安全关键系统高可信保障技术的研究*)

杨仕平　熊光泽　桑　楠
(电子科技大学计算机科学与工程学院　成都610054)

Research on Ultradependability Safeguard Technology for Safety Critical Systems

YANG Shi-Ping　XIONG Guang-Ze　SANG Nan
(College of Computer Science and Engineering, UEST of China Chengdu 610054, China)

Abstract In order to improve dependability of safety critical systems, whose safeguard technologies are researched in this paper. Firstly, origins and meanings of dependability are expatiated, afterwards, respective meaning of and relations among fault、error、failure are analyzed. In succession, how to apply with fault prevention、fault tolerance、fault removal、fault forecasting to enhance dependability of safety critical systems are expounded concisely by diagrams. Because error detecting is very important in fault tolerant systems, an error detecting technology based on safety kernel is proposed in this paper. Future trends of research in safety critical field are listed at the end.

Keywords Safety critical, Real time systems, Ultradependability, Fault tolerance, Safeguard technology, Safety kernel

1 引言

安全关键系统SCS(Safety Critical Systems)是指系统功能一旦失效将引起生命、财产的重大损失以及环境可能遭到严重破坏的系统。这类系统广泛存在于航空航天、国防、交通运输、核电能源和医疗卫生等诸多安全关键领域中。而高可信(Ultradependability)则是指系统在任务开始时可用性给定的情况下，在规定的时间和环境内能够使用且能完成规定功能的能力，即系统"动则成功"的能力。随着现代社会的高速发展及不稳定因素的存在，安全关键系统日益庞大和复杂，带来了系统可靠性和安全性的下降、投资增加、研发周期加长、风险增加。安全关键系统的应用环境也更加复杂和恶劣，从陆地、海洋到天空、太空，安全关键系统的使用环境不断地扩展和更加严酷。严酷的环境对系统高可靠、高安全性等综合特性的实现提出了严峻的挑战。除此，系统要求的持续无故障任务时间加长，如太空探测器的长时间无故障飞行要求、通信网络的关键任务不停机要求等，迫使安全关键系统必须具有良好的可靠性、可维护性等专门特性。系统的高可信性与使用者的生命安全直接相关，如核能核电系统、载人航空航天器、高速列车等系统的可靠与安全是生命安全的基本保证，受到强烈的关注。市场竞争的影响，如"性能优良、功能齐全"并不是用户选择产品时考虑的唯一因素，产品是否可靠、是否好修、使用维护保养费用多少、寿命多长都对用户的选择产生重要的影响。对于研究开发者来说，总是希望投资小、周期短、研发一次成功、系统安全可靠——这既是目标也是矛盾所在，解决矛盾达到目标对安全关键系统开发设计人员而言无疑具有极大的挑战性。

2 可信性的起源与内涵

Babbage于1830's年在他的论文"计算机器"中首次提到了可信计算(dependable computing)的概念。第一代电子计算机出现在20世纪中期，那时的计算机是非常不可靠的，为提高计算机的可靠性，先驱们研究了大量切实可行的可信性保障技术，如错误控制码、复式比较、三逻辑表决、失效组件的诊断与定位等。而J. von Neumann、E. F. Moore和C. E. Shannon与他们的后继者则提出了冗余的思想，通过冗余技术则可用不可靠组件来建立可靠的逻辑结构，其基本思想是：不可靠逻辑组件的缺陷由于多个冗余组件的存在而被屏蔽掉了。屏蔽冗余理论被W. H. Pierce于1965年统一为失效容忍(Failure Tolerance)。1967年，A. Avizienis、Carter与Schneider等人则把屏蔽冗余理论连同错误检测、故障诊断、错误恢复等技术融入到容错系统中去。与此同时，国际上也成立了一些可信性研究机构专门研究高可信保障技术，如IEEE-CS TC于1970年成立了"容错计算"研究小组，IFIP WG10. 4于1980年成立了"可信计算与容错"研究小组，它们的成立加速了安全关键系统的发展。1985年Laprie正式提出可信性(dependability)以便与可靠性(reliability)相区别。简言之，可信性指系统在规定时间与环境内可交付可信服务的能力。可信性是一个复杂的综合性概念，如图1所示，它所包含的特征有：可用性、可靠性、防危性、安全性(1995年，Laprie把安全性Security细分为保密性与完整性)、可维护性。其中可用性表示系统在给定的时间内可运行的概率，它通常用来度量可延迟或短暂停止提供服务而不会导致系统发生严重后果的品质。可靠性表示系统在给定的时间内连续提供期望服务的能力。防危性表示系统在给定的时间内不发生灾难性事故的概率。保密性表示未经访问许可禁止访问系统敏感数据的能力。完整性指保持数据一致性的能力。可维护性指系统具有可修复和升级的能力。

