Aspect Programming, LLP

The Future of Aspect Oriented Programming

Dean Wampler
Aspect Programming, LLP

The Future of Aspect Oriented Programming

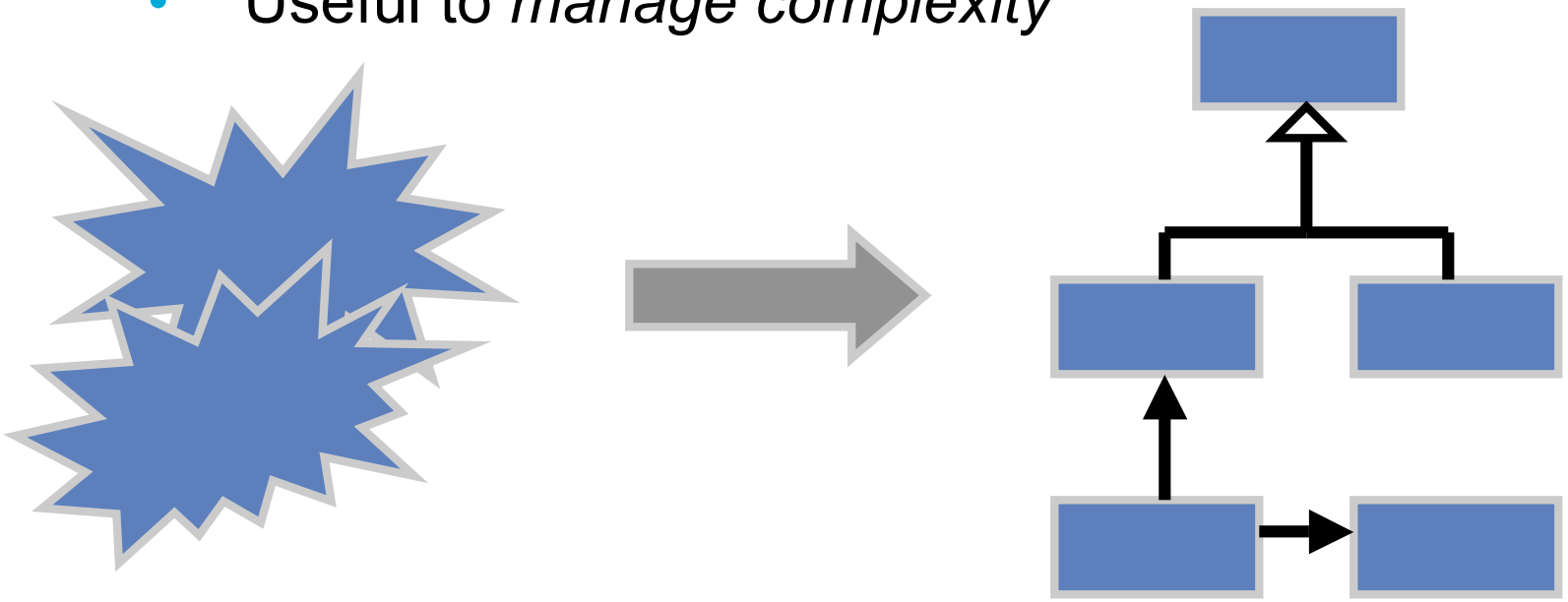
- What Is AOP?
- The Future of AOP
  - What Will Software Look Like?
- AOP Today

The Future of Aspect Oriented Programming

- **What Is AOP?**
- The Future of AOP
  - What Will Software Look Like?
- AOP Today

What is AOP?

