東華大學
DONGHUA UNIVERSITY

第八章 软件维护

CST
计算机科学与技术学院

主要内容

8.1 基本概念

8.2 软件可维护性

8.3 软件维护策略

8.4 软件维护过程

8.5 软件再工程

8.6 小结

习题

8.1 基本概念

软件维护的定义

- 软件维护就是在软件已经交付使用之后，为了改正错误或满足新的需要而修改软件的过程。
- **ANSI / IEEE（1980'S）：“软件维护是指软件成品提供使用后，为了修改差错、改善功能和性能、适应环境变化而进行的软件修正。”**

8.1.1 维护类型

- 改正性维护(Corrective Maintenance)
- 预防性维护(Preventive Maintenance)
- 适应性维护(Adaptive Maintenance)
- 完善性维护(Perfective Maintenance)

  - 注：
    - 后二者属增强性维护(Maintenance Enhancement)
    - 不属于软件纠正的软件变更。

8.1.1 维护类型

- 改正性维护
  - 在任何大型程序的使用期间，用户必然会发现程序错误，并且把他们遇到的问题报告给维护人员。把诊断和改正错误的过程称为改正性维护。
  - GB/T20157-2006/ISO/IEC14764:1999：
    - 软件产品交付后执行的反应性修改，以纠正发现的问题
  - 其主要任务是完成软件潜在错误的改正。
  - 软件测试通过选取少量的、高效的测试数据进行模拟使用，尽可能地发现软件的错误，有利于提高软件的可靠性，但不能从根本上完全杜绝软件系统中潜在的错误，这类错误的诊断和更正属于软件改正性维护的范畴。

8.1.1 维护类型

- 适应性维护
  - 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。
  - GB/T20157-2006/ISO/IEC14764:1999：
    - 在交付后执行的软件产品的修改，以保持这个软件产品可以在已变更或正在变更的环境中使用。
  - 注:
    - 适应性维护提供必要的改进，以适应软件产品必须运行于其中的环境的变更。为了与不断变更的环境保持同步应作出相应的变更，例如，操作系统可能升级并且可以作某些变更，以适应新的运行系统。

8.1.1 维护类型

- 适应性维护（续）
  - 对软件来说，环境变化源于以下几个方面：
    - 外部环境（新的硬、软件配置）
      - 硬件和操作系统更新。
      - 系统运行环境的变化。如由主机方式变为客户／服务器方式，由客户／服务器方式变为**Web**方式。
    - 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）
    - 开发环境的升级

8.1.1 维护类型

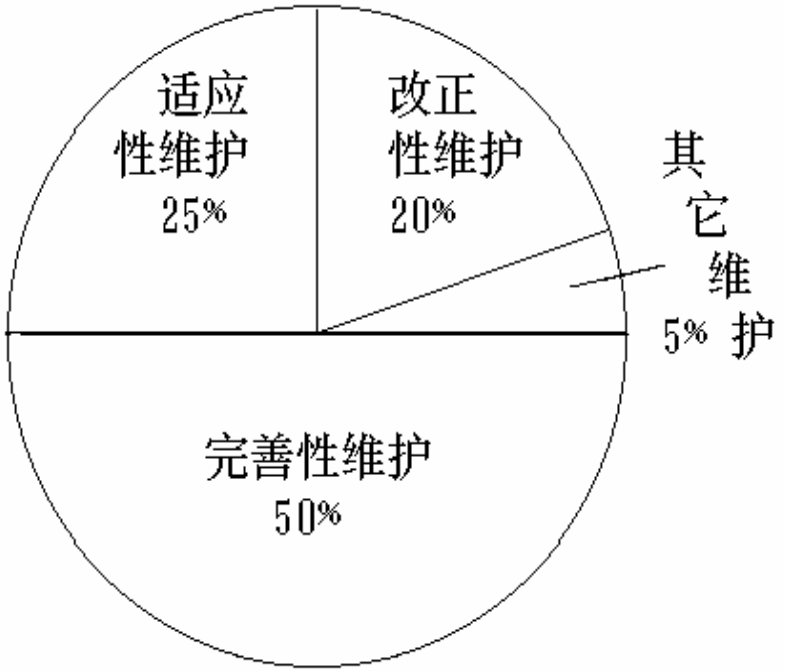
- 完善性维护
  - 完善性维护指为扩充系统的功能和改善系统性能而进行的修改，一般包括增加或修改功能，提高系统的安全性、处理能力等任务。
  - GB/T20157-2006/ISO/IEC14764:1999：
    - 软件产品交付后为改进性能或维护性所作的修改。
  - 注：
    - 完善性维护给用户提供增强性的程序文档改进和重编码，以改进软件性能、维护性或其他软件属性。

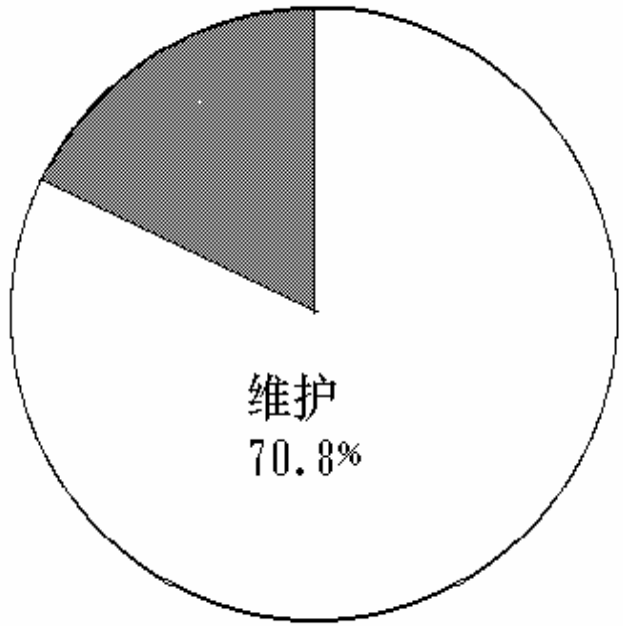
8.1.1 维护类型

预防性维护

- 预防性维护是为减少或避免以后可能需要的前三类维护而对软件配置进行的工作，通过再结构化、再标准化等系统优化方法，提高系统的可维护性，对文档进行维护，对数据进行重组。
- **GB/T20157-2006/ISO/IEC14764:1999：**
  - 软件产品交付后的修改，用来检测和纠正软件产品中的潜在故障，使其不致成为有效故障。
- 预防性维护有较强的前瞻性，而且做好预防性维护工作能降低或避免因为维护工作不充分、不及时导致软件系统瘫痪带来的灾难性后果。

8.1.1 维护类型





三类维护占维护在软件生存期总维护比例所占比例

8.1.2 软件维护的内容

程序维护

- 程序维护是指根据使用的要求，对程序进行全部或部分修改。程序维护要修改用户需求的概念模型和详细设计，并对源代码进行重新修改，所以工作量较大且容易对原有功能产生破坏，这是程序维护中不可忽视的一个问题。另外，根据需要进行修改以后，必须书写修改设计报告。修改设计报告必须和源代码同时维护，只有与程序完全一致的修改设计报告才是真正有价值的文档。

8.1.2 软件维护的内容

- 文件备份及修复
  - 文件备份及修复属于数据维护的范畴，包括安装与转换新的数据库出现异常或攫出时的补救和维护工作。
- 查杀病毒
  - 降低病毒危害最有效的两个方法：缩短数据备份周期和加快杀毒软件的升级频率。
- 系统优化
  - 系统优化是从系统全局出发，协调系统内部各组成单元之间的连接关系，使整个系统按预定目标在整体上达到最优。

8.1.3 软件维护的特点

- 软件维护的特点
  - 软件维护耗时费力
  - 软件维护的代价昂贵
  - 远程维护是现代软件维护的新途径
  - 软件复用技术简化了软件维护
  - 结构化维护与非结构化维护差异明显

8.1.3 软件维护的特点

- 结构化维护起点是评价设计文档，维护步骤如下：
  - 确定软件结构特点。
  - 性能特点分析。
  - 接口持点分析。
  - 估量改动带来的影响。
  - 计划实施途径。
  - 修改设计并仔细复查。
  - 编写相应的源代码。
  - 进行回归测试。
  - 交付使用。
- 非结构化维护的起点是评价程序代码。步骤如下：
  - 分析用户需求。
  - 代码评价。
  - 评价反馈。
  - 重编程序。
  - 复查。
  - 交付使用。

8.2 软件可维护性

- 定义
  - 软件可维护性是指纠正软件系统出现的错误和缺陷，以及为满足新的要求进行修改、扩充或压缩的容易程度。
- 维护性作为一种软件质量特性，影响到软件发布供使用后变更软件的便捷性和简易性。

8.2.1 影响因素

- 四个主要特性：
  - 可理解性
  - 可测试性
  - 可修改性
  - 可移植性

8.2.1 影响因素

- 可理解性
  - 软件的可理解性是指对软件的结构、接口、功能、内部过程理解的难易程度。
  - 程序模块化的结构特性，详细一致的设计文档，结构化设计的模块化，源代码选取结构化的语言描述，采取良好的设计风格，简明直接都对软件可理解性起决定性的作用。
- 一个可理解的程序应具备以下一些特性：
  - 模块化（模块结构良好，高内聚，松耦合）
  - 详细的设计文档
  - 结构化设计
  - 程序内部的文档
  - 良好的高级程序设计语言

