GRASP 1

Low Coupling, High Cohesion, Information Expert

NIFR - 2021

GRASP

Agenda

- GRASP
  - Low Coupling & High Cohesion
  - Information Expert
- Refactoring

Case: Coffe Collective

Coffee Collective Case

Group development & Reviews

- Working as groups
- Adding features + Fixing bugs in existing code base
- Review of code

Forking from my repository

General Responsibility Assignment Software Patterns

“Understanding responsibilities is key to good object-oriented design”

Martin Fowler

GRASP patterns

9 patterns

Information expert

Creator

Controller

Indirection

Low Coupling

High Cohesion

Polymorphism

Protected Variations

Pure Fabrication

GRASP patterns

9 patterns

Information expert

Creator

Controller

Indirection

Low Coupling

High Cohesion

Polymorphism

Protected Variations

Pure Fabrication

Why patterns?
Technical Debt

“If we failed to make our program align with what we then understood to be the proper way to think about our financial objects, then we were going to continue to stumble on that disagreement which is like paying interest on a loan.”

Ward Cunningham

Technical Debt

Oracle Database 12.2.
It is close to 25 million lines of C code.
What an unimaginable horror! You can't change a single line of code in the product without breaking 1000s of existing tests. Generations of programmers have worked on that code under difficult deadlines and filled the code with all kinds of crap.
Very complex pieces of logic, memory management, context switching, etc. are all held together with thousands of flags. The whole code is ridden with mysterious macros that one cannot decipher without picking a notebook and expanding relevant pats of the macros by hand. It can take a day to two days to really understand what a macro does.
Sometimes one needs to understand the values and the effects of 20 different flag to predict how the code would behave in different situations. Sometimes 100s too! I am not exaggerating.
The only reason why this product is still surviving and still works is due to literally millions of tests!

https://news.ycombinator.com/item?id=18442637

```
if (appForm.getFileCV() == null || StringUtils.isEmpty(appForm.getFileCV().getFileName())){
   errors.add("fileCV", new ActionError("error.fileCV.required"));
}else if (!appForm.getFileCV().getFileName().toLowerCase().endsWith(".doc") && !appForm.getFileCV().getFileName().toLowerCase().endsWith(".pdf")
      && !appForm.getFileCV().getFileName().toLowerCase().endsWith(".rtf") && !appForm.getFileCV().getFileName().toLowerCase().endsWith(".sdc")
      && !appForm.getFileCV().getFileName().toLowerCase().endsWith(".zip") )
   errorExtension = true;
if (appForm.getFileLetter() == null || StringUtils.isEmpty(appForm.getFileLetter().getFileName())){
   errors.add("fileLetter", new ActionError("error.fileLetter.required"));
}else if (!appForm.getFileLetter().getFileName().toLowerCase().endsWith(".doc") && !appForm.getFileLetter().getFileName().toLowerCase().endsWith(".pdf")
      && !appForm.getFileLetter().getFileName().toLowerCase().endsWith(".rtf") && !appForm.getFileLetter().getFileName().toLowerCase().endsWith(".zip")
      && !appForm.getFileLetter().getFileName().toLowerCase().endsWith(".zip"))
   errorExtension = true;
/* if (fileForm == null || StringUtils.isEmpty(fileForm.getFileName())){
   errors.add("fileForm", new ActionError("error.fileForm.required"));
}else*/
if(appForm.getFileForm() != null && !StringUtils.isEmpty(appForm.getFileForm().getFileName()))
   if (!appForm.getFileForm().getFileName().toLowerCase().endsWith(".doc") && !appForm.getFileForm().getFileName().toLowerCase().endsWith(".pdf")
      && !appForm.getFileForm().getFileName().toLowerCase().endsWith(".rtf") && !appForm.getFileForm().getFileName().toLowerCase().endsWith(".sdc")
      && !appForm.getFileForm().getFileName().toLowerCase().endsWith(".zip"))
   errorExtension = true;
if(appForm.getFileOther1() != null && !StringUtils.isEmpty(appForm.getFileOther1().getFileName()))
   if (!appForm.getFileOther1().getFileName().toLowerCase().endsWith(".doc") && !appForm.getFileOther1().getFileName().toLowerCase().endsWith(".pdf")
         && !appForm.getFileOther1().getFileName().toLowerCase().endsWith(".rtf")&& !appForm.getFileOther1().getFileName().toLowerCase().endsWith(".sdc")
         && !appForm.getFileOther1().getFileName().toLowerCase().endsWith(".zip"))
      errorExtension = true;
if(appForm.getFileOther2() != null && !StringUtils.isEmpty(appForm.getFileOther2().getFileName()))
   if (!appForm.getFileOther2().getFileName().toLowerCase().endsWith(".doc") && !appForm.getFileOther2().getFileName().toLowerCase().endsWith(".pdf")
         && !appForm.getFileOther2().getFileName().toLowerCase().endsWith(".rtf")&& !appForm.getFileOther2().getFileName().toLowerCase().endsWith(".sdc")
         && !appForm.getFileOther2().getFileName().toLowerCase().endsWith(".zip"))
      errorExtension = true;
if(appForm.getFileOther3() != null && !StringUtils.isEmpty(appForm.getFileOther3().getFileName()))
   if (!appForm.getFileOther3().getFileName().toLowerCase().endsWith(".doc") && !appForm.getFileOther3().getFileName().toLowerCase().endsWith(".pdf")
         && !appForm.getFileOther3().getFileName().toLowerCase().endsWith(".rtf")&& !appForm.getFileOther3().getFileName().toLowerCase().endsWith(".sdc")
         && !appForm.getFileOther3().getFileName().toLowerCase().endsWith(".zip"))
      errorExtension = true;
```