3 缺陷、错误、失效及三者之间的关系

3.1 失效及失效模式

服务是指系统根据用户的输入或其它外部条件而进行的一系列操作。正确的服务是指正确实现系统功能的服务。失效

*)国防科技预研基金项目(2000J6. 7. 1. DZ0206)。杨仕平　博士研究生，主要研究方向：实时操作系统的防危核机制与实现、安全关键系统。熊光泽　博士生导师，教授，主要研究方向：实时计算机系统及应用，系统可靠性评测。桑　楠　副教授，主要研究方向：实时软件工程。

(Failure)指系统实际所交付的服务不能完成规定的功能或不能达到规定的性能要求,即正确服务向不正确服务的转换。系统失效则是指系统的实现未能与系统需求规范保持一致,或系统规范未能完全描述系统本身应具有的功能。失效的根源是由于系统(或子系统)内部出现了错误的状态,当错误到达服务界面并改变服务时便产生失效。缺陷是导致错误发生的根源,它一般处于静止状态,当缺陷产生错误时,称缺陷被激活。

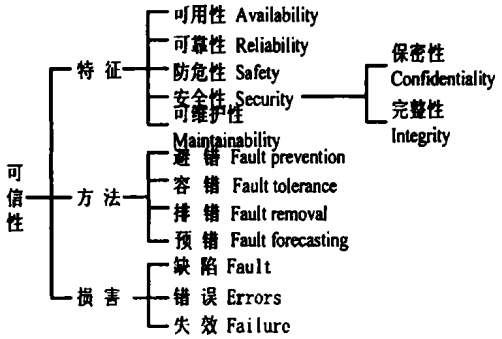

图1 可信性的特征、实现方法及损害

系统并非总是以同样的方式失效,其失效的方式被称为失效模式。如图2所示,失效模式可从三个观点来划分:1)失效的属性;2)系统用户对失效的主观观点;3)失效后果的严重性。系统失效具有多种属性,而各种属性的综合决定失效所属的失效模式。

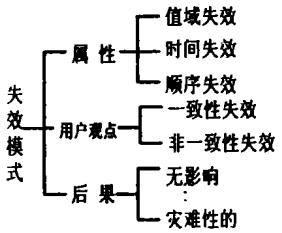

图2 失效模式的分类

3.2 错误及其分类

错误(Error)是指在一定的运行条件下,导致系统运行中出现可感知的不正常、不正确或未按规范执行的系统状态。而错误最终能否导致系统失效由系统组成、系统行为和应用领域决定。在设计高可信性的安全关键系统时,一个至关重要的问题是错误被检测的程度。错误完全被检测是不可能的,只能一定程度上被检测到,关于错误检测技术还有待深入研究。能被检测到的错误通常以消息或信号的方式出现,不能被检测到的错误称为潜在错误。若系统能够在其运行过程中检测并处理出现的错误,则系统行为能够避免错误导致系统失效。应用于不同领域的安全关键系统,错误产生的后果也不尽相同,根据系统在出现可检测错误后的症状,可以将错误分为:1)报警;2)降级服务;3)安全关闭(可进一步细分为部分或完全关闭、立即停止犯规机器而不关闭整个系统、当前行为一旦完成便立即停止。);4)安全导航;5)坠毁或崩溃。

