型的现代车，符合韩国汽车标准，里面有一个GPS。

软件的质量属性建筑学

鄂尔多斯市
北京交通大学

内容

- 质量属性介绍
- 质量属性情况
- 典型的质量属性
  - 性能属性
  - 可扩展性属性
  - 可修改属性
  - 安全属性
  - 可用性属性
  - 可用性属性
- 设计权衡

质量属性介绍

质量属性

- 业绩
- 安全问题
- 可利用性
- 可扩展性
- 可用性
- 可靠性
- 便携性
- 可修改性

- 可维护性

质量属性的功能

- 实现质量属性必须在整个设计、实施和部署过程中加以考虑
  - 可用性
    - 为用户提供撤销操作（架构）。
    - 使用户界面易于使用（非结构性的）。
  - 可修改性
    - 功能是如何划分的（架构）。
    - 通过模块内的编码技术（非结构性的）。
  - 业绩
    - 组件之间需要多少通信，以及如何分配共享资源（架构）。

- 算法的选择及其编码方式（非架构）。

质量属性的规范

- 建筑师经常被告知，他们的方式太不精确了
  - "我的应用程序必须是快速/安全/规模化的"
- 对于特定的系统设计，质量属性（QAs）必须是精确的/可测量的
  - "必须能够将部署从最初的100个地理上分散的用户桌面扩展到10000个，而不增加安装和配

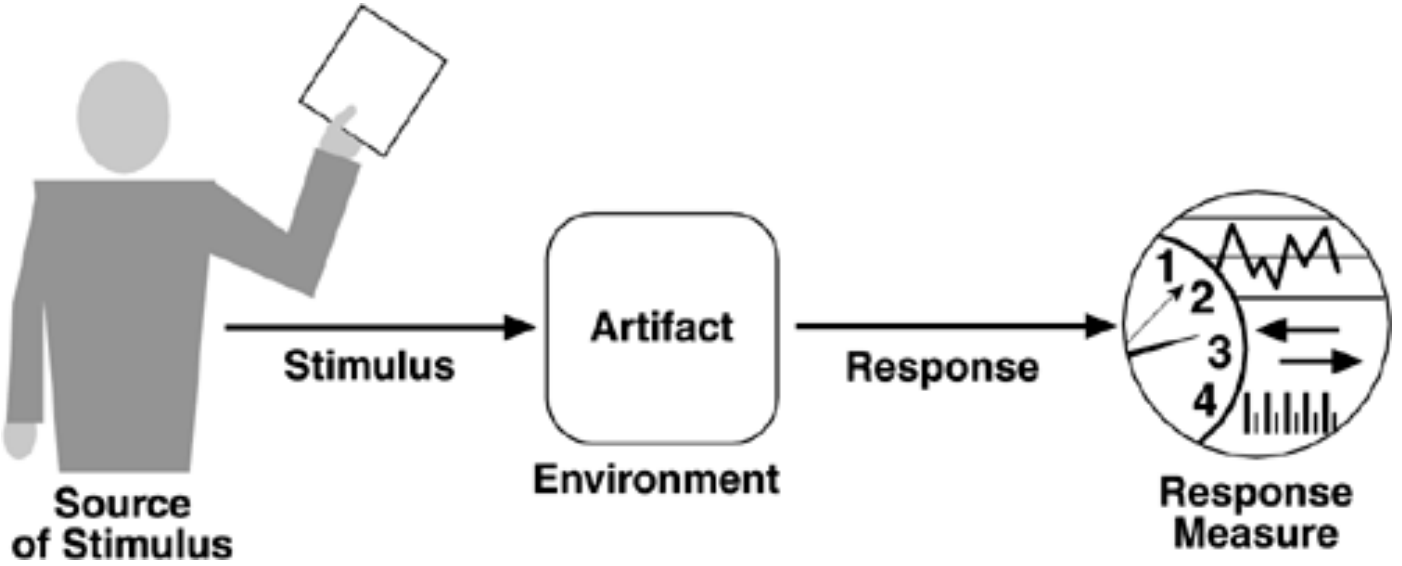
置的努力/成本"

质量属性情况

质量属性情况

- 一个质量属性的场景是一个质量属性的具体要求
  - 刺激的来源--产生刺激的实体
  - 刺激--当它到达一个系统时需要考虑的条件
  - 环境 - 刺激物发生的特定条件
  - 人工制品--系统或被刺激的部分
  - 反应 - 刺激物到达后进行的活动
  - 响应测量--当响应发生时，它应该可以通过某种方式进行测量，以便需求可以得到测试

