1 绪论

1.1 软件可靠性研究与实践的意义

随着计算机技术的发展和软件的广泛使用,武器装备系统和自动化指挥系统对软件的依赖性越来越强,软件对现代军事装备发展起着越来越重要的作用。但是,随着计算机技术的飞速发展,软件的规模越来越大,复杂性越来越高,软件可靠性的有效控制变得越来越复杂, 软件质量中最重要的问题一软件可靠性问题变得日益突出，据有关资料统计，美军武器装备软件可靠性整整比硬件可靠性低一个数量级，有的系统故障统计结果是软件故障占系统故障的60%-70%[1]。

随着软件核心地位日益突出，武器装备软件系统规模庞大、结构和功能复杂，软件越来越影响着整个系统装备的可用性，软件失效造成的故障已成为新的焦点，不可靠的软件引发的失效可能给软件的使用者或软件开发人员带来灾难性的后果。当前,软件的可靠性问题已经暴露得相当突出，其中较为重大的由软件故障引发的事故有：

（1）美国九十年代Therac-25型放射治疗仪的超剂量辐射事故，Therac-25是建立在一款没有经过正规培训的程序员开发的操作系统上，由于这款不易被察觉的“竞争条件(Race Condition)”的Bug，技术员可能在病人没有进行任何防护的情况下，意外地将Therac-25配置为高能模式,并发生了6起严重的超剂量辐射事故；

（2）1991年在海湾战争中，美国“爱国者”导弹的雷达跟踪系统发生故障，造成美军28人死亡，近百人受伤；

（3）1996年欧洲空间局发射的阿丽亚娜5号火箭由于由于Ariane 4火箭的工作代码在Ariane 5中被重新使用，但是Ariane 5更高速的运算引擎在火箭航天计算机中的算法程序中触发了Bug，最终导致了航天计算机的崩溃，使得在处女航开启数秒后被人为摧毁，同时被摧毁的还包括4颗卫星，造成经济损失达5亿美元。

（4）2019年埃塞俄比亚航空公司一架波音737 MAX 8 客机在飞往肯尼亚途中，因自动纠正失速系统（MCAS）使得飞机信号系统接收到一个假信号，信号显示飞机‘抬头’，所以控制系统持续给出了‘低头’的指令坠毁。机上有149 名乘客和8 名机组成员，无人生还。

（5）2020年加拿大军方CH148直升机坠毁因直升机飞行过程中出现了“严重颠簸”，软件故障导致飞行控制系统重启，直升机突然失去高度控制；

（6）· · ·

国内也出现了由于软件故障损失数以百万计的事故，凡此种种,已不胜枚举。这些事例提醒人们，应当重视软件可靠性问题，以避免这些灾难重演。高端装备是国之重器，在国民经济关键部门发挥着关键作用。其故障和失效往往造成较为严重的社会以及经济影响。因此，如何保障高端装备稳定而安全地运营是可靠性工程(Reliability Engineering)研究的主要问题。软件的质量和可靠性已成为制约装备质量和性能的瓶颈,军用软件的可靠性已成为确保军事和武器系统质量的关键。研究和分析军用软件可靠性要求的特点, 建立和完善军用软件可靠性设计、测试和评价技术,进行军用软件产品可靠性度量和评测,成为提高军用软件产品质量非常迫切而重要的课题。

随着软件规模和复杂程度的大幅提高，大型系统对软件依赖程度的逐渐增加，使得软件

质量问题引起了人们广泛地关注和重视。软件可靠性工程（Software Reliability Engineering, SRE）是对软件的质量(特别是软件的可靠性)进行管理和控制的学科，是软件工程与可靠性工程的结合，是为保证经济、及时地实现软件可靠性目标而采取的系统的活动和方法，是软件质量的一个重要方面[2]。

软件可靠性评估与预测是SRE中的重要一环，而软件可靠性模型是随机过程的一种表示，主要描述软件可靠性与其他相关因素(如：时间、软件产品特性等)之间的关系，是软件可靠性定量分析技术的基础。软件可靠性建模可归结为根据软件可靠性相关数据进行统计预估的过程。软件系统的失效率随着软件失效的消除而降低，可以根据时间来观察软件失效率的历史变化，进而估计当前软件系统的可靠性和未来完成规定目标所需的时间。以软件可靠性模型为支撑的软件可靠性定量分析技术，在软件开发过程中具有重要作用：可以对各种软件开发技术的优劣做出定量的评估；使项目管理人员能够在软件的测试阶段，进行可靠性增长分析，及时评估软件的开发水平；监控软件的运行性能，控制软件的设计变更，控制软件的功能扩充。

软件可靠性建模可归结为根据软件可靠性相关数据进行统计预估的过程。软件系统的失效率随着软件失效的消除而降低，可以根据时间来观察软件失效率的历史变化，进而估计当前软件系统的可靠性和未来完成规定目标所需的时间。软件可靠性模型是软件可靠性预测评估的基础，它根据所选模型关联参数的统计估计来确定系统失效情况和完成可靠性目标所需时间。软件可靠性模型的建立可以有效地预测系统的失效行为，它对资源计划、进度安排和软件维护等也有重要的意义。

