文章编号: 1671- 251X(2010) 11- 0080- 04

四级阶梯式软件可靠性设计方法研究

陈建伟

(煤炭科学研究总院常州自动化研究院, 江苏 常州 213015)

摘要: 介绍了软件错误、软件缺陷、软件故障和软件失效的演变过程, 提出了相应的四级阶梯式软件可靠性设计方法, 即避错设计、查错设计、纠错设计和容错设计方法。该方法从软件错误的产生到失效的演变过程出发, 强调尽早地截断软件错误, 使其不向更严重的方向发展。该方法可以从根源上提高应用软件的可靠性。

关键词: 软件错误; 演变过程; 可靠性设计; 四级阶梯式软件设计; 避错设计; 查错设计; 纠错设计; 容错设计

中图分类号: **TD672** 文献标识码: **A**

Research of Four-tier Design Method of Software Reliability

CHEN Jian-wei

(Changzhou Automation Research Institute of CCRI., Changzhou 213015, China)

Abstract: The paper introduced development process of software bug, software defect, software fault and software failure, and put forward corresponding four-tier design methods of software reliability, including error avoided design, error checked design, error corrected design and fault-tolerant design. In view of the development process of software error from occurring to failure, the design method captures software error as early as possible in order to prevent error from developing. The method can improve reliability of application software radically.

Key words: software error, development process, reliability design, four-tier software design, error avoided design, error checked design, error-corrected design, fault-tolerant design

0 引言

可靠性是通过设计来赋予的。要保证和提高软件的可靠性, 关键在于可靠性设计, 这是软件可靠性工程的核心问题。本文从软件错误的产生到失效的 4 个阶段的演变过程出发, 在软件运行时尽早地截断软件错误, 使其不向更严重的方向发展, 加强了软件可靠性设计中的避错设计、查错设计、纠错设计和容错设计, 软件设计的 4 个阶段正好和软件错误演变的 4 个阶段相互对应, 以此思路来改进软件可靠性设计, 可以从根源上提高应用软件的可靠性。

1 软件错误的演变过程

一般来说, 软件错误需要经历一个演变过程才能最终导致软件功能或需求的部分甚至全部失效, 其演变过程如图 1 所示。

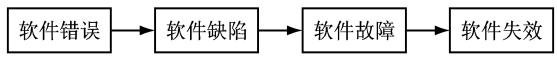

图 1 软件错误的演变过程

演变过程说明如下:

(1) 软件错误主要是一种人为错误;

(2) 一个软件错误必定产生一个或多个软件缺陷;

(3) 当一个软件缺陷被激活时, 产生一个软件故障, 同一个缺陷在不同的条件下被激活, 可能产生不同的软件故障;

收稿日期: 2010- 06- 29

基金项目: "十一五" 国家科技支撑计划项目(2009BAK54B05)

作者简介: 陈建伟(1974-), 男, 江苏常州人, 高级项目经理, 工程师, 硕士, 2000 年毕业于南京航空航天大学, 现主要从事工程质量总监工作。E-mail: cjw2002@21cn.com

(4) 软件故障若没有相应的容错措施并及时加以处理, 可导致软件失效, 同一个软件故障在不同条件下可能产生不同的软件失效。

从图 1 可以看出, 为了有效提高软件可靠性, 可靠性设计就应当从软件错误源头入手, 尽量避免或减少软件错误的出现, 一旦出现软件错误, 应当通过软件可靠性设计切断软件错误继续向后面各个阶段演变, 最终实现软件自我纠错功能, 防止错误蔓延扩散, 确保业务功能正常进行而性能只受微弱影响。

2　四级阶梯式设计方法

图 2 即为针对控制软件错误逐步蔓延和扩散、提高软件可靠性的四级阶梯式软件可靠性的设计方法。避错设计是在软件开发过程中尽可能减少或避免软件错误引入的一种设计方法, 它适用于一切类型的软件, 体现了预防为主的思想, 是首选方法, 贯穿于软件开发的整个过程。查错设计是指在软件开发过程中赋予某些特殊的功能, 使软件在运行过程中能自动地进行诊断和定位错误的一种方法。纠错设计是指在软件开发设计过程中赋予程序自我纠正错误、减少错误危害程度的一种设计方法。容错设计是指在软件开发设计过程中赋予程序某种特殊的功能, 使软件在已被错误触发的情况下, 系统仍能运行的一种设计方法。

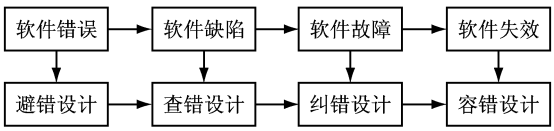

图 2　四级阶梯式软件可靠性设计方法

四级阶梯式软件可靠性设计方法按照串联顺序依次深化软件设计, 每个阶段的设计成果评审通过后即意味着最终软件失效数量减少和危害程度的减轻。判定具体的设计效果可以采用软件可靠性模型进行算法分析与评价, 譬如利用 Delphi 开发完成的可靠性分析评价工具, 能够输出 Goel- okumoto 等常用模型算法的评估结果和评估曲线[1]。

