



# Using Code Metrics for Targeted Code Refactoring

# Today's Discussion

- Why do we refactor?
- How do we know when to refactor?
- Refactoring techniques
- Refactoring in the real world

# Why do we refactor code?

- What is the *goal* of refactoring?
  - » Reduce Complexity

# Why do we refactor code?

- Why Reduce Complexity?

*Reducing  
Complexity*

Increases  
testability  
Decreases  
maintenance

# How do we traditionally know *when* to refactor?

- “Code Smell”
  - » feeling = subjective

# How do we traditionally know *when* to refactor?

- Example Code Smells
  - » Comments
  - » Long Methods
  - » Long Classes

# How can we approach refactoring *objectively*?

- Code Metrics
  - » What are they?
  - » Why aren't they used often?

# Code Metrics

- Cyclomatic Complexity
  - » Distinct paths
    - conditionals, loops

# Code Metrics: Cyclomatic Complexity

- How can one determine CC?
  - » PMD
  - » JavaNCSS
  - » Eclipse Metrics plug-in

# Code Metrics: Cyclomatic Complexity

- Rules of Thumb:
  - » methods  $> 10$  = complex
  - » excellent coverage:
    - 1:1 ratio of test cases to cc

# Code Metrics: Cyclomatic Complexity

- False Positives:
  - » “update” logic
  - » ignore aggregates

# Code Metrics

- Depth of Inheritance
  - » Hierarchy tree
    - extends clause only

# Code Metrics: Depth of Inheritance

- How can one determine DIT?
  - » Eclipse Metrics plug-in
  - » Adana Maven plug-in

# Code Metrics: Depth of Inheritance

- Rule of Thumb
  - » excessive depths:
    - increase difficulty in testing
    - decrease comprehensibility

# Code Metrics: Depth of Inheritance

- False Positives
  - » Exception Hierarchies &

JUnit tests usually have high

DIT

# Code Metrics

- Non Commenting LOC
  - » Method Length
  - » Class Length

# Code Metrics: Method Length

- Method Length
  - » 100+ NLOC
    - too much work
    - increased complexity

# Code Metrics: Class Length

- Class Length
  - » 1000+ NCLCOC
  - too much responsibility
  - complex

# Code Metrics: Class Length

- Public Method Count
  - » Many = lots of responsibility
    - difficult to test

# Code Metrics: Class Length

- Unique Attributes

- » 45+ unique types (think imports)
- Members, parameters, variables
  - brittle

# Code Metrics: Lines of Code

- How to determine high LOC?
  - » PMD
  - » JavaNCSS
  - » Eclipse Metrics plug-in

# Code Metrics: Lines of Code

- Rules of Thumb
  - » High LOCs
    - complexity
    - difficulty in testing
    - brittleness

# Code Metrics: Lines of Code

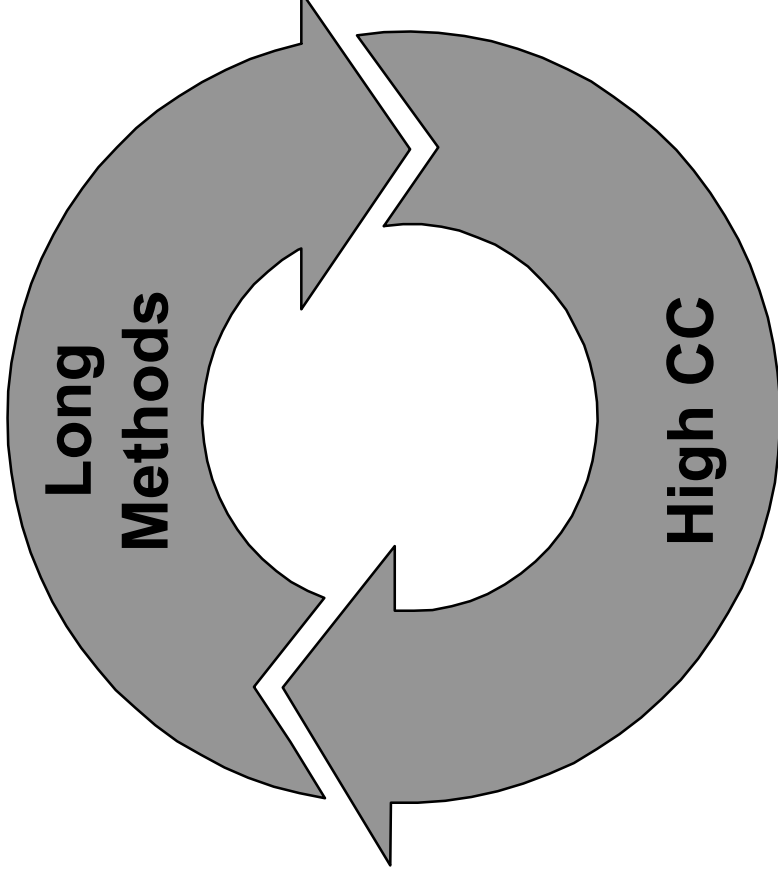
- **False Positives**
  - » JavaBeans, Transfer Objects, Entity Beans, POJOs, etc
    - many public methods
      - LOC is high

