# 可信软件系统考虑的质量属性

## 1.正确性

正确性一般分为三种定义形式，

1. 它可以被定义为系统或组件在其规范，设计和实施中没有故障的程度；
2. 它也可以被定义为软件、文档或者是其他的一些东西满足某些具体指定的需求；
3. 除此之外，正确性还可以被定义为软件或者文档满足用户的需要或者是预期，无论这种需要或者是预期是不是被具体说明了。

第一种定义是说软件系统不能包含错误，第二种定义和程序验证有关，第三种定义和需求有效性有关。

## 1.1 Requirements Validation

Requirements Validation,意为需求有效性，需求有效性会检查需求文档以保证需求是在定义一个有效的系统。

### 1.1.1相关场景：

- 刺激源：需求设计人员

- 刺激：系统需求是有效的

- 环境：需求设计构建时

- 制品：系统

- 响应：处理刺激，证明系统需求是有效的

- 相应度量：需求是否有效/确实是用户需要的

## 1.2Program Verification

Program Verification意为程序验证，程序验证是形式上证明计算机程序完全按照编写的程序规范中所述执行的过程。

### 1.2.1 相关场景

- 刺激源：运维，测试人员

- 刺激：希望验证程序完全按照规范执行

- 环境：编译时，链接时，运行时。

- 制品：系统

- 响应：处理刺激，对软件系统做正确的形式验证，

- 相应度量：是否能够按照编写时的规范完全正确的执行。

## 安全性

安全性一般是说软件系统运行的时候没有坏事发生，它也有三种定义的方式。

1. 安全性可以被定义为系统不会危害人的安全以及环境的属性；
2. 安全性也可以被定义为对风险可接受性的判断，风险是对人类健康造成伤害的可能性和严重性的度量，如果一件事物的伴随风险被认为是可以接受的，那它就是安全的；
3. 安全性还可以被定义为不会对用户和环境造成灾难性后果。

## 2.1 Safety-Critical Systems

非常关心安全性的系统可以被称之为“安全关键系统”。

### 2.1.1 Safety-Critical software

关键安全软件是可以直接或间接导致危险系统状态发生的任何软件。

### 2.1.2 Safety-Critical function

关键安全接口函数是指那些系统功能，如果操作正确，操作错误（包括在错误的时间进行正确操作）或缺少操作，可能会导致系统风险。

### 2.1.3 Safety-Critical Software Function

关键安全软件功能是指那些可以直接或间接地与其他系统组件的行为或环境条件相辅相成的软件功能，导致存在危险状态。

## 2.2 Safety and Liveness Properties

在系统和软件验证领域，稳定意味着永远不会发生坏事。 因此，稳定属性可以被定义为不会发生某些事情的属性。稳定性与活动性相辅相成，活动属性表示好事情最终会发生。

## 2.3 Legal Issues of Safety

信息的可软件系统需要遵守某些安全标准，这些标准会通过安全预防措施影响用性，完整性和机密性。

## 2.4 相关场景

- 刺激源：系统

- 刺激：正确或者错误的指令，函数，结果。

- 环境：正常、降级操作

- 制品：安全关键系统

- 响应：执行操作/拒绝执行指令

- 相应度量：系统的决定是否会对用户和环境造成很大的风险。

# 服务质量（Quality of Service)

在执行给定服务时保证预定义的质量级别通常用术语服务质量和首字母缩写 QoS 来表示。这也意味着质量可以通过某种方式使用度量来衡量。在 Internet 和其他网络中，QoS是指传输速率、错误率和其他特性可以被测量、改进，并且在某种程度上可以预先得到保证的属性。一般来说，QoS 指的是非功能属性，如性能、可靠性、可用性和安全性。

### 3.1 Availability

Availability的意思为可用性和可行性，可用性和可靠性密切相关，两者都涉及动态系统的行为，但是他们通常被视为系统的不同方面。可用性是一个系统对正确服务的准备状态，它是断言资源在给定时间段内可用或可操作的属性，它描述系统在某个时间点，是否可以运行并能够提供所要求的服务。

### 3.2 Reliability

可靠性是安全关键系统和任务关键系统最重要的质量属性之一。与正确性相反，可靠性指的是系统的动态行为，而不是静态源代码或集成电路的特征。经典的可靠性模型区分了硬件可靠性和软件可靠性。这种区别建立在软件可靠性受设计错误影响的假设上，而硬件可靠性的主要影响因素是退化和环境干扰。目前，软件可靠性和硬件可靠性的严格分离已不再有效。一方面，硬件开发也处理设计错误。另一方面，执行环境（例如并发操作过程）影响软件产品的可靠性。在概念层面上，区分硬件和软件可靠性有助于预测系统可靠性，因为这两个概念使用不同的预测模型，并对系统作出不同的假设。因此，经典的区分需要继续存在，但不能被软件和硬件分开。相反，有价值的是区分设计故障可靠性和劣化可靠性。此外，软件和硬件可靠性并不独立存在。软件应用程序的可靠性受底层硬件可靠性的影响，反之亦然。任务可靠性是衡量一个项目在一个特定的任务概要期间执行其所需功能的能力。考虑到所有可能的冗余操作模式，它定义了系统不会失败完成任务的概率。

### 相关场景

- 刺激源：内部、外部

- 刺激：安排一个任务

- 环境：正常、降级、冗余操作

- 制品：系统的处理器、通信信道、持久存储器、进程

- 响应：执行任务并返回成功/未成功完成任务的结果

- 相应度量：可以用成功完成任务的概率表示

### 3.3 Performance

性能是指实现某一目标的能力。在计算机科学中，术语的内涵是指计时行为（处理请求需要多长时间或者一段时间内可以执行多少个请求）和计算机系统的资源效率（一个设备的利用率是多少）。大多数关于计算机性能分析的文献没有明确定义性能这一术语。然而，这些出版物中术语的相关含义可以从用于衡量性能的指标中得出。常用的度量标准是响应时间（以秒为单位）、吞吐量（以每个时间单位的请求为单位）和利用率（以百分比为单位，以每个总时间的繁忙时间为单位）。因此，计时行为和资源利用率是 TrustSoft 中“性能”一词的基本内涵。一般来说，性能是软件系统或组件满足其及时性目标的程度。因此，性能是软件产品的基本特征，原则上，用户可以坐在电脑前，手里拿着秒表来衡量。

### 相关场景

- 刺激源：大量的独立源中的一个，可能来自系统内部

- 刺激：定期事件到达、随时时间到达、偶然事件到达

- 环境：正常模式、超载模式

- 制品：系统

- 响应：处理刺激、改变服务级别

- 相应度量：等待响应、期限、吞吐量、抖动、缺失率、数据丢失