第八章 软件维护

周 瑞

QQ群：235458708

信息与软件工程学院

本章学习目标

1. 掌握软件维护的概念、类型、软件可维护性的决定因素、软件维护过程以及IEEE维护模型
2. 理解软件维护技术
3. 理解软件再工程的概念及主要内容

本章内容

- 软件维护的概念
- 软件维护的类型
- 软件可维护性及影响因素
- 软件维护过程及IEEE维护模型
- 软件维护技术

软件维护的概念

- 软件交付给最终用户，
  - 几天之内，缺陷报告就有可能送到软件工程组织。
  - 几周之内，某类用户就可能会提出必须修改软件以适应他们所处环境的特殊要求。
  - 几个月之内，另一个公司可能意识到该软件可能会给他们带来意想不到的好处，因此他们需要做些改进，使软件可以用于他们的环境。
- 所有这些工作都是软件维护

软件维护的概念

- 简单的说：
  - 维护是在软件运行阶段对软件产品所做的修改。
- IEEE/EIA 12207[ISO/IEC2008]中对软件维护的定义是：
  - 软件维护是指由于软件产品出现问题或需要改进而对代码及相关文档的修改，其目的是对现有软件产品进行修改的同时保持其完整性。

软件维护的必要性

1. 软件维护能够改正错误
2. 软件维护能够改善设计
3. 软件维护能够实现软件的改进
4. 软件维护能够与其它系统进行交互
5. 软件维护能够为使用不同的硬件、软件、系统的新性能以及通讯设备等而对软件进行改进
6. 软件维护能够完成遗留程序的移植
7. 软件退出使用

软件维护的基本类型

- 维护的类型主要有四种：
  - 纠错性维护
  - 适应性维护
  - 完善性维护
  - 预防性维护

纠错性维护

- 在软件交付使用后，因开发时测试的不彻底、不完全，必然会有部分隐藏的错误遗留到运行阶段
- 这些隐藏下来的错误在某些特定的使用环境下就会暴露出来
- 为了识别和纠正软件错误、改正软件性能上的缺陷、排除实施中的误使用，应当进行错误诊断和改正
- 这种情况下进行的维护活动叫做纠错性维护

适应性维护

- 在软件使用过程中，环境可能发生变化
  - 外部环境：新的硬件、软件配置
  - 数据环境：数据库、数据格式、数据输入/输出方式、数据存储介质
- 为使软件适应这种变化，需要修改软件
- 这种情况下进行的维护活动叫做适应性维护

完善性维护

- 软件使用过程中，用户往往会对软件提出新的功能与性能要求
- 为满足这些要求，需要修改或再开发软件，以扩充软件功能、增强软件性能、改进加工效率、提高软件的可维护性
- 这种情况下进行的维护活动叫做完善性维护
- 完善性维护不一定是救火式的紧急维修，而可以是有计划、有预谋的一种再开发活动
- 实践表明，在几种维护活动中，完善性维护所占的比重最大，即大部分维护工作是改变和加强软件，而不是纠错
- 事实证明，来自用户要求扩充、加强软件功能、性能的维护活动约占整个维护工作的50%以上

预防性维护

- 预防性维护定义为：采用先进的软件工程方法对需要维护的软件或软件中的某一部分（重新）进行设计、编制和测试
- 预防性维护是为了提高软件的可维护性、可靠性等，为以后进一步改进软件打下良好基础

- 统计数字表明
  - 完善性维护占全部维护活动的50%～66%
  - 适应性维护占18%～25%
  - 纠错性维护占17%～21%
  - 其它维护活动只占4%左右

