第三部分 软件测试应用

3.3 系统测试

内容回顾

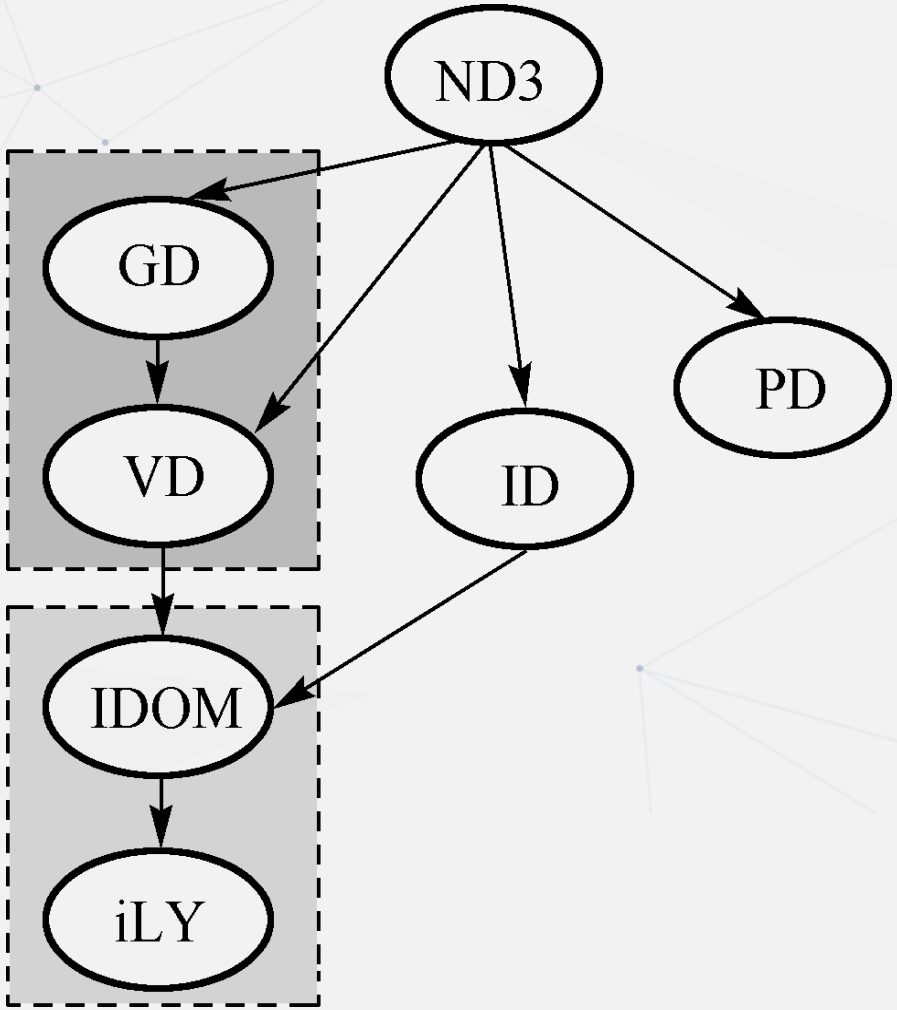
- 单元测试
  - 基本概念：软件中最小可测试单元或基本组成单位进行检查和验证（函数、类、窗口或菜单）
  - 测试内容与方式：
    - 静态检查：通过审查、走查等方式，查看其是否符合标准和规范
    - 动态检查：模块接口、模块边界条件、模块独立路径和错误处理进行测试
  - 单元测试步骤：
    - 做静态和动态检查
    - 编写测试用例进行测试（借鉴黑盒测试用例设计方法）
    - 使用判定覆盖或独立路径覆盖进行测试

内容回顾

- 集成测试
  - 基本概念：将所有已通过单元测试的模块按照概要设计的要求组装为子系统或系统，并进行测试的过程，目的是确保各单元模块组合在一起能够按照既定意图协作运行，并确保增量的行为正确
  - 测试内容：穿越模块接口的数据是否会丢失；子模块组合是否能达到预期结果；一个模块的功能是否会对其他模块造成影响.....
  - 驱动模块和桩模块的概念

内容回顾

- 集成测试的方法
  - 成对集成
  - 邻居集成
  - 基于独立路径集成
- 集成测试的遍历顺序
  - 自顶向下
  - 自底向上
  - 混合（三明治）方法



本节教学目标

- 理解系统测试的基本概念
- 掌握系统测试的过程及方法

目录

系统测试概述

- 如果产品/项目经过了单元测试、集成测试，是否还需要系统测试？
- 系统测试包含什么？
- 系统测试是公司和项目组最关心的测试阶段，在任何情况下都必须执行，一般由测试经理统一组织和制订系统测试计划，其他测试人员分别负责测试的分析、设计、实施和执行

目录

系统测试概述

- 什么是系统测试？
  - 例如：
    - 手机软件完成集成测试后，将其植入相应硬件做整体测试
    - 图书馆借书软件完成集成测试后，将其植入借书系统的硬件做整体测试
  - 定义：系统测试就是将经过良好的集成测试的软件系统，作为整个计算机系统的一部分，与计算机硬件、外部设备、支持软件、数据及人员等其他系统元素结合在一起，在实际使用(运行)环境下对计算机系统进行一系列的严格测试来发现软件中的潜在缺陷，保证系统交付给用户之后能够正常使用

系统测试概述

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

功能测试

- 功能测试(Function Testing)主要针对系统的功能需求展开测试，以确认被测系统是否满足用户的功能使用要求
- 是系统测试中最基本的测试

功能测试举例

- 例如：以数据为中心的系统
- 核心是数据处理
  - 从实体关系模型设计测试
  - 从对数据的操作设计测试

功能测试举例

- 从实体关系模型设计测试
  - 1对1：此时仅需一类测试用例，创建1对1的对象实例即可；
  - 1对多：可结合边界值和等价类测试，分别创建1对1、1对2、1对多这三类对象实例；
  - 多对1：与1对多相似，分别创建1对1、2对1、多对1这三类对象实例；
  - 多对多：应参照1对1、1对多、多对1这三种情况分别创建对象实例

功能测试举例

- 从对数据的操作设计测试
- 1、增加
  - 能否正常实现增加操作
  - 针对唯一性字段，测试输入重复的情况，判断系统是否会报错
  - 针对必填项，测试是否有提示信息
  - 测试增加成功后能否方便地看到增加的结果
  - 测试增加一项或一组数据是否对其他数据产生影响，以及该影响是否符合用户需求

功能测试举例

- 从对数据的操作设计测试
- 2、删除
  - 针对一项或一组对象的删除操作能否正常实现
  - 测试是否会错误地删除不存在的对象，或未选中的对象
  - 测试删除之前是否有提示信息，以及删除成功后能否方便地看到删除的结果
  - 测试删除一项或一组数据是否对其他数据产生影响，以及该影响是否符合用户需求

功能测试举例

- 从对数据的操作设计测试
- 3、修改
  - 测试是否会错误地修改不存在的对象，或未选中的对象
  - 测试通过明确修改某些信息后能否确保所有隐含信息得到正确的修改
  - 参照增加操作需测试的各个方面展开测试

功能测试举例

- 从对数据的操作设计测试
- 4、查找
  - 测试系统能否支持简单查询和高级查询
  - 测试系统是否针对存在和不存在的内容均给出正确的查找结果
  - 测试系统能否针对合理和不合理的条件进行正确的处理
  - 测试系统能否将查找结果与删除、修改等操作方便地结合起来

功能测试举例

- 结合黑盒测试的思想设计测试
- 针对系统输入和输出，考虑对所有输入和输出的覆盖测试
  - 测试所有可以接受输入和进行输出的硬件设备
  - 测试所有的软件输入条件和输出结果
  - 测试输入(输出)条件的边界情况
  - 测试输入(输出)条件的典型情况
  - 测试所有不合理的输入情况