http://blog.cherouvim.com/the-worst-codebase-ive-seen-in-my-life/

The GRASP principles or patterns are a learning aid to help you understand essential object design and apply design reasoning in a methodical, rational, explainable way. This approach to understanding and using design principles is based on *patterns of assigning responsibilities*

Larman, Applying UML and Patterns, 276

Information Expert

GRASP principle

Problem: What is a basic principle by which to assign responsibilities to objects?

Solution: Assign responsibility to the class that has the information needed to fulfill it.

Information Expert

GRASP principle

Using the principle of information expert, a general approach to assigning responsibilities is to look at a given responsibility, determine the information needed to fulfill it, and then determine where that information is stored.

This will lead to placing the responsibility on the class with the most information required to fulfill it.

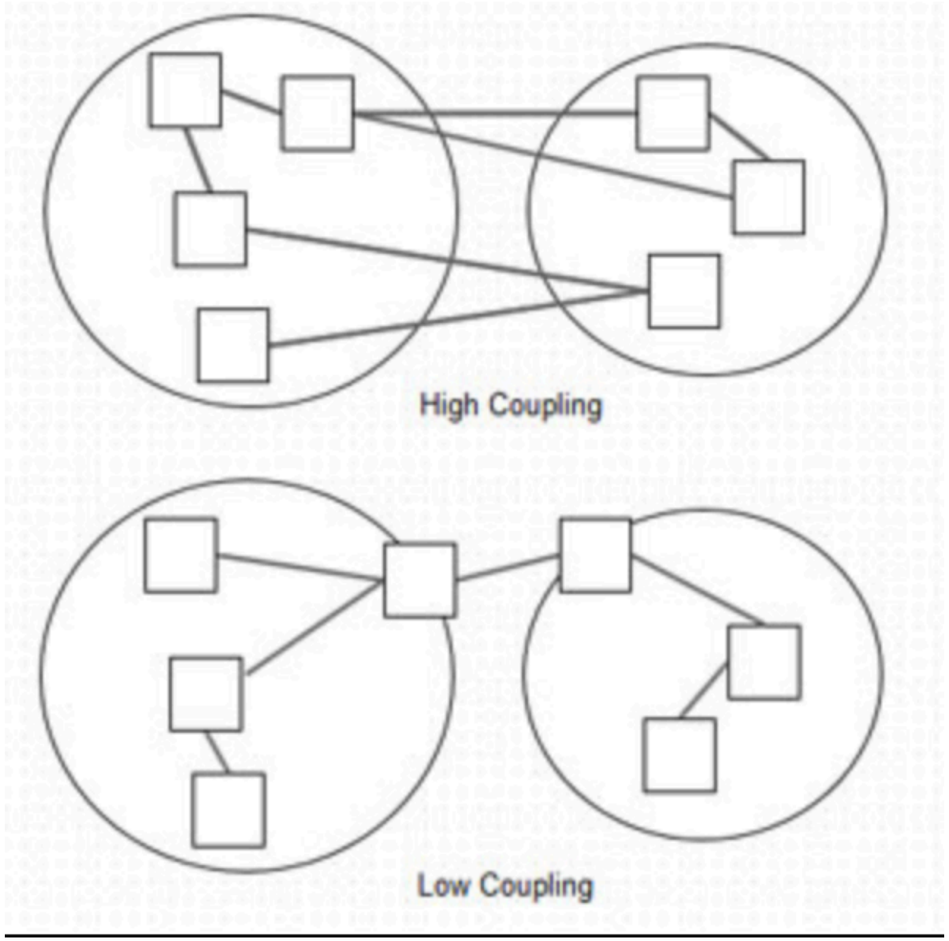
