第 10 章　软件重用和再工程

本章内容

10.1 可重用的软件成分

软件重用是在软件开发过程中重复使用相同或相似的软件元素的过程。这些软件元素包括应用领域知识、开发经验、设计经验、体系结构、需求分析文档、设计文档、程序代码及测试用例等。

1 知识重用

- 例如软件工程知识、开发经验、设计经验、应用领域知识等的重用

2 方法和标准的重用

- 包括传统软件工程方法、面向对象方法、有关软件开发的国家标准和国际标准等的重用

3 软件成分的重用

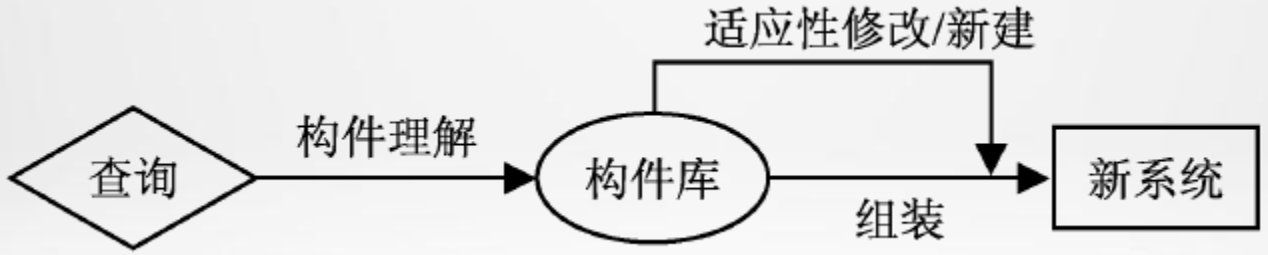
- 重用级别：源代码重用、设计结果重用、分析结果重用
- 可重用的软件成分：项目计划、成本估计、体系结构、需求模型和规格说明、设计、源代码、用户文档和技术文档、用户界面、数据、测试用例

本章内容

10.2.1 软件重用过程的模型

01 OPTION 组装模型

最简单的软件重用过程是，先将以往软件工程项目中建立的软件构件存储在构件库中，然后通过对构件库进行查询，提取可以重用的软件构件；接着，为了适应新系统对它们进行一些修改，并建造新系统需要的其他构件；最后对新系统需要的所有构件进行组装。



