第九章题目不多，只有判断填空

流图、软件结构图、层次图、…树、…方向盘

ER图、状态转换图、 盒图 PAD图 判定表 判定树 流图 McCabe方法 计算画图 因果图法、P180稳态可用性、P305、P308、最佳规模5.5人、Gantt图、最多通信信道、

问答提1234的点、罗列出来

13章的内容较为重要 问答、填空、公式

P111软件结构图

测试的步骤

区分黑白测试，判断、原则

Chap1

软件

软件的定义：软件＝程序＋数据＋文档

软件是计算机系统中与硬件相互依存的另一部分，它是包括程序，数据及其相关文档的完整集合

程序是按事先设计的功能和性能要求执行的指令序列

数据是使程序能正常操纵信息的数据结构

文档是与程序开发，维护和使用有关的图文材料

软件的特点：

软件是一种逻辑实体，具有抽象性和不可见性  
软件的生产与硬件不同，没有明显的制造过程，由开发和工程化形成，成本主要集中在开发上

在软件运行、使用期间没有机械磨损、老化问题  
软甲开发和运行常常由计算机系统限制，对其有依赖性  
软件的开发未摆脱手工艺的开发方式  
软件本身复杂（实际问题、程序逻辑结构）

软件成本较高

软件的分类：

|  |  |
| --- | --- |
| 按功能划分 | 系统软件、支撑软件、应用软件 |
| 按规模划分 | 微型、小型、中型、大型、甚大型、极大型 |
| 按工作方式 | 实时处理软件、分时软件、交互式软件、批处理软件 |

软件技术面临的问题：复杂性、生产率

|  |  |
| --- | --- |
| 软件危机定义 | 在计算机软件开发、维护过程中所遇到的一系列严重问题。（bug 新需求 硬件操作系统更新） |
| 软件危机的主要特征 | 软件开发周期大大超过规定日期 软件开发成本严重超标 软件质量难以保证 |
| 主要问题 | 如何开发软件、满足对软件日益增长的需求 如何维护数量不断膨胀的已有软件 |

软件工程

|  |  |
| --- | --- |
| 定义 | 为了经济地获得可靠的且能在实际机器上运行的软件。而建立和使用完善的工程原理 （工程学科、工程原理） |
| 本质 | 关注与大型程序的构造 中心课题是控制复杂性 软件经常变化 开发软件的效率非常重要 和谐地合作是开发软件的关键 软件必须有效地支持它的客户 在软件工程领域中是由具有一种文化背景的人替具有另一种文化背景的人创造产品 |
| 基本原理 | 用分阶段的生命周期计划严格管理 坚持进行阶段评审 实行严格 的产品控制 结果应能清楚地审查 采用现代程序设计技术 开发小组人员应该少而精 承认不断改进软件工程实践的必要性 |

软件工程方法学：把在软件生命周期全过程中使用的一整套技术方法的集合称为方法学

软件工程方法学三要素：方法、工具和过程

质量焦点：支持软件工程的根基就在于对质量的关注。

传统方法学：生命周期方法学（划分阶段，顺序完成）

面向对象方法学：将数据和对数据的操作结合起来

OOM四要素

|  |  |
| --- | --- |
| 对象 | 世界由对象组成 |
| 类 | 相同数据、相同操作的对象 |
| 继承 | 类可分层，子类与父类有相同特征 |
| 消息 | 对象间的信息传递 |

问 题 定 义 
软 件 定 义 
可 行 性 研 宄 
需 求 分 析 
总 体 设 计 
系 统 设 计 
详 细 设 计 
软 件 生 命 周 期 
软 件 开 发 
编 码 和 单 元 测 试 
系 统 实 现 
综 合 测 试 
行 
维 
护 

|  |  |
| --- | --- |
| 定义 | 软件有一个孕育、诞生、成长、成熟、衰亡的生存过程。这个过程即为软件的生命周期（生存期） |
|  | 软件生命周期由软件定义、软件开发和软件维护三个时期组成。每个时期又进一步划分为若干个阶段。 |
| 定义时期 | 问题定义、可行性研究和需求分析。 |
| 开发时期 | 总体设计、详细设计、编码、单元测试、综合测试 |
| 维护时期 | 运行维护 |

|  |  |  |
| --- | --- | --- |
| 阶段 | 主要问题 | 结束标准 |
| 问题定义 | 问题是什么 | 关于规模和目标的报告书 |
| 可行性研究 | 有可行解码 | 系统的高层逻辑模型：数据流图DFD、成本/效益分析 |
| 需求分析 | 系统必须做什么 | 系统的逻辑模型：数据流图DFD、数据字典、算法描述 |
| 总体设计 | 概括 如何解决这个问题 | 可能的解法：系统流程图、成本/效益分析 推荐的系统结构：层次图/结构图 |
| 详细设计 | 怎样具体实现这个系统 | 编码规格说明：HIPO图（层序图+输入/处理/输出图） 或PDL图（过程程序语言） |
| 单元和编码测试 | 正确的程序模块 | 源程序清单、单元测试方案 结果 |
| 综合测试 | 符合要求的软件 | 综合测试方案、结果 ：集成测试、验收测试、完整一致的软件配置 |
| 维护 | 持久地满足客户需要 | 维护记录=四类维护：改成性维护-运行中发现错误、 适应性维护-适应变化软件环境、改善性维护-用户需求改进/扩充 预防性维护 |

软件过程

是为了获得高质量软件所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤。

软件过程定义了:

方法使用的顺序

要求交付的文档资料

为保证质量和适应变化所需要的管理

软件开发各个阶段完成的里程碑。

软件生存期模型是跨越整个生存期的系统开发、运作和维护所实施的全部过程、活动和任务的结构框架。  
软件生存期模型规定了把软件生命周期划分成那些阶段及各个阶段的执行顺序。

