复习-人机交互

人机交互发展历史

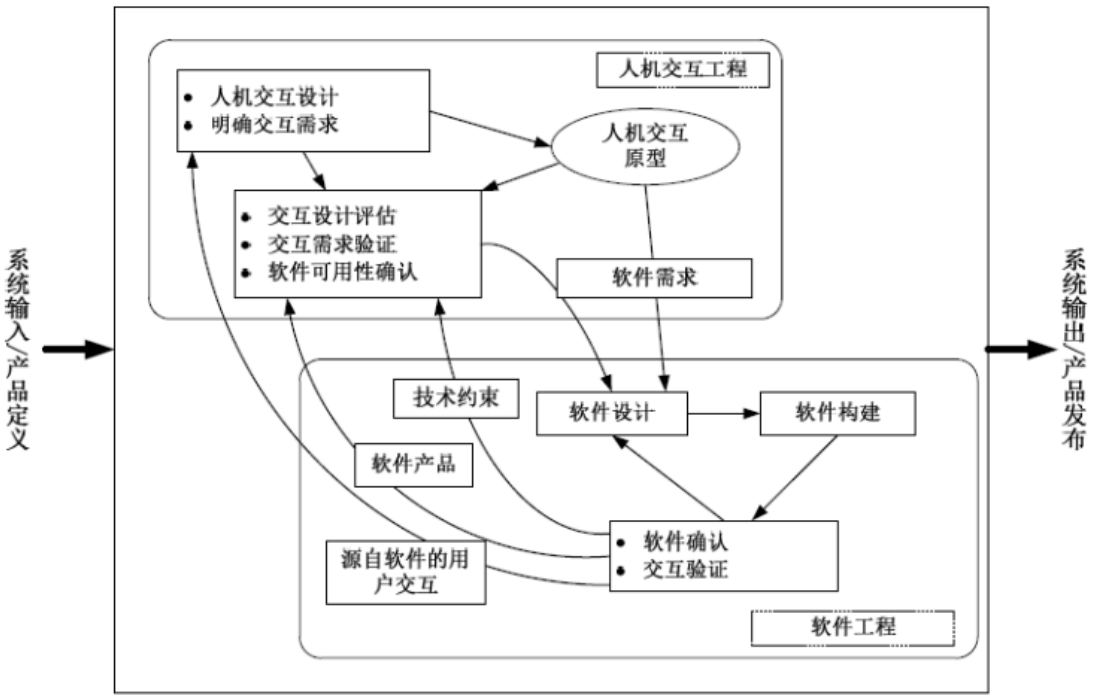
- 新的界面变革包含了上一代界面：作为一种特例
- 旧的交互方式仍有其存在的必要性：以前的用户从未消失
- 学习目的：利用原有技术实现新的交互手段

发展历史

- 批处理
  - 每次只能由一个用户对计算机进行操作
  - 使用以 01 串表示的机器语言
- 终端（命令行）
  - 1950s
- 图形用户界面
  - 1962 - Sketchpad：第一个交互式绘图系统
  - 1964 - 鼠标
  - WIMP
    - 用户可在窗口内选取任意交互位置
    - 不同窗口之间可以叠加（二维半）
- 未来的人机交互
  - 多媒体界面：引入动画、音视频等动态媒体
  - 多通道交互技术：具有并行性，可同时接受来自多个通道的信息
  - 虚拟现实
  - 脑机交互

人机交互与软件工程

- HCI 对 SE 的促进作用
  - SE 没有提出明确的对用户界面及可用性需求进行描述的方法
  - SE 不能够在系统开发过程进行中对用户界面进行终端测试
  - SE 产品的可用性、有效性及满意度并不高
- 二者在工程中的关系



人机交互基础知识

交互泛型

命令行交互

- 基于字符的界面：用户通过在屏幕某个位置上键入特定命令的方式来执行任务
- 优点
  - 专家用户能够快速完成任务
  - 可动态配置可操作选项
  - 支持用户自定义命令
  - 键盘操作较鼠标操作更加精确
  - 较 GUI 更节约系统资源
- 缺点
  - 基于记忆：命令语言对用户的记忆能力要求较高
  - 键盘操作，出错频率较高
  - 要求用户记忆指令的表示方式：与可用性理论所强调的 “不应要求用户了解计算机底层的实现细节” 相违背

菜单驱动界面

- 以一组层次化菜单的方式提供用户可用的功能选项；通过鼠标，数字/字母/方向键进行选择
- 优点
  - 基于识别，对记忆的需求较低
  - 具有自解释性
  - 容易纠错
  - 适合新手用户（若提供了较好的快捷键功能，则同样适合专家用户）
- 缺点
  - 导航方式不够灵活
  - 当菜单规模较大时，导航效率不高
  - 占用屏幕空间，不适合小型显示设备（节省空间：下拉菜单，弹出式菜单）

    - 对专家用户而言效率不高

基于表格的界面

- 显示给用户一个表格
- 优点
    - 简化数据输入
    - 只需识别无需学习
    - 适合日常文书处理等需要键入大量数据的工作
- 缺点
    - 占用大量屏幕空间
    - 导致业务流程较形式化