维护的困难性

1、配置管理工作不到位
2、人员变动造成的影响
3、许多软件的可读性差
4、任务紧、时间急的情况下处理维护请求

软件可维护性

- 软件可维护性:对软件进行维护的容易程度

- 影响可维护性的主要因素
  - ①可理解性
  - ②可测试性
  - ③可修改性
  - ④可移植性
  - ⑤可重用性

- 影响软件可维护性的维护环境的因素
  - ①软件维护的文档
  - ②软件的运行环境
  - ③软件的维护组织
  - ④软件维护质量

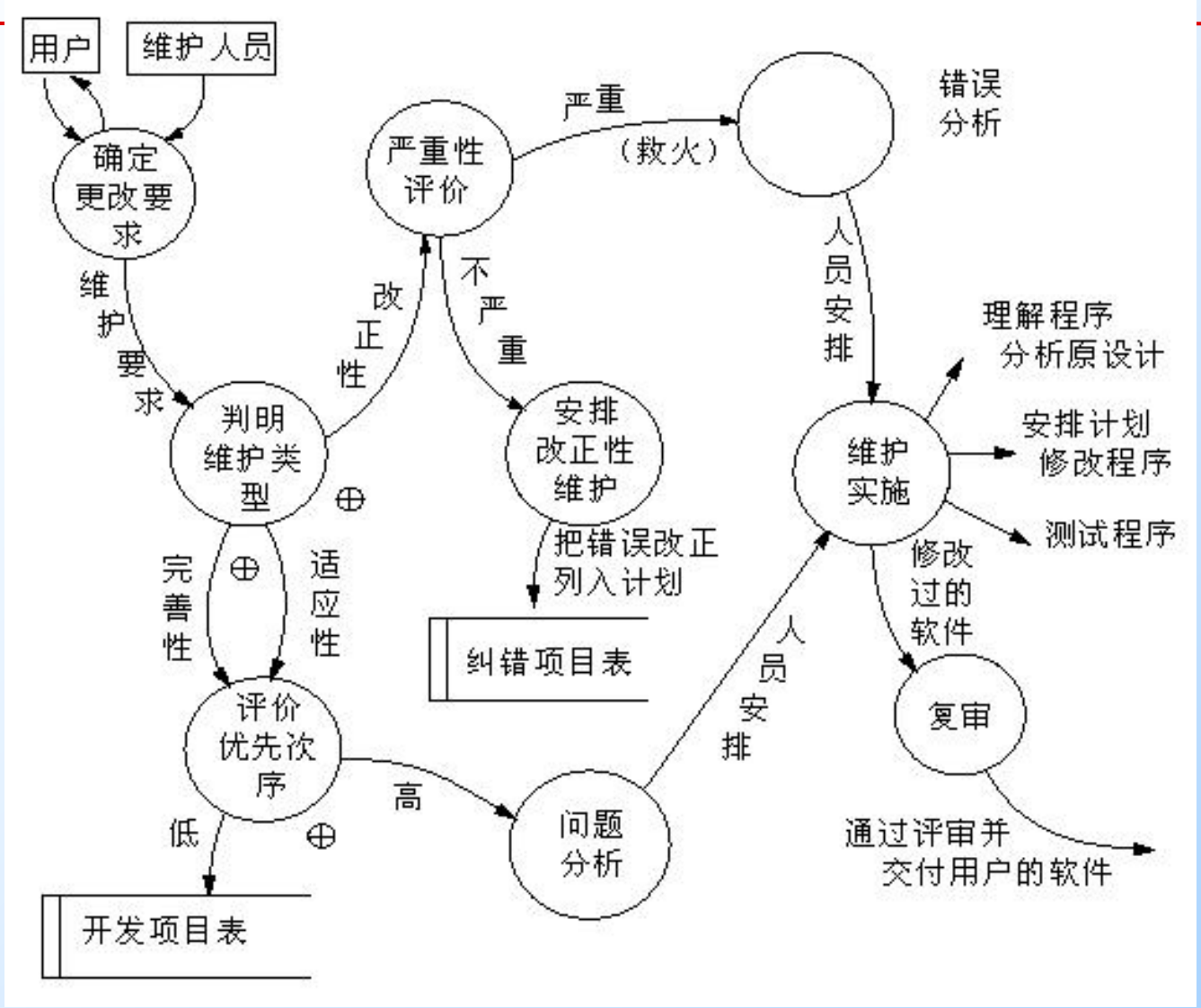
软件可维护性

- 可维护的软件表现为：
  - 有效的模块性
  - 采用易于理解的设计模式
  - 采用明确定义的编码标准和约定
  - 源代码能够自身文档化且易于理解
  - 应用质量保证技术，软件交付之前已找到潜在维护问题

