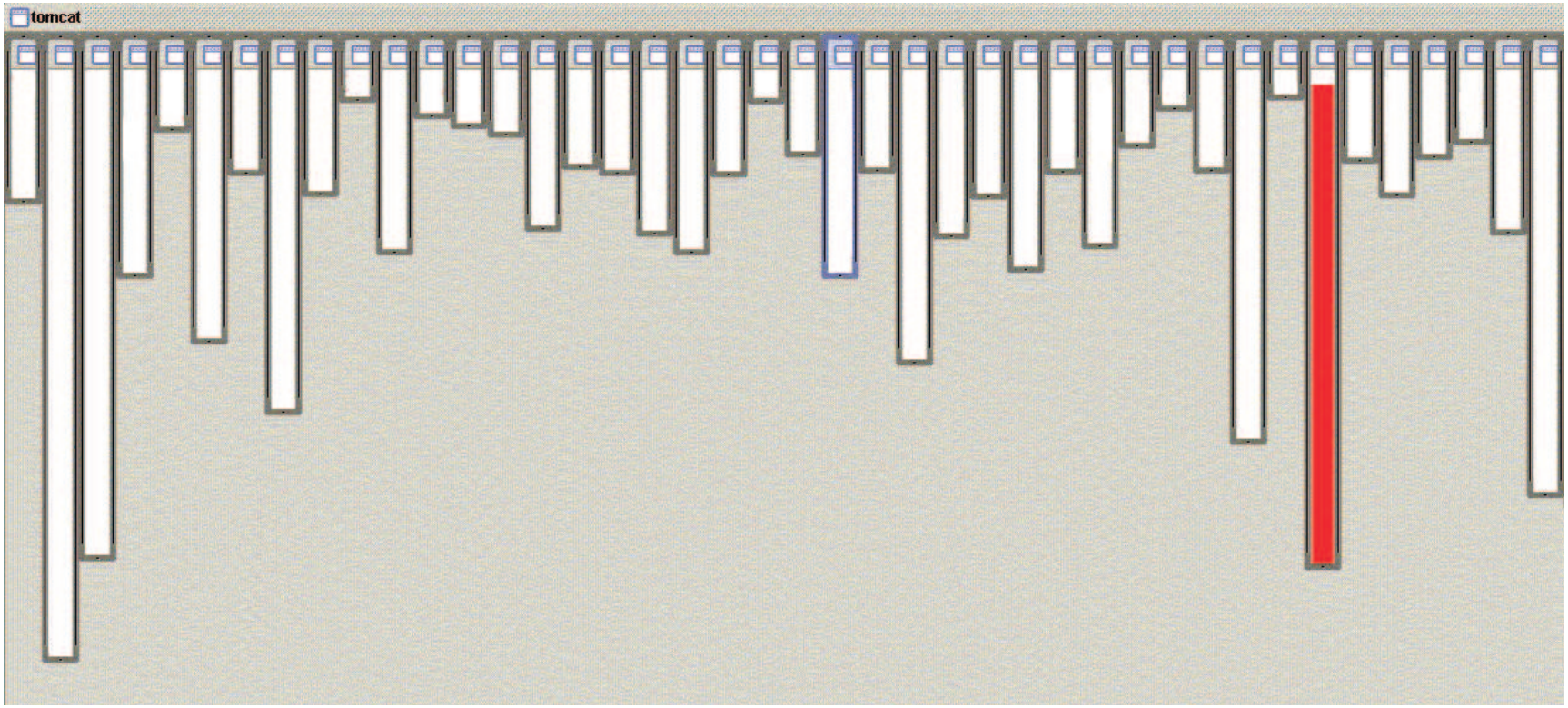
Access Control

Fault Tolerance v2.0

Encryption

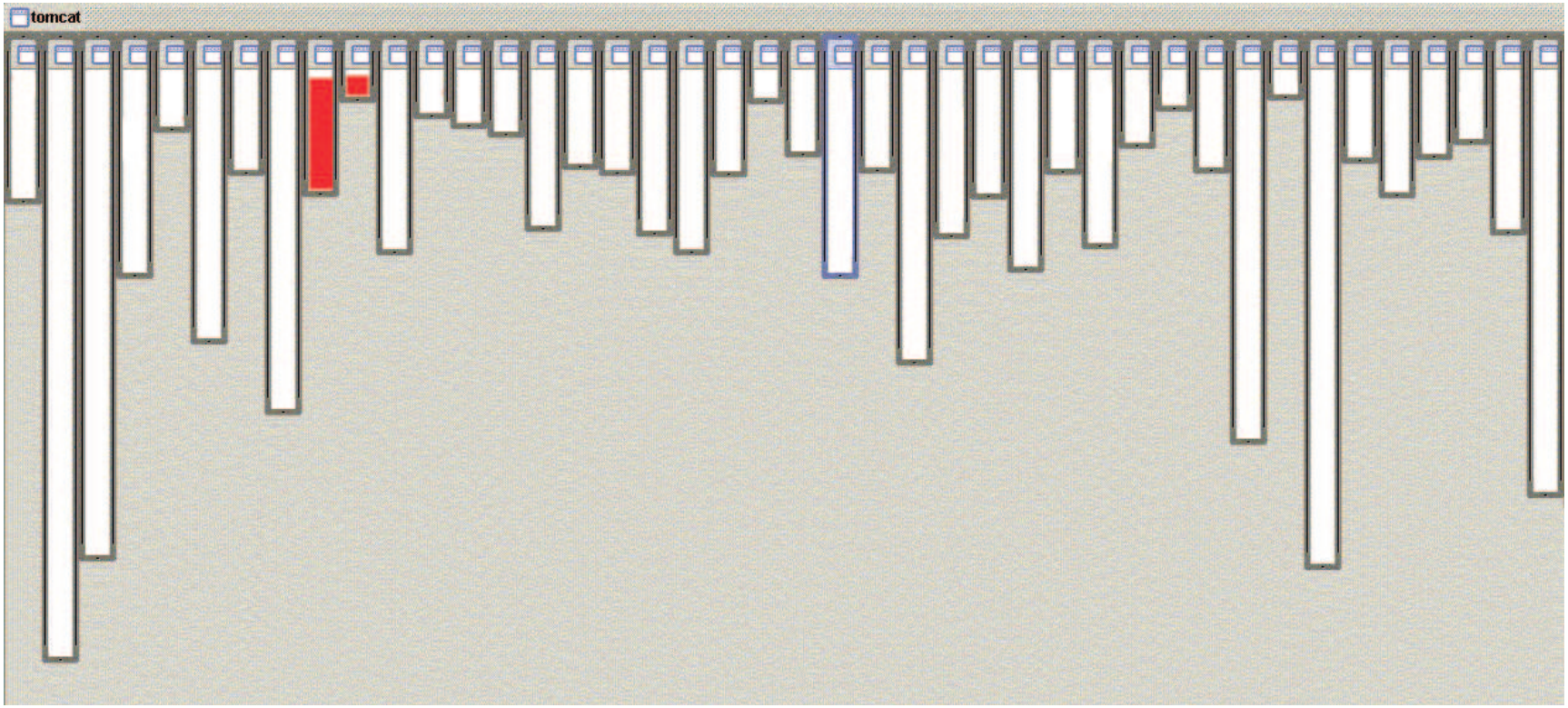


Case Study: Apache Tomcat



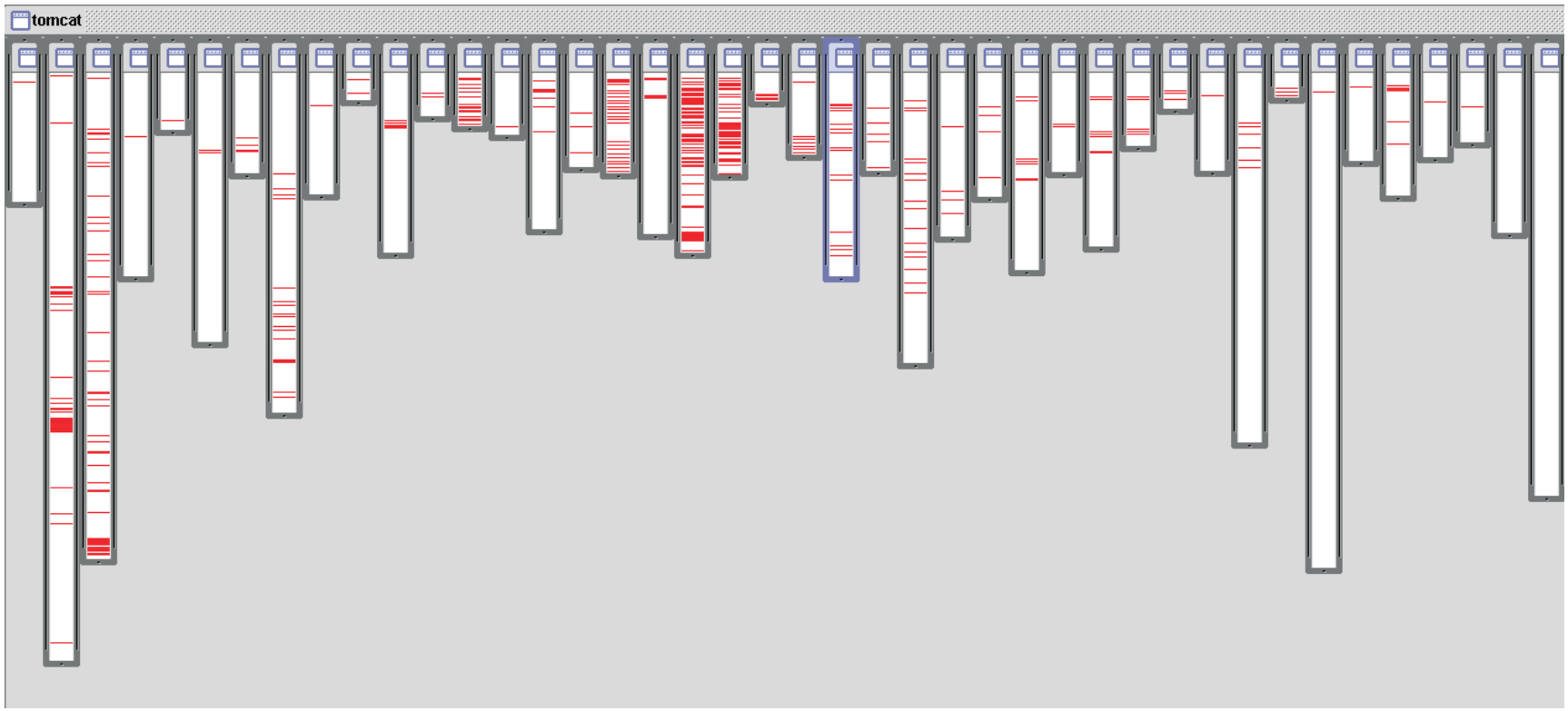
XML parsing

Case Study: Apache Tomcat



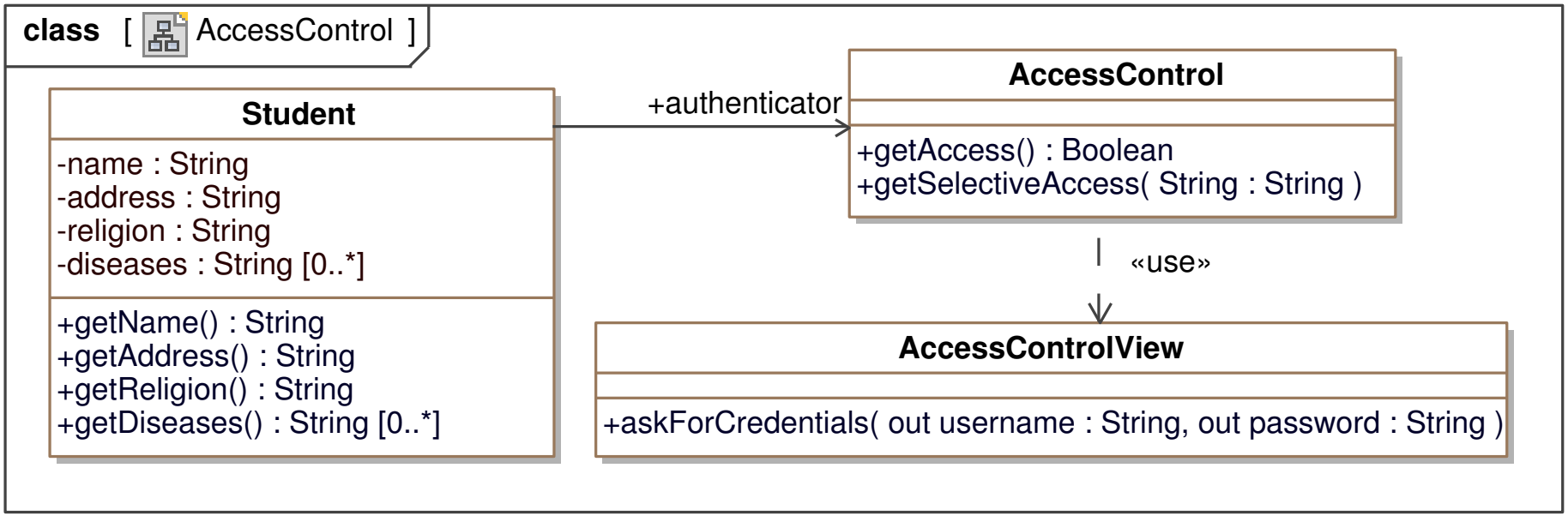
URL pattern matching

Case Study: Apache Tomcat



Logging

Motivating example: Access Control



Motivating example: Access Control

```
public class Student {

  protected String religion;
  protected AccessControl authenticator =
                          new AccessControl();

  public String getReligion() {

     String result = null;

     if (authenticator.getAccess()) {
          result = this.religion;
     } // if

     return result;
  } // getReligion
} // Student
```

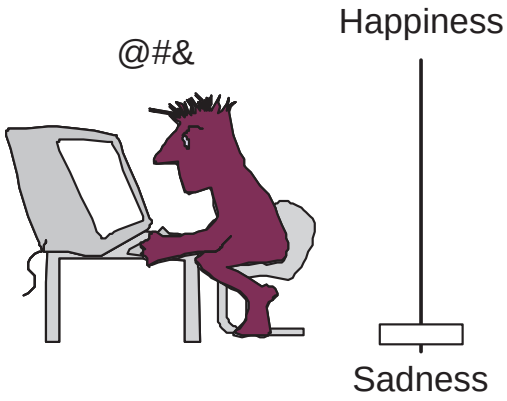
Crosscutting code

Motivating example: Access Control

```
public List<String> getDiseases() {

  List<String> result = null;

  if (authenticator.getAccess()) {
     result = this.diseases;
  }  // if

  return result;
} // getDiseases
```

