SE-Lec7

TECHNICAL OO METRIC

metric

- Weighted Methods Per Class (WMC)
  - WMC for a class is the sum of the complexities of the methods in the class
  - Possible method complexities
    - 1 (number of method)
    - Lines of code
    - Number of method call
    - Cyclomatic complexity
  - 方法的数量和方法的复杂性预测了开发和维护类所需的时间和精力
  - 一个类中方法的数量越多，对子类的潜在影响就越大
  - 具有大量方法的类更有可能是特定于应用程序的，并且不易重用
- Depth of Inheritance Tree (DIT)
  - Maximum length from a class to the root (继承关系树高度)
  - 类在层次结构中越深，继承的方法就越多，因此很难预测其行为
  - 类在层次结构中越深，重用的方法就越多
  - 越深的关系树越复杂
- Number of Children (NOC)
  - Number of immediate subclasses
  - 子类越多，重用就越多
  - 一个类可能有很多子类是因为子类的错误使用
  - 有大量子类的类可能非常重要，需要进行大量测试
  - 几乎所有的类都有0个子类，只有少数类会有5个以上的子类
- Coupling between Object Classes (CBO)
  - Number of other classes to which a class is coupled
  - Class A is coupled to class B if there is a method in A that invokes a method of B
  - Want to be coupled only with abstract classes high in the inheritance hierarchy
  - 耦合使设计难以改变，使得类很难重用
  - 耦合是衡量一个类的难易程度的指标 C++ project: median 0, max 84, Smalltalk project: median 9, max 234
- Response for a Class (RFC)
  - Number of methods in a class or called by a class
  - 类的响应集是一组方法，这些方法可能会响应该类对象接收到的消息而执行
  - 如果响应消息可以调用大量方法，测试就会变得更加复杂
  - 可以从类中调用的方法越多，类的复杂性就越大　C++: median 6, max 120, Smalltalk: median 29,
- Lack of Cohesion in Methods (LCOM)

    - Number of pairs of methods that don't share instance variables minus number of pairs of methods that share instance variables
    - Cohesiveness of methods is a sign of encapsulation 方法的内聚性是封装的标志
    - Lack of cohesion implies classes should be split 缺乏内聚意味着类应该被分割
    - C++: median 0, max 200, Smalltalk: median 2, max 17 (had only a few zero)

SUMMARY: METRICS (IN GENERAL)

Non-technical: about process

Technical: about product

- Size, complexity (cyclomatic, function points)

How to use metrics

- Prioritize work - compare modules in a system version, compare system versions over time
  确定工作优先级-比较系统版本中的模块，随时间比较系统版本
- Measure programmer productivity
  衡量程序员的工作效率
- Make plans based on predicted effort
  根据预测的工作量制定计划

REVERSE ENGINEERING

How to learn an existing software system that is poorly documented?

Reverse engineering definition by Chkofsky & Cross:

"Reverse Engineering is the process of analyzing a subject system to identify the system's components and their interrelationships and create representations of the system in another form or at a higher level of abstraction."

逆向工程是分析主题系统以识别系统组件及其相互关系并以另一种形式或更高抽象级别创建系统表示的过程。

WHAT IS REVERSE ENGINEERING?

- Discovering design of an artifact
    - From lower level to higher level; for example:
        - Given binary, discover source code
        - Given code, discover specification & design rationale 规范和设计原理
- Layman: trying to understand how the system works
- When are you done?
    - Learn enough to:
        - Change it, or
        - Replace it, or

        - Write a book about it

WHY TO REVERSE ENGINEER?

- 更换整个系统
- 重新设计或修改系统
- 使用系统
- 为系统编写文档
- 了解你的意图

WHEN TO REVERSE ENGINEER?

- In the large:
    - Big changes: Y2K (99 → 00 → 01)
    - Technology change: desktop → web → mobile
- In the small
    - Add a feature, fix a bug, improve performance
- Your project
    - Understand and modify parts of some large codebase

