

AOSD Tools & Research

Corso 3° livello



SoftEng
<http://softeng.polito.it>

Paolo Falcarin

AOP tools

- **AspectC++** is an AOP extension to C++
 - <http://www.aspectc.org/>
- **Aspect#**: adding aspect support to C#
 - <http://aspectsharp.sourceforge.net/>
- **AspectDNG**: based on .NET platform
 - <http://www.dotnetguru.fr/aspectdng/>
- **AspectS** is a simple AOP extension to Smalltalk.
- **Pythius** is a project for AOP in Python

AOP related methodologies

- **IBM Hyper/J** supports "multi-dimensional" separation and integration of concerns in standard Java software.
- **Afanti** : project aimed at developing tools to support testing and analysis of aspect-oriented software.
- **Composition Filters**: is a modular extension to the object-oriented model that allows the modular specification of aspects in a composable way.
- **Abc-compiler**: is an Aspect-Oriented compiler of compilers: provides a new frame processor language to implement generators, e.g., aspect weavers.

Java & Dynamic AOP

- AOP platforms allowing to insert and withdraw aspects at runtime....
- Why and Which concerns?
 - ◆ Middleware and protocols update
 - ◆ Quality of Service adaptation
 - ◆ Dynamic reconfiguration and adaptation
 - ◆ JVM not reachable (robots, mobile devices)
 - ◆ Context-adaptation of applications
 - ◆ Spontaneous containers

Dynamic AOP tools

- **JASCO**: best performance, and also for C#
- **JAC** is a Java framework for AO distributed programming: sponsored by ObjectWeb App. Server
<http://jac.objectweb.org/>
- **Aspectwerkz**: sponsored by BEA WebLogic: now it has joined to AspectJ in AspectJ 5 (pointcuts based on Java 5 annotations)
- **JBoss/AOP**
- **PROSE** is an AOP platform based on the Java Virtual Machine that allows dynamic weaving and un-weaving.
- **Spring Framework** (based on AspectJ 5)
 - <http://www.springframework.org/>

JBoss-AOP

- Jboss is a reflective and reconfigurable Java application server
- It uses the JBoss AOP framework – where aspects are implemented as interceptors – to provide a set of crosscutting services (e.g. persistence, remote access etc.).
- JBoss uses load time weaving with a custom class loader (called unified class loader) that:
 - ◆ It loads all classes into a flat namespace, for the components to be able to share non-system classes,
 - ◆ inserts hooks into the bytecode to allow interceptor attachment for interception for field, constructor, and method manipulation

PROSE

- Prose framework uses JIT compiler weaving, and allows both total and actual hook weaving.
- It relies on JVMTI interface provided by all JVMs in Java 5.
- It is lightweight and also runs on mobile platforms (PDA)
- <http://prose.ethz.ch>

Dynamic AOP tools

Dynamic AOP tools can be distinguished in two types:

1 Fixed pointcut definition

Join-points are identified at first-deployment

You can replace advice implementation, but...

