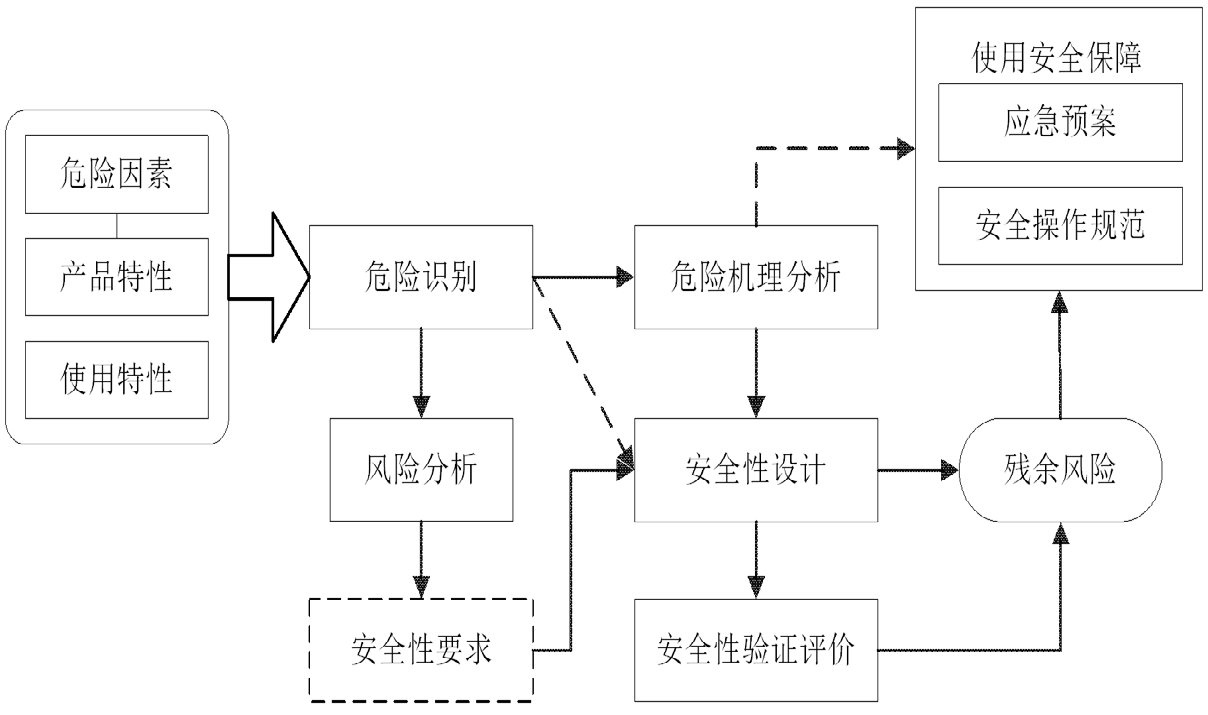
软件安全性复习

# 1.安全性核心原理



## 基本概念

### 系统

系统是由相互制约的各部分组成的**具有一定功能的整体**。（钱学森）

系统的基本特征：多元性、关联性、整体性。

### 安全性

产品具有的不导致人员伤亡、装备毁坏、财产损失 或不危及人员健康和环境的能力。 （GJB900A-2012《装备 安全性工作通用要求》）

### 安全原理

**以事故为对象、以危险为核心**，阐述安全的本质与属性，事故为何发生和如何发生的基本规律，以及 如何预防、控制事故和救援的一般原理与方法论。

### 基本方法

 安全性要求确定

 危险分析

 安全性设计

 安全性验证

 安全性评价

 危险跟踪

 变更安全性保证

### 软件开发与管理的三维模型

现代软件工程化管理的理念为软件开发与管理构建一个全方位、全过程、多层次的三维框架，这三维是：

 **时间维**：对软件生存期的全过程进行控制；全过程控制

 **空间维**：对软件质量有关的关键因素实施全方位管理；

 **组织维**：构建从软件开发个人、软件开发小组到软件开发单位的多层次的管理模式。全组织管理

### 事故

造成人员伤亡、职业病、设备损坏或财产损失的一个或一系 列意外事件。--GJB/Z 142-2004

### 危险

🞐可能导致事故的状态。--GJB 900A

🞐可能导致或有助于事故或灾难（人员伤亡、或系统毁坏、或 财产损失或环境破坏等）发生的实际条件或潜在条件（1维）

🞐危险可能是硬件或软件的缺陷或人的错误引起，或一个非预 期输入。

### 风险

用事故的可能性和事故严重性表示发生事故的可能程度（2维）

### 软件安全性相关概念

#### 软件可靠性

在规定的条件下，在规定的时间内， 软件运行不发生失效的概率。

#### 失效

是指程序的运行偏离了软件需求规格说明文档中的要求，是软件动态运行的输出结果。

#### 软件安全性

软件在系统中运行而不导致系统发生事故的能力。

#### 安全相关软件

1） **软件驻留在某个（由危险分析确定的）安全性关键系统之中，且至少满足下述条件之一**：

 ·若该软件失效能导致或促成某个危险发生；

 ·用于控制或缓解危险；

 ·用于控制安全性关键功能；

 ·用于处理安全性关键命令或数据(注：如果数据是用于人或者系统进行安全性决策，则该数据被确定为安全性关键的；获取、处理和传输该数据的软件也被确定为安全性关键的；但对于可提供安全性关键信息，但这些信息不被用来进行安全性或危险控制的数据或软件，则不是安全性关键的(如：工程自动测量信号)；

