

中华人民共和国国家标准

GB/T 14079—93

软件维护指南

Guideline on software maintenance

1 主题内容与适用范围

本标准描述软件维护的内容和类型、维护过程及维护的控制和改进。

本标准适用于软件生存周期的运行和维护阶段,主要供软件管理人员和维护人员使用。

2 引用标准

GB 8567 计算机软件产品开发文件编制指南

GB/T 11457 软件工程术语

3 术语

本标准使用 GB/T 11457 中的术语及下列术语:

3.1 自底向上法

在层次结构的软件中,一种从最低层成份开始逐级向上扩展,直到最高层成份的开发方法。

3.2 自顶向下法

在层次结构的软件中,一种从最高层成份开始逐级向下扩展,直到最低层成份的开发方法。

3.3 编译扩展

一种程序设计语言的特征。这种特征超越了该语言的标准特征,但仍可以为专门的编译程序所接受并加以编译。

3.4 同级评审

一种质量保证方法,由两个或多个同级程序员互相检查、评估,以确保被检查内容正确,且与软件的其他部分相一致。

3.5 软件维护管理机构

为评审修改带来的影响、制订维护计划、复查修改结果、管理维护工作等而设立的机构。

3.6 软件维护主管

组织、管理和协调维护工作的负责人。

3.7 维护管理人员

管理一个或几个软件的维护工作的技术人员。

3.8 软件维护人员

具体完成软件维护的工作人员。

4 软件维护的内容与类型

软件维护是在软件产品交付使用之后,为纠正故障,改善性能和其他属性,或使产品适应改变了的环境所进行的修改活动。

软件维护一般分为完善性维护、适应性维护和改正性维护三种类型。

国家技术监督局1993-01-07批准

1993-08-01实施

4.1 完善性维护

完善性维护是为扩充功能和改善性能而进行修改和扩充,以满足用户变化了的需求。主要包括:

- a. 为扩充或增强功能而作的修改(如扩充解题范围和算法优化);
- b. 为提高性能而作的修改(如提高精度,节省存储空间等);
- c. 为便于维护而作的修改(如增加注释,改进易读性)。

4.2 适应性维护

适应性维护是为适应软件运行环境的变化而作的修改,变化的主要内容包括:

- a. 影响系统的规定、法律和规则的变化;
- b. 硬件配置的变化,如机型、终端、打印机等的变化;
- c. 数据格式或文卷结构的变化;
- d. 系统软件的变化,如操作系统、编译系统或实用程序的变化。

4.3 改正性维护

改正性维护是为维持系统操作运行,对在开发过程产生而在测试和验收时没有发现的错误而进行的改正。所必需改正的错误包括:

- a. 设计错误;
- b. 逻辑错误;
- c. 编码错误;
- d. 文档错误;
- e. 数据错误。

5 软件维护过程

软件生存周期中的维护阶段通常起始于软件产品交付给用户、用户验收之时。软件维护活动通常可定义成软件生存周期中前几个阶段的重复。软件维护与软件开发有许多相同的活动,但也有其独特之处:

a. 维护活动限定在已有系统的框架之内完成,维护人员必须在已有的设计和编码结构的约束下作出修改,一般系统越旧,软件维护越困难和越费时。

b. 通常软件维护阶段的时间比软件开发的时间长得多,但一项具体的软件维护一般比该软件的开发时间短得多。

c. 软件开发必须从无到有产生所有测试数据,而软件维护通常可以使用现有的测试数据进行回归测试。有时还要产生新的数据,对软件修改及修改后的影响进行必要的测试。

完成一项软件维护的过程是复杂的。下面按顺序列出完成一项软件维护过程的步骤:

- a. 确定修改类型;
- b. 确定修改的需要;
- c. 提出修改请求;
- d. 需求分析;
- e. 认可或否决修改请求;
- f. 安排任务进度;
- g. 设计;
- h. 设计评审;
- i. 编码修改和排错;
- j. 评审编码修改;
- k. 测试;

- l. 更新文档;
- m. 标准审计;
- n. 用户验收;
- o. 安装后评审修改及其对系统的影响。

其中有几个步骤会经常发生循环,但并不是每次修改都要执行所有的步骤。

6 软件维护的控制和改进

软件维护必须有控制地进行,使整个过程中都处于适当的管理和控制之下。除了控制预算、进度和人员,关键在于要由软件维护主管来负责控制和修改系统。

大量的编码在开发过程中并非都考虑到了维护。即使原来是良好设计及良好实现的编码和逻辑,也会因无休止的“快速排错”和修补工作受到破坏。所以一个系统不仅在开发时要考虑到维护,还要在维护时考虑到将来的维护。

6.1 软件维护的控制

软件系统的可维护性常常随着时间的推移而降低,这是许多因素综合的结果。如果没有为软件维护管理制定严格的条例,或条例贯彻不力,许多系统都将蜕变到无法继续维护的地步。

软件维护的目标是保持系统功能和及时、满意地响应用户的请求。

软件维护的控制是保持一个有秩序的维护过程,在这个过程中所有的维护请求要正式提出、评审,给予一个优先级并安排进度。

6.1.1 确立软件维护的策略

软件维护策略的确定是软件维护控制的一个关键步骤。软件维护策略应充分地描述软件维护组织的责任、权利、职能及操作,它应全面地考虑到软件系统和它的环境的任何类型变化。该策略应由软件维护管理机构制定和支持。