系统测试概述

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

性能测试概述

- 定义：性能测试(Performance Testing)就是对软件的运行性能指标进行测试，判断系统集成之后在实际的使用环境下能否稳定、可靠地运行
- 主要考虑系统的时间和空间性能
  - 时间主要指软件的一个具体事务的响应时间
  - 空间性能主要指软件运行时消耗的系统资源

性能测试主要内容

- 性能测试的主要内容：
  - 常规性能测试
  - 压力测试
  - 负载测试
  - 可靠性测试
  - 大数据量测试

性能测试举例

- 电信服务器1台能承载1万用户的用户量，那让其为1万用户提供服务即可
  - 常规性能测试
- 让一台服务器为2万用户，3万用户，4万用户，直到将其压到不能运行，如果这个极限是5万，结论是：其最大压力能承担5万用户量
  - 压力测试
- 让1台服务器为4万用户提供服务，查看其能稳定运行多长时间
  - 负载测试
- 让1台服务器为1万用户提供服务，查看其能稳定运行多长时间
  - 可靠性测试

性能测试主要内容

- 常规性能测试
  - 软件在正常的软、硬件环境下运行，不向其施加任何压力的性能测试
- 压力测试
  - 是指持续不断地给被测系统增加压力，直至被测系统被压垮，以确定系统能承受的最大压力
  - 压力测试应注意累积效应问题

性能测试主要内容

- 负载测试：通常是让被测系统在其能忍受的压力极限范围内(或临界状态下)连续运行，来测试系统的稳定性
- 目的是找到系统的处理极限，为系统调优提供依据
- 负载测试侧重于压力持续的时间，压力测试则更加强调施加压力的大小

性能测试主要内容

- **可靠性测试**：是在给被测系统加载一定业务压力的情况下，使系统运行一段时间，以此来测试系统是否稳定
- 通常采用24×7(24小时×7天)的方式来连续运行系统，一般采用平均错误时间间隔(Mean Time Between Failure，MTBF)来衡量被测系统的可靠性。该值越大，系统越稳定

性能测试主要内容

- **大数据量测试：**针对某些系统存储、传输、统计、查询等业务进行大数据量的独立数据量测试
- 与压力测试、负载测试、疲劳测试等并发测试相结合的极限状态下的综合数据量测试

系统测试概述

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

安全性测试概述

- 安全性是指“使得伤害或损害的风险限制在可接受的水平内”
- 安全性测试(Security Testing)用于检验系统对非法侵入的防范能力

黑客攻击系统的动机

- 挑战/成名
  - 为了挑战性的任务或在黑客同行中形成成功者的威望
- 好奇
  - 不满足于挑战，一旦进入，就会查看什么信息有价值

黑客攻击系统的动机

- 使用/借用
  - 分布式计算机比个人计算机能完成更多的事情
  - 使用别人的计算机掩饰自己的痕迹
- 恶意破坏
  - 丑化
  - 破坏
  - 拒绝服务
    - 组织提供服务

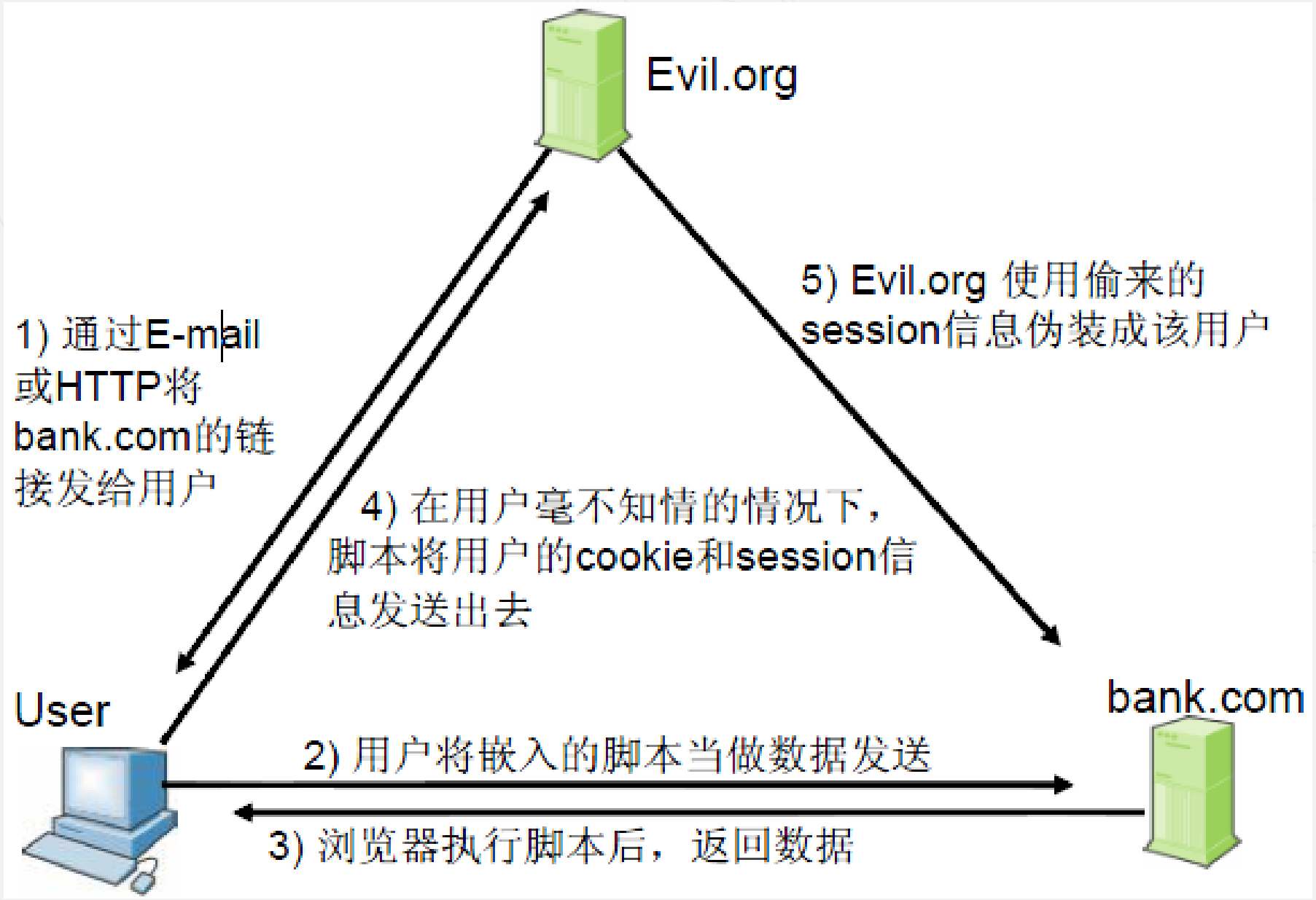
黑客攻击的动机

- 偷窃
  - 找到可以出卖的有价值信息
    - 信用卡号、个人信息、商品和服务等等

常见安全问题举例

- 安全性威胁
  - 跨站脚本攻击
  - 缓冲区溢出
  - SQL注入
  - ……

跨站脚本（XSS,Cross-site Scripting）攻击



跨站脚本（XSS,Cross-site Scripting）攻击

- 是一类专门针对Web应用程序的漏洞
- 使产生漏洞的Web服务器绑定的用户数据（通常保存在cookie中）被泄漏给恶意的第三方
- 所谓“跨站”是指：当一个客户端访问了可正常提供服务，但是有漏洞的Web服务器后，cookie从此客户端传递给了攻击者控制的站点