HELPDESK

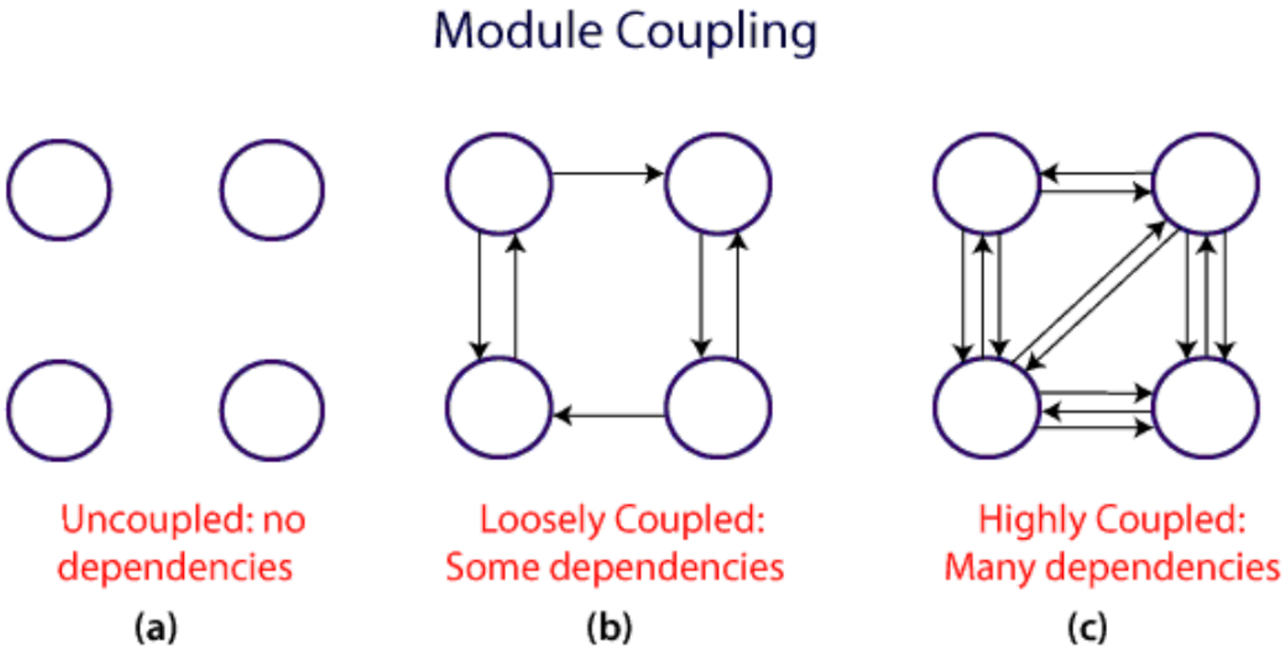
Coupling & Cohesion

Coupling is a measure of how strongly one element is connected to, has knowledge of, or relies on other elements.

[**High] cohesion** is an evaluative pattern that attempts to keep objects appropriately focused, manageable and understandable. High cohesion is generally used in support of low coupling. High cohesion means that the responsibilities of a given element are strongly related and highly focused.

Coupling


High Coupling
Low Coupling


Module Coupling
Uncoupled: no dependencies
(a)
Loosely Coupled: Some dependencies
(b)
Highly Coupled: Many dependencies
(c)

Cohesion

```
public class Student {
    private int studentid;
}
```

```
public class FileReader {
    private int howManyStudents;
    private File fileToRead;

    public String readFile(){
        return "not implemented yet";
    }
}
```

Debug & Bring down technical debt

CoffeeCollective

- In groups
- Note findings & problems
- Note solutions for problems
- Note assumptions