...you cannot pick up new join points

In some tools, you can redefine pointcuts

But you can select a sub-set of join-points

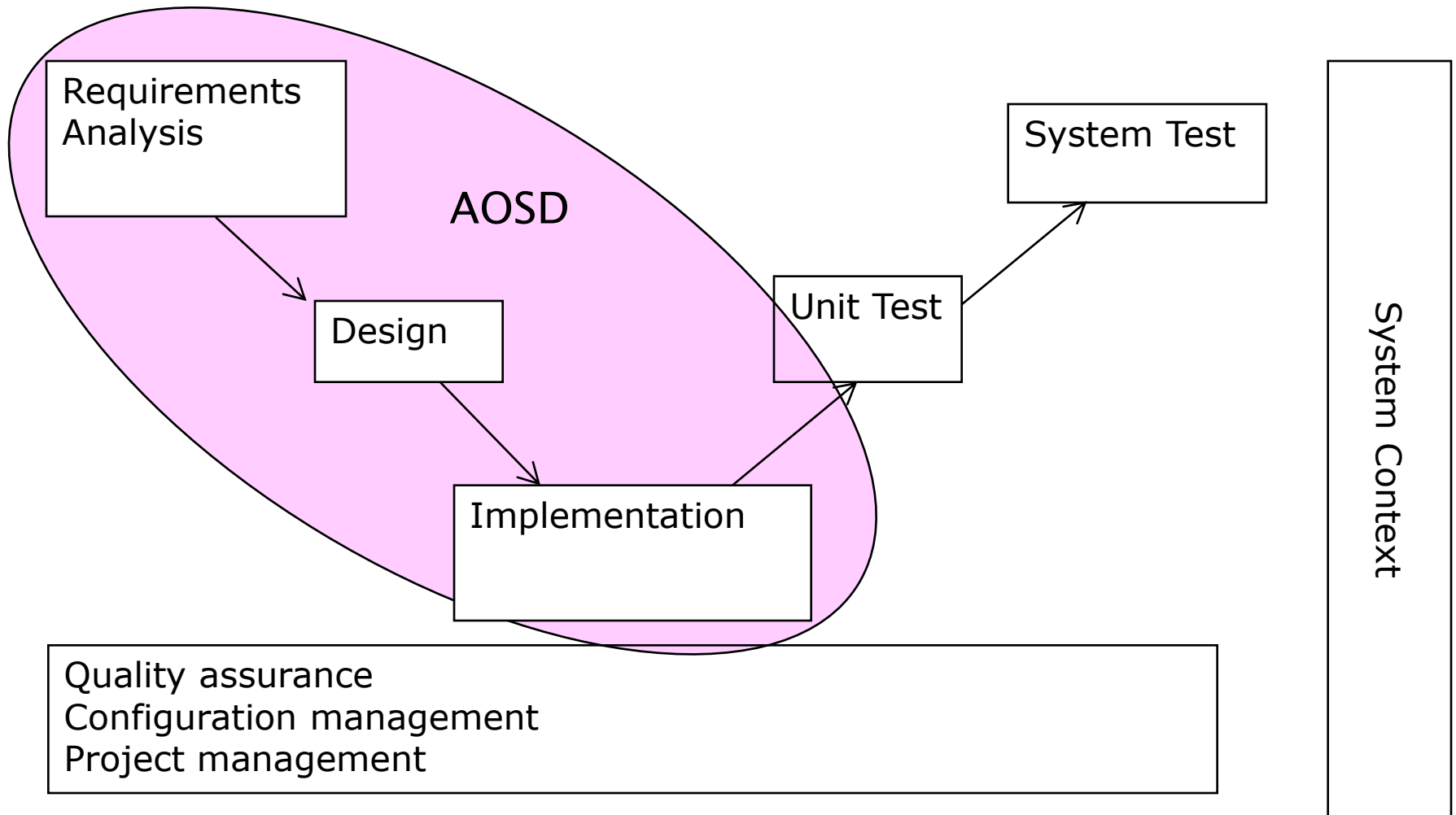
identified at first deployment

2. Variable pointcut definition

Can pick up new join points

More powerful

AO Software Development



AOP at analysis & design level

The *Early Aspects Initiative* (<http://www.early-aspects.net/>) focuses on managing crosscutting properties at the early development stages of:

- requirements engineering
- architecture design

Other existing projects:

- **Aspect-Oriented Software Engineering**
- **AO Requirement Engineering**
- **Theme/UML** is one of the many aspect-oriented extension to the UML.
- **XBel** is a framework for reasoning about compositions of concerns on the requirements level.

AO Requirement Engineering

- How to identify aspects at the requirements level?
 - ◆ What is the relationship between aspects and non-functional requirements, constraints and concerns?
- How to model aspects at the requirements level?
- How to trace requirements level aspects through later development stages and during re-engineering?
- How to validate aspects identified at the requirements level?

Identifying Aspects (1)

- Solution 1: Model using viewpoints/use cases/scenarios/stakeholder concerns and then identify crosscutting requirements.
 - + Exploit the power and potential of existing mechanisms
 - Scalability problems. Hard to observe crosscutting in the presence of a large number of viewpoints, use cases or scenarios.

Identifying Aspects (2)

- Solution 2: Brainstorming without the modelling.
 - + No effort required for initial structuring
 - ? Very close to eXtreme Programming
 - You'll probably miss something

You could find aspect you might not find using other techniques

Identifying Aspects (3)

- Solution 3: Look at global properties, non-functional requirements and constraints as they are potential aspects
 - + Easier to identify
 - may not be global

Modeling Aspects

- Extension of existing modelling mechanisms e.g. OO, FSM, Concern-based
 - + Ride the power of existing techniques
 - + Ease of integration with existing techniques, tools, etc.
 - + Ease of learning
 - Inherit the shortcomings of existing techniques

AO Modeling

- There are a lot of proposals (2 categories):