直接操纵

- 用户通过在可视化对象上面进行某些操作来达到执行任务的目的
    - 是 图形用户界面 的主要特征
- 三个阶段
    - 自由阶段：用户动作执行前的屏幕显示情况
    - 捕获阶段：用户动作执行过程中的屏幕显示情况
    - 终止阶段：用户动作执行后的屏幕显示情况
- 优点
    - 将任务的概念可视化，用户可以非常方便地识别他们
    - 基于识别，对记忆的要求不高，可减少错误发生
    - 容易学习，适合新手用户
    - 支持空间线索，鼓励用户对界面进行探索
    - 可实现对用户操作的快速反馈，具有较高的用户主观满意度
- 缺点
    - 实现较难
    - 对专家用户而言效率不高
    - 对图形显示性能的需求较高
    - 不适合小屏幕显示设备（目前看来这个问题应该已经解决）

其它

- 问答界面
    - 通过询问用户一系列问题实现人机交互
    - 典例：Web 问卷
    - 优点
        - 对记忆要求较低
        - 每个界面都具有自解释性
        - 将任务流程以简单的线性表示
        - 适合新手用户
    - 缺点
        - 要求从用户端获得有效输入
        - 要求用户熟悉界面控制
        - 纠错过程比较乏味
- 隐喻界面

    - 在用户已有的知识基础上建立一组新的知识，实现界面视觉提示和系统功能之间的知觉联系，进而帮助用户从新手用户转变为专家用户
    - 典例：回收站（有垃圾的图标表示有文件，无垃圾表示无文件）
    - 优点
        - 直观生动，无需学习
    - 缺点
        - 不具有可扩展性
        - 不同用户对同一事物的联想可能不同
        - 将理念和物理世界束缚在一起
        - 寻找恰当的隐喻可能存在困难

信息处理模型

- 研究人对外界信息的接收、存储、集成、检索和使用，可推算人需要多长时间来感知和响应某个刺激（反应时间）

人类处理机模型

- 1983 Card 等
- 最著名的信息处理模型
- 三个组件
    - 感知处理器：信息将被输出到声音和视觉存储区域
    - 认知处理器：输入将被输出到工作记忆
    - 动作处理器：执行动作
- 缺点
    - 把认知过程简单描述为一系列处理步骤
    - 仅关注单个人和单个任务的执行过程
    - 忽视了环境和其他人可能带来的影响

格式塔心理学

- 格式塔 => Gestalt => 完形
- 研究人是如何感知一个良好组织的模式的，而不是将其视为一系列相互独立的部分
- 用户在感知事物的时候总是尽可能将其视为一个“好”的型式
- 相近性原则
    - 空间上靠近的物体容易被视为整体
    - 设计界面时，应按照相关性对组件进行分组
- 相似性原则
    - 看上去相似的物体容易被视为整体
    - 功能相近的组件应该使用相同或相近的表现形式
- 连续性原则
    - 共线或具有相同方向的物体会被组合在一起
    - 将组件对齐，更有助于增强用户的主观感知效果
    - 典例：文本段落的对齐
- 完整性和闭合性原则
    - 人们倾向于忽视轮廓的间隙而将其视作一个完整的整体
    - 页面上的空白可帮助实现分组
- 对称性原则

    - 相互对称且能够组合为有意义单元的物体会被组合在一起
- 格式塔反例
    - 对比：让重点更突出

人脑的记忆结构

- 感觉记忆
    - 又称瞬时记忆，在人脑中持续约 1s
    - 帮助我们把相继出现的一组图片组合成一个连续的图像序列，产生动态的影像信息
- 短时记忆
    - 感觉记忆经编码后形成
    - 又称工作记忆，约保持 30s
    - 储存的是当前正在使用的信息，是信息加工系统的核心（可理解为计算机的内存）
    - 存储能力约为 7±2 个信息单元、
        - 那么我们需要保证菜单中最多只有 7 个选项吗?
        - 答：不需要。因为菜单和工具栏基于识别
        - 启发：界面设计时要可能减小对用户的记忆需求，可以通过将信息放置于一定的上下文中来减少信息单元的数目
- 长时记忆
    - 短时记忆中的信息经进一步加工后会变为长时记忆
    - 只有与长时记忆区的信息具有某种联系的新信息才能够进入长时记忆
    - 信息容量几乎是无限的
    - 启发
        - 注意使用线索来引导用户完成特定任务
        - 在追求独特的创新设计时也应注重结合优秀的交互范型
    - 遗忘
        - 长时记忆中的信息有时无法提取，但这不代表信息丢失了
    - 易出错
        - 从表面上看是由于用户的误解、误操作或一时大意，但大部分交互问题都源于系统设计本身
- 视觉错觉
    - 启示：对物体的视觉感知与物体所处的上下文密切相关

交互设计目标与原则

可用性目标

- 易学性
    - 系统应当容易学习，从而用户可以在较短时间内应用系统来完成某些任务
    - **最基本** 的可用性属性
- 高效率
    - 在用户学会使用系统后，应该具有更高的生产力水平（效率）
- 易记性
    - 在用户学会使用系统后，应当能迅速回想起它的使用方法
    - 启发
        - 位置：将特定对象放在固定位置