软件维护策略必须具体地阐述修改的需要和理由、修改的责任和步骤。规定控制修改软件的过程和步骤,使请求的修改从提议到完成有控制地进行。

为保证维护策略的贯彻执行,需进行评审和审计。

6.1.2 评审和评价所有修改请求

- a. 所有的修改要求应先提出正规的书面请求;
- b. 评审所有修改请求;
- c. 分析和评价修改请求的类型和频度;
- d. 考虑对修改的需要程度和它可预见的使用,所有修改都需有充足的理由;
- e. 评价修改,以确保与原来的系统设计和用意不冲突,对每个修改都应该仔细考虑其影响;
- f. 应特别强调确定所建议的修改是增强还是降低系统的性能;
- g. 仅当修改的效益超过其成本时方可修改。

6.1.3 为维护安排进度

- a. 给每个修改请求分配一个优先级;
- b. 为每个认可的修改请求安排进度;
- c. 遵守安排的进度。

6.1.4 将代码修改限制于批准的工作范围内

软件维护主管必须监督维护人员的工作,确保只在授权的工作范围内作修改。为有效实行监督,必须将所有的维护活动记入文档,包括修改请求报告和完成修改后的源程序清单,并为系统复原做好安排。

6.1.5 强制实施文档标准和编码约定

必须贯彻编码约定和文档标准,以对软件维护人员的所有工作进行经常不断的强制性评审和检查。

在开始一项新的维护工作之前,应当为更新文档分配足够的时间。

6.2 软件维护的改进

可维护性是对软件进行修改的难易程度。一个系统的可维护性必须放在系统的整个生存周期中加以考虑。在系统最初的设计和开发阶段就应考虑到可维护性。

由于维护阶段的处理过程同开发阶段相似,因此许多技术和开发工具也可用在维护阶段。为提高软件可维护性,应在系统的整个生存周期中综合地使用下列技术和原理。

6.2.1 编码指南

编码指南和标准提供了一种提高系统可维护性的结构和框架,它使得系统以一种共同的、更易理解的方式进行开发和维护。编码应遵循下列基本原则。

6.2.1.1 单一高级语言

尽可能只用一种符合标准的高级语言。

6.2.1.2 编码约定

维护人员首先必须克服的困难是编码本身,开发人员和维护人员编写大量源码时很少考虑到以后的维护人员,结果使得源码的可读性很差。源码一定要加注解并用结构化格式编写。下列技术可提高程序的可读性:

- a. 尽量采用较简单的方法;
- b. 代码的每节开始行使用行首空格把一系列代码分成段。行首空格和字间的间隔是显示从属关系的两种方法;
- c. 用有意义的注释来适当地为代码加说明;
- d. 使用有意义的变量名,以表达此数据项是什么以及为何要使用它;
- e. 避免使用相似的变量名;
- f. 在程序的过程/函数之间用参数来传递数据;
- g. 在变量名中使用数字时,应放在末端。用作程序序标签或标号的数字应按顺序给出;
- h. 逻辑上相关的功能应集中安排在同一模块或模块集,尽可能使逻辑流向自顶向下;
- i. 避免使用程序语言版本的非标准特征。

6.2.1.3 结构化和模块化

应采用自顶向下的程序设计方法,使程序的静态结构与执行时的动态结构相一致。

模块化是指用一组小的层次结构的单元或例行程序构成程序,其中每个单元或例行程序集完成特定的单一功能。模块性不是仅仅将程序分段,模块的结构必须遵循下列设计原则:

- a. 一个模块应只完成一个主要功能;
- b. 模块间的相互作用应最少;
- c. 一个模块应只有一个入口和一个出口。

6.2.1.4 标准数据定义

一定要为系统制定一组数据定义的标准。这些数据定义可汇集于数据字典。字典项定义了系统中使用的每个数据元素名字、属性、用途和内容。这些名字要尽可能具有描述性和意义。正确一致地定义数据标准,就会大大简化阅读和理解各模块,并确保各模块间的正确通信。

6.2.1.5 良好注释的代码

好的注释可增强源码的可理解性。除了提高程序可读性,注释还有两个重要用途,即提供程序的用途和历史信息、它的起源(作者、生成和修改日期)、子程序名和个数以及输入/输出需求和格式,其次也提供操作控制信息、指示和建议来帮助维护人员理解代码中不清楚的部分。

6.2.1.6 编译程序扩展

使用编译程序的非标准特征会严重影响系统的可维护性。如果编译程序更改了,或如果系统必须移至新机器,则以前的编译程序扩展很可能与新的编译程序相冲突。因此最好限制语言的扩展和保留语言

基本特征的一致。如果需要使用编译程序扩展,应编制良好文档加以说明。

6.2.2 文档编写指南

一个系统的文档是良好维护的基础,文档编写工作应贯穿系统的整个生存周期。应有计划地建立和及时地更新文档,使维护人员能很快地找到所需的信息。应参照 GB 8567 编制文档。