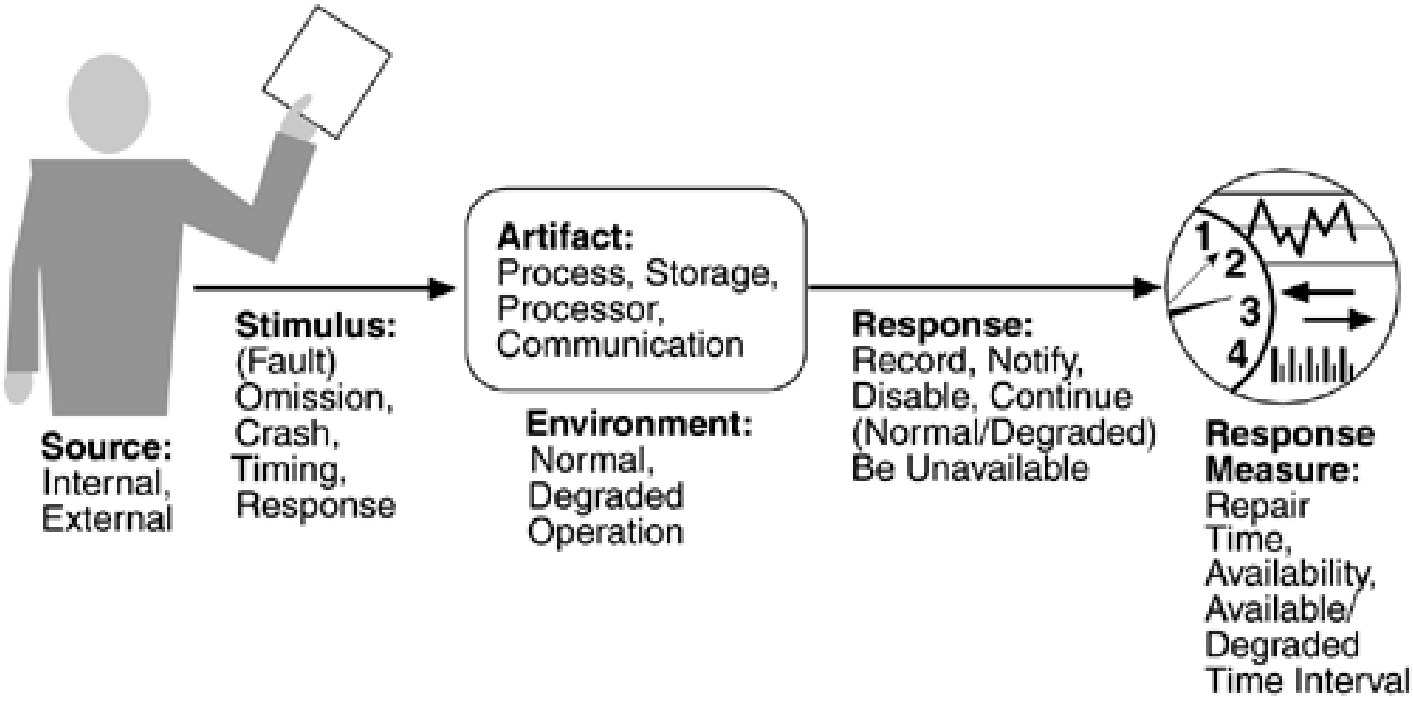
质量属性情况


Source
of Stimulus
Stimulus
Artifact
Environment
Response
1
2
3
4
Response
Measure

