

Higher-Order Functions

- Replace Simple Hierarchies with higher-order functions



17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

1

Higher-order functions means a function that either accepts a function or returns a function.

If a hierarchy has only one abstract function, derived class has no instance variables and all classes are small, then replace hierarchy by higher-order function.

Q77

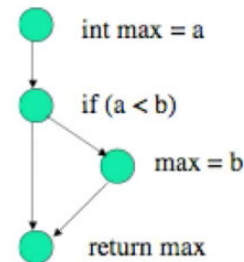
Cyclomatic Complexity

measures complexity of a method/function

$$V(G) = e - n + 2$$

$V(G)$ = cyclomatic complexity of G
 e = # edges
 n = # of nodes

```
int max (int a, int b) {  
    int max = a;  
    if (a < b) {  
        max = b;  
    }  
    return max;  
}
```



17-Feb-23 5:34 PM

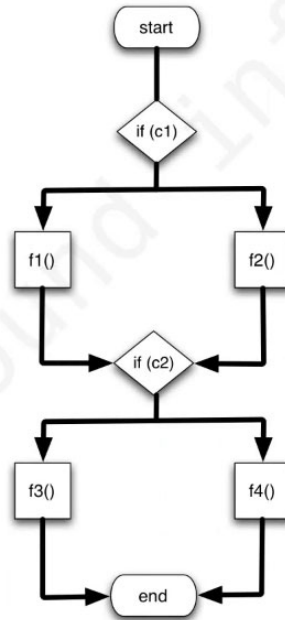
<https://vijaynathani.github.io/>

2

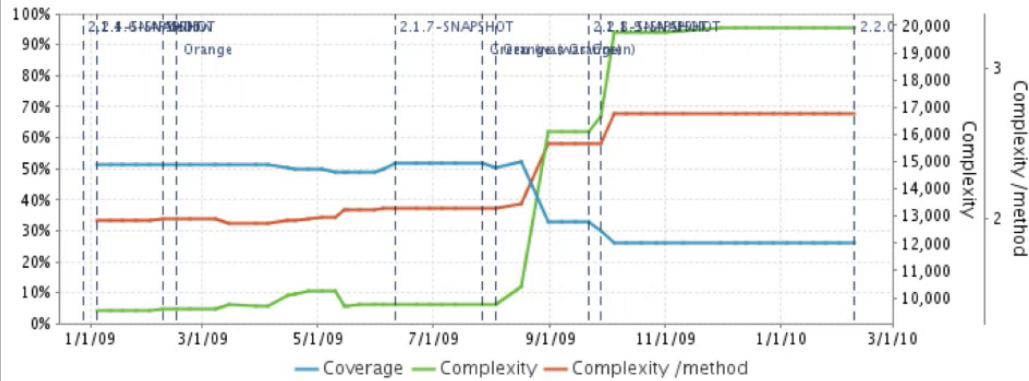
Cyclomatic complexity is a [software metric](#) (measurement). It was developed by [Thomas J. McCabe, Sr.](#) in 1976 and is used to indicate the complexity of a program. It is a quantitative measure of the complexity of programming instructions. It directly measures the number of linearly independent paths through a program's [source code](#).

Cyclomatic Complexity?

```
public void doIt() {  
    if (c1) {  
        f1();  
    } else {  
        f2();  
    }  
    if (c2) {  
        f3();  
    } else {  
        f4();  
    }  
}
```



Struts – Time Machine



17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

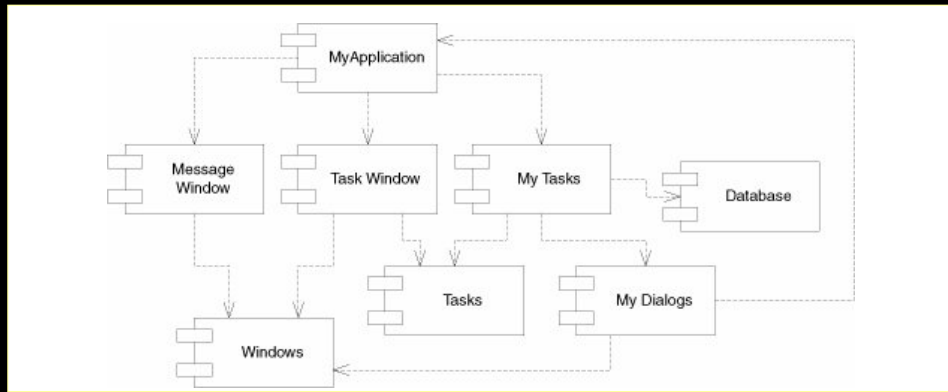
4

Principles

- The Release/Reuse Equivalency Principle
 - The granule of reuse is the granule of release.
 - Either all the classes in a component are reusable, or none of them are.
- The Common Reuse Principle
 - The classes in a component are reused together.
 - If we reuse one of the classes in a component, we reuse them all.
- The Common Closure Principle
 - Classes that change together, belong together in a package.

