1. **物理安全**
   1. 概念：实体安全（Physical Security），是保护计算机设备、设施（网络及通信线路）免遭地震、水灾、火灾、有害气体和其他环境事故（如电磁污染等）破坏的措施和过程。主要是指对计算机及网络系统的环境、场地、设备和通信线路等采取的安全技术措施。
   2. 包括：环境安全、电源系统安全、设备安全和通信线路安全。
   3. 环境安全：应具备消防报警、安全照明、不间断供电、温湿度控制系统和防盗报警。
      1. 机房安全；
      2. 机房防盗：装监控、贴磁条、光缆串接+在线监测是否联通；
      3. 三度：温度、湿度、洁净度；
      4. 防火防水。
   4. 电源系统安全：电源安全主要包括电力能源供应、输电线路安全、保持电源的稳定性等。
      1. 供电安全；
      2. 防静电：不挂毯、采用聚乙烯材料、装防静电地板、保持合适湿度；
      3. 接地与防雷；
   5. 设备安全：要保证硬件设备随时处于良好的工作状态，建立健全使用管理规章制度，建立设备运行日志。同时要注意保护存储媒体的安全性，包括存储媒体自身和数据的安全。
      1. 硬件设备的维护与管理：建立设备使用情况日志、故障情况登记表、对设备的物理访问权限限制在最小范围内；
      2. 电磁辐射防护：电磁辐射有泄密的风险。

防护措施：

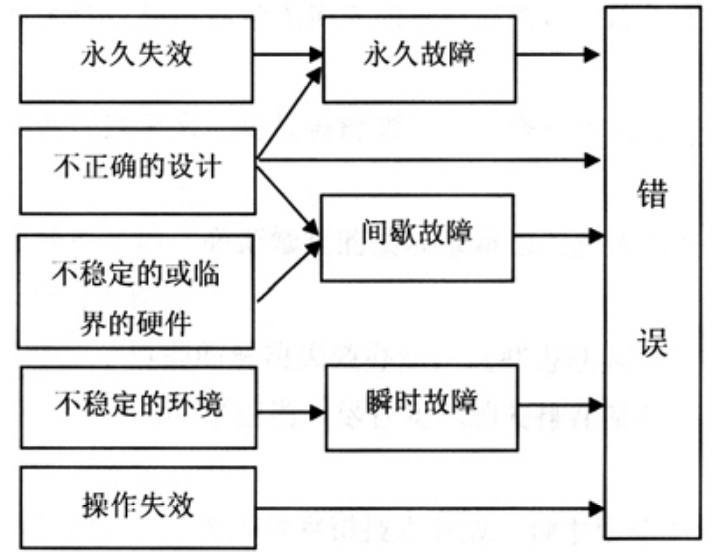
* + - 1. 对传导发射的防护：对电源线和信号线加装**滤波器**；
      2. 对辐射的防护：
         1. 采用各种电磁屏蔽措施；
         2. 干扰的防护措施：利用干扰装置产生一种与计算机系统辐射相关的**伪噪声**向空间辐射来掩盖计算机系统的工作频率和信息特征。
    1. 信息存储媒体安全；
  1. 通信线路安全：包括防止电磁信息的泄漏、线路截获，以及抗电磁干扰。

通信电缆密封在塑料套管中，并在线缆的两端充气加压。线上连接了带有报警器的监示器，用来测量压力。如果压力下降，则意味电缆可能被破坏了，技术人员还可以进一步检测出破坏点的位置，以便及时进行修复。

1. **容错和可靠性**
   1. 失效：硬件**物理特性**异变，或软件不能完成规定功能的能力。
   2. 故障：硬件或软件的错误状态，是失效在逻辑上的等效。一个故障可以用**种类、值、影响范围**和**发生时间**来描述。
      1. 按逻辑性分：

* 逻辑故障：造成逻辑值发生变化的故障
* 非逻辑故障：造成象时钟（clock）或电源出错等错误的故障
  + 1. 按时间分：
* 永久性故障：调用**诊断程序**进行故障定位，然后采取纠错措施；
* 间隔性故障：可以通过**更换硬件或软件**等途径来达到修复的目的；
* 偶然性故障：只能靠**改善环境条件**等努力来减少这类故障。
  1. 错误：程序或数据结构中的故障表现形式，是故障和失效所造成的后果。

错误的根源：



* 1. 容错设计的软件可以**有某些规定数目的故障**但**不导致失效**，但对无容错的软件而言，故障即失效。
  2. 可靠性：【广义】一切旨在避免、减少、处理、度量软件/硬件故障（错误、缺陷、失效）的分析、设计、测试等方法、技术和实践活动。【狭义】软件/硬件无效运行的定量度量。
* 可靠度：在规定的**运行环境**中和**规定的时间**内软件**无失效运行的机会**。
  1. 系统可靠性的指标：
     1. 可靠性（Reliability）：计算机在**规定条件**和**规定时间**内完成**规定功能**的概率。

失效：产品在规定条件下和规定时间内丧失了规定功能。

失效率：计算机在**某一瞬间**元件**失效数**与**元件总数**的比率。

**规定条件**：环境条件，使用条件，维修条件， 操作技术等。

* + 1. 可维性（Serviceability）：在**规定时间**内，按照**规定的程序和方法**进行维修时，保持或恢复到**能完成规定的功能**的能力。

即从**判定故障**到**排除故障**所需要的时间，包括**故障诊断**、**故障定位**、**系统校正**和**恢复**等时间。

可保持性（Maintainability）：系统在**给定的时间内**可**隔离故障**或**修复**的概率。表征了系统可以**正常运行的效率**。

* + 1. 可用性（Availability）：【**有效率**或**利用率**】可维修部件在**某时间**具有**维持规定功能**的能力，即计算机系统的**利用效率**，也是系统在执行任务的任意时刻能**正常工作的概率**。
  1. 排错：避免故障，通过对组成系统的**部件**进行严格的**筛选**、对系统进行严格的**测试**、对系统进行**屏蔽**以**减少外界的干扰**等方法来提高系统的可靠性。
  2. 容错：即容忍故障，一旦发生故障能自动检测出故障并使系统自动恢复正常运行，使计算机系统在发生故障后仍能正确运行。当出现某些指定的硬件**故障**或软件**错误**时，系统仍能执行规定的一组程序，或者说程序**不会因系统中的故障而中止或被修改**，并且执行**结果也不包含系统中故障所引起的差错**。