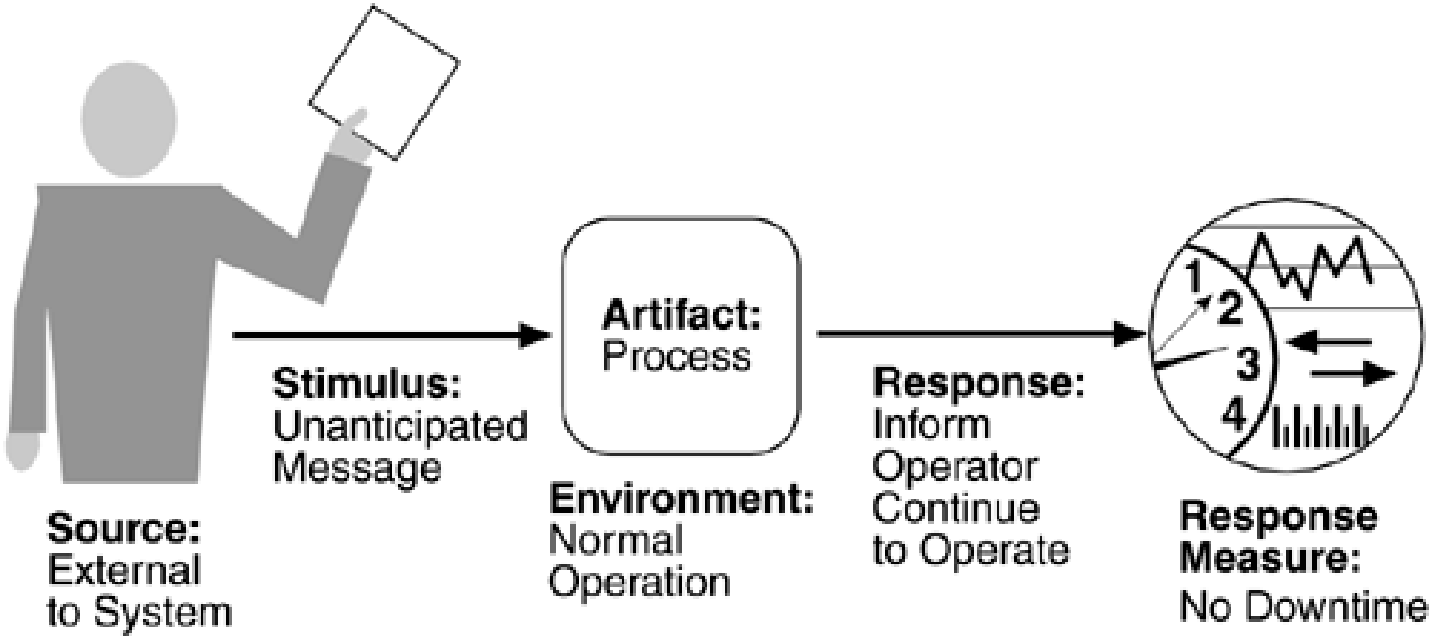
一般情况与混凝土质量属性方案

- 一般情况下是独立于系统的，有可能涉及到任何系统
- 一个具体的场景是针对所考虑的特定系统的
  - 要使质量要求具有可操作性，需要有具体的场景
- 具体情景是一般情景的一个子集

普遍可用情况


Source:
Internal,
External
Stimulus:
(Fault)
Omission,
Crash,
Timing,
Response
Artifact:
Process, Storage,
Processor,
Communication
Environment:
Normal,
Degraded
Operation
Response:
Record, Notify,
Disable, Continue
(Normal/Degraded)
Be Unavailable
1
2
3
4
Response
Measure:
Repair
Time,
Availability,
Available/
Degraded
Time Interval

混凝土供应情况


Source:
External
to System
Stimulus:
Unanticipated
Message
Artifact:
Process
Environment:
Normal
Operation
Response:
Inform
Operator
Continue
to Operate
1
2
3
4
Response
Measure:
No Downtime

具体方案的功能

- 一组具体的场景可以作为一个系统的质量属性要求。
  - 具体场景在质量属性规范中的作用与用例在功能需求规范中的作用相同。
- 响应的细节有足够的意义，以便有可能测试系统是否实现了该响应

具体方案的产生

- 理论上，质量属性要求应该在需求分析期间获得，但在实践中很少这样做。
- 架构师的任务是通过生成具体的质量属性方案来确保这一点的实现。

具体方案的产生

- 质量属性的具体表格被用来创建一般方案，并从这些方案中指定具体方案
  - 这些表格可以作为检查表，以确保所有的可能性都得到考虑
- 从不同的质量属性生成相同或相似的场景并不重要，因为冗余可以很容易地被删除。
  - 重要的是，没有遗漏任何重要的要求。

一般可用性表
场景

方案的部分内容	可能的价值
来源	系统内部；系统外部
刺激措施	过错；遗漏、崩溃、时机、反应
创作	系统的处理器、通信通道、持久性存储进程
环境	正常操作；降级模式（即较少的功能，是一种回避方案。）
响应	系统应检测到事件并做以下一项或多项工作：记录事件 通知适当的各方，包括用户和其他系统禁用的导致故障或失效的事件的来源 在一个预先指定的时间间隔内无法使用 继续在正常或降级模式下运行
响应措施	系统必须可用的时间间隔；可用性时间；系统可以处于降级模式的时间间隔；修复时间

典型的质量属性
业绩

业绩

- 性能要求
  - 单位时间内完成的工作量的度量
  - 必须遵守的最后期限
  - 以此类推
- 解决方案
  - 将关键业务本地化，尽量减少通信
  - 使用粗粒度而非细粒度的组件

