软件工程标准手册

基础和管理卷

石柱 编著

中国标准出版社

软件工程标准手册

基础和管理卷

石柱　编著

中国标准出版社
北　京

图书在版编目（CIP）数据

软件工程标准手册. 基础和管理卷/石柱编著. —北京：中国标准出版社，2007

ISBN 978-7-5066-4618-5

Ⅰ.软… Ⅱ.石… Ⅲ.软件工程-技术手册 Ⅳ.TP311.5-62

中国版本图书馆 CIP 数据核字（2007）第 125604 号

中国标准出版社出版发行
北京复兴门外三里河北街 16 号
邮政编码：100045
网址 www.spc.net.cn
电话：68523946 68517548
中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销
*
开本 787×1092 1/16 印张 15.75 字数 372 千字
2007 年 9 月第一版 2007 年 9 月第一次印刷
*
定价 34.00 元

前　言

自从拙著《软件工程标准手册》(以下简称该手册)于2004年出版以来,国家标准化管理委员会又陆续发布了一批新的或修订的软件工程标准。不少读者希望我对该手册进行修订,以反映国家软件工程标准的最新情况并包含该手册原来没有包含的有关内容。在中国标准出版社的支持下,对该手册进行了扩充和修订,将以"软件工程标准手册"丛书(以下简称本丛书)的形式出版,本丛书将分三卷出版,分别是《软件工程标准手册　基础和管理卷》、《软件工程标准手册　开发和维护卷》和《软件工程标准手册　质量度量和产品评价卷》。

《软件工程标准手册　基础和管理卷》(以下简称本卷)重点阐述了有关软件分类、软件工程标准分类、软件生存周期过程、软件配置管理、软件质量保证、软件文档编制和管理、软件开发工具和环境选择等方面的软件工程国家标准的内容、实施方法及相关技术,所涉及的软件工程国家标准包括:GB/T 8566—2007《信息技术　软件生存周期过程》、GB/T 8567—2006《计算机软件文档编制规范》、GB/T 11457—2006《信息技术　软件工程术语》、GB/T 14394—1993《计算机软件可靠性和可维护性管理》、GB/T 16680—1996《软件文档管理指南》、GB/T 18234—2002《信息技术　CASE工具的评价与选择指南》、GB/T 18492—2001《信息技术　系统及软件完整性级别》、GB/Z 18493—2001《信息技术　软件生存周期过程指南》、GB/Z 18914—2002《信息技术　软件工程　CASE工具的采用指南》、GB/Z 20156—2006《软件工程　软件生存周期过程　用于项目管理的指南》、GB/T 20158—2006《信息技术　软件生存周期过程配置管理》等11项现行有效标准。

本卷共由11章组成,其主要内容和结构如下:

第1章　基本概念。主要介绍软件及其特点、软件分类与代码、系统和软件完整性级别、软件危机、软件工程及其基本原理、标准和标准化、软件工程标准和标准化、软件工程标准分类等内容。

第2章　国内外软件工程标准概况。主要介绍美国和美军、欧洲和欧空局、以及我国的软件工程标准概况。

第3章　软件生存周期过程。主要介绍GB/T 8566的3类21个过程、100个活动和336项任务。

第4章　软件生存周期过程指南。简要介绍GB/T 8566的基本概念和实施策略，阐述在项目中应用GB/T 8566应考虑的关键要素，扼要介绍GB/T 8566在组织中应用的要求和理由，给出在系统生存周期模型中应用GB/T 8566的要点和示例，列出软件生存周期基本过程的输出。

第5章　软件生存周期模型及其选择原则。简要介绍GB/T 8566和GB/Z 18493—2001关于软件生存周期模型的选择要求，介绍目前在国内外已得到成功应用的5种软件生存周期模型，结合作者的实践和体会阐述了软件生存周期模型的选择原则。

第6章　软件配置管理。扼要说明软件配置管理的基本概念和基本要求，介绍配置管理组织与职责、配置标识、配置控制、配置状态记录与报告、配置审计等内容，详细阐述软件配置管理计划、配置审核报告等文档的编制指南。

第7章　软件质量保证。扼要说明软件质量保证的基本概念和基本要求，详细阐述软件质量保证计划、设计评审报告等文档的编制指南。

第8章　软件文档编制与管理。扼要说明软件文档的作用，介绍软件文档的编制和管理要求，详细阐述软件文档的编制策略、编制标准和指南，阐述软件文档的评审过程、分类和要求。