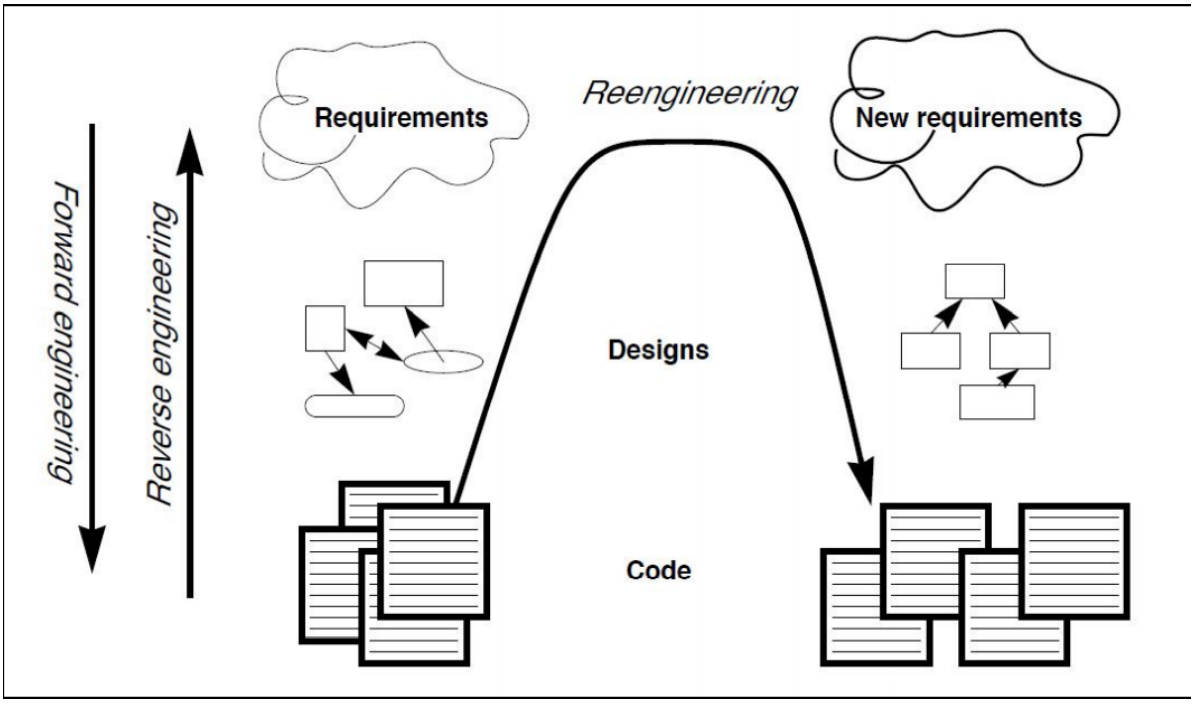
HOW TO REVERSE ENGINEER?

Make use of various/any sources, to incrementally build an explicit or a conceptual model of software. E.g.:

- Read existing documentation
- Read source code
- Run the software
- Interview users and developers
- Execute existing or write new test cases
- Generate and analyze traces
- Use tools to generate high-level views of code / traces
- Analyze the version history
- Etc., basically you can use anything that helps