而软件可靠性模型又是软件可靠性估测的核心，尽管它尚处于发展期甚至可以说还处于启蒙阶段，还有很多的问题需要研究解决。但是，自从第一个软件可靠性模型出现以后，各种各样的模型相继涌现，经过几十年的发展，在软件可靠性模型的理论研究方面有了很大的进步，有效地提高了软件的可靠性设计、测试及其可靠性工程管理能力，推进了软、硬件可靠性工程的均衡发展，提高了系统的可靠性水平。

SRE使用的模型有可靠性结构模型（反映系统结构逻辑关系的数学过程，在掌握软件单元可靠性特征的基础上，对系统的可靠性特征及其发展变化规律作出评价）和可靠性预计模型（本质上是描述软件失效与软件错误的关系，软件失效与运行剖面的关系的数学方程，借助这类模型，可以对软件可靠性做出定量的评估或预计）。基于时间域的软件可靠性增长模型是可靠性预计模型的重要类型之一，而非齐次泊松过程(Non-Homogeneous Poison Process，简称NHPP)类模型则是应用最广、影响最大的一类模型，这类模型拟合效果好，结构和应用最简单，已经成为软件可靠性工程实践活动中很成功的工具。软件可靠性模型是软件可靠性评估和预测的核心，它是根据软件可靠性测试收集到失效数据对软件可靠性进行评估的工具，对保证软件质量有重要意义。

尽管现有的NHPP类软件可靠性增长模型很多，但模型仍具有一定的局限性，能普遍应用的并不多。对于每个软件可靠性模型，都是以某些假设为基础，这些假设是模型建立的主要依据，也是模型在理论上进行处理的先决条件，由此可见，模型的成功与否，与建模的假设有着很大的关系。模型假设的局限性太多，就会影响到它们的应用范围。目前，软件工程界对软件可靠性模型的诸多疑虑，也多半来自于此。如何改进这些模型，是软件可靠性今后理论研究的重大课题之一。而它的突破，也将会消除软件工程界的疑虑，使软件可靠性理论得到更广泛的应用，从而，必然反过来又促进软件可靠性理论的发展。另外，现有模型虽有

很多，但是这些软件可靠性模型没有普遍适应性，一个模型可能仅对一个或几个软件做出较为准确的评估和预计，因此，如何选择、评价模型或者建造一个适用范围更广泛的统一的模型框架就成为一个很有必要研究的课题。

1.2 武器装备软件系统中软件可靠性技术应用趋势

针对军用软件特点和武器装备通用质量特性等有关要求，分析研究国内外软件可靠性技术发展现状和标准；重点开展软件可靠性分析，软件可靠性设计和实现，软件可靠性测量、测试和评价，软件可靠性管理等技术和方法研究；可靠性是军用软件研制工作中的薄弱环节，急需结合实际情况，研制军用软件可靠性工作要求标准，指导军用软件开发和使用工作，提升军用软件可靠性。

武器装备软件系统设计时更注重系统层次上的可靠性和安全性的综合分析和设计，在软件可靠性分析、设计、测试、验证方面有下述趋势：

（1）更加注重软件可靠性与安全性分析和设计的系统性

主要表现在以下两个方面：

1）分析范围进一步扩展，在原有分析的基础上加强系统级的分析与设计，更注重软件与其外部运行环境之间的相互作用的分析；

2）注重对软件运行异常环境的分析以及异常情况下软件处理功能的分析一设计：包括硬件异常状态、时序和软硬件异常交互的分析，软硬件综合容错分析与设计、软件降级分析与设计。

美国麻省理工学院系统与软件安全性项目组经过对大量的与软件相关的事帮进行了统计分析发现，几乎所有与软件相关的事故都涉及到软件需求问题。而且。软件作为一种逻辑产品，其失效模式与硬件失效模式不同。很多事故发生时，操作人员操作正确，硬件也未出现故障，从软件工程的角度看，软件符合“软件需求规模说明”，也没有失效。导致故障的原因在于，软件与外部运行环境之间出现了一种超出设计人员设想的相互作用方式，也就是说与软件相关的大部分系统失效是由于软件对外部输入处理及相关时序的设计遗漏造成，而非软件失效造成。为此，NASA定义了一种软件交互失效模式，并规定全部的安全性关键软件和任务关键软件均需在系统、功能等顶层设计加强软件与外部环境的交互动态分析。

在缺少软硬件容错设计的情况下，软件的输入错误必然导致软件的输出错误。由于许多实时嵌入式系统的输入来自于外部的硬件或软件，硬件故障或外部的干扰很可能造成外部环境的变化或输入信号时序、幅度的变化，也可能产生输入信号错误，为避免由于这种原因造成的软件失效，必须在软件研制前期进行深入、细致的分析，并采取针对性的设计措施。