Crosscutting code

Main Shortcoming of Crosscutting Concerns

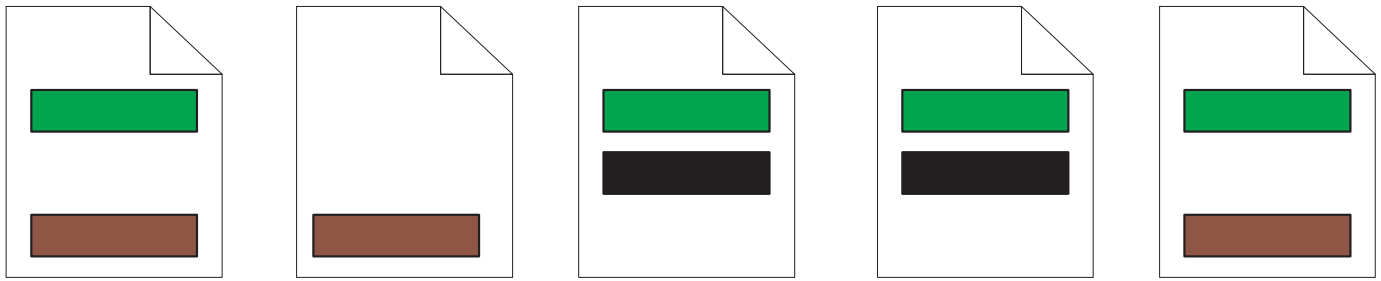


Base Functionality

Access Control

Fault Tolerance v2.0

Encryption



Module A | Module B | Module C | Module D | Module E

Conclusions

Scattering

A concern appears spread over several software modules.

Tangling

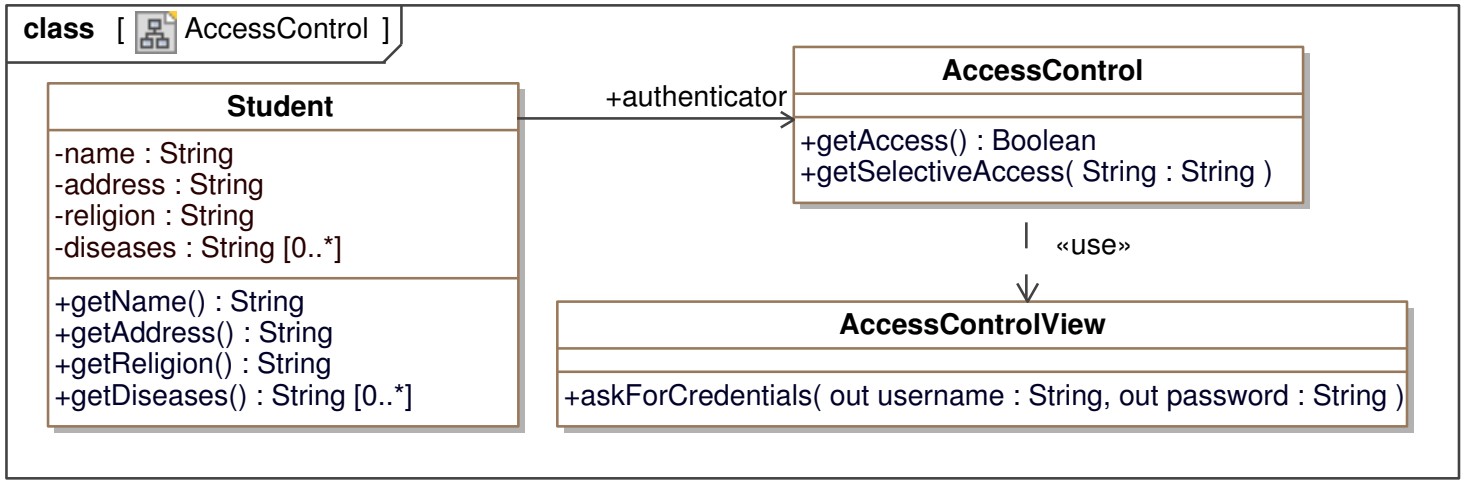
Several concern appears software modules.

1. **Scattering:** Decreases encapsulation, (consequently decreases cohesion), creates redundancies, hinders reutilization of the crosscutting concern and increase coupling.
2. **Tangling:** Decreases cohesion, (consequently hampers reutilization), and increases coupling.

Concern-Oriented Metrics [Sant'Anna et al., 2007]

CDC	Concern Diffusion over Classes
CDO	Concern Diffusion over Operations
CIC	Class-Level Interlacing between Concerns
OOC	Operation-Level Overlapping between Concerns
ACC	Afferent Coupling between Classes
ECC	Efferent Coupling between Classes
LCC	Lack of Concern-Based Cohesion

Concern-Oriented Metrics [Sant'Anna et al., 2007]

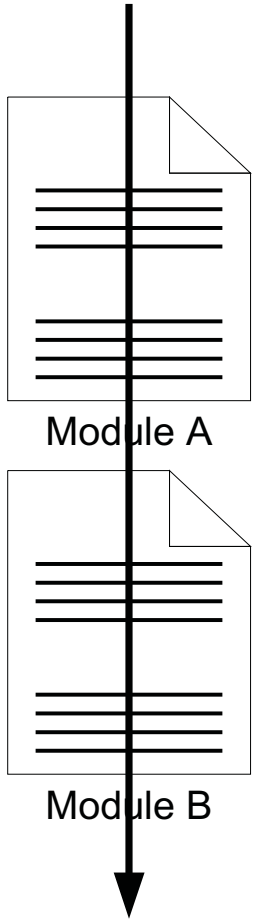


	CDC	CDO	CIC	OOC
Data Storage (base)	1	4	1	1
AccessControl	3	5	1	1

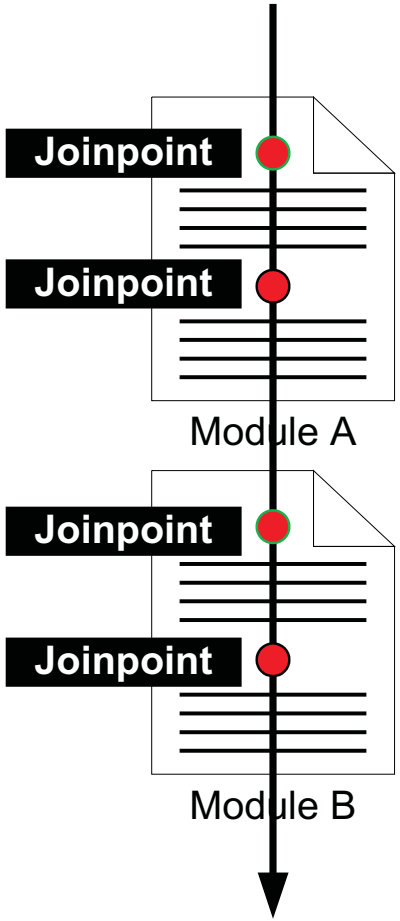
	ACC	ECC	LCC
Student	0	1	2
AccessControl	1	1	1
AccessControlView	1	0	1