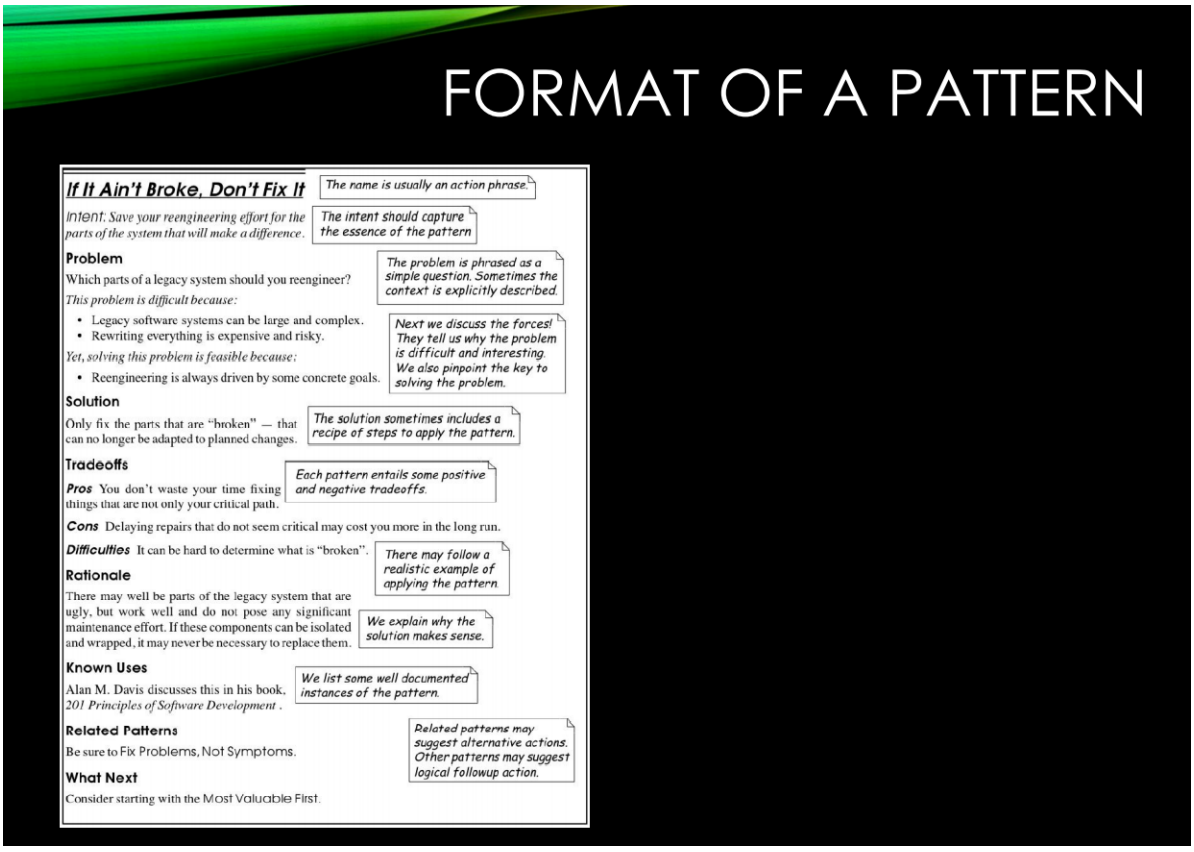
SOME TERMINOLOGY

- Forward engineering
    - From requirements to design to code
- Reverse engineering
    - From code to design, maybe to requirements
- Reengineering
    - From old code to new code via some desige