1. UML extensions (UML profiles)

- 1.1. I. Groer, S. Schulze
- 1.2. M. Basch, A. Sanchez
- 1.3. M.E. Fayad, A. Ranganath

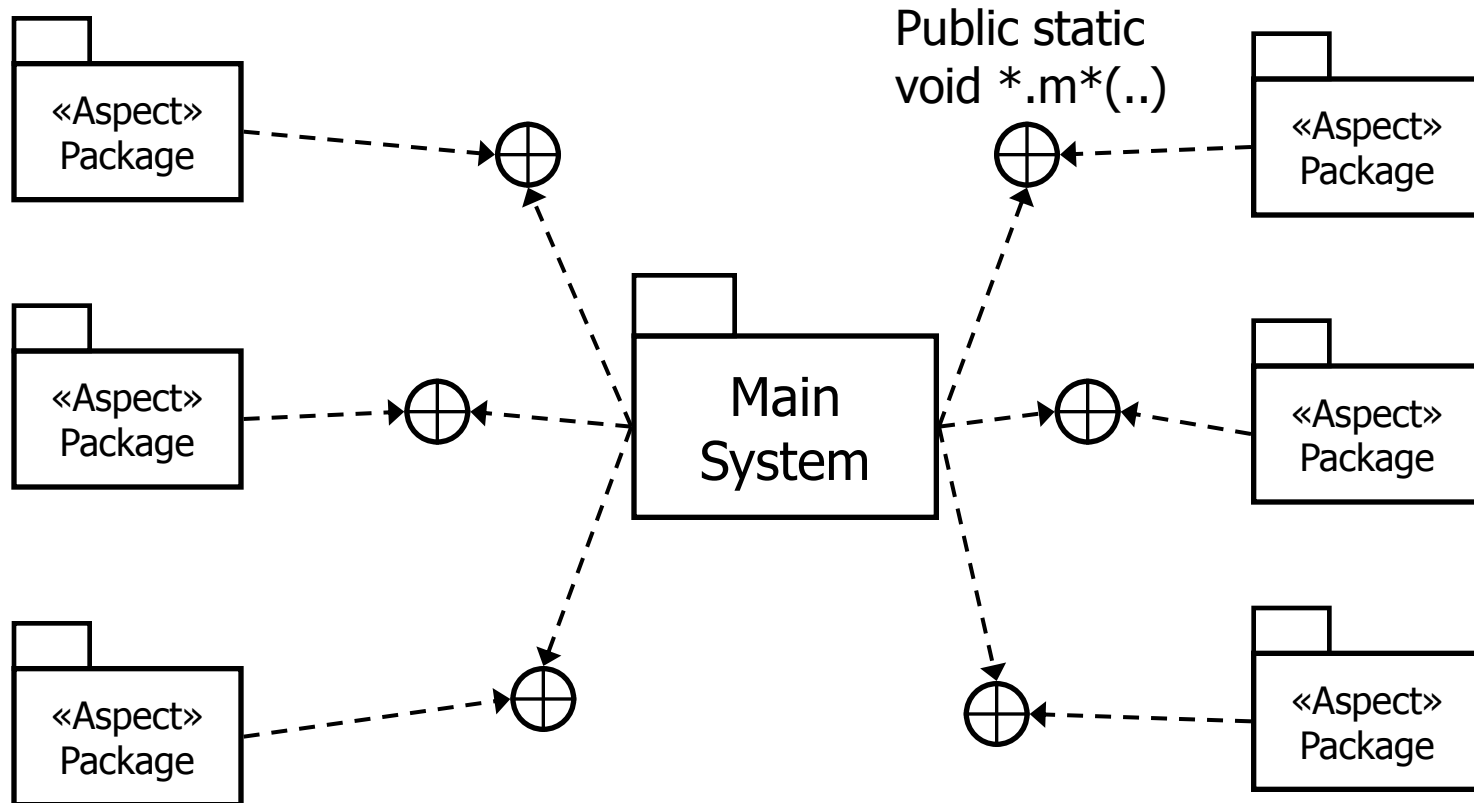
2. UML meta-model extension

- 2.1. J. Whittle, J. Araújo, D. Kim
- 2.2. M. Saeki, H. kaiya
- 2.3. G. Georg, R. France, I. Ray