8.2.1 影响因素

- 可测试性
  - 软件可测试性是指诊断和测试出软件错误和预期的结果的难易程度。
  - 一个可测试的程序应当是可理解的，可靠的，简单的。

8.2.1 影响因素

- 可修改性
  - 是指软件容易修改的程度，这里的修改主要针对源代码或有关文档。
  - 软件的可修改性和软件设计原则直接有关。可修改性好的软件在维护时出现连带错误的概率相对较小。
- 一个可修改的程序应当是可理解的、通用的、灵活的、简单的。
  - 通用性是指程序适用于各种功能变化而无需修改。
  - 灵活性是指能够容易地对程序进行修改。

8.2.1 影响因素

- 可移植性
  - 可移植性指软件的部分程序和某些模块转移到另一个计算机环境中去并能发挥功能的可能性。
  - 可移植性强的软件不依赖于具体的计算机环境，可以在不同的计算机和操作系统中正常工作。
- 一个可移植的程序应具有结构良好、灵活、不依赖于某一具体计算机或操作系统的性能。

8.2.1 影响因素

- 下列各项影响维护性，宜在选择程序设计语言时加以考虑：
  - 语言可移植性
  - 语言易读性；
  - 语言稳定性；
  - 自带文档；
  - 对降低程序清晰度的程序设计“窍门”的容限；
  - 程序结构化的可能性；
  - 生产新的发布版本的简易性；
  - 数据结构化的可能性；
  - 编译程序及其他工具的可用性；
  - 编译程序及其他工具的稳定性；
  - 在编译和运行期间测试的可能性；
  - 辅助生产、调试、配置管理的软件工程和软件测试环境的可用性，以及可靠性和质量需求满意度；
  - 各种开发工具的生存能力。

GB/T20157-2006

ISO/IEC14764:1999:

8.2.1 影响因素

- 软件文档对可维护性的意义
  - 文档是影响软件可维护性的最重要因素，有时甚至比可执行代码更为重要。
  - 文档可分为用户文档和系统文档两大类。不管是哪一类文档都必须和源代码同时维护，只有与程序完全一致的文档才是真正有价值的文档。

8.2.2 开发过程中的具体活动

- 软件需求分析阶段
  - 软件规范宜详尽地、无歧义地描述软件的维护性需求。
  - 下面各项影响维护性，要予以考虑：
    - 功能，特别是可选功能的标识和定义；
    - 数据的准确性和逻辑性；
    - 接口(机器和用户)，特别是将有的接口；
    - 性能需求，包括纠正和补充的性能要求的影响；
    - 受计划的环境影响的需求；
    - 需求粒度，它影响可追踪性的难易程度；
    - 强调文档编制及其依从性的软件质量保证计划。

8.2.2 开发过程中的具体活动

概要设计阶段

- 影响维护性的这项开发过程活动的主要特征是程序结构的选择、分解产生的实体以及贯穿这些实体的数据流。
- 与其他活动一样，重要的是运用程序设计团队的数据处理知识，因为这样做能揭示采用已证明其可信的现有程序或库的可能性。
- 由若干自顶向下的分析构成的模块化设计和足够的文档(必要时很容易补充)，是持续实现维护性要求的两个主要特征。

8.2.2 开发过程中的具体活动

- 软件详细设计阶段
  - 软件维护性将通过在该项活动中引人前面所述的质量特性得到改进。

8.2.2 开发过程中的具体活动

- 软件编码和测试阶段
  - 软件维护性通过文档质量的升级加以改进。
  - 质量文档应提供有助于执行维护过程的信息。利用质量文档改进维护性的建议包括:
    - 确保易读性;
    - 避免非结构化代码;
    - 考虑语言本身的弱点，排除典型的陷阱;
    - 在详细设计中检测差错;
    - 使用有助于差错追踪的技术

8.2.2 开发过程中的具体活动

- 软件合格性测试阶段
  - 这项活动确保每个软件需求的实现都进行符合性测试(GB/T 8566)。有关质量的软件需求在这项活动期间测试。
  - 保存软件开发期间所用的测试用例，用于修改后的回归测试。
  - 为了在开发期间避免重复相同错误，要保留项目的开发历史供维护使用。

8.2.2 开发过程中的具体活动

软件移交阶段

- 软件移交是一个受控且需协调的活动序列，软件开发由最初开发机构转移到软件维护机构。
- 如果维护职责从一个组织转移到另一个组织，要制定移交计划。此计划涉及：
  - — 硬件、软件、数据及经验由开发者移交给维护者；
  - — 维护者为实现软件维护策略所需要的任务(例如，人员配备、培训、安装、再现维护问题)。

8.2.2 开发过程中的具体活动

- 文档编制
  - 维护者往往面临只有很少文档甚至没有文档。如果没有文档，维护者要建立所需文档。文档创建是完善性维护的一部分。
  - 当面临这种形势时，维护者宜进行下列维护准备。
    - 了解问题的领域(应用类型)。
    - 学习软件产品的结构和组成。
    - 确定软件产品做什么。评审规格说明(若可用)，评审整个结构，分析调用树，阅读代码，向其他维护者提供口头说明，并给代码补加注释。
    - 定位低优先级的修改请求或问题报告。
  - 按上述指南实施时，维护者应编制软件产品文档。必要时，更新或创建文档(如规格说明、程序员维护手册、用户手册及安装指南)。
  - 维护环境中存在各种影响文档的创建和更新的因素，例如:源代码访问、代码分析工具的可用性，运行软件产品以确定性能的能力、以及软件测试环境的可用性。

8.3 软件维护策略

- 包括下列各项内容:
  - 维护概念
  - 维护计划
  - 资源分析

8.3.1 维护概念

- 确定维护概念是制订软件维护策略的第一步。
- 维护概念涉及：
  - 软件维护范围；
  - 过程的剪裁；
  - 指定维护提供者；
  - 维护成本估算。

8.3.1 维护概念

范围

- 范围与维护者将如何响应有关。要确定维护者的支持程度。预算上的约束往往限定维护的范围。
- 维护范围涉及:
  - 拟进行的维护的类型;
  - 拟维护的文档的级别;
  - 响应度;
  - 拟提供的培训级别;
  - 交付支持;
  - 前台支持。

8.3.1 维护概念

- 过程剪裁
  - 维护概念涉及软件交付后的维护任务。
  - 不同的机构在维护期间可能执行不同的任务。宜早作尝试以标识这些机构并记人维护概念文档。
  - 维护概念也要反映将采用的维护过程。

8.3.1 维护概念

- 指定维护提供者
  - 指定由谁提供维护是一个重要议题，宜早处理并记入维护概念文档。这对内部维护工作同样适用。
  - 对外包第三方协议的维护工作，维护概念要注明外包的维护。
  - 制约维护者指定的基本因素有多种，包括：
    - 软件产品的寿命；
    - 长期成本；
    - 启动成本；
    - 空间的可用性；
    - 资格；
    - 可用性；
    - 进度安排；
    - 领域知识。

8.3.1 维护概念

维护成本估算

- 要估算维护成本。成本是维护范围的函数。
- 涉及成本的附加因素是:
  - 到用户处的差旅费;
  - 对维护者以及用户的培训费;
  - 软件工程环境和软件测试环境的成本和年度维护费;
  - 薪水和津贴之类的人员成本.
- 建立维护概念时，要根据有限的可用数据估算成本。随着开发工作的推进，估算要进一步细化历史度量数据应用作估算维护成本的输人。

8.3.2 维护计划

- 维护概念确定时，要立即开始维护活动和任务的策划，拟订出维护计划。
- 维护计划在软件开发期间由维护者制订，宜包含用户如何提出更改软件产品的请求
- 维护计划应包含：
  - 为何需要维护；
  - 由谁做什么工作；
  - 所参与的每个人的角色和职责是什么；
  - 工作如何执行；
  - 可用于维护的资源是什么；
  - 维护在何处执行；
  - 维护何时开始。

8.3.2 维护计划

- 维护计划指南
  - a) 导引
    - 1) 描述拟支持的系统;
    - 2) 标识软件的初始状态;
    - 3) 描述为何需要支持;
    - 4) 标识维护者和(或)支持组织;
    - 5) 描述顾客与供方间达成的协议。
  - b) 维护概念
    - 1) 描述此概念;
    - 2) 描述对系统的支持级别;
    - 3) 确定支持期;
    - 4) 剪裁维护过程。
  - c) 机构和维护活动
    - 1) 交付前维护者的角色和职责
      - I) 过程实施;
      - II) 建立基础设施;
      - III) 建立培训过程;
      - IV) 建立维护过程。
    - 2) 交付后维护者的角色和职责
      - I) 过程实施;
      - II) 问题和修改分析;
      - III) 修改实施;
      - IV) 维护评审或验收;
      - V) 迁移;
      - VI) 退役;
      - VII) 问题解决(包括前台);
      - VIII) 培训人员(维护者和用户)
      - IX) 改进过程。
    - 3) 用户角色
      - I)验收测试;
      - II)与其他组织的接口。