        - 分组：对事物按照逻辑进行恰当的分组
        - 惯例：尽可能使用通用的对象或符号
        - 冗余：使用多个感知通道对信息进行编码
- 少出错
    - 保证导致灾难性后果错误的发生频率降到最低
    - 保证错误发生后迅速恢复到正常状态
- 主观满意度
    - 用户对系统的主观喜爱程度（某些情况下比效率更重要，如游戏等非工作环境）
    - 观念的转变：传统软件重视效率和可靠性 => 用户体验

可用性度量

> 区分用户是否 “有经验”：通过使用系统的小时数

- 易学性
    - 找一些从未使用过系统的用户，统计他们学习使用系统直至达到 特定熟练程度 的时间
        - 用户：能够代表系统的目标用户
        - 特定熟练程度：能在特定的时间内完成特定的任务
- 使用效率
    - 为用户绘制学习曲线，当发现用户的效率水平在一段时间内不再提高时，认为其稳定
- 易记性
    - 对非频繁使用用户进行测试最能体现系统的易记性
    - 对在特定长时间内没有使用系统的用户进行标准用户测试，记录下这些用户执行特定任务所用的时间
- 错误率
    - 在用户执行特定任务时统计错误操作的次数
- 主观满意度
    - 为减少单个用户评价的主观性，把多个用户的结果综合起来取其平均值
    - 用 1-5 或 1-7 分的调查问卷对系统进行打分

可用性技术

- 用户和任务观察
    - 了解产品的目标用户是可用性工程的第一个步骤
    - 要与潜在用户进行直接接触
    - “你” 不是用户！！！
- 场景
    - 原型：通过省略整个系统的若干部分来减少实现的复杂性
    - 水平原型：减少功能深度
    - 垂直原型：减少功能数量
- 边做边说法
    - **最有价值的单个可用性工程方法**
    - 让真实用户在使用系统执行特定任务的时候，讲出他们的所思所想
    - 实验人员需要不断地提示用户，或请他们事先观摩
- 启发式评估
    - 剩余的问题可以通过简化的边做边说法来发现
    - 为避免个人的偏见，应当让多个不同的人来进行评估
    - 问题：需要多少人参与?

    - 回答：五名专家
        - 已经可以发现 80% 问题，更多的专家不能发现很多问题

设计规则

- 十项启发式规则（Jacob Nielson）
    - 系统状态的可见度（下载时的进度条）
    - 系统和现实世界的吻合（不要给用户展示代码的异常错误，而要对应到现实世界的错误）
    - 用户享有控制权和自主权（撤销&恢复；导航栏&面包屑）
    - 一致性和标准化（图标不能产生歧义）
    - 避免出错（用日历而非让用户键入日期字符串）
    - 依赖识别而非记忆
    - 使用的灵活性和高效性（菜单 + 快捷键）
    - 审美感的最小化设计
    - 帮助用户识别、诊断和恢复错误（给出用户纠正错误的建议）
    - 帮助和文档

交互需求

用户类别

- 新手用户
    - 特点
        - 敏感，容易在开始有挫折感
    - 设计要求
        - 不能将新手状态视为目标
        - 让学习过程快速且有针对性
        - 确保程序充分反映用户关于任务的心智模型
        - 具有向导功能的对话框帮助
        - 菜单项应该是解释性的
- 专家用户
    - 特点
        - 对缺少经验的用户有很大的影响力
        - 欣赏更新的、更强大的功能
        - 不会受到复杂性增加的干扰
    - 设计要求
        - 对经常使用的工具集，要能快速访问
- 中间用户
    - 特点
        - 需要工具
        - 知道如何使用参考资料
        - 能够区分经常使用和很少使用的功能
        - 高级功能的存在让永久的中间用户放心
    - 设计要求
        - Tooltip 是中间用户最好的习惯用法
        - 在线帮助是永久中间用户的极佳工具
        - 常用功能中的工具放在用户界面的前端和中心位置
        - 提供一些额外的高级特性

用户建模

- 人物角色
  - 并非真实的人，但在整个设计过程中代表真实的人
  - 基于观察真是人的行为和动机
  - 在人口统计学调查收集到的实际用户行为数据基础上形成的综合原型
- 构造人物角色
  - 同一个用户可以扮演系统的不同角色
  - 基于以下问题
    - 谁将使用系统？他们属于哪类人群？
    - 他们与软件的关系有什么特征？
    - 他们通常需要软件提供什么支持？
    - 他们会对软件有什么行为？
    - 他们对软件的行为有什么期望？
- 日常最低要求演示者
  - 经常使用；简单使用；快速、方便操作
- 建模过程
  - 拼凑：头脑风暴，产生一些模型片段
  - 组织：将这些片段按需分组，归并删冗
  - 细节：建立和完善响应描述，补充遗漏数据
  - 求精：对模型进行推敲，以便改进完善
  - 以上过程反复循环