几种典型的软件过程模型（生命周期模型）

瀑布模型

顺序性、依赖性、保证质量，可强迫开发人员采用规范的方法，保证每一阶段产品的质量，但用户很难只通过文档了解产品

演化模型---------快速实现、不断迭代

快速还原型模型：快速建立其原型，让用户使用，通过实践来了解系统概貌，能加速软件开发，节约成本

增量模型：分模块

螺旋模型：风险控制，有利于软件的重用，减少过多测试和风险；但需要开发人员有丰富的风险评估能力

面向对象模型

喷泉模型：开发过程中没有明显间隙

软件工程过程

|  |  |
| --- | --- |
| 软件规格说明 | 规定软件的功能及其运行的限制 |
| 软件开发 | 产生满足规格说明的软件 |
| 软件确认 | 确认软件能够完成客户提出的需求 |
| 软件演进 | 为满足客户的变更要求，软件必须在使用的过程中演进 |

软件工程过程的特性：易理解性（可读性）、可见性、可支持性、可接受性、可靠性、健壮性、可维护型、速度

Chap2 可行性研究

|  |  |
| --- | --- |
| 目的 | 用最小的代价在尽可能短的时间内确定问题是否能够解决。问题是否值得去解决 |
| 任务 | 进一步分析和澄清问题定义 到出系统的逻辑模型 探索若干可供选择的主要方法 研究每种解法的可行性 |
| 根本任务 | 对以后的行动方针提出建议 |
| 时间 | 可行性研究的时间长短取决于工程的规模 |
| 内容 | 经济可行性：现有技术能否完成这个项目 经济可行性：系统经济效益能否超过其开发成本 操作可行性：系统的操作方式在这个用户组织内是否行得通 社会可行性（法律）：社会上或政治上的侵权、破坏或其他责任问题 |
| 研究过程1-8 | 复查系统规模和目标：我们做的和用户需求是否相同  研究目前正在使用的系统：分析现有系统完善和改进  导出新系统的高层逻辑模型：从物理系统->现有逻辑模型->参考现有系统设想目标逻辑模型->新物理系统  进一步定义问题：再次分析问题，1-4循环，知道提出的逻辑模型完全符合系统目标  导出和评价供选择的解法：从逻辑模型出发，根据现有技术、操作、经济确定解决方案  推荐行动方针：进行成本/效益分析，确定是否实行方案  草拟开发计划书：制定工程进度表，需求分析的详细进度表、成本估计  写文档提交审查：形成文档，交付评审，是否接受分析员的解决方案    1-4用户 1-8分析员 |
| 系统流程图 | 事 务  库 存 清 单 程 序  定 货  信 息  报 眚 生 成 私 ， 序  定 货 报 眚  阼 存 清  主 文 件 |
| 分层 | 面对复杂的系统时，最好采用分层的方法来进行描绘。 |

数据流图

描绘数据在软件中流动和被处理的逻辑过程，是系统逻辑功能的图形表示  
绘制时只需考虑系统必须完成的基本逻辑功能

存 存 
清 清 库 信 
单 单 存 息 
定 货 
报 表 
仓 管 
接 收 
更 新 库 
处 理 产 生 
采 购 员 
事 务 
存 清 单 
定 货 报 表 
定 货 
定 货 
D2: 定 货 信 息 

符号

|  |  |
| --- | --- |
| 数据流图的4种成分 | 源点或终点，处理，数据存储和数据流 数据的源点或终点：表示数据的外部来源和去处。它通常是系统之外的人员或组织，不受系统控制。 |
| 数据处理 | 是对数据进行的操作，它把流入的数据流转换为流出的数据流。  任何改变数据的操作都是处理 处理不一定是一个程序（人工、单程序、一系列程序） |
| 数据存储 | 是存储数据的工具，数据存储名应与它的内容一致 |
| 数据流 | 用带有名字的具有箭头的线段表示，表示流经的数据，箭头表示流向 |
|  | 数据存储和数据流都是数据，仅仅是所处的状态不同。数据存储是处于静止状态的数据，数据流是处于运动状态的数据 |
| 说明 | 数据流图的基本要点是描绘“做什么”，而不考虑“怎么做”  在数据流图中应该描绘所有可能的数据流向，而不应该描绘出现某个数据流的条件。  若数据的源点和终点相同，通常是再重复画一个同样的符号表示终点。  为避免引起误解，如果代表同一事物的同样符号在图中出现在n个地方，则在这个符号的一个角上画(n-1)条斜线做标记。  应该对数据处理和数据存储进行编号。    除了4种基本符号外，有时也会使用几种附加符号 |
| 画图步骤 | 从问题中提取4种成分 画出基本系统偶像 将基本模型细化，描绘系统的主要功能，得到功能级数据流图 进一步细化，直至不能分解  分解前后，输入、输出数据流必须相同 |
| 命名 |  |
| 用途 | 作为交流信息的工具 作为分析和设计的工具    数据流图只描述了系统的“分解”，它并没有表达出每个数据和加工的具体含义  这些信息需要在“数据词典” 中表达出来 |

数据字典

|  |  |
| --- | --- |
| 定义 | 对数据流图中包含的所有元素的定义的集合 |
| 作用 | 在软件分析和设计的过程中给人提供关于数据的描述信息 |
|  | 数据流图和数据字典共同构成系统的逻辑模型。 |
| 组成 | 数据流、数据流分量（数据元素）、数据存储、处理  书中的数据字典主要由对数据的定义组成 |
| 方法 | 数据字典定义数据的方法是对数据自顶向下的分解。  由数据元素组成数据的方式有4种类型：  顺序 即以确定次序连接两个或多个分量  选择 即从两个或多个可能的元素中选取一个  重复 即把指定的分量重复零次或多次  可选 即一个分量是可有可无的 |
| 实现 | 通常借助于CASE“结构化分析与设计工具”来实现 |
| 用途 | 作为分析阶段的工具  有助于改进分析员和用户、不同开发人员或不同开发小组之间的通信  数据字典中包括的每个数据元素的控制信息是很有价值的  数据字典是开发数据库的第一步，而且是很有价值的一步 |