缓冲区溢出

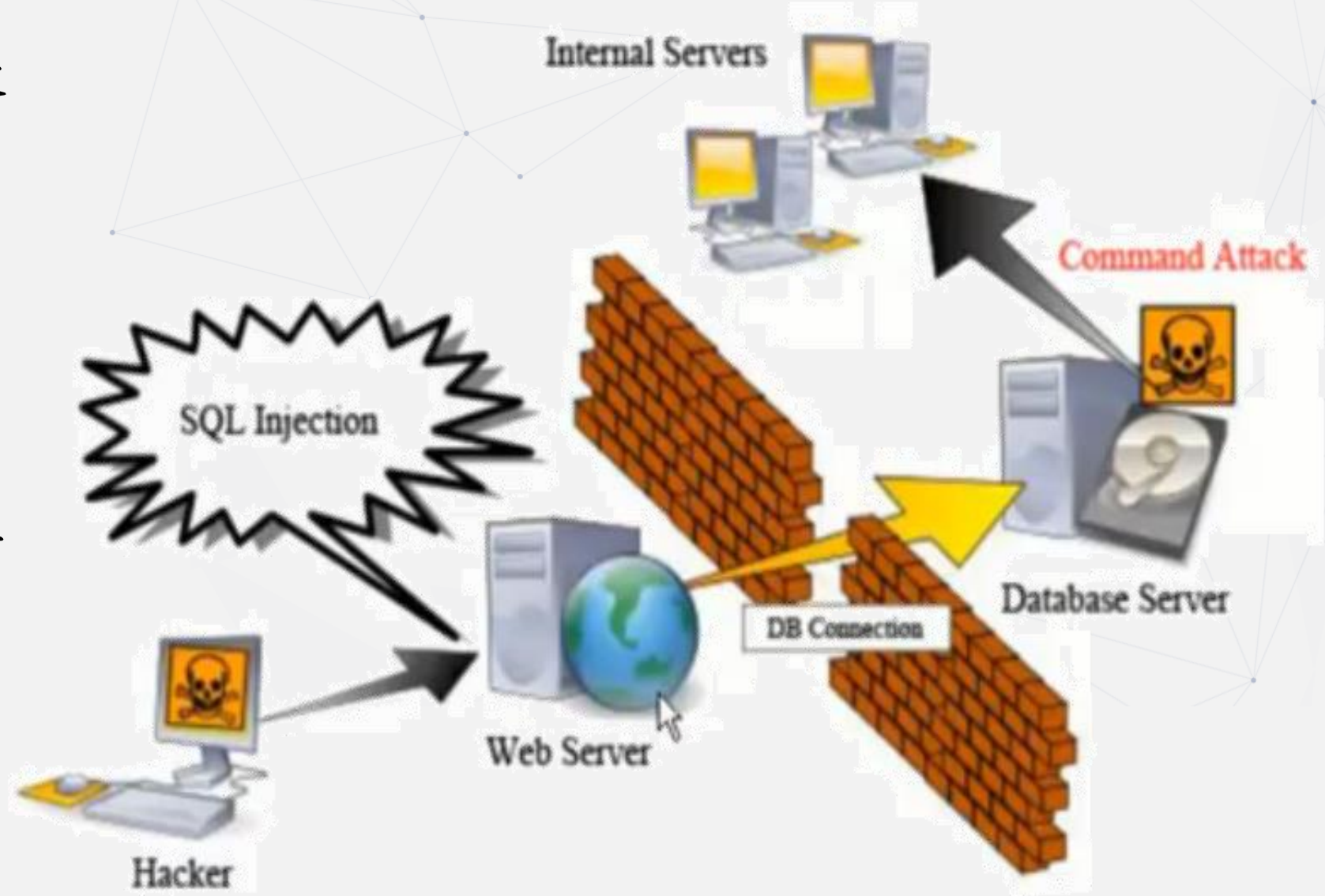
```
1: void myBufferCopy(char * pSourceStr) {
2:         char pDestStr[100];
3:         int nLocalVar1 = 123;
4:         int nLocalVar2 = 456;
5:         strcpy(pDestStr, pSourceStr);
            ...
6: }
7: void myValidate()
8: {
9:     char psourStr[200];
10: /* Assume this function's code validates a user password and grants
access to millions of private customer records /*
11: }
```

SQL注入

- SQL 注入：就是通过把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的SQL命令

SQL注入

- 穿过防火墙，入侵检测系统
- 执行增、删、改、查脚本
- 尝试输入特殊字符



安全性测试方法

- 安全性测试方法：
  - **功能验证**：对涉及安全的软件功能，如权限管理、系统加密和认证等进行测试，验证这些功能是否有效
  - **程序数据扫描**：通过内存测试发现诸如缓冲区溢出之类的漏洞
  - **静态测试**：对源代码进行安全扫描，找出代码中的潜在安全漏洞
  - **动态测试**：以人工或自动化工具模拟黑客对应用系统进行攻击性测试，找出运行时所存在的安全漏洞

安全测试工具

- **IBM Rational AppScan**
- **Jsky**
- **WebPecker**

系统测试

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

兼容性测试

- 定义：兼容性测试是指测试软件在特定的硬件平台上、不同的应用软件之间、不同的操作系统平台上、不同的网络等环境中是否能很好地运行的测试

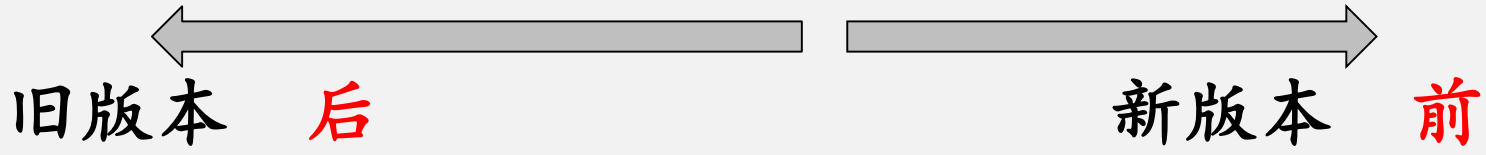
- 例如：
- 安装Vista系统是微软花费人力、物力和财力时间最多的产品
  - 使用半年仍然不能为用户接受
  - Nero刻录软件，瑞星等许多主流软件不能安装、使用