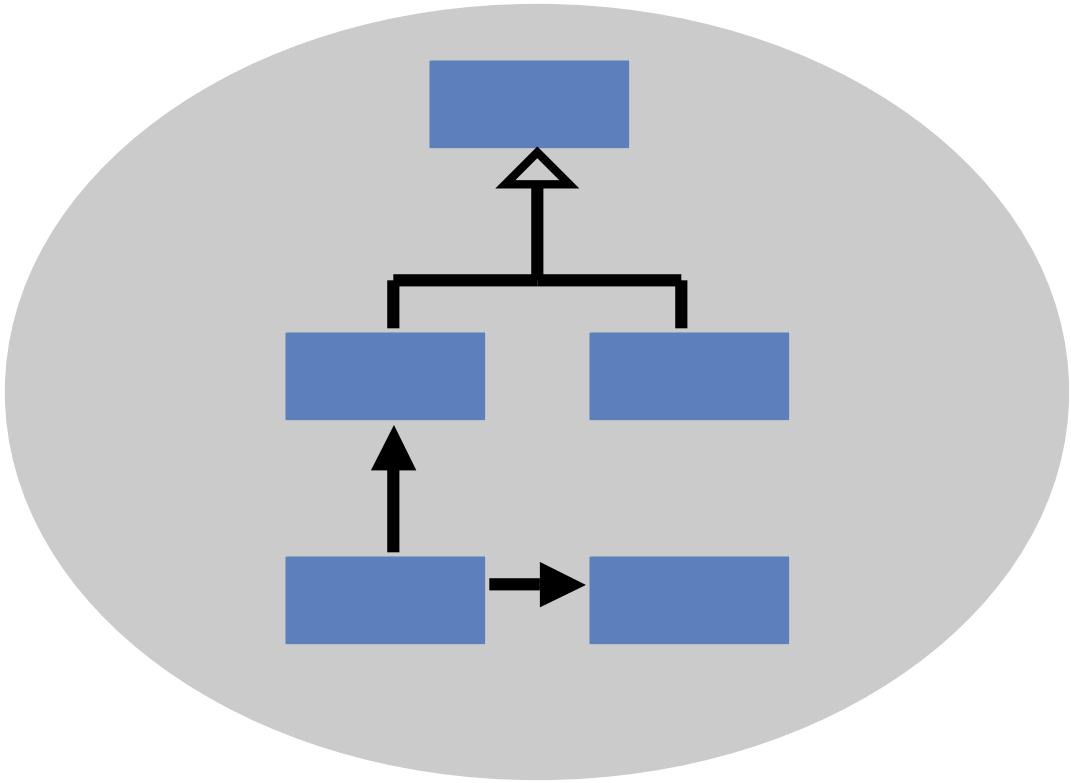
- **Object Oriented Programming**
  - Modularizes the Problem Domain, Design Space
    - Useful to *manage complexity*



What is AOP?