– 以此类推

业绩

- 企业应用往往有严格的性能要求，例如
  - 每秒1000个交易
  - 一个请求的平均延迟时间为3秒
- 许多企业应用程序性能不佳的例子

吞吐量

- 衡量一个应用程序在单位时间内必须完成的工作量
  - 每秒交易量
  - 每分钟信息量
- 是所需的吞吐量。
  - 平均值？
  - 峰值？

响应时间

- 衡量一个应用程序在处理一个请求时表现出的延迟。
  - 通常以秒（或毫秒）为单位。
- 是要求的响应时间。
  - 担保？
  - 平均值？

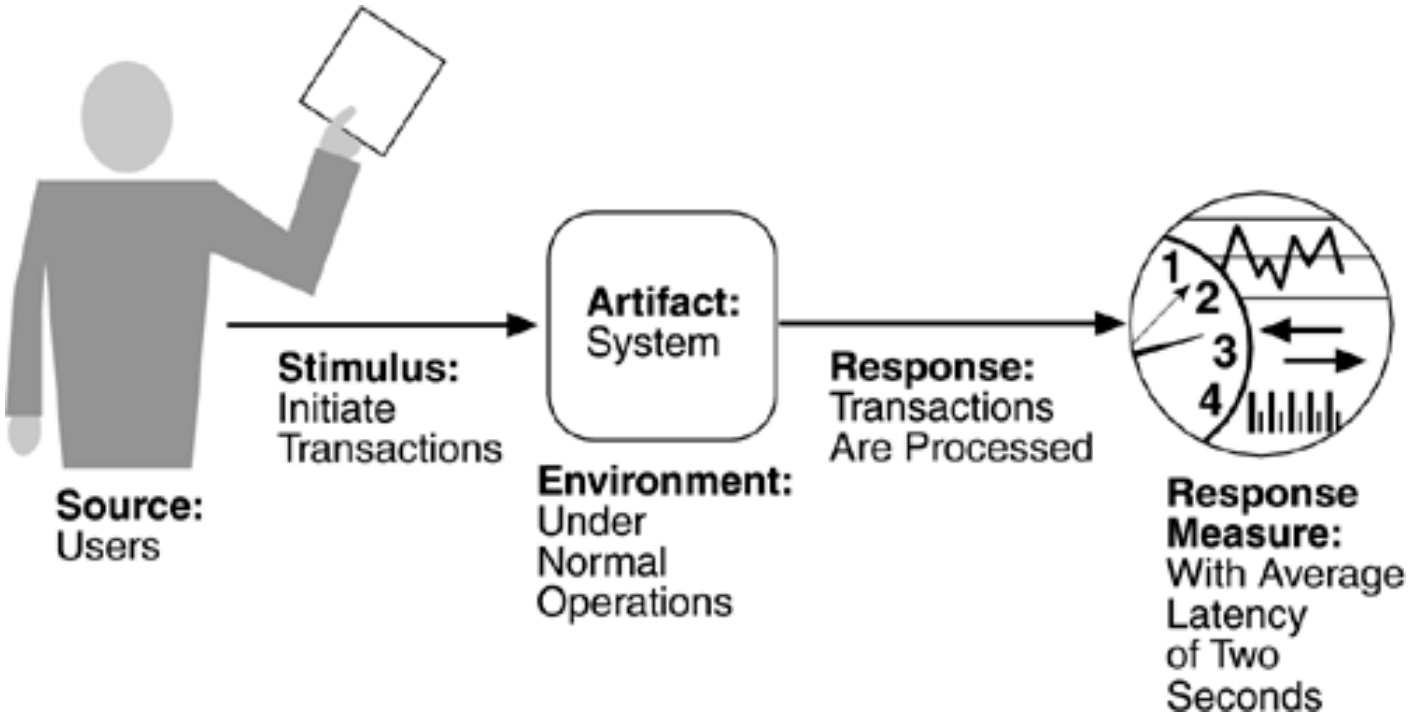
最后期限

- 某事必须在某个特定时间之前完成
  - 薪资系统必须在凌晨2点前完成，以便向银行发送电子转账。
  - 每周的会计工作必须在周一早上6点前完成，以便向管理层提供数字。
- 最后期限通常与IT系统中的批处理工作有关

一般性能表
场景

方案的部分内容	可能的价值
来源	若干独立来源之一，可能来自系统内部
刺激措施	周期性事件到来；零星事件到来；随机事件到来
创作	系统
环境	正常模式；过载模式
响应	处理刺激物；改变服务水平
响应措施	延迟、期限、吞吐量、失误率、数据丢失

混凝土性能情况


Source:
Users
Stimulus:
Initiate
Transactions
Artifact:
System
Environment:
Under
Normal
Operations
Response:
Transactions
Are Processed
1
2
3
4
Response
Measure:
With Average
Latency
of Two
Seconds