容错技术：设计与分析容错计算机系统的各种技术。

* 从**系统结构**方面来提高计算机系统的可靠性
* 容错技术与排错技术并**不是相互对立**的，它们可以**相互补充**，构成高可信的计算机系统
  1. 实现容错计算的四个方面：
     1. 不希望事件的检测。

**不希望事件**：失效、 故障、差错等。

* + 1. 损坏估价。在作出一个被检测的故障有关的决定之前，有必要判定系统已被破坏的程度。

原因：故障的出现和它的失效结果之间可能存在延迟，故障可能已经传播到该系统的其他地方，导致故障的扩大。

影响因素：设计策略、探测技术。

* + 1. 不希望事件的恢复。把目前的错误系统状态转换成一个正确的系统状态。
    2. 不希望事件处理和继续服务。确保已被恢复的不希望事件效应不会立即再现，以使系统继续提供规定的服务。
  1. 容错系统的一般阶段：
     1. 故障限制：把故障效应的传播限制到一个区域内，防止污染其他区域。
     2. 故障检测：大多数失效最终导致产生逻辑故障，可用奇偶校验、一致性校验检测出来。

脱机检测设备不能进行有用工作；联机检测提供实时检测能力；联机检测技术包括奇偶校验和二模冗余校验。

* + 1. 故障屏蔽：把失效效应掩盖起来，可以说是冗余信息战胜了错误信息，如**多数表决冗余设计**。
    2. 重试：在许多场合（尤其是瞬间故障），对一个操作的第二次试验可能是成功的。
    3. 诊断：如果故障检测技术没有提供有关**故障位置**和/或**性质**的信息，那么就需要一个诊断。
    4. 重组：当发生永久故障时，**重组系统器件**以便替代失效的器件或把失效的器件与系统其他部分**隔离**开来，也可使用**冗余系统**，系统能力不降低。
    5. 恢复：系统回到故障检测前处理过程的某一点，并从该点重新开始操作（称作**卷回**）。通常需要后备文件、校验点和应用记录方法。
    6. 重启动：破坏过多或无恢复功能时，使用重启动。

热重启：仅系统未受任何破坏，可从故障检测点恢复所有操作。

温重启：仅有某些过程可以毫无损失的重新启动。

冷重启：系统需要完全重新加载。

* + 1. 修复：把诊断为故障的器件换下来，可联机也可脱机。
    2. 重构：对元件进行物理替换之后，把修复的模块重新加入到该系统中去。对联机修复来说，实现重构不中断系统的工作。
  1. 容错技术的发展概况
     1. 第一代：
* 元件：电子管、继电器及延迟线存储器
* 问题：**元件的失效率**相当高，并易受**瞬时故障**的影响
* 措施：特别设置的**硬件故障检测**和**人工恢复**
  + 1. 第二代：
* 元件：晶体管及磁芯存储器
* 问题：失效率比第一代计算机元件大为降低
* 措施：**避错技术**占统治地位，对故障一般采用**诊断程序**进行**脱机检测**
  + 1. 第三代
* 元件：集成电路
* 问题：元件的失效率继续降低，但计算机应用范围扩大，对计算机系统的**可信性**要求更高
* 措施：容错技术重新提出，并得到了较快的发展，并出现了许多**容错计算机**
  + 1. 第四代
* 元件：大规模和超大规模集成电路
* 问题：硬件可靠性大大提高而价格却大幅度降低，使采用各种容错技术在经济上**更易接受**
* 需求：容错技术应用范围扩展于**银行事务处理**及各种**实时控制系统**，甚至许多**通用计算机**系统也采用了容错技术
  + 1. 80年代以来

出现商用容错计算机市场、分布式容错计算机系统、容错的VLSI技术、人工智能在容错技术上的应用——计算机故障诊断专家系统

* 1. 容错技术主要内容
     1. 故障检测与诊断技术

是容错技术的主要组成部分，包括故障检测（Fault Detection）【判断是否存在故障】、故障定位（Fault Location）【判断系统在哪里发生故障的过程】、故障测试（Fault Testing）【又称故障诊断（Fault Diagnosis），故障检测和故障定位，主要包括：测试集生成技术，功能测试技术，系统诊断技术】。

检测的作用：确认系统是否发生了故障，指示故障的状态，即查找**故障源**和**故障性质**。只能找到错误点（错误单元），不能准确找到故障点。

诊断的作用：判断是否故障并给出故障定位。

技术基础：检错纠错码技术。奇偶校验码、循环码、海明码等。

* + 1. 故障屏蔽技术

作用：防止系统中的故障在该系统的信息结构中产生差错。

措施/实质：在故障效应达到模块的输出以前，利用冗余资源将故障影响 掩盖起来，达到容错目的。

特点：不改变系统的结构，即系统部件之间的逻辑关系相互固定，又称**静态冗余技术**。

* + 1. 动态冗余技术

分级：元件级故障冗余技术、逻辑级故障冗余技术【高可靠性的容错专用集成电路芯片和不宜放在功能模块一级上进行容错设计的关键硬件】、模块级故障冗余技术、系统级故障冗余技术【由两个以上的相同系统合成一个系统，冗余**系统间通过相互校验**来保持正常运行】

硬件冗余：在常规设计的硬件之外附加**备份硬件**，包括**静态冗余**、**动态冗余**。

时间冗余：重复地执行指令或一段程序而附加额外的时间。

信息冗余：增加信息的多余度，使其具有检错和纠错能力。

软件冗余：用于测试、检错的外加程序。

* + 1. 软件容错技术

在出现有限数目的软件故障的情况下，系统仍可提供连续正确执行的内在能力。主要包括**N版本程序设计**和**恢复块技术**。

目的：屏蔽软件故障，恢复因出故障而影响的运行进程。

* + 1. 信息保护技术

为了防止信息被不正当地存取或破坏而采取的措施。分为编码化与密码化、资格检查、内存保护、外存保护。

1. **故障自检测与自诊断技术**
   1. 联机检测和脱机检测
      1. 联机检测与诊断：在系统**运行期间**实现自我检测与诊断的过程，也叫自检测与自诊断，是提高系统**可靠性**的重要环节。