2.1　避错设计

避错设计贯穿于软件开发的整个生命周期, 它是可靠性设计的起始阶段, 是最基础、也是最重要的设计阶段, 具体包括以下几个方面。

2.1.1　模块化设计

通过对系统数据、事件、用户输入、高层功能等设计视角可实现软件模块分解, 形成模块结构层次图, 这是模块化设计的第一步。模块化分解后需要根据独立性原则进行模块范围和边界的调整, 模块耦合是影响模块独立性, 即影响避错设计的一个重要因素, 在软件开发过程中应该严格坚持如下原则: 尽量使用数据耦合, 少用控制耦合, 限制公共环境耦合的范围, 完全不用内容耦合。

对于软件模块内各个元素相关性的内聚度量, 不需要确定内聚的精确级别, 重要的是尽量争取高内聚和识别低内聚, 这样就可以通过改进设计来提高模块的内聚程度, 从而实现模块单个功能的独立性。模块分解方法如下:

(1) 功能分解: 将功能分解并分配给软件直至软件单元, 开发人员从功能实现的高层描述开始, 构造每个构件以及该构件与其它构件的连接关系。

(2) 面向数据的分解: 基于外部数据结构, 高层描述给出一般的数据结构, 底层描述提供包括数据元素以及与其相关的具体内容。

(3) 面向事件的分解: 基于系统必须处理的事件, 利用事件如何改变系统状态的信息。高层描述为不同状态编写目录, 底层描述给出状态转换。

(4) 面向对象的设计: 确定对象的类型以及它们之间的相互关系, 在最高层描述对象类型, 在较低层确定对象的数据和行为[2]。

2.1.2　制定公司级的可靠性设计准则

可靠性设计准则是把长期的行业软件开发实践中的经验总结出来并使之条理化、系统化, 经过公司内部评审发布后形成规范化的设计准则文档, 公司内每个开发项目设计时都应当贯彻执行。准则以条款的形式输出, 针对某个具体项目可以根据其项目性质和规模大小对设计准则进行适当的裁减后再进行应用。

2.1.3　健壮性设计

有效实施避错设计的另一个途径是加强健壮性设计, 其核心是能够抵御各种干扰, 能具备一定的防止错误输入和防止误操作的能力。主要措施:

(1) 检查输入数据的类型, 模块调用时检查参数的合法性, 控制错误蔓延。

(2) 在人机界面设计过程中, 列出所有的非法输入和误操作模式并进行分类处理, 通过操作提示灯和预处理措施, 阻止错误操作向程序内部转移。

(3) 进行简化设计,实现信息隐蔽。

2.2 查错设计

软件查错设计技术可分为被动式错误检测和主动式错误检测 2 种类型,二者的区别在于被动式错误检测是在程序的若干部位设置检测点,等待错误征兆的出现,而主动式错误检测是主动对程序状态进行检查。

2.2.1 被动式错误检测

在进行查错设计时,被动式错误检测方式通常将自动错误检测模块与执行模块分离,具有以下好处:

(1) 有利于预防错误检测模块干预程序的主干处理过程。

(2) 有利于软件的开发设计。主程序解决的问题是"应该怎么做",错误检测模块的任务是裁定"做的结果是否正确"。

(3) 有利于测试和维护。

(4) 有利于系统的再启动。

被动式错误检测的实施方法:

(1) 检查重要输入数据的属性。可以按规定的属性(如输入数据类型、数据长度、数据的正负号等)进行检查。

(2) 为表格、记录和控制块设置识别标志并以此检测输入数据。

(3) 按已知的数据极限和数据取值范围检查输入数据。

(4) 检查所有枚举数据的有效性。

(5) 如果输入数据中不存在冗余,可以对输入数据求和;如果系统中包括一个关键表格,则可在表格中增添一个总和项。

(6) 比较输入数据与内部常量或数据库现有数据的一致性[1]。

2.2.2 主动式错误检测

只有当错误征兆被传送到具有检查功能的部位时,被动式错误检测才能察觉到错误的存在。而主动式错误检测则是通过错误检测程序主动地对系统进行搜索,并指示所搜索到的错误。

主动式错误检测通常由一个检测监视器来承担,它可以作为周期性的任务来安排,规定固定时间,如每小时进行一次周期性检测。主动式错误检测也可以当作一个低优先权的任务来执行,在系统处于等待状态时主动进行错误检测。

2.3 纠错设计

纠错的前提是已经准确地检测到软件错误及其诱因并定位错误,这样程序才有能力修改、剔除错误。很多纠错的方法是在静态开发环境中实施的,要能使系统在运行过程中自动纠错,就必须先进行纠错设计。

纠错先要查错。查错的工作量通常占整个纠错工作量的 9/10 以上。所谓纠错的技术,主要是指查明程序错误时可能采用的工具和手段[3]。查错设计和纠错设计同步进行,互为补充,就能明显地提高软件的可靠性。