典型的质量属性
可扩展性

可扩展性

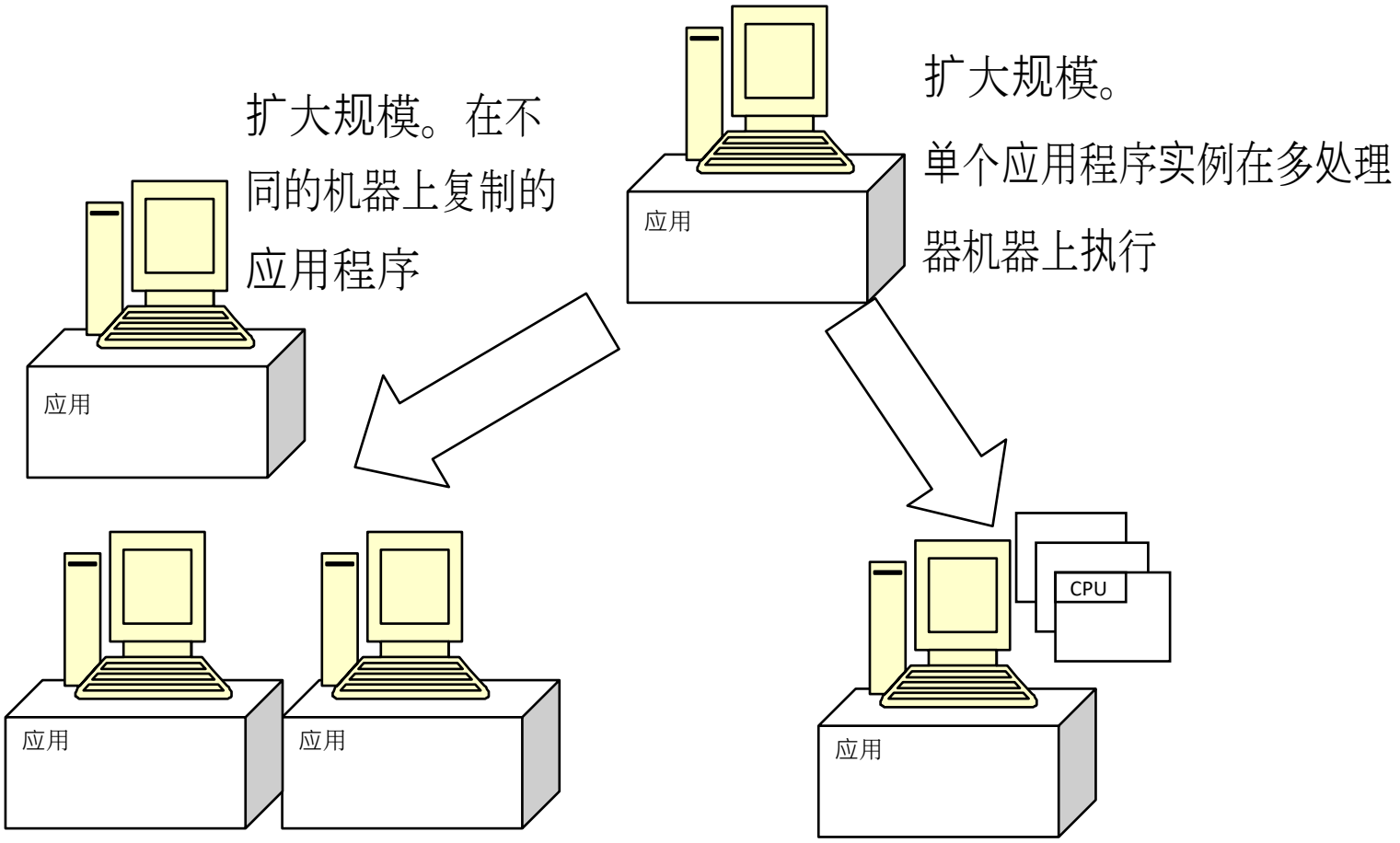
- 可扩展性是一个系统、一个网络或一个进程的理想属性，它表明它有能力以一种优雅的方式处理不断增长的工作量，或随时扩大规模。
- IT系统中4个常见的可扩展性问题
  - 请求加载
  - 连接
  - 数据大小

– 部署

请求负载

- 当同时请求负载增长时，一个100rps的应用程序是如何表现的？
  - 从每秒钟100个请求到1000个请求？
- 理想的解决方案是随着负载的增加，吞吐量保持不变（即100rps），每个请求的响应时间仅线性增加（即10秒）。

另一种解决方案


扩大规模。在不同的机器上复制的应用程序
扩大规模。
单个应用程序实例在多处理器机器上执行
应用
应用
应用
应用
应用
CPU

连接

- 如果一个应用程序的同时连接数增加会怎样？
  - 如果每个连接都要消耗一个资源？
  - 如果连接数超过了最大数量？

数据大小

- 当一个应用程序处理的数据规模增加时，它是如何表现的？应用程序/算法能否扩展以处理增加的数据需求？
  - 聊天应用程序的平均信息量增加了一倍？
  - 数据库表的大小从100万行增长到2000万行？
  - 图像分析算法处理的图像是100MB而不是