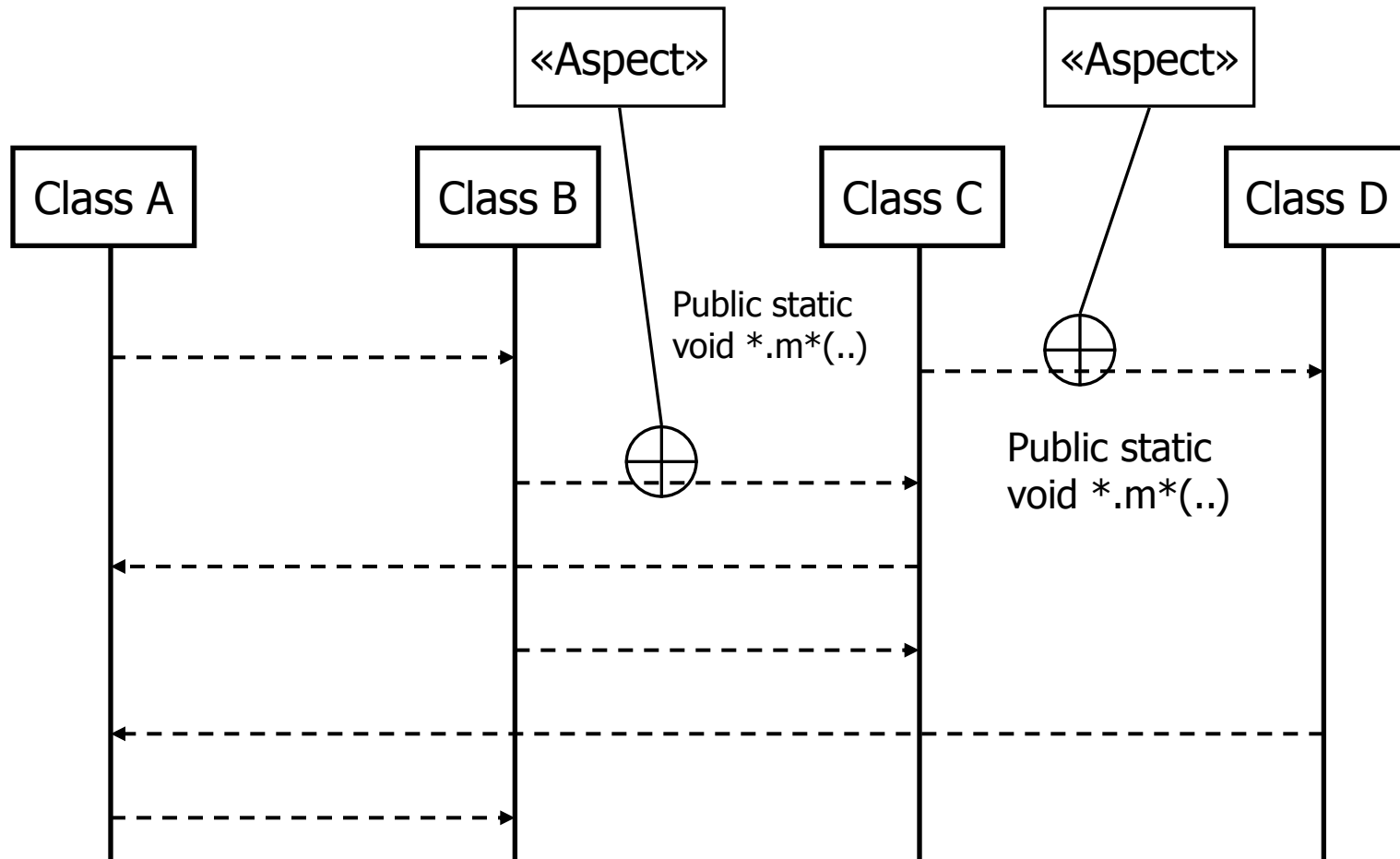
AOM: UML extensions

- UML is extensible with profiles made up of:
 - ◆ **Stereotypes**: new elements extending UML elements
 - ◆ **Tagged Values**: name–value properties
 - ◆ **Constrains**: defined in in OCL
- They are compliant with the current UML standard

AOM: M. Basch, A.Sanchez



AOM: M. Basch, A.Sanchez



AOM: A new UML Meta-model

- UML meta-model defines rules and constraints on all UML diagrams elements
- These approaches aim modifying UML standard

Re-engineering: Aspect Mining

- Tool to find aspects in your code. Some form of code mining and extract patterns of crosscutting.
 - + Tool support
 - Effort to develop a tool: accuracy ?
- **Aspect Browser** is a graphical tool that uses the map metaphor to assist finding, exploring, and manipulating crosscutting concerns in software.
- **Aspect Mining Tool (AMT)** is a tool for identification and exploring latent concerns in software systems.
- **Concern Graphs** are a manipulable representation of concerns that can be extracted from an existing code base.