Table of Contents

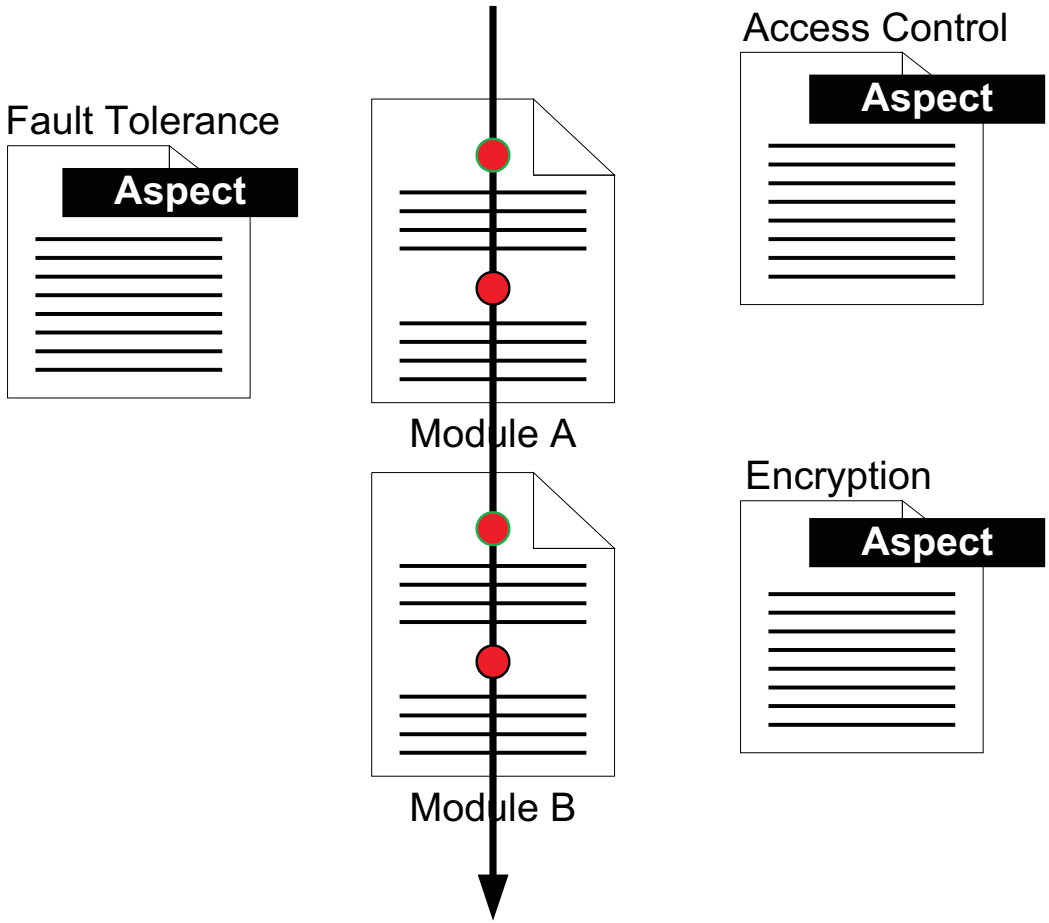
Aspect-Oriented Programming Principle



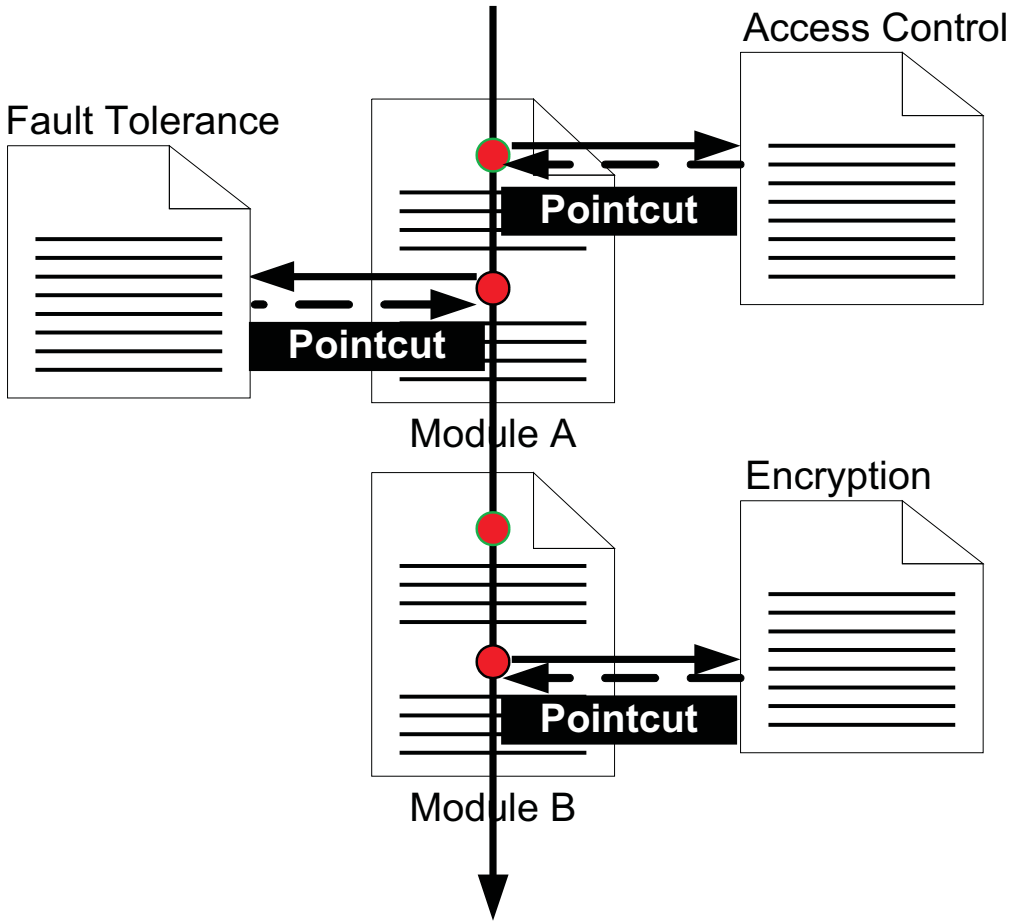
Aspect-Oriented Programming Principle



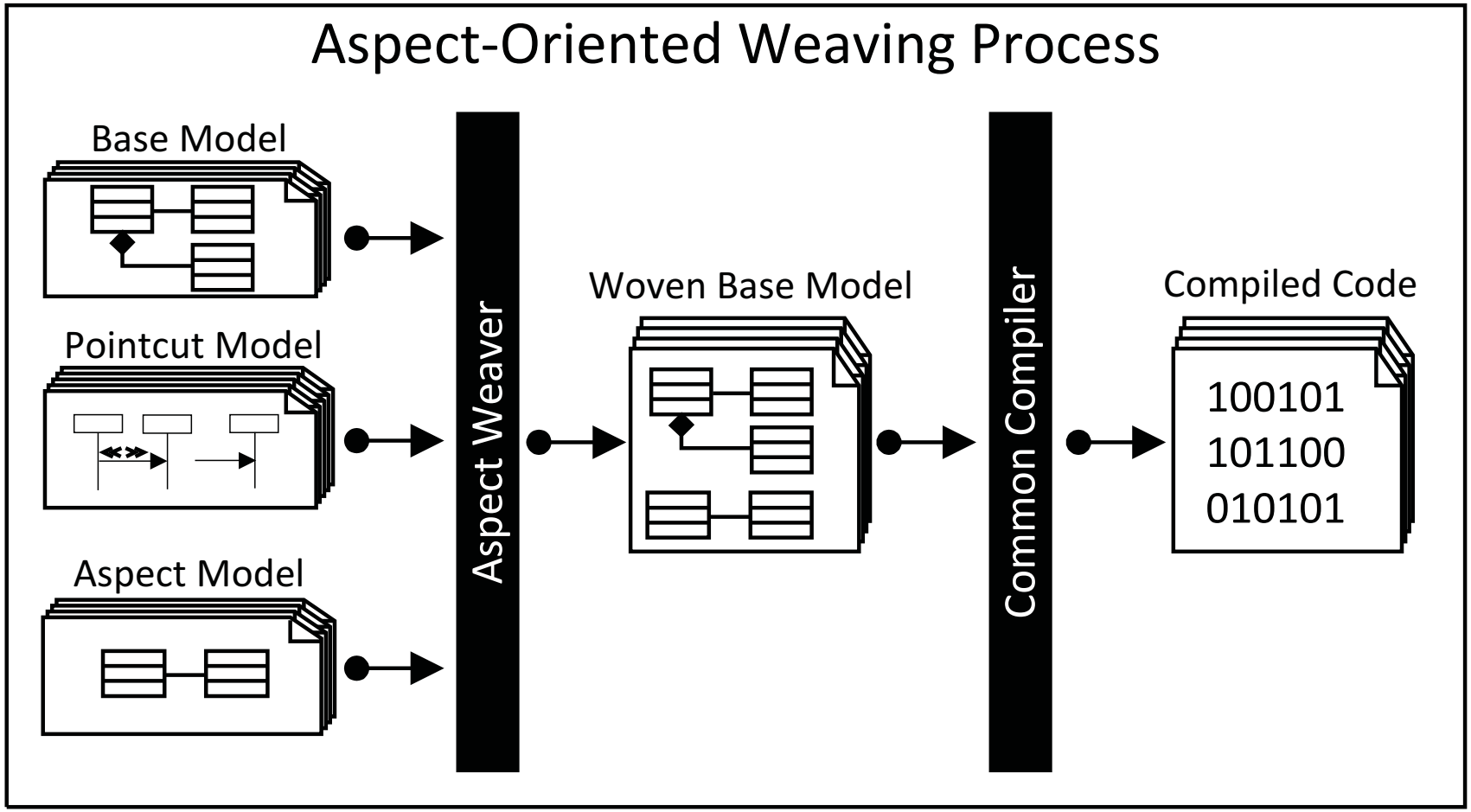
Aspect-Oriented Programming Principle



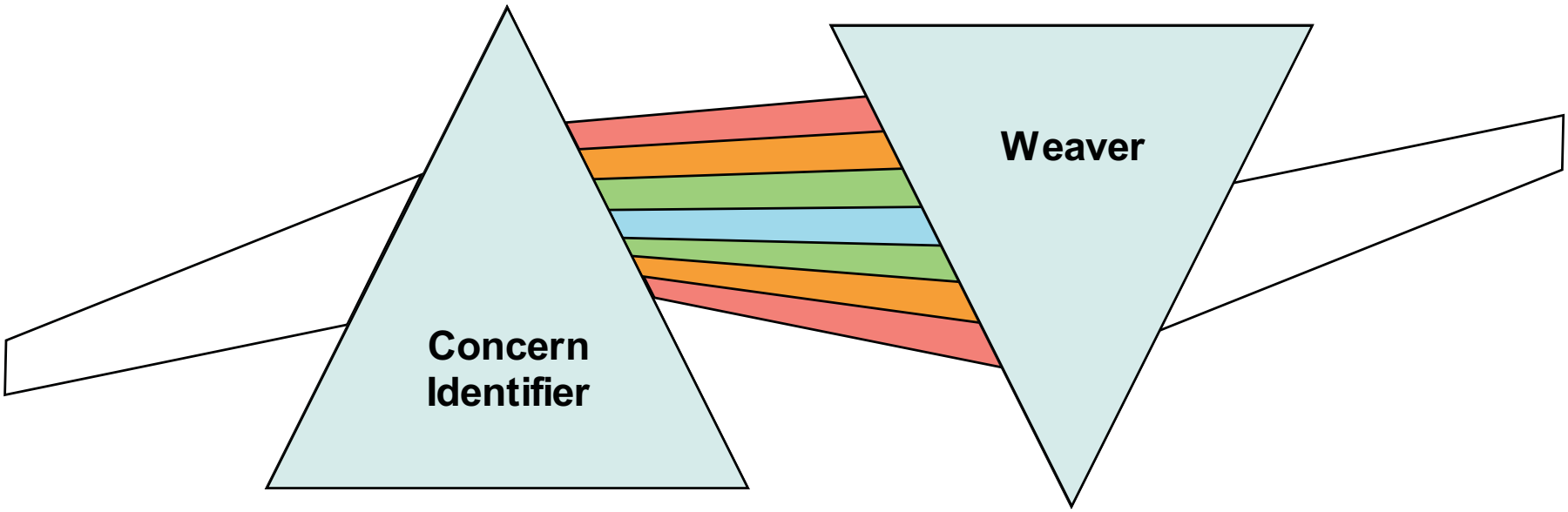
Aspect-Oriented Programming Principle



Weaving Process



Weaving Process



Kinds of Aspect-Oriented Weaving

Static Weaving Aspects are woven at compile time [Kiczales et al., 2001].

Load-Time Weaving Aspects are woven at class load-time [Laddad, 2001, Chiba, 2000].

Dynamic Weaving Aspects can be woven and even unwoven at runtime [Pinto et al., 2005, Suvée et al., 2003]. It is particulary interesting for context-aware systems [Fuentes et al., 2009].

Access Control With AspectJ

```
public aspect AccessControlAspect {

  pointcut guardedMethods():
              call(String Student.getReligion());

  String around() : guardedMethods() {

    AccessControl guard = new AccessControl();

    String returnValue = null;

    if (guard.getAccess()) {