名 字 ： 订 货 报 表 
别 名 ： 订 货 信 息 
描 述 ！ 每 天 一 次 送 给 采 购 员 的 需 要 订 货 的 零 
件 表 
定 义 ： 订 货 报 表 = 零 件 编 号 + 零 件 名 称 + 订 
货 数 量 + 目 前 价 格 + 主 要 
供 应 者 + 次 要 供 应 者 
位 置 ： 输 出 到 打 印 机 
名 字 ： 订 货 数 量 
别 名 ： 
描 述 ： 某 个 零 件 一 次 订 货 的 数 量 
定 义 ： 订 货 数 量 = 11 数 字 》 5 
位 置 ： 订 货 报 表 
订 货 信 息 
名 字 ： 零 件 编 号 
别 名 ： 
描 述 ： 唯 一 地 标 识 库 存 清 单 中 一 个 特 定 零 
件 的 关 谜 域 
定 义 ： 零 件 编 号 = 8 { 字 符 } 8 
位 置 ： 订 货 报 表 
订 货 信 息 
库 存 清 单 
事 务 

成本/效益分析

|  |  |
| --- | --- |
| 目的 | 从经济角度分析开发一个特定的新系统是否划算，从而帮助客户组织的负责人正确地作出是否投资于这项开发工程的决定。 |
| 成本估计 | 软件开发成本主要表现为人力消耗 |
| 估算技术 | 1. 代码行技术  2. 任务分解技术  3. 自动估计成本技术 |
| 分析方法 | 运行费用取决于系统的操作费用(操作员人数，工作时间，消耗的物资等)和维护费用。  系统的经济效益等于因使用新系统而增加的收入加上使用新系统可以节省的运行费用。   1. 货币的时间价值 2. 投资回收期 3. 纯收入 4. 投资回收率 |

Chap3 需求分析

基本任务: 系统必须做什么

结果：软件需求规格说明书

原则：

(1) 必须理解并描述问题的信息域，根据这条准则应该建立数据模型。

(2) 必须定义软件应完成的功能，这条准则要求建立功能模型。

(3) 必须描述作为外部事件结果的软件行为，这条准则要求建立行为模型。

(4) 必须对描述信息、功能和行为的模型进行分解，用层次的方式展示细节。

需求分析任务

1. 确定对系统的综合要求

功能需求、性能需求、可靠性和可用性需求、出错处理需求、接口需求、约束、逆向需求、将来可能提出的需求

1. 分析系统的数据要求

通常采用建立数据模型的方法，描述数据直接的逻辑关系

1. 导出系统的逻辑模型

用数据流图、实体-联系图、状态转换图、数据字典和主要的处理算法描述这个逻辑模型

1. 修正系统开发计划

根据对系统更深入的了解，准确估计系统的成本和进度，修正以前的开发计划

与用户沟通获取需求的方法

访谈 填写调查表

面向数据流自顶向下求精 结构化分析方法，把数据流和数据存储定义到元素级

简易的应用规格说明技术 倡导用户与开发者密切合作

快速建立软件模型 最准确、最有效、最强大的需求分析技术 快速，容易修改

分析建模

模型：对事务的一种无歧义的书面表述，由一组图形符号和组织这些符号的规则组成

需求分析过程应该建立的三种模型

数据模型 ---- 实体-联系图 ER图

功能模型---- 数据流图

行为模型----状态转换图

实体-联系图（ER图）

教 
职 称 
姓 名 
系 
性 别 
职 务 
学 号 
性 学 黑 
年 
级 
学 生 属 性 联 系 属 性 课 程 属 性 
姓 名 教 师 
教 工 号 教 
成 
绩 
N 
关 系 
课 程 
课 程 号 课 名 学 时 学 分 
图 3 ． 2 某 校 教 学 管 理 ER 图 

状态转换图p66

通过描绘系统的状态及引起系统状态转换的事件，来表示系统的行为

状态：任何可以被观察到的系统行为模式

只能有一个初态，0至多个终态

状态图可以是循环的，可也以是系统单程生命周期（需标注起始和终点状态）

不关心怎样启动

事件是在某个特定时刻发生的事情，从一个状态转换到另一个状态

初态用实心圆表示

终态用一对同心圆（内圆为实心圆）表示

中间状态用圆角矩形表示，可以用两条水平横线把它分成上、中、下3个部分，分别放置状态名（必选）、状态变量的名字和值（可选）、活动表（可选）

初 始 事 件 
状 态 丨 
状 态 变 量 ] 
活 动 表 1 
事 件 表 达 式 
状 态 2 
状 态 变 量 2 
活 动 表 2 
结 束 事 件 

例子

一个保险箱上装了一个复合锁，锁有三个位置，分别标记为1、2、3，转盘可向左(L)或向右(R)转动。这样，在任意时刻转盘都有6种可能的运动，即L、1R、2L、2R、3L和3R。保险箱的组合密码是1L、3R、2L，转盘的任何其他运动都将引起报警。试用状态转换图描述保险箱的行为。

保 险 箱 
初 始 态 
锁 定 
转 盘 的 
任 何 其 
他 移 动 
1 L 
3R 
A 
转 盘 的 
任 何 其 
他 移 动 
报 警 
2L 
B 
转 盘 的 
任 何 其 
他 移 动 
保 险 箱 
终 态 
解 锁 

に ぎ 窰 
0 朝 ま き 0 ま び 医 
圄 Y き 
に す 臂 
、 当 三 一 当 U 
0—1 三 三 一 
に マ 一 
号 感 当 1 拏 一 
に ト こ 、 。 p 