# Code Metrics

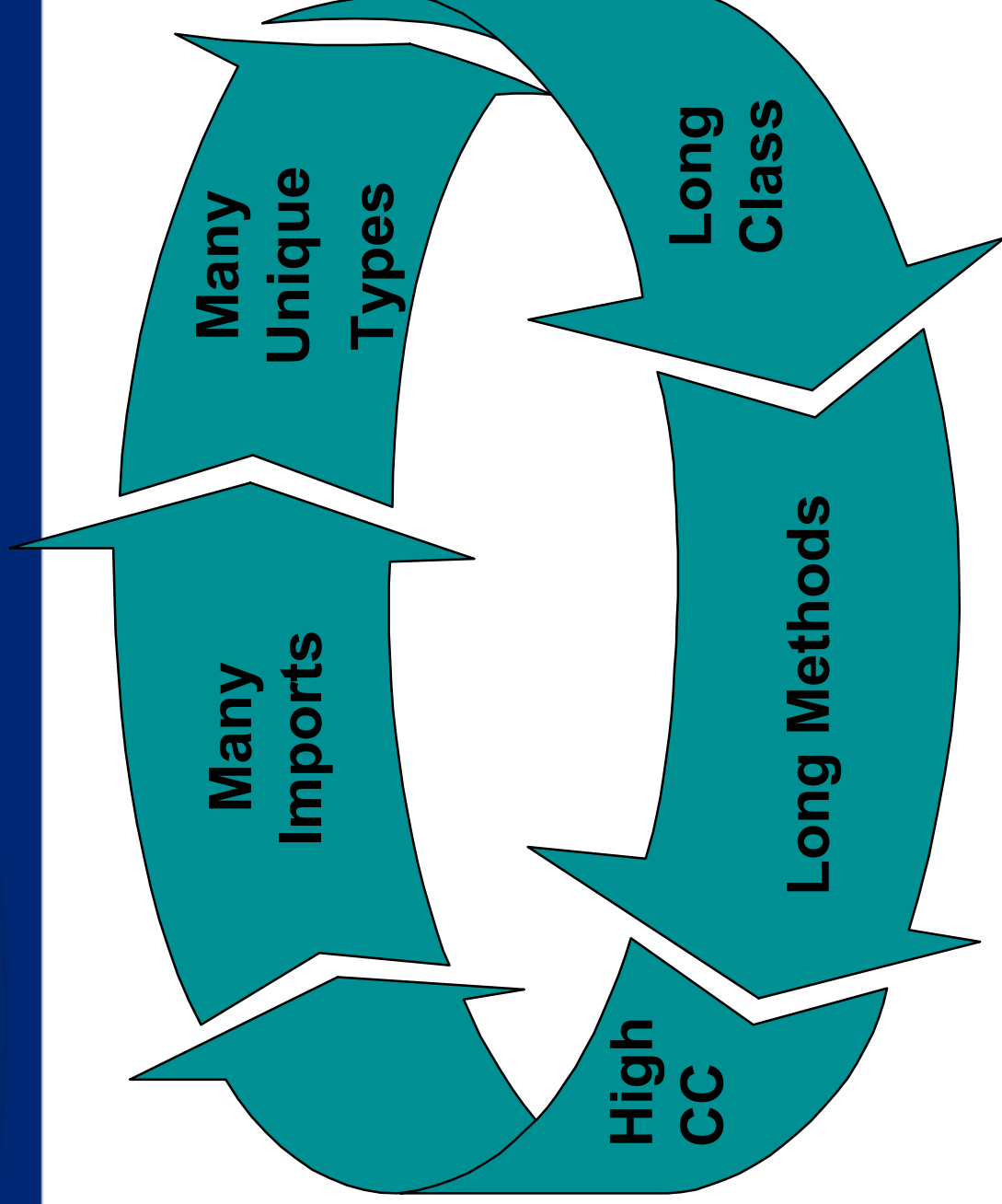
- Rules of Thumb
  - » Code metrics are objective
    - Application subjective
  - » High LOC != Productivity