1MB？

部署

- 安装/部署一个应用程序的努力如何随着安装基础的增长而增加？
  - 安装新用户？
  - 安装新的服务器？
- 解决方案通常围绕着自动下载/安装展开
  - 例如，从互联网上下载应用程序

可扩展性

- 经常被忽视，因为不是应用失败的主要原因
- 难以预测，难以测试/验证
- 依靠成熟的设计和技术是至关重要的

典型的质量属性
可修改性

可修改性

- 可修改性衡量改变一个应用程序以满足新的（非）功能需求的难易程度。
  - 5月'--几乎总是无法确定的
- 可修改性是关于变化的成本
  - 必须估计成本/努力
- 应在系统可能发生变化的情况下评估可修改性
  - 不需要考虑极不可能发生的变化

战略

- 策略1：事先思考
  - 令人信服的变化的可能性和影响分析
  - 演示如何通过相应的解决方案来适应修改的要求
- 策略2：尽量减少依赖性
  - 孤立于单个组件的变化可能比那些在整个架构中引起连锁反应的变化成本低。

典型的质量属性
安全问题

安全问题

- 安全性是衡量系统在向合法用户提供服务的同时抵制未经授权使用的能力。
- 试图破坏安全是一种攻击--可能是为了获取数据或服务，或拒绝向他人提供服务

战略

- 认证：应用程序可以验证其用户和与之通信的其他应用程序的身份。
- 授权：经过认证的用户和应用程序对系统的资源有明确的访问权。
- 加密：向/从应用程序发送的信息是加密的

战略

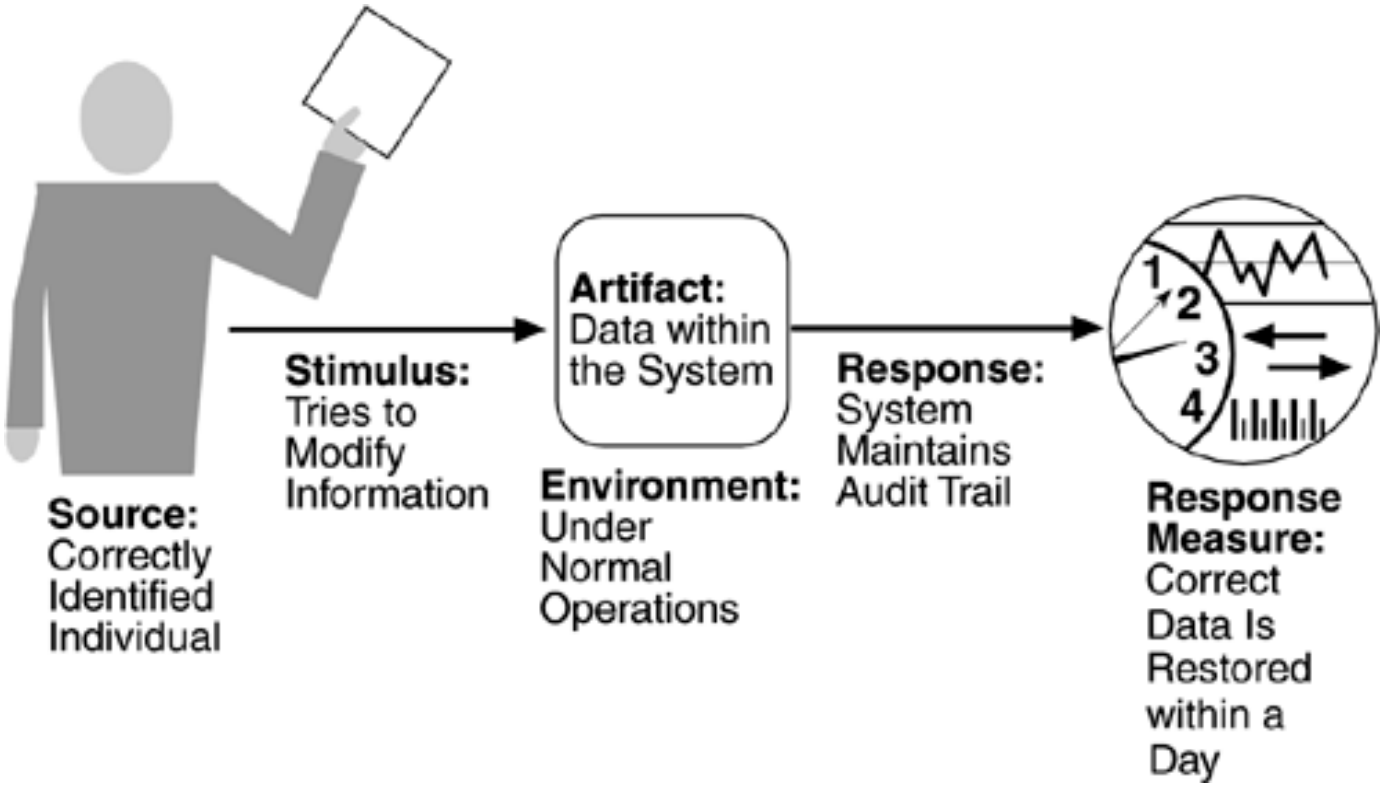
- 完整性：信息的内容在传输过程中不被改变。
- 不可否认性：信息的发送者有交付证明，接收者确信发送者的身份。
  - 两者随后都不能反驳他们对信息交换的参与，也就是说，交易的任何一方都不能否认。
- 审计：系统跟踪其内部的活动，其水平足以重建这些活动