3.3 缺陷及缺陷的组合

缺陷(Fault)是指因人为的差错或其他客观原因,使所设计的系统中隐含有不正确的系统需求定义、设计及实现。这些缺陷将有可能导致系统在运行中出现不希望的行为或结果。缺陷是造成错误出现的原因,其来源十分广泛。同时,由于缺陷具有多种不同的属性,因此具有多种不同的分类方法,其分类方法有:1)根据产生的表象原因,缺陷可以分为自然缺陷和人为缺陷,前者来自于客观物理世界,后者由于设计人员或操作人员的错误行为而产生。2)根据产生的意图,缺陷可以分为偶然缺陷和必然缺陷。偶然缺陷是由于无意识的行为而产生的,具有偶然性。必然缺陷是人类有意识的行为产生的缺陷,又可分为无恶意和恶意两种。例如,操作者错误地理解操作工序是无恶意的,而计算机病毒则是专门为攻击计算机系统而设计的。3)根据造成或产生的阶段,缺陷可以分为开发缺陷、生产缺陷和操作缺陷,开发缺陷在系统的开发阶段引入,如错误的需求分析、功能定义和结构设计等;生产缺陷是在产品生产制造的过程中引入的;操作缺陷是在系统的运行过程中被引入的,如操作人员错误地使用系统。4)根据缺陷所产生的范围,可分为物理缺陷和信息缺陷。前者主要来自于物理世界,如机械、电子元器件所产生的缺陷,后者主要来自于抽象的范畴,如数据定义、需求捕获等。5)根据所处的位置,缺陷可以分为内部缺陷和外部缺陷。系统自身内部的缺陷称为内部缺陷。外界环境引发系统发生错误的缺陷称为外部缺陷。6)根据其时间属性,可以将缺陷分为:永久缺陷和暂态缺陷。永久缺陷不会因为产生条件的消失而消失,一旦产生,则永远存在。暂态缺陷随产生条件的消失而消失。图3为缺陷的分类及其组合。在实际应用中,往往把缺陷分为物理缺陷、设计缺陷和交互缺陷。物理缺陷由硬件产生,随着硬件技术的不断成熟,这类缺陷所占比例逐渐减少。目前,系统的主要缺陷来源于软件设计缺陷,如对系统需求的错误理解或理解不完全、对系统可能出现的状态估计不足和错误的数据结构定义等。交互缺陷

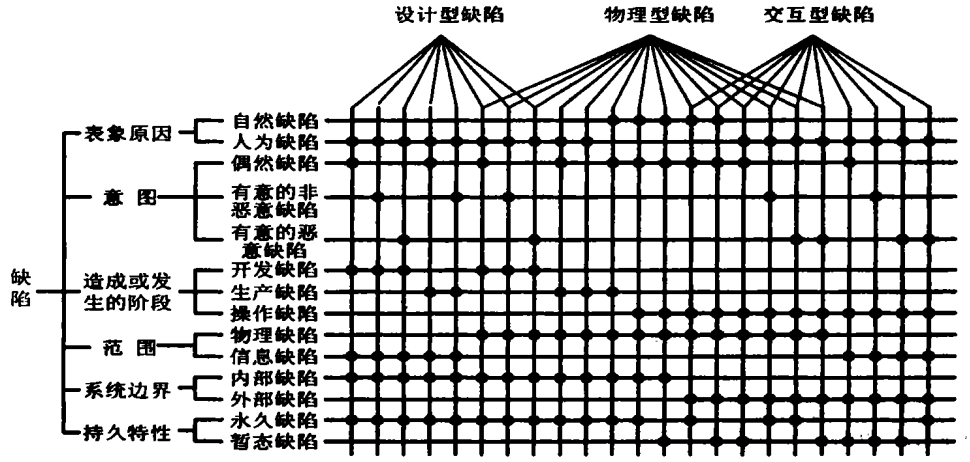

图3 缺陷的分类及其组合

则指不正确的操作或使用，其根源也可归属于设计缺陷。如何避免或减少设计缺陷将成为创建高可信安全关键系统的关键所在。

过去十五年国外所发生的重大灾难性事故	缺陷			失效		可用性	可靠性	防危性
	物理	设计	交互	本地	分布			
·1980年6月：北美防空系统错误报警	√			√		√	√	
·1981年3月：航天飞机的首次发射被推迟		√		√		√	√	
·1985年6月—1987年1月：Terac-25型医疗器械辐射量过度		√		√				√
·1990年1月15日：美国长途电话业务瘫痪9小时		√			√	√	√	
·1991年1月：爱国者导弹在海湾战争中误击巴格达的难民营		√	√	√		√	√	√
·1992年11月：伦敦的紧急救护服务通信系统崩溃		√	√		√	√	√	√
·1993年6月26日至27日：法国的信用卡系统认证遭到否决	√	√				√	√	√
·1996年6月4日：亚利安娜Ⅴ型火箭发射失败		√		√		√	√	√