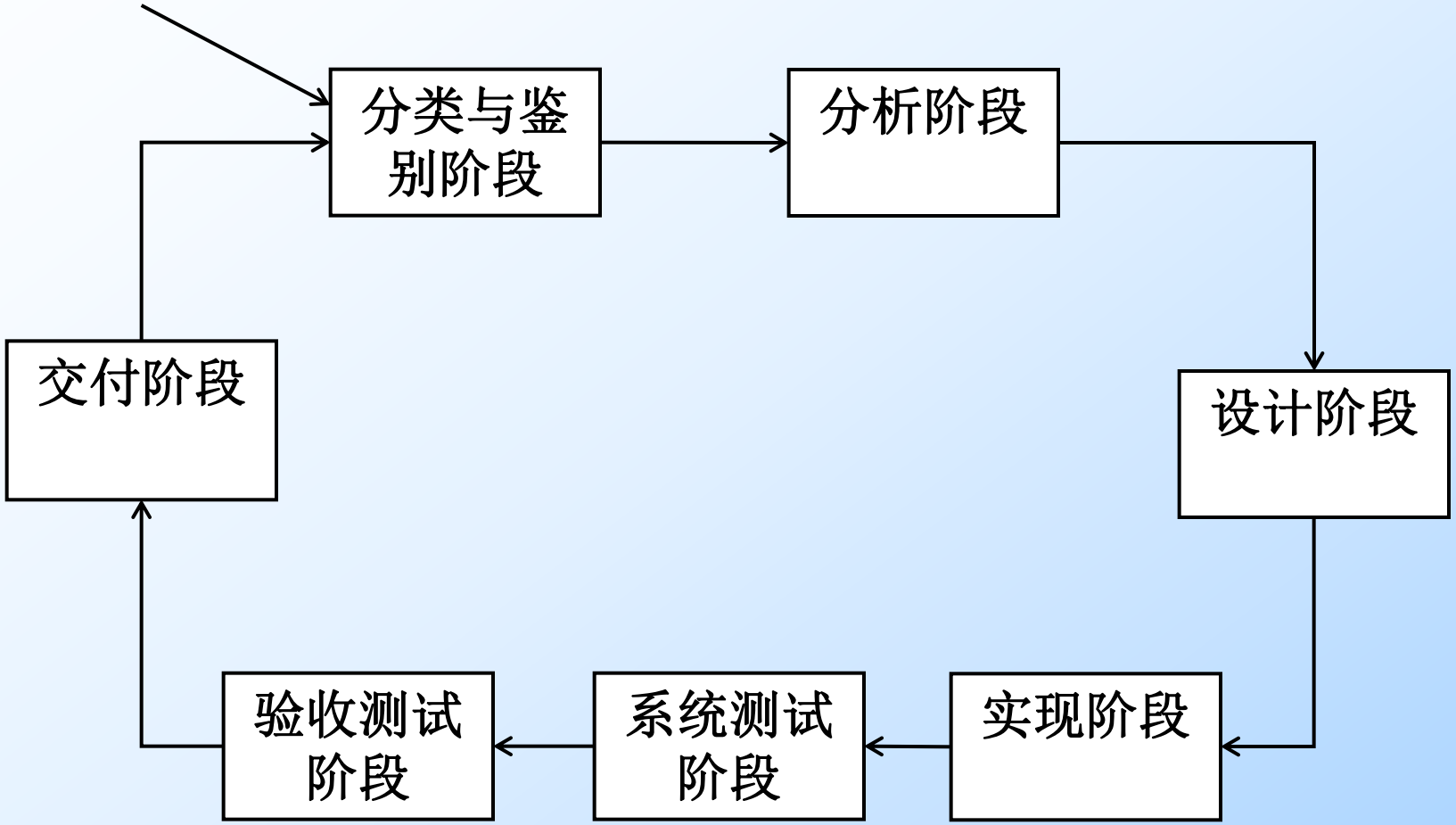
维护费用估算

- 参数模型：$M = P + K \times \exp(c - d)$
  - M是维护用的总工作量
  - P是生产性工作量
  - K是经验常数，
  - c是复杂程度，d是维护人员对软件的熟悉程度
- 基于经验：专家判断、类推、工作分解结构
- 维护费用：
  - 软件维护阶段一般要消耗软件生命周期中经费开支的大部分
  - 70年代：软件总预算的35%-40%；80年代：40%-60%；90年代：70%-80%