Other initiatives

AORE (AO Reqs. Eng.): David Schaefer, Joao Araujo, Isabel Brito, Awais Rashid, Claudia Mesquita

FOAL (Foundations Of AO Languages): Semantics, specification of AOP languages. <http://www.cs.iastate.edu/~leavens/FOAL/index.shtml>

CME: Concern Manipulation Environment, IBM

FEAT: <http://www.cs.ubc.ca/labs/spl/projects/feat>

AO Data Bases, Awais Rashid, Univ. Lancaster (UK)

Sync Gen (Gudrup Singh, Masaaki Mizuno) – Kansas Univ.
<http://syncgen.projects.cis.ksu.edu/>

Design Pattern implementation with AOP (Hannemann, Kiczales)
<http://www.cs.ubc.ca/~jan/papers/oopsla2002/oopsla2002.html>

aTrack : Reusable aspect library. <https://atrack.dev.java.net/>

Conclusions

- **AOP advantages**

- ◆ enables flexible software development
- ◆ improve quality with consistent policies
- ◆ manage complexity and variability
- ◆ design decisions mapped to code

- **Open issues**

- ◆ Reuse and design for large aspect libraries
- ◆ AO Modeling
- ◆ AOSD in sw process and requirements
- ◆ Aspect Verification
- ◆ Dynamic AOP

Conclusions

- **AOP advantages**

- ◆ enables flexible software development
- ◆ improve quality with consistent policies
- ◆ manage complexity and variability
- ◆ design decisions mapped to code

- **Open issues**

- ◆ Reuse and design for large aspect libraries
- ◆ AO Modeling
- ◆ AOSD in sw process and requirements
- ◆ Aspect Verification
- ◆ Dynamic AOP

References – AOP

- [1] Aspect-Oriented Software Development website. <http://www.aosd.net/> .
- [2] The AspectJ Team. The AspectJ Programming Guide. <http://eclipse.org/aspectj> .
- [3] M. Aksit, K. Wakita, J. Bosch, L. Bergmans, and A. Yonezawa. Abstracting object-interactions using composition-filters. *Object-Oriented Distributed Processing*, 1993.
- [4] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. V. Lopes, J. Loingtier, and J. Irwan. Aspect-oriented programming. *Proc. of ECOOP 97*.
- [5] M. Mezini and K. Lieberherr. Adaptive plug and play components for evolutionary software development. *Proc. Of OOPSLA'98*.
- [6] Various Authors. Special Issue on Aspect-Oriented Software Development. *Communications of the ACM*, 44(10), October 2001.
- [7] Y. Smaragdakis and D. Batory. Implementing layered designs with Mixin layers. *Proc. of ECOOP 98*.
- [8] P. Tarr, H. Ossher, W. Harrison, and S. Sutton Jr. N-degrees of separation: Multi-dimensional separation of concerns. *Proc. of ICSE 99*.
- [9] Shigeru Chiba. A Metaobject Protocol for C++. In 10th Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA'95), volume 30(10) of SIGPLAN Notices, pages 285{299, Austin, Texas, October 1995. ACM.
- [10] Yvonne Coady and Gregor Kiczales. Back to the future: a retroactive study of aspect evolution in operating system code. In Mehmet Aksit, editor, 2nd International Conference on Aspect-Oriented Software Development, Boston, Massachusetts, March 2003. ACM.

References – AOP Tools

- [1] A. Popovici, G. Alonso, and T. Gross. Just in time aspects: Efficient dynamic weaving for Java. *Proc. of the Intl. Conf. on AOSD 2003*.
- [2] PROSE website. On-line at <http://prose.ethz.ch/>.
- [3] R. Pawlak, L. Duchien, G. Florin, L. Martelli, and L. Seinturier. Distributed separation of concerns with aspect components. *Proc. of TOOLS Europe 2000*.
- [4] Gunter Kniesel, Pascal Costanza, and Michael Austermann. JMangler – A Framework for Load-Time Transformation of Java Class Files. In IEEE Workshop on Source Code Analysis and Manipulation (SCAM, ICSM), Florence, Italy, November 2001. <http://javalab.cs.uni-bonn.de/research/jmangler/>
- [5] Mira Mezini and Klaus Ostermann. Conquering Aspects with Caesar. In Mehmet Aksit, editor, 2nd Int. Conf. on Aspect-Oriented Software Development, 2003, ACM. <http://caesarj.org/>
- [6] JBoss 4.0 Application Server. <http://www.jboss.org/> .
- [7] Aspectwerkz. On-line at <http://aspectwerkz.codehaus.org/>
- [8] JASCO. On-line at <http://ssel.vub.ac.be/jasco/>
- [9] Steamloom
- [10] aspectC . On-line at www.cs.ubc.ca/labs/spl/projects/aspectc.html
- [11] Mahrenholz, D., Spinczyk, O., and Schröder-Preikschat, W. Program Instrumentation for Debugging and Monitoring with AspectC++. In *Proc. of the 5th IEEE Int. Symposium on Object-oriented Real-time Distributed Computing*. On-line at www.aspectc.org/