·用于检测并报告系统是否进入某个特定的危险状态，或者在系统达到一个特定的危险状态时采取纠正措施；

 ·用于危险发生时减轻危险造成的损害；

 ·与其它安全性关键软件一起驻留于同一个安全性关键的系统（处理器）中的原本非安全性关键的软件。（注：把与安全性关键软件驻留在一起的原本是非安全性关键软件也确定为安全性关键软件，其原因是该软件的失效有可能使安全性关键软件的功能丧失或者削弱。但若采取某种方法（如：划分）使二者的代码隔离开，则应将该隔离方法确定为安全性关键的，而原本非安全性关键的软件依旧可确定为非安全性关键的）；

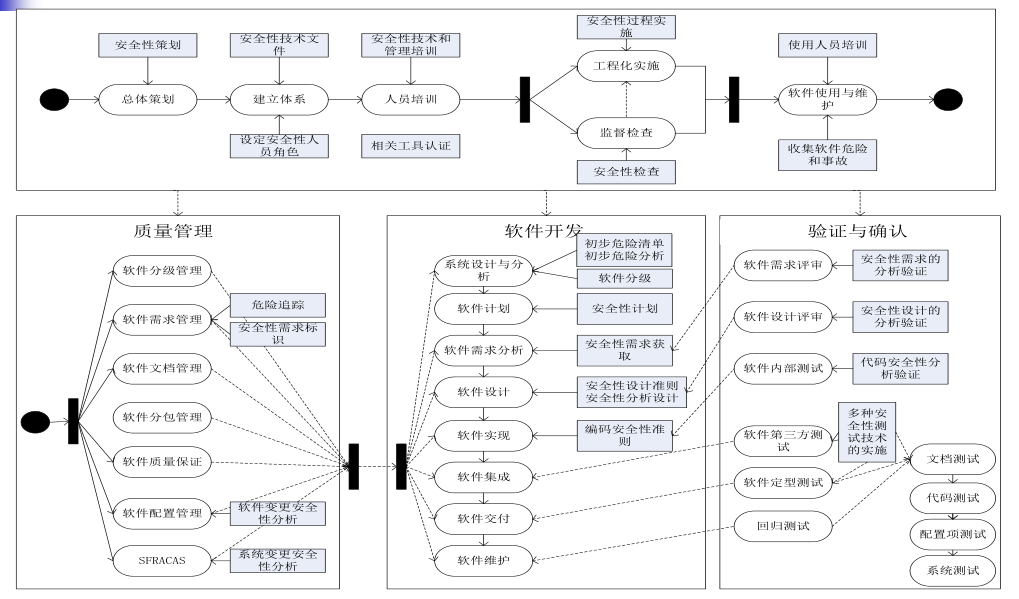
2 ） **用于处理数据或进行趋势分析，其结果直接提供给安全性决策（如：确定何时断开风洞电源以防止系统损坏）**；

3） **用于为安全性关键系统（其中包括硬件或者软件子系统）提供全部或部分的验证或确认。**

#### ！软件安全性需求

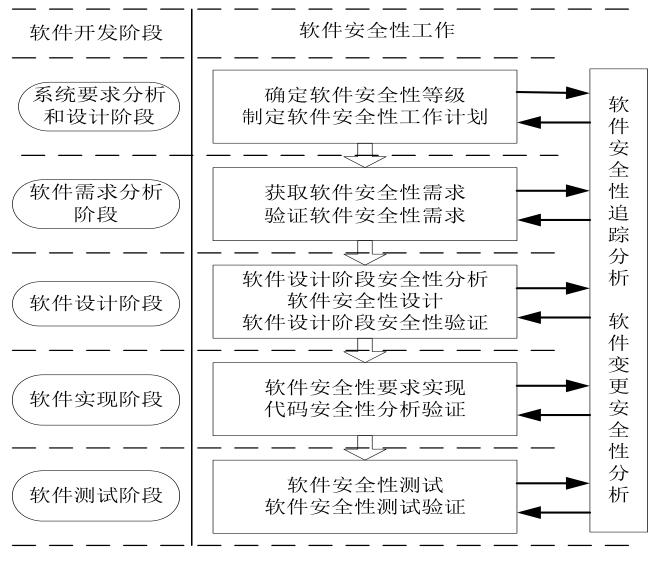
既包括安全关键的功能、性能等用户实际需要的需求，又包括用于防止、消除、检测、处理危险的需求。