8.3.2 维护计划

维护计划指南（续）

- d) 资 源
  - 1) 人员
  - 2) 软件：确定支持系统所需的软件。
  - 3) 硬件： 确定支持系统所需硬件 。
  - 4) 设施：确定设施需求 。
  - 5) 文档
    - 1) 软件质量计划 ;
    - II) 项目管理计划 ;
    - III) 配置管理计划 ;
    - IV) 开发文档 ;
    - V) 维护手册 ;
    - VI) 验证计划 ;
    - VII) 确认计划 ;
    - VIII) 测试计划、测试规程及测试报告;
    - IX) 培训计划 ;
    - X) 用户手册 。
  - 6) 数据
  - 7) 其他资源需求
- e) 过 程(如何开展工作)
  - 1) 维护者过程 (给出过程综述，不用详细说明维护计划中的整个过程);
  - 2) 经剪裁的过程。
- f) 培 训
  - 确定维护者和用户的培训需求。
- g) 维护记录和报告
  - 1) 请求帮助 、修改请求或问题报告的清单;
  - 2) 请求状态分类 ;
  - 3) 请求的优先级;
  - 4) 对维护活动拟收集的度量数据。

8.3.3 资源分析

- 通常由需方在供方(开发方>协助下确定软件维护的资源需求。人员、环境和财政资源均宜加以讨论。
  - 人力资源
  - 环境资源
    - 软件开发和维护是专业化活动，需要单独的专用系统。建议软件工程环境和软件测试环境分开。
    - 维护者要协助需方策划维护环境。
  - 财政资源
    - 为了提供有效的维护支持，维护者需要做出预算，其中涉及:
      - 工资、培训(每人每年2-3周)、软件许可证的年维护费、差旅费、技术出版物、工程和测试环境的硬、软件、工程和测试环境的硬、软件的升级。

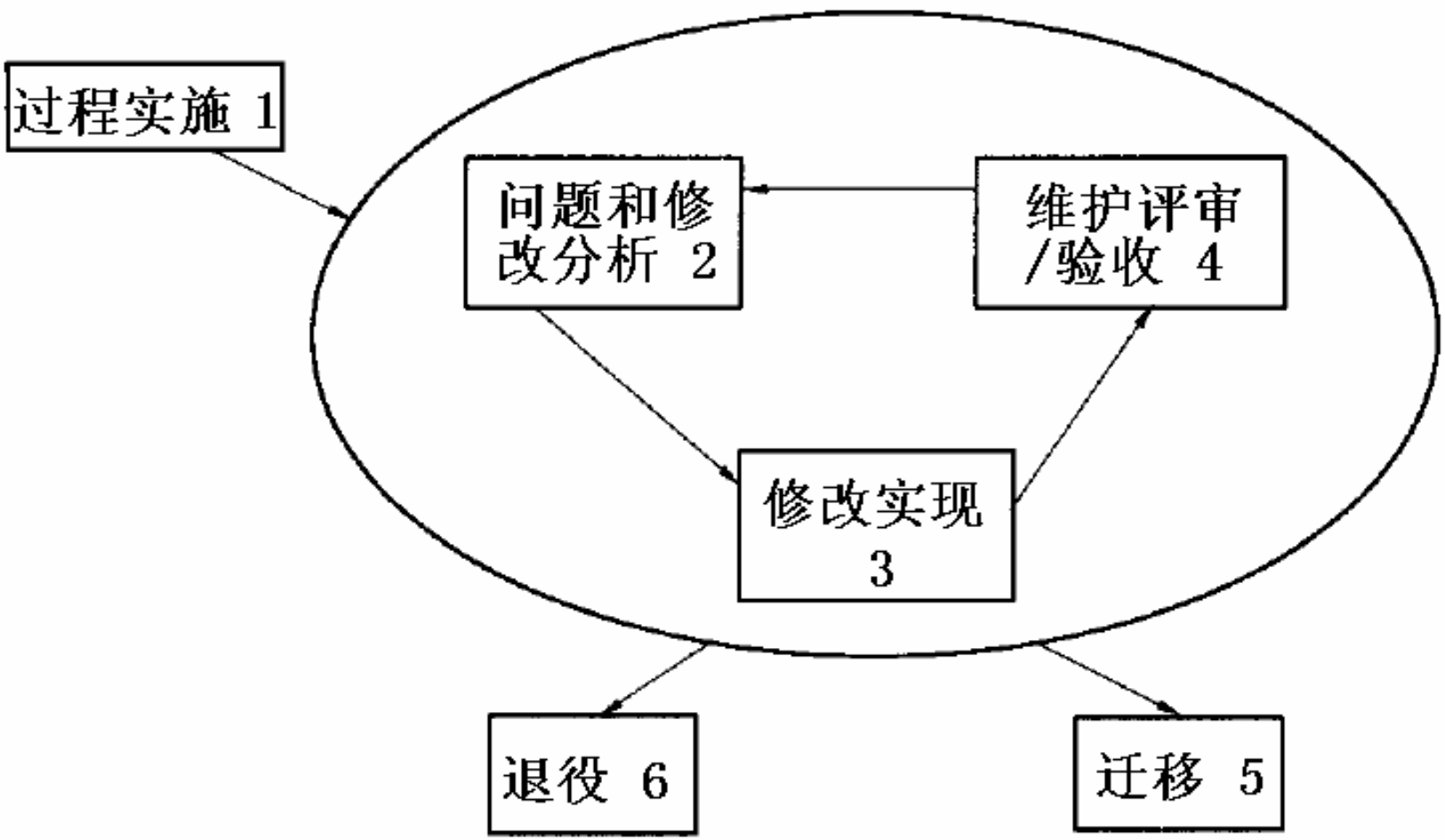
8.4 软件维护过程

- 维护过程包含为修改现行软件产品同时保持其完整性所必需的活动和任务。这些活动和任务是维护者的责任。
- 一旦该过程启动，应立即制订维护计划和规程并且分配维护专用资源。

8.4.1 维护过程的活动（国标）

- 组成维护过程的活动有（**GB/T20157-2006/ISO/IEC14764:1999**）：
  - **a)** 过程实施;
  - **b)** 问题和修改分析;
  - **c)** 修改实现;
  - **d)** 维护评审/验收;
  - **e)** 迁移;
  - **f)** 退役。

8.4.1 维护过程的活动（国标）



8.4.1 维护过程的活动（国标）

一、过程实施

- 在过程实施期间，维护者建立维护过程期间应执行的计划和规程。
- （一）输入
  - 旧基线;
  - 系统文档;
  - 修改请求或问题报告。
- （二）任务
  - （1）制维护计划和规程;
  - （2）建立修改请求/问题报告规程;
  - （3）实施配置管理。

8.4.1 维护过程的活动（国标）

- （1）维护计划和规程
  - 维护者应为实施维护过程的活动和任务制订并执行计划和规程，并形成文档。
  - 维护计划应包含所使用的系统维护策略文档，而维护规程应给出更详细的关于如何实际完成维护的方法。

8.4.1 维护过程的活动（国标）

（1）维护计划和规程

为制订有效的维护计划和规程，维护者应执行下列任务：

- a) 协助需方提出维护概念；
- b) 协助需方确定维护范围；
- c) 协助需方分析维护组织的替代方案；
- d) 确保书面指定软件产品维护者；
- e) 进行资源分析；
- f) 估算维护成本；
- g) 进行系统的维护性评估；
- h) 确定移交需求；
- i) 确定移交里程碑；
- J) 确定应使用的维护过程；
- k) 以运行规程的形式编制维护过程文档。

8.4.1 维护过程的活动（国标）

- （2）修改请求/问题报告规程
  - 维护者应建立接收、记录、追踪问题报告、用户修改请求以及向用户提供反馈的规程。
  - 无论何时遇到问题，都应记录并进入问题解决过程。

8.4.1 维护过程的活动（国标）

（2）修改请求/问题报告规程

维护者应执行下列任务:

- a) 为修改请求/问题报告制订标识编号方案;
- b) 为修改请求/问题报告制订分类和排列优先顺序的方案;
- c) 制订趋势分析规程;
- d) 确定运行员提交修改请求/问题报告规程;
- e) 确定如何向用户提供初始反馈;
- f) 确定如何为用户提供变通办法;
- 9) 确定数据如何录人状态统计数据库;
- h) 确 定向用户提供何种后续反馈。

8.4.1 维护过程的活动（国标）

- （3）实施配置管理
  - 略

8.4.1 维护过程的活动（国标）

- 一、过程实施
  - （三）输出
    - 维护计划;
    - 维护规程;
    - 问题解决规程;
    - 用户反馈计划;
    - 移交计划;
    - 配置管理计划。

所有的输出均置于配置管理之下。

8.4.1 维护过程的活动（国标）

- 二、问题和修改分析
  - 在修改系统前，维护者要分析修改请求/问题报告，以确定其对组织、现行系统和接口系统的影响，应提出可能的解决方案建议并形成文档，并且求得批准实施期望的解决方案。
  - （一）输入
    - 修改请求/问题报告;
    - 基线;
    - 软件存放库;
    - 系统文档:
      - 配置状态信息;
      - 功能需求;
      - 接口需求;
      - 项目策划数据;
      - 过程实施活动的输出。

8.4.1 维护过程的活动（国标）

- 二、问题和修改分析
  - （二）任务
    - 维护者应就下列各项分析问题报告或修改请求对组织、现行系统和接口系统的影响。
      - a) 类型：例如:纠正、改进、预防或对新环境的适应;
      - b) 范围：例如:修改规模、涉及的费用、修改时机;
      - c) 关键性：例如，对性能、安全、保密的影响。