层次方框图

顶层为完整的数据结构

底层不能在分割，为实际的数据元素

培 训 
处 理 
机 
硬 件 
存 储 
器 
外 部 
设 备 
操 作 
系 统 
系 统 
软 件 
编 泽 
程 序 
产 品 
软 件 
应 用 
软 件 
软 件 
工 具 
软 件 
服 务 
服 务 
硬 件 
维 修 

Warnier图

树形图，描绘手段比层次方框图更丰富

可以清楚地描绘信息的逻辑结构

它可以表明一个（或一类）信息元素是重复出现的

也可以表示特定信息在某一类信息中是有条件地出现的

重复和条件约束是软件处理过程的基础，所以很容易把Warnier图转变成软件设计的工具。

1.在一个开括号内的所有名字都属于同一类信息

2.异或符号⊕表明一类信息或数据元素在一定条件下才出现，而且这个符号上、下方的

两个名字所代表的数据只能出现一个

3.在一个名字下方（或右侧）的圆括号中的数字指明这个名字代表的信息（或元素）在这个数据结构中重复出现的次数

例子：一种软件产品要么是系统软件要么是应用软件。系统软件中有P1种操作系统，P2种编译程序，此外还有软件工具。软件工具又可以进一步细分为P3种编辑程序、P4种测试驱动程序和P5种设备辅助工具。

操 作 系 统 （ PI) 
编 译 程 序 （ P2 ） 
系 统 软 件 
编 辑 程 序 （ P3 ） 
软 件 产 品 9 
软 件 工 具 测 试 驱 动 程 序 （ P4 ） 
应 用 软 件 
设 计 辅 助 工 具 （ P5 ） 

IPO图

IPO图是输入、处理、输出图的简称

左边框中列出有关的输入数据

中间框中列出主要的处理

右边框中列出产生的输出数据

处理框中列出处理的次序暗示了执行的顺序

粗大箭头指出数据通信的情况

旧 的 主 文 件 
事 务 文 件 
处 理 
上 校 验 
主 记 录 
上 校 验 
事 务 记 录 
3 ． 更 新 
主 记 录 
输 出 
有 效 的 
主 记 录 
有 效 的 
事 务 记 录 
史 新 后 的 
主 文 件 

名 称 ： 
戛 块 名 ： 
层 愤 调 用 块 ： 可 赁 处 理 
一 11 
售 管 理 
确 定 能 否 订 貨 
层 甲 块 ： 订 贳 处 理 
伴 名 ： 
斥 存 文 件 
入 数 冕 ： 订 单 订 貨 量 1 
相 应 賃 物 库 存 量 Y 
理 ： 
IF 卜 T 腫 （ 照 用 “ 可 处 畦 ” 
EISE （ 谰 甲 “ ” 赁 订 单 底 ） 
ENDIF 
、 计 人 ： 
日 期 
出 据 ： 
缺 货 订 留 
确 定 否 货 块 的 1?0 图 

验证软件需求的正确性

一致性。所有需求必须是一致的，任何一条需求不能和其他需求互相矛盾。

完整性。需求必须是完整的，规格说明书应该包括用户需要的每一个功能或性能。

现实性。指定的需求应该是用现有的硬件技术和软件技术基本上可以实现的。

有效性。必须证明需求是正确有效的，确实能解决用户面对的问题。

Chap5 总体设计

基本目的：概括地说，系统应该如何实现

任务：

划分出组成系统的物理元素----程序、文件、数据库、人工过程和文档

设计软件的结构，划分出程序的模块组成，模块间的相互关系

编写 总体/概要设计说明书

九个步骤

1. 设想供选择的方案

从系统逻辑模型出发，分析比较各种不同的物理实现方案

2. 选取合理的方案

列举若干（3个）方案：系统流程图，组成系统的物理元素清单，成本/收益分析，实现这个系统的进度计划

3. 推荐最佳方案

比较各种合理方案的利弊，推荐一个最佳 ，审查，接受

4. 功能分解

结构设计，确定由哪些模块组成系统，及其之间的关系，是总体设计阶段的任务

过程设计，设计确定每个模块的处理过程。过程设计是详细设计阶段的任务

5. 设计软件结构

将模块组织成良好的层次系统。层次图、结构图描绘

6. 设计数据库

在需求分析阶段确定的数据库上修改

7. 制定测试计划

8. 书写文档

9. 审查和复审

设计原理

模块化

C(x)定义问题的复杂度，E（x）解决问题的复杂度（可能考这个证明）

若 C（P1）>C（P2）

则 E（P1）>E（P2）

根据人类一般经验 有 C(P1+P2)>C(P1)+C(P2)

则 E(P1+P2)>E(P1)+E(P2)

说明：P1 P2 两个问题分开的复杂度要小于将其组合到一起的复杂度

抽象

现实世界一定事物、状态或过程之间总存在某些相似的方面（共性）把这些相似的方面总结概括起来，忽略之间的差异---抽象（暂时不考虑他们的细节）

软件工程过程的每一步都是对软件解法的抽象层次的一次精化

逐步求精

逐步求精定义为：“为了能集中精力解决主要问题而尽量推迟对问题细节的考虑。”

Miller法则：一个人在任何时候都只能把注意力集中在（7±2）个知识块上。

求精实质上是细化过程

信息隐蔽和局部化

信息隐藏原理指出：应该这样设计和确定模块，使得一个模块内包含的信息对于不需要这些信息的模块来说，是不能访问的。(访问权限)