兼容性测试

- 为什么进行兼容性测试
  - 通用软件越来越多，保证这些通用软件支持跨硬件、跨平台、跨软件运行

兼容性测试标准

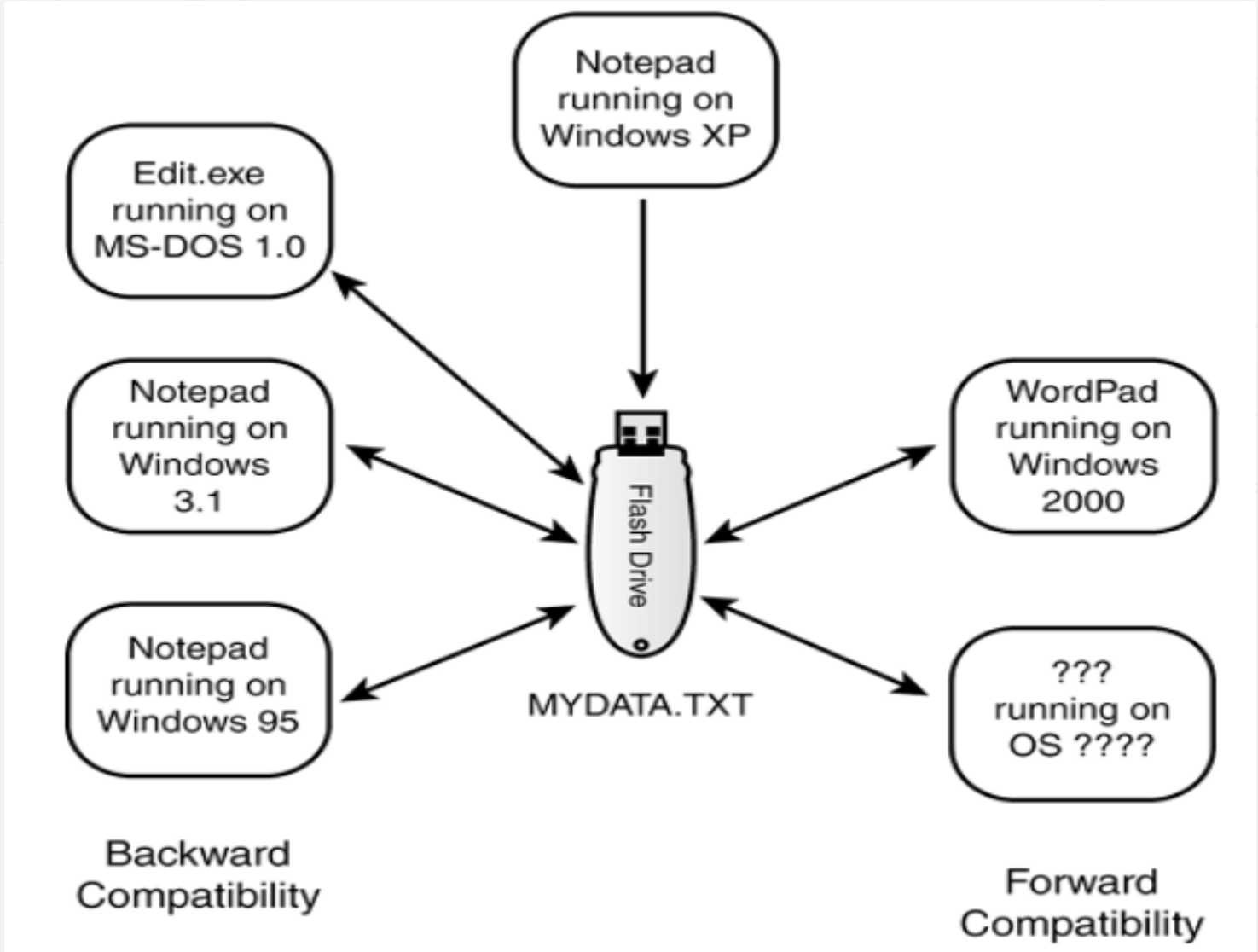
- 向后兼容
  - 可以使用软件的以前版本
  - 如：Office 2007/2013 可以打开Office 2003下存储的文档
- 向前兼容
  - 可以使用软件的以后版本
  - 如：Office 2003可以打开Office2013存储的文档



兼容性测试举例

- 向前、向后兼容举例
  - **Windows Notepad**

    文本编辑器



标准和规范

- 高级标准和规范
  - 遵守平台（操作系统）的标准
  - 例如：Windows认证徽标，为了得到徽标，软件必须通过由独立测试实验室执行的兼容性测试
    - 支持三建以上的鼠标
    - 支持在C：和D：以外的磁盘上安装
    - 文件名长度
    - ……400多页标准

- 低级标准

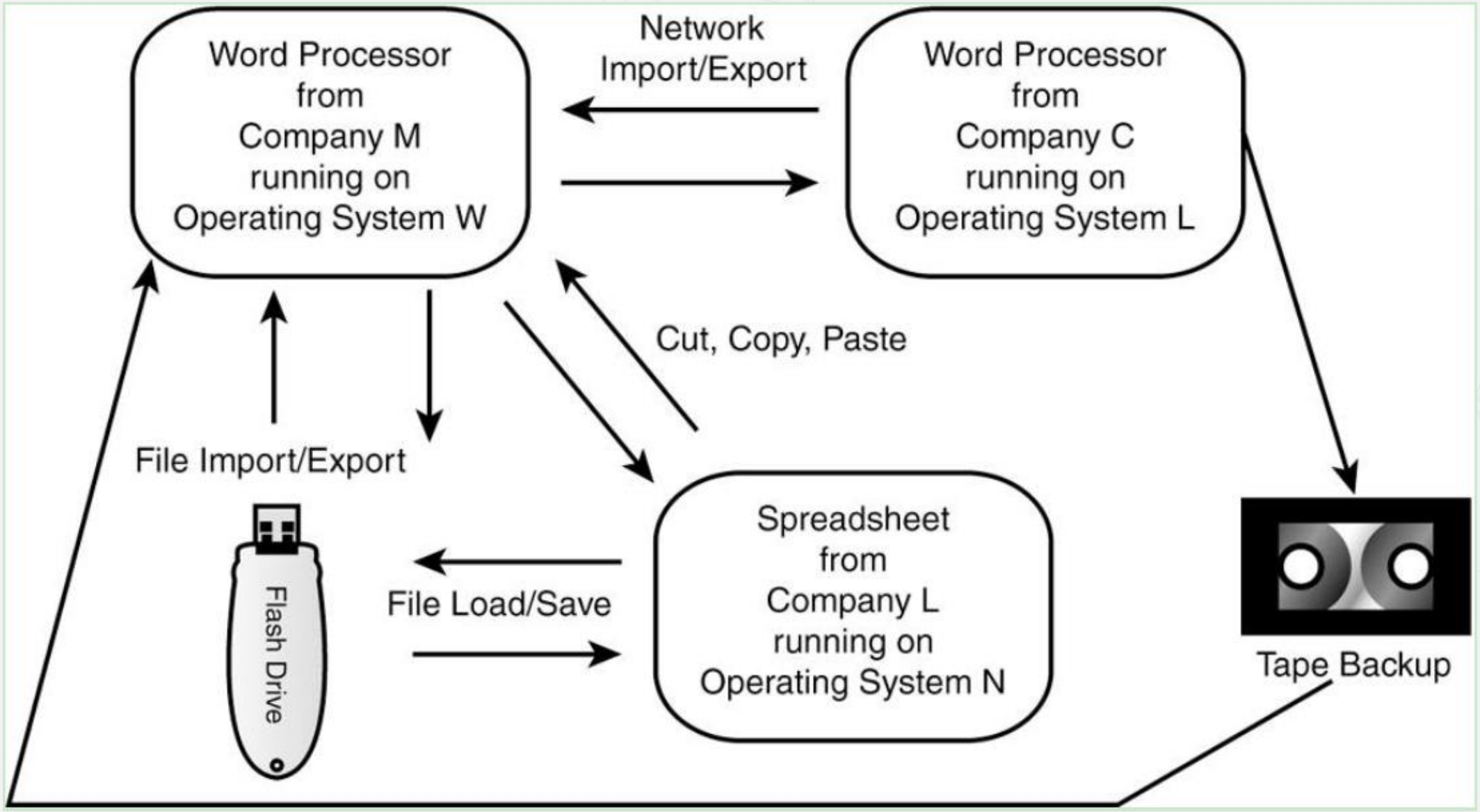
标准和规范

- 低级标准和规范
  - 产品本身的标准
  - 例如：图形软件，需支持保存jpg,bmp,gif……等格式，如果不符合这些格式的标准，则无法在其他程序查看该文件

