An Aspect-Oriented Implementation Method

Sérgio Soares and Paulo Borba
Software Productivity Group
Informatics Center - Federal University of Pernambuco - Brazil
{scbs,phmb}@cin.ufpe.br
http://www.cin.ufpe.br/spg

Characteristics:
- Implementation activities
- Modifications in analysis, design, and tests activities
- Patterns for structuring the system architecture
- Aspects frameworks to implement crosscutting concerns:
  - Persistence, distribution, and concurrency control
  - Integration with Use Case Driven Development

Benefits:
- Guarantees the effort given to requirements and design
- Guidelines to aspect-oriented development
- Guidelines to aspect-oriented refactoring
- Two implementation approaches
  - Experimentation for analysis
  - Tool support
    - Aspect and classes generation
    - Increased productivity

Drawbacks:
- Interferences between aspects
- Tailored to a specific software architecture
- Allows implementing several kinds of systems
- Used in several real systems
- More precise guidelines

Software architecture

Persistence aspects framework

<<Aspect>> TransactionControl
<<Aspect>> DataCollectionCustomization
<<Aspect>> PersistenceControl

Concurrency control aspects framework

<<Aspect>> Synchronization
<<Aspect>> PessimisticSynchronization
<<Aspect>> OptimisticSynchronization

Aspects frameworks to implement crosscutting concerns:
- Persistence
- Distribution
- Concurrency control

Integration with Use Case Driven Development

The implementation method is integrated to Use-case Driven Development in two different approaches

The implementation method can be used to refactor an object-oriented application to an aspect-oriented one

Software architecture

 iPersistenceMechanism
  connect()
  disconnect()
  beginTransaction()
  commitTransaction()
  rollbackTransaction()
  getCommunicationChannel()

PersistenceMechanism
  BusinessCollection
  BusinessDataInterface
  PersistentDataCollection
  NonPersistentDataCollection
  BasicClass

PersistenceMechanism

Remote
(from java.rmi)

<<Aspect>> ClientSide
<<Aspect>> ServerSide

Distribution aspects framework

Progressive approach

Non-violative approach

The implementation method is integrated to Use-case Driven Development in two different approaches

Milestone (end of iteration)

a and b are use-cases or sets of use-cases