软件维护过程


用户
维护人员
确定更改要求
维护要求
判明维护类型
严重性评价
改正性
不严重
严重
（救火）
错误分析
人员安排
安排改正性维护
把错误改正列入计划
纠错项目表
维护实施
理解程序
分析原设计
安排计划
修改程序
测试程序
修改过的软件
复审
通过评审并交付用户的软件
完善性
适应性
评价优先次序
高
低
问题分析
人员安排
开发项目表

IEEE维护模型图



软件维护技术

- 程序理解
- 软件再工程
- 软件逆向工程

程序理解

- 清晰简明的文档
- 代码浏览工具（Source Insight）
- 程序理解的任务：以软件维护、升级和再工程为目的，在不同的抽象级别上(实现层、结构层、功能层、领域层)建立基本软件的概念模型，包括从代码本身的模型到基本应用领域的模型，即建立从问题（应用）域到程序设计(实现)域的映射清晰。

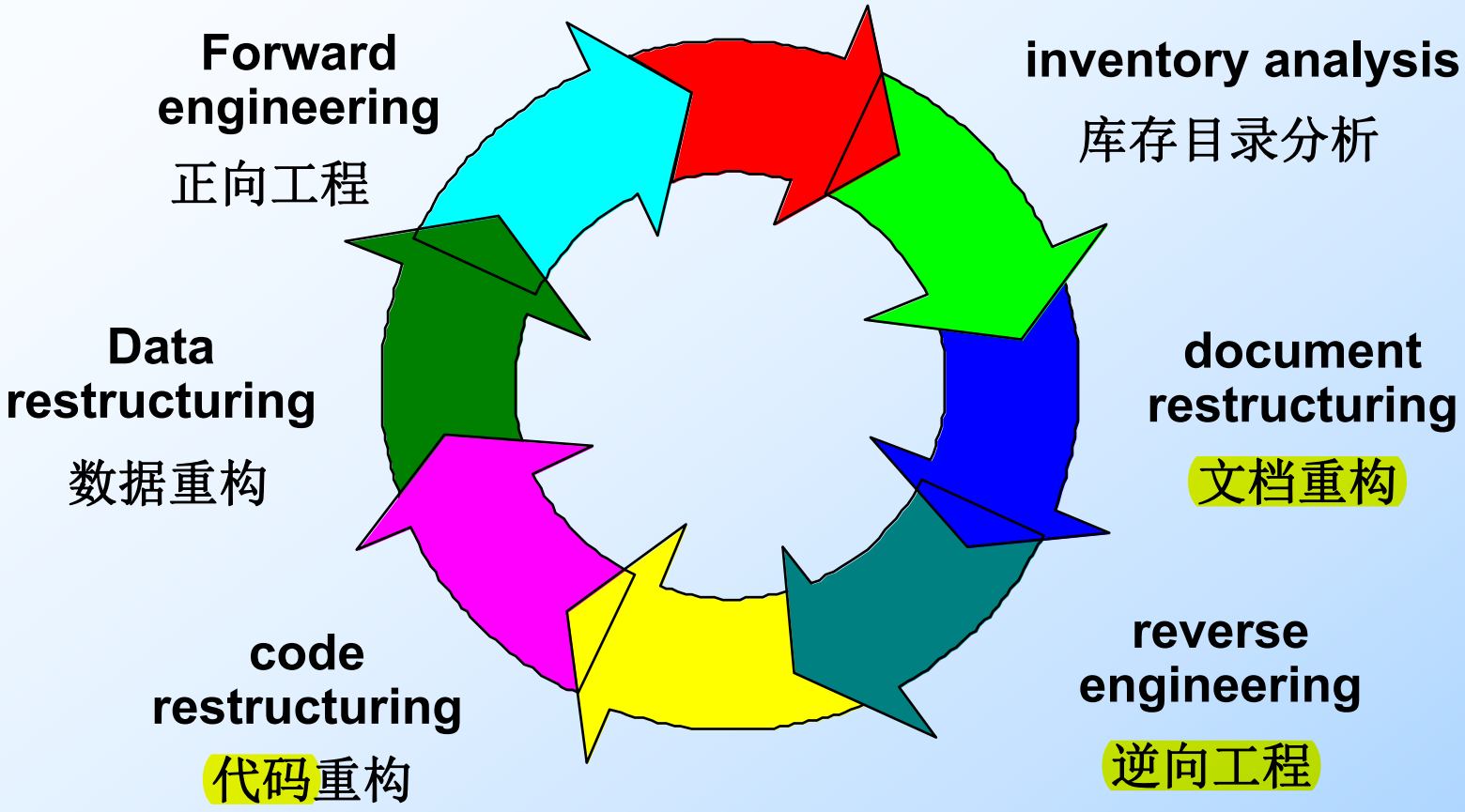
程序理解的任务

- 通过检查单个的程序设计结构，程序被表示成抽象语法树、符号表或普通源文本
- 尽量做到程序隐含信息的显性表示及程序内部关系的可视化
- 从源代码中提取信息，并存放在通用的数据库中，然后通过查询语言对数据库进行查询
- 检查程序构造过程中的结构关系，明确表示程序组成部分之间的依赖关系
- 识别程序的高层概念，如标准算法、数据结构、语法及语义匹配等

软件再工程

- Re-engineering
  - 指对现有软件进行仔细审查和改造，对其进行重新构造，使之成为一个新的形式，同时包括随之产生的对新形式的实现

软件再工程模型


Forward engineering
正向工程
inventory analysis
库存目录分析
Data restructuring
数据重构
document restructuring
文档重构
code restructuring
代码重构
reverse engineering
逆向工程

软件再工程：库存目录分析

- 对软件组织用语的每个应用系统都进行预防性维护是不现实的，也是不必要的。一般说来，下述3类程序有可能成为预防性的对象：
  - 该程序将在今后数年内继续维护的对象
  - 当前正在成功地使用着该程序
  - 可能在最近的将来要对该程序做较大程度的修改或扩充

软件再工程：库存目录分析

- 分析库存目录：构建一张包含所有应用系统的表
  - 应用系统的名字
  - 最初创建时间
  - 所做实质性变更的数量
  - 使这些变更得以应用的总的工作量
  - 上一个实质性变更的日期
  - 使上一个变更得以应用的工作量
  - 它驻留的系统
  - 应用程序接口……
- 按照业务重要程度、寿命、当前可维护性、预期的修改次数等标准，把库中的应用排序
- 从中选出再工程的侯选者
- 然后合理地分配再工程所需要的资源