10.2.1 软件重用过程的模型

02 OPTION 类重用模型

利用面向对象技术，可以比较方便、有效地实现软件重用。

1 实例重用

按照需要创建类的实例，向该实例发送适当的消息，启动相应的服务，完成所需要的工作

2 继承重用

利用面向对象方法的继承机制，子类可以继承父类已经定义的所有数据和操作，子类也可以定义新的数据和操作

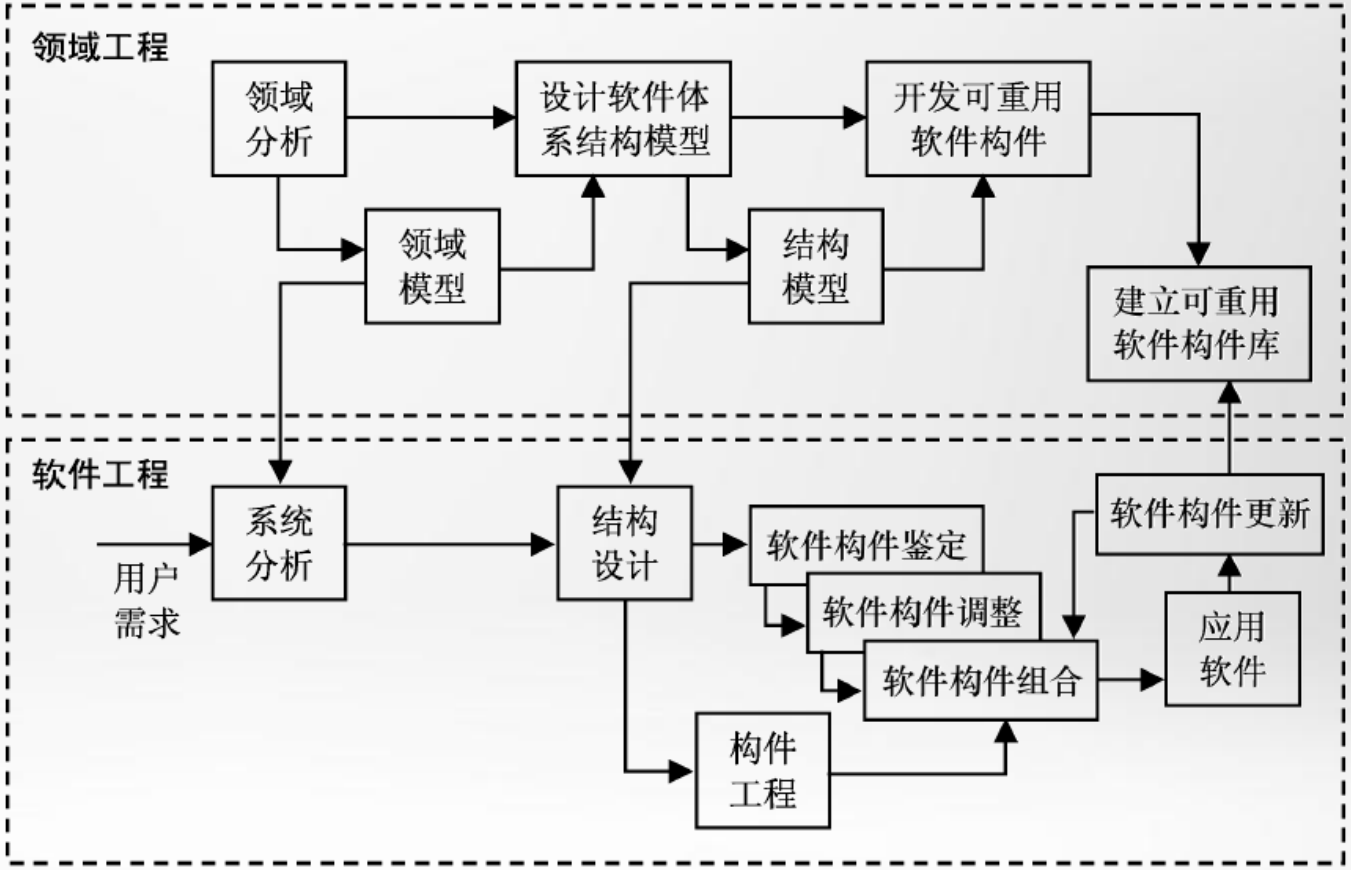
3 多态重用

多态重用方式根据接收消息的对象类型不同，在响应一个一般化的消息时，由多态性机制启动正确的方法，执行不同的操作

10.2.1 软件重用过程的模型

03 OPTION 软件重用过程模型

为了实现软件重用，已经出现了许多过程模型，这些模型都强调领域工程和软件工程同时进行。



10.2.2 开发可重用的软件构件

01 OPTION 分析过程

- 构件的功能在未来的工作中需要吗?
- 构件的功能通用性强吗?
- 构件是否依赖于硬件?
- 构件的设计是否足够优化?
- 构件在重用时需要做大的修改吗?
- 能否将一个不可重用的构件分解成一组可重用的构件?

10.2.2 开发可重用的软件构件

02 OPTION

开发可重用的软件构件

1 标准的数据结构

例如文件结构或数据库结构，所有构件都使用这些标准的数据结构

2 标准的接口协议

建立模块内部接口、外部接口和人机交互界面3 个层次的接口协议

3 程序模板

- 人机交互界面。
- 安全范围设置。
- 与监控传感器通信的管理机制。
- 响应机制。
- 控制机制。

10.2.2 开发可重用的软件构件

03 OPTION 传播软件构件

传播软件构件的目的就是让用户能在成千上万的软件构件中找到自己所需要的构件，这需要很好地描述构件。构件的描述包括构件的功能、使用条件、接口、实现等。对于构件的实现方法，只有准备修改构件的人需要知道，其他人只需了解构件的功能、使用条件和接口等。

10.2.3 分类和检索软件构件

01 OPTION 枚举分类

枚举分类模式通过层次结构来描述构件，在该结构中定义软件构件的类以及子类的不同层次。实际构件放在枚举层次的适当路径的最底层。

枚举分类模式的层次结构容易理解和使用，但在建立层次之前必须完成领域工程，使层次中的项具有足够的信息。

10.2.3 分类和检索软件构件

02 刻面分类

OPTION

- 分析应用领域并标识出一组基本的描述特征，这些描述特征称为刻面。
- 描述一个构件的刻面的集合称为刻面描述表。
- 根据重要性确定刻面的优先次序，并把它们与构件联系起来。
- 刻面可以描述构件所完成的功能、加工的数据、应用构件的操作、实现方法等特征。
- 通常，刻面描述不超过8 个。
- 把关键词的值赋给软件构件库中每个构件的刻面集。
- 使用自动工具完成同义词词典功能，从而可以根据关键词或关键词的同义词，在软件构件库中查找所需要的构件。

10.2.3 分类和检索软件构件

03 OPTION 属性值分类

属性值分类（Attribute-Value Classification）模式为一个领域中的所有构件定义一组属性，然后给这些属性赋值。属性值分类模式与刻面分类模式相似，只是有以下区别。