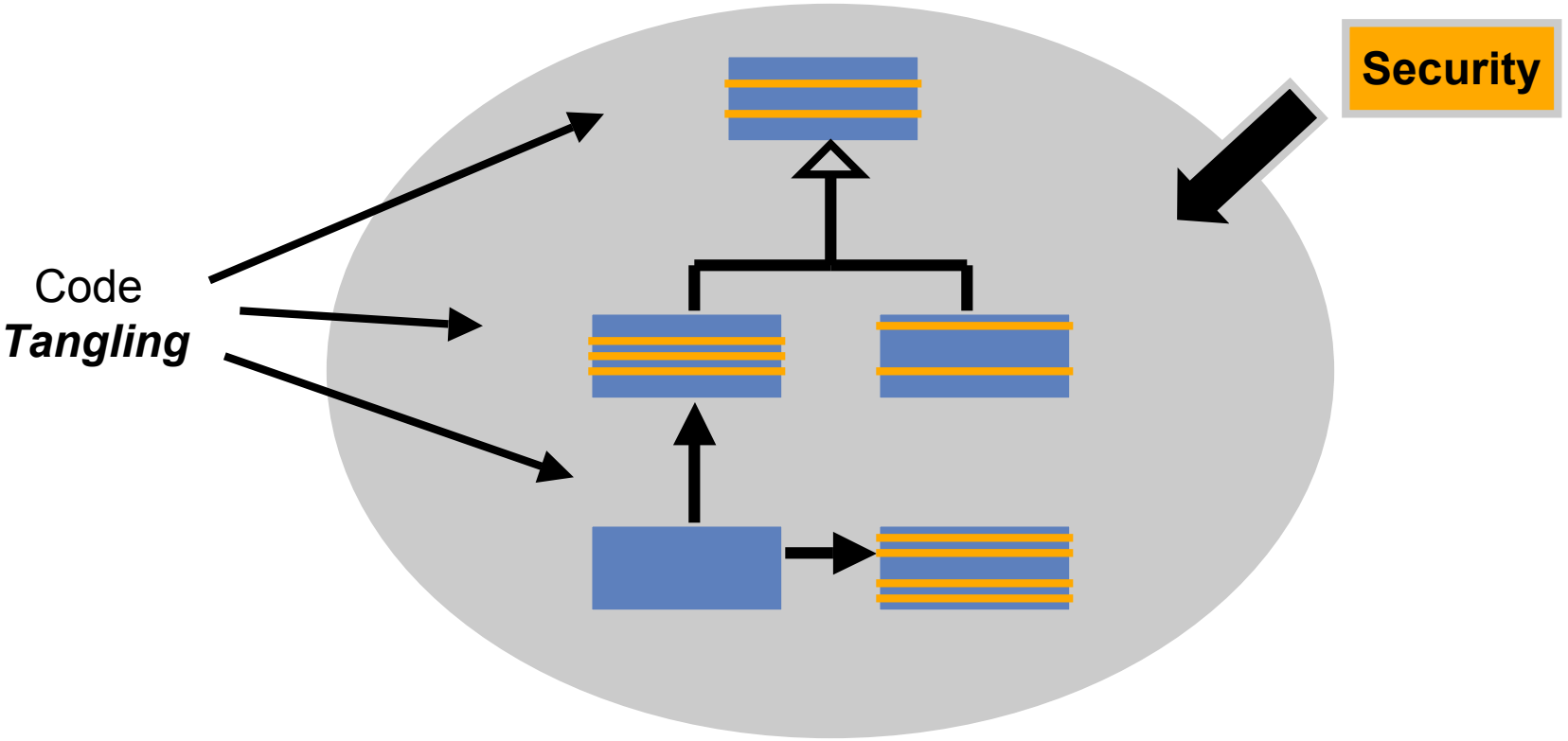
- **Object Oriented Programming**

  One ***dominant decomposition***



What is AOP?

- **Aspect Oriented Programming – Why?**
  - Dominant decomposition ***scatters*** other modules



What is AOP?

- **Aspect Oriented Programming – Why?**
  - Concerns that don't fit in the domain of the dominant decomposition get ***scattered***
    - *E.g.,* security, logging, persistence, …
  - Code from different concerns gets ***tangled*** into objects of the dominant decomposition

What is AOP?

- **AOP's Goal: *Restore Concern Modularity***
  - Identify crosscutting concerns
  - Extract them as separate modules, called ***Aspects***
  - Develop and maintain them independently
  - ***Weave*** the modules together during…
    - » Build - to compose application
    - » Runtime - to dynamically change behavior
- **AOP Modularizes Crosscutting Structure**
- **OOP Modularizes Hierarchical Structure**

What is AOP?

- ***Weaving***
  - The process of combining aspect and object models to create the desired runtime behavior
    - Can happen at build time or dynamically during runtime
    - Depending on technology, ***what*** to weave specified programmatically or declaratively
  - ***Weaving*** *is a fundamentally new technology for application development and runtime modification*

What is AOP?