方法：投入一定的**冗余资源**。使系统在运行中不仅能输出功能所要求的信息，而且能**输出一些额外的信息**。

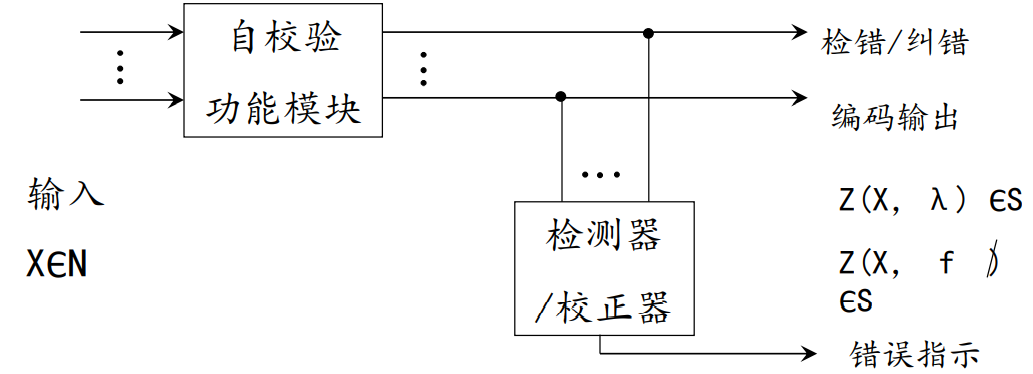
实质：可靠性编码技术在系统设计中的应用，是一种以**硬件冗余**为主的检测与诊断技术。

* + 1. 脱机检测与诊断：在系统**非运行期间**进行检测与诊断，是提高系统**可用度、可维性**的重要环节。

方法：**软件诊断程序**。先优化生成并输入一组测试向量给系统，然后观测并分析系统的测试响应，以确定系统是否发生故障或哪个部件发生的故障。

* 1. 逻辑网络的故障安全与自校验特性
     1. 故障安全特性：
* **故障安全（Fault-Secure，FSE）**：一个数字逻辑网络，对给定故障集F，如果在任何**有效输入码**激励下，都不会因为故障而输出错误的有效输出码，则称该网络对故障集F是故障安全的。
* **强故障安全**：若该网络在**任意输入码**激励下，对给定故障集F都不会输出错误的有效输出码。
* 输入正确则输出**正确有效码**或**无效码**，不会输出错误的有效码，这样系统就不会因为故障存在而造成系统出现错误的有效输出，则该系统为故障安全的。
* 若输出无效码，则系统必有故障；但若输出有效码，系统有无故障无法得知【因为有故障也可能输出正确有效码】，只能知道输出是没错的【注意区分输入正误和系统正误】，系统可能存在不可测故障。
  + 1. 自校验特性：
* **可自校验（Self-Testing，ST）**：一个数字逻辑网络对于给定故障集F内的每一个故障，至少存在一个**有效的输入码**能使它的输出产生一个**无效输出码字**。【就是保证在系统出现故障的时候能在输出端体现出来】
* 可自校验的系统，必可从观测输出码字是否为有效码字来判别其是否有故障，而且一定有一个校验码就包含在有效的输出码字集合中。
  1. 自校验网络的基本结构

**自校验网络**：在正常工作的过程中，由一个称为**校验器**的电路通过监视**自校验功能模块的输出状态**而自动指示有无错误。无需施加特定的测试输入码。



* 检测器：当出现无效码时，指示错误状态，可构成自检测网络。
* 校正器：当出现无效码时，指示出错位置，并**予以纠正**，可构成自诊断网络或故障屏蔽网络。

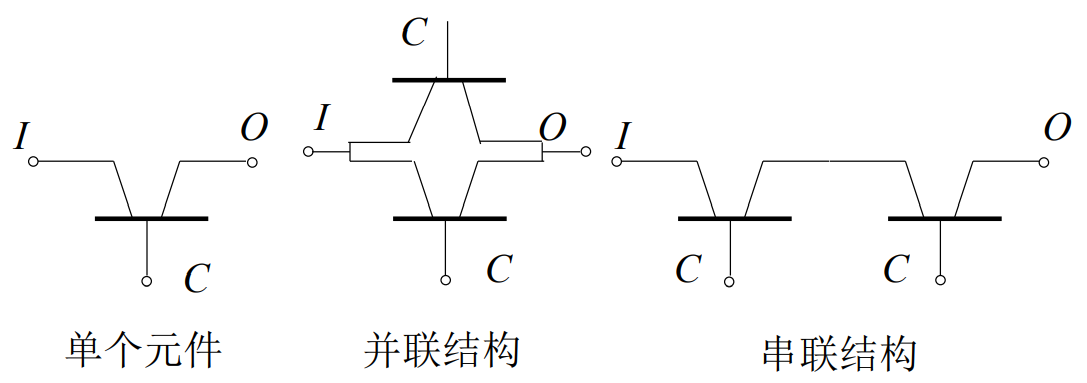
1. **故障屏蔽技术（静态冗余）**

定义：

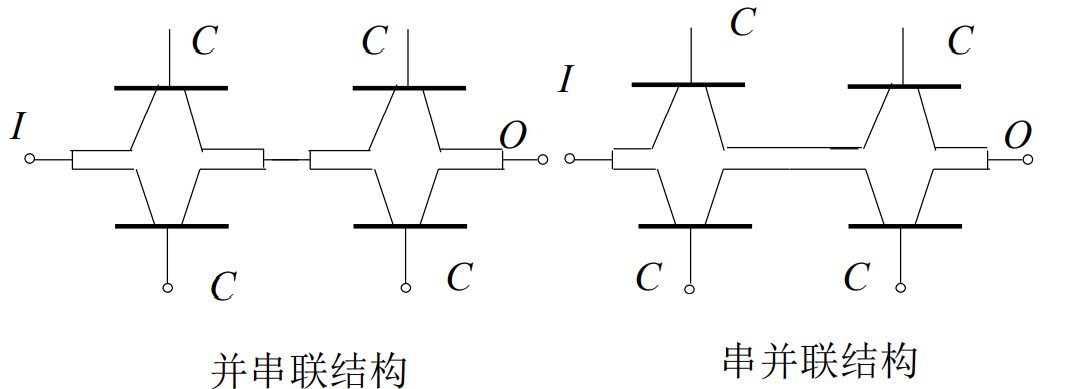
**故障屏蔽技术（Fault Masking Technology）**：防止系统中的**故障**在该系统的信息结构中**产生差错**的各种措施的总称。