## 软件安全性工程的框架



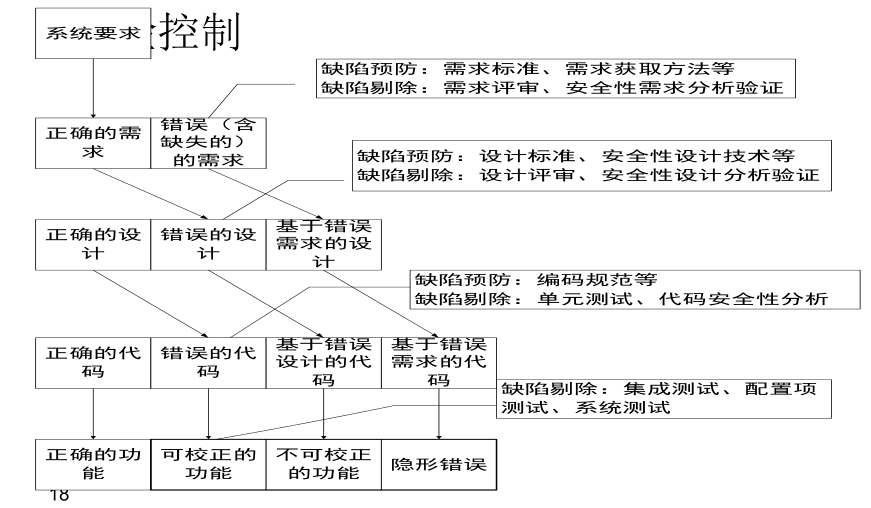
技术、过程、管理

## 软件开发各阶段的主要安全性工作

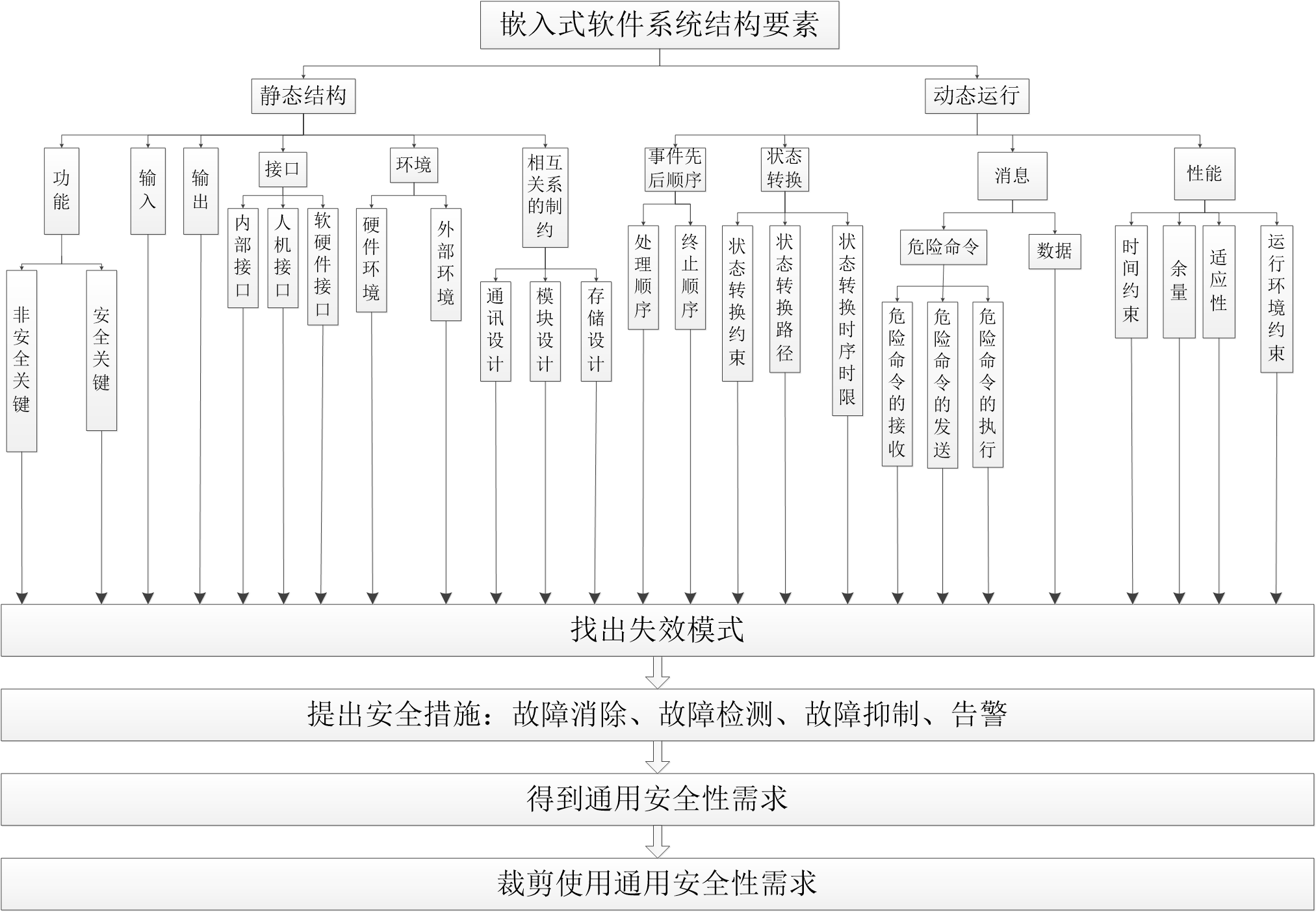


# 2.软件安全性机理和危险控制方法

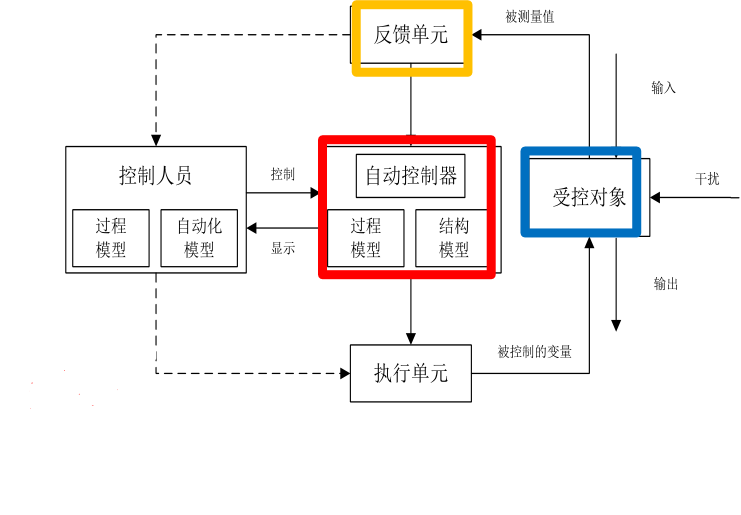
## 基于开发过程的软件安全性机理及危险控制



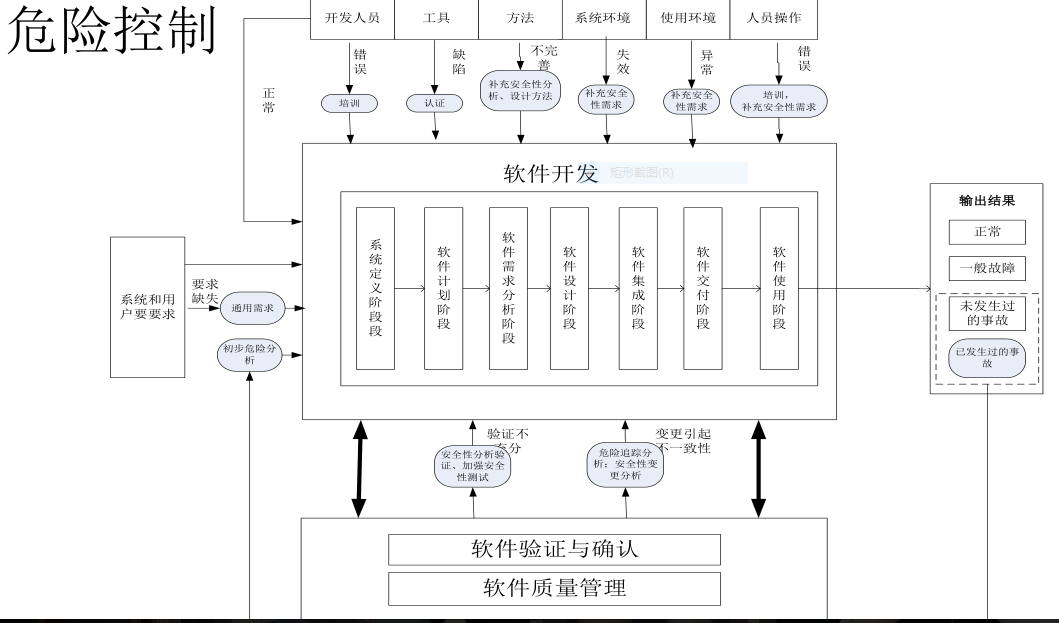
## 基于软件产品的软件安全性机理及危险控制

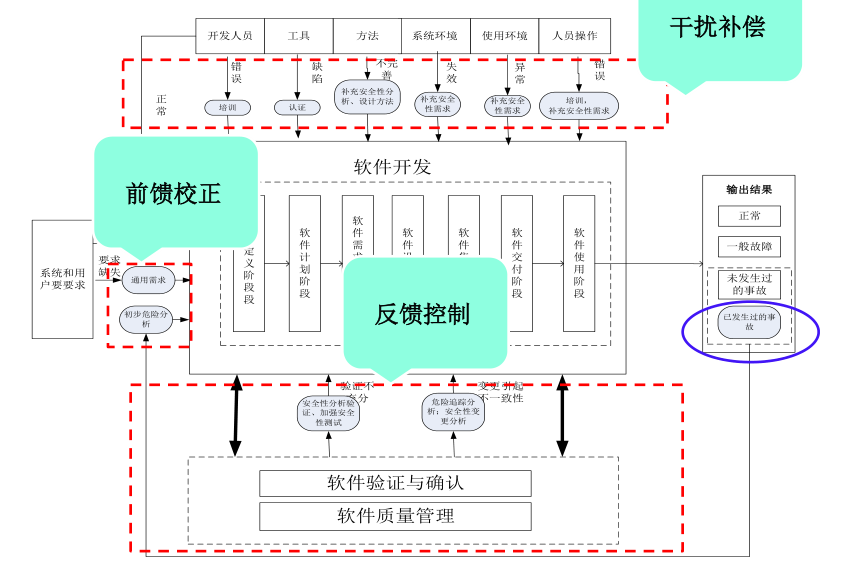


## 基于系统交互的软件安全性机理及危险控制



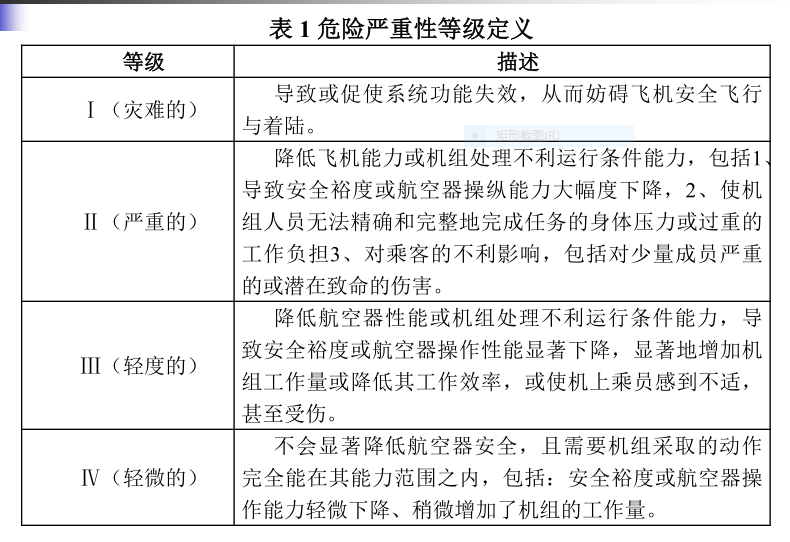
## 基于开发和使用环境的软件安全性机理及危险控制



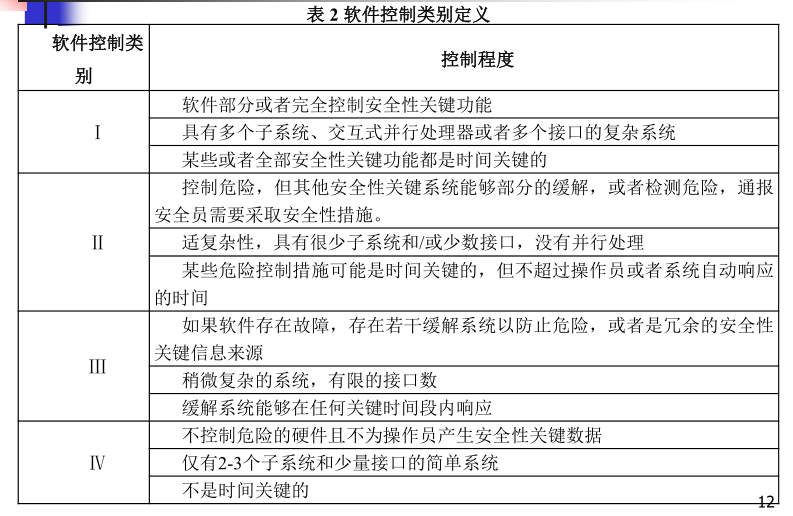


# 3.分级方法

确定软件安全性等级是软件安全性工作的重要内容之一，是在软件开发过程中的第一步工作。确定软件等级是开展后续安全性计划和安全性技术活动的基础。



## 软件控制类别



## ！等级划分过程

### 确定软件所在系统的危险严重性等级

根据**FHA**确定的各个危险的影响，选择所有危险中最严重的影响

### 确定软件对系统的控制类别

根据软件系统需求文档提供的信息，确定软件对系统功能行使的控制程度，继而确定软件所属的控制类别。

### 确定软件等级

按照建立的矩阵，可以确定软件等级，并据此决定软件后续的安全性工作开展程度。

## ！FHA

### FHA定义

对产品的功能进行系统而全面的检查，以识别各种功能故障状态，并根据故障影响的严重程度对其进行分类的一种安全性分析方法。