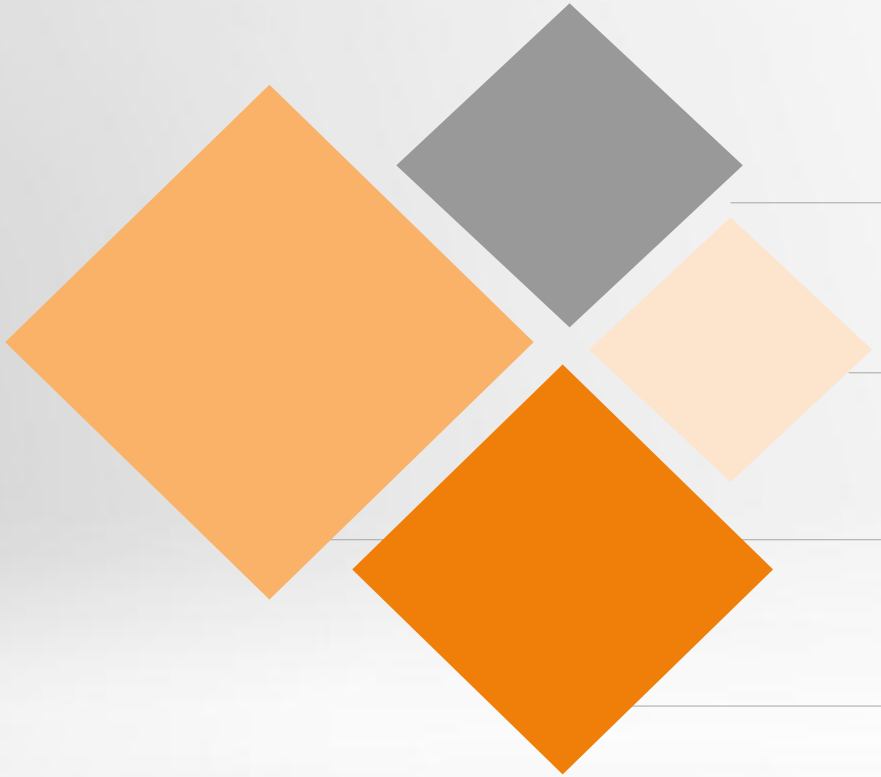
对可重用的属性个数没有限制

属性没有优先级

不使用同义词词典功能

10.2.4 软件重用环境

软件构件的重用必须由相应的环境来支持，环境应包含下列元素。



- 软件构件库：存放软件构件和检索软件构件所需要的分类信息
- 软件构件库管理系统：管理对软件构件库的访问
- 软件构件库检索系统：用户应用系统通过检索系统检索构件、重用构件
- CASE 工具：帮助用户把重用的构件集成到新的设计中

本章内容

10.3 软件逆向工程

01 OPTION 逆向工程概念

逆向工程（Reverse Engineering）是一种产品设计技术的再现过程，即对一项目标产品进行逆向分析及研究，从而演绎并得出该产品的处理流程、组织结构、功能特性及技术规格等设计要素，进而制作出功能相近但又不完全一样的产品。简单地说，逆向工程就是根据已有的产品，反向推出产品设计数据（包括各类设计图或数据模型）的过程。

10.3 软件逆向工程

02 OPTION 逆向工程实现方法

1 分析通过信息交换所得的观察

常用于协议逆向工程，涉及使用总线分析器和数据包嗅探器

2 反汇编

适用于任何计算机程序，对不熟悉机器码的人特别有用，流行的相关工具有OllyDebug 和IDA Pro

3 反编译

使用反编译器，尝试通过程序的机器码或字节码重现高级语言形式的源代码

本章内容

10.4 软件再工程

01 OPTION 再分析阶段

再分析阶段的主要任务是对既有系统的规模、体系结构、外部功能、内部算法、复杂度等进行调查、分析。这一阶段最直接的目的，就是调查和预测再工程涉及的范围。重用是软件工程经济学最重要的原则之一，重用得越多，再工程成本就越低。

10.4 软件再工程

02 OPTION 再编码阶段

根据再分析阶段做成的再工程设计书，再编码阶段将在系统整体再分析基础上对代码做进一步分析。如果说再分析阶段的产品是再工程设计书，那么再编码阶段就要产生类似详细设计书的编码设计书。

10.4 软件再工程

03 OPTION 再测试阶段

一般来说，再测试阶段是再工程过程中工作量最大的一个阶段。如果能够重用原有的测试用例及运行结果，将能大大降低再工程成本。对于可重用的部分，特别是可重用的局部系统，还可以免除测试，这也正是重用技术被再工程高度重视的关键原因之一。

当然，再工程后的系统总有变动和增加的部分，对受其影响的整个范围都要毫无遗漏地进行测试。