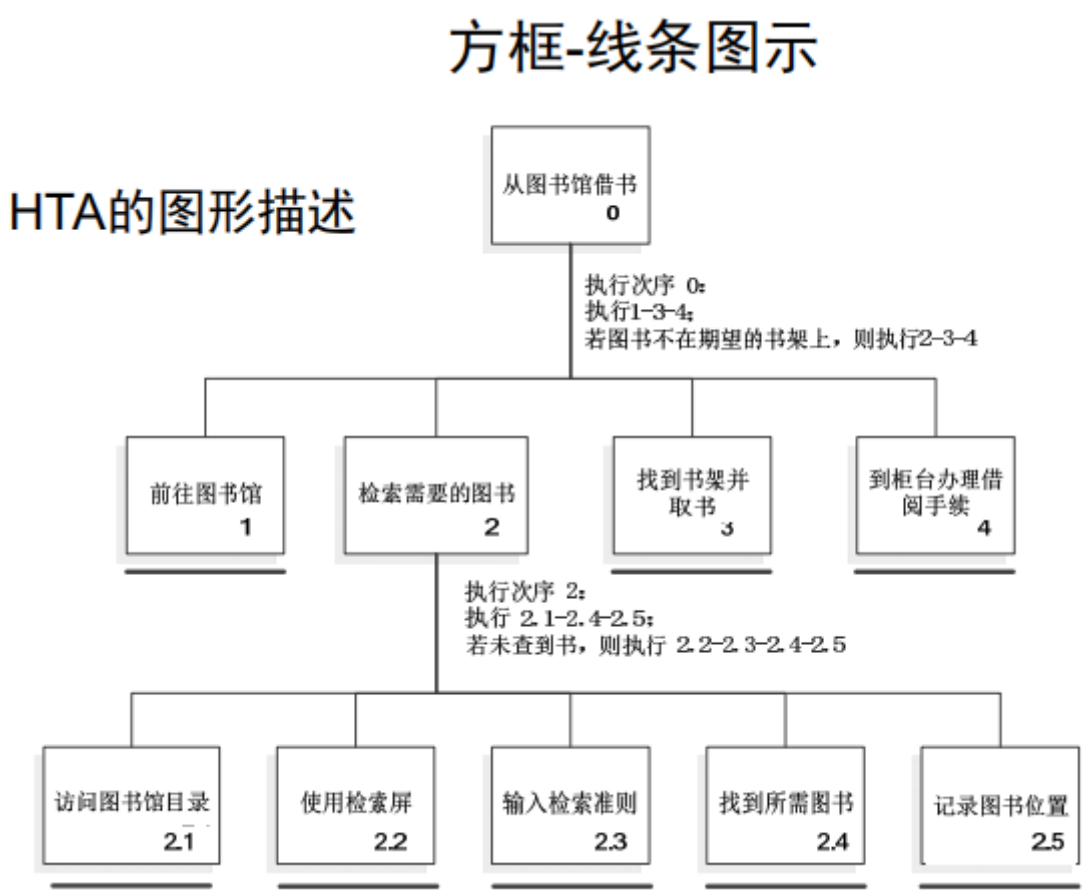
需求定义

- 步骤
  - 创建问题和前景综述
  - 头脑风暴
  - 确定人物角色
  - 构建场景：讲故事，关注心理模型
  - 确立需求

任务分析

- 层次化任务分析
  - 把任务分解为若干子任务，再把子任务分解
    - 终止规则
      - 任务包含复杂的机械响应：如移动鼠标
      - 涉及内部决断：若决断需要进行查找文档等外部动作，则继续分解，否则终止
  - 之后，把他们组织成一个 **执行次序**，说明在什么样的情形下执行什么任务
- 举例：借书
  - 1. 前往图书馆
  - 2. 检索所需图书
    - 2.1 访问图书馆目录
    - 2.2 使用检索屏
    - 2.3 输入检索规则
    - 2.4 找出需要的图书
    - 2.5 记录图书位置
  - 3. 找到书架取书

    - 4. 到柜台办理借阅手续
    - 执行次序：执行 1-3-4；若图书不在架上执行 2-3-4
    - 执行次序-2：执行 2.1-2.4-2.5；若未查到此书，则执行 2.2-2.3-2.4-2.5
- 图形描述



- 用途
    - 作为手册教学
    - 需求获取：任务分析本身不是需求获取，但有助于需求的完整表达
    - 详细的接口设计：应用于菜单设计

原型

- 低保真原型
    - **多数项目的首选**
    - 与最终产品不太相似，使用不同的材料：如用纸板代替电脑屏幕
    - 优点是简单、便宜、易于制作和修改
- 高保真原型
    - 与最终产品更为接近，使用相同的材料
    - 风险
        - 用户可能会认为原型就是最终的系统
        - 开发人员可能会认为已经找到了一个用户满意的设计