一般安全情况表

方案的部分内容	可能的价值
来源	正确识别、错误识别、身份不明的个人或系统，他们是内部/外部、授权/非授权的，可以使用有限的资源、巨大的资源。
刺激措施	试图显示数据，改变/删除数据，访问系统服务，减少对系统服务的可用性
创作	系统服务；系统内的数据
环境	无论是在线或离线，连接或断开，防火墙或开放
响应	认证用户；隐藏用户的身份；阻止对数据和/或服务的访问；允许对数据和/或服务的访问；授予或撤销对数据和/或服务的访问权限；记录访问/修改或试图按身份访问/修改数据/服务；以不可读的格式存储数据；识别无法解释的高服务需求，并通知用户或其他系统，并限制服务的可用性

响应措施	规避安全措施所需的时间/精力/资源，以及成功的概率；发现攻击的概率；确定攻击或访问/修改数据和/或服务的责任人的概率；在拒绝服务攻击下仍然可用的服务百分比；恢复数据/服务；数据/服务受损和/或合法访问被拒绝的程度

具体的安全情况


Source:
Correctly
Identified
Individual
Stimulus:
Tries to
Modify
Information
Artifact:
Data within
the System
Environment:
Under
Normal
Operations
Response:
System
Maintains
Audit Trail
1
2
3
4
Response
Measure:
Correct
Data Is
Restored
within a
Day

典型的质量属性
可利用性

可利用性

- 可用性涉及到系统故障及其后果
  - 大多数IT应用的关键要求
  - 如何检测系统故障
  - 系统故障发生的频率如何
  - 当故障发生时，会发生什么
  - 一个系统被允许停止运行多长时间
  - 何时可能安全地发生故障
  - 如何防止故障的发生
  - 当发生故障时，需要什么样的通知

可利用性

- 以可使用的规定时间比例来衡量
  - 工作时间内100%可用
  - 每周安排的停机时间不超过2小时
  - 24 x 7 x 52 (100%可用)
- 丧失可用性的时间由以下因素决定
  - 检测故障的时间
  - 纠正故障的时间
  - 重新启动应用程序的时间
- 与一个应用程序的可靠性和可恢复性有关
  - 不可靠的应用程序的可用性很差
  - 系统故障后恢复受影响数据的能力

战略

- 消除单点故障
- 复制和故障转移
- 自动检测和重新启动

一般可用性表
场景

方案的部分内容	可能的价值
来源	系统内部；系统外部
刺激措施	过错；遗漏、崩溃、时机、反应
创作	系统的处理器、通信通道、持久性存储进程
环境	正常操作；降级模式（即较少的功能，是一种回避方案。）
响应	系统应检测到事件并做以下一项或多项工作：记录事件 通知适当的各方，包括用户和其他系统禁用的导致故障或失灵的事件的来源 在一个预先指定的时间段内无法使用 继续在正常或降级模式下运行

响应措施	系统必须可用的时间间隔；可用性时间；系统可以处于降级模式的时间间隔；修复时间

典型的质量属性
可用性

可用性

- 可用性是指用户完成一项预期任务的难易程度以及所提供的用户支持的种类。
  - 学习系统的特点
  - 有效地使用一个系统
  - 最大限度地减少错误的影响
  - 使系统适应用户需求