# Code Metrics

- Common Correlations



# Code Metrics



# Refactoring Techniques

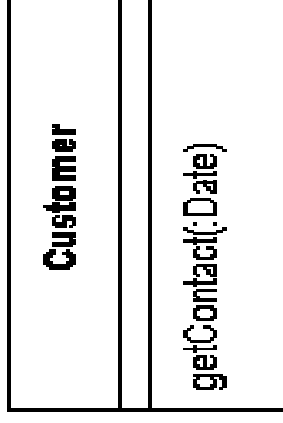
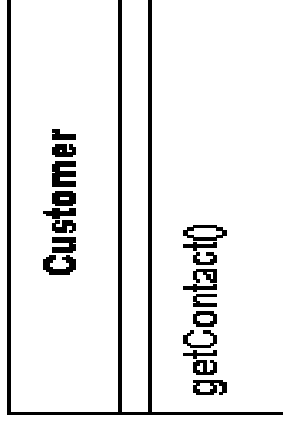
- **Books**
  - Refactoring: Improving the Design of Existing Code
  - Refactoring to Patterns
  - Working Effectively with Legacy Code
- **Website**
  - <http://www.refactoring.com/catalog/index.html>

# Refactoring Techniques

- Long Methods
  - » *Extract Method*
  - » *Substitute Algorithm*
  - » *Add Parameter*

# Refactoring Techniques

- Add Parameter

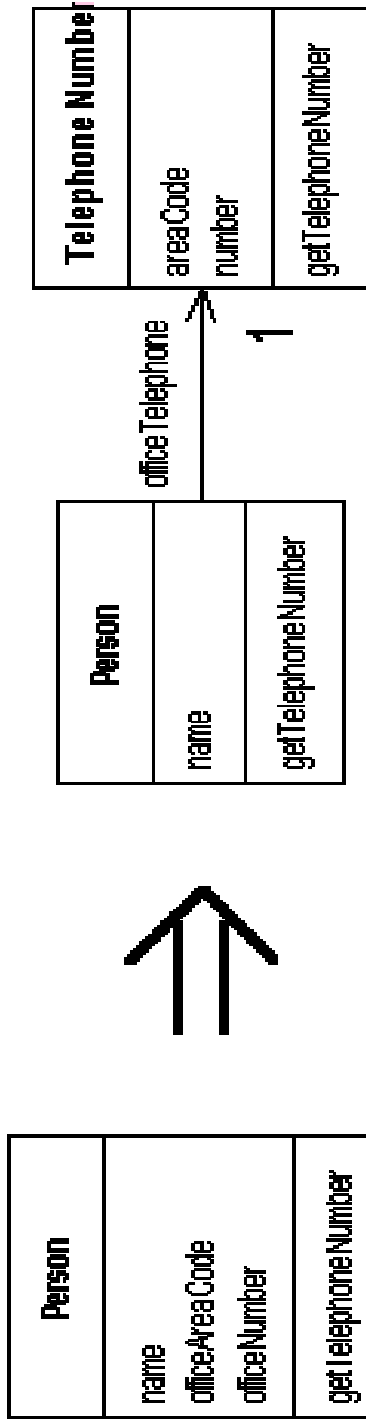


# Refactoring Techniques

- Long Classes
  - » *Extract Class*
  - » *Extract Subclass*
  - *Push Down Method*
  - *Form Template Pattern*