        returnValue = proceed();
    } // if

    return returnValue;
 } // around guardedMethods
} // AccessControlAspect
```

A Clean Student Class

```
public class Student {

      protected String religion;

      public String getReligion() {
            return this.religion;
      } // getReligion

} // Student
```

Wildcard-based Quantified Pointcuts

```
public aspect QuantifiedAccessControlAspect {

   pointcut guardedMethods(): call(* Student.get*(..));

   Object around() : guardedMethods() {

      AccessControl guard = new AccessControl();

      Object returnValue = null;

      if (guard.getAccess()) {
         returnValue = proceed();
      } // if

      return returnValue;
   } // around guardedMethods

} // QuantifiedAccessControlAspect
```

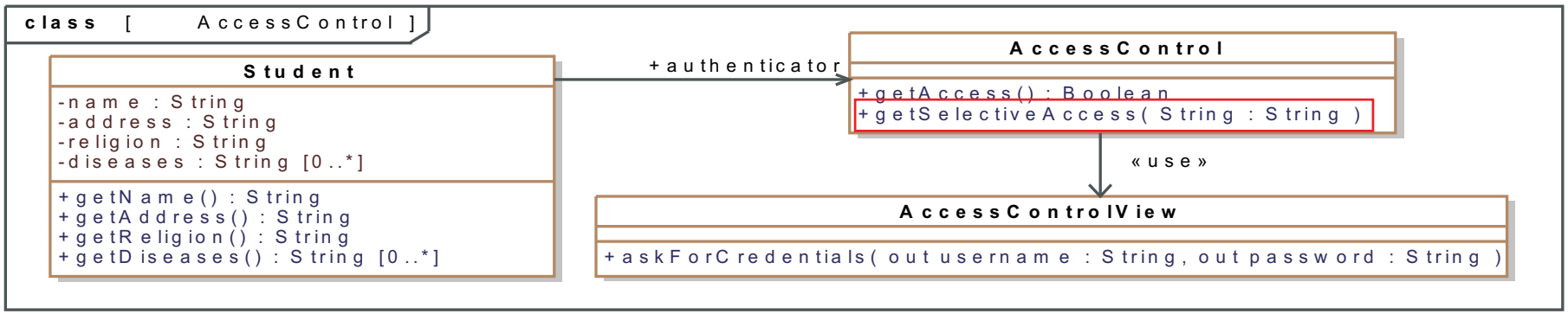
Annotation-based Quantified Pointcuts

```
public class Student {

      protected String religion;

      @AccessControlRequired
      public String getReligion() {
            return this.religion;
      } // getReligion

} // Student
```