第9章　软件可靠性和可维护性管理。扼要介绍软件可靠性和可维护性的基本概念，阐述软件可靠性和可维护性大纲的编制要求，详细阐述了软件可靠性和可维护性大纲各要素的基本要求，阐述了软件可靠性和可维护性管理准则。

第10章　软件工具和环境。扼要介绍软件工具和环境的基本概念，阐述软件支持环境的基本要求，详细阐述CASE工具的评价、选择和采用指南。

第11章　软件项目管理。主要介绍GB/Z 20156—2006《软件工程　软件生存周期过程　用于项目管理的指南》的有关内容。

在本丛书编写过程中曾得到了何新贵院士、刘继忠、陈政、冯惠、唐为仁、韩素珍和唐莉梅等人的关心、支持和指导，在此表示谢意。

本书可供从事软件开发和维护的软件技术人员、技术管理人员、项目管理人员、质量管理人员、标准化人员、大专院校本科生、研究生学习及参考。

限于作者的水平，难免存在遗漏、欠缺和错误，敬请读者批评指正。

石　柱

2007年5月

目　录

第1章 基本概念

1.1 软件及其特点

从构成软件的基本要素来看，软件是与计算机系统的操作有关的程序、规程、规则及任何与之有关的文档[1]。软件的概念是逐渐发展起来的，在早期，软件即指计算机程序，此后将文档也包括在软件之中，再进一步发展为包含了程序、规程、规则和文档的定义，并强调文档是软件的重要组成部分。

程序是“按具体要求产生的、适合计算机处理的指令序列”[1]。程序是软件的重要组成部分，但绝不是软件的全部。

规程是“为解决某一问题而采取的动作的经过的描述”或“每次完成某一任务时要遵循的一组手工的步骤”[1]，主要描述在软件生存周期中应如何实施有关政策、规则和标准。例如，测试规程，用于描述进行软件测试时应遵循的测试步骤。

规则指软件开发人员在开发软件时应共同遵守的准则和法规。

文档指一种数据媒体及其记录的数据。它具有永久性并可以由人或机器阅读。通常仅用于描述人工可读的内容[1]。例如，技术文件和设计文件等。

从软件的组织结构来看，软件是一个由计算机软件配置项、计算机软件部件(CSC)和计算机软件单元(CSU)构成的层次结构[2]。

计算机软件配置项是“为独立的配置管理而设计的、且能满足最终用户功能的一组软件”。值得指出的是，软件配置项与计算机软件配置项(CSCI)是不同的两个概念。软件配置项指“为配置管理目的而作为一个单位来看待的软件成分，是进行软件配置管理的基本单位”。软件配置项可以是一个软件单元，也可以是软件开发过程中的某一最终的或中间的文档，例如，软件更改单、软件开发计划等。可见，软件配置项是 CSCI 的组成部分。

计算机软件部件是计算机软件配置项中的一个明确的部分。它可以进一步分解成其他计算机软件部件和计算机软件单元。

计算机软件单元是计算机软件部件设计中确定的、且能单独测试的部分。

软件作为一种逻辑实体，具有下述几个显著的特点[3]：

1) 抽象性：软件产品不是实物产品，其可见性差。

2) 严密性：软件工程是一个严密的逻辑工程。软件产品无正品和次品之分，不存在误差，误差即是错误。

3) “一次性”：任何软件的研制都是一个新的开发过程，即一次性、“创造性”的劳动。而

软件的成批生产只不过是简单的复制。

4）智力性：软件研制主要靠人的脑力劳动。软件研制的绝大部分工作都是靠人来完成的。

5）持久性：软件产品的质量与使用时间的长短无关，即软件产品无磨损性。因此软件的故障不能用普通产品更换零部件的方法来解决。

6）依赖性：软件产品常常受限于具体的计算机系统，为了减少软件产品对计算机系统的依赖性，应提高软件产品的可移植性。

7）复杂性：有人认为，计算机软件是人类能够创造的最复杂的产品之一。软件的复杂性既来自它所处理的实际问题的复杂性，又来自程序逻辑结构本身的复杂性。因此，软件技术的发展落后于人们对软件的需求，并且随着时间的推移，这种差距还在日益加大。