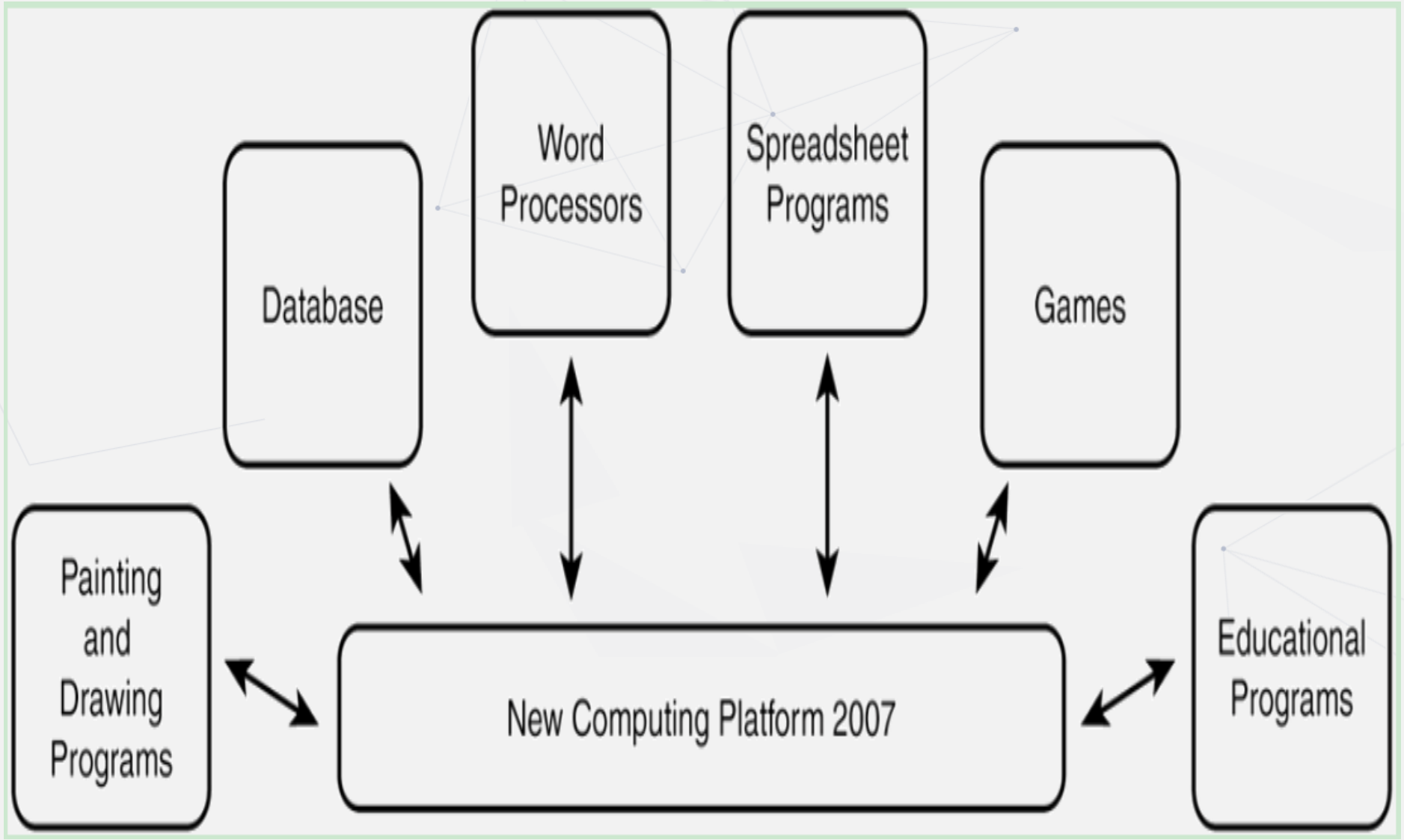
兼容性测试包括哪些方面

- 操作系统
- 应用软件之间
  - 如：不同浏览器之间
- 数据库兼容
- 软硬件配合兼容

兼容性测试适用场景


Word Processor from Company M running on Operating System W
Network Import/Export
Word Processor from Company C running on Operating System L
Cut, Copy, Paste
File Import/Export
Flash Drive
File Load/Save
Spreadsheet from Company L running on Operating System N
Tape Backup

兼容性测试举例—新平台兼容性测试



平台兼容性应用软件选择

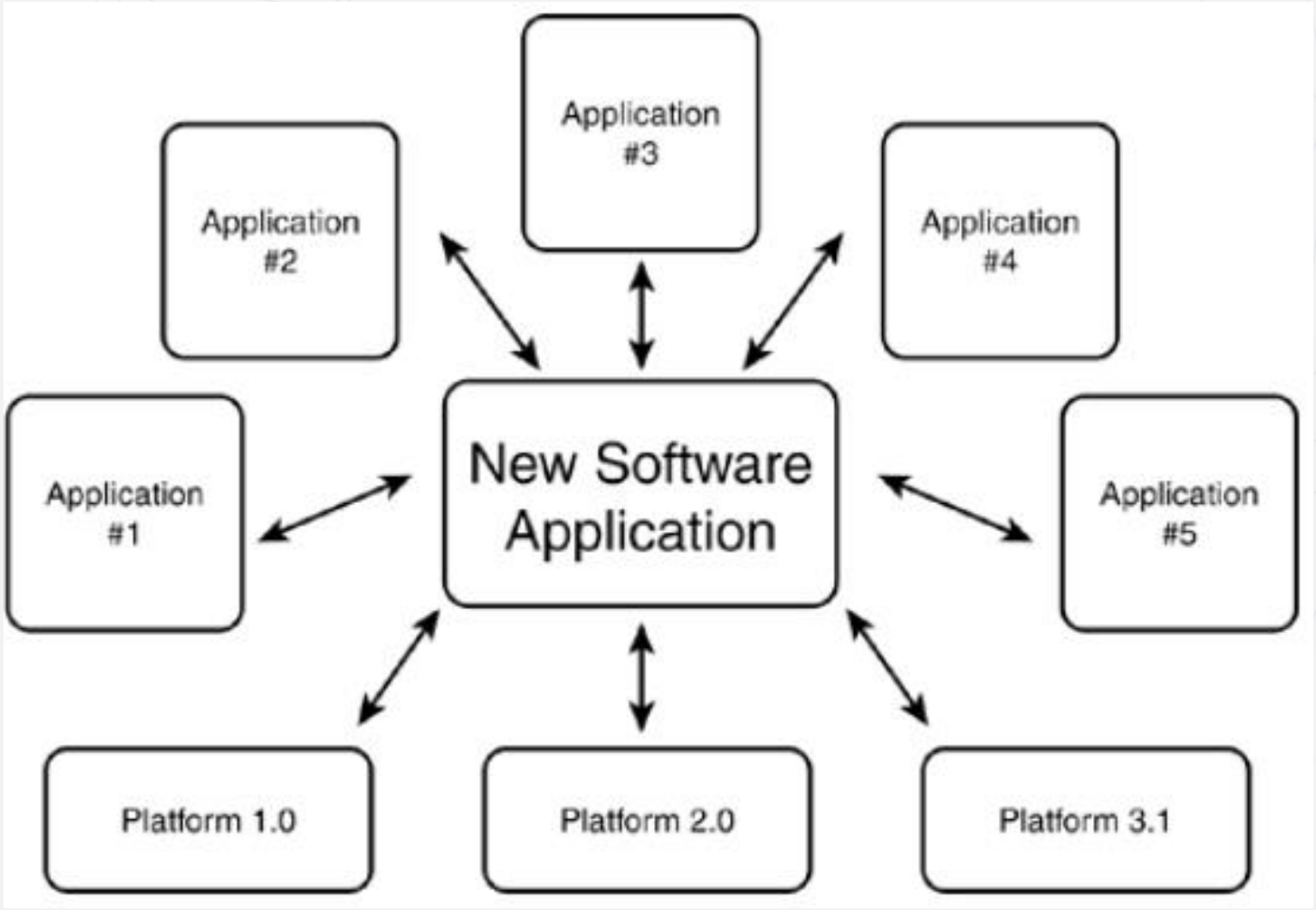
- 利用等价类划分，使验证软件之间正确交互的最小有效集合
  - 流行程度
    - 利用销售记录选择前100或1000个最流行的程序
  - 年头
    - 选择近3年以内的程序和版本