Annotation-based Quantified Pointcuts

```
public aspect AnnotationBasedAspect {

  pointcut guardedMethods():
                call(@AccessControlRequired * *.*(..));

  Object around() : guardedMethods() {

    AccessControl guard = new AccessControl();

    Object returnValue = null;

      if (guard.getAccess()) {
         returnValue = proceed();
      } // if

      return returnValue;
  } // around guardedMethods

} // AnnotationBasedAspect
```

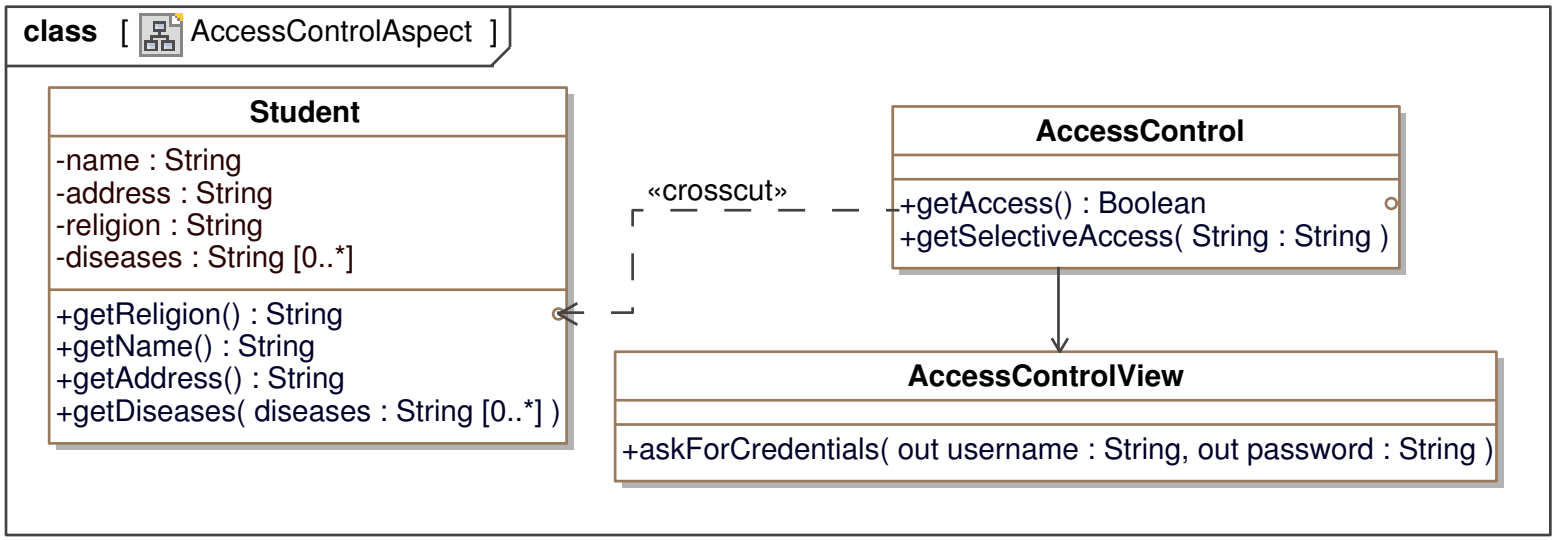
Some Basic & Free Reflection



Some Basic & Free Reflection

```
public aspect ReflectionAccessControlAspect {

   pointcut guardedMethods():
            call(String Student.getReligion());

   String around() : guardedMethods() {

      AccessControl guard = new AccessControl();

      String returnValue = null;

      if (guard.getSelectiveAccess(
                  thisJoinPoint.getSignature().
                                getName())) {
            returnValue = proceed();
         } // if

         return returnValue;
      } // around guardedMethods


} // ReflectionAccessControlAspect
```

Concern-Oriented Metrics Revisited



	CDC	CDO	CIC	OOC
Data Storage (base)	1 (1)	4 (4)	0 (1)	0 (1)
AccessControl	2 (2)	3 (5)	0 (1)	0 (1)

	ACC	ECC	LCC
Student	1 (0)	0 (1)	1 (2)
AccessControl	0 (1)	2 (1)	1 (1)
AccessControlView	1 (1)	0 (0)	1 (1)

Table of Contents

Current Hot Research Topics on AOSD

1. Semantic pointcuts [Ostermann et al., 2005].
2. Reusable aspects (e.g Complex Transaction Management) [Kienzle and Gélineau, 2006].
3. Aspect interference/conflicts [Douence et al., 2002, Douence et al., 2004, Altahat et al., 2008].
4. Complex pointcuts [Allan et al., 2005].
5. Aspect Influence on Interfaces [Ostermann, 2008].