图4　发生灾难性事故的安全关键系统

随着计算机控制技术在各个安全关键领域的不断应用，计算机所导致的灾难性事故也不断发生。图4为过去十五年国外所发生的重大灾难性事故，由图可知，失效的方式越来越趋于分布式系统，而设计型缺陷则是导致事故发生的主要根源。因此，解决分布式系统的设计型缺陷将成为最具挑战性的课题之一。

3.4　缺陷、错误及失效三者之间的关系

缺陷是产生错误的根源，但并非所有缺陷都能产生错误。通常，缺陷处于静止状态，当缺陷由于系统或子系统在特定环境下运行而被激活时，将导致系统或子系统进入错误的状态，当一个或多个错误进一步在系统或子系统中传播并到达服务界面时，将导致系统或子系统失效，图5为缺陷、错误及失效三者之间的关系。

4. 安全关键系统的高可信保障技术

由于安全关键系统极为复杂，而其可信性保障技术目前还很不成熟，因此用现有的可信性保障技术创建永不失效的安全关键系统是不现实的。为此，我们只可能创建失效概率足够低的安全关键系统，以满足其关键应用的需求，即使系统的可信赖程度需要达到所要求的水平。对于不同的应用，对系统的可信度需求也不同，采取的可信性保障技术也不同。由前可知，导致系统失效的因素来源于系统生命周期的每个阶段，分布在系统的内部及其运行环境中。若缺陷所引发的错误不能被及时地处理，则有可能造成系统失效。可信性保障技术就是尽早、尽可能多地发现和处理缺陷、错误，将系统失效的概率

图5　为缺陷、错误及失效三者之间的关系

尽可能地降低。由前可知，导致错误发生的根源是由于各种缺陷的存在，在图3中本文对缺陷作了详细的分类，这样针对不同类型的缺陷和错误，安全关键系统可采取不同的可信性保障技术，即"对症下药"。为提高安全关键系统的可信性，本文把其切实可行的高可信保障技术总结为图8。由图8可知，安全关键系统的高可信性保障技术大致可分为避错、容错、排错及预错四种，其中：

安全关键软件设计技术/方法	SIL1	SIL2	SIL3	SIL4
1. 故障检测与诊断	—	R	HR	HR
2. 错误检测与校正码	R	R	R	HR
3a. 失效判定程序	R	R	R	HR
3b. Safety Bug 技术	—	R	R	R
3c. 多版本程序	R	R	R	HR
3d. 恢复块	R	R	R	R
3e. 反向恢复	R	R	R	R
3f. 前向恢复	R	R	R	R
3g. 重试故障恢复机制	R	R	R	HR
3h. 存储已执行用例	—	R	R	HR
4. 优美降级	R	R	HR	HR
5. 智能故障校正	—	NR	NR	NR
6. 动态重配	—	NR	NR	NR

图6　安全关键软件的结构设计

避　错　这一技术主要运用于系统的设计和维护阶段。在系统的设计阶段，从需求分析、系统定义、系统设计到代码编制，每个步骤都必须最大限度地保证其合理性和正确性，以避免缺陷的引入。因此，采用高可信的软件工程方法和高水平的设计人员是必不可少的，如图6、图7所示为 IEC 1508所推荐的使用两阶段设计方法来设计高可信的安全关键软件，其中 SIL 表示系统完整性级别，完整性级别越高，系统的可信性也越高。图中"R"表示推荐使用，"HR"表示强烈推荐使用，"NR"表示不推荐使用，"—"表示可自由选择。

安全关键软件的详细设计	SIL1	SIL2	SIL3	SIL4
1. 结构化方法：JDS，MASCOT，SADT，SSADM 及 Yourdon	HR	HR	HR	HR
2a. 计算机辅助设计工具	R	R	HR	HR
2b. 半形式化方法	R	HR	HR	HR
2c. 形式化方法 CCS，CSP，HOL，LOTLS，OBJ，VDM 及 Z	—	R	R	HR
3. 防御性编程	—	R	HR	HR
4. 模块化方法	HR	HR	HR	HR
5. 设计与编码标准	R	HR	HR	HR
6. 可分析性程序	R	HR	NR	HR
7. 结构化编程	HR	HR	HR	HR