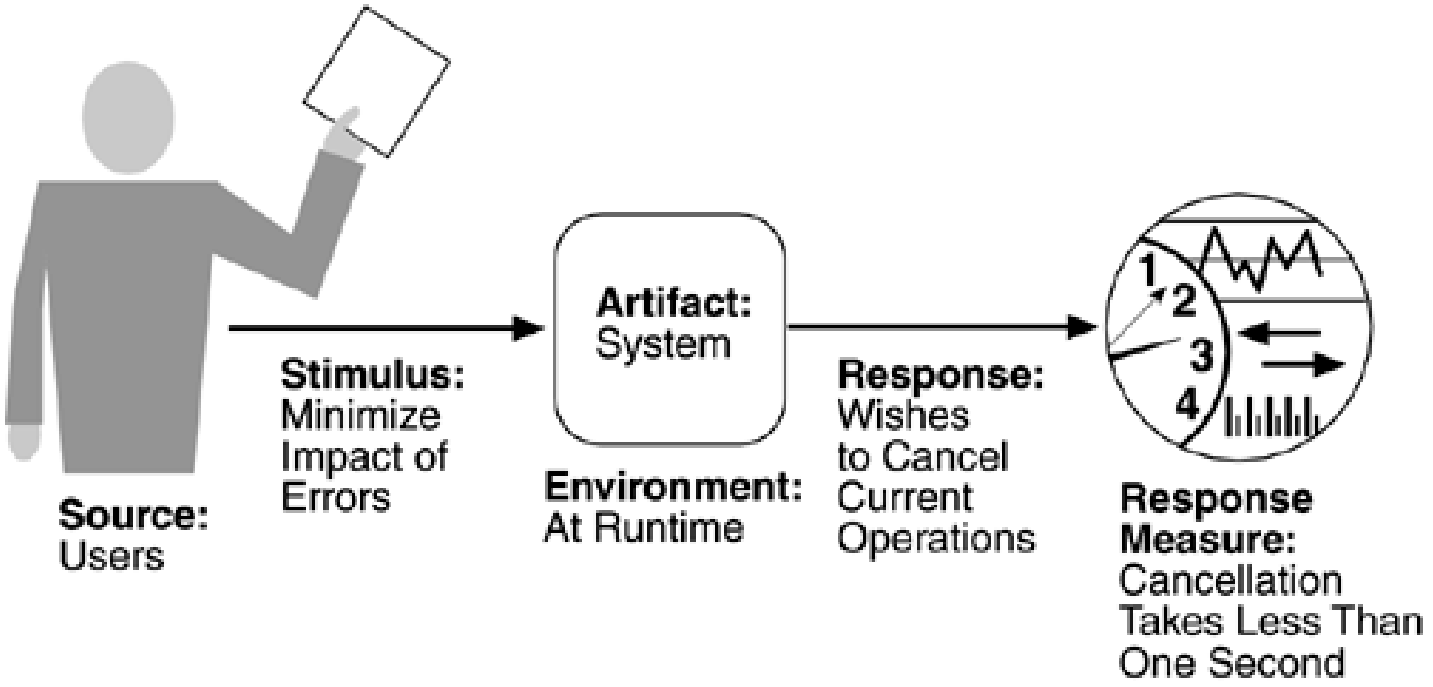
– 增加信心和满意度

一般可用性情况表

方案的部分内容	可能的价值
来源	终端用户
刺激措施	希望学习系统功能；有效使用系统；尽量减少错误的影响；适应系统；感觉舒适
创作	系统
环境	在运行时或配置时
响应	系统提供以下一个或多个回应。 支持 "学习系统功能"--帮助系统对环境敏感；界面对用户来说是熟悉的；界面在不熟悉的环境中是可用的 支持 "有效使用系统"--数据和/或命令的汇总；重复使用已经输入的数据和/或命令；支持屏幕内的有效导航；具有一致操作的不同视图；全面搜索；多个同时进行的活动，以尽量减少 "错误的影响"--撤销、取消、从系统故障中恢复、识别和纠正用户错误、检索遗忘的密码、验证系统资源 "适应系统"--可定制性；国际化 以 "感觉舒适" - 显示系统状态；以用户的速度工作

响应措施	任务时间、错误数量、解决的问题数量、用户满意度、用户知识的获得、成功操作与总操作的比率；时间/数据损失量

具体的可用性方案


Source:
Users
Stimulus:
Minimize
Impact of
Errors
Artifact:
System
Environment:
At Runtime
Response:
Wishes
to Cancel
Current
Operations
1
2
3
4
Response
Measure:
Cancellation
Takes Less Than
One Second

设计权衡

设计权衡

- 质量属性很少是正交的，它们相互作用，相互影响。
  - 使用粗粒度的组件可以提高性能，但会降低可维护性
  - 引入冗余数据可以提高可用性，但会使安全变得更加困难
  - 安全相关功能的本地化通常意味着更多的通信，因此降低了性能。

设计权衡

- 建筑师必须创造解决方案，做出合理的设计妥协
  - 不可能完全满足所有竞争的要求
  - 必须满足所有利益相关者的需求