局部化 ：是指把一些关系密切的软件元素物理地放得彼此靠近。

模块独立（高内聚 低耦合）

模块独立的概念是模块化、抽象、信息隐藏和局部化概念的直接结果

重要性：

有模块化的软件能比较容易的开发出来

独立的模块比较容易测试和维护

两个标准：

耦合：模块之间相互连接的紧密程度的度量

内聚：一个模块内部各个元素彼此结合的紧密程度的度量

七种耦合：非直接耦合、数据耦合、标记耦合、控制耦合、外部耦合、公共耦合、内容耦合（耦合性越来越高）

七种内聚：功能内聚、顺序内聚、通信内聚、过程内聚、时间内聚、逻辑

内聚、偶然内聚（内聚越来越低）

优先提高内聚，耦合自然降低

启发规则

改进软件结构提高模块独立性

模块规模应该适中

深度、宽度、扇出和扇入都应适当

深度：软件结构中控制的层数

宽度：软件同一层次上的模块总数的最大值

扇出：一个模块直接控制（调用）的模块总数（平均3-4,上限5-9）

扇入：有多少个上级模块直接调用它

设计的好的软件结构通常顶层扇出比较高，中层扇出较少，底层模块有高扇入

模块的作用域应该在控制域之内

力争降低模块接口的复杂程度

单出单入，避免内容耦合

模块功能可预测

层次图（区别于5.2）连线表示调用而非组成

正 文 加 
工 系 统 
输 人 
输 出 
添 加 
编 辑 
删 除 
加 标 题 
插 人 
存 储 
修 改 
检 索 
合 并 
编 目 录 
列 表 
格 式 化 

HIPO图（除了最顶层的方框之外，每个方框都加了编号）

正 文 加 
工 系 统 
输 人 
输 出 
添 加 
编 辑 
删 除 
加 标 题 
插 人 
存 储 
修 改 
检 索 
合 并 
编 目 录 格 式 化 
列 表 

结构图

方框代表模块

箭头代表调用关系

产 生 最 佳 解 
计 算 最 佳 解 
得 到 好 输 人 
编 辑 输 人 
输 出 结 果 
读 输 人 
结 果 格 式 化 
显 示 结 果 

面向数据流的设计方法：把信息流映射成软件结构，信息流的类型决定了映射的方法

目标：给出设计软件结构的一个系统化途径

思想：定义不同的“映射”，用数据流映射反映软件结构

变换分析 p104-114

事务分析

设计优化（都不会 自己看书吧）

Chap6 详细设计

根本目标：确定该怎样具体实现所要求的系统

任务：不是具体的编程，而是设计出程序的“蓝图”

详细设计的结果基本上决定了最终的程序代码的质量。

衡量程序的质量的因素有：

逻辑是否正确

性能是否满足要求

是否容易阅读和理解

三种基本的控制结构 顺序、选择、循环

结构化的程序：只有一个出口和入口，且只用三种基本控制结构，本质上是让程序代码跟容易被理解（尽可能的少用go to语句）

人机界面设计

四个问题

1. 系统响应时间-------长度和易变性

长度：太长---沮丧，太短----增加错误率

易变性：响应时间的偏差

1. 用户帮助设施-----集成（自带）、附加（后加的）两类
2. 出错信息处理--减少用户挫败感，提高交互系统的质量
3. 交互命令----既可以从菜单中选择，也能使用快捷键

用户界面设计是一个迭代的过程

主要依靠设计者的经验

设计指南

一般交互指南

信息交互指南

数据输入指南

过程设计的工具

三种：图形（程序流程图、盒图、PAD图）

表格（判定表）

过程设计语言

程序流程图（应该不考）

盒图

(1) 功能域(即，一个特定控制结构的作用域)明确，可以从盒图上一眼就看出来。

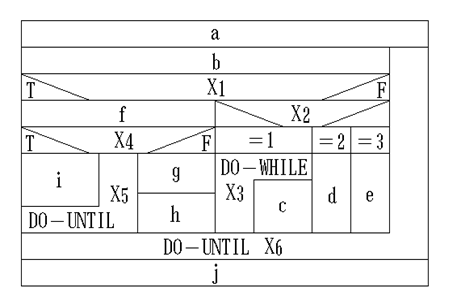
(2) 没有箭头，不可能任意转移控制。

(3) 很容易确定局部和全程数据的作用域。

(4) 很容易表现嵌套关系，也可以表示模块的层次结构。

第 一 个 任 务 
第 二 个 任 务 
第 三 个 任 务 
(a) 
循 环 条 件 
DO-WHI LE 
部 分 
F 条 件 T 
ELSE TH EN 
部 分 部 分 
CASE 条 件 
值 1 值 2 
CASE ] CASE 2 
部 分 部 分 
CASE 
部 分 
DO-UNTI L 
部 分 
A 
循 环 条 件 
(d) 
(e) 

DO —WHILE 
DO — UNT IL 
DO- UNTIL 



4 ． 由 伪 码 程 序 画 出 程 序 流 程 图 和 盒 图 。 
画 出 下 列 伪 码 程 序 的 程 序 流 程 图 和 盒 图 ， ， 
START 
q DOO 
0 
END DOW 
ELSEV 
BLOCK 
G 
END BLOCK 
END IFe 
STOP 
流 程 冬 
盒 图 

PAD图 p126

使用PAD符号设计出来的程序必然是结构化程序

UNTIL Х5 
UNTIL 

判定表

能够清晰地表示复杂的条件组合与应做的动作之间的对应关系。

优点：能够简洁而又无歧义地描述处理规则

缺点：并不适于作为一种通用的设计工具，清晰的表示顺序和避免重复不能兼得

假设某航空公司规定，乘客可以免费托运重量不超过30公斤的行李。

当行李重量超过30公斤时：

对头等舱的国内乘客超重部分每公斤收费4元；

对其他舱的国内乘客超重部分每公斤收费6元；

对外国乘客超重部分每公斤收费比国内乘客多一倍；

对残疾乘客超重部分每公斤收费比正常乘客少一半。

表 6 ． 1 用 判 定 表 表 示 计 算 行 李 费 的 算 法 
规 
1 
． 00 到 0 一 0 到 到 过 到 0 
． 0000000000 
． 0000000000 
00 到 000 到 00 到 
0000 一 000 到 00 
0 立 00 一 00 河 河 00 
0000000000 
0000000000 
内 等 疾 李 费 
乘 舱 乘 重 
客 
国 头 残 行 免 
（ W 一 30 ） × 2 
（ W 一 30 ） × 3 
（ W 一 30 ） × 4 
（ W 一 30 ） × 6 
（ W 一 30 ） × 8 
（ W 一 30 ） × 12 