（2）强调量化风险控制，注重对研制的软硬件复合系统的概率风险进行评估。

1986年挑战者号失事后，挑战者事故调查委员会批评NASA未能估计出每个组件失效的风险传统的安全性定性分析方法不足以预计或消弱全部的安全风险，有关学者建议概率风险评价方法应尽可能早的应用于飞机的风险管理程序。早期的概率风险评价研究与试点工作由专业研究人员进行。1995年4月，NASA在NFG7120.5A《NASA程序和项目管理过程与要求》中规定，概率风险分析应作为保证程序和技术成功的一种决策工具，要求程序与项目管理决策必须在概率风险排序的基础上进行。2002年11月，NASA在NFG 7120.5B规定NASA独立验

证与确认机构负责全部的安全性关键软件和任务关键软件全生命周期各阶段产品的独立确认与验证工作，包括软件可靠性与安全性的分析，测试与验证，以及软件概率风险危险分析与评价工作。

概率风险分析是系统应用可靠性和安全性等相关技术的一种综合分析方法，包括对软件可靠性和安全性的量化分析与评价，其目的是识别与评价为保证安全性和任务完成所需采取的各种行动、措施的风险，为决策提供支持。首先，需使用定性分析方法，如初步危险分析PHA，危险与可运行性研究HAZOP，故障模式、影响及危害性分析FMECA，系统检查单，主逻辑框图等技术对软硬件组合系统进行分析，获得可能导致系统不期望状态发生的初始事件表。在定性方法不足以提供对失效、后果、事件的充分理解时使用定量的方法。然后就系统、人、软件对初始事件的不同响应而导致的事件链的不同发展过程进行分析鉴别，生成系统的功能事件序列图。然后，分析各事件的发生概率（包括共因分析，人因分析以及软件各种失效分析等），用故障树和概率统计技术归纳各事件序列最终状态的发生的概率，分析各终结状态的严重度，结合状态发生概率与严重度，获得概率风险描述与风险排序。

NASA使用概率风险分析技术进行风险管理，按照风险调整资源，使资源的占用与风险相匹配，在不增加风险的前提下，减少44%的资源占有率。火星采样返回项目Mars Sample Return Mission要求任务失效概率必须小于10^{-6}，NASA采用概率风险分析方法检验系统的的可靠性，取得了很好的效果。

（3）产品验证更加注重分析技术和测试技术的综合应用

测试验证的不充分性与高成本决定了这一趋势的必然性。在系统的功能分析和设计阶段加强仿真验证与分析：对常规状态下的功能验证以实际测试验证为主，分析验证为辅；对导演状态下的功能验证以仿真验证为主，分析验证为辅；对软件小概率失效和软件危险失效则以分析验证为主，仿真验证为辅。

（4）产品可靠性分析与测试技术紧随新技术的应用发展趋势

装备软件产品的研发技术随着人工智能、大数据、5G技术、云计算等新技术的发展在不同软件产品中应用，软件可靠性的分析方法与测试技术也紧扣这些软件产品的特点进行演变发展，新的软件可靠性测评技术需求正在不断满足新一代武器装备软件系统质量的要求。

2 软件可靠性基本概念

随着计算机和信息处理的广泛应用，计算机系统的可靠性问题越来越得到人们的关注。而软件体系规模的日益增大及其复杂性的日益增强，使软件的可靠性问题更为突出。所谓软件可靠性是指软件系统在规定环境下、给定时间内无故障运行的概率，是软件质量的一个重要组成部分。软件可靠性模型是软件可靠性评估与预测的核心，目前，公开发表的软件可靠性模型已经有一百多种，但是，由于它们假设不同，性能各异，因此，即使用它们估测同一软件也可能得到存在巨大差异的预测结果，至今仍然没有一个既简单又广泛适用的软件可靠性模型。

建立软件可靠性模型旨在根据软件可靠性相关测试数据，运用统计方法得出软件可靠性的预测值或估计值。软件可靠性模型对于预防软件故障，预估软件性能都有积极意义。软件失效总体来说随着故障的检出和排除而逐渐降低，在任意给定的时间，能够观测到软件失效的历史[3]。软件可靠性建模的目标如下：1）预测软件系统达到预期目标所还需要的资源开销及测试时间；2）预测测试结束后系统的期望可靠性。

2.1 软件可靠性的度量指标

2.1.1 软件可靠性的定义

1983年美国IEEE计算机学会对“软件可靠性”作出了明确定义，此后该定义被美国标准化研究所接受为国家标准，1989年我国也接受该定义为国家标准。该定义包括两方面的含义：

（1）在规定的条件下，在规定的时间内，软件不引起系统失效的概率。该概率是系统输入和系统使用的函数，也是软件中存在的故障的函数，系统输入将确定是否会遇到已存在的故障（如果故障存在的话）。

（2）在规定的时间周期内，在所述条件下程序执行所要求的功能的能力。

在定义中, “规定的条件”主要是指软件的运行(使用) 环境。它涉及软件运行所需要的一切支持系统及有关的因素,如支持硬件、操作系统及其他支持软件等。“规定的时间”是由用户和软件实际使用要求确定的,但是在这里需要明确的一点是,规定的时间是指软件的运行时间,而不是普通意义上的日期时间。