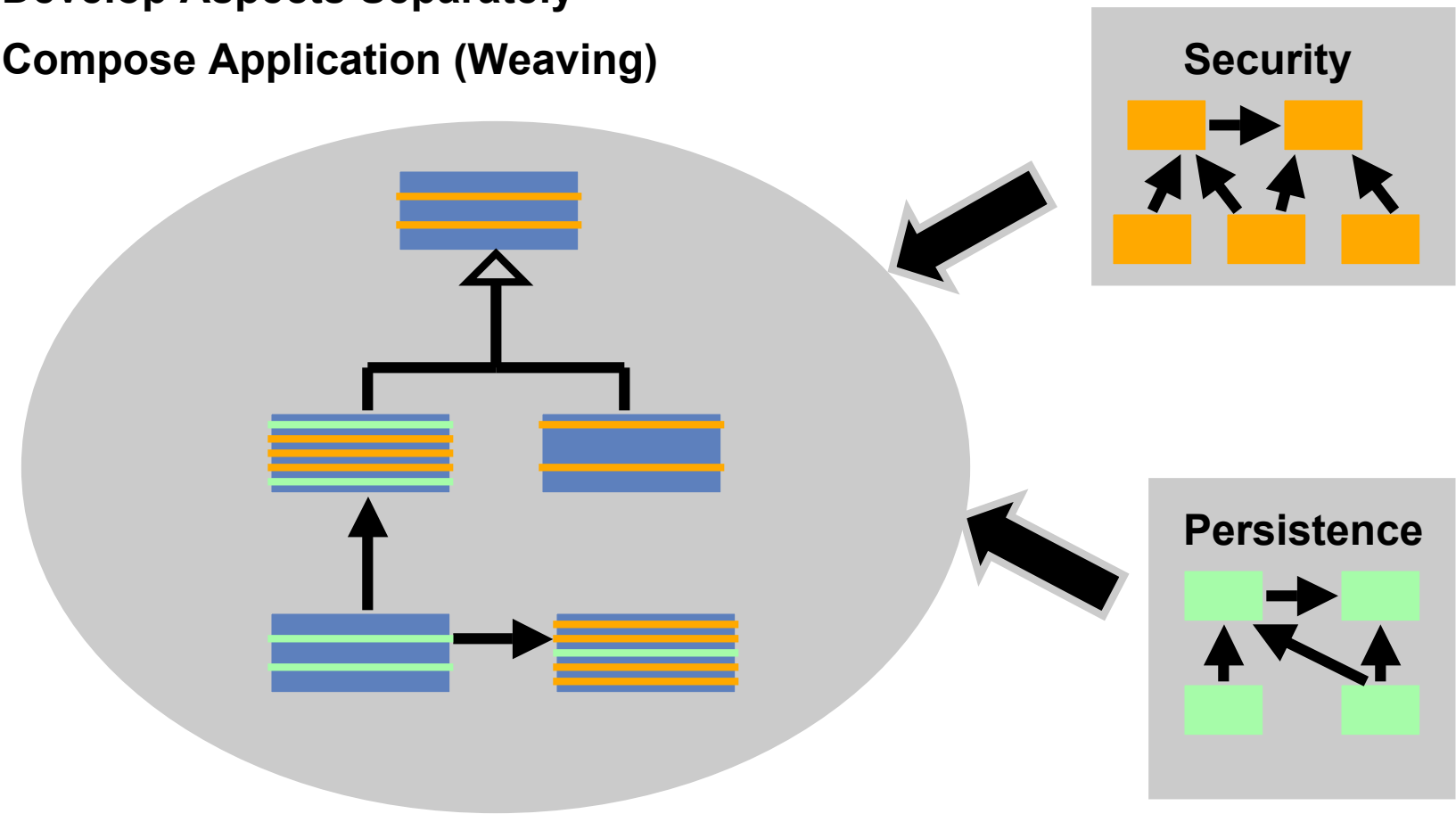
- **Aspect Oriented Programming**
  - Complements, rather than replaces OOP
    - Some concepts are better expressed as objects
    - Other concepts are better expressed as aspects
  - Can even use AOP with procedural code
    - *E.g.,* Univ. of BC project studying aspects in BSD Unix “C” source code.
  - AOP adds another “dimension” of modularity to older approaches

AOP Process (Version 1.0)

#1 Develop Dominant Decomposition (Problem Domain, Framework)

#2 Develop Aspects Separately

#3 Compose Application (Weaving)



The Future of Aspect Oriented Programming

- What Is AOP?
- **The Future of AOP**
  - What Will Software Look Like?
- AOP Today

The Future of Aspect Oriented Programming

- **OOP History – A Useful Comparison**
  - Late 60's: Simula
  - Early 80's: Smalltalk, CLOS, …
    - Mature, sophisticated, but not widely used
  - Late 80's – Early 90's: Mainstream breakout
    - What drove the mainstream breakout?
    - Widespread development of desktop applications with complex GUI's
      - » Structured programming couldn't handle complexity
      - » *Needed OOP modularity - a natural fit!*

The Future of Aspect Oriented Programming

- **AOP "History"**
  - 90's: Research (MDSoC-Hyper/J…), pre-1.0 AspectJ
  - Early 00's: Aspect*X*, AspectWerkz, Spring, JAC, JBoss AOP, …
  - ???: Mainstream breakout
    - If GUI's drove OOP adoption, then what will drive AOP's adoption?
    - *Distributed enterprise applications have lots of cross-cutting concerns*
      - They need AOP modularity today!