8）难以度量：目前对智力劳动尚无有效的度量方法，而软件研制又是新开发的智力产业，因此就更难于对它进行度量。

9）易出错：软件生产过程涉及一系列的“信息转移”，在有信息转移的环节都有可能发生信息转移的错误。

10）必须维护：软件产品在交付使用后还可能需要经常更改，因而软件维护是软件工程的一个必不可少的阶段。

11）成本昂贵：软件产品的研制需要投入大量的、复杂的、高强度的脑力劳动，其成本是非常高的。在国外，软件已占整个计算机系统成本的70%以上。据有关统计，美国普通民用软件的开发费用约为每行源代码10美元，而宇航级软件的开发费用约为每行源代码1000美元。

1.2 软件和硬件的异同

软件与硬件的主要不同点如表1-1所示[3]。总的说来，软件与硬件的本质区别是智力产品与物理产品之间的区别。

表1-1 软件与硬件的差异

软　　件	硬　　件
逻辑实体，始终不会自然变化，只是其载体可变。	物理实体，同规格产品的质量特性之间有散布，会随时间和使用而老化、磨损以至失效。
其接口是不可见的。	其接口是可见的。
尚没有标准件。	有标准的零部件。
研制过程主要是脑力劳动过程，在本质上是无形的，不可见，难以控制。	研制过程不只是脑力劳动过程，还有体力劳动过程，其过程有形，便于测控。
其不可靠问题基本是由于开发过程中的人为差错所造成的缺陷而引起的。	硬件失效通常是由其零部件或其结合的故障所引起的。
程序是指令序列，即使每条指令都正确，但由于在执行时其逻辑组合状态千变万化，不一定完全正确。	其不可靠问题不只是设计问题，在生产和使用过程中也会产生新的故障。

续表 1-1

软　　件	硬　　件
系统的数学模型是离散型的，其输入在合理范围内的微小变化可能引起输出的巨大变化，故障的形成无物理原因，失效的发展取决于输入值和运行状态的组合，无前兆。	系统在正常工作条件下其行为是渐变的，故障的形成和失效的发生一般都有物理原因，有前兆。
对生产过程进行严格的控制，可得到完全一致的产品。	对生产过程进行严格的控制，可将产品的容差控制在可接受的范围。
需在研制的全过程采取措施防错、检错、纠错和容错。	除了研制过程外，生产过程对产品的影响也很大，均需加强控制。
精心设计测试实例，执行严格的测试，查出错误并加以排除。	建立适当的环境应力条件，进行筛选，排除缺陷。
冗余设计应确保冗余软件相异，否则，不仅不能提高可靠性，反而会增加复杂性，降低可靠性。	相同部件之间自然是独立的，适当的冗余可以提高可靠性。
在使用过程中出现故障后，必须修改原产品以解决问题；若在修改时未引入新问题，其可靠性就会增长。	在使用过程中出现故障后，一般只需更换或修复失效的部件，使产品恢复良好状态，其可靠性一般不会提高。
软件维护通常涉及软件更改，软件更改通常会对其他部分造成影响。	硬件维修通常涉及零部件更换，零部件更换一般不会对其他部分造成影响。
维护过程复杂，通常需要专业软件人员进行。	维护过程相对简单，通过一定培训的人员即可胜任。
本身没有危险，但会对系统的安全性造成影响。应从系统的角度考虑软件的安全性。	本身可能有危险，可以也必须单独对安全关键部件进行分析和评估。
某处的修改会影响它处，维护时必须考虑这种影响。	维修某处一般不会影响它处。
失效率随故障的排除而下降，如图 1-1 所示。	失效率的变化类似于浴盆曲线，如图 1-2 所示。
可靠性参数估计无物理基础。	可靠性参数估计有物理基础。

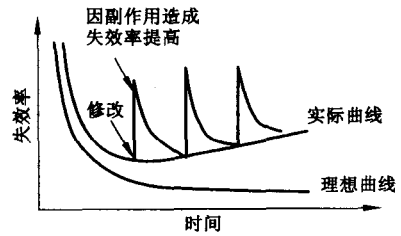

图 1-1　软件的失效率曲线

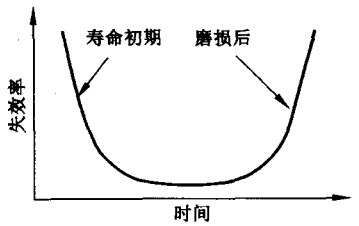

图 1-2　硬件的失效率曲线

软件与硬件的相似之处有如下几点[3]：

1）硬件可靠性是设备复杂性的函数；直观地看，软件可靠性也是其复杂性的函数。系统越复杂，其可靠性越低。

2）尽管硬件与软件的失效机理不同，但硬件与软件失效的外部表现都具有明显的随机性，都采用概率统计方法研究硬件和软件产品的可靠性。

3）如果固态电子器件（如晶体管和微型电路）的制造质量好，那么，在很长的时期内，它们没有耗损机理。造成故障的缺陷（不包括明显误用器件造成的故障）是在器件的制造过程中引入的。软件也是如此。