### 目的

 识别危险，确定安全性关键部位；

 评估各种危险的风险，确定危险影响等级；

 确定危险控制建议。

### ！分析等级

#### 飞机级（整机级）

飞机研制开始时对所规定的飞机基本功能进行的高层次定性评估，应识别与飞机级功能相关的故障状态并进行分类。

#### 系统级

实质上是FHA的迭代过程。在设计过程中将飞机功能分配到系统后，综合了多重飞机功能的每个系统必须使用系统级FHA过程重新被检查。

### 实施流程

#### 总体流程

·在方案阶段，开展飞机级FHA。

·以飞机FHA得到的故障状态为根节点，开展飞机FTA，得到系统级可能存在的各类故障状态。

·在初步设计阶段，开展系统级FHA，此时各系统的故障状态应充分考虑飞机FTA的分析结果。

·对系统级的故障模式开展FTA，得到低层次的故障原因。

·飞机和系统FTA相结合可以进行定量计算。

注意：此时的FTA同样只是对功能的分析。

#### FHA实施流程

##### 识别与分析等级相关的所有功能

内部功能和交互功能：

 内部功能

 对于整机级，内部功能即飞机的主要功能和飞机内部系统间的交互功能。

 对于系统级，内部功能是指所分析系统的功能和系统内部设备间的交互功能。

 交互功能（外部功能）

 对于整机级，外部功能是指与其它飞机或地面系统的接口功能；

 对于系统级，外部功能是指所分析系统为其它系统（包括其它飞机系统或地面系统）提供的功能或从其它系统获得的功能。

##### 2. 识别和说明故障状态

##### 3. 确定故障状态的影响

##### 4. 确定故障状态影响的分类

##### 5. 对于较低层次的故障状态分配概率要求

##### 6. 识别对故障状态的验证方法

##### 7. 给出控制建议

# 4.软件安全性需求获取方法流程、验证

软件安全性需求是从系统安全性需求中分解而来，保证系统维持在一个安全状态，同时可以对潜在失效做出充分的反应的软件需求。