8.4.1 维护过程的活动（国标）

二、问题和修改分析

（二）任务

为确保所要求的修改请求/问题报告可行，维护者宜执行以下任务：

- （1）分析修改请求/问题报告；
- （2）复现或验证问题；
- （3）提出修改实施意见；
- （4）编制修改请求/问题报告、结果和实施意见的文档；
- （5）求得批准所选择的修改意见。

8.4.1 维护过程的活动（国标）

- （1）修改请求/问题报告分析
  - a) 确定维护者是否为实现变更申请适当配备了人员;
  - b) 确定项目是否为实现变更申请做了适当的预算;
  - c) 确定是否有足够的可用资源，这种修改是否影响推进中的或预定的项目(对于问题报告可能不需要) ;
  - d) 确定要考虑的运行问题。
    - 例如，对系统接口需求、系统的预期有用生存期、运行优先级别、安全 性 和 保 密 性 等 的预期有哪些变更，如果不变更，对保密性有什么影响?(对于问题报告可能不 需 要) ;

8.4.1 维护过程的活动（国标）

- （1）修改请求/问题报告分析
  - e) 确定安全性和保密性含义(对于问题报告可能不需要);
  - f) 确定短时期成本和长时期成本(对于问题报告可能不需要);
  - g) 确定修改的利益价值;
  - h) 确定对进度的影响;
  - i) 确定所要求的测试和评价的级别;
  - j) 确定实现更改的估算管理成本(对于问题报告可能不需要)。

8.4.1 维护过程的活动（国标）

（2）验证

- 维护者应重现或验证问题；
- 任务：
  - 为确保请求的问题报告有效，维护者要通过执行下列任务来重现或验证问题：
    - a) 制订验证问题的测试策略；
    - b) 从配置管理中提取受影响的软件版本；
    - c) 安装受影响的版本；
    - d) 运行测试以验证问题，最好用受影响的数据的拷贝；
    - e) 编制测试结果文档。
  - 若由于某些原因，例如数据的机密性，问题不能重现，则应检查其他项，如组织规则、方针和文档。
  - 对适应性或完善性维护不要求执行验证任务。

8.4.1 维护过程的活动（国标）

- （3）选项
  - 在分析的基础上，维护者应制订实施修改的方案。
  - 维护者应执行下列任务:
    - 对修改请求/问题报告赋予优先级别;
    - 确定对问题是否有变通办法。如果有，提供给运行者或用户;
      - 这一步骤对适应性或完善性维护不需要
    - 规定修改确认要求;
    - 估算修改的规模和范围;
    - 提出进行修改的至少三个可供选择的方案;
    - 确定这些可选方案对系统硬件的影响;
    - 针对每个可选方案进行风险分析。

8.4.1 维护过程的活动（国标）

- （4）文档
  - 维护者应将问题/修改请求、分析结果和实施方案形成文档；
  - 任务：
    - **a) 验证所有对应的分析和项目文档是否已更新。如果没有文档，则编制文档；**
    - **b) 评审建议的测试策略和进度表的准确性；**
    - **c) 评审资源估算的准确性；**
    - **d) 更新状态统计数据库；**
    - **e) 提出处置建议，指出是否应该批准修改请求/问题报告。**

8.4.1 维护过程的活动（国标）

- （5）批准
  - 维护者应按合同规定使选定的修改方案得到批准；
  - 任务：
    - **a) 由相应的配置管理组提供针对批准的分析结果；**
    - **b) 参与有关修改的讨论；**
    - **c) 一经批准，立即更新修改请求状态；**
    - **d) 一经批准，若请求是增强(改进)性的，立即更新需求。**

8.4.1 维护过程的活动（国标）

- 二、问题和修改分析
  - （三）输出
    - 影响分析;
      - 对间题或新需求的陈述; 问题或需求评价; 所要求的维护类型分类; 初始优先级别; 验证数据(用于纠正性修改); 修改现行系统所要求的初始资源估算。
    - 推荐的可选方案;
    - 批准的修改;
    - 已更新的文档。
      - 测试策略;
      - 更新的测试文档，包括测试计划、测试规程和测试报告;
      - 软件开发文件夹;
      - 更新的需求。

8.4.1 维护过程的活动（国标）

三、修改实施

（一）输入

- 基线：
  - 系统体系结构定义；
  - 修改请求记录；
  - 源代码。
- 批准的修改请求/问题报告；
- 批准的修改文档：
  - 影响分析报告；
  - “问题和修改分析”活动的输出。

8.4.1 维护过程的活动（国标）

三、修改实施

（二）任务

（1）分析

- a) 确定现行系统中拟更改的元素;
- b) 确定受修改影响的接口元素;
- c) 确定拟更新的文档;
- d) 更新软件开发文件夹。

一旦修改请求/问题报告获准，维护者应实施分析并确定需修改的文档、软件单元和版本。

8.4.1 维护过程的活动（国标）

- （2）开发过程
  - 维护者应进入开发过程以实施修改。
  - 开发过程的需求补充如下：
    - a) 应规定测试和评价系统中已修改的与未修改的部分〔软件单元、部件和配置项)的准则，并形成文档；
    - b) 应确保新的和已修改的需求完整与正确地实现。同时确保原来的、未修改的需求不受影响。测试结果 应形成文档。

8.4.1 维护过程的活动（国标）

三、修改实施

（三）输出

- 更新的测试计划和规程;
- 更新的文档;
  - 更新的修改记录;
  - 详细分析报告;
  - 更新的需求;
  - 更新的测试计划、测试规程和测试报告;
  - 更新的培训资料。
- 修改的源代码;
- 测试报告;
- 度量。

8.4.1 维护过程的活动（国标）

- 四、维护评审和(或)验收
  - 这项活动确保对系统的修改是正确的，并且这些修改是使用正确的方法按批准的标准完成的。
  - （一）输入
    - 修改的软件;
    - 修改侧试结果

8.4.1 维护过程的活动（国标）

- （二）任务
  - （1）评审
    - 维护者应与授权修改的组织一起实施评审以确定已修改的系统的完整性
    - 应执行下列任务:
      - a) 从需求到设计，到编码追踪修改请求/问题报告;
      - b) 验证代码的可测试性;
      - c) 验证编码标准是否得到遵循;
      - d) 验证只对必要的软件部件作了修改，
      - e) 验证新软件部件集成的正确性;
      - f) 检查文档确保其已予更新;
      - g) 执行测试;
      - h) 拟制测试报告。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （2）批准
    - 维护者应按合同规定使修改满意完成，得到批准；
    - 如果实施维护时没有协议，也应得到批准。
    - 宜执行下列任务：
      - a) 通过质量保证生存周期过程获得批准；
      - b) 验证此过程得到遵循；
      - c) 实施功能和物理配置审核。

8.4.1 维护过程的活动（国标）

- 四、维护评审和(或)验收
  - （一）输出：
    - 新基线，它吸收了接受的修改；
    - 拒绝的修改；
    - 验收报告；
    - 审核和评审报告；
    - 软件合格性测试报告。

8.4.1 维护过程的活动（国标）

五、迁移

- 在系统的生存周期期间，可能应修改系统，以便其在不同的环境中运行。为了将某个系统迁移到某个新环境，维护者需要确定完成迁移所需的活动，然后考虑实现迁移所要求的步骤并且形成文档。
- （一）输入
  - 旧环境；
  - 新环境；
  - 旧基线；
  - 新基线。

8.4.1 维护过程的活动（国标）

- （二）任务
  - 遵循GB/T 8566-2001
  - 制订迁移计划
  - 通告此迁移的用户
  - 提供培训
  - 通告完成情况
  - 评估新环境的影响以及归档数据。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （1）迁移准备
    - 如果一个系统或软件产品(包括数据)从一个老的运行环境迁移到一个新的运行环境，应确保在迁移过程中任何软件或产生修改的数据遵循标准。
    - 宜执行下列任务:
      - a) 确定附加的或修改的所有软件产品或数据;
      - b) 验证这些任务遵守GB/T 8566-2001.

8.4.1 维护过程的活动（国标）

- （二）任务
  - （2）迁移计划
    - 应制订一个迁移计划并实施和形成文档。
    - 计划应包括下列各项:
      - a) 需求的分析和迁移的定义;
      - b) 迁移工具的开发;
      - c) 软件产品和数据的变换;
      - d) 迁移的执行;
      - e) 迁移的验证;
      - f) 未来对老环境的支持

8.4.1 维护过程的活动（国标）

- （二）任务
  - （2）迁移计划
    - 作为此项任务的一部分，维护者应执行下列任务:
      - a) 分析迁移需求;
      - b) 确定迁移软件产品的影响;
      - c) 拟订迁移进度表;
      - d) 为运行后评审确定数据收集需求;
      - e) 确定迁移工作，并形成文档;
      - f) 确定和缓解风险;
      - g) 确定需要的迁移工具;
      - h) 确定对旧环境的支持内容;