平台兼容性应用软件选择

- 类型
  - 把软件分为文字编辑、图形、音频、视频、财务、数据库、财务等等
- 生产厂商
  - 根据制作软件的公司来选择软件

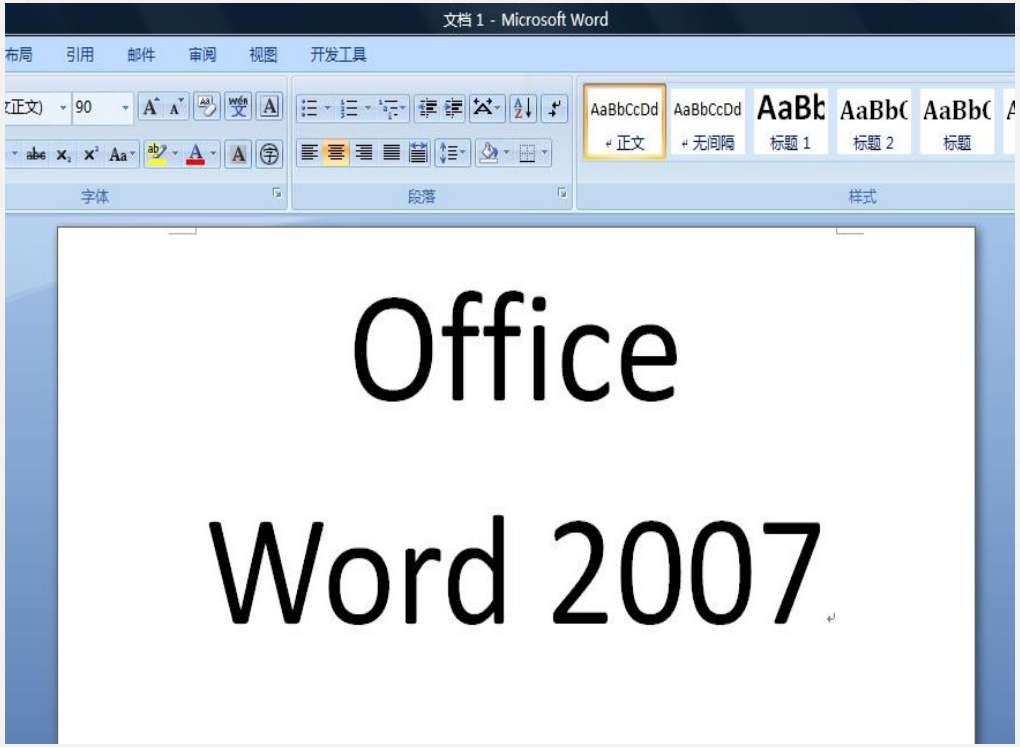
新应用软件兼容性测试

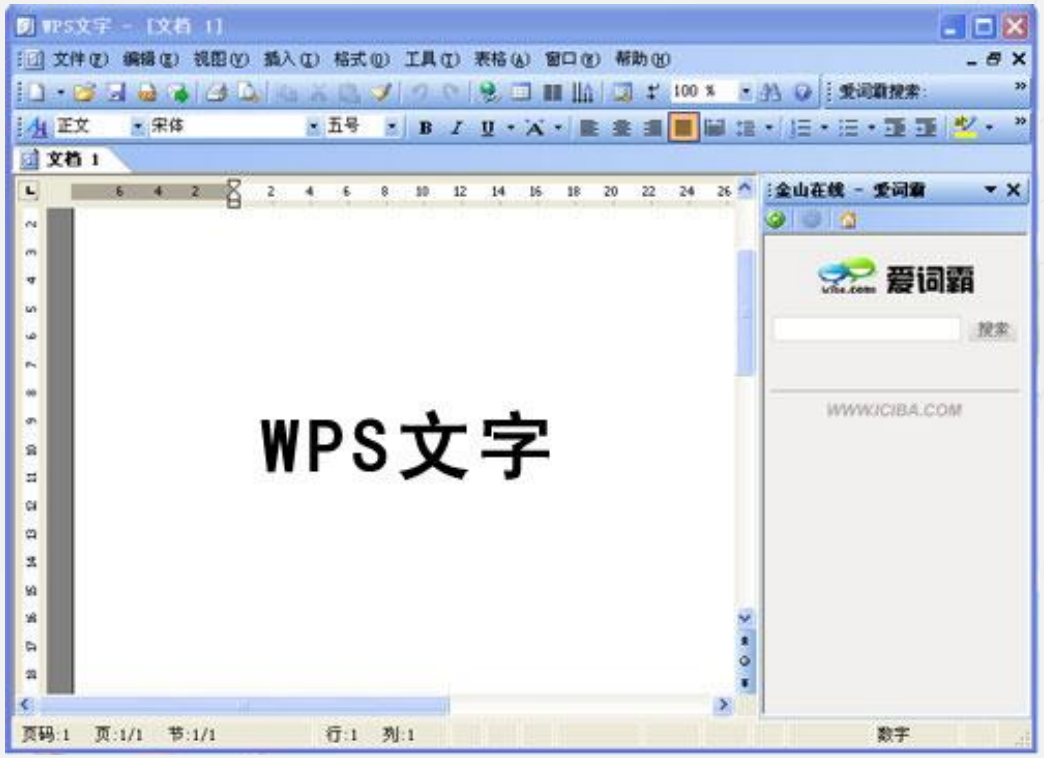
- 决定在哪些平台上测试软件
- 和什么应用程序一起测试



兼容性测试—数据共享兼容性

- 数据兼容性指要在应用程序之间共享数据，它要求支持并遵守公开的标准，允许用户与其他软件无障碍的传输数据。



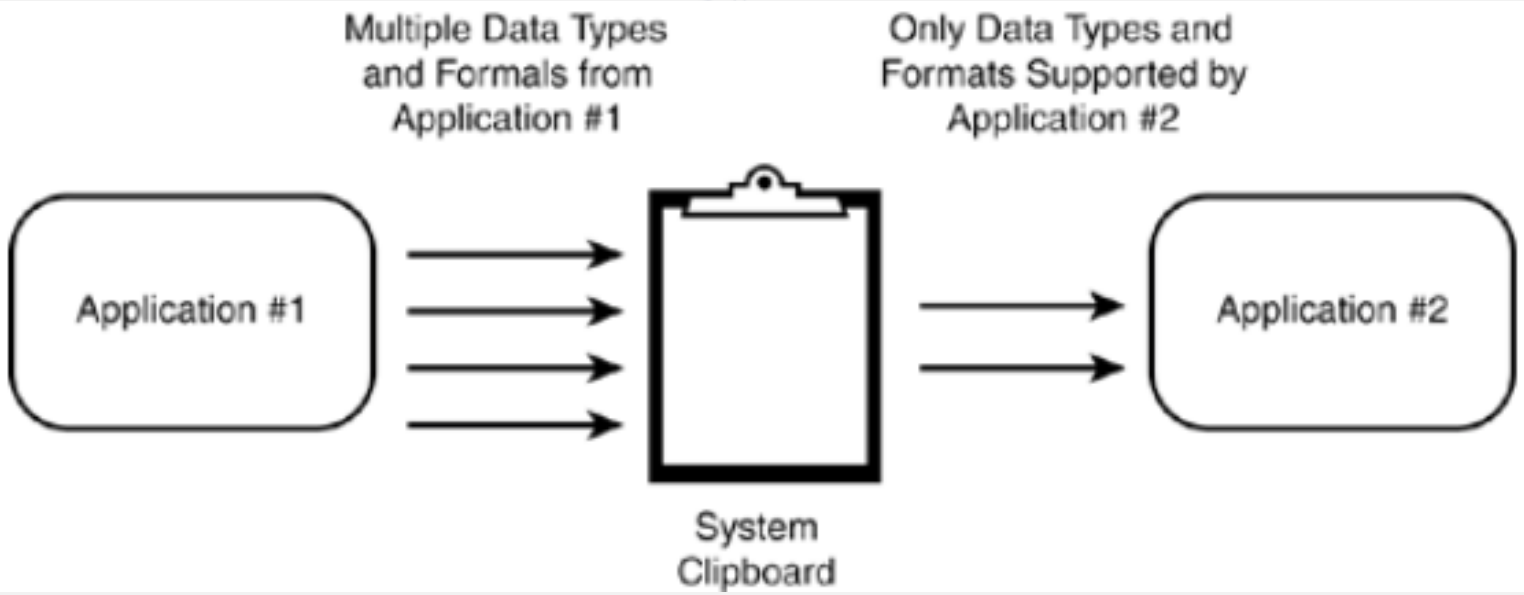


数据共享兼容性

- 文件保存和读取，只有符合标准才能在两台计算机上保持兼容
- 文件导出和导入是许多程序与自身以前版本、其他程序保持兼容的方式

数据共享兼容性

- 数据被复制、剪切，当进行粘贴时，一些应用程序只接受特定数据类型和格式



软硬件配合兼容

- 新操作系统对于主流硬件型号上进行兼容性测试
- 新应用软件对于主流硬件型号及主流操作系统的兼容性测试
- B/S 架构的软件，必须考虑浏览器的兼容性
- 移动端的软件，必须考虑主流移动设备厂商的设备，主流平台（Android,IOS等），主流屏幕大小等等