判定树 p128

能清晰地表示复杂的条件组合与应做的动作之间的对应关系，易于掌握和使用

行 李 重 量 
伊 > 30kg 
行 李 费 
算 法 
行 李 重 量 
缈 《 30kg 
国 内 乘 客 
外 国 乘 客 
免 费 
残 疾 乘 客 
头 等 舱 
《 一 一 正 常 乘 客 
其 他 舱 「 一 一 一 残 疾 乘 客 
I— 正 常 乘 客 
《 一 一 残 疾 乘 客 
头 等 舱 1_ 正 常 乘 客 
残 疾 乘 客 
《 一 一 正 常 乘 客 
（ 一 30 ） × 2 
（ 一 30 ） × 4 
（ 一 30 ） × 3 
（ 一 30 ） × 6 
（ 一 30 ） × 4 
（ 一 30 ） × 8 
（ 一 30 ） × 6 
（ 一 30 ） × 12 

过程设计语言

用正文形式表示数据和处理过程的设计工具。

特点

关键字固定的语法

自然语言自由的语法

数据说明手段。有数据结构

模块定义和调用的技术，应该提供各种接口描述模式

面向数据结构的设计方法

Jackson方法（P130-137 自己看书吧 个人觉得不会考）

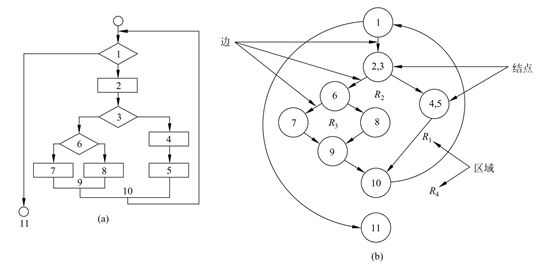
程序复杂成都的定量度量

McCabe方法：根据程序控制流的复杂程度定量度量程序的复杂程度，这样度量出的结果称为程序的环形复杂度

流图：弱化的流程图

边必须终结与一个结点（可能结点无意义）

若遇到符合条件，将其转化为若干简单条件



计算环形复杂度（三种方法）

(1) 流图中的区域数等于环形复杂度。4

(2) 流图G的环形复杂度V(G)=E-N+2,其中，E是流图中边的条数，N是结点数。11-9=4

(3) 流图G的环形复杂度V(G)=P+1，其中，P是流图中判定结点的数目。3+1=4

程序的环形复杂度取决于程序控制流的复杂程度，也即是取决于程序结构的复杂程度。

McCabe研究大量程序后发现，环形复杂度高的程序往往是最困难、最容易出问题的程序。

模块规模以V(G)≤10为宜，也就是说，V(G)=10是模块规模的一个更科学更精确的上限。

Halstead方法根据程序中运算符和操作数的总数来度量程序的复杂程度

令N1为程序中运算符出现的总次数，N2为操作数出现的总次数，程序长度N定义为： N=N1+N2

详细设计完成后，可以知道程序中不同运算符(包括关键字)的个数n1，不同操作数(变量和常数)的个数n2。Halstead给出预测程序长度的公式为：

H=n110yn1+n210g2n2 

预测程序中包含错误的个数的公式如下：

C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image025.png

Chap7 实现（编码和测试）

编码p146

测试目标

(1) 测试是为了发现程序中的错误而执行程序的过程；

(2) 好的测试方案是极可能发现迄今为止尚未发现的错误的测试方案；

(3) 成功的测试是发现了至今为止尚未发现的错误的测试

测试准则

（1） 所有测试都应该能追溯到用户需求。

（2） 应该远在测试开始之前就制定出测试计划。

（3） 把Pareto原理应用到软件测试中。80-20

（4） 应该从“小规模”测试开始，并逐步进行“大规模”测试。

（5） 穷举测试是不可能的。

（6） 为了达到最佳的测试效果，应该由独立的第三方从事测试工作。

测试方法

白盒测试-----结构测试

按照程序内部的逻辑测试程序，检测程序中的主要执行通路是否都能按照预定要求正确工作

黑盒测试----功能测试

完全不考虑程序的内部和处理过程，值检查程序功能是否能按照规格说明书正常使用

软件测试步骤

模块测试（白盒测试）

每个模块与其他模块没有相互依赖关系。是确保每个模块正确运行，又称为单元测试，在这个过程中发现的往往是编码和详细设计的错误

子系统测试

将单元模块放到一起，着重测试模块间的接口

系统测试（集成测试）

将子系统装配成完整的系统来测试，这个步骤中往往能发现 软件设计 中的错误

验收测试（确认测试）

在用户的参与下，用实际数据测试----->发现需求说明中的错误

平行运行

与旧系统同时运行一段时间

单元测试

测试重点：模块接口、局部数据结构、重要的执行通路、出错处理通路、边界条件（最重要）

代码审查

计算机测试 ：模块并不是一个独立的程序，因此必须为每个单元测试开发驱动程序和(或)存根程序。通常驱动程序也就是一个“主程序” ，调用该模块。存根程序代替被测试的模块所调用的模块。因此存根程序也可以称为“虚拟子程序”

