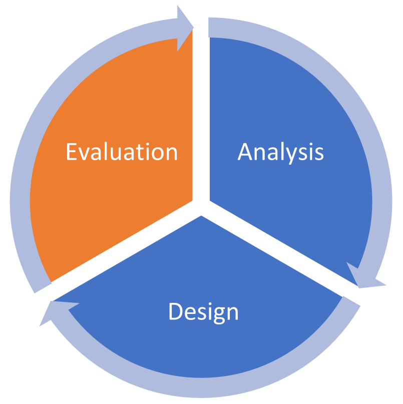
Human Factors and Human-Machine Interaction