系统测试概述

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

用户界面测试

- 什么是用户界面（UI）
  - 用于与软件程序交互的方式，提供用户输入和系统输出称为用户界面
    - 网站中的页面
    - 电视机的遥控器
    - 抽油烟机按键
    - 电梯按键
    - 早期计算机界面：触发开关和发光二极管
    - 50-60年代：纸带、穿孔机
    - DOS窗口
    - 图形界面
- 讨论界面测试重要吗？
  - 界面测试是用户使用的第一或第二印象，所以非常重要

用户界面测试包含

- Ron Patton给出了有些用户界面的基本构成标准：
  - 规范化
  - 灵活性
  - 正确性
  - 直观性
  - ……

规范化

- **规范化**：经过大量测试，总结出的方便用户的规则
  - 第一次使用系统，应有“关于系统”的介绍，让用户认识系统
  - 应有代表应用程序的正确图标
  - 所有屏幕、对话框应有与内容相对应的正确图标
  - 可在Windows 的任务条和状态条中显示应用程序（C/S）
  - 注意数据显示的规范性，如数据精度、时间及日期显示格式

灵活性

- 灵活性是对于熟练用户而言，如变换皮肤、变换界面字体等。但灵活性与稳定性往往相互矛盾

正确性

- 界面显示内容的准确性

- 操作处理的正确性
  - 例：某系统提示信息：

    Error:Keyboard not found.Press F1 to continue

直观性

- 直观性反映用户学习掌握该软件所耗费的时间及在具体业务流程上的简化，即：

1）易见：用户凭直觉观察就知道设备的状态，以及提供可采取的行动

2）易学：用户不查阅帮助文档，就能对一个陌生的产品有清晰的认识

3）易用：用户不查阅帮助文档，就能使用软件

舒适性

- 舒适性是个模糊的概念，典型的舒适性指标包括：

1）内容的友好性

2）提示信息的指导性

3）界面美观协调（注意：颜色搭配，少用大红、大绿等深色）

4）菜单快捷键中使用快捷方式与用户习惯保持一致

其他

- 实用性
- 一致性
- 帮助文档
- 独特性

系统测试概述

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

易用性测试

- 为什么进行易用性测试
  - 在缺陷定义的第5条规则规定：软件难以理解，不易使用，运行缓慢或者—从测试员的角度看——最终用户认为不好

易用性测试


外部和内部质量
功能性
可靠性
易用性
效率
维护性
可移植性
适合性
准确性
互操作性
保密安全性
功能性的
依从性
成熟性
容错性
易恢复性
可靠性的
依从性
易理解性
易学性
易操作性
吸引性
易用性的
依从性
时间特性
资源利用性
效率依从性
易分析性
易改变性
稳定性
易测试性
维护性的
依从性
适应性
易安装性
共存性
易替换性
可移植性的
依从性

易用性测试

- 易理解性
  - 软件产品使用户能理解软件是否合适以及如何能将软件用于特定的任务和使用条件的能力。 注:这要依赖于软件提供的文档和初始印象
- 易学性
  - 软件产品使用户能学习其应用的能力

易用性测试

- 易操作性
  - 软件产品使用户能操作和控制它的能力
- 吸引性
  - 软件产品吸引用户的能力 。 注:这涉及到软件旨在使自身对用户更具吸引力的属性，例如颜色的使用和图形化设计的特征。
- 易用性的依从性
  - 软件产品遵循与易用性相关的标准、约定、风格指南或法规的能力

辅助选项测试

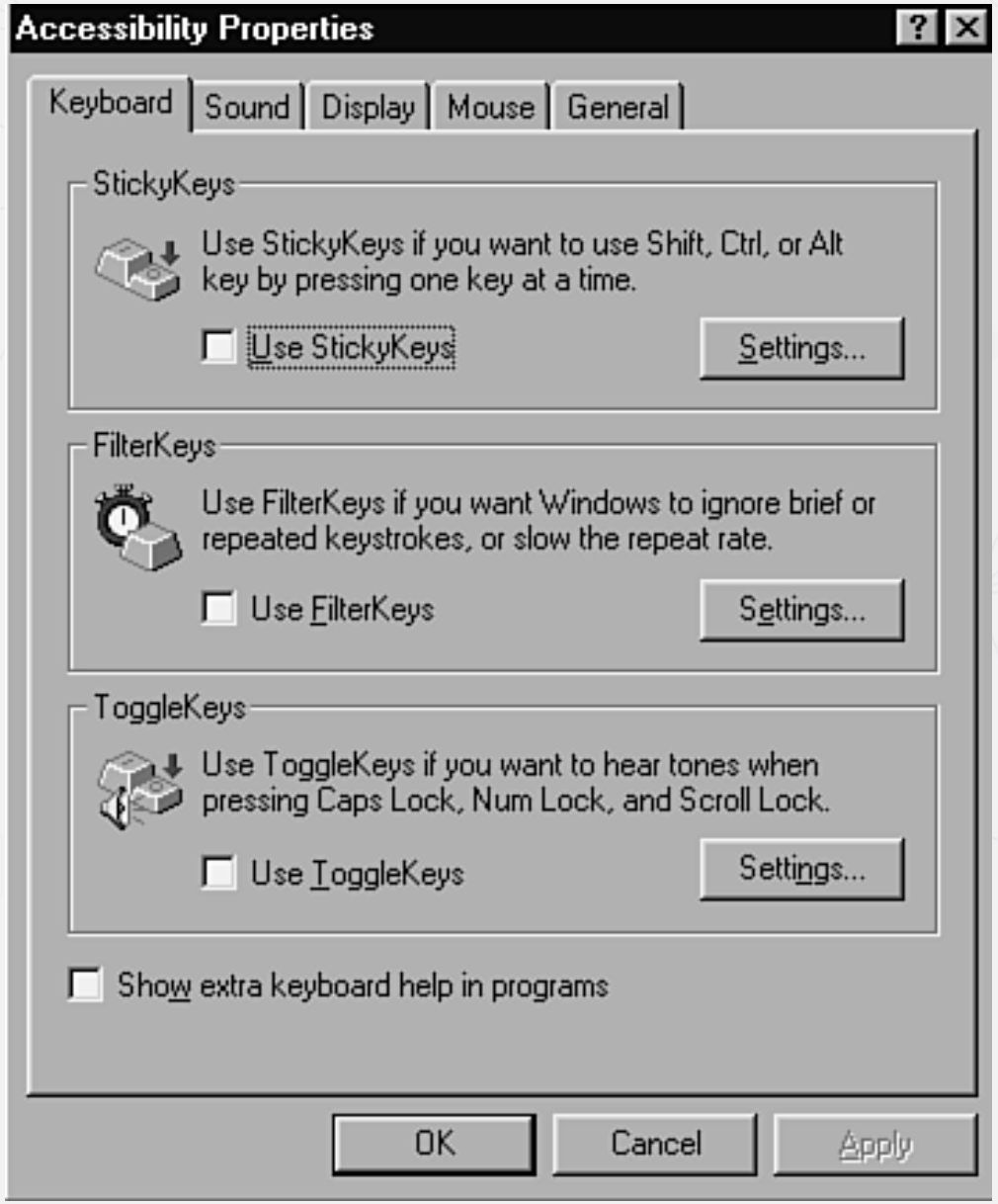
- 易用性测试中的一个严肃主题是辅助选项测试，也就是为有残疾障碍的人测试
  - 视力损伤
  - 听力损伤
  - 运动损伤
  - 认知和语言障碍