# Refactoring Techniques

## ■ Extract Class

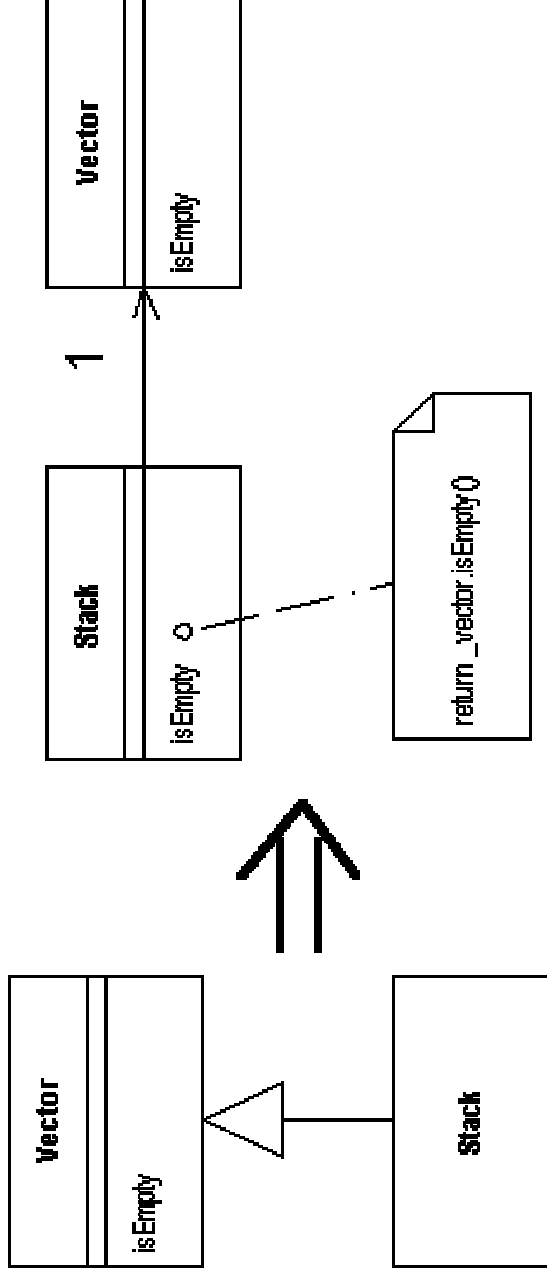


# Refactoring Techniques

- Depth of Inheritance
  - » *Replace Inheritance with*  
*Delegation*
  - » *Pull up and Push down Method*