需求开发（获取需求）+需求分析（验证需求）

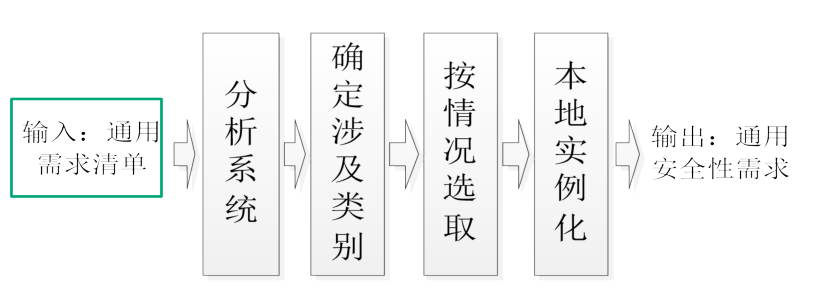


## 获取过程

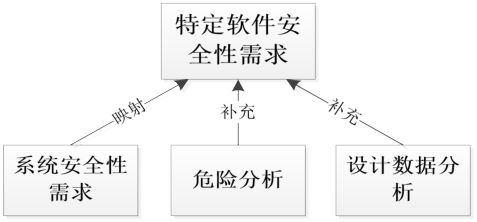
### 1）获取系统及软件信息

软件安全性工作的主线：软件安全性需求的获取和验证以及软件安全性需求在各开发阶段的实现和验证。

### 2）剪裁通用机载软件安全性需求



### 3）获取特定软件安全性需求



#### 系统安全性需求映射

获取系统需求追踪矩阵；

将系统需求追踪矩阵中的安全性关键需求映射到软件，得到软件安全性关键需求。

#### （2）危险分析

 FHA-SFTA法

 SFMEA法

 基于系统理论的危险分析方法

 需求关键性分析

#### （3）设计数据分析

# 5.软件安全性设计

软件安全性设计包括广义和狭义的概念

软件安全性设计阶段的工作包括针对设计的分析、设计、验证。

是软件安全性需求的具体落实

对于安全性关键软件，安全性设计的目标是实现最小风险设计，其中的风险包括软件缺陷产生的风险、用户操作产生的风险、费用风险和进度风险。

## 避错设计原理、准则

### 软件避错设计原理

 简单原理

 同型原理

 对称原理

 层次原理

 线型原理

 易证原理

 安全原理

### 定义

根据所标识的信息域的软件需求，以及功能和性能需求，进行数据设计、系统结构设计、过程设计、界面设计

### 避错设计的四个方面:

 程序结构化设计

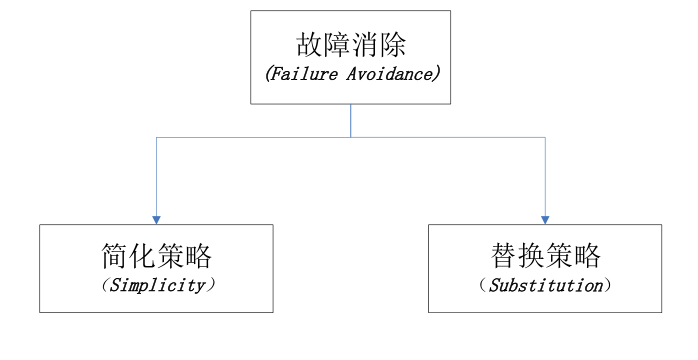