8.4.1 维护过程的活动（国标）

- （二）任务
  - （2）迁移计划
    - 作为此项任务的一部分，维护者应执行下列任务(续)
      - i)开发和(或)获取迁移工具;
      - j)为软件产品和数据的转化，对软件产品和数据进行渐进式分解;
      - k)确定软件产品和数据转化的优先顺序;
      - l) 转化软件产品和数据;
      - m)迁移软件产品和数据到新环境;
      - n)执行并行运行;
      - o)通过测试验证迁移;
      - p)提供对旧环境的支持。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （3）意图通告
    - 一旦维护者完成了迁移计划，应将迁移计划和活动通知用户，通知应包括:
      - a) 为何不再支持老环境的说明;
      - b) 对新环境及其生效日期的描述;
      - c) 一旦对旧环境的支持取消，应描述其他可用的支持方案，如果有的话。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （3）意图通告
    - 维护者还要向用户提供计划、规程和进度表。作为此项任务的一部分，维护者宜执行下列任务:
      - a) 确定所有的将受影响的位置;
      - b) 处理现场反馈;
      - c) 识别现场特殊问题;
      - d) 公布进度。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （4）实施运行和培训
    - 旧环境和新环境可以并行进行工作，以便平稳迁移到新环境。维护者可执行下列有关并行运行的任务:
      - **a)** 现场调查;
      - **b)** 安装设备;
      - **c)** 安装软件;
      - **d)** 进行初步测试以确保成功安装硬、软件;
      - **e)** 以并行方式与旧系统一起在某个运行负载下运行软件;
      - **f)** 收集新旧产品的数据;
      - **g)** 执行数据整理和分析。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （4）实施运行和培训
    - 在此期间，应按合同规定提供必要的培训,任务:
      - a) 确定迁移培训需求;
      - b) 安排迁移培训进度;
      - c) 进行迁移培训评审;
      - d) 更新培训计划。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （5）完成通告
    - 应进行运行后评审以评定对新环境变更的影响。
    - 当预定的迁移到来时，应通知所有相关部门，所有相关的老环境的文档、日志和编码应放入档案中；
    - 作为此项任务的一部分，维护者宜执行下列任务：
      - **a)** 公布迁移进度的变更情况；
      - **b)** 把现场特殊问题以及将如何解决形成文档；
      - **c)** 旧软件和数据归档；
      - **d)** 移走旧设备。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （6）运行后评审
    - 应进行运行后评审以评定对新环境变更的影响。评审结果应送到各相应部门，以便了解信息、进行指导和采取行动。
    - 作为此项任务的一部分，维护者宜执行下列任务：
      - a) 评审以并行方式运行系统的结果；
      - b) 确定潜在的风险范围；
      - c) 识别现场特殊问题；
      - d) 把经验总结形成文档；
      - e) 生成并提交影响分析报告。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （7）数据归档
    - a) 存储旧软件和数据;
    - b) 拷贝旧软件和数据;
    - c) 在安全位置存贮媒体。

8.4.1 维护过程的活动（国标）

六、软件退役

- 软件产品一旦结束使用生存周期，必须退役。进行分析以帮助作出软件产品退役决定。
- 这种分析通常基于经济考虑，可以包含在退役计划中。分析中应确定下列做法从成本考虑是否合适:
  - — 保留过时的技术;
  - — 通过开发新软件产品转向新技术;
  - — 开发新软件产品以达到模块化;
  - — 开发新软件产品以便利于维护;
  - — 开发新软件产品以达到标准化;
  - — 开 发新软件产品以有利于销售商无关性。
- 可以用新软件产品替换旧的软件产品，但是在某些情况下不会替换。为了使某软件产品退役，维护者要确定完成退役所要求的行动，然后提出实现退役所要求的步骤并形成文档。应考虑对退役软件产品存储的数据的访问。

8.4.1 维护过程的活动（国标）

- 六、软件退役
  - （一）输入
    - 迁移计划;
    - 迁移工具;
    - 意图通告;
    - 迁移的软件产品;
    - 完成通告;
    - 归档的数据。

8.4.1 维护过程的活动（国标）

六、软件退役

（二）任务

维护者通过下列工作实施软件退役:

- 制定退役计划
- 通知用户退役
- 提供培训
- 告完成情况
- 归档数据。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （1）制定退役计划
    - 应制订退役计划以撤消运行和维护组织的积极支持，并形成文档。策划活动应包括使用者。计划应叙述下列项目。计划应予以执行。
      - a) 一定时期之后，全部或局部支持终止；
      - b) 软件产品及其有关文档的归档；
      - c) 任何未来后续支持事项的职责；
      - d) 如适合，转换为新的软件产品；
      - e) 归档数据副本的可访问性。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （1）制定退役计划
    - 作为此项任务的一部分，维护者宜执行下列任务:
      - a) 分析退役需求;
      - b) 确定软件产品退役的影响;
      - c) 确定替换的软件产品(如果有);
      - d) 拟订软件产品退役进度表;
      - e) 确定所有遗留支持的责任;
      - f) 确定退役工作并编制文档。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （2）通知用户退役
    - 用户应得到退役计划和活动的通知。通知应包括下述内容:
      - a) 替代或升级及其生效日期的说明;
      - b) 为什么该软件产品不再得到支持的说明;
      - c) 一旦失去支持时，其他可用支持方案的说明。
    - 作为此项任务的一部分，维护者宜执行下列任务:
      - a) 确定将受影响的所有位置;
      - b) 识别现场特殊问题;
      - c) 公布进度;
      - d) 处理现场反馈。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （3）实行并行运行和培训
    - 退役软件和新软件应并行工作、以便平稳过渡到新系统。在此期间，应按合同规定提供用户培训。
    - 作为此项任务的一部分，维护者宜执行下列任务:
      - **a)** 现场调查;
      - **b)** 安装设备;
      - **c)** 安装软件产品;
      - **d)** 执行初步测试以确保成功安装硬、软件;
      - **e)** 以并行方式在某运行负载下与旧系统一起运行软件产品;
      - **f)** 收集新、旧产品的数据;
      - **g)** 进行数据整理和分析。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （4）完成通告
    - 当完成某退役进度时，应通告所有有关方面。适当时，应归档所有相应的开发文档、日志和代码。
    - 作为此项任务的一部分，维护者宜执行下列任务:
      - **a)** 公布退役进度变更情况;
      - **b)** 编制有关现场特殊问题以及如何解决的文档;
      - **c)** 归档旧软件和数据;
      - **d)** 移走旧设备。

8.4.1 维护过程的活动（国标）

- （二）任务
  - （5）数据归档
    - 退役软件使用的或相关的数据，其中关于数据的保护和审核根据合同应是可访问的。
    - 最好将归档媒体更新为**CD-ROM**和其他数字盘产品，以便检索。作为此项任务的一部分，维护者宜执行下列任务:
      - **a)** 把“任务执行通告”期间获得的旧软件和数据存储起来;
      - **b)** 为“任务执行通告”期间获得的旧软件和数据做出拷贝;
      - **c)** 在 安全地方存贮媒体。

8.4.1 维护过程的活动（国标）

- （三）输出
  - — 退役计划;
  - — 意图通告;
  - — 退役结果;
  - — 经培训的人;
  - — 退役的软件产品;
  - — 完成通告;
  - — 归档的基线。

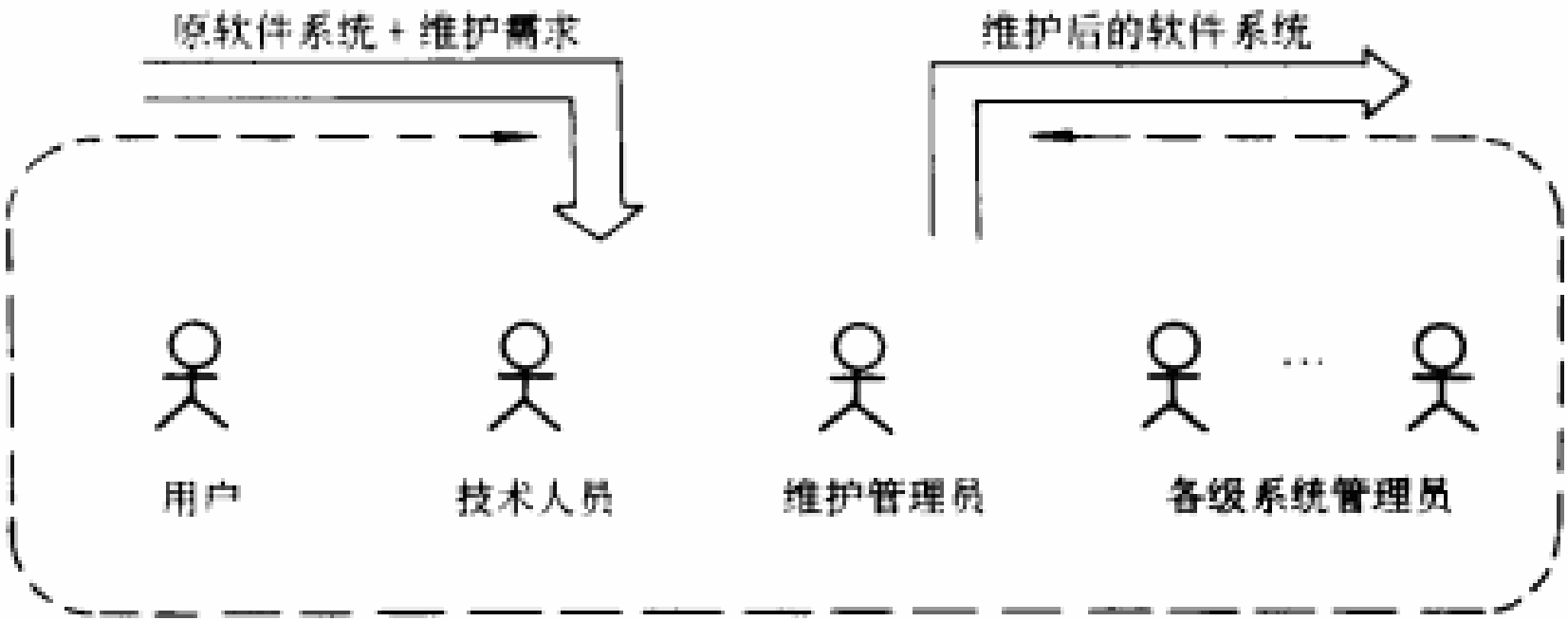
8.4.2 一般软件维护步骤

1. 建立维护组织

- 软件维护组织不是一个正式的常设组织，往往是根据维护要求的提出而临时成立，它是一个为实现软件维护而成立的人员团体。
- 软件维护组织通常由用户、技术员、维护管理员、高级系统管理员和局部系统管理员组成，