交互设计

简化设计的四种策略

- 删除
  - **最明显**
  - 目标：删除杂乱的特性
    - 可以让设计师专注于把有限的重要的问题解决好
    - 有助于用户心无旁骛地完成自己的目标
    - 避免简单功能堆叠成的毫无特色的产品，保证只交付那些真正有价值的功能和内容
  - 如何删除
    - 关注核心功能
    - 砍掉残缺功能
      - 沉没成本误区：为什么要留着它?
    - 不要简单地因为客户要求就增加功能
- 组织
  - **最快捷**
  - 分块
    - 确定清晰的分类标准
    - 典例：Word 顶部的工具栏
  - 利用不可见的网格来对齐元素
  - 大小与重要性挂钩
  - 把相似的元素放在一起
- 隐藏
  - **低成本**
  - 用户不会被不常用的功能分散注意力
  - 可从删除不必要的功能开始
  - 隐藏什么
    - 主流用户很少使用，但自身需要更新的功能
    - 选项和偏好
    - 特定于地区的信息
  - 自定义
    - 主流用户感兴趣的是展示自己的个性，如自定义桌面图片，而非重新设计用户界面
    - 一般来说，不应该让用户自定义软件，但如果自定义的工具很简单，还是有价值的
  - 渐进展示
    - 隐藏精确的控制部件
    - 为主流用户提供包含少数核心功能的控制部件，为专家级用户准备扩展性的精确的控制部件
    - 比自定义的效果更好
  - 让功能易于发现
    - 为隐藏功能入口打上 “更多”“高级” 等标签
    - 把标签放在用户关注点的范围内
- 转移
  - 把精简掉的按钮通过电视屏幕上的菜单来管理
  - 不同平台负责不同任务
    - 比如手机可以拍摄任何景物，而电脑只能拍到用户

        - 但手机只能输入和显示少量信息，但电脑可以输入大量信息
    - 把复杂的工作留给用户
        - 不要为用户提供复杂的、限制多的功能
        - 提供简单的功能，复杂功能的构建由用户自己完成

交互模型

- 预测模型：能够预测用户的执行情况，但不需要对用户做实际测试
    - 适合于无法进行用户测试的情形

击键层次模型

- 1983 Card 等
- 对用户执行情况（时间）进行量化预测
- 用途
    - 预测无错情况下，专家用户，在特定输入前提下完成任务的时间
    - 便于比较不同系统
    - 确定何种方案能最有效地支持特定任务
- 操作符表

操作符名称	描述	时间（秒）
K	按下一个单独按键或按钮	0.35（平均值）
	熟练打字员（每分钟键入 55 个单词）	0.22
	一般打字员（每分钟键入 40 个单词）	0.28
	对键盘不熟悉的人	1.20
	按下 shift 键或 ctrl 键	0.08
P	使用鼠标或其他设备指向屏幕上某一位置	1.10
P_1	按下鼠标或其他相似设备的按键	0.20
H	把手放回键盘或其他设备	0.40
D	用鼠标画线	取决于画线的长度
M	做某件事的心理准备（例如做决定）	1.35
R(t)	系统响应时间——仅当用户执行任务过程中需要等待时才被计算	t

- 使用
    - 例一：在 DOS 环境下执行 ipconfig 命令
        - 列出操作次序
            - M：决定要执行 ipconfig
            - K[i] ... K[g]：按键输入
            - K[回车]：回车执行
            - 可简写为 M9K[ipconfig回车]
        - 累加操作时间
            - T = 1.35 + 0.28 * 8 + 0.28 = 3.87s
    - 例二：右键网络连接点击修复
        - H[鼠标]MP[网络连接图标]P1[鼠标右键]P[修复]P1[鼠标左键]
        - T = 0.4 + 1.35 + (1.1 + 0.2) * 2 = 4.35s
- 放置 M 操作符的启发规则

    - 【总结：删除整体字符串中间和复合操作中间的 M】
    - 使用规则 0 放置所有候选 M，然后循环执行规则 1-4，删除 M
    - 规则 0：在所有 K 前插入 M，要求 K 的值不能是变量字符串的一部分；在所有选择命令的 P 前插入 M
    - 规则 1：若某个操作符前的 M 完全可以由之前的某个 M 预测，则删除这个 M
    - 规则 2：若一串 MK 是连续的认知单元，则只保留第一个 M
    - 规则 3：若 K 是一个冗余的终结符，则删除 K 前的那个 M
    - 规则 4：如果 K 是变量字符串的终结符，则保留 K 之前的 M

Fitts 定律

- 能够预测使用某种定位设备指向某个目标的时间
    - 根据目标大小、至目标的距离，计算指向目标的时间
    - 对图形用户界面设计有明显的意义
    - 最健壮并被广泛采用的人类运动模型之一
- 建议
    - 大目标、小距离的移动时间更短
    - 屏幕元素应该尽可能多地占据屏幕空间
    - 最好的像素是光标所处的像素
    - 屏幕元素应尽可能利用屏幕边缘的优势
    - 大菜单，如饼型菜单，比其它类型的菜单使用更简单
- 应用
    - HCI 领域
        - 鼠标的定位时间和错误率都优于其它设备
        - 鼠标速率接近最快速率
        - 使用鼠标完成运动任务比其它设备更加协调
    - 缩短当前位置到目标区域的距离
        - 右键菜单
    - 增大目标大小以缩短定位时间
        - Mac OS 工具栏图标大小可以动态改变

交互评估

- 评估优点
    - 能够在交付产品之前修复错误
    - 设计小组能够专注于真实问题而不是假象问题
    - 能够缩短开发时间
    - 销售部门能够获得稳定的设计
- 评估目标
    - 评估系统功能的范围和可达性
    - 评估交互中用户的体验
    - 确定系统的某些特定问题
- 评估原则
    - 评估应该依赖于产品的用户
    - 评估与设计应该结合进行
    - 评估应在用户实际工作的操作环境下进行
    - 要选择具有广泛代表性的用户