Principles

- Acyclic Dependencies Principles (ADP)
 - No cycles in the package diagram of a particular layer.
 - No cycles in the component dependency graph



17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

6

Q80reference – zipengine Q81reference - FileCopier Q82reference - faxNo

A component contains one or more packages. If there are no cycles in package diagram, then there will be no cycles in component diagram.

Cycles are introduced in package diagram as an example of code deterioration. So run JDepend tools regularly.

Example of violation

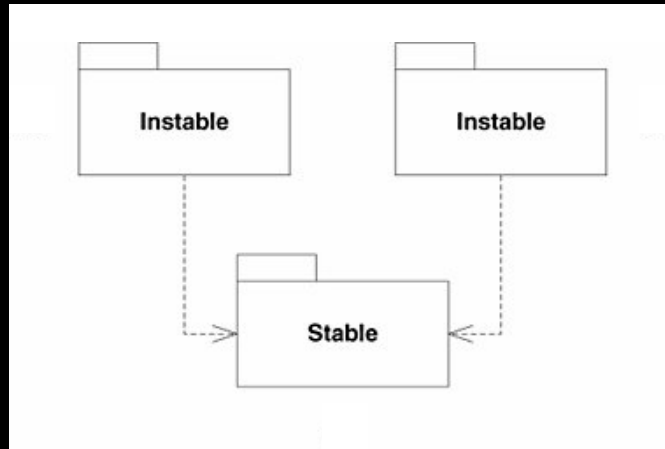
- java.util <-> java.lang : Java.lang.String imports java.util.Locale. java.util.Locale uses String.

- Hibernate

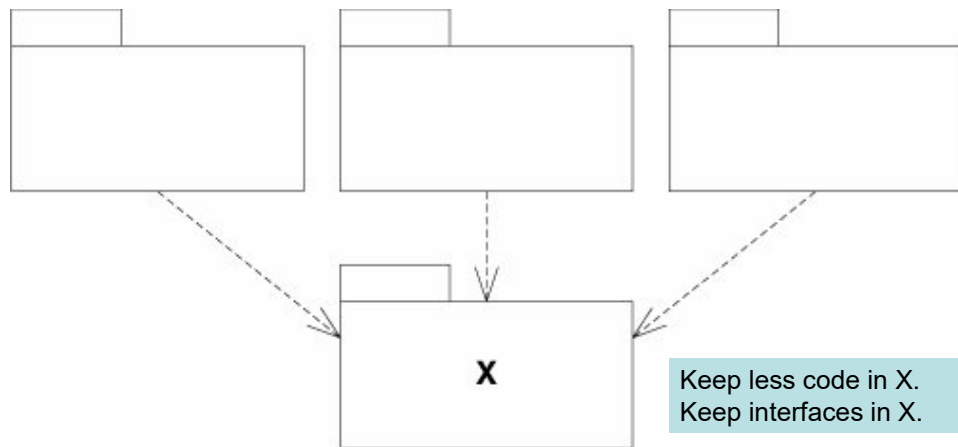
Good example: Spring does not contain a single package circle.

Principles

- Stable Dependencies Principle
 - A package should only depend upon packages that are more stable than it is.



X - a Stable component



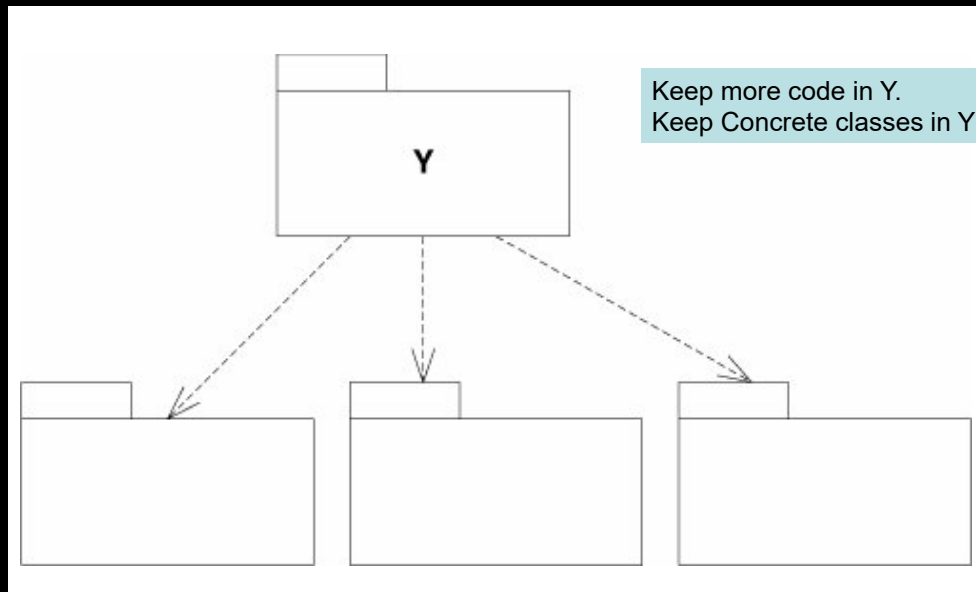
17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

8

X should be really well-designed.