4）硬件可靠性可以采用可靠性增长试验来提高，也就是采用试验—分析—改正计划来发现、确定及改正可能造成设备早期故障的故障模式及机理。这与在软件中寻找和消除“缺陷”，并且因此提高可靠性的做法类似。

5）软件质量管理和保证方法与硬件质量管理和保证方法有许多相似之处，可以将许多成熟的硬件质量管理和保证方法直接应用于软件质量管理和保证。

软件与硬件的异同决定了软件质量管理和保证方法必然要借用硬件（产品）质量管理和保证的成功经验和成熟方法，针对软件的特点进行改造和改进，以使软件企业能在规定的时间和预算内开发出符合质量要求的软件。

1.3 软件分类与代码

计算机软件通常分为系统软件、支持软件和应用软件3大类。

系统软件指为特定的计算机系统或一族计算机系统所开发的软件，用以管理计算机系统资源，促进计算机系统及有关程序的运行和维护。例如，操作系统和网络系统软件等。

支持软件指所有用于帮助和支持软件开发的软件。例如，软件开发工具和软件评测工具等。

应用软件指为使用一个计算机系统以得到某种功能而专门开发的软件。例如，文字处理软件和科学计算软件等。

常用的软件分类原则是：

1）按照计算机软件的基本特征和主要功能进行分类；

2）采用线分类法（即层次分类法）。例如，代码的结构可分为如下3层：第1层表示大类，采用1位数字代码，其中“1”表示系统软件，“3”表示支持软件，“6”表示应用软件；第2层表示中类，采用2位数字代码，数字“99”表示收容类目；第3层表示小类，采用2位数字代码，数字“99”表示收容类目[4]。

计算机软件分类和代码的示例如表1-2所示[4]。

表1-2 计算机软件分类和代码的示例

代码			名称	代码			名称
1			系统软件	1	30		系统扩充程序（包含操作系统的扩充，汉化）
1	10		操作系统				
1	20		系统实用程序	1	40		网络系统软件

续表 1-2

代码			名称	代码			名称
1	99		其他系统软件	6	10	30	工程应用软件
3			支持软件	6	10	99	其他科学和工程计算软件
3	10		软件开发工具	6	15		文字处理软件
3	10	10	需求分析工具	6	15	10	编辑排版软件
3	10	20	设计工具	6	15	20	表处理软件
3	10	30	编码实现工具	6	15	30	汉字输入处理软件
3	10	40	测试工具	6	15	40	多种文字处理软件
3	10	50	维护工具	6	15	99	其他文字处理软件
3	10	60	文档生成工具	6	20		数据处理软件
3	10	70	集成化软件开发工具	6	20	10	数据采集软件
3	10	99	其他软件开发工具	6	20	20	统计及预测软件
3	20		软件评测工具(包括软件度量工具)	6	20	30	数据综合处理软件
3	30		界面工具	6	20	40	信息检索软件
3	40		转换工具	6	20	99	其他数据处理软件
3	50		软件管理工具	6	25		图形软件
3	50	10	项目管理工具	6	25	10	图形输入软件
3	50	20	配置管理工具	6	25	20	图形处理软件
3	50	30	质量管理工具	6	25	30	图形输出软件
3	50	99	其他软件管理工具	6	25	99	其他图形软件
3	60		语言处理程序	6	30		图像处理软件
3	70		数据库管理系统	6	30	10	图像生成软件
3	70	10	关系型数据库管理系统	6	30	20	图像编辑软件
3	70	20	层次型数据库管理系统	6	30	30	图像识别软件
3	70	30	网状型数据库管理系统	6	30	99	其他图像处理软件
3	70	99	其他数据库管理系统	6	40		应用数据库软件
3	80		网络支持软件	6	50		事务管理软件
3	99		其他支持软件	6	55		辅助类软件
6			应用软件	6	55	10	辅助设计软件
6	10		科学和工程计算软件	6	55	20	辅助制造软件
6	10	10	数学软件	6	55	30	辅助测试软件
6	10	20	科学计算软件	6	55	40	辅助教育软件