8.4.2 一般软件维护步骤

- 1. 建立维护组织（续）



8.4.2 一般软件维护步骤

2. 编写维护报告

- 让用户填写用户维护申请表，获得用户提出的维护需求。
  - 对适应性和完善性的维护需求，用户应该给出尽可能简洁的、准确的需求描述。
  - 改正性维护往往是由于遇到软件运行的错误，则用户必须详尽地描述错误发生的原因、导致的相关后果，以及错误发生时软件运行的环境。
- 根据用户给出的维护需求描述(表格或说明书)，软件维护组织要给软件修改报告。
  - 软件修改报告的内容比软件维护申请报告的批复要详细，要求给出维护工作量需求、维护要求的性质和与修改有关的数据。
  - 软件修改报告交系统变化授权人审查后生效，软件组织可以进行下一步详细的软件修改计划的制定和软件修改的实施。

8.4.2 一般软件维护步骤

软件维护申请表

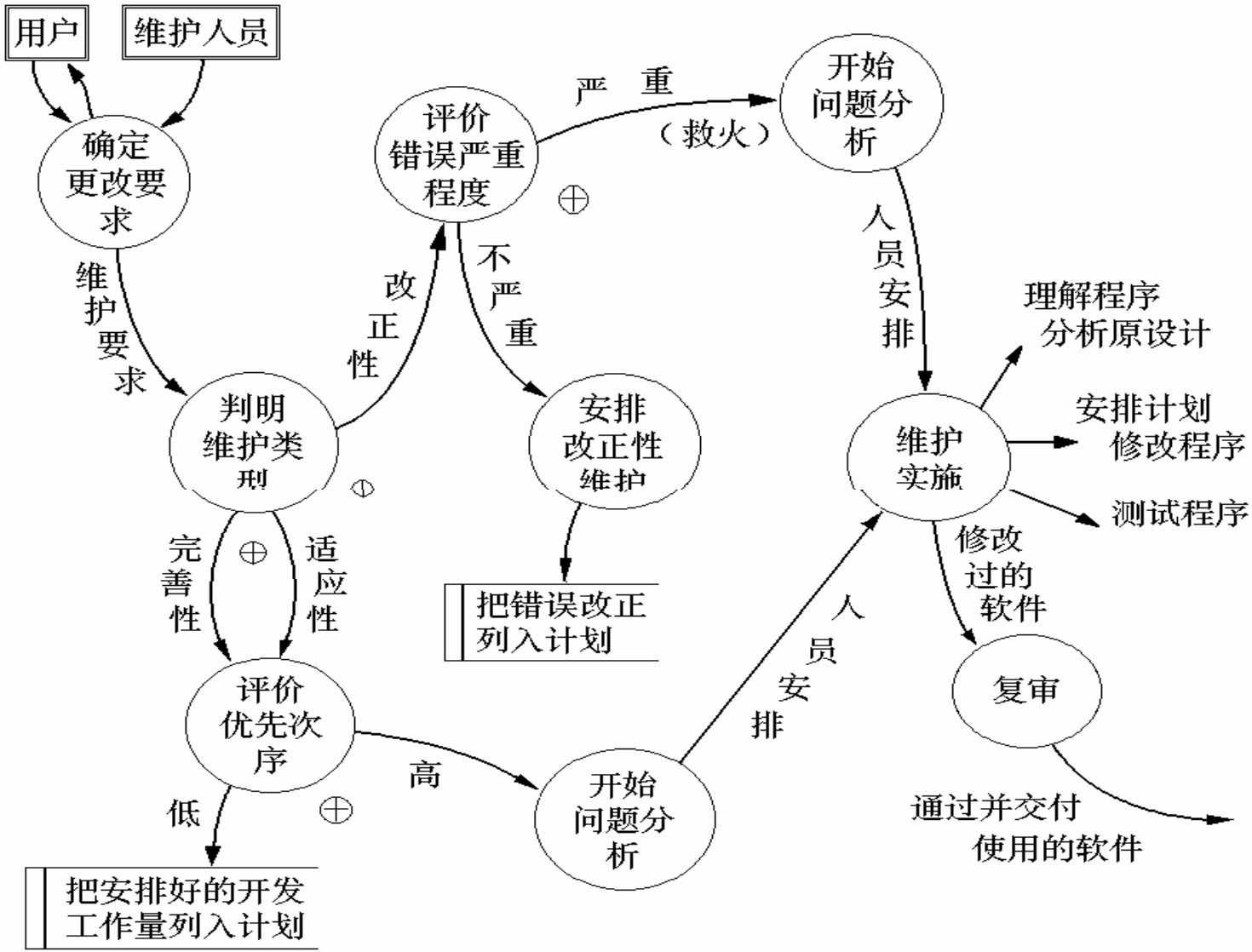
申请表编号：　　申请日期：　年　月　日

项目编号			项目名称	
维护类别	软件维护	改正性		问题说明：
		完善性		
		适应性		
		预防性		维修要求：
	硬件维护	系统设备		
		外围设备		
维修优先级				申请评价结论：
维护方式		远程 / 现场		
申请人				评价负责人：　　评价时间：

8.4.2 一般软件维护步骤

- 3、进行软件修改
  - 1)由系统管理员提出软件修改请求报告。
  - 2)由有关领导审批请求报告。
  - 3)源程序清单存档。
  - 4)手续完备后，实施软件的修改。
  - 5)软件修改后形成新的文档资料。
  - 6)发出软件修改和使用变更通知。
  - 7)进行软件修改后的试运行。
  - 8)根据运行情况作总结调整并修改文档资料
  - 9)发出软件修改后正式运行通知。
  - 10)软件做新的备份并同定稿的文档资料一起存档。这里的文档主要应包括以下内容护的提请人、审批人、维护人的姓名、维护时间、修改原因、修改的内容、修改后的现状。

8.4.2 一般软件维护步骤



8.4.2 一般软件维护步骤

- 4、保存维护记录
  - 1)程序标识。
  - 2)源语句数。
  - 3)机器指令条数。
  - 4)使用的程序设计语言。
  - 5)程序安装的日期。
  - 6)安装程序后运行的次数。
  - 7)安装程序后失效的次数。
  - 8)程序变动的层次和标识。