评估泛型

- 快速评估
  - 设计人员非正式地向用户或顾问了解反馈信息，以证实设计构思是否符合用户需要
  - 特征：快速
  - 可在任何阶段进行
  - 得到的数据通常是非正式的、叙述性的
  - 是设计网站的常用方法
- 可用性测试
  - 评测典型用户执行典型任务时的情况（出错次数，完成时间）
  - 特征：在评估人员的密切控制之下进行
  - 作用：测试原型或产品
  - 缺点
    - 测试用户的数量通常较少
    - 不适合进行细致的统计分析
- 实地研究
  - 特征：在自然工作环境中进行
  - 目的：理解用户的实际工作情形以及技术对他们的影响
  - 作用：常用于设计初期，以检查设计是否满足用户需要，发现问题，发掘应用契机
- 预测性评估
  - 专家们根据自己对典型用户的了解预测可用性问题
  - 特征
    - 用户可以不在场
    - 过程快速、成本较低
  - 典例：启发式评估

评估范型	快速评估	可用性测试	实地研究	预测性评估
用户角色	自然行为	执行测试任务集	自然行为	用户通常不参与
控制权	评估人员实施最低限度控制	评估人员密切控制	评估人员与用户合作	评估人员为专家
评估地点	自然工作环境或实验室	实验室	自然工作环境	类似实验室的环境，通常在客户处进行
适用情形	快速了解设计反馈。可使用其他交互范型的技术，如专家评测	测试原型或产品	常用于设计初期，以检查设计是否满足用户需要，发现问题，发掘应用契机	专家（通常是开发顾问）检查原型，可在任何阶段进行。使用模型评测潜在设计的特定方面
数据类型	通常是定性的非正式描述	量化数据，有时是统计数据。可采用问卷调查或访谈搜集用户意见	应用草图、场景、例证等的定性描述	专家们列出问题清单，由模型导出量化数据，如两种设计的任务执行时间
反馈到设计	通过草图、例证、报告	通过性能评测、错误统计报告等为未来版本提供设计标准	通过描述性的例证、草图、场景和工作日志	专家列出一组问题，通常附带解决方案建议。为设计人员提供根据模型计算出的时间值
基本思想	以用户为中心，非常实用	基于试验的实用方法，即可用性工程	可以是客观观察或现场研究	专家检查以实用的启发式原则和实践经验为基础，采用基于理论的分析模型

评估技术

- 观察用户
  - 有助于确定新产品需求，评估原型
  - 难题
    - 如何在不干扰用户的前提下观察用户
    - 如何分析大量数据
- 询问用户意见
  - 简单
  - 调查数量从几个到几百不等
- 询问专家意见
  - “角色扮演”方式评估
  - 同时，专家会提出解决方案
- 测试用户的执行情况
  - 比较不同设计方案的优劣
  - 通常在受控环境中进行
- 基于模型和理论预测界面的有效性
  - GMOS 和 KLM 模型等

评估技术	评估范型			
	快速评估	可用性测试	实地研究	预测性评估
观察用户	观察用户实际行为的重要方法	使用摄像和交互日志的记录方式，可做进一步分析，以找出问题，了解操作步骤，计算执行时间	实地研究的核心方法。在现场研究中，评测人员与测试环境相融合；在其他类型的研究中，评测人员只做客观观察	——
询问用户意见	与用户和潜在用户讨论，可采用个别会谈、集体会谈或专门小组的形式	通过问卷调查了解用户满意度，也可通过访谈了解更多详情	评测人员可采用访谈的形式，与用户讨论观察到的问题。现场研究可采用现场访谈	——
询问专家意见	专家评估原型的可用性（提供“评估报告”）	——	——	在设计初期，专家使用启发式原则预测界面的有效性
用户测试	——	在受控环境中，测试典型用户执行典型任务的情况，是可用性测试的基本方法	——	——
用户执行情况分析模型	——	——	——	使用分析模型预测界面的有效性，或比较用户使用不同设计方案的执行效率

- 评估技术比较

方法	生命周期阶段	用户人数	主要优点	主要缺点
启发式评估	早期设计，反复设计过程的“内循环”	无	能发现单个可用性问题，能发现熟练用户碰到的问题	没有涉及真实的用户，故无法再用户需求方面有“惊人发现”
绩效度量	竞争性分析，最终测试	至少 10 人	硬性数据，对结果容易进行比较	不能发现单个可用性问题
边做边说	反复设计，形成性评估	3~5 人	准确了解用户的错误想法，测试费用低	用户感到不自然，熟练用户感到很难用语言表述
观察	任务分析，后续研究	3 人或以上	生态有效性；发现用户的真实人物；建议系统功能与特征	很难约定安排，实验人员无法控制
问卷调查	任务分析，后续研究	至少 30 人	发现用户主观偏好，容易重复进行	需要进行问卷预答（避免出现误解）
访谈	任务分析，后续研究	5 人	灵活，可以深入了解用户观点和用户体验	耗时，难以进行分析、比较
焦点小组	任务分析，用于参与	每组 6~9 人		分析起来困难，有效性低
使用过程记录	最终测试，后续研究	至少 20 人		
用户反馈	后续研究	上百人	跟踪用户需求和想法上的变化	需要专门部门来处理回复