图7　安全关键软件的详细设计

容　错　容错是一种通用的可信性保障机制，其目的是使系统在出现错误时能够继续提供标准或降级服务。容错技术能够处理多种类型的缺陷和错误，如硬件设计缺陷和软件

设计缺陷。通常，容错被分为硬件容错、软件容错和系统容错。硬件容错用以避免由于硬件造成的系统失效。软件容错用以避免软件造成的系统失效。系统容错用以避免运行环境造成的系统失效。随着软件应用的飞速膨胀，以及硬件可靠性的增长和硬件容错技术的成熟，软件错误已经成为影响系统可信性的最主要的因素。在具有硬件容错能力的计算机系统中，其65%的失效来自于软件，仅有8%来自于硬件。因此，软件的容错能力成为决定安全关键系统可信性的关键。软件容错设计的主要方法有：信息容错、时间容错及软件结构容错。而任何一种容错方法都包含错误检测、错误处理、错误恢复三个过程，实现每一过程中的可用方法如图6所示，具体含义请参考相关文献。鉴于错误检测是容错系统的关键所在，本文将在第5节中讨论一种基于防危核(safety kernel)的错误检测机制。

排　错　这一技术通常应用于系统的测试和维护阶段。通过模拟真实工作环境进行系统测试，发现错误并分析产生错误的原因，然后改进系统以消除、减少产生错误的原因。由于现有的测试方法其测试覆盖率有限，因此不可能穷尽所有的错误，用测试的方法来创建高可信的安全关键系统是不现实的，通过测试只可能降低系统发生故障的概率。在系统的维护阶段，维护人员可以根据系统的错误报告，减少产生错误的缺陷，然而必须注意的是维护操作不能引入新的缺陷。排错对始终处于休眠状态的缺陷无效，因此维护人员可对系统进行重新配置或升级以便排除尚处于休眠状态的缺陷。

预　错　在系统的运行过程中，系统可以通过分析当前所获得的系统状态信息，预测可能发生的错误，并采取措施加以避免。这一技术必须依靠正确的系统状态分析，也是该类技术的最难以解决的问题。

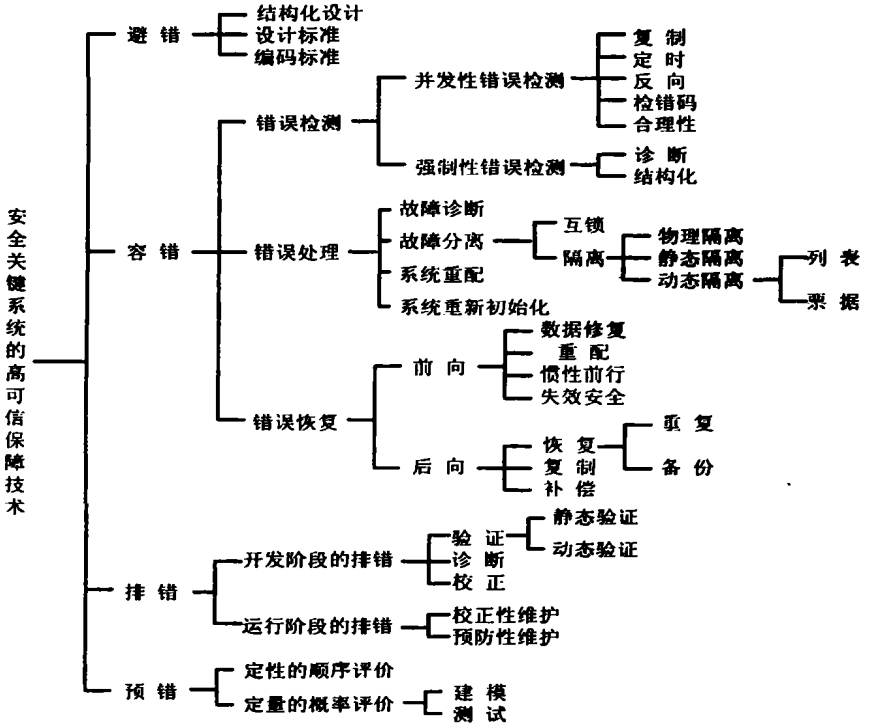

图8　安全关键系统的高可信保障技术

图8中的每种可信性保障技术均具有多种不同的实现形式，其成本、效能各不相同。图9则表示了四种可信性保障技术在安全关键系统开发过程中不同阶段的应用情况。在具体的工程实际中，设计人员首先根据系统的应用领域、功能和性能需求、成本限制和资源限制等诸多因素，确定系统的失效语义。然后，根据失效语义，与多种合适的可信性保障手段相结合，处理在系统生命周期的不同阶段出现的缺陷和错误，保障安全关键系统的可信性。