(1) 不允许一个用户的应用程序引用或修改其他用户的应用程序或数据。

(2) 查错模块、纠错模块和运行程序逻辑上相互隔离,确定好纠错模块的边界,防止错误渗透和蔓延。

(3) 考虑到操作人员有可能失误,输入模块对输入数据进行合法性检查,检查是否合法、越权,程序查出错误时及时纠错。

(4) 程序应该不能中止系统工作,不能诱发操作系统去改变其它应用程序及数据。

2.4 容错设计

完全或部分消除软件错误,尤其是对故障后果特别严重的错误对软件系统的影响,是容错设计的基本目标。容错和避错不同,容错是针对软件中的故障向系统提供保护的技术。构成容错软件的每一个版本程序应采用避错设计,确保单版本可靠性。常用的软件容错技术有 $N-$ 版本技术、恢复块技术、多备份技术等。$N-$ 版本技术是依据相同规范要求"独立"设计 N 个功能相等的程序(即版本),"独立"是指使用不同的算法、不同的设计语言、不同的测试技术甚至不同的指令系统等[4]。

$N-$ 版本技术的需求说明具有完全性和精确性,这是保证软件设计错误不相关的前提。因为软件的需求说明是不同设计组织和人员的唯一共同出发点[5]。它是指将若干个根据同一个需求规格说明、由不同软件开发人员完成的不同程序,在不同的"空间"同时运行或在同一"空间"依次运行,然后在每一个预定的检测点,通过测试或最终通过表决进行"裁决",在明确其正确性或一致性后接受这个结果,否则拒绝,并进行报警。

图 3 为软件容错的基本结构。从图 3 可以看出,具有这种结构的软件可能会出现以下 3 种情况:

文章编号: 1671- 251X(2010) 11- 0083- 03

基于组态软件的矿井生产自动化系统的设计

韩朝晖

(煤炭科学研究总院常州自动化研究院, 江苏 常州 213015)

摘要: 以黄陵二号矿井为例, 提出了一种基于组态软件的矿井生产自动化系统的设计方案, 介绍了系统的总体设计方案、使用的关键技术、系统实现的功能及特点。实际应用表明, 该系统安全可靠, 控制方便, 可扩展强。

关键词: 矿井; 生产自动化; 组态软件; RSView SE

中图分类号: **TD672** 文献标识码: **B**

Design of Mine Production Automation System Based on Configuration Software

HAN Chao-hui

(Changzhou Automation Research Institute of CCRI., Changzhou 213015, China)

Abstract: The paper put forward a design scheme of mine production automation system based on configuration software taking Huangling No. 2 Coal Mine as an example, and introduced general design

收稿日期: 2010- 07- 13

作者简介: 韩朝晖(1976-), 男, 江苏海安人, 工程师, 工程硕士, 2008 年毕业于南京理工大学, 现主要从事煤矿工业控制方面的研究工作。E-mail: chaohuihan@126.com

(1) 表决器判断正确, 软件正确实现软件需求规格说明所要求的功能;

(2) 表决器判断正确, 发现故障, 发出报警, 有效地防止因错误可能导致的严重结果;

(3) 表决器判断错误或软件本身存在不可诊断的故障, 软件错误运行[1]。

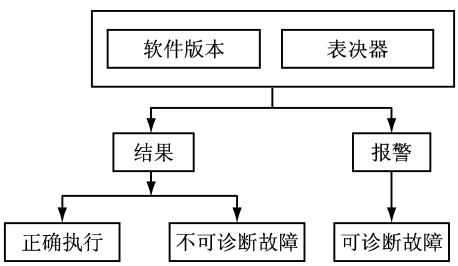

图 3 软件容错的基本结构

3 结语

四级阶梯式软件可靠性设计方法的优点是自上而下、层次明晰地将软件可靠性设计的过程流程化, 适用于各类性质的中大规模软件系统的设计工作。经过规范的 4 个阶段的、步步为营的可靠性设计以后, 软件内在和外在的质量将会大幅度地提高, 软件缺陷的数量也会大大减少。不过它也存在一个缺点, 那就是设计的总体工作量比一般的软件设计的工作量大, 而且设计工程师需要有丰富的专业设计工作经验才能胜任。

参考文献:

[1] 尹晶杰. 软件可靠性模型算法分析与评价[D]. 邯郸: 河北工程学院, 2008.

[2] 孙志安, 裴晓黎, 宋昕, 等. 软件可靠性工程[M]. 北京: 北京航空航天大学出版社, 2008.

[3] 中国 IT 实验室. 软件测试中常用的几种纠错技术[EB/OL]. (2009- 05- 20). [2010- 05- 16]. http://tech.ddvip.com/2009-05/1242804403120085.html.

[4] 张振华. 试论软件的可靠性及其保证[EB/OL]. (2004- 10- 20). [2010- 05- 20]. http://pm.csai.cn/quality/NO262.htm.

[5] 可靠性设计和环境试验学习园地. 软件可靠性中的容错设计分析[EB/OL]. (2008- 08- 31). [2010- 05- 12]. http://emcgarden.l4.bizcn.com/article.php/2293.