- 常用组合
  - 启发式评估 + 边做边说
    - 专家可以通过启发式评估排除显而易见的可用性问题
    - 重新设计后，经用户测试，反复检查设计效果
  - 访谈 + 问卷调查
    - 先对小部分用户进行访谈，以确定问卷中的具体问题

观察用户

> 略，见 PPT 3

询问用户和专家

询问用户

- 访谈
  - 步骤
    - 开始阶段：解释访谈原因，消除道德疑虑，询问受访人是否介意被记录
    - 热身阶段：先提出简单问题
    - 主要访谈阶段：按逻辑次序有易到难提问
    - 冷却阶段：提出若干容易的问题，消除用户紧张的感觉
    - 结束访谈
  - 类型
    - 非结构化：开放式问题，受访人自行决定详细回答or简要回答
    - 结构化：封闭式问题，预先准备一组问题，要求准确回答
    - 半结构化
    - 集体访谈：个别成员的看法是在应用的上下文中通过与其他用户的交流而形成的
      - 焦点小组 是集体访谈的一种形式
- 焦点小组
  - 由大约 6-9 个典型用户组成

    - 如在评估大学的网站时，可考虑由行政人员、教师、学生分别组成 3 个焦点小组
    - 主持人事先列出一张讨论问题和数据收集目标的清单
- 问卷调查★
    - 用于搜集统计数据和用户意见的常用方法
    - 设计原则
        - 确保问题明确、具体
        - 如可能，采用封闭式问题并提供充分的答案选项
        - 对于征求用户意见的问题，应提供一个“无看法”的答案选项
        - 先提出一般化问题，再提出具体问题
        - 避免使用复杂的多重问题
        - 在使用等级标度时，应设定适当的等级范围，并确保它们不重叠
        - 避免使用专业术语
        - 明确说明如何完成问卷（如说明，在选项前的方框内打勾）
        - 在设计问卷时，既要做到紧凑，也应适当留空
    - 问题类型
        - 常规问题：年龄，性别等
        - 自由回答问题：你能提出一些改进意见吗?
        - 量化分级问题：不同意 1 2 3 4 5 同意（Likert 尺度）
        - 多选题：手机用户以前的经验信息
    - 关键问题
        - 如何寻找有代表性的用户?
        - 如何达到合理的回复率?
            - 精心设计问卷，避免用户因为厌烦而拒绝回复
            - 提供简要描述，说明用户若没有时间完成全部问卷，可以只完成简短部分
            - 解释为什么要进行问卷调查，并说明将为参与者保密
            - 通过后续邮件、电话联系参与者
            - 采取一些激励措施
    - 在线问卷调查
        - 更快速，成本更低，数据分析更方便，容易修改，但回复率可能更低
        - 电子邮件：能针对特定用户，但能容纳的内容有限
        - 网页：形式灵活，能验证数据的有效性，但调查对象随机
    - 对比访谈
        - 都属于间接方法
            - 因为都不读用户界面本身进行研究，而是研究用户对其看法
            - 所以都不能完全听信和采纳用户的说法
        - 访谈
            - 形式更自由，可在访谈后立即得到结果
            - 但难以获得确切数据，需要花费更多时间，用户可能回避某些敏感问题的真实想法

询问专家

- 认知走查
    - 目标：确定使一个系统如何易于学习
        - 试图想象出人们在 **第一次** 使用某个产品时的想法以及所采取的动作
    - 步骤
        - 标识并记录典型用户的特性
        - 基于评估重点，设计样本任务（大多数用户需要执行的典型任务）

        - 制作界面原型（或界面描述），明确用户执行任务的具体步骤
        - 由设计人员和专家级评估人员共同进行分析
        - 评估人员结合应用的上下文，逐步检查每项任务的操作步骤
            - 正确的操作对于用户是否足够明显？
            - 用户能否理解正确的操作？
            - 能否正确解释操作的响应？
        - 在完成逐步检查之后，汇总关键信息
        - 修改设计，更正发现的问题
    - 优点
        - 不需要用户参与
        - 也不需要可运行的原型
        - 能找出非常具体的用户问题
    - 缺点
        - 工作量大，非常费时
        - 只适合于评价易学性，不能发现其他可用性问题
- 启发式评估★ Nilsen提出
    - 一种灵活而又相当廉价的评估方式
    - 由可用性专家完成
    - 步骤
        - 彻底检查界面
        - 将界面与启发式规则进行对比
        - 列举可用性问题
        - 应用启发式规则对每一个问题进行解释与确认
    - 问题严重性
        - 考察三个方面：频率，影响，持续时间
        - 分为四个等级：表面问题，次要问题，主要问题，灾难性问题
    - 评估步骤
        - 准备阶段
            - 确定可用性准则
            - 确定由 3-5 个可用性专家组成的评估组
            - 计划地点、日期和每个专家评估的时间
            - 准备或收集材料，让评估者熟悉系统的目标和用户
            - 设定评估和记录的策略（基于个人or小组？）
        - 评估阶段
            - 尝试并建立对系统概况的认知
            - 阅读提供的材料以熟悉系统设计
            - 按评估者认为完成用户人物所需的操作进行实际操作
            - 发现并列出系统中违背可用性原则之处
        - 分析阶段
            - 回顾记录的问题，确保每个问题能让所有评估者都理解
            - *看不见*
        - 汇总阶段
            - *看不见*
            - 让项目组的另一个成员审查报告，并由项目领导审定
    - 如何正确评估
        - 把每个问题对应到启发式规则