The Future of Aspect Oriented Programming

- **Modularity**

  OOP went mainstream when its modularity capabilities became essential (*e.g.,* GUI apps.)

  AOP will go mainstream when its modularity capabilities become essential (*e.g.,* distributed enterprise apps.?)

  ***but** only when tools and processes are ready!*

The Future of Aspect Oriented Programming

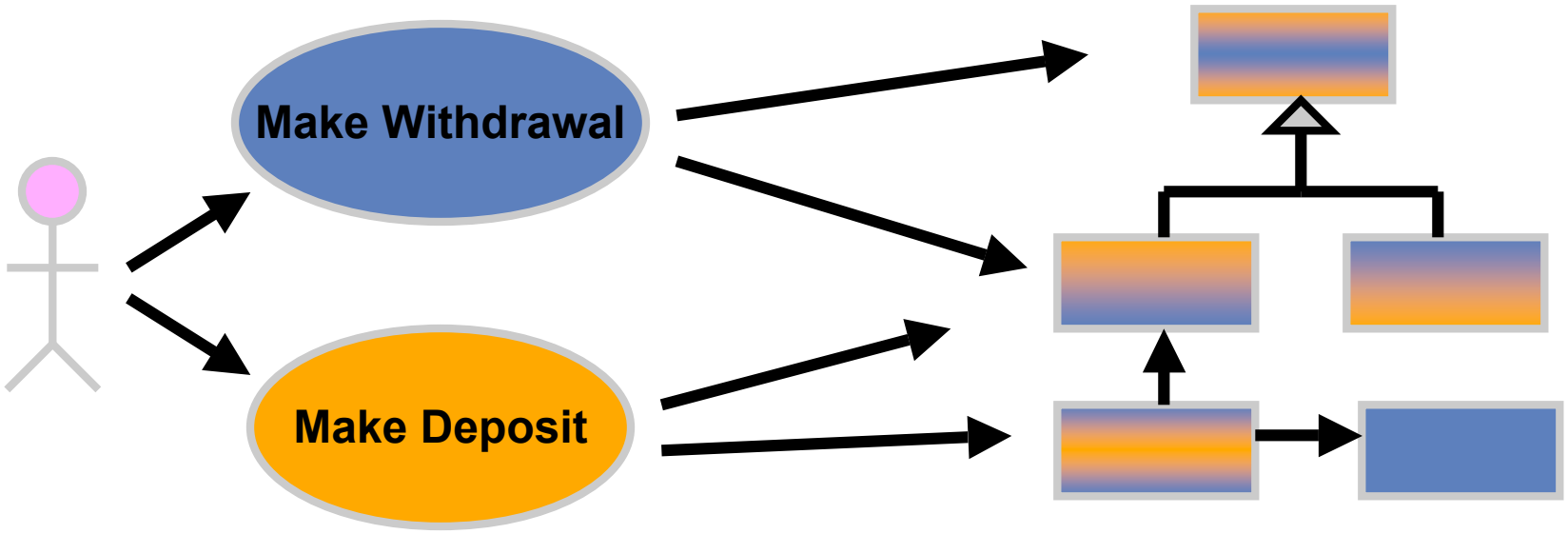
- What Is AOP?
- The Future of AOP
  - **What Will Software Look Like?**
- AOP Today

What Will Software Look Like in Five (?) Years?

- How will software be developed and structured in the future?
  - … assuming that AOP succeeds
- Five Year Horizon?
  - Aggressive compared to OOP's history
    - But things move faster these days!

In Five (?) Years: AOP-Based Applications

- **Are Use Cases Aspects?**
  - **Ivar Jacobson's view:**
    - **Develop use cases separately and weave them to compose applications**