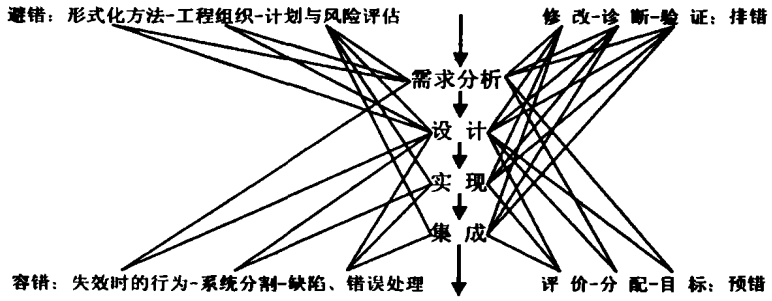

图9　四种高可信保障技术在安全关键系统开发中的应用

5. 基于防危核的错误检测技术

由前可知，容错系统的关键是错误检测。当系统中出现错误状态时，不同的应用需采用不同的错误处理手段，如核电系统出现致命错误时应紧急关堆，而对于正在飞行的飞机当检测到有致命错误发生时，显然不能简单关闭发动机，而应采取其他错误处理手段来保证飞机的安全。因而根据不同应用所要求的错误处理方式的不同，与之相应的错误检测技术也不

应相同。通常错误检测技术可分为并发性错误检测和强制性错误检测两种。并发性错误检测指在服务交付的过程中进行检测，可进一步细分为复制、看门狗定时器、反向、编码及可接受性等检测技术；强制性错误检测则在任务被悬挂之后进行，它主要用于系统潜在错误及静止缺陷的检测，可进一步细分为故障诊断及结构化检查。对于安全关键系统中的错误检测器必须给予高度的重视，一个重要的问题是错误检测器本身不能引入任何缺陷，且错误检测器自身的正确性必须具备较好的可验证性。譬如，错误检测器的引入可能减缓系统对关键数据的处理，当被处理数据其输出结果的值正确但错过其死限(deadline)时，称该错误检测器是不合理的，由此可见，错误检测器的设计是十分关键的。错误检测器的优良可通过三个指标进行评估(同时也是错误检测器的设计准则)：1)错误检测器的设计应只来源于系统规范；2)错误检测器应具备高度的完备性；3)系统与错误检测器应是独立不相关的。实际工程中，很难保证系统与错误检测器应是独立不相关的，唯一可能保证的是系统与检测器不存在相同失效点，具体实施时应保证系统开发人员不能同时开发错误检测器，系统与错误检测器也不能访问相同的信息。另外，由于检测器所引入的性能与造价开销，使检测器对系统作完备检测也是不现实的，唯一可能的是对与防危相关(safety-related)的系统属性作检测。而基于防危核(safety kernel)的错误检测器正是一理想的检测器。

为提高安全关键系统的可信性，J. Rushby 等人提出了防危核的概念，它是一个应用级的核，防危核在安全关键系统中的地位和作用如图10所示。通常，导致安全关键系统发生灾难性事故的系统错误可分为消极错误与积极错误两种，其中消极错误是指系统做了不该或不允许发生的事，如自动飞行控制软件在飞机处于飞行状态时，启动关闭发动机装置；而积极错误则是指系统没有完成规定要做的事，如飞机飞行过程中，检测到迎面有飞行物时，本应绕道飞行，但却继续向前飞行。防危核的主要优势在于检测消极错误的发生，其原因是防止系统做不该做的事比保障系统完成所期望的任务更易于实现和验证。所以在设计高可信的安全关键系统时，为防止系统灾难性事故的发生，直接检测系统发生了不期望的事是切实可行、简单有效的，同时这也是防危核的精髓所在。实际使用时防危核利用一组专门定义的防危检测策略，检测外部条件对系统设备的操作请求，通过防危核隔离应用软件与系统设备，检测其中可能导致生命、财产损失的错误操作请求，并通过应用软件的配合，对检测到的错误作出恰当的处理，进而避免由于错误的设备操作请求而引起的设备错误动作，从而保证系统达到所需的可信性要求。