 软件简化设计

 软件健壮性设计

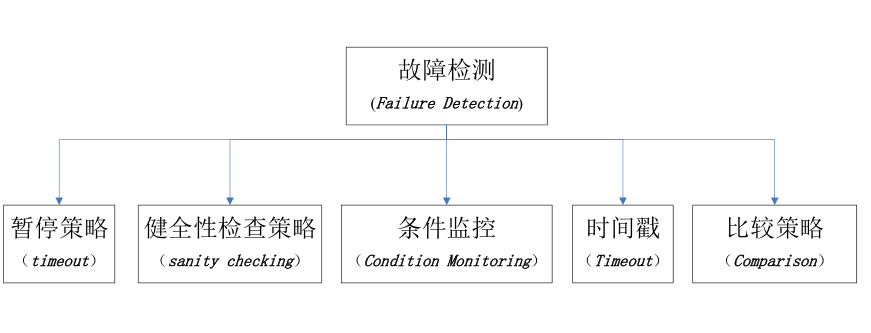
 软件冗余设计

## 安全性三种设计策略和方法、验证

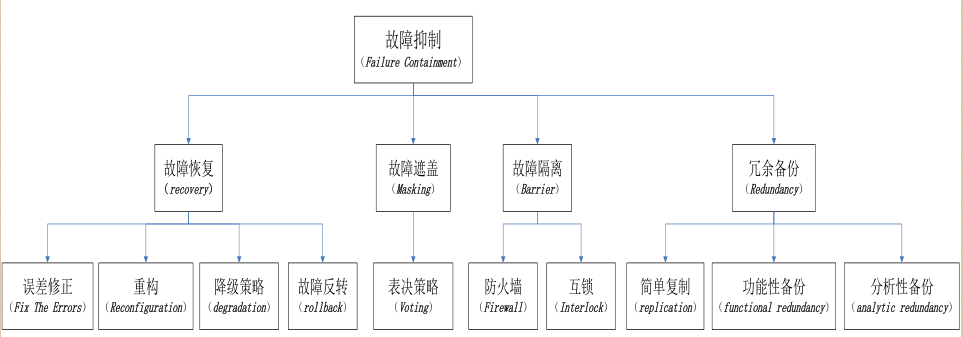
### 故障消除



### 故障检测



### 故障抑制



## 安全性设计的验证

可用需求追溯性矩阵进行跟踪，并用“如果怎么样”分析，安全性设计检查单、独立性分析等方法来验证。

# 6.软件安全性测试

## 定义

### 软件测试

软件测试以检验是否满足需求（或设计）为目标。

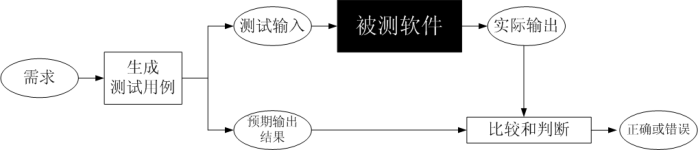
重点是发现问题

### 软件安全性测试

软件安全性测试是为了**验证软件获得的安全性**而进行的测试。

## 安全性测试方法

黑盒测试

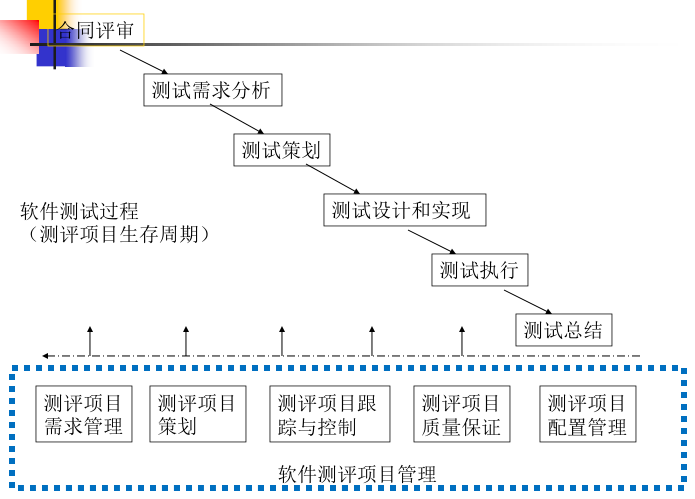


（生成测试用例、各种可能的异常输入）

### 测试用例

主要包括测试的输入和预期结果。

### 软件系统测试过程



#### 1）软件测试过程包括：

 测试需求分析

 测试策划

 测试设计和实现

 测试执行

 测试总结（包括评价过程和总结）

#### 2）软件测评项目管理包括：

 测评项目需求管理

 测评项目策划管理

 测评项目跟踪与控制

 测评项目质量保证

 测评项目配置管理

## 测试充分性验证

应验证所有的安全性需求项都经过了充分的测

试，可采取的方法有测试覆盖度量和需求追踪

矩阵等方法；可采用检查单的方法检查是否进

行了足够异常模式的测试。

# ◼其他因素

## 人为**因**素

### 过程问题

 设计失效——缺少全局观