集成测试

自顶向下（深度优先 宽度优先）

自底向上（不需要存根程序）

优点：不需要测试驱动程序，能够在测试阶段的早期实现并验证系统的主要功能。

缺点：需要存根程序，低层关键模块中的错误发现较晚。

而自底向上与之相反。

回归测试：用于保证其他操作引起的变化不会带来额外的错误

确认测试

Alpha测试----在受控环境中

Beta测试-----真实环境（不受控）

白盒测试技术（玻璃盒测试）

语句覆盖（每句话测试一遍）

判定覆盖（每一判定的每个分支至少执行一次）

条件覆盖（每个判别式应该取到所有结果）

判定/条件覆盖 ：同时满足判定覆盖和条件覆盖的要求

条件组合覆盖：求出判定中所有条件的各种可能组合值，每一可能的条件组合至少执行一次

黑盒测试技术

①功能不正确或遗漏了功能；

②界面错误；

③数据结构错误或外部数据库访问错误；

④性能错误；

⑤初始化和终止错误。

使用技术

1、等价划分

把程序的输入域划分成若干个数据类，据此导出测试用例。

2、边界值分析

处理边界情况时程序最容易发生错误。

3、错误推测

列举出程序中所有可能有的错误和容易发生错误的特殊情况

4、因果图法

因果图法主要检查各种输入条件的组合。

〔 a 〕 恒 等 0—0 El 
El 
Cl 
C2 
El 

例如，有一个处理单价为5角钱的饮料的自动售货机软件测试用例的设计。其规格说明如下：

若投入5角钱或1元钱的硬币，押下〖橙汁〗或〖啤酒〗的按钮，则相应的饮料就送出来。若售货机没有零钱找，则一个显示〖零钱找完〗的红灯亮，这时在投入1元硬币并押下按钮后，饮料不送出来而且1元硬币也退出来；若有零钱找，则显示〖零钱找完〗的红灯灭，在送出饮料的同时退还5角硬币

(1) 分析这一段说明，列出原因和结果  
原因: 1. 售货机有零钱找

2. 投入1元硬币

3. 投入5角硬币

4. 押下橙汁按钮

5. 押下啤酒按钮

建立中间结点，表示处理中间状态  
11. 投入1元硬币且押下饮料按钮  
12. 押下〖橙汁〗或〖啤酒〗的按钮  
13. 应当找5角零钱并且售货机有零钱找  
14. 钱已付清

结果： 21. 售货机〖零钱找完〗灯亮

22. 退还1元硬币

23. 退还5角硬币

24. 送出橙汁饮料

25. 送出啤酒饮料  
(2) 画出因果图。所有原因结点列在左边，所有结果结点列在右边。

(3) 由于 2 与 3 ，4 与 5 不能同时发生，分别加上约束条件E。

(4) 因果图

(5) 转换成判定表

售 货 机 有 零 钱 找 1 
投 入 1 元 硬 币 2 
E 
11 
投 入 5 角 硬 币 3 
该 找 5 角 
V 
押 下 按 钮 
押 下 橙 汁 按 钮 4 
押 下 啤 酒 按 钮 5 
可 找 5 角 
V 
21 售 货 机 " 零 钱 找 完 " 灯 亮 
退 还 1 元 硬 币 
3 找 回 5 角 硬 币 
送 出 橙 汁 饮 料 
2 
5 送 出 啤 酒 饮 料 

调试：在测试发现错误之后排除错误的过程。

两种结果：1.找到问题原因解决问题 2.找不到问题就猜想，直到验证正确并解决这个问题

调试方法：蛮干法、回溯法、原因排除法

软件可靠性：程序在给定的时间间隔内，按照规格说明书的规定成功地运行的概率。

软件可用性：程序在给定的时间点，按照规格说明书的规定，成功地运行的概率。

系统稳态可用性：Ass=Tup/(Tup+Tdown)

如果引入系统平均无故障时间MTTF和平均维修时间MTTR的概念

Ass=MTTF/(MTTF+MTTR)

在测试过程的早期阶段，由测试员甲和测试员乙分别测试同一个程序的两个副本，由另一名分析员分析他们的测试结果。用τ表示测试时间，假设

τ=0时错误总数为B0;

τ=τ1时测试员甲发现的错误数为B1;

τ=τ1时测试员乙发现的错误数为B2;

τ=τ1时两个测试员发现的相同错误数为bc。

假定测试员甲发现的错误是有标记的，则有

B0^ /B1=B2/bc

B0^ =B2/bc×B1

Chap8 维护

软件维护是软件生命周期的最后一个阶段，它处于系统投入生产性运行以后的时期中，因此不属于系统开发过程。

基本任务：保证软件在一个相当长的时期能够正常运行

软件维护需要的工作量非常大大型软件的维护成本高达开发成本的四倍左右

软件维护包括：

诊断和改正错误的过程：改正性维护

为了和变化了的环境适当地配合而进行的修改软件的活动：适应性维护

为了满足在使用软件的过程中用户的建议和改进意见而作的维护：完善性维护

为了给未来的改进奠定更好的基础而修改软件：预防性维护

非结构化维护：软件配置的唯一成分是程序代码

结构化维护：有一个完整的软件配置存在。能减少精力浪费并且能提高维护的总体质量。

评 价 设 计 
计 划 途 径 
修 改 设 计 
重 编 程 序 
雄 护 要 求 
代 码 
评 价 代 码 
重 编 程 序 
交 付 使 用 
结 构 化 维 护 与 非 结 构 化 维 护 的 对 比 

维护工作量模型

M＝ P＋ K \* 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是因缺乏好的设计和文档而导致复杂性的度量）

d是维护人员对软件的熟悉程度

维护过程

维护过程本质上是修改和压缩了的软件定义和开发过程。

1. 组织维护
2. 维护报告
3. 维护的事件流
4. 保存维护记录
5. 评价维护活动

软件的可维护性的决定因素

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

文档：影响软件可维护性的决定性因素

用户文档：描述功能和使用方法，并不关心功能怎样实现

系统文档：描述系统设计、实现和测试等各方面的内容

用户文档应包括5个方面：

功能描述

安装文档

使用手册

参考手册

操作员指南

系统文档：指从问题定义、需求说明到验收测试计划这样一系列和系统实现有关的文档。

软件逆向工程： 分析程序以便在比源代码更高的抽象层次上创建出程序的某种表示的过程。

Chap9 面向对象方法学引导

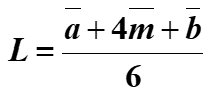
OO = 对象+类+派生+消息传递

Chap13 软件项目管理

管理：通过计划、组织和控制等一系列活动，合理地配置和使用各种资源，以达到既定目标的过程（贯穿与软件的整个生命周期）

代码行技术

由多名有经验的软件工程师分别做出估计。每个人都估计程序的最小规模(a)、最大规模(b)和最可能的规模(m)，分别算出这3种规模的平均值、和之后，再用下式计算程序规模的估计值