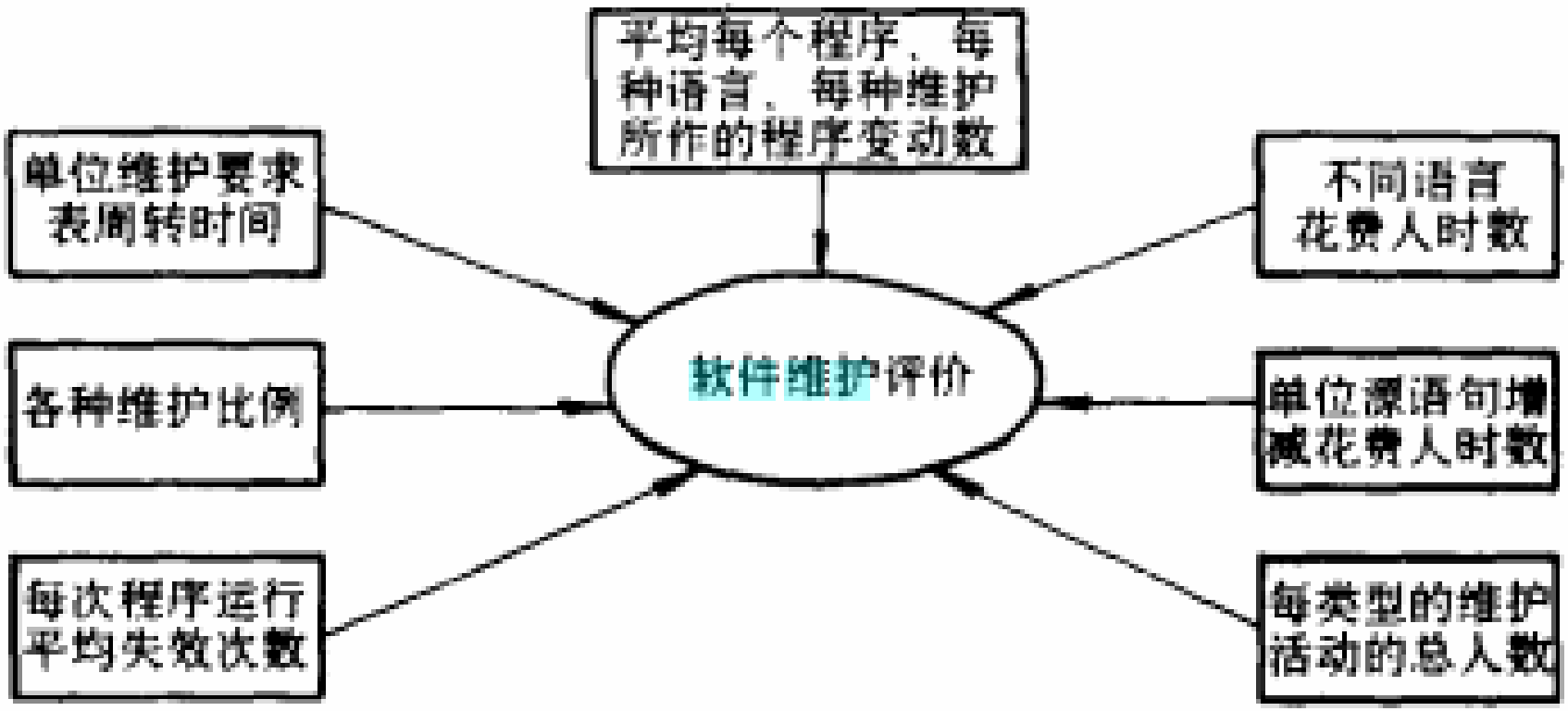
8.4.2 一般软件维护步骤

4、保存维护记录（续）

- 9)因为程序变动而增加的源语句数。
- 10)因为程序变动而删除的源语句数。
- 11)每个改动耗费的人时数。
- 12)程序改动的日期。
- 13)软件工程师的名字。
- 14)维护要求表的识别。
- 15)维护类型。
- 16)维护开始和完成的日期。
- 17)累计用于维护的人时数。
- 18)与完成的维护相联系的纯效益。

8.4.2 一般软件维护步骤

- 评价维护结果
  - 维护结果的评价建立在相关数据的基础之上，所以，维护的档案记录要做好。



8.5 软件维护的副作用

- 编码修改的副作用
  - 即使是对一个语句的简单修改，例如粗心地将逗号错改为句号，就曾经使美国宇航局的阿波罗宇宙飞船的控制软件失效，几乎造成悲剧性的后果。虽然并非所有的副作用都会造成如此严重的后果，但是修改编码的确可能引入错误，从而导致各种各样的问题发生。
  - 每个程序的编码修改都可能引入新错误、产生副作用。尤其在删除或修改子程序(函数、过程)、标识符、语句标号、逻辑运算符、边界测试条件，以及将设计修改转换成编码修改、为改善执行性能所作的修改和修改文件打开与关闭的格式等情况下，更加容易引入各种各样的错误。
  - 编码修改副作用的范围很广，从回归测试验证期间的检错、纠错到软件运行期间的故障，都有可能源于编码修改而引入的错误。

8.5 软件维护的副作用

编码修改的副作用（续）

- 经常发生修改代码产生副作用的可能有：
  - 1．删除和修改一个子程序
    - 必须查明与此子程序相关的模块和语句，修改删除之前做到有把握时再修改，修改后进行回归测试。
  - 2．删除和改变一个语句标号 （处理同1 ）。
  - 3．删除和改变一个标识符（ 处理同1）。
  - 4．为改进执行性能所作的修改（ 处理同1，附加性能测试）。

8.5 软件维护的副作用

编码修改的副作用（续）

- 经常发生修改代码产生副作用的可能有(续)：
  - **5. 改变文件的打开或关闭**
    - 反复检查处理的文件状态和内容的正确性。
  - **6. 改变运算逻辑符**
    - 反复检查计算结果和算法正确性。
  - **7. 把设计的修改翻译成代码的修改**
    - 修改后，应立即撤消这个修改，并作回归测试。
  - **8. 对边界测试所作的修改**

8.5 软件维护的副作用

数据修改的剧作用

- 所谓数据修改副作用是指，维护期间对数据结构本身或其个别域进行修改时，很有可能使得软件设计与数据结构不匹配，从而导致软件错误。 .
- 最容易产生数据修改副作用的修改有：增加或减少数组的大小，修改全局数据，重新初始化控制标志或指针，重新排列输入输出程序或过程(函数或子程序)中自变量的次序等。
- 为了减少数据修改的副作用，应当设计完善的文档资料来反映修改状态。这样的文档资料应该详细描述数据结构，并且提供将数据域、记录、文件及其它结构与软件模块联系起来的交叉对照表。

8.5 软件维护的副作用

数据修改的剧作用（续）

- 经常发生的如下修改产生副作用的问题：
  1. 重新定义局部或全局常量，
  2. 重新定义记录或文件格式；
  3. 增大或减小一个数组、高阶数据结构的大小，全程数据；
  4. 重新韧始化控制标记或指针，
  5. 重新排列I／O或子程序的自变量。
- 为避免修改数据的副作用，一是要有严密的数据要求与描述的文件，二是严格记录修改并进行回归测试。

8.5 软件维护的副作用

- 文档资料修改的副作用
  - 如果没有将源程序或数据结构的任何修改在相应的文档资料中予以反映，就会产生文档资料修改副作用。
  - 无论是修改数据流、软件结构和模块过程，还是修改其它有关的属性，都应该更新相关的技术文档资料以反映出这种修改。不能正确地反映软件当前状态的文档资料甚至比没有文档资料还要糟糕，因为它将导致维护过程中对软件性能结构的误解，并由此引入错误、造成软件失效。
  - 另一方面，最终用户希望描述软件用法的文档资料是完善的，任何对软件的修改若没有反映到用户手册中都会造成使用上的困难。
    - 例如，交互式输入格式或次序的修改在用户手册上没有得到反映时，要么发生输入错误要么产生错误的结果，致使用户感到软件性能比修改之前更糟，降低了用户对软件质量的信心。

8.6 软件维护的工作量估算

- 软件开发过程中潜伏有错误，软件维护工作中又有可能引入新的错误。因而维护工作难度和工作量都比较大。
- 1970车维护费用占软件总开支的35%一40%。1980年为40%一60%，而1990年约为70%一80%，
- 一种简单的维护工作量的估算模型为：

$$M = P + Ke^{(c-d)}$$

  - 其中P为生产性工作量，即维护过程中分析与评价软件、修改设计和编程等所需的工作量。
  - $Ke^{(c-d)}$ 表示对软件代码的说明，即对数据结构、接口特性和性能范围等的解释所花费的工作量。这里的K为维护人员的经验常数，c为程序复杂性系数，d为维护人员对软件熟悉程度的度量系数。

8.6 软件维护的工作量估算

- 耗时记录法：
  - 实地记录维护耗时的人月即可，此法准确但预测性差。
- 按维护指令比计算工作量
  - 统计并计算维护指令中的修改指令数Im与增加指令数In占总指令数I Σ的比，由已确定开发工作量Ed(人月)求出维护工作量Em(人月)，Em越大越难维护，Em越小越容易维护。

$$E_m = \frac{I_m + I_n}{I_\Sigma} E_d(\text{人月})$$

8.6 软件维护的工作量估算

- 根据维护的难易与维护人员水平计算
  - 设c为结构复杂度，d为维护人员对软件的熟悉程度，K为经验常数，则维护工作量Em计算如下：

$$E_m = \frac{I_m + I_a}{I\sum} E_d + K^{(c-d)}$$

8.7 软件维护标准文档

- 软件问题报告

软件问题报告	登记　号： 记证日期： 时　　间：							
单元测试 □　组装测试 □　确认测试 □　运行维护 □	状态	1	2	3	4	5	6	7
报告人：　姓名 地址： 电话：								
问题：　程序：＿＿＿＿　数据库：＿＿＿＿　文件：＿＿＿＿								
子程序：子系统： 数据库： 测试用例：　修订版本号： 文件： 硬件：　磁带：								
问题描述	影响							
附注：								

8.7 软件维护标准文档

- 软件问题报告各项含义解释如下：
  - 登记号：由软件配置控制部门统一规定的唯一顺序编号。
  - 登记日期：软件配配置控制部门登记该报告的日期。
  - 问题发现日期，发现该问题的日期和时间。
  - 活动： 指明在哪个阶段发现的问题，分为单元测试，组装测试，确认测试和运行维护四个阶段。
  - 状态：
    - 1．软件问题报告正在被复查；
    - 2．软件问题报告指定开发人员去进行处理；
    - 3．修改已做完，测试好，正准备提交主程序库；
    - 4．主程序已更新．但尚未重新测试；
    - 5．测试重做，但问题依旧在；
    - 6．测试重做．所作修改无故障，“软件问题报告”关闭；
    - 7．问题报告留待以后关闭，处理改善性的维护

8.7 软件维护标准文档

- 软件问题报告各项含义解释如下（续）：
  - 报告人： 是指写软件问题报告的姓名、地址、电话
  - 问题方面：区分程序问题属于子程序还是属于数据库问题或两者皆有
  - 子程序/子系统：标出问题所在的子程序名或者子系统名
  - 修订版本号：出现问题的子程序版本号
  - 磁带：包含问题的子程序的磁带标识符
  - 数据库：发现问题时所用数据库标识符
  - 文件号：有错误的文件编号
  - 测试用例：出现错误的主要测试用例标识符
  - 硬件：发现问题时使用计算机系统的标识符
  - 问题描述/影响：问题征兆的详细描述，或实际问题所在，给出该问题对将来测试、接口、文件的影响
  - 附注：记载补充信息