 不良的操作规程

 错误发生的条件——过度加班

 系统目标与安全性不符

 组织失效——不切实际的目标和工作量

 沟通失效——缺少沟通渠道

 不充分的培训

 缺少安全性文化——安全性不是首要原则

### 人机交互

 改进人机交互设计可以减少错误

 界面设计

• 功能性分析

• 任务分析

• 信息分析

 需要关注的问题

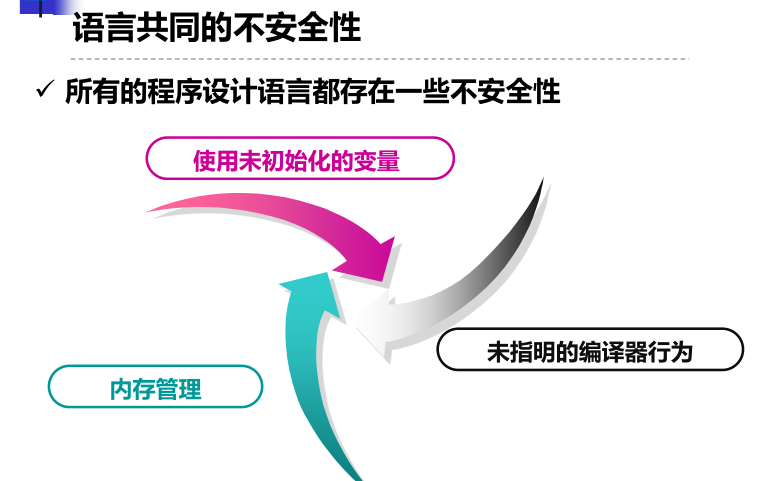
• 显示格式、布局、一致性

• 符合用户期望

• 提供适当、及时的反馈

• 允许撤销和恢复……

## 程序设计语言



### 评估方法

 语言是“安全的”——严格实施了标准，检查了常见错误

• 程序是否不能跳跃到任意位置？

• 是否可以阻止任意的内存位置被重写？

• 语义的定义是否完备？

• 是否对整型和浮点型运算都定义了严格的模型？

• 是否有可执行程序遵循运算模型的检测过程？

• 类型方法是否足够强壮以避免变量误用？

• 语言中是否有工具来防范运行时内存溢出？

• 是否提供工具对需要跨模块边界类型检查的模块进行分别编译？

• 是否可以被良好理解？

### java

 Java是平台无关的

• 不是编译成机器代码，而是字节码（.class）

• 由Java虚拟机（JVM）根据环境解释字节码

• 时间开销具有不确定性（普遍认为效率低于C语言20%）

 Java的特性

• 完全面向对象，动态读取新类

• 不支持指针，可以通过引用访问对象

• 具有垃圾回收机制

• 支持多线程、分布式系统

 Java的局限

• 无法与硬件交互、JVM需要被验证、未被标准化

• 垃圾回收机制不能被关闭

• 不能发现越界值

### Ada

 军事和安全关键应用最常用的语言之一

• 面向对象

• 强类型和范围检查

