lecture6

metrics

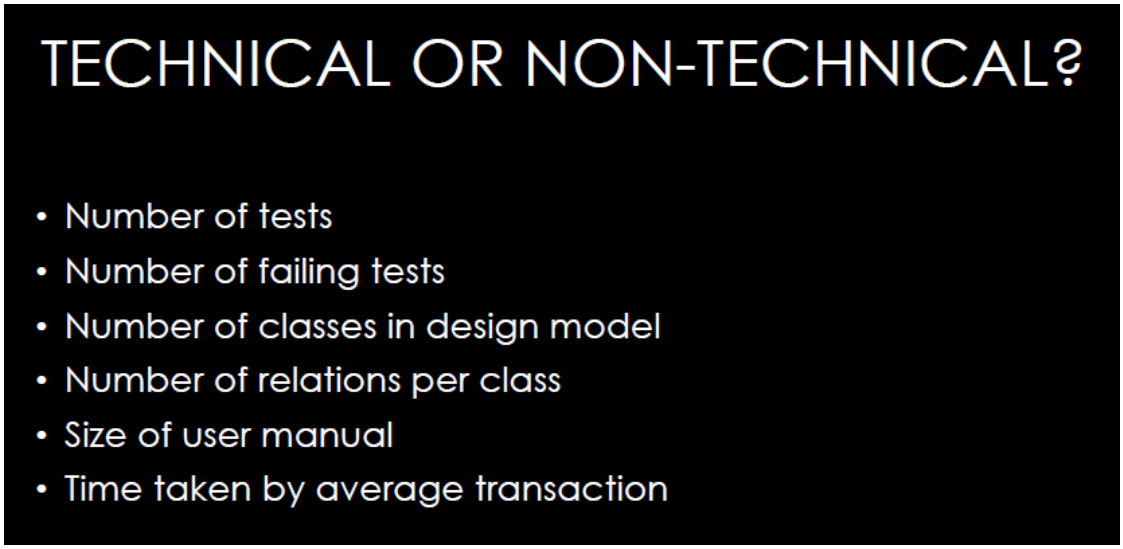
- process
  - man hours
  - bugs reported
  - stories implemented
- product (technical metrics)

non technical metrics

- 项目组成员数量
- 时间/金钱花费
- bugs found/reported
  - by testers/developers
  - by users
- bugs fixed, features added

technical metrics

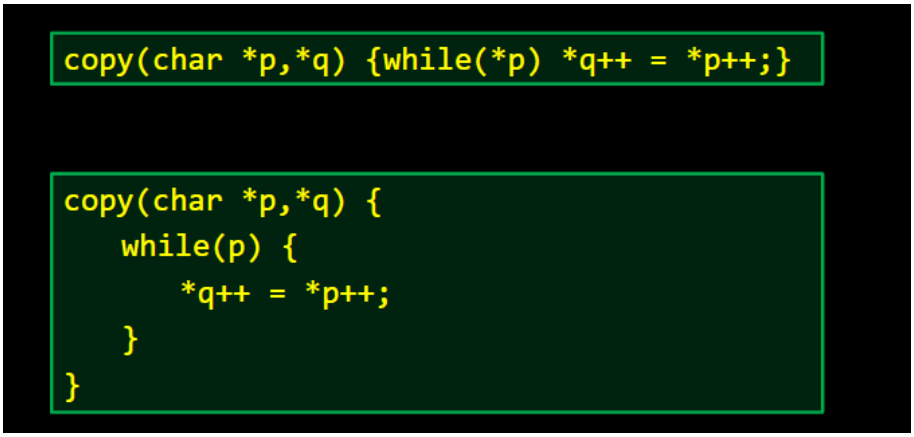
- size of code
  - # of files
  - # of classes
  - # pf processes
- 代码复杂度
  - dependencies coupling cohesion
  - depth of nesting
  - cyclomatic complexity