8.7 软件维护标准文档

- 软件修改报告

软件修改报告	登记号： 登记时间： 时间：
报告人： 子系统名： 子程序名：	
响应哪些问题软件问题报告	编号
修改： 程序 ☐ 文件 ☐ 数据库 ☐ 解释 ☐ ☐	
修改描述：	
修改： 语句类型 I/O ☐ 计算 ☐ 逻辑 ☐ 数据处理 ☐ 程序名： 老版本： 新版本： 数据库： 数据库修改报告： 文件： 文件更新： 修改已测试否： 硬件： 成功否： 单元： 子系统： 组装： 确认： 运行： “软件问题报告”问题叙述准确否： 是 ☐ 否 ☐ 附注：	
问题来自：软件需求说明书 ☐ 设计说明书 ☐ 数据库 ☐ 程序 ☐	
资源估计： 人时数： 计算机时间：	

8.7 软件维护标准文档

- 软件修改报告各项含义解释如下：
  - 登记号：软件配置控制部门收到“软件修改报告”指定的编号
  - 登记日期：软件配置控制部门登记该“软件修改报告”的日期
  - 时间：准备好“软件修改报告”的时间
  - 报告人：写软件修改报告的作者
  - 子系统名：受修改影响的子系统
  - 子程序名：被修改的子程序名
  - 响应“软件问题报告”的问题编号：被软件修改报告处理的“软件问题报告”的编号，部分处理在编号后加上p
  - 修改：包括程序修改、文件更新、数据库修改或它们的组合
  - 修改描述：关于修改的详细描述，如文件修改列出更新通知
  - 批准人： 批准人签字后正式进入修改

8.7 软件维护标准文档

- 软件修改报告各项含义解释如下（续）：
  - 语句类型：按表中给出填写修改涉及到的数据类型
  - 程序名： 被修改的程序、文件、数据名字
  - 老版本：当前版本|修订版本的标识
  - 新版本：修改后版本|修订版本标识
  - 数据库：修改数据库标识数据库
  - 修改报告：数据库修改申请号
  - 文件：文件修改的文件名
  - 修改后是否测试：填写测试状况
  - 软件问题报告：叙述回答问题准确情况
  - 附注：准确叙述要维护的内容
  - 问题来自：哪个文档
  - 资源估计：完成修改所需资源人时数，计算机机时数

8.7 软件维护标准文档

- 系统开发日志
  - 记录了项目的目标、开发的原则、优先考虑的问题、实验技术和工具、每天发生的问题、项目成功的原因和失败的教训。系统开发日志这一历史文件便于维护人员理解系统开发。
- 出错历史
  - 记录程序的出错历史，帮助维护员分析程序出错的类型和出错的频率，确定员麻烦的程序模块，从而提高维护人员的工作效率。

8.7 软件维护标准文档

- 系统维护日志
  - 记录维护阶段系统如何改变和为什么进行改变，包括改变的原因，改变的策略，问题所在的位置。
  - 系统的测试历史包括：
    - (1)在开发和修改期间漓试的模块效
    - (2)在开发的修改期间发现的错误数
    - (3)每个模块发现错误的平均数
    - (4)发现和纠正错误的平均时间
    - (5)发现错误类型和频率

8.7 软件维护标准文档

- 系统维护日志
  - 系统的测试历史包括（续）：
    - (6)硬件故障
    - (7)受硬件故障影响的软件错误
    - (8)代码错误
    - (9)设计错误
    - (10)说明书错误
    - (11)列出较多错误的模块名
    - (12)列出复杂的模块名

8.7 软件维护标准文档

系统可维护性的检查表

可靠性

1. 程序中是否包括对于可能出现的未定义的数学运算的检查?
2. 循环终止与多重转换变址参数范围是否在使用之前测试过?
3. 下标范围是否在使用之前测试过?
4. 是否包括错误恢复与再启动过程?
5. 所有数值方法是否准确?
6. 输入数据是否证实过?
7. 实际输出结果是否与预期的一致?
8. 是否大多数执行路径已经测试过?
9. 测试是否集中在最复杂模块与最复杂的模块接口?
10. 正常的、特殊的、异常的测试用例是否都包括在测试用例中?
11. 测试中是否用了假设数据与实用数据?
12. 系统是否用了标准库程序还是另搞一套?

8.7 软件维护标准文档

- 系统可维护性的检查表（续）
  - 可移植性

1. 程序是否可以高级的独立于机器的语言写成的？
2. 程序是否使用了标准化程序设计语言写成，且使用了标准版本？
3. 程序是否使用了标准的普遍使用的库功能和子程序？
4. 程序是否极少使用或根本不使用操作系统功能？
5. 程序达到计算精度是否与字长与存储容量大小无关？
6. 程序在执行之前是否初始化内容？
7. 程序在执行之前是否要测定输入或输出设备？
8. 程序是否把与机器相关的语句分离出来，并有说明文件？
9. 程序是否结构化，并允许在小一些的的计算机上分段覆盖运行？
10. 程序中是否避免了依赖于字母数字或特殊字符内部表示式或有说明文件？

8.7 软件维护标准文档

- 系统可维护性的检查表（续）
  - 可测试性

1. 程序是否模块化，结构是否良好？
2. 程序是否可理解？
3. 程序是否可靠？
4. 程序能否任意显示中间结果？
5. 能否以清楚的描述方式说明程序输出？
6. 程序能否一提出要求就能显示所有的输入？
7. 程序有否能跟踪与显示逻辑控制流的能力？
8. 程序是否有检查点再启动的能力？
9. 程序是否能显示带说明的错误信息？

8.7 软件维护标准文档

系统可维护性的检查表（续）

可理解性

1. 程序是否结构化、模块化、结构良好？
2. 文件编制：程序是否文件化？模块、子程序都要有一个注释要求
   - 2.1 用一两句话说明模块是干什么
   - 2.2 列出在模块中用于修改的变量值
   - 2.3 列出调用该模块的模块
   - 2.4 列出该模块所调用的模块
3. 列出其它有用的注释内容

8.7 软件维护标准文档

系统可维护性的检查表（续）

可理解性（续）

3.1 输入和输出
3.2 精确度检查
3.3 限制范围和约束条件
3.4 假设
3.5 为所有可预见错误留有一个出错处理过程出口
3.6 程序修改记载
3.7 程序编写日期和最后一次修改日期

4. 一致性
4.1 程序中的缩进和间隔是否一致？
4.2 每一行代码是否最多一个语句？
4.3 变量名、过程名是否单一的、描述性的、符合有关标准？
4.4 在程序中每个变量、每个过程是否仅有唯一的名字？
4.5 每个变量是否仅表示一个值？每个过程是否仅表示一个逻辑功能？

8.7 软件维护标准文档

系统可维护性的检查表（续）

可理解性（续）

4.6 程序是否真正体现了设计的整体性？

4.6.1 是否既没有增加又没有减少设计结构？

4.6.2 代码是否准确而清楚地体现了设计结构？

4.7 一个数组和一个表的所有元素功能是否相关？

4.8 是否使用了括号来标明复杂算术和逻辑表达式的运算顺序？

5. 完整性

5.1 是否提供了变量名的交叉引用表及调用与被调用子程序的映象图？

5.2 是否所有的外部引用都是可辨识的，所有输入输出都有说明？

5.3 程序是否限制使用所有一般系统库中没有的被引用子程序？

5.4 是否所有终端条件都得到了说明？

5.5 是否包括出错处理过程？

5.6 错误信息是否说明并能清楚地显示？

8.7 软件维护标准文档

系统可维护性的检查表（续）

可理解性（续）

6. 简明性

6.1 所有代码是否都有作用？

6.2 所有变量是否都是必不可少的？

6.3 是否能通过建造公共模块m子程序来避免多余的代码？

6.4 所有标号是否都能被转换？

6.5 是否避免过多地把一个程序分解成过多的模块、函数或子程序？

6.6 是否对表达式进行了因式分解，避免公共子表达式的不必要重复？

6.7 程序中是否避免了对同一变量的多次求反，去掉后程序不变？

6.8 程序是否避免了很难理解的非标准的语言特性？

8.7 软件维护标准文档

- 系统可维护性的检查表（续）
  - 可修改性

1. 程序是否模块化？结构是否良好？
2. 程序是否可理解？
3. 程序是否避免了在算术表达式、逻辑表达式、表|数组的大小，以及输入|输出设备命名符中使用文字常数？
4. 是否有用于支持程序扩充的附加存储空间？
5. 程序是否提供了评价修改的变化影响以及说明变化的资料？
6. 是否取消了冗余代码？
7. 是否提供常用功能的标准程序库？

8.7 软件维护标准文档

系统可维护性的检查表（续）

可修改性（续）

8. 是否有通用性
   8.1 在不同硬件配置上运行
   8.2 能否用不同输入输出操作
   8.3 能否在子模式下选定一组特性功能
   8.4 能否根据资源的可用性以及不同的数据结构和算法运行？
9. 程序是否在下述能力方面具有灵活性？
   9.1 是否把那些可变化因素都分离到单独模块中？
   9.2 是否提供不受个别功能发生预期变化影响的模块接口？
   9.3 能否确定一个能当做应急措施的一部分或在小的计算机上运行的子程序？
   9.4 是否一个功能模块只执行一个功能？
   9.5 能否在模块执行功能基础上定义模块通信？
10. 每一个变量的使用是否尽可能固定了？

8.7 软件维护标准文档

- 可维护性评估