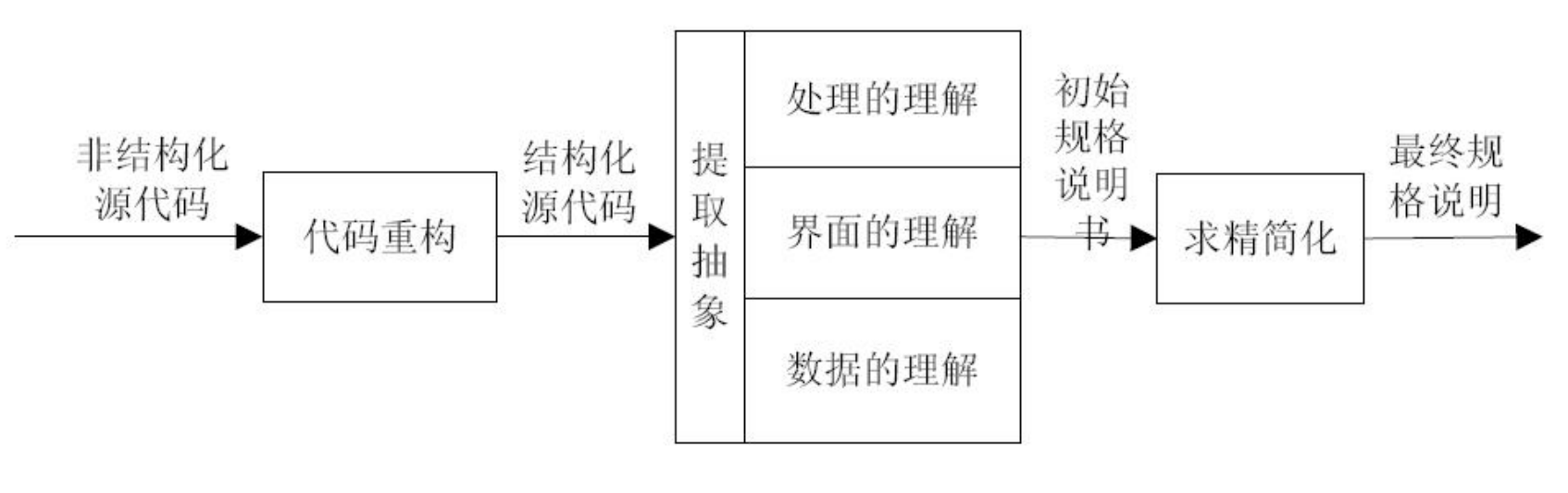
软件再工程：文档重构

- 老程序固有的特点是缺乏文档，根据具体情况可采用下述三种方法之一来处理这个问题：
  - 如果一个程序是相对稳定的，正在走向生命的终点，而且可能不会再修改它，则不必为它建立文档；
  - 为了便于今后的维护，必须更新文档，但是由于资源有限，应该采用“使用时建立文档”的方法；
  - 如果某应用系统是用户完成业务工作的关键，而且必须重构全部文档，则仍然应该尽量把文档工作减少到必需的最小量。

软件再工程：逆向工程

- Revsese engineering
- 分析程序，在比源程序更高的抽象层次上创建出程序的某种描述
- 逆向工程是一个恢复设计结果的过程，从现有程序中抽取数据、体系结构和过程的设计
- 软件逆向工程是分析目标系统，识别系统的构件及其交互关系，并且通过高层抽象或其它形式来展现目标系统的过程
- 对逆向工程而言，抽象的层次、完备性、工具与分析人员协同工作的程度、过程的方向性等因素是需要考虑的

软件再工程：逆向工程



逆向工程过程

软件再工程：逆向工程

- 逆向工程的主要内容：
  1. 数据的逆向工程
     - 内部数据结构的逆向工程、数据库结构的逆向工程
  2. 处理的逆向工程
     - 需要在不同的抽象级别（系统级、程序级、构件级、模式级和语句级）分析代码
     - 对大型系统，通常用半自动方法完成逆向工程
  3. 用户界面的逆向工程
     - 界面必须处理的基本动作是什么？
     - 系统对这些动作的行为反应的简要描述是什么？

软件再工程：代码重构

- 某些老程序的体系结构比较合理，但是，一些模块的编码方式却难于理解、测试和维护
- 在这种情况下，可以重构这些模块的代码
- 通常，代码重构并不修改程序的体系结构，它只关注个体模块的设计细节以及在模块中定义的局部数据结构
- 如果重构扩展到模块边界之外并涉及软件体系结构，则重构成了正向工程

软件再工程：数据重构

- 对数据体系结构差的程序很难进行适应性和完善性维护
- 因此，数据体系结构比源代码对程序的长期生存力有更大的影响
- 由于数据结构对程序体系结构及程序中的算法有很大影响，对数据的修改必然会导致程序体系结构或代码层的改变
- 数据重构是一种全范围的再工程活动

软件再工程：正向工程

- 正向工程也称为革新或改造
- 正向工程过程应用现代软件工程的概念、原理、技术和方法，重新开发现有的某些应用系统
- 在大多数情况下，经过正向工程后，软件不仅重新实现了现有系统的功能，而且增加了新功能，提高了整体性能

本章复习要点

- 软件维护的概念
- 软件维护的主要类型
  - 纠错性维护、适应性维护、完善性维护、预防性维护
- 软件可维护性及影响因素
- 软件维护过程及IEEE维护模型
- 软件维护技术
  - 程序理解、软件再工程、软件逆向工程