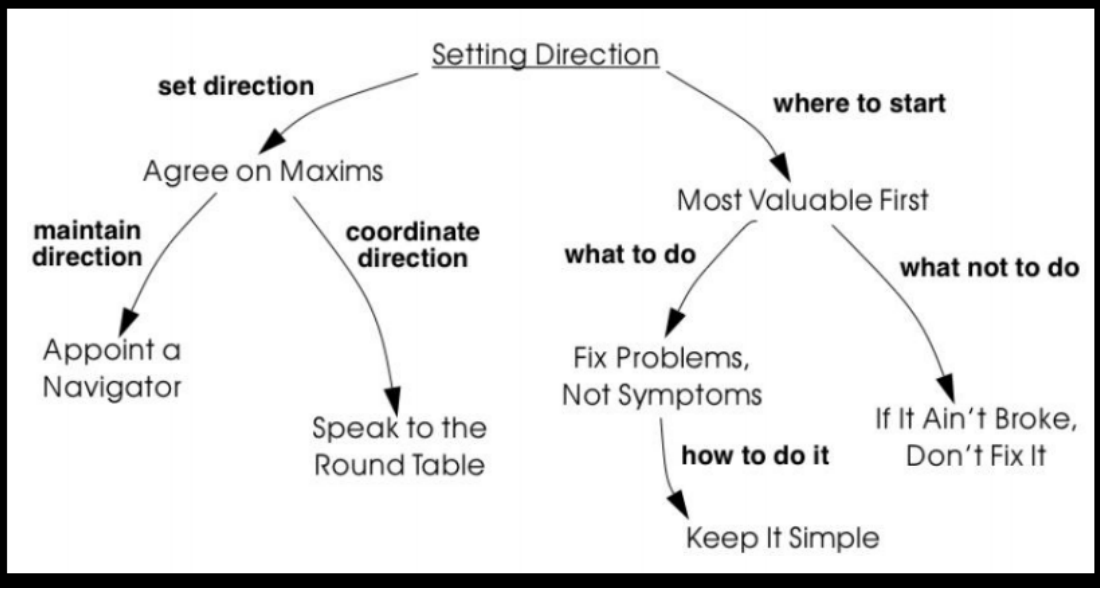


Reverse engineering needed for re-engineering

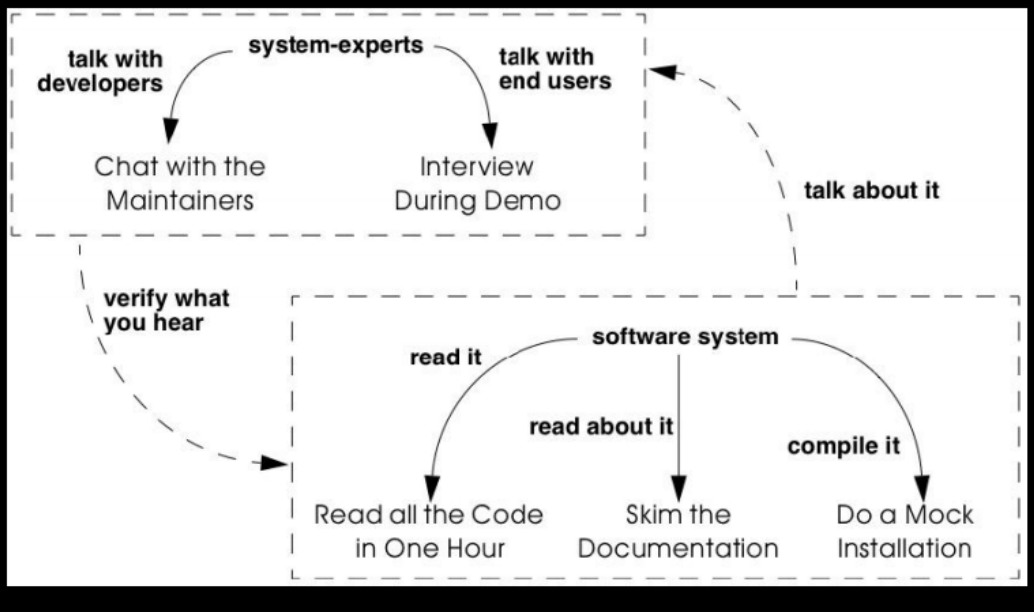
PATTERN



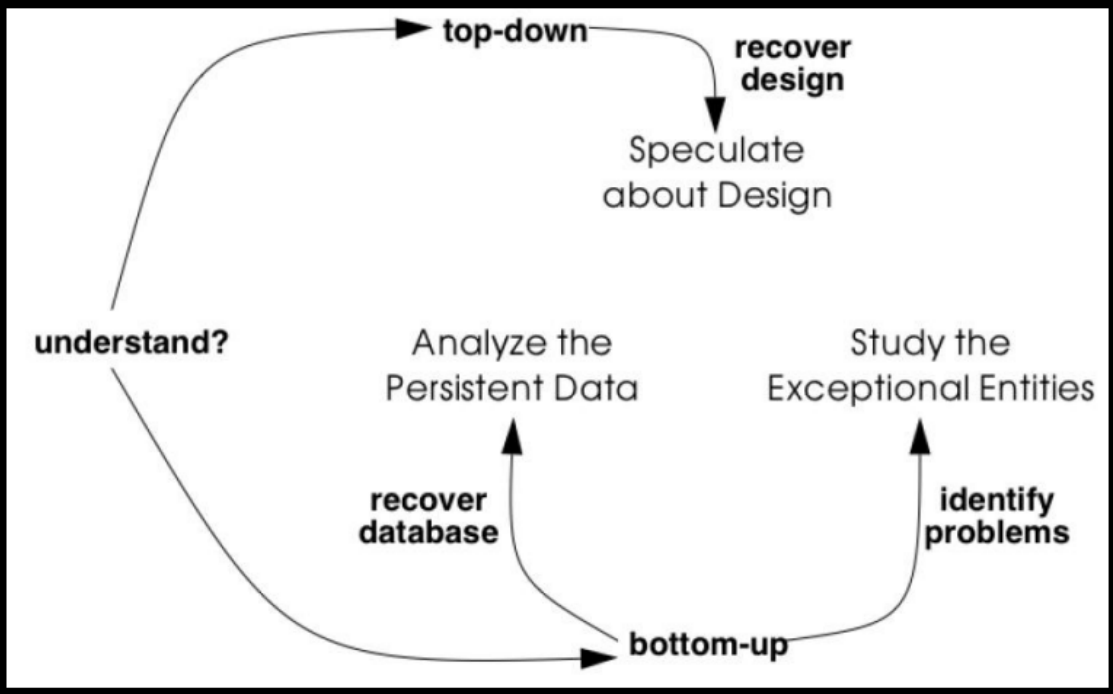
SETTING DIRECTION PATTERNS



FIRST CONTACT PATTERNS



PATTERNS FOR INITIAL UNDERSTANDING



PATTERNS FOR DETAILED MODEL CAPTURE