续表 1-2

代码			名称	代码			名称
6	55	99	其他辅助类软件	6	65	60	其他智能软件
6	60		控制类软件	6	70		仿真软件
6	60	10	实时控制软件	6	70	10	模拟仿真软件
6	60	20	非实时控制软件	6	70	20	数字仿真软件
6	60	99	其他控制类软件	6	70	30	混合仿真软件
6	65		智能软件	6	70	99	其他仿真软件
6	65	10	专家系统	6	75		网络应用软件
6	65	20	模式识别软件	6	80		安全与保密软件
6	65	30	自然语言理解与处理软件	6	85		社会公益服务软件
6	65	40	机器证明软件	6	90		游戏软件
6	65	50	机器翻译软件	6	99		其他应用软件

1.4 系统及软件完整性

1.4.1 确定和应用软件完整性级别的过程

完整性指“在计算机系统中，对软件或数据所受到的未经获准的存取或修改可加以控制的程度”[1]。完整性级别指“项的某个特性的取值范围的一种表示，该特性的取值范围对将系统风险保持在可容忍的限度内是必须的。对于执行缓减功能的项，此特性是指项必须执行缓减功能的可靠性。对于其失效能导致一个威胁的项，此特性是指对该失效的频率或概率的限制”[5]。项是能够作为单独考虑的一个实体，可以是硬件、软件或者软硬件构成的系统等。软件完整性需求是软件开发中软件工程过程所必须满足的需求，是软件工程产品必须满足的需求，或者是为提供与软件完整性级别相适应的软件置信度而对软件在某一时段的性能的需求[5]。

图 1-3 给出了一个确定和应用软件完整性级别的过程概貌。表 1-3 分别列出了确定系统完整性级别、软件完整性级别和软件完整性需求三个主要过程的输入和输出。

GB/T 18492—2001 采用了基于风险(分析)的完整性级别确定方法。所以，确定相应系统完整性级别的第一步是进行风险分析。为了实施风险分析，必须获取关于系统、系统环境、以及与系统相关的风险维方面的足够信息。风险分析应覆盖所有相关的风险维(如安全性、经济性和安全保密性等)。

对在风险分析中标识出来的任何风险，必须予以评估，以确定该风险是否可以容忍。一旦系统设计经过分析和评价有可容忍的风险，就为系统分配一个系统完整性级别。系统完整性级别反映了系统中存在的在最坏情况下的风险。

系统中软件的完整性级别最初被分配为与系统的完整性级别相同的级别。可以对系统的设计加以分析，以确定系统设计中是否存在某种体系结构特征，该特征可以为软件分配一个比系统完整性级别低的完整性级别。

系统是由一个或多个部件组合集成的。一个部件可以是单独的软件、单独的硬件或由分解为更细的部件组成的子系统。最初，系统的完整性级别将分配给系统的任何软件部件。确定软件完整性级别需要对系统体系结构进行分析，以便确定子系统的完整性级别能否低于系统的完整性级别。分析过程可以循环地进行，直至仅包含软件的子系统的完整性级别得以确定；或者包含软件的子系统的完整性级别被设计机构和完整性保证机构所认可，该认可的级别可以分配给子系统中的任何软件部件。