Y - a Unstable component



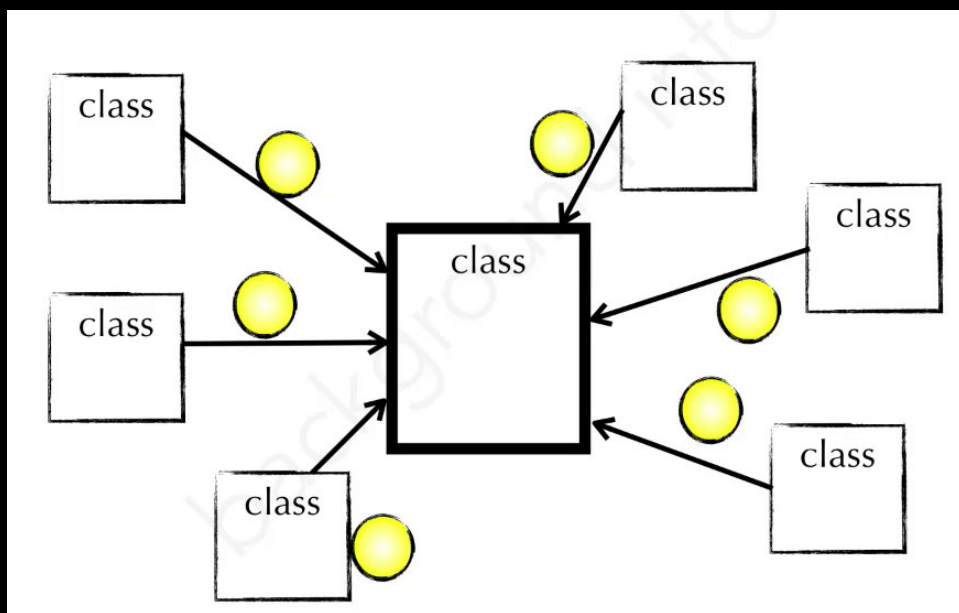
17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

9

If Y depends on many others, then Y is likely a God-class anti-pattern.

Afferent Coupling



17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

10

God Class in center should be refactored.

Struts

classname	WMC	Ca
org.apache.struts2.components.DoubleListUIBean	66	3
org.apache.struts2.views.jsp.ui.AbstractDoubleListTag	66	2
org.apache.struts2.views.xmlt.AbstractAdapterNode	57	4
org.apache.struts2.portlet.util.HttpServletRequestMock	57	1
org.apache.struts2.components.UIBean	53	22
org.apache.struts2.components.Tree	51	3
org.apache.struts2.views.freemarker.tags.StrutsModels	50	1
org.apache.struts2.views.jsp.ui.TreeTag	49	0
org.apache.struts2.components.OptionTransferSelect	44	3
org.apache.struts2.views.xmlt.SimpleAdapterDocument	44	2
org.apache.struts2.views.jsp.ui.OptionTransferSelectTag	43	0
org.apache.struts2.dispatcher.Dispatcher	37	19
org.apache.struts2.views.jsp.ui.AbstractUITag	34	18
org.apache.struts2.components.InputTransferSelect	34	3
org.apache.struts2.components.table.WebTable	33	12
org.apache.struts2.views.jsp.ui.InputTransferSelectTag	33	0
org.apache.struts2.components.Component	28	177
org.apache.struts2.components.AutoComplete	26	3
org.apache.struts2.views.jsp.ui.SubmitTag	25	0
org.apache.struts2.components.Form	24	10
org.apache.struts2.views.xmlt.AbstractAdapterElement	24	6

17-Feb-23 5:34 PM

<https://vijaynathani.github.io/>

11

WMC is weighted method count. WMC for a class is the sum of cyclomatic complexity of all its methods.

For every class in Struts, we have cyclomatic complexity and Afferent coupling.

There may be little value in refactoring a class with high cyclomatic complexity, but low Afferent coupling.

A class with low cyclomatic complexity is likely not badly designed.

We need to concentrate on classes where sum of cyclomatic complexity and Afferent coupling is high.