measuring size of system

- 代码行数 (不是很好的一个标准)
- 类，函数，文件数量
- function Points

代码行数



这两种是等价的（就代码行数来说）

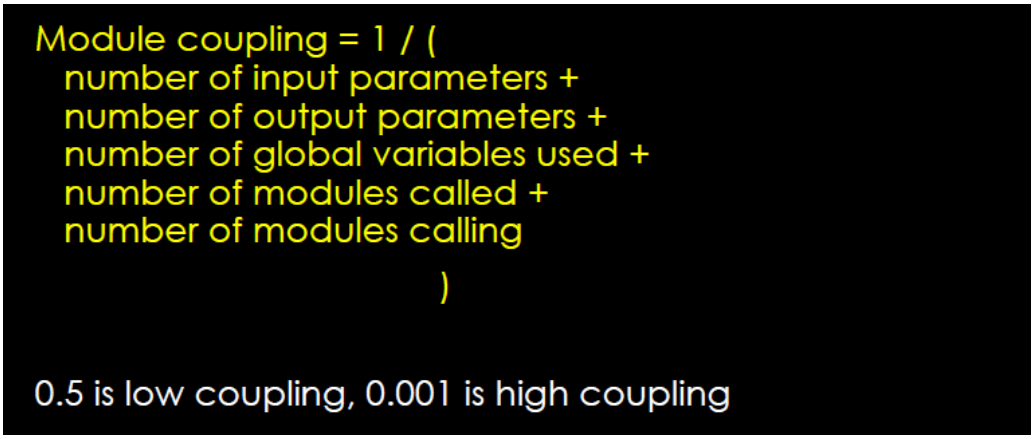
有效性

- 同一种语言
- 标准格式
- code has been reviewed

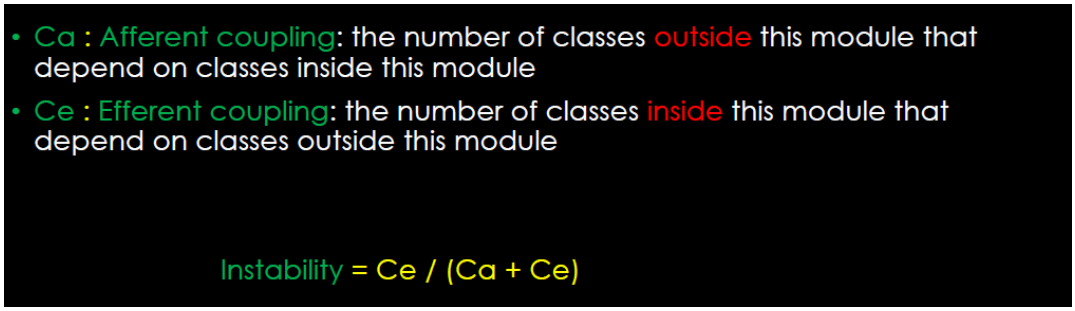
complexity measurement

- cyclomtic complexity
  - Testing view
    - **the number of independent path through the procedure**
    - gives an upper bound on the number of tests
  - metrics view
    - > 10 的很难做测试，并且容易出错
  - $= \#of\ braches\{if,while,for\} + 1$. (# of predicates + 1)
  - number of edges - number of nodes + 2
  - number of regions of the flow graph

- function points
  - measure "functionality" of system
  - measure of how bug a system ought to be
  - used to predict size
  - several methods of computing function points 都很复杂
  - most are proprietary
  - count number of inputs, output, algorithms and tables in database
  - function points is function of above, plus fudge factor (容差系数) for complexity and developer expertise
- coupling and cohesion
  - low coupling and high cohesion
  - coupling – dependencies among modules
  - cohesion – dependencies within modules
  - dependencies
    - call methods, refer to class, share variable
  - number and complexity of shared variables
    - functions **in a module should share** variables
    - functions in **different modules should not**
  - number and complexity of 参数
  - number of functions/modules that are called
  - number of functions/modules that call me
  - DHAMA's coupling metric

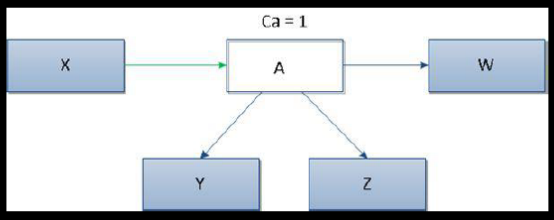


  - Martin's coupling metric (越接近0越稳定)



AFFERENT COUPLING (CA)

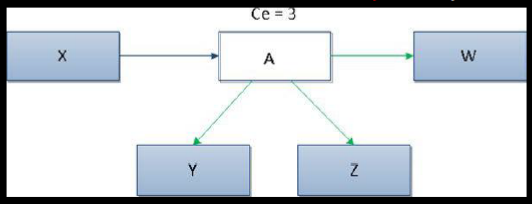
- Measure incoming dependencies.
- Enables us to measure the sensitivity of remaining packages to changes in the analysed package.
- High values of metric Ca usually suggest high component stability.



Incoming dependencies

EFFERENT COUPLING (CE)

- Definition: A number of classes in a given package, which depends on the classes in other packages
- Measure interrelationships between classes.
- Enable us to measure the vulnerability of the package to changes in packages on which it depends.
- High value of the metric Ce> 20 indicates instability of a package,

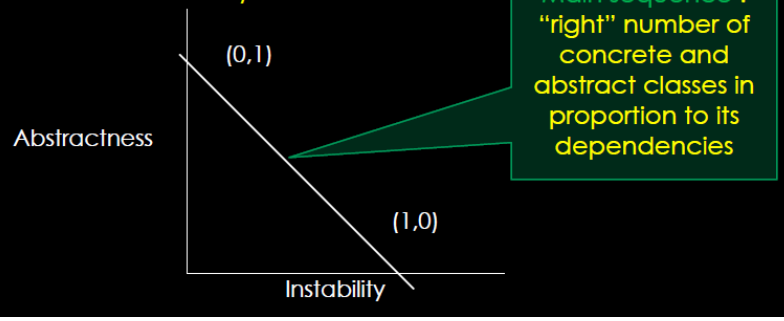


Outgoing dependencies

    - Ce多，Ca少是不稳定的（Instability = 1）: easy changes to these packages
    - 相反，（ins = 0）: more difficult in modifying due to their greater responsibility.
  - Abstractness

Abstractness = $A = \frac{T_{abstract}}{T_{abstract} + T_{concrete}}$

(number of abstract classes in module / number of classes in module)



- oo-specific metrics

pop quiz

cyclometric complexity
=Number of branches (`if`, `while`, `for`) + 1

- What is the cyclometric complexity of the code below?
  - 1
  - 12
  - 13
  - 14

```
String getMonthName (int month) {
        switch (month) {
                case 0: return "January";
                case 1: return "February";
                case 2: return "March";
                case 3: return "April";
                case 4: return "May";
                case 5: return "June";
                case 6: return "July";
                case 7: return "August";
                case 8: return "September";
                case 9: return "October";
                case 10: return "November";
                case 11: return "December";
                default: throw new IllegalArgumentException();
        }
}
```

```
int detect(int seconds, boolean isTimeSensitive) {
1       boolean isSlow = false;
2       int isPerIssue = 0;
3       if (seconds > 10 && seconds < 100) {
4           isSlow = true;
        }
5       if (isTimeSensitive) {
6           if (isSlow) {
7               isPerIssue = 1;
            }
        }
8       return isPerIssue;
    }
```

What is the cyclomatic complexity?

- 3
- 4
- 5
- 6