AOP

Aspect-Oriented Programming

What is AOP?

- Procedural Programming

  Abstract program into procedures.
- Object-Oriented Programming

  Abstract program into objects.
- Aspect-Oriented Programming

  Abstract program into aspects.

What is Aspect?

- Aspects are stand-alone modules that allow the programmer to express crosscutting concerns in.
- Crosscutting concerns are concerns of a program which affect (crosscut) other concerns.
- A concern is any piece of interest or focus in a program.
- separation of concerns (SoC) is the process of breaking a computer program into distinct features that overlap in functionality as little as possible.

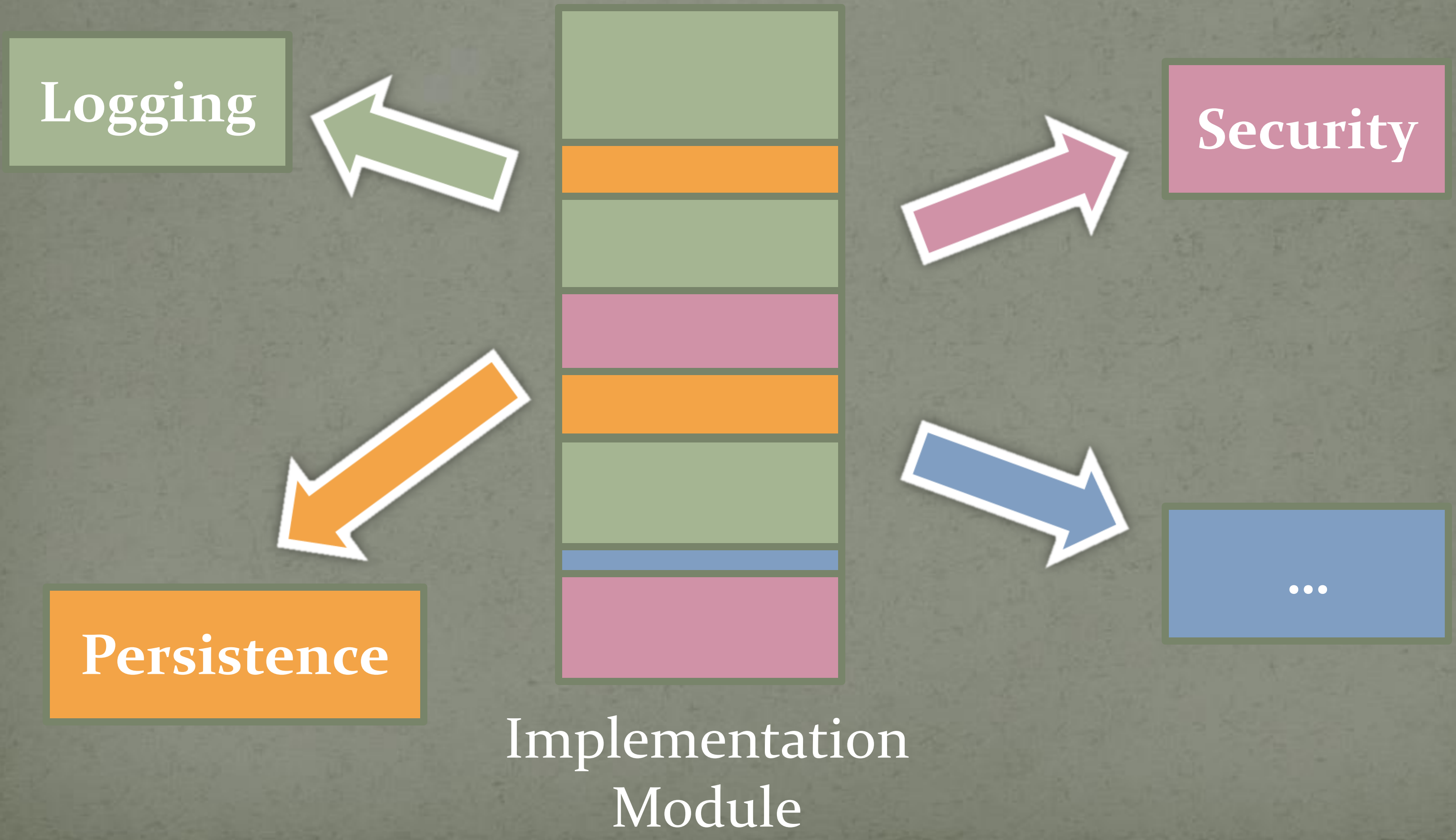
What is Crosscutting Concerns?

```
void transfer(Account from, Account to, int amount) {
  if (!getCurrentUser().canPerform(OP_TRANSFER)) {
    throw new SecurityException();
  }
  if (amount < 0) { throw new NegativeTransferException(); }
  Transaction tx = database.newTransaction();
  try {
     if (from.getBalance() < amount) {
       throw new InsufficientFundsException();
     }
     from.withdraw(amount);  to.deposit(amount);
     tx.commit();
     systemLog.logOperation(OP_TRANSFER, from, to, amount);
  } catch(Exception e) {
     tx.rollback();
     throw e;
  }
}
```