根据定义可以得到，软件可靠性主要包含了如下三个要素：

1) 规定的条件

规定的条件是指两个方面，第一个方面是软件的运行环境，它主要包括软件系统运行时所需要的支持条件，如支持的硬件、操作系统、输入数据的格式和输入范围、操作规程以及支持软件等。第二个方面是指软件操作剖面——软件运行的输入空间极其概率分布。软件的可靠性在不同的环境条件下是不同的，具体来说就是规定的环境条件描述的主要是软件系统在运行时对计算机的配置和输入数据的要求，并且假定其它所需的一切因素都是理想的。

2) 规定的时间

这里“规定的时间”主要有种度量方式：1) 日历时间(Calendar time)指日常使用的时间变量；2)时钟时间(Clock time)，指程序从开始运行到结束所用的时、分、秒，其中还包括等

待其他程序运行的时间；3)执行时间(Execution time)指程序在执行时，实际占用处理器的时间。具体使用什么时间作为软件可靠性的量化取决于采用哪个可靠性模型为软件做可靠性评估。

3) 规定的功能

软件的可靠性还与规定的任务和功能有关。对于同一个软件来说，规定要完成的任务不同，则软件的运行剖面不同，调用的子模块（即程序路径）也不同，其可靠性也就可能不同。所以，要准确度量软件系统的可靠性必须首先要明确它的任务和功能。

从上述软件可靠性的定义可以看出，软件如何能无故障运行并使其功能更好地满足用户的需求是软件可靠性的关键，而软件可靠性又与软件中的缺陷(defect)直接相关，所以接下来介绍几个与缺陷有关的概念：错误(error)、故障(fault) 和失效(failure)。

软件错误是指人们在设计和开发软件时所产生的缺陷，这些错误的原因可能有以下几个方面：需求转换错误、设计错误、编码和逻辑错误等。这些错误不一定能被终端用户观察或检测出来。在实际的项目中，错误的总数是不可能知道的，只能通过检测出来的错误数来估计得到。

当软件运行时，如果某些输入对应的输出结果不正确，则认为该软件出了故障。所以软件故障是软件错误的表现，是一种动态行为，是程序代码中能引起软件的一个或多个功能模块不能执行所要求功能的错误，因而它是由用户通过某种形式检测出来的。总之，故障总是由错误引起，但是一个错误不一定能引起故障。

失效是指软件运行时不能履行特定的功能而导致软件系统功能的失败，它是软件故障运行时产生的后果。软件失效的本质是软件故障，也就是说所有的软件失效都是由软件故障引起的，但不是所有的软件故障都能引起失效。

从上述解释可以看出：对于需求实现的功能来说，软件中存在的错误是导致软件出现故障乃至产生失效的原因。在软件可靠性的研究中，通常要选取一些软件可靠性指标，以便对软件可靠性进行度量评估。

软件可靠性与硬件可靠性相比体现出了不同的特点：

1）软件错误主要是由设计错误造成的，使用和维护对软件可靠性的影响不大；

2）软件错误不随使用时间的延长而增加；

3）软件修理就是通过再设计排除原有错误；

4）软件可靠性增长与软件使用的时间无关，与检测并改正错误的努力有关；

5）软件寿命期内无老化与耗损现象；

6）正确的软件不因使用环境的变化而改变其正确性；

7）同一软件的冗余技术不能提高软件的可靠性；

8）采用高可靠性的标准件－系统软件所提供的各种库函数，未必能提高整体软件的可靠性。

2.1.2 常见软件可靠性度量指标

软件可靠性是从用户使用的角度对产品的质量进行评定的。软件可靠性指标就是指其可

靠性所应达到目标值的规定。在对于软件可靠性的表述上, 软件可靠性指标比较多，下面主要从技术的角度列举几个关键的指标：

初始故障数：是测试开始时软件中存在的故障数，这只是一个理想值，只能通过一定的方法进行评估，如软件可靠性模型。

剩余故障数：经过测试和故障修复排除后，仍然残留在软件中的故障数。它也是一个理想值，通常可以利用测试出来的故障数和软件可靠性模型进行评估，这种可靠性指标比较直观。

设随机变量T表示从软件运行时间（$t=0$）开始到系统失效所经历的时间，则在时刻t的软件可靠度$R(t)$定义为: 在规定条件和规定时间t 内完成规定功能的概率，即软件产品正常工作时间T 大于规定时间t 的概率：

$$R(t)=P\{T>t\} \tag{2.1}$$

不可靠度(失效分布函数、累积失效概率)：软件产品在规定条件和规定时间t内丧失规定功能的概率，即软件产品正常工作时间T小于或等于规定时间t的概率，即

$$F(t)=P\{T\leq t\}=1-R(t) \tag{2.2}$$

对要求在规定时间内能无失效工作比较高的系统，可选可靠度作为软件可靠性参数，如指挥控制系统软件。

失效密度函数：失效密度函数$f(t)$是累积失效概率的导数，即：

$$f(t)=\frac{dF(t)}{dt} \tag{2.3}$$

软件失效率(Software Failure Rate)，又称为风险函数(Hazard Function)，是指软件在t时刻尚未发生失效的条件下，在t时刻之后的单位时间$(t,t+\Delta t)$ 内发生失效的概率，记为$z(t)$，即软件在单位时间内t时刻的产品瞬态失效概率：