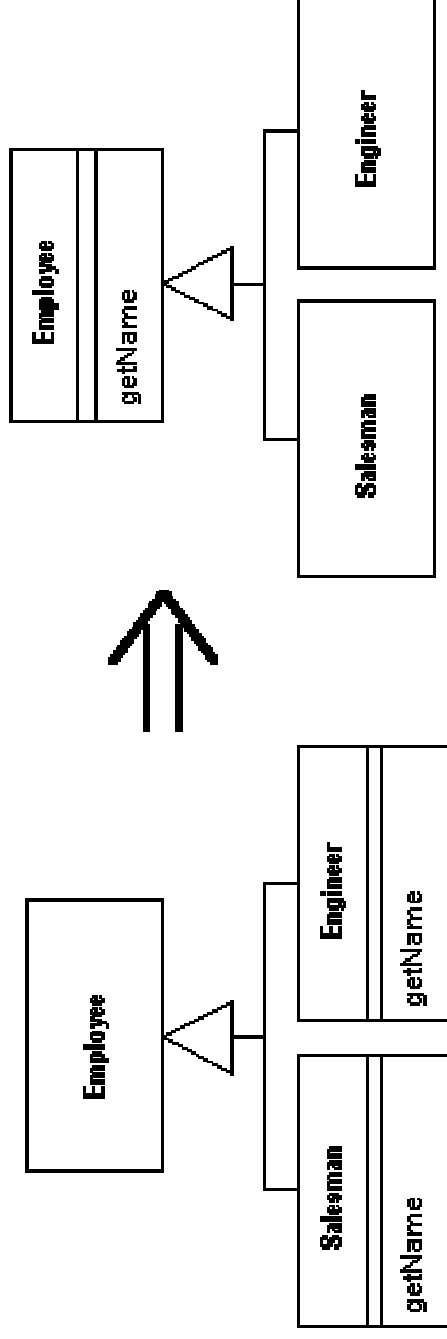
# Refactoring Techniques

## ■ *Replace Inheritance with Delegation*



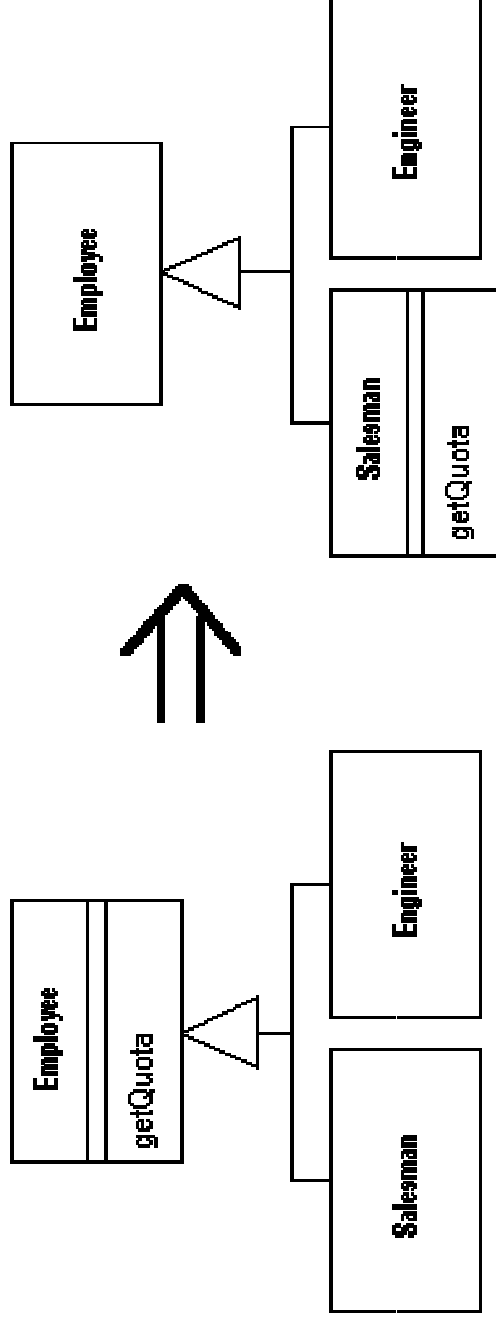
# Refactoring Techniques

## ■ Pull up Method



# Refactoring Techniques

## ■ Push Down Method



# Refactoring Techniques

- Too Many Imports
  - » *Extract Class*

# Refactoring Techniques

- Cyclomatic Complexity
  - » *Replace Conditional with Polymorphism*
  - » *Extract Method*

# Refactoring for Real

- Find Complexity
  - » *Run PMD*
    - visual confirms added
  - maintenance issue

# Refactoring for Real

- Conditionals
  - » Keep growing
  - » Affect testability

# Refactoring for Real

- Solution
  - » *Replace Conditionals with Polymorphism*
  - » *Extract Class*
  - » Liberal interpretation of GOF patterns

# Refactoring for Real

- GOF Patterns
  - » *Abstract Factory*
  - » *Chain of Responsibility*
  - » *Strategy*

# Refactoring for Real

- Benefits?
  - » Plug-ability
  - » Testability
    - pushed complexity into manageable pieces

# Review of Today's Discussion

- Why do we refactor?
- How do we know when to refactor?
- Refactoring techniques
- Refactoring in the real world

# Using Metrics For Targeted Refactoring

**Thank you!**

Questions? Comments?

Please Contact:

Andrew Glover

Telephone (866) 682-9259

or

[aglover@vanwardtechnologies.com](mailto:aglover@vanwardtechnologies.com)