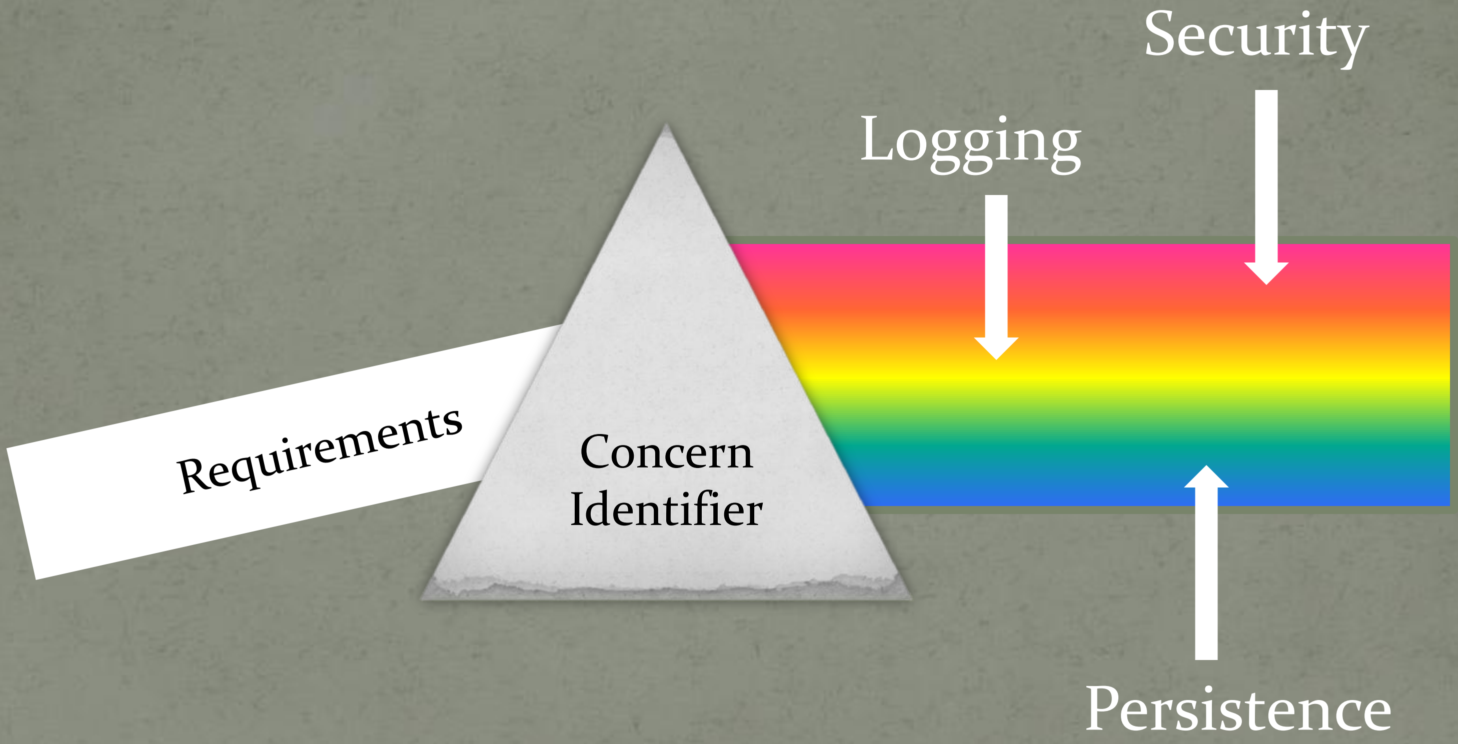
What is Crosscutting Concerns?

```
void transfer(Account from, Account to, int amount) {
  if (!getCurrentUser().canPerform(OP_TRANSFER)) {
    throw new SecurityException();
  }
  if (amount < 0) { throw new NegativeTransferException(); }
  Transaction tx = database.newTransaction();
  try {
     if (from.getBalance() < amount) {
       throw new InsufficientFundsException();
     }
     from.withdraw(amount);  to.deposit(amount);
     tx.commit();
     systemLog.logOperation(OP_TRANSFER, from, to, amount);
  } catch(Exception e) {
     tx.rollback();
     throw e;
  }
}
```

CC not properly encapsulated



Abstract Concerns into Aspects



AspectJ

- A seamless aspect-oriented extension to the Java programming language.
- The widely-used de-facto standard for AOP.
- Born at Xerox Palo Alto Research Center (PARC) and later available in Eclipse Foundation open-source projects.
- Java-like syntax and IDE integration.
- Easy to learn and use.