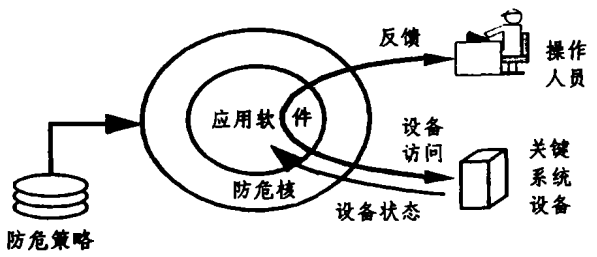

图10　防危核在安全关键系统中的地位与作用

由上可知，为了便于防危核正确性的检验，防危核应尽可能地小，让系统所有的防危检测策略都由防危核来实施是不现实的。因此，防危检测策略的选择是实现防危核的关键之一。在实际应用中，通过故障树分析，将造成设备错误操作的故障源分为若干类，对不同的故障源制定相应的防危检测策略。然后，根据一定的原则选择由防危核强化的防危检测策略，剩余防危检测策略由应用软件配合完成。使用防危核技术，将过去为保证安全关键系统正常运行而对整个安全关键软件进行检测转变为对一个规模较小的防危核进行检测，因而防危核本身的可信性更易得到保证。基于防危核的错误检测器其优点可总结为：① 防危检测策略由防危核实现，无须过多考虑应用软件的实现、更改和验证；② 防危核自身代码量较小，结构简单，易于验证；③ 简化了错误检测器的实现；④ 系统设备由防危核统一检测。最后，为保证基于防危核的错误检测器的正确性，应保证系统开发人员不能同时开发错误检测器，系统与错误检测器也不能访问相同的信息。

结束语　为提高安全关键系统的可信性，本文研究了可信性的保障技术。可信性是一个综合性的概念，为此文中首先阐述了可信性的起源与内涵，紧接着分析了缺陷、错误及失效三者各自的内涵及其之间的关系。为提高系统的可信性，一般可采用避错、容错、排错、预错四种保障技术。鉴于篇幅有限，本文以图表的方式简明扼要地阐述了如何应用以上四种保障技术来创建高可信安全关键系统。然而无论我们怎样设计系统，无缺陷的系统在实际中总是不存在的，因而容错技术总是必需的，但容错的关键是错误检测，为此文中最后提出了基于防危核的错误检测技术。作为今后的研究方向，我们正在研究高可信安全关键实时系统的通用保障机制，其主要目的是改善现有保障技术的专业性强、难移植、性价比低、升级困难等缺陷，具体实施策略将采用 COTS 组件来构建高可信的安全关键实时系统。

参考文献

1 Laprie J C. Dependable computing and fault tolerance: concepts and terminology. In: Proc. 15th IEEE Int. Symposium On Fault-Tolerant Computing (FTCS-15), Ann Arbor, Michigan, June 1985. 2～11

2 Mahmood A, McCluskey E J. Concurrent Error Detection Using Watchdog Processor-A survey. IEEE Transaction on Computers, 1988, 37: 160～174

3 Barrett P A, et al. The Delta-4 eXtra performance Architecture (XPA). In: Proc. 20th Intl. Fault Tolerant Computing Symposium (FTCS-20), Newcastle upon Type, June 1990

4 Powell D. Failure mode assumptions and assumption coverage. presented at Twenty-Second International Symposium on Fault-Tolerant Computing, FTCS-22., 1992

5 Kopetz H, Grunsteidl G. TTP-a Time-Triggered Protocol for fault-tolerant real-time systems. presented at 23rd Annual IEEE International Symp. on Fault-Tolerant Computing, FTCS-23., 1993

6 Lala J H, Harper R E. Architectural principles for safety-critical real-time application. Proceeding of the IEEE, 1994, 28: 25～40

7 Lee P A. Software Faults: The Remaining Problem in Fault Tolerant Systems?. In: M. Banatre and P. A. Lee, eds. Hardware and Software Architecture for Fault Tolerance: Experience and Perspectives, Springer-Verlag, 1994. 171～181

8 Wika K G. Safety Kernel Enforcement of Software Safety Policies: [Ph. D. thesis]. University of Virginia, May, 1995

9 Laprie J C. Dependability-Its Attributes, Impairments and Means: Springer-Verlag, 1995

10 Jones A. The challenge of building survivable information-intensive systems. IEEE Computer, 2000, 33(8): 39～43

11 Avizienis A, Laprie J -C, Randell B. Dependability of computer systems: Fundamental concepts, terminology, and examples, LAAS Report No., UCLA Report No., Oct. 2000

12 Chen Yu. Research on Supporting Techniques for High-Reliable Fault-Tolerant Real Time Systems: [Ph. D. thesis]. UESTC, 2002