$$\begin{aligned} z(t) &= \lim_{\Delta t\to 0}\frac{Pr(t+\Delta t\geq T>t|T>t)}{\Delta t} && (2.4)\\ &= \lim_{\Delta t\to 0}\frac{Pr(t+\Delta t\geq T>t)}{\Delta t\cdot Pr(T>t)} && (2.5)\\ &= \lim_{\Delta t\to 0}\frac{F(t+\Delta t)-F(t)}{\Delta t\cdot R(t)} && (2.6)\\ &= \frac{f(t)}{R(t)} && (2.7) \end{aligned}$$

由于$f(t)$是随机变量T的概率密度函数，因此，上式可以表示为

$$z(t)=-\frac{R^{'}(t)}{R(t)} \tag{2.8}$$

在初始条件$R(0)=1$下,求解该常微分方程可得到

$$R(t)=e^{-\int_0^t z(t)dt} \tag{2.9}$$

失效强度(Software Failure Intensity)是指单位时间内软件失效的机会或可能。软件在时间区间$(t,t+\Delta t)$上失效数的期望与这个时间长度Δt 之值，在$\Delta\to 0$的极限值。例如，在非

齐次泊松过程(NHPP)模型中，假定软件在t时刻发生的失效数为$N(t)$, 显然在t给定的条件下$N(t)$是一个随机变量，且随时间t的变化而变化，即$\{N(t), t \geq 0\}$为一随机过程。设$m(t)$为随机过程的均值函数，即

$$m(t) = E[N(t)] \tag{2.10}$$

则软件在t时刻的失效强度定义为：

$$\lambda(t) = \frac{dm(t)}{dt} \tag{2.11}$$

从失效率和失效强度的定义可以看出，失效率是从寿命分布的角度定义，是一个条件概率，而失效强度则是基于随机过程定义，是失效次数均值的变化率。可以证明，在软件的稳定运行期，不对软件作任何修改的条件下，软件的失效强度应为一个常数，此时对应的失效过程为一个齐次泊松过程（HPP）,其失效间隔服从参数为λ的指数分布，任一时间点上的失效率均为λ。

实际上，可以证明，若软件失效计数过程$\{N(t), t \geq 0\}$是一个泊松过程，条件失效率函数$z(\Delta t|t_i)$和失效强度函数$\lambda(t_i + \Delta t)$ 是相同的，即有

$$z(\Delta t|t_i) = \lambda(t_i + \Delta t), \Delta t \geq 0 \tag{2.12}$$

其中，t_i为第i次失效的累计时间。此时，这两个概念是可以通用的。

一般地，若软件失效计数过程$\{N(t), t \geq 0\}$不是一个泊松过程，则失效率和失效强度是不相同的，两个概念是有区别的，例如在经典的JM(Jelinski-Moranda)软件可靠性模型中，失效率为

$$z(\Delta t|t_i) = \phi(N - i), \Delta t \geq 0 \tag{2.13}$$

而失效强度则为

$$\lambda(t_i + \Delta t) = N\phi e^{-\phi(t_i+\Delta t)}, \Delta t \geq 0 \tag{2.14}$$

其中，N为从观测开始之后，软件的失效个数；ϕ为一比例常数，t_i为第i次失效时间。显然。失效强度和失效率是不同的。

通常情况下,可靠性高的武器装备软件系统一般都具有的特点有:故障的发生次数少、正常工作时间长、在故障发生后需要的修复时间短. 在长期的工程实践中,工程人员总结了 3 个时间指标来衡量武器装备软件系统可靠性:平均无故障时间、平均故障修复时间和平均故障间隔时间,这些指标体现了系统在规定时间内保持正常运行状态的能力，这几个时间之间的关系如图2.1所示.