优点：容易计算、有大量参考文献和数据。缺点：源程序只是软件的一部分，由此度量规模不合理，不同语言不一样，不适合非过程性语言

功能点技术

1. 信息域特性

功能点技术定义了信息域的5个特性：

输入项数(Inp)：用户向软件输入的项数，这些输入给软件提供面向应用的数据。

输出项数(Out)：软件向用户输出的项数，它们向用户提供面向应用的信息，

查询数(Inq)：查询即是一次联机输入，它导致软件以联机输出方式产生某种即时响应。

主文件数(Maf)：逻辑主文件的数目。

外部接口数(Inf)：机器可读的全部接口的数量，用这些接口把信息传送给另一个系统。

2. 估算功能点的步骤

(1) 计算未调整的功能点数UFP

UFP=a1×Inp+a2×Out+a3×Inq+a4×Maf+a5×Inf

杂 级 别 
简 单 
复 杂 
特 性 系 数 
输 入 系 数 al 
输 出 系 数 “ 2 
查 询 系 数 “ 3 
文 件 系 数 “ 4 
接 凵 系 数 “ 5 

(2) 计算技术复杂性因子TCF

14 
i=l 

TCF = 0.65 + 0.01 × DI

因为DI的值在0~70之间，所以TCF的值在0.65~1.35之间。

表 13 ． 2 技 术 因 素 
序 号 
技 术 因 素 
数 据 通 信 
分 布 式 数 据 处 理 
性 能 标 准 
高 负 荷 的 硬 件 
高 处 理 率 
联 机 数 据 输 入 
终 端 用 户 效 率 
联 机 更 新 
复 杂 的 计 算 
可 重 用 性 
安 装 方 便 
操 作 方 便 
可 移 植 性 
可 维 护 性 

(3) 计算功能点数FP

FP = UFP × TCF

功能点技术优点：与所用的编程语言无关，比代码行技术更合理。

功能点技术缺点：在判断信息域特性复杂级别和技术因素的影响程度时主观因素较大，对经验依赖性较强。

工作量估算

静态单变量模型

E = A + B × (ev) C 单位 人月

ev是估算变量（KLOC或FP）

Gantt图

优点：

Gantt图能很形象地描绘任务分解情况，以及每个子任务(作业)的开始和结束时间。

具有直观简明和容易掌握、容易绘制的优点。

缺点：

不能显式地描绘各项作业彼此间的依赖关系；

进度计划的关键部分不明确，难于判定哪些部分应当是主攻和主控的对象；

计划中有潜力的部分及潜力的大小不明确，往往造成潜力的浪费。

例子：旧木板房刷漆工程(15名工人，工具各5把)

表 13 · 5 各 道 工 序 估 计 需 用 的 时 间 （ 小 时 ） 
工 序 
墙 壁 
1 或 3 
2 或 4 
刮 旧 漆 
2 
4 
刷 新 漆 
3 
6 
清 理 
1 
2 

2 4 6 8 10 12 
14 
16 
18 
20 
22 
24 
刮 旧 漆 
刷 新 漆 
清 理 
勿 第 一 面 墙 
乙 第 二 面 墙 
勿 第 三 面 墙 
囫 第 四 面 墙 

网络工程图

事 件 号 
最 早 时 刻 
EET 
持 续 时 间 
（ 机 动 时 间 ） 
最 迟 时 刻 
LET 

( 3 ) 
( 3 ) 
( 0 ) 
12 ( 0 ) 
( 5 ) 
12 
( 1 ) 
20 
( 11 ) 
12 
18 

如果小组内有n个成员，则可能的通信信道共有n(n-1)/2条。

软件质量

定义强调了下述的3个要点：

软件需求是度量软件质量的基础，与需求不一致就是质量不高。

指定的开发标准定义了一组指导软件开发的准则，如果没有遵守这些准则，几乎肯定会导致软件质量不高。

通常，有一组没有显式描述的隐含需求。如果软件满足明确描述的需求，但却不满足隐含的需求，那么软件的质量仍然是值得怀疑的。

影响软件质量的主要因素，是从管理角度对软件质量的度量。可以把这些质量因素分成3组，分别反映用户在使用软件产品时的3种不同倾向或观点。这3种倾向是：产品运行、产品修改和产品转移。

可 理 解 性 （ 我 能 理 解 它 吗 ？ 
可 维 修 性 （ 我 能 修 复 它 吗 ？ 
灵 活 性 
（ 我 能 改 变 它 吗 ？ 
可 测 试 性 （ 我 能 测 试 它 吗 ？ 
效 率 
风 险 
可 移 植 性 （ 我 能 在 另 一 台 机 器 上 使 用 它 吗 ？ ） 
可 再 用 性 （ 我 能 再 用 它 的 某 些 部 分 吗 9 ） 
互 运 行 性 （ 我 能 把 它 和 另 一 个 系 统 结 合 吗 ？ ） 
产 品 运 行 
正 确 性 （ 它 按 我 的 需 要 工 作 吗 ？ ） 
健 壮 性 （ 对 意 外 环 境 它 能 适 当 地 响 应 吗 ？ ） 
（ 完 成 预 定 功 能 时 它 需 要 的 计 算 机 资 源 多 吗 ？ ） 
完 整 性 （ 它 是 安 全 的 吗 ？ ） 
可 用 性 （ 我 能 使 用 它 吗 ？ ） 
（ 能 按 预 定 计 划 完 成 它 吗 ？ 〕 