实质：在故障效应达到模块的输出以前，利用冗余资源将故障影响掩盖起来，达到容错目的。

* 1. 元件级故障屏蔽技术
     1. 二倍冗余



* + 1. 四倍冗余

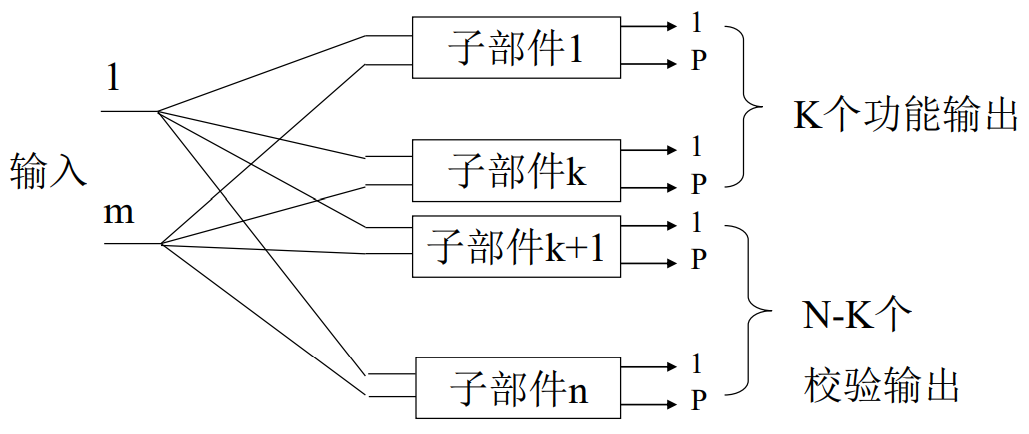


优点：

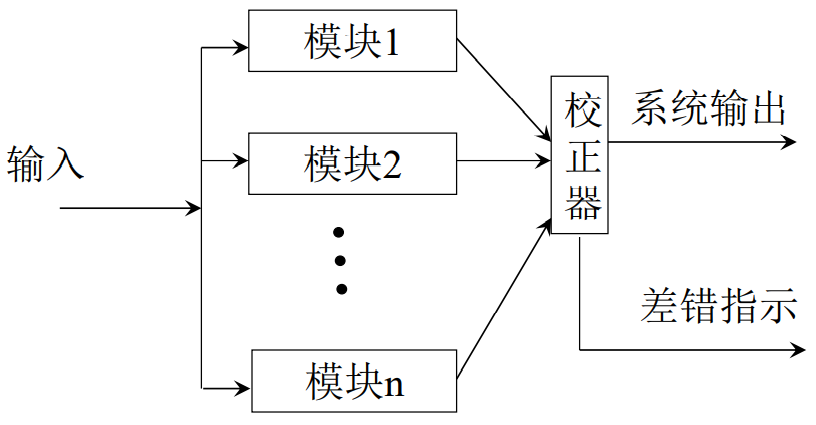
* + - 1. 可用于**元件级**
      2. 构成**独立自足冗余**，即不必增添逻辑线路来检测故障与恢复信息
      3. 能满足**实时**与**连续可用**条件

缺点：

* + - 1. **功耗大**
      2. 对元件适应条件**要求较高**
      3. **昂贵**
      4. 结构不能变，**无灵活**性
    1. 桥接冗余
  1. 逻辑级故障屏蔽技术
* 主要用于高可靠性、的容错专用集成电路芯片和不宜放在功能模块一级上进行容错设计的关键硬件。主要用于超大规模集成电路中。
* 多采用**四倍冗余**方案。



* 1. 模块级故障屏蔽技术
* 实现：以**N倍冗余码**为基础，构成N模冗余（NMR，N-Modular Redundancy）系统。



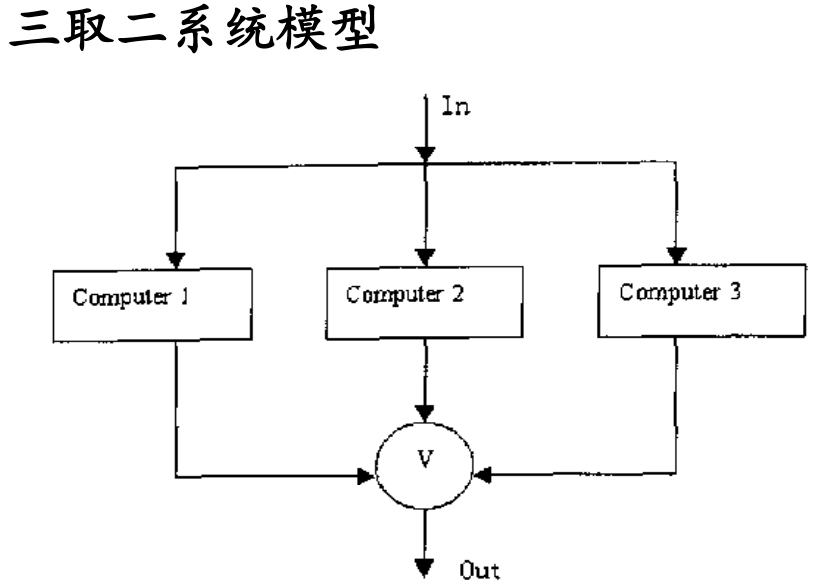
* 1. 系统级故障屏蔽技术
     1. 采用多个计算机系统实现系统的功能。也称**多处理机**或**多计算机系统**。
     2. 常用结构：
* 均分负载系统结构
* 主备用系统结构
* 双机协同系统结构
* 紧耦合多处理机系统结构
* 分布式系统结构
* 网络结构