        - 列出所有问题
        - 至少遍历两次界面，一次获得系统的初始体验，另一次关注特定元素
        - 不仅仅局限于十条启发式规则
    - 优点
        - 不涉及用户，所以面临的实际限制和道德问题较少
        - 成本相对较低，不需要特殊设备，而且较为快捷
    - 缺点
        - 评估人员需要经过长时间的训练才能成为专家
            - 理想的专家应同时具备交互设计和产品应用域的知识
        - 可能出现“虚假警报”
            - 专家每找到一个真实的问题，将发出约一个假警报，忽略半个问题

用户测试

> 主要看自己的作业

- 在受控环境中（比如实验室环境）测量典型用户执行典型任务的情况
- 目的是获得客观的性能数据，从而评价产品或系统的可用性
- 最适合对原型和能够运行的系统进行测试
- 可对设计提供重要的反馈
- 测试设计
    - 考虑实际限制
        - 确保不同参与者测试条件相同
        - 确保评估目标特征具有代表性
    - 1、定义目标和问题
        - 目标描述了开展测试的原因，定义了测试在整个项目中的价值
        - 目标是对关注点的说明和解答（比如：对菜单结构的关注，目标是“用户在少于 5s 的时间内能够导航到正确的三级菜单”）
    - 2、选择参与者
        - 了解用户的特性有助于选择典型用户（要尽可能接近实际用户）‘
        - 通常需要平衡性别比例
        - 通常 5-12 位
        - 安排参与者参与不同的情形
            - 随机指派参与者执行某个情形
                - 缺点
                    - 需要有足够多的参与者
                    - 实验结果可能受到个别参与者的影响
                - 优点
                    - 不存在“顺序效应”，即前一组任务获得的经验会影响后续任务
            - 参与者执行所有情形
                - 缺点
                    - 可能产生”顺序效应“（但可以用均衡处理解决，比如有两项任务 A 和 B，让一般的参与者 先 A 再 B，另一半 先 B 再 A）
                - 优点
                    - 需要更少的参与者

                - 能够消除个别差异带来的影响
                - 便于比较参与者执行不同实验情形的差异
            - 根据用户特性，将两位参与者组成一组，再随机安排他们执行某个情形
                - 缺点
                    - 实验结果可能会受一些未考虑到的重要变量的影响
                    - 最好的办法是将这些有影响的重要变量作为配对标准
    - 3、设计测试任务
        - 测试任务应当与定义的目标相关
        - 测试任务通常是简单任务，有时采用较为复杂的任务
        - 任务不能仅限于要测试的功能，应使用户全面地使用设计的各个区域
        - 每项任务的时间应介于 5-20min
        - 应当与某些合乎逻辑的方法安排任务（先提出简单要求有助于增强用户信心）
    - 4、明确测试步骤
        - 在测试之前，准备好测试进度表和说明，设置好各种设备
        - 正式测试前应进行小规模测试
        - 在必要时，评估人员应询问参与者遇到了什么问题
        - 若用户确实无法完成某些任务，应让他们继续下一项任务
        - 测试过程应控制在1小时之内
        - 必须分析所有搜集到的数据
    - 5、数据收集
        - 确定如何度量观测的结果
        - 常用的定量度量
            - 完成任务的时间
            - 执行每项任务时的出错次数和错误类型
            - 单位时间内的出错次数
            - 求助或使用手册的次数
            - 犯某个特定错误的次数
            - 成功完成任务的用户数
    - 6、数据分析
        - 自变量：比如，字体大小
        - 因变量：比如，阅读文本的时间
    - 7、总结报告
        - 将测试的结果以书面形式反馈给产品的设计人员

以用户为中心

- 设计思想
    - 以真实用户和用户目标作为产品开发的驱动力，而不仅仅是以技术为驱动力
    - 充分利用人们的技能和判断力；支持用户，而不是限制用户
    - 需要透彻了解用户及用户的任务，并使用这些信息指导设计
- 三个假设
    - 好的设计结果使客户感到满意
    - 设计过程是设计人员与客户之间的协作过程
    - 在整个过程中，客户和设计人员要不断沟通
- 设计原则
    - 尽早以用户为中心：在设计过程早期就致力于了解用户需要
    - 综合设计：设计的各方面应该齐头并进
    - 尽早并持续性进行测试：如果实际用户认为设计是可行的，那它就是可行的

    - 迭代设计
- UCD 方法
    - 用户参与：成为设计团队的一部分
        - 全职/兼职参与开发会议：用户能彻底理解系统，但会逐渐失去普通用户的代表性（可以考虑周期性更换用户）
        - 以邮件、专题讨论等形式定期参与
    - （剩下的就是评估技术）

以活动为中心

- 以用户为中心的缺陷
    - 影响产品的创新性
    - 可操作性受到时间、预算、任务规模的限制
    - 忽略了人的主观能动性和对技术的适应能力
- 设计思想
    - 把用户要做的事（活动）座位重点关注对象
    - 更适合于复杂项目的设计
- 其实就是把人和技术综合起来考虑，不单纯考虑人/技术，而是关注事情本身