图2.1中,T_1表示系统正常工作时间，T_2表示系统处于故障状态时间, T_3表示故障修复时间。

1)平均故障间隔时间

平均故障间隔时间(mean time between failure, MTBF)是指系统发生相邻2次故障所经历的时间间隔的平均值，是系统寿命T的期望，即

$$MTBF = \int_0^\infty tf(t)dt = \int_0^\infty R(t)dt$$

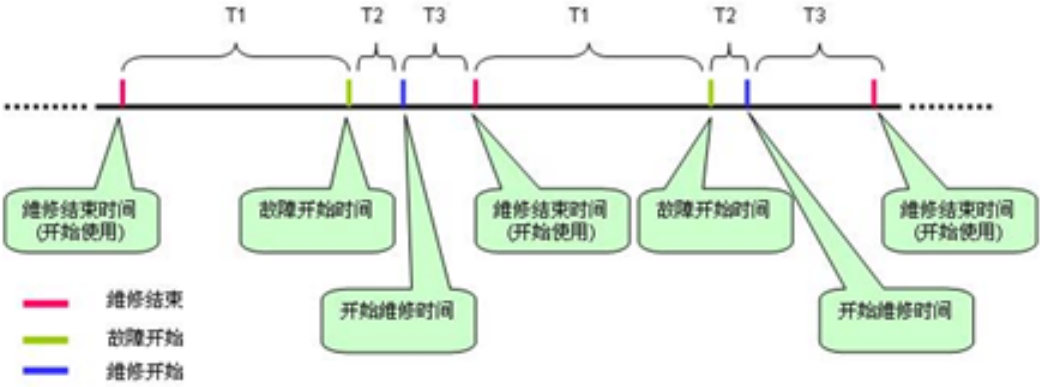

图 2.1 图解MTBF、MTTF和MTTR三个时间关系

系统的平均故障间隔时间越长表示系统具有较高的可靠性并且具有较强的正确工作性能.

2)平均无故障时间/平均失效前时间

平均无故障时间(mean time to failure, MTTF)是指系统从开始正常工作到故障发生时所经历的时间间隔的平均值,即系统无故障运行的平均时间. 假设当前时间到下一次失效的时间为X, X具有分布函数$F(t) = Pr(X \leq t)$，即可靠度函数$R(t) = 1 - F(t) = Pr(T > t)$，则有

$$MTBF = \int_0^\infty tf(t)dt = \int_0^\infty R(t)dt$$

系统的平均无故障时间或者平均失效前时间越长,说明系统的可靠性越高.

一般地，在硬件可靠性中，$MTTF$用于不可修复产品，$MTBF$用于可修复产品；由于软件不存在不可修复的失效，即软件是可修复的，因此，目前在软件可靠性建模中，一般未加以区别。

易知，当失效时间T服从指数分布时，$MTTF$和失效率存在倒数关系；当失效率不为常数，而是随时间变化而变化时，如测试、调试阶段，$MTTF$和失效率之间不存在倒数关系；对于某些软件可靠性增长模型，$MTTF$是不存在的，例如，Musa的基本执行时间模型，发生了i次失效后的可靠度函数为

$$R(\Delta t|t_i) = e^{-[\beta_0 e^{-\beta_1 t_i}][1-e^{-\beta_1 \Delta t}]} \tag{2.15}$$

显然，$MTTF$不存在。

3)平均故障修复时间

平均故障修复时间(mean time to repair, MTTR)是指从系统发生故障到故障维修结束并且可以重新正常工作所经历的时间间隔的平均值. 系统的平均无故障时间越长,意味着系统的恢复性能越好.

4) 可用性

软件可用性(Software Availability)是指软件在任一随机时刻需要开始执行任务时，处于工作或可使用状态的程度，即“开则能用，用则成功”的能力。可表示

$$A = f(R, M) = \frac{MTTF}{MTTF + MTTR} \tag{2.16}$$

5）故障检测率

故障检测率(fault detection rate, FDR), 即故障检测率函数,表示当前时刻单位时间内单个故障被检测到的平均概率,或每个故障的查出率,通常用$b(t)$来表示. 假定NHPP类型的SRGM 用一个计数过程$\{N(t), t \geq 0\}$表示到时刻t为止检测到的软件故障累计数.如果故障累计数的期望值函数用$m(t)$表示,则故障检测率定义为

$$b(t) = \frac{\lambda(t)}{a(t) - m(t)} \tag{2.17}$$

其中$a(t)$表示基于时间的潜伏故障总数的函数,即到时刻t已排除的软件故障数和潜伏在软件中尚未被发现的软件故障数的和。

FDR 表征了测试环境下故障被查出的效率,具有描述综合测试策略的能力,因而与包括测试人员、测试技术、测试工具等构成的整体测试环境紧密相关.

从早期提出将FDR 看作常数的软件可靠性增长模型,到提出整体呈现递减趋势的幂函数类型FDR,再到能够基本刻画测试环境平缓变化的S 型FDR,以及(复杂)指数类型FDR 的研究,整体上,对FDR 的研究方向呈现出贴近工程实际化特点,因而能够更好地描述测试环境的改变,帮助提高可靠性模型的性能.

2.1.3 武器装备软件系统可靠性指标

而结合武器装备特点的软件可靠性参数则有:

①成功率

软件的成功率是指在规定的条件下软件完成规定功能的概率。某些一次性使用的系统或设备, 如弹射救生系统、导弹系统的软件,其可靠性参数即可选用成功率。

②任务成功概率

任务成功概率是指在规定的条件下和规定的任务剖面内,软件能完成规定任务的概率。如军事飞行任务等,人们有理由关心其成功完成任务的概率,此时,即可选择任务成功概率作为软件可靠性参数。

③由平均失效前时间派生的参数

对于不同的武器装备可派生出不同的软件可靠性参数,例如,飞机、宇宙飞船的系统可用平均失效前飞行小时; 坦克、车辆系统可用平均失效前里程;舰艇指控系统可用平均失效小时数。