1. **冗余容错技术**

定义：

在系统结构上通过**增加冗余资源（硬件、软件或设备）**的方法来**掩盖故障造成的影响**，使得即使出错或发生故障，系统的功能仍不受影响，仍能够正常执行预定任务的技术。

* 1. 硬件冗余
     1. 静态冗余
* 定义：利用**逻辑重叠**技术有效地掩蔽硬件故障，又称**掩蔽冗余**。典型代表是**N取r系统** （三取二的冗余）。
* 原理：采用**并联结构**，从输入端输入相同的信息，并执行同样的程序，完成同样的任务，得到的结果送入**表决器**，以多数的结果作为最后的输出。



* 优点：系统可**不进行故障检测、定位和系统的恢复工作**，省掉了大量的软件工作，简单易行。
* 缺点：增加了硬件设备，提高了系统的**费用**。
  + 1. 动态冗余

采用辅助系统作为主系统的**热备份**，正常状态下主系统工作，并对主系统进行**故障检测和定位**，一旦诊断出故障发生的位置，系统能进行**自动修复**。

* + 1. 主要方法
       1. 多级硬件冗余方法：通过**多种设备**的冗余，提高系统的可靠性。
       2. 并联冗余，备份冗余和表决系统。
       3. 冷备份，热备份。【热备份：处于工作状态时备份；冷备份：退出工作状态后备份】
* 优点：不可间断系统采用了计算机局域网络或分布式计算机系统，各子系统**无需或少许增加硬件设备**，就可以起到互为冗余的作用。即不 增加系统的体积、重量、功能和成功，设备充分共享，提高了利用率，弱化了故障。
* 缺点：虽然不增加硬件设备却**增加了大量软件工作**，系统研制的难度、周期和费用增加了；用户的应用程序在系统中运行，经常被诊断程序中断，遇到故障时**自动定位和恢复过程较慢**；不适合**实时性很强**的系统。
  1. 软件冗余

定义：针对软件本身的故障， 采用方法防止由于**软件错误**而导致系统失效。

* + 1. 动态冗余
       1. 程序回卷（Program Rollback）：

当在程序中设置的某个测试点上**检查到错误的输出结果**时，就认为正在执行的程序在一个错误的系统中运行，这段程序要被**重新执行**，即**程序的卷回**。

* 优点：不增加硬件设备，软件也不复杂。
* 缺点：系统的**反应时间和精度**要受损失，对偶发性错误有效，对**固定的故障无能为力**。
  + - 1. 多道程序表决法：

定义：对同一个问题按**不同的算法编多个程序**，对执行的**结果进行表决**。

* 串行执行：**费时**，过多地**占用主存空间**，不符合实时性和实用性的要求。
* 并行执行：需要硬件支持。
  + - 1. 模块恢复法：

定义：程序按照**模块执行**，每执行完一块，接着执行一个测试程序，对刚执行的程序**进行测试**，通过后再执行下一个模块，通不过则**重新执行**。

缺陷：无法适应实时系统的要求【实时性差】。

* + 1. 静态冗余

定义：又称**软件存储冗余**，把关键的系统程序和应用程序在系统中**多点存储**在主存或高速辅助存储器中。正常状态下只执行一处或几处的程序，其它做备份，一旦程序遭破坏，则快速调入备用程序，使系统仍能正常工作。系统程序和应用程序可采取**分布式存储**，或**集中式存储**，或两者**结合**。

|  |  |
| --- | --- |
| **软件** | **硬件** |
| 软件拷贝不可作为软件备份 | 硬件拷贝可作为硬件备份 |
| 暂时性替换，被替换掉的部分仍可重新使用 | 硬件部件需重新修复才可使用 |
| 技术复杂 | 技术简单 |

* 1. 数据冗余

应用：数据库系统

定义：若某种故障使数据库中当前状态**不正确**或**可疑**，就必须把数据库**恢复到某一正确的状态**。

分类：根据数据库**存储的位置和方式**，分为**集中式**的数据冗余和**分布式**的数据冗余。

1. **软件可靠性技术**
   1. 概述
      1. 软件与硬件可靠性的联系与区别
         1. 联系

* 都指系统（或产品、模块）在一定的条件下和一定时间内能完成预定功能的性质
* 都是**复杂性**的函数
* 都可利用**可靠性增长**来提高它们的可靠性
  + - 1. 区别