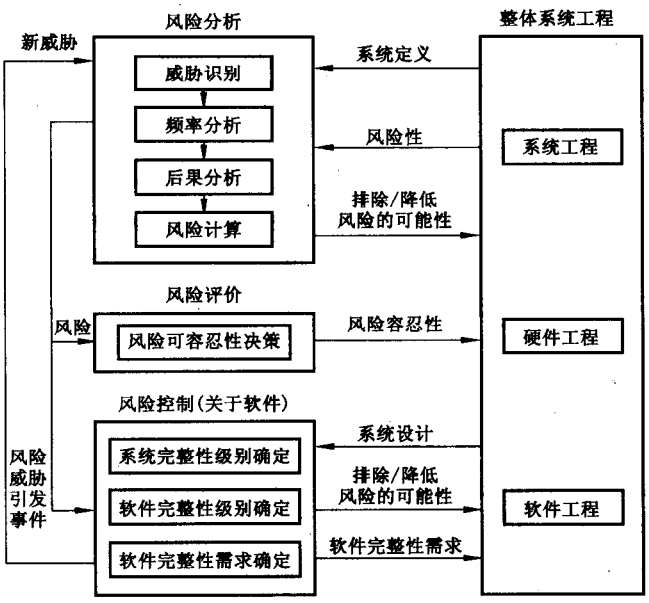

图 1-3 确定和应用软件完整性级别的过程

表 1-3 输入和输出

过 程	输 入	输 出
系统完整性级别确定	——相关风险维 ——系统定义 ——环境定义 ——系统体系结构(若可提供)	——风险 ——威胁 ——可容忍频率/威胁发生的概率 ——引发事件 ——引发事件的频率/概率 ——系统完整性级别
软件完整性级别确定	——系统完整性级别 ——子系统/软件体系结构 ——威胁清单以及对每个威胁而言： ·威胁可容忍频率或威胁发生的概率 ·可能导致威胁发生的引发事件 ·引发事件的期望频率或每个引发事件发生的概率	——子系统/软件完整性级别 ——被确认的降低完整性级别的体系结构特性
软件完整性需求确定	——子系统/软件完整性级别	——软件完整性需求

在系统的体系结构分析的过程中，很可能会发现在先前的风险分析中未曾遇到的新威胁和新风险。因此，应考虑新的风险信息并重新进行风险分析。

软件完整性级别既可以是分配给软件的、提供缓减功能的可靠性程度，也可以是分配给软件的、可能导致威胁产生失效的频率的限制。由于软件失效是系统性失效的函数，所以软件完整性级别表示一种置信度指标，它表示可靠地提供缓减功能的真实程度，或者表示不导致某威胁产生失效的真实程度。

确定和应用软件完整性级别是整个风险分析过程的一个部分。风险管理应在系统和软件产品的生存周期内实施，并可能随着设计的逐渐深入或设计的不断改进而反复进行。图1-1给出了整个系统工程过程与风险分析、风险评价和风险控制等风险管理过程之间的相互关系。

如果系统设计的改进可以消除或降低风险，在分配了系统和软件完整性级别之后又发现了新的威胁，或者系统设计具有不可容忍的风险，那么应重新进行风险管理。

软件完整性级别用来确定如何控制风险。相对于软件部件而言，就是明确在软件和软件工程方面的需求，从而获得在系统风险中起相应作用的软件的置信级别。这些需求就是软件完整性需求。

1.4.2 系统完整性级别的确定

系统完整性级别是系统的完整性级别，它对应于与系统相关的风险可容忍级别。因此，系统完整性级别与风险密切相关。确定系统完整性级别的步骤如下：

1）实施风险分析，确定系统相关的风险级别；

2）实施风险评估，判断风险的可容忍度；

3）确定系统的完整性级别。

风险分析通常包括威胁识别、频率分析、后果分析和风险计算等内容。对于每个识别出的风险，其风险分析的输出包括用适当术语表述的风险、与风险相关的威胁、与每个威胁相关的引发事件、以及引发事件发生的假设频率。

经过风险分析而得出的风险表述可以用于风险评估过程，以确定风险是否可以容忍。在确定系统和软件完整性级别的过程中，风险分析的所有结果可用于确定对系统中软件部件所需求的完整性级别。

风险分析可以覆盖安全性、经济性和保密性等诸多风险维。要覆盖的具体的风险维应由设计机构和完整性保证机构确定。

威胁识别标识与系统相关的威胁和可能导致威胁的引发事件。如果系统失效，按规定的系统操作或者系统在其环境中执行缓减引发事件后果的功能可导致威胁的产生，则这些威胁同系统相关联。发现先前未曾识别出来的威胁，将采用适于具体情况的方法，并将这些方法记录备案。在识别威胁过程中应考虑系统体系结构，以确保系统中所使用技术的特定失效模式也得到考虑。常用的威胁识别方法有：检查单法、工程经验法、FMECA、FTA、ETA、HAZOP、SCA等[6]。

频率分析用来估计经威胁识别而确定的每个引发事件发生的可能性。当系统的一个失效是一个引发事件，频率分析不用来估计失效发生的可能性，而是用来确定与可容忍风险目标相一致，并且是可实际达到的失效的频率的限度。用相关的历史数据估计事件频率，用诸