Current Hot Research Topics on AOSD

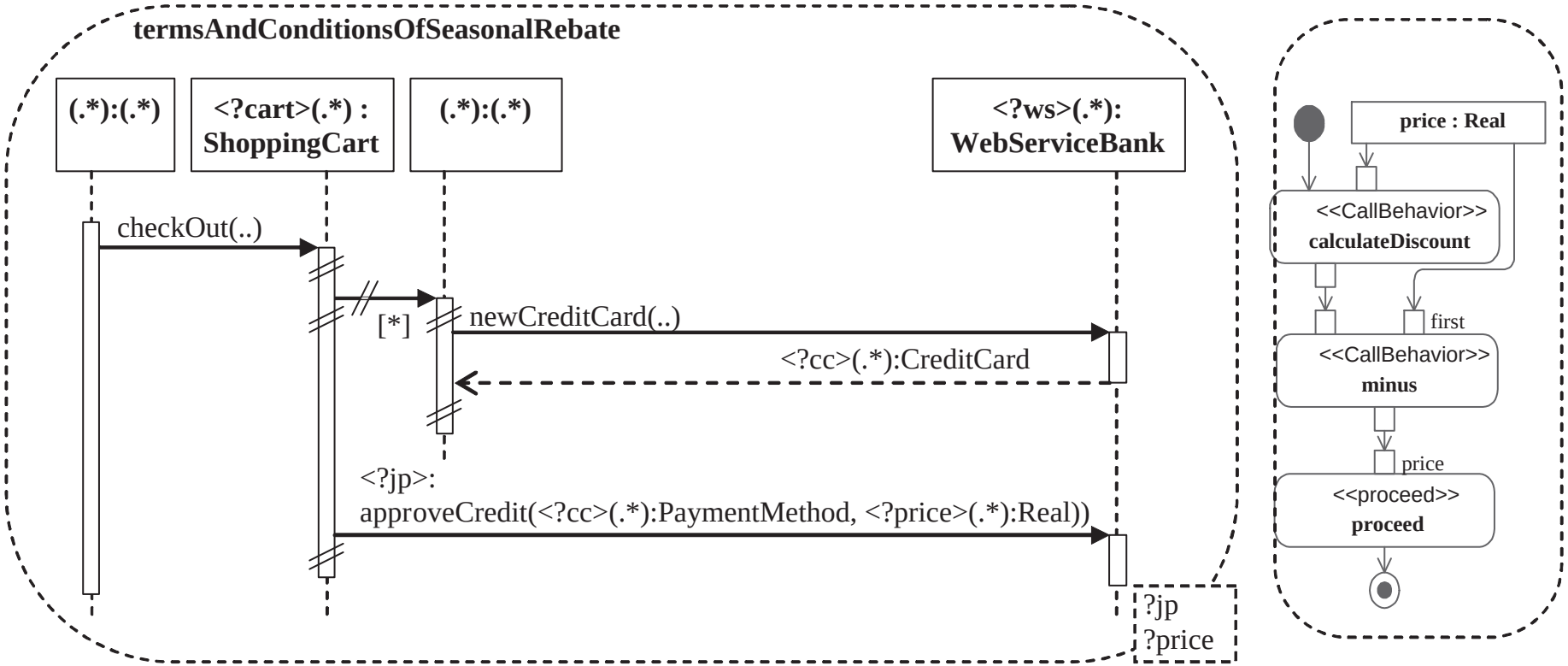
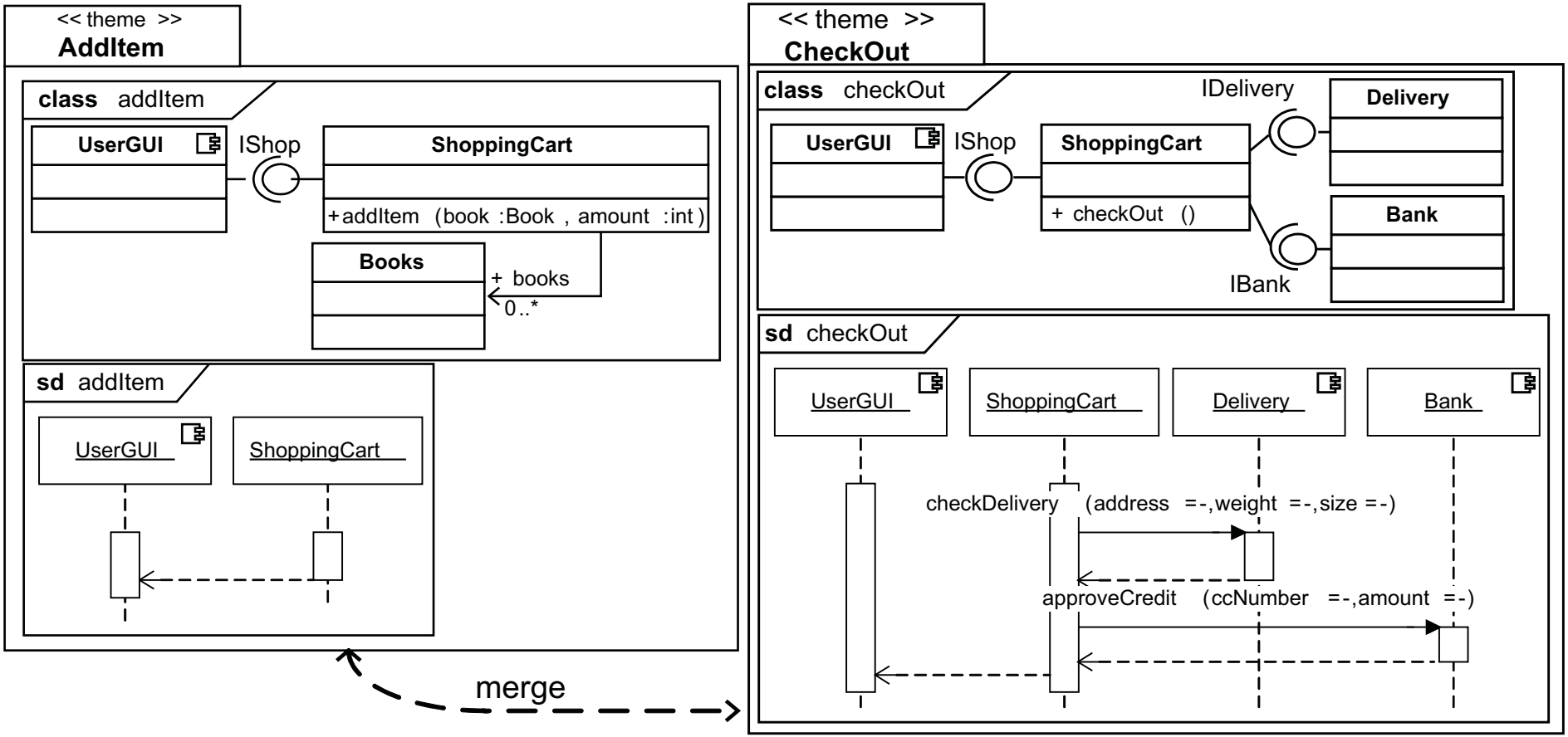


Table of Contents

AOSD is not View-Based Software Development



AOSD is not Feature-Oriented Software Development

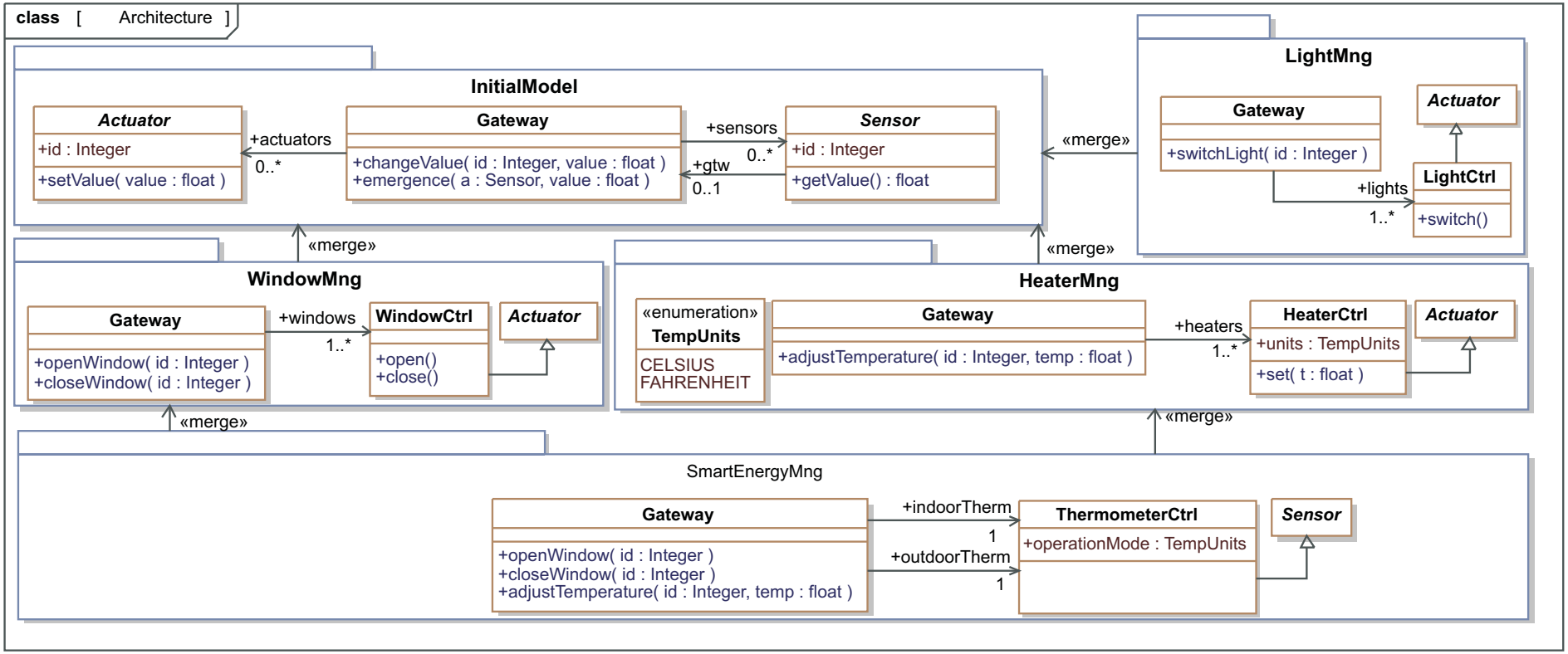


Table of Contents

AOSD and Related Technology

1. **Component platforms/Application Servers/Container Technology**
   - Component platforms offer a set of fixed services.
   - Configuration of each service is often different.
   - Component platforms are often heavier.
2. **Reflection:** AOP is reflection for human beings.
3. **Design Patterns, Dynamic Proxies:** The aspect-oriented weaver does it for you.