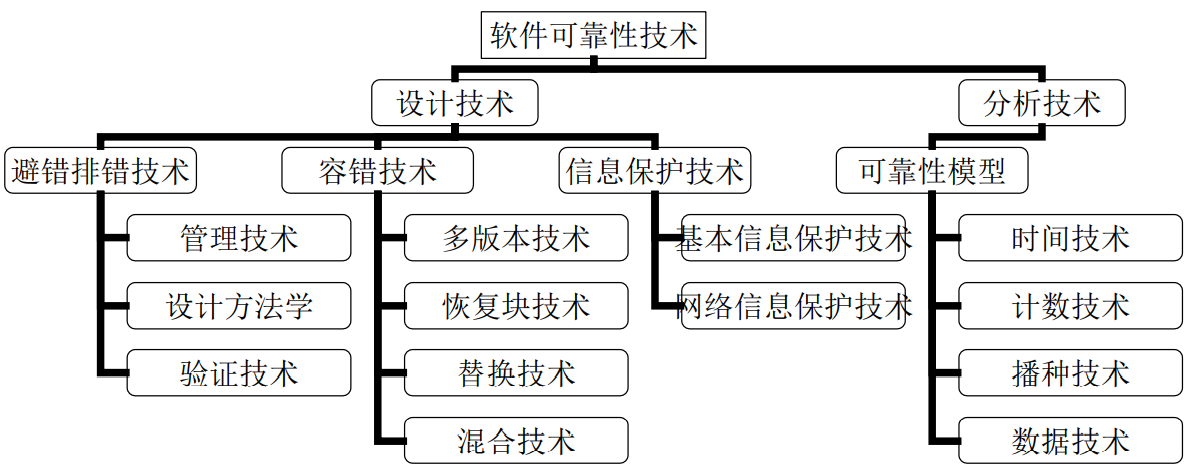
概念内涵、指标选择、设计分析手段、提高可靠性的方法途径。

* + - 1. 软硬件故障机理

硬件：元部件的失效，“**后生**”的故障【用着用着用坏了】。

软件：人的失误和水平、能力的局限性，“**先天**”的故障【设计之初就存在隐患】。

* + - 1. 软件故障特性
* 固有性
* 环境敏感性：**运行环境**【硬件平台、硬件配置、支撑软件】、输**入环境**【应用对象、用户要求、输入数据等】
* 故障影响的传染性：未排除的软件故障将会始终存在在该软件中，一旦引起错误，是可以传染给其他软件的。
  + 1. 内涵
* 可靠性设计：为了获得高可靠性的软件，包括避错排错设计、容错设计、信息保护。
* 可靠性分析：通过建立一定的**可靠性模型**，为软件可靠性**设计**和软件**维护**提供必要的**依据**。



* + 1. 定义
* 计算机软件：与计算机系统的操作有关的程序、规程、规则以及与之有关的文件和数据。
* 软件质量：软件产品满足规定需求或隐含能力所有的特征和特征之和。

指标：功能性【实用性、准确性、互操作性、一致性和安全性】、可靠性【成熟性、容错性、可恢复性】、易使用性【易学性、易理解性、易操作性】、效率【时间性、资源性】、维护性【可分析性、易修改性、稳定性和易测试性】、可移植性【适应性、可安装性、规范性和可换性】。

* 软件可靠性：在规定的条件下和规定的时间内，软件**成功完成**功能的能力或**不引起系统故障**的能力。

特点：与软件开发方法、验证方法、使用的程序设计语言、软件的运行环境条件、操作人员的素质均有关。

* + 1. 指标

需遵循的规则：与**系统可靠性**表示方法相协调、用户概念、以使用过程中**易观测的参数**来表示、针对具体的任务，对不同的功能应用不同的指标和要求【具体问题具体分析】。

* + - 1. 沿袭**硬件可靠性**的软件可靠性指标
* 软件可靠度(Rs(t))：软件系统在特定的环境下，在规定的时间内**不发生故障**地运行的概率。
* 故障率(λs(t))：软件工作到某时刻t尚未失效，在时间**t后单位时间内**发生故障的概率。
* 平均故障间隔时间(MTBF)：**交付用户使用**的操作期间软件各次故障的间隔时间的期望值。
* 平均故障前时间(MTTF)：软件经**测试过程**中，软件各次故障之间的间隔时间的期望值。
* 平均修复时间(MTTR)：软件系统在特定的环境下，在规定的时间内，在规定的维修级别上，**维修时间的平均值**。
  + - 1. 表征**软件特殊性**的软件可靠性指标
* 平均不工作时间：软件系统平均不工作时的时间。
* 平均操作错误时间：软件操作错误的平均间隔时间。
* 软件系统不工作时间：因软件故障，系统不工作时间的平均值。
* 可用性：软件在规定的开始时刻t0运行正常的条件下，在规定的未来时间t正常运行的概率。
* 初始错误个数和剩余错误个数。
* 使用方错误率：使用方在使用软件的总次数中，误用次数所占的百分率。
  1. 软件容错技术
     1. 容错软件的基本概述及原理
        1. 容错软件的定义

在一定程度上上对自身故障的作用具有**屏蔽能力** 或 在一定程度上能从故障状态**自动恢复**到正常状态 或 在因缺陷而**出故障**时仍然能在一定程度上完成预期的功能 或 在一定程序上具有**容错能力**的规定功能的软件。

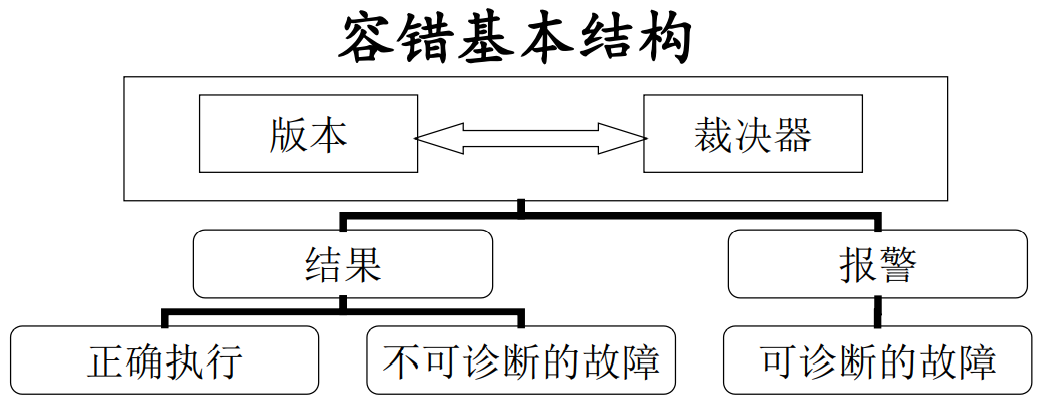
* 特点：

1. 容错对象：规定功能的软件，这些功能是由需求规范定义的。
2. 容错的能力总是**有限度**的。
3. 当软件由于自身存在缺陷而在运行中**出故障**时，若其为容错软件，应能**屏蔽这一故障**，对其进行处理以**避免失效**（通过故障检测算法、故障恢复算法、软件冗余备份来实现）。

* 排错并不能解决所有问题：

1. 缺陷与因其而引起的故障间的关系十分复杂，依据对故障的观测来确定缺陷的难度是很大的。
2. 设计中的**残存缺陷**大多已经和软件总体以及各阶段中形成的文档、资料等建立了**密切联系**，**难以一举排除**。
3. 在研制时间上，不允许作旷日持久的检验和修改。
   * + 1. 实现软件容错的基本原理

* 原理：将若干个根据**同一规范编写**的**不同程序** （或程序块），在不同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上或最终通过表决或接收测试进行裁决。在**判明其正确或一致**后接收这个结果，否则便加以拒绝，并作出报警。