文档合格的关键不仅是将必需的信息记录下来,以保持文档的及时更新和一致;而且必须使维护人员能迅速地获得它。对于维护人员来说,具有受控的存取和修改能力的联机文档是文档的最佳形式,如果不能提供联机文档,应保证有一机制使维护人员在任何时候能取用硬拷贝的文档。

6.2.3 编码和评审技术

本条列出有助于提高软件可维护性的设计和评审技术。

6.2.3.1 自顶向下/自底向上法

应将自顶向下与自底向上的方法组合起来使用。

6.2.3.2 同级评审

同级评审是一种质量保证方法。参加评审人员务必明白他们不是要评价其他程序员的能力或表现,而是分析和评价编码。评审内容应包括可维护性。

6.2.3.3 审查

审查是一种质量评估技术,在软件生存周期中检查各阶段工作,然后产生一个报告指出发现的错误和提出错误改正要求。

6.2.3.4 走查

简单的走查方式是让两个维护人员一起讨论正在进行的工作,复杂的走查方式可以有一份日程表、报告书和一位记录秘书。不论何种方式,目标是通过公开直接的交流,提炼好的主意,修改原来的方案。

6.2.4 测试标准和过程

测试是软件维护的关键部分,因此测试过程必须强调一致性,并以合理的原则为基础,测试计划要定义预期的输入,测试有效的、无效的、预期的和出乎意料的情况。测试要检查程序是否执行预期任务,测试的目的是发现错误,而不是证明错误不存在。

只要有可能,测试过程和测试数据均需由其他人完成,而不是由做系统实际维护的人来完成。

6.3 软件维护人员的管理

管理是改进软件维护过程的主要因素之一。管理必须指导怎样维护软件,行使对整个过程的控制,并保证使用高效的软件维护技术和工具。

为确保实现成功的维护,在维护过程中要有效使用良好的管理技术和方法,必须建立软件维护组织机构。

软件维护机构由维护主管、维护管理机构、维护管理员和维护人员组成。

软件维护机构的主要任务是审批维护请求,制订并实施维护策略,控制和管理维护过程,负责软件维护的审查,组织评审和验收,确保软件维护任务的完成。

软件维护人员的素质对于有效地进行维护是十分重要的,因此应为维护项目选择合格的各级人员。

下面列出挑选软件维护人员和进行维护管理的要点:

- a. 维护与开发同等重要,同样具有难度;
- b. 维护人员应是合格的、有责任心的人;
- c. 维护不能当作初级人员“放任自流”式的培训;
- d. 全体人员应轮流分配去做维护和开发工作;
- e. 出色的维护工作应同出色的开发工作一样受到奖励;
- f. 必须强调对维护人员进行良好的培训;
- g. 轮换分配,不应让一个系统或一个系统的主要部分成为某个人的专有领地。

7 软件维护与软件重新设计

维护是一种不断进行的过程,但有时也应考虑是否要重新设计一个软件系统。当一个软件已变得易出差错、效率降低和耗费增大,再对其继续维护的成本/效益比可能会超出重新设计一个系统时,应考虑是否要重新设计一个软件系统。下列特征可帮助管理人员决定是否应重建软件。

7.1 软件经常出错与性能恶化

代码越久,则经常的更新、新的需求和功能增强就越会引起系统的故障和性能恶化。

7.2 程序结构和逻辑流过分复杂

具有部分或全部下列属性的软件通常很难维护,需重新设计:

- a. 过多使用 DO 循环;
- b. 过多使用 IF 语句;
- c. 使用不必要的 GOTO 语句;
- d. 过多使用嵌入的常数和文字;
- e. 使用不必要的全程变量;
- f. 使用自我修改的代码;
- g. 使用多入口或多出口的模块;
- h. 使用相互作用过多的模块;
- i. 使用执行同样或相似功能的模块。

7.3 过时的代码

过时的代码严重影响新系统的性能发挥。

7.4 在仿真方式下运行

采用仿真方法,常阻止系统发挥全部能力和所有功能。仿真系统往往介于功能上尚可实用,但效率较低这二者之间。

7.5 模块或单个子程序非常大

此时,大模块结构应重新构造,分成较小的、功能上相关的部分,这可增强系统的可维护性。

7.6 过多的资源需求

需要过多资源的系统会成为用户的沉重负担,因此需考虑是增加更多的计算机设备还是重新设计和实现该系统。

7.7 将易变的参数编在代码中

尽可能对程序进行更新,以使它们能从输入模块或一个数据表中读入参数。

7.8 难于拥有维护人员

用低级语言编写的程序,尤其是汇编,需大量的时间和人力去维护。一般这类语言不为人们广泛了解,因此要寻找了解这类语言的维护人员日益困难。

7.9 文档严重不全或失真

文档不全、过时或失真,将造成维护工作极其困难。

附加说明：

本标准由中华人民共和国机械电子工业部提出。

本标准由上海计算机软件技术开发中心负责起草。

本标准主要起草人朱三元、刘光龙、王景寅、周庆隆。