CS 683 Emerging Technologies: Embracing Change
Spring Semester, 2001
Doc 15 Code Smells
Contents

References

Refactoring: Improving the Design of Existing Code, Fowler, Addison-Wesley, 1999

More metrics (Was: Woohoo! 1000 tests!), Jim Little, XP Mailing list, 22 Mar 2001

Code Smells

If it stinks, change it

-- Grandma Beck on child-rearing

Some Smells

Duplicate Code	Long Method
Large Class	Long Parameter List
Divergent Change	Shotgun Surgery
Feature Envy	Data Clumps
Primitive Obsession	Switch Statements
Parallel Inheritance Hierarchies	Lazy Class
Speculative Generality	Temporary Field
Message Chains	Middle Man
Inappropriate Intimacy	Alternative Classes with Different Interfaces
Incomplete Library Class	Data Class
Refused Bequest	Comments

Duplicate Code[1]

Same code in two methods in same class

Try Extract method

Same code in two methods in two sibling subclass

Try Extract Method and Pull Up Field

If code is similar try:

Extract Method(110) to separate similar bits

Form Template Method(345) may help reform method

If two methods use different algorithms to do the same thing try Substitute Algorithm(139)

Same code in two unrelated classes

Try Extract Class(149)

Consider moving the code to one of the existing classes

[1] Fowler, pp. 76

Form Template Method[2]

You have two methods in subclasses that perform similar steps in the same order, yet the steps are different

Get the steps into methods with the same signature, so that the original methods become the same. Then you can pull them up

Example

Note "_" is used to indicate an instance variable

Site subclass: #ResidentialSite

Site subclass: #LifelineSite

```
ResidentialSite>>billableAmount
   | base tax |
   base := _units * _rate.
   tax := base * TaxRate.
   ^base + tax

LifelineSite>>billableAmount
   | base tax |
   base := _units * _rate * 0.7.
   tax := base * TaxRate * 0.2.
   ^base + tax
```

[2] Fowler, pp. 345-351

With Template Method[3]

```
Site>>billableAmount
  ^self baseAmount + self taxAmount

Site>>taxAmount
  ^self subclassResponsibility

Site>>baseAmount
  ^self subclassResponsibility


ResidentialSite>>taxAmount
  ^self baseAmount * TaxRate

ResidentialSite>>baseAmount
  ^_units * _rate


LifelineSite>>taxAmount
  ^self baseAmount * TaxRate * 0.2

LifelineSite>>baseAmount
  ^_units * _rate * 0.7
```

[3] Can you find a better Template Method?

Extract Class[4]

You have one class doing the work that should be done by two

Create a new class and move the relevant fields and methods from the old class into the new class

Substitute Algorithm[5]

You want to replace an algorithm with one that is clearer

Replace the body of the method with the new algorithm

[4] Fowler, pp. 149-153
[5] Fowler, pp. 139-140

Long Method[6]

Metrics from XP Java Project[7]

	Production code	Test code
Classes	288	273
Methods	2158	2107
Statements	6201	9080
Methods / Class	7.4	7.7
Statements / Method	2.9	4.3

Try Extract Method(110) to shorten a method

Comments often indicate code to extract from a method

Conditionals and loops provide code to extract
Decompose Conditional(238)

Too many temporary variables and/or parameters may prohibit using Extract Method

If have too many temporary variables try:
Replace Temp with Query(120)

If have long parameter list try:
Introduce Parameter Object(295)
Preserve Whole Object(288)

[6] Fowler, .pp. 76-77
[7] Jim Little

Replace Temp with Query[8]

You are using a temporary variable to hold the result of an expression

Extract the expression into a method. Replace all references to the temp with the expression

Example

```
cost
   | basePrice |
   basePrice := _quantity * _itemPrice.
   basePrice > 1000
      ifTrue:[^basePrice * 0.95]
      ifFalse:[^basePrice * 0.98]
```

Replace with

```
cost
   self basePrice > 1000
      ifTrue:[^self basePrice * 0.95]
      ifFalse:[^self basePrice * 0.98]

basePrice
   ^_quantity * _itemPrice.
```

[8] Fowler, pp. 120-123

Introduce Parameter Object[9]

You have a group of parameters that naturally go together

Replace them with an object

Example

Customer>>amountInvoicedfrom: aDate to: anotherDate

Replace with

Customer>>amountInvoicedIn: aDateRange

[9] Fowler, pp. 295-299

Preserve Whole Object[10]

You are getting several values from an object and passing these values as parameters in a method call

Send the whole object instead

Example

```
| low high |
low := range high.
high := range low.
plan validRangeFrom: low to: high.
```

Replace with

```
plan validRange: range.
```

[10] Fowler, pp. 288-291

Comments

Comments are a good smell, but:

"Comments are often used as a deodorant"[11]

When you feel the need for a comment:

Try refactoring to make the comment unneeded

Have comment explaining a block of code:

Try Extract Method(110)

If you need a comment to explain what a method does:

Try Rename Method(273)

If you to comment on required state of the system/parameters

Try Introduce Assertion(267)

[11] Fowler pp. 87

Rename Method[12]

The name of a method does not reveal its purpose

Change the name of the method.

Example

cstrcdlmt

Replace with

customerCreditLimit

[12] Fowler pp. 273-274

Long Parameter List[13]

Long parameter lists are:
- Hard to understand
- Difficult to use
- Error prone
- Change often

Reduce parameter list by trying:

- Replace Parameter with Method(292)
- Preserve Whole Object(288)
- Introduce Parameter Object(295)

Beware of coupling when reducing parameter lists

[13] Fowler pp. 78-79

Replace Parameter with Method[14]

An object invokes a method, then passes the result as a parameter for a method. The receiver can also invoke this method.

Remove the parameter and let the receiver invoke this method

Example

```
cost
   | basePrice discount |
   basePrice := _quantity * _itemPrice.
   discountLevel := self discountLevel.
   ^self discountedPriceOn: basePrice discount: discountLevel
```

Replace with:

```
cost
   | basePrice discount |
   basePrice := _quantity * _itemPrice.
   ^self discountedPriceOn: basePrice
```

discountedPriceOn: can call self discountLevel

[14] Fowler pp. 292-294