* 结果：

1. 裁决器**判断正确**，软件**正确无误**地实现了需求规范所载的功能；
2. 裁决器**判断正确**，发现**故障**，发出报警，能制止系统失效导致严重的后果；
3. 裁决器**判断错误**或软件存在**不可诊断的故障**， 使系统最终失效。

* 容错软件的结构方案：

1. 多（N）版本程序设计（N-Version Programming，NVP）结构
2. 恢复块（Recovery Block，RB）技术
   * + 1. 实现容错软件的有关技术

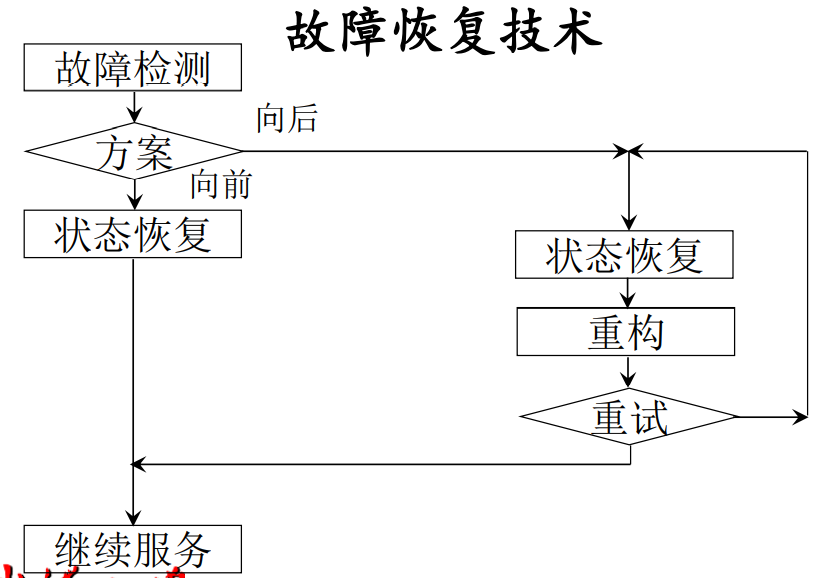
* 版本冗余
* 故障检测技术
* 软件断言（Software Assertions）：软件在宿主系统中运行时，能对其进程或功能的正确与否作出判断的条件。断言提供三个结论：正确、不正确、不能判别。
* 软件自测试

正面校验（Positive Check）原则：测试软件将**输入转化为输出**的功能是否正确。

反面校验（Negative Check）原则：将软件**输出逆转化为输入**，检查是否正确。

* 故障恢复技术

向前恢复、向后恢复：

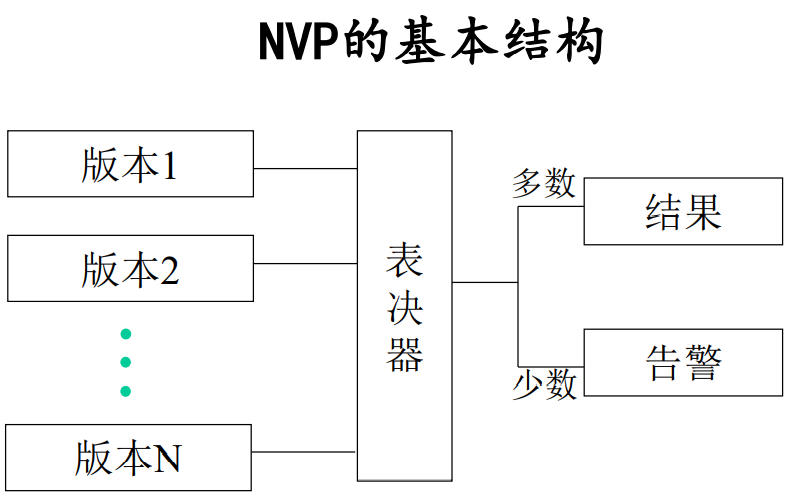


* 破坏估计
* 判定故障被**检测出前**已经引起的破坏
* 发生故障后，在处理的**延滞**或**恢复**实施过程中，**无效信息**在系统中传播的可能性
* 故障可能导致的**其他**未被检测到的后续故障
* 故障隔离技术
* 定义：主动地采取措施，防止故障的破坏性**蔓延**的技术。
* 权限最小化原则：对过程的数据加以严格的**定义和限制**，令过程不能提供任何超过事先**规定限度**的功能，也无权接受来自限定**数据库之外的数据**。
* 继续服务
* 确保**向前恢复**后的**输出序列中所失去的部分**不致于影响软件的基本功能。
* 确保**向后恢复**后输出序列中**重复多余**的部分和**差错状态**不致影响输出的正确执行。
  + 1. 容错软件设计的基本技术
       1. N版本程序设计（NVP）

定义：NVP的思想来自硬件NMR（N-Modular Redundant）结构，是一种**静态冗余**技术，由N个实现**相同功能**的不同程序同时（或几乎同时）在松耦合计算机上运行，然后比较运行结果，在出现不一致的情况下，利用**多数表决**决定一个最优先的结果。

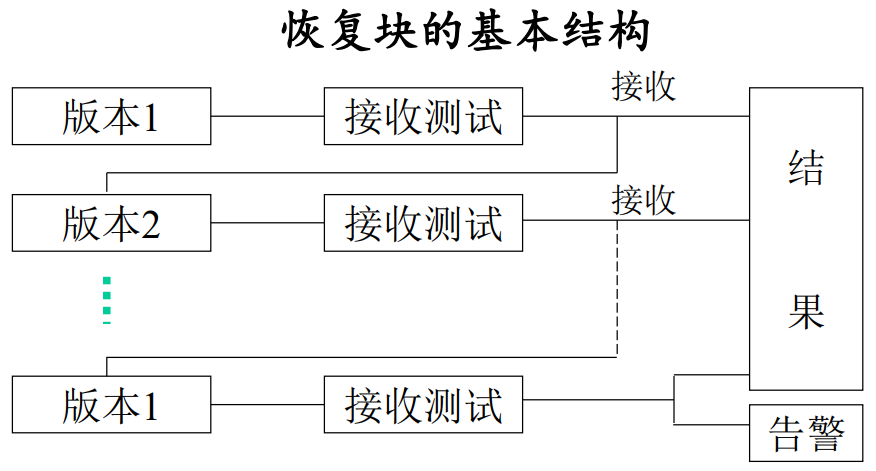
优点：**结构简单**，不设计检测程序来确定运行结果的正确与否。

缺点：① 处理时间是单版本的N倍；② 设计所花的力量是单版本的N倍；③ 表决程序设计较困难；④ 当该程序需要与**外部事件**同步时，或在并行处理中在**交互**过程时，难于处理。