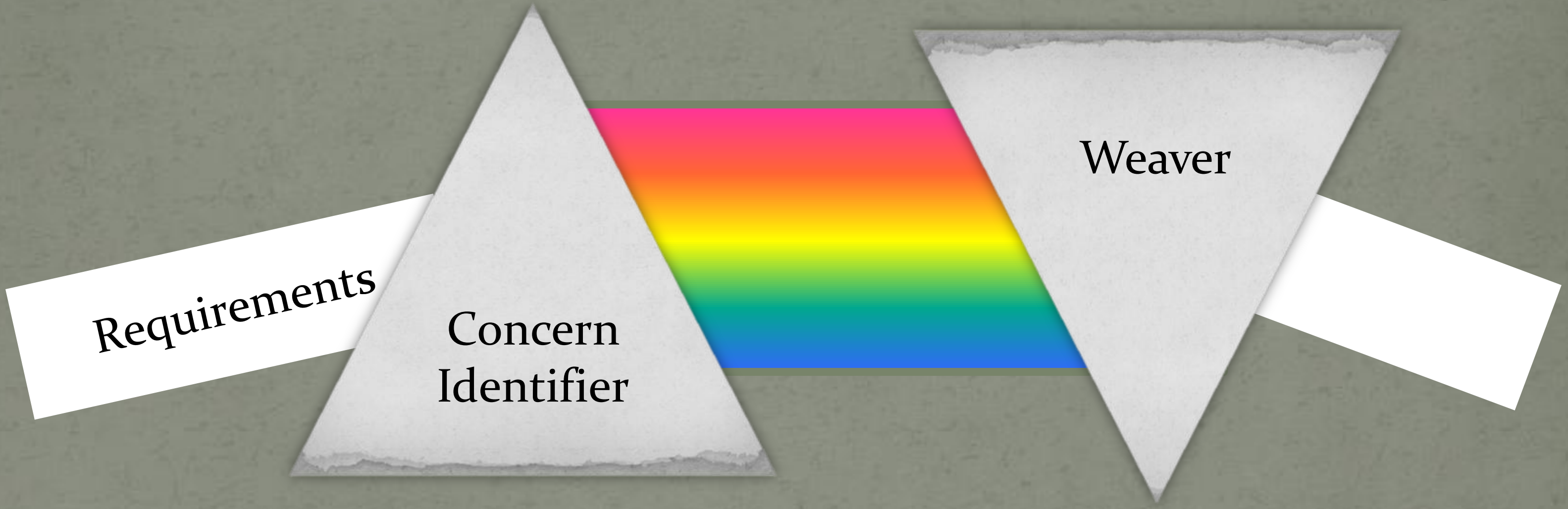
Join Point Model

- **Join Points** -- Points in a running program where additional behavior can be usefully joined.
- **Pointcuts** -- Ways to specify (or *quantify*) join points. A pointcut determine whether a given join point matches.
- **Advice** -- A means of specifying code to run at a join point.

A Simple Logging Example

```
public aspect TraceAspect {
  private Logger _logger = Logger.getLogger("trace");
  pointcut traceMethods()
    : execution(* *.*(..)) && !within(TraceAspect);
  before() : traceMethods() {
    Signature sig =
        thisJoinPointStaticPart.getSignature();
    _logger.logp(Level.INFO,
                 sig.getDeclaringType().getName(),
                 sig.getName(), "Entering");
  }
}
```

Weaving



Related Concept: Visitor Pattern

```
class Wheel implements Visitable {
    // ...
    public void accept(Visitor visitor) {
        visitor.visit(this);
    }
}
```

Related Concept: Mixin

```
class GPA
  include Comparable

  def <=>(another)
    # ...
  end
end

a, b = cat.gpa, dog.gpa
cat.bg if cat.gpa > dog.gpa
```

Related Concept: Policy-based Design

```
template
<
  typename T,
  template <class> class OwnershipPolicy =
                     RefCounted,
  class ConversionPolicy =
                     DisallowConversion,
  template <class> class CheckingPolicy =
                     AssertCheck,
  template <class> class StoragePolicy =
                     DefaultSPStorage
>
class SmartPtr;
```

Example: cc98 CFF 1st

```
# equip the spider with cache
def equip(spider)
  orig_get_page = spider.class.instance_method :get_page
  dir = @dir

  spider.define_singleton_method :get_page do |url|
    filename = dir + url
    if File.exists? filename
      File.open(filename) { |file| file.read }
    else
      page = orig_get_page.bind(self).call(url)
      File.open(filename, "w") { |file| file.write page }
      page
    end
  end
end
```

Example: Rails filter

```
before_filter :determine_locale
before_filter :configure_charsets
before_filter :load_personal_preferences
before_filter :login_required,
              :except => [:index, :login]
before_init_gettext :default_locale
```

Adoption Risks

- Lack of tool support, and widespread education.
- Unfriendly to refactor: something as simple as renaming a function can lead to an aspect no longer being applied leading to negative side effects.
- Security concern: injecting bad code.
- …