References – security

- [1] Viren Shah and Frank Hill. Using Aspect–Oriented Programming for Addressing Security Concerns. In International Symposium on Software Reliability Engineering (ISSRE), 2002.
- [2] John Viega, J.T. Bloch, and Pravir Chandra. Applying Aspect–Oriented Programming to Security. Cutter IT Journal, 14(2), February 2001.
- [3] Ian S.Welch and Robert J. Stroud. Re–engineering Security as a Crosscutting Concern. The Computer Journal, 46(5):578{589, September 2003.
- [4] Bart De Win, Bart Vanhaute, and Bart De Decker. Security Through Aspect–Oriented Programming. In B. De Decker, F. Piessens, J. Smits, and E. Van Herreweghen, editors, Advances in Network and Distributed Systems Security, volume 206 of IFIP International Federation for Information Processing, 2001. Kluwer Academic Publishers.
- [5] Bart De Win, Wouter Joosen, and Frank Piessens. AOSD & Security: a practical assessment. In Workshop on Software engineering Properties of Languages for Aspect Technologies (SPLAT, AOSD), Boston, Massachusetts, March 2003.
- [6] Bart De Win, Wouter Joosen, and Frank Piessens. AOSD as an enabler for good enough security. Proceedings of the First ACM Workshop on Business Driven Security Engineering, November 2003.
- [7] Bart De Win, Wouter Joosen, and Frank Piessens. Developing Secure Applications through Aspect–Oriented Programming. In Mehmet Aksit, Siobhan Clarke, Tzilla Elrad, and Robert E. Filman, editors, Aspect–Oriented Software Development. Addison–Wesley, 2004.
- [8] Bart De Win, Frank Piessens, Wouter Joosen, and Tine Verhanneman. On the importance of the separation–of–concerns principle in secure software engineering.

References – apps

- [1] Premkumar Devanbu and Stuart Stubblebine. Software Engineering for Security: a Roadmap. In *The Future of Software Engineering (ICSE2000)*.
- [2] E. Truyen, B. Vanhaute, W. Joosen, P. Verbaeten, and B. N. Jorgensen. Dynamic and selective combination of extensions in component-based applications. *Proc. of ICSE 2001*.
- [3] E. Wohlstadter, S. Jackson, and P. Devanbu. DADO: enhancing middleware to support crosscutting features in distributed, heterogeneous systems. *Proc. of ICSE 2003*.
- [4] Rashid, A. and R. Chitchyan (2003) *Persistence as an Aspect*. 2nd International Conference on Aspect-Oriented Software Development. ACM. Pages 120-129
- [5] Marius Marin, Aspect Mining Project. <http://swerl.tudelft.nl/>

References – others

- [1] Stefan Hanenberg and Rainer Unland. Parametric Introductions. In Mehmet Aksit, editor, 2nd International Conference on Aspect-Oriented Software Development (AOSD), Boston, Massachusetts, March 2003. ACM.
- [2] Robert J. Walker, L. A. Baniassad, and Gail C. Murphy. An Initial Assessment of Aspect-Oriented Programming. In 21st International Conference on Software Engineering (ICSE), Los Angeles, CA, USA, May 1999. ACM/IEEE.
- [3] Dharma Shukla, Simon Fell, and Chris Sells. Aspect-Oriented Programming Enables Better Code Encapsulation and Reuse. MSDN Magazine, March 2002.
- [4] Gail C. Murphy, Albert Lai, Robert J. Walker, and Martin P. Robillard. Separating Features in Source Code: An Exploratory Study. In 23rd International Conference on Software Engineering (ICSE2001), May 2001. IEEE.