* + - 1. 恢复块（Recovery Block）技术

其思想来源于硬件的待机（Standby）冗余的动态结构。



* 接收测试设计
* 逆向检查：接收测试利用模块中的结果，计算本应采用的输入值，并与真实输入值比较以决定结果是否可接受。
* 编码校验：利用**纠、检错码**的原理建立接收测试。
* 合理性范围测试：根据**可能的变化范围**，以便检查变量是否越出范围或状态变化违反允许的序列。
* 结构校验：比如利用**数据结构**链表的链接特性来进行校验。
* 账目校验：在事务处理中，帐目校验是用**帐目平衡**来作接收测试最方便的基准。
* 时间运行校验。
* 替换块的设计
* 相同加权，独立设计
* 优先的、全功能设计
* 功能降级设计
  + 1. 容错软件设计的先进技术
       1. 一致性恢复块（CRB）

定义：按顺序结合N版本程序设计（NVP）和恢复块技术（RB）的混合系统。如果NVP失效，系统将恢复到RB。只有当NVP和RB都失效时系统才发生故障。

* + - 1. 接收表决

定义：与CRB混合方向反向，多个模块并行执行，模块输出先经过**接收测试**，接收测试接受输出后，结果再经**表决器表决**。

表决器是**动态**的，因为接收测试每次**输出结果的数目各不相同**。只有**两个或多个输出正确**时，表决器才产生一个决定。

* + - 1. N自检程序设计

定义：N个模块**成对执行**（N取偶数），采用**比较模块输出**或其他方法来判断系统正确与否。

比较模块输出：如果每一对的输出之间**不相同**，则**放弃输出**。

* 1. 容错软件的相异性设计准则
* 各冗余软件由**相互独立**的不同人员进行开发。
* 各冗余软件以**不同的形式**说明。
* 各冗余软件的**设计评审**的**参与人员不重复**。
* 各冗余软件的最终规范及最终设计、最终编程由**不重复的审核人员**对照软件需求、软件规范、软件设计进行审核。
* 如果开发采用了CAD工具，则各冗余软件应采用**不同的CAD工具**。
* 各冗余软件析测试程序的规范、测试方式、测试程序，尽可能由不重复的**互相独立的人员组开发**。
* 各冗余软件应在**算法**上相异。
* 各冗余软件应在**逻辑结构**上相异。
* 如有可能，各冗余软件最好由不同开发单位设计，**开发单位之间是相互独立**的。
* 各冗余软件用不同的**程序设计语言**设计。
* 各冗余软件用不同的**编译程序或汇编程序**翻译成目标码。

1. **信息安全工程**
   1. 容错信息安全工程概述
      1. 信息安全具有全面性。
      2. 信息安全具有过程性或生命周期性。
      3. 信息安全具有动态性。【不断发展完善】
      4. 信息安全具有层次性。
      5. 安全具有相对性。【没有绝对的安全】
   2. 信息安全工程的设计准则
      1. 信息安全的木桶原则
      2. 网络信息安全的整体性原则
      3. 安全性评价与平衡原则
      4. 标准化与一致性原则
      5. 技术与管理相结合原则
      6. 统筹规划，分步实施原则
      7. 等级性原则
      8. 动态发展原则
      9. 易操作性原则
   3. 信息安全工程的设计步骤

* **安全风险分析与评估**
  + 1. 风险分析与评估

通过一系列的管理和技术手段来检测当前运行的信息系统所处的安全级别、安全问题、安全漏洞，以及当前安全策略和实际安全级别的差别，评估运行系统的风险，根据审计报告，可制定适合具体情况的安全策略及其管理和实施规范，为安全体系的设计提供参考。

* + 1. 风险分析的内容
       1. 网络基本情况分析。
       2. 信息系统基本安全状况调查。
       3. 信息系统安全组织、政策情况分析。
       4. 网络安全技术措施使用情况分析。
       5. 防火墙布控及外联业务安全状况分析。
       6. 动态安全管理状况分析。
       7. 链路、数据及应用加密情况分析。
       8. 网络系统访问控制状况分析。
       9. 模拟攻击测试。
    2. 风险分析的步骤
       1. 确定要保护的资产及价值。
       2. 分析信息资产之间的相互依赖性。
       3. 确定存在的风险和威胁。
       4. 分析可能的入侵者。
* **安全策略和需求分析**
  + 1. 制定安全策略

定义：为发布、管理和保护敏感的信息资源而制定的一组法律、法规和措施的总合，是对信息资源使用、管理规则的正式描述，是企业内所有成员都必须遵守的规则。

内容：保护的内容和目标、实施保护的方法、明确的责任、事故的处理。

* + 1. 进行需求分析：五个层面：管理层、物理层、系统层、网络层、应用层
* **设计企业信息系统的安全体系**

一个完整的安全体系应该包含以下几个基本的部分

* + 1. 风险管理
    2. 行为管理
    3. 信息管理
    4. 安全边界【信息系统与外部环境的连接处】
    5. 系统安全【为操作系统添加安全外壳】
    6. 身份认证与授权
    7. 应用安全
    8. 数据库安全
    9. 链路安全
    10. 桌面系统安全
    11. 病毒防治
    12. 灾难恢复与备份【尽快恢复、周期性备份】
    13. 集中安全管理
* **安全工程的实施与监理**

关键：保证安全工程 的质量，避免重复建设和垃圾工程。

重视：选择一个科学、合适的实施方案作为工程实施的指导；选择一个工程能力可靠的施工单位负责工程的建设；对工程实施的整个过程进行监理。

流程：工程实施前、中、后的监理。