④平均致命性失效前时间

平均致命性失效前时间是指仅考虑致命失效的平均失效前时间。所谓致命性失效是指使系统不能完成规定任务的或可能导致人或物重大损失的软件失效或失效组合。例如,飞机、宇宙飞船的系统可用平均致命失效前飞行小时;坦克、车辆系统可用平均致命失效前里程;舰艇指控系统可用平均致命失效小时数。

2.2 软件可靠性建模方法

2.2.1 软件可靠性建模思想

软件可靠性建模旨在根据可靠性数据和软件结构信息以统计的方法给出软件可靠性的估计值或预测值，从本质上理解软件可靠性行为，是软件可靠性工程的基础。

当前的软件开发多数是基于组件或模块进行的，但软件整体并不是各个模块的简单相加，每个模块都可能对全局的可靠性产生不同的影响。软件可靠性分析利用软件的整体架构、软件运行剖面、以及每个模块的可靠性搭建数学模型，进而分析软件整体的可靠性。模型一方面可以在设计阶段根据架构、控制流、已知组件可靠性数据或预估的可靠性对软件整体的可靠性进行预测；另一方面也可以应用在软件的验证和运行阶段，通过各种测试方法得到每个模块的可靠性和实际运行环境中的使用剖面，再代入数学模型得到软件的可靠性，从而评估软件是否达到了预期的要求。

建立软件可靠性模型旨在根据软件可靠性相关测试数据，运用统计方法得出软件可靠性的预测值或估计值，图2.2给出了软件可靠性建模的基本思想。

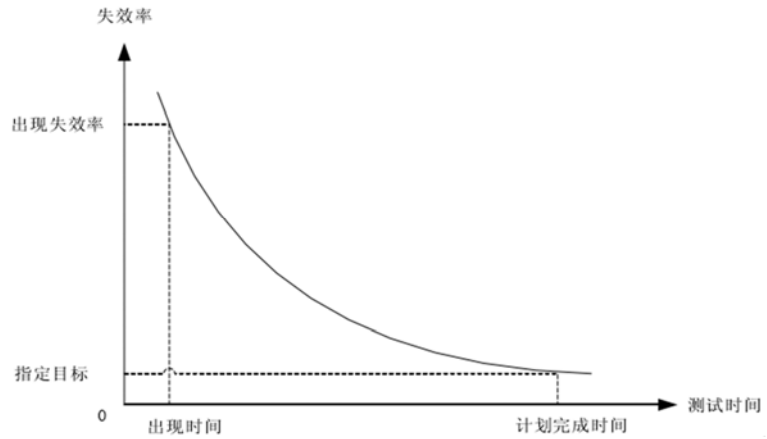

图 2.2 软件可靠性建模基本思想

从图2.2中可以看出软件失效总体来说随着故障的检出和排除而逐渐降低，在任意给定的时间，能够观测到软件失效的历史。软件可靠性建模的目标如下：

（1）预测软件系统达到预期目标所还需要的资源开销及测试时间；

（2）预测测试结束后系统的期望可靠性。

2.2.2 软件可靠性建模过程

为了满足软件可靠性指标要求，需要对软件进行测试-可靠性分析-再测试-再分析的循环迭代过程，软件可靠性建模的目的是为了对软件中的失效趋势和软件可靠性进行有效的估计与预测分析，来判断软件是否达到可靠性要求水平。软件可靠性建模是根据软件过去的失效行为，建立模拟软件可靠性行为的数学模型，然后对这种可靠性行为进行估计和预测，建模过程如图2.3所示。