• 支持多任务和线程

• 源代码的清晰性

• 多语言支持

• 支持实时系统

Ada 被设计用于 **高完整度** 的应用

#### Ada是最安全的语言，就选它！——太天真了！

• Ada不是最流行的语言

• 寻找和拥有优秀的Ada程序员非常困难

#### 人事问题

• 学习一种语言对项目进度的影响

• 不熟悉的语言需要训练和额外的开发时间

• 需要对软件保障和软件安全性进行培训

• 缺陷率更高，需要额外的复核、检查和测试

#### 选择Ada以外的语言

• 小心局限性，建立并实施代码标准

• 预先采取主动的方法减少错误，进行完全测试

### 大部分语言表现出来的问题

#### 海森堡错误（heisenbug）

• 一种近乎无法重现的错误（和运行环境有关）

• 例如：使用未初始化自变量，调试器对其处理的不同

#### 波尔错误（Bohr bug）

• 可以重现的错误，在明确的条件下错误肯定会再次显现

#### 曼德博错误（mandelbug）

• 出现原因过于复杂和模糊以至于行为混乱的错误

• 更像是波尔错误，而不是海森堡错误

#### 薛定谔错误（schroedinbug）

• 一种几乎不会被发现的错误

• 直到有人用不平常的方式运行程序，导致程序立即停止

## 编译器

## 操作系统

 操作系统（OS）是运行应用程序的软件

### 操作系统的类型有

• 没有操作系统：通过BIOS直接装载应用程序

• 简单任务调度器：是一个迷你操作系统，在不同的应用程序（任

务）间转换

• 嵌入式操作系统：是一种功能完整的操作系统，是为小型系统设

计的

• 实时操作系统：一个RTOS具有必须达到的时间约束

• “传统”操作系统：这些系统没有时间约束，是为那些不具有资

源限制的系统设计的

# 几个核心图表