AOSD in Industry

1. Spring framework for Enterprise Java.
2. SpringSource dm Server, SpringSource tc Server, Spring Roo y Magma.
3. GlassBox, Perf4J, Contract4J, JXInsight, MaintainJ.
4. Websphere, MySQL, JBoss, Oracle Toplink, J2ME [Rashid et al., 2010].
5. Siemens Healthcare [Rashid et al., 2010].
6. Motorola Weavr [Cottenier et al., 2007].
7. Oficina Virtual Fundación Retevisión [Pinto et al., 2005]
8. Digital Publishing [de Beeck et al., 2009].
9. Isis Project (Stop Pedophilia).

Table of Contents

Conclusions

Aspect-Oriented Software Development

Aspect-Oriented Software Development can help to improve modularization of crosscutting concerns, easing maintenance and evolution.

Access Control Code:
`http://personales.unican.es/sanchezbp/teaching/ao/accesscontrol.zip`

More Information:
`http://www.aosd-europe.net`

Table of Contents

Referencias I


Allan, C., Avgustinov, P., Christensen, A. S., Hendren, L. J., Kuzins, S., Lhoták, O., de Moor, O., Sereni, D., Sittampalam, G., and Tibble, J. (2005).
Adding trace matching with free variables to AspectJ.
In Johnson, R. E. and Gabriel, R. P., editors, *Proc. of the 20th Annual Conference on Object-Oriented Programming, Systems, Languages, and Applications, (OOPSLA)*, pages 345–364.


Altahat, Z., Elrad, T., and Tahat, L. (2008).
Applying Critical Pair Analysis in Graph Transformation Systems to Detect Syntactic Aspect Interaction in UML State Diagrams.
In *Proc. of the 20th Int. Conference on Software Engineering & Knowledge Engineering (SEKE)*, pages 905–911, San Francisco (California, USA).

Referencias II

Brichau, J. and D'Hondt, T. (2005).
Introduction to Aspect-Oriented Software Development.
Deliverable D17 AOSD-Europe-VUB-02, AOSD-Europe Network of Excellence, Brussels.

Chiba, S. (2000).
Load-Time Structural Reflection in Java.
In Bertino, E., editor, *Proc. of the 14th European Conference on Object-Oriented Programming (ECOOP)*, pages 313–336, Sophia Antipolis and Cannes (France).

Colyer, A., Clement, A., Harley, G., and Webster, M. (2004).
Eclipse AspectJ: Aspect-Oriented Programming with AspectJ and the Eclipse AspectJ Development Tools.
Addison-Wesley Professional.

Referencias III


Cottenier, T., van den Berg, A., and Elrad, T. (2007).
Motorola WEAVR: Aspect and model-Driven Engineering.
Journal of Object Technology (JOT), 6(7):51–88.


de Beeck, S. O., Landuyt, D. V., Truyen, E., and Joosen, W. (2009).
Building a Next-Generation Digital News Publishing Platform with AOSD.
In *Proc. of the 8th Int. Conference on Aspect-Oriented Software Development (AOSD)*, Charlottesvile (Virginia, USA). Demostration.

Referencias IV

Douence, R., Fradet, P., and Südholt, M. (2002).
A Framework for the Detection and Resolution of Aspect Interactions.
In Batory, D. S., Consel, C., and Taha, W., editors, *Proc. of the Int. Conference on Generative Programming and Component Engineering (GPCE)*, volume 2487 of *LNCS*, pages 173–188, Pittsburgh, (Pennsylvania, USA).

Douence, R., Fradet, P., and Südholt, M. (2004).
Composition, Reuse and Interaction Analysis of Stateful Aspects.
In *Proc. of the 3rd Int. Conference on Aspect-Oriented Software Development (AOSD)*, pages 141–150, Lancaster (UK).

Elrad, T., Akşit, M., Kiczales, G., Lieberherr, K., and Ossher, H. (2001).
Discussing Aspects of AOP.
Communications of the ACM, 44(10):33–38.

Referencias V

Filman, R. E., Elrad, T., Clarke, S., and Akşit, M., editors (2004).
Aspect-Oriented Software Development.
Addison-Wesley, Boston.

Fuentes, L., Gámez, N., and Sánchez, P. (2009).
Aspect-Oriented Design and Implementation of Context-Aware Pervasive Applications.
Innovations in Systems and Software Engineering, 5(1):79–93.

Hannemann, J. and Kiczales, G. (2002).
Design Pattern Implementation in Java and AspectJ.
In *Proc. of the 17th Int. Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, pages 161–173, Seattle (Washington, USA).

Referencias VI


Kiczales, G., Hilsdale, E., Hugunin, J., Kersten, M., Palm, J., and Griswold, W. G. (2001).
An Overview of AspectJ.
In Knudsen, J. L., editor, *Proc. of the 15th European Conference on Object-Oriented Programming (ECOOP)*, volume 2072 of *Lecture Notes in Computer Science*, pages 327–353, Budapest (Hungary).


Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C. V., Loingtier, J.-M., and Irwin, J. (1997).
Aspect-Oriented Programming.
In Akşit, M. and Matsuoka, S., editors, *Proc. of the 11th European Conference on Object-Oriented Programming (ECOOP)*, volume 1241 of *Lecture Notes in Computer Science*, pages 220–242, Jyväskylä (Finland).

Referencias VII


Kienzle, J. and Gélineau, S. (2006).
AO challenge - Implementing the ACID properties for Transactional Objects.
In Filman, R. E., editor, *Proc. of the 5th Int. Conference on Aspect-Oriented Software Development (AOSD)*, pages 202–213.


Laddad, R. (2001).
I want my AOP!
Java World,
http://www.javaworld.com/javaworld/jw-01-2002/jw-0118-aspe


Ostermann, K. (2008).
Reasoning about Aspects with Common Sense.
In D'Hondt, T., editor, *Proc. of the 7th Int. Conference on Aspect-Oriented Software Development (AOSD)*, pages 48–59, Brussels (Belgium).

Referencias VIII

Ostermann, K., Mezini, M., and Bockisch, C. (2005).
Expressive Pointcuts for Increased Modularity.
In Black, A. P., editor, *Proc. of the 19th European Conference on Object-Oriented Programming (ECOOP)*, volume 3586 of *LNCS*, pages 214–240, Glasgow (UK).

Pinto, M., Fuentes, L., and Troya, J. M. (2005).
A Dynamic Component and Aspect-Oriented Platform.
Computer Journal, 48(4):401–420.

Rashid, A., Cottenier, T., Greenwood, P., Chitchyan, R., Meunier, R., Coelho, R., Südholt, M., and Joosen, W. (2010).
Aspect-Oriented Software Development in Practice: Tales from AOSD-Europe.
IEEE Computer, 43(2):19–26.

Referencias IX


Sant'Anna, C., Figueiredo, E., Garcia, A. F., and de Lucena, C. J. P. (2007).
On the Modularity of Software Architectures: A Concern-Driven Measurement Framework.
In Oquendo, F., editor, *Proc. of the 1st European Conference on Software Architecture (ECSA)*, pages 207–224, Aranjuez (Spain).


Suvée, D., Vanderperren, W., and Jonckers, V. (2003).
JAsCo: an Aspect-Oriented Approach Tailored for Component-Based Software Development.
In *Proc. of the 2nd Int. Conference on Aspect-Oriented Software Development (AOSD)*, pages 21–29, Boston (Massachusetts, USA).

Questions ?