软件可靠性建模过程通常由以下几个部分组成：

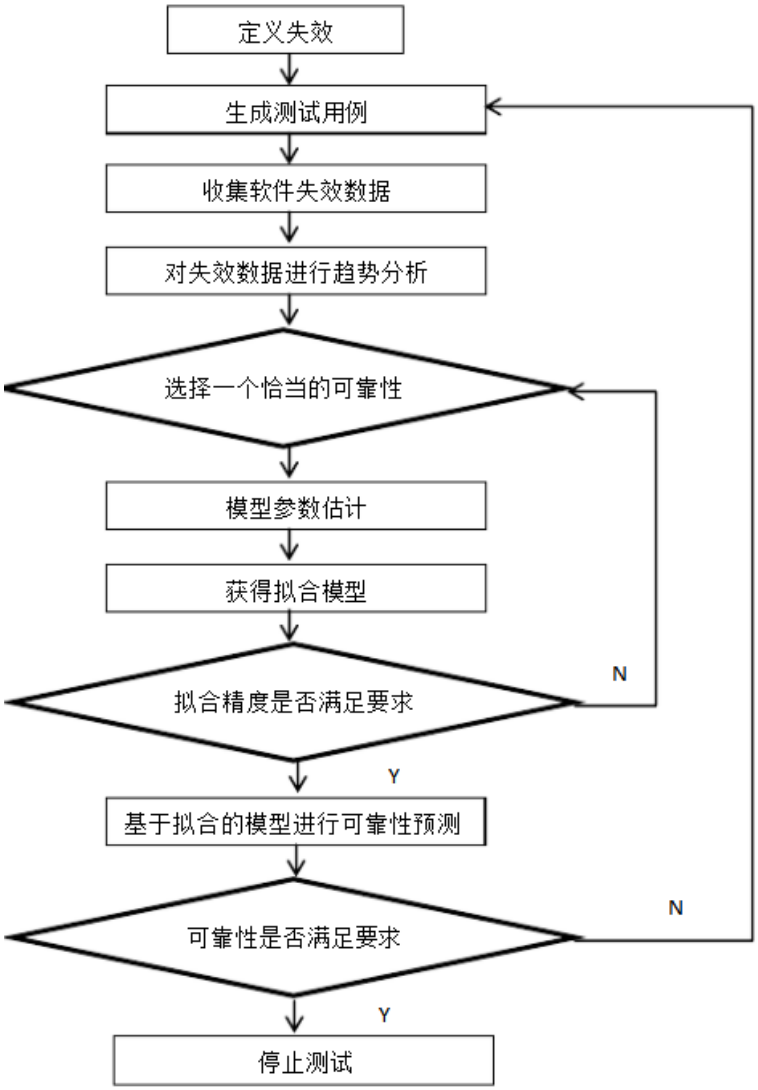

图 2.3 软件可靠性建模过程

（1）在进行软件可靠性建模时都要先做一些假设，主要原因有两个：一是人们还无法确知软件可靠性行为中的某些特征，或者是某些特征具有不确定性；二是为了数学上处理方便；

（2）通过对被测试软件进行测试，获得软件失效的历史数据；

（3）建立数学模型：对已经选择的可靠性度量即失效数据进行统计分析，并将其拟合成为表示软件产品某些特性的函数，如概率分布函数等；

（4）进行参数估计：对于一些通过模型不能直接获得的度量或者参数，使用参数估计的方法来确定它们的值；

（5）根据这种与软件历史失效数据相拟合的、度量参数确定的分布函数即软件可靠性模型，确定某些期望的可靠性度量值并估计软件将来可能发生的失效行为。

根据上述内容可知作为软件可靠性度量及可靠性建模的基础，必须要解决的基本问题如下：选择适合故障特点的分布函数、确定分布函数的某些特征值与软件可靠性指标之间的关

系、选择适合分布函数的参数估计方法。

2.2.3 软件可靠性建模基本问题

软件可靠性建模需要考虑以下基本问题：

（1）模型建立

模型建立指的是怎样去建立软件可靠性模型。一方面是考虑模型建立的角度，例如从时间域角度、数据域角度、将软件失效时刻作为建模对象，还可以将一定时间内软件故障数作为建模对象；另一方面是考虑运用的数学语言，例如概率语言。

（2）模型比较

在软件可靠性模型分类的基础上，对不同的模型分析比较，并对模型的有效性、适用性、简洁性等进行综合权衡，从而确定出模型的适用范围。

（3）模型应用

软件可靠性模型的应用需要从以下两方面考虑：一是给定了软件的开发计划，如何选择适当的模型；二是给定了软件可靠性模型，如何指导软件可靠性工程实践。软件系统的失效历史可以通过对测试得到的失效数据分析获得，而实际情况中，人们最为关注的是软件未来的失效趋势。软件可靠性模型基本都是建立在一定的假设基础之上，所以，即使花费了大量的时间和精力对软件的可靠性进行预计，也只是一种预测，这种预测的不确定性是许多未知原因交互作用的结果，根据软件可靠性模型的预测只能以概率形式表示。

2.2.4 软件可靠性模型的概念及特点

软件可靠性模型是为了预估软件的可靠性而建立的数学模型，它根据软件测试提供的失效数据来估算软件的可靠性，并对软件未来发生的失效行为进行预测，为软件开发过程的监督和软件过程的管理提供了保障。软件可靠性模型选择的依据是软件的可靠性指标，每个可靠性指标的侧重点不同，所以不同的可靠性指标就需要不同的可靠性模型来评估。

通常情况下一个软件可靠性模型由如下四部分组成：模型假设、性能度量、参数估计方法、数据要求。

（1）模型假设其实是对实际情况的简化或规范，因此总会包括若干假设，比如选取的测试用例代表实际运行剖面，不同软件失效的发生是独立的等。

（2）性能度量就是软件可靠性模型的输出（如平均无故障时间、故障率等），常以数学表达式给出。

（3）参数估计方法是在无法直接获得可靠性度量的实际值的情况下，根据收集的失效数据来估计模型中的未知参数而间接确定可靠性度量的值，通常所用的方法有：最大似然估计法、最小二乘法、Bayesian估计等。

（4）数据要求是指软件可靠性模型要求一定的输入数据，就是软件可靠性数据；不同的软件可靠性模型要求的软件可靠性数据的类型可能不同。

由于很多可靠性模型为了简化数学上的处理难度，在理论上做了不少假设，但其中一些假设与实际情况并不相符，所以在实际选择模型预估软件可靠性时要仔细研究，选择那些与