软件中的辅助特性

- 软件可以有两种方式提供辅助
  - 操作系统内置
  - 平台
- 软件本身需要遵守启用辅助选项
  - 遵守与键盘、鼠标、声卡和显示器通信的平台标准
  - 如果被测软件本身就是平台，就需要定义、编制和测试自己的辅助选项

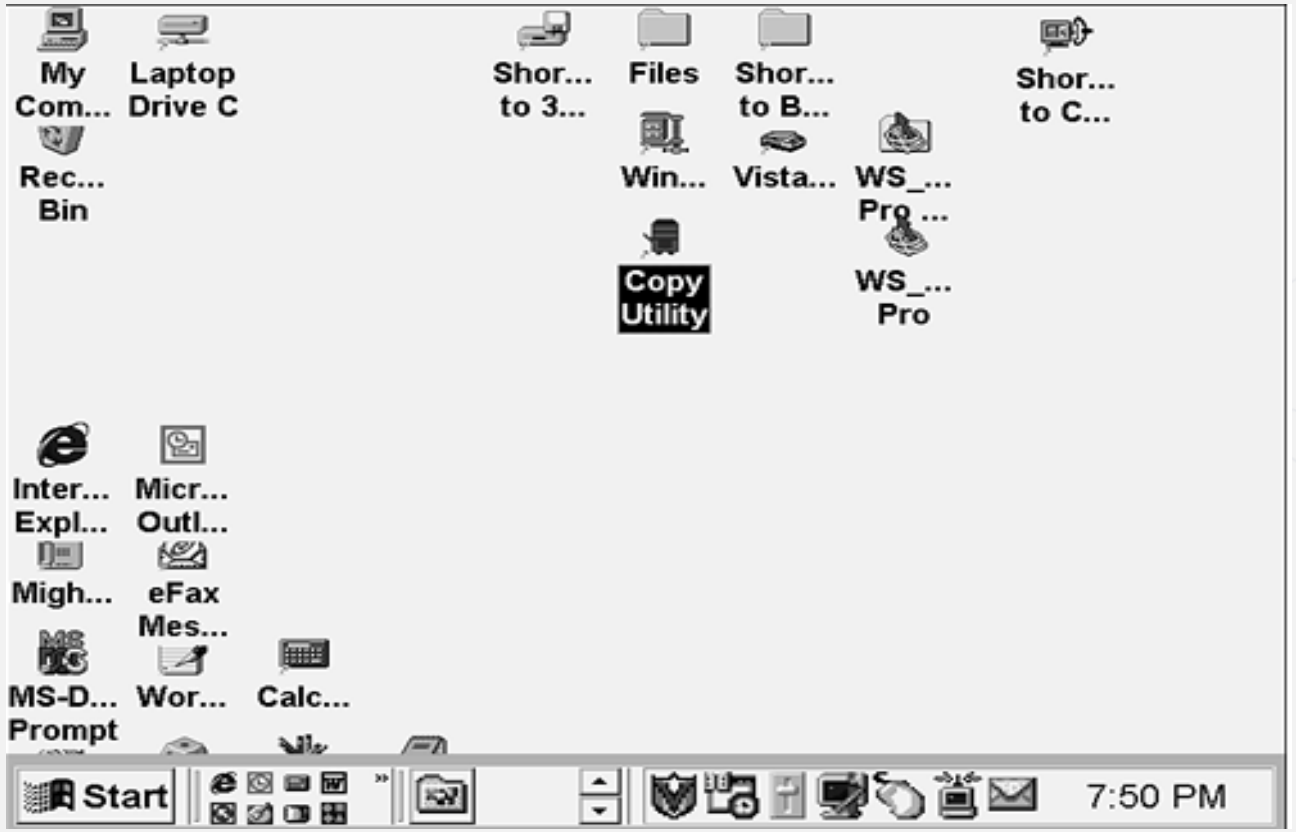
Windows 提供的辅助选项

- 粘贴键
- 筛选键
- 切换键
- 声音卫士（每当系统发出声音时，给出警告）
- 声音显示（让程序显示其声音或讲话的标题，需要在软件中编制）



Windows 提供的辅助选项

- 高对比度（利于实例损伤者阅读而设计的颜色和字体）
- 鼠标键：允许用键盘来代替鼠标操作
- 串行键：设置通信接口读取来自外部非键盘设备的击键



内置辅助选项详情

- **www.microsoft.com/enable**
- **www.apple.com/accessibility**
- **www.linux.org/docs/ldp/howto/Accessibility-HOWTO**

系统测试概述

- 系统测试包含：
  - 功能测试
  - 性能测试
  - 安全性测试
  - 兼容性测试
  - 界面测试
  - 易用性测试
  - 安装测试

安装测试(Installation Testing)概述

- 定义：是指广义的安装测试，包括安装和卸载
- 安装前的测试重点：
  - 是否需要专业人员安装
  - 确认打包程序的特性，确认对安装环境是否有限制和要求，不同的打包发布程序支持的系统不一样，且至少应在标准配置和最低配置条件下进行安装测试

安装测试

- 安装过程中的测试重点
- 正常安装应注意：
  - 安装过程与安装手册中描述的所有步骤保持一致，包括所有界面、提示信息等内容
  - 安装过程应符合一般的安装流程，否则应关注哪些步骤被省去，是否对应有默认设置，是否满足大部分用户的意愿或硬件配置
  - 测试安装过程中的所有默认和典型选项

安装测试

- 安装过程中的测试重点
- 正常安装应注意（续）
  - 测试各种安装组合(包括参数、控件执行顺序、产品组件、产品组件安装顺序等的组合)
  - 安装过程是否简单，容易掌握
  - 安装过程中应有明显、合理的操作提示
  - 应验证软件使用许可证号或注册码
  - 应能识别大部分硬件

安装测试

- 安装过程中的测试重点
- 安装中的异常应注意
  - 测试安装空间不足的情况
  - 测试异常配置或状态(非法和不合理配置)，如断电、数据库终止、断网等
  - 安装过程中应允许终止，终止安装后应能确保系统恢复原状。安装软件不应破坏系统原有的系统文件，否则一旦停止安装将造成原有系统无法正常使用

安装测试

- 安装后的测试重点
  - 能否产生正确的目录结构和文件，文件属性是否正确
  - 动态链接库是否正确
  - 软件能否正确运行
  - 是否产生多余的目录结构、文件、注册表信息、快捷方式等

安装测试

- 安装后的测试重点（续）
  - 安装后系统是否对其他应用程序造成不正常影响(如操作系统、应用软件等)
  - Web服务是否有冲突
  - 系统升级后原有应用程序能否正常运行
  - 软件卸载后所有占用的资源、文件、目录、快捷方式等内容都应予以清除，且不应影响到基础的系统文件，不影响系统应保留的用户数据及其他软件的使用

目录

系统测试总结

- 系统测试是整个测试环节中比较重要的环节，涉及的测试面较广
- 系统测试包含：功能、性能、安全、兼容、界面、易用、安装等方面的测试
- 根据不同的被测系统，选择的侧重点不同

内容总结

- 系统测试概述
- 系统测试内容
- 系统测试总结

Question