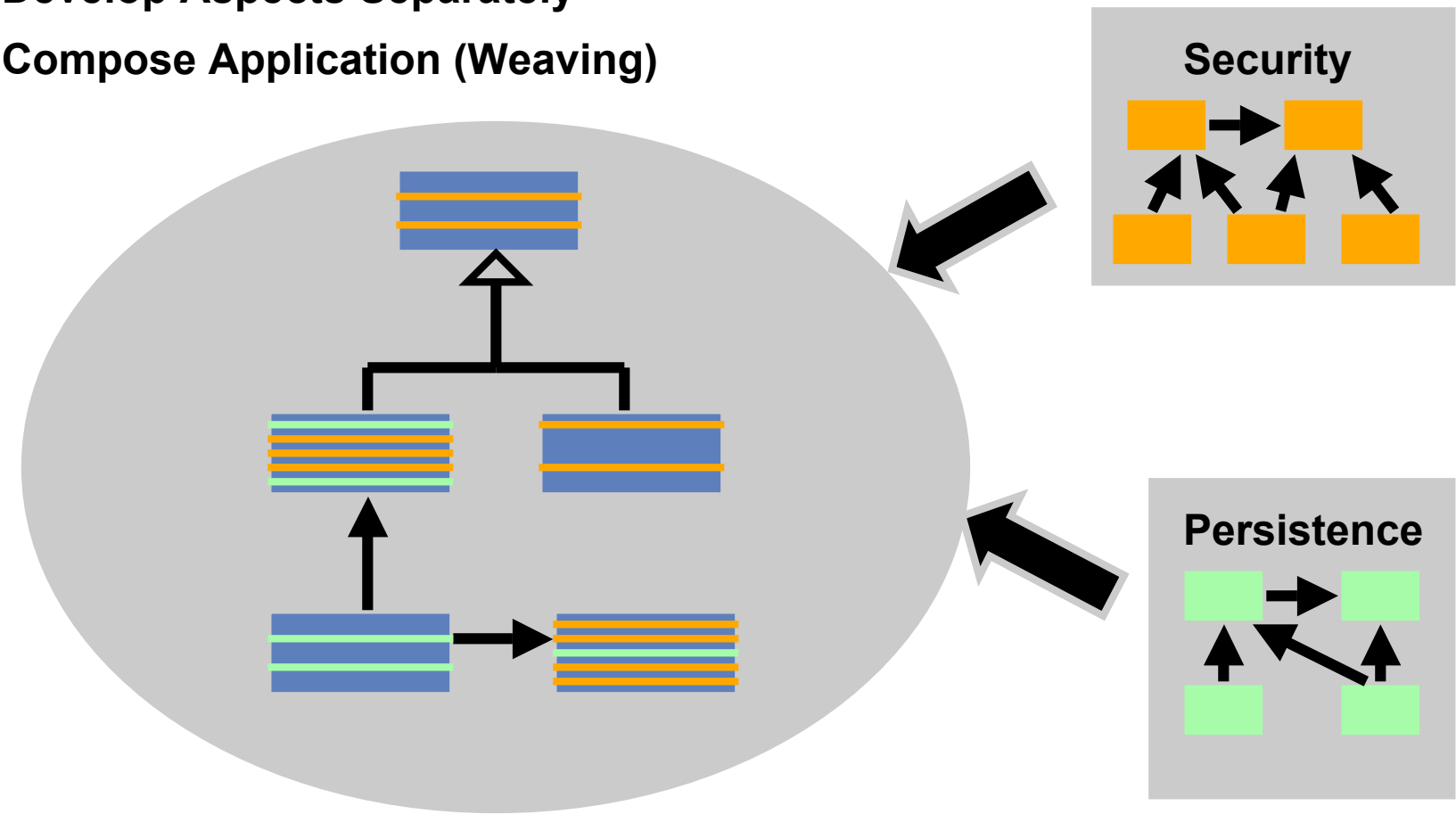


Recall My Earlier Slide:

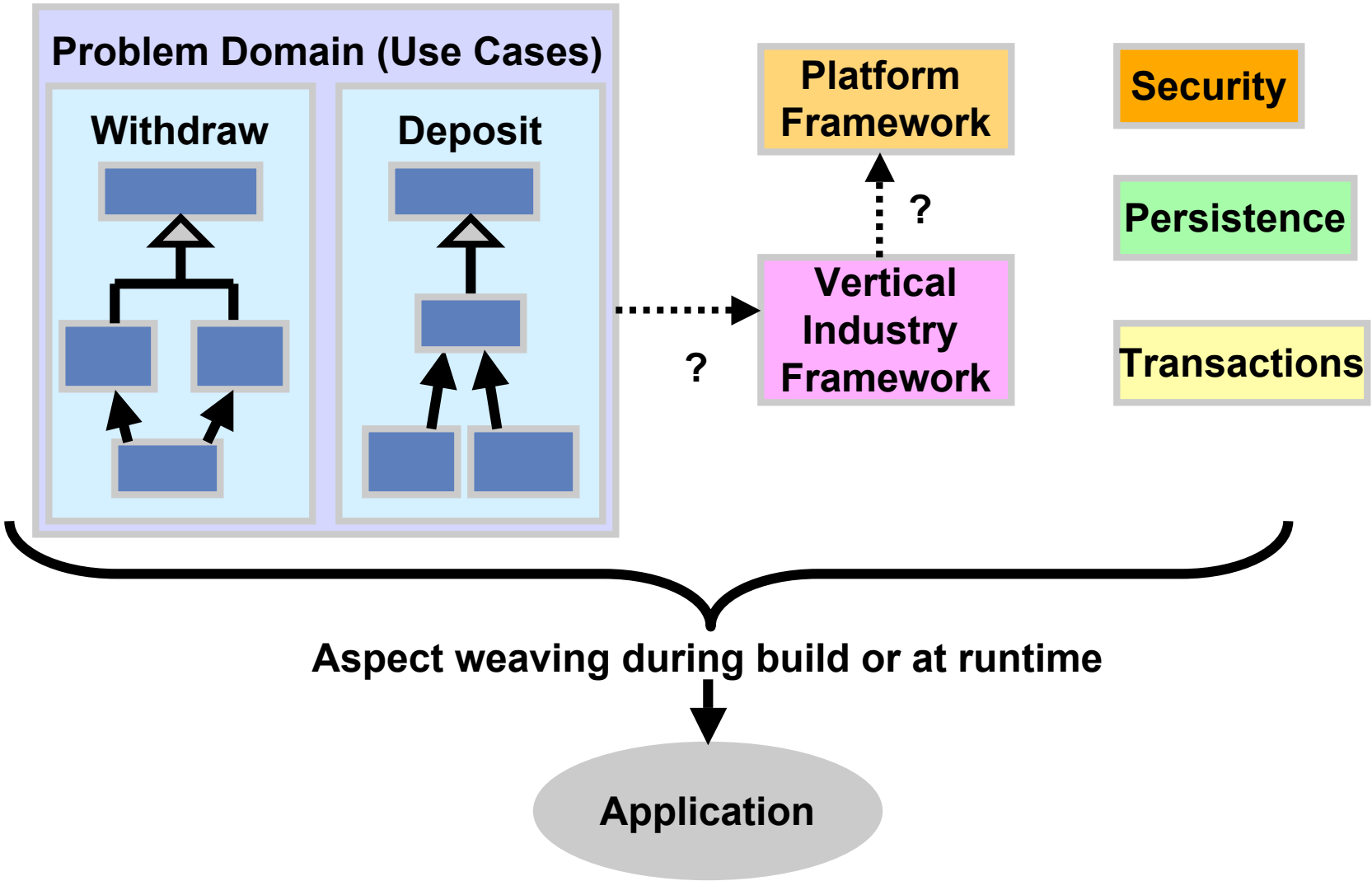
#1 Develop Dominant Decomposition (Problem Domain, Framework)

#2 Develop Aspects Separately

#3 Compose Application (Weaving)



In Five Years: AOP-Based Applications



The Future of Aspect Oriented Programming

- What Is AOP? My Perspective
- The Future of AOP
  - What Will Software Look Like?
- **AOP Today**

AOP Today

- **Current Technology Assessment**
  - AOP still immature
    - Aspect languages need development
      - » AspectJ is like Smalltalk
    - Theory of AOP needs development
      - » E.g., modeling hard with poor foundations
    - Industry experience needed
    - AOP relies on tools more than OOP
      - » E.g., weaving!

AOP Today - Recommendations

- **Learn About AOP**
- **Adopt AOP *Gradually***
  - Apply AOP concepts to your designs
  - Use AOP tools for development tasks
  - Gradually introduce Aspects into Production Code
  - See
    - Kiczales, SD Magazine, Nov. 2003
    - Laddad, “AspectJ in Action”, Manning, 2003

Thank You

Dean Wampler
dean@aspectprogramming.com

Come to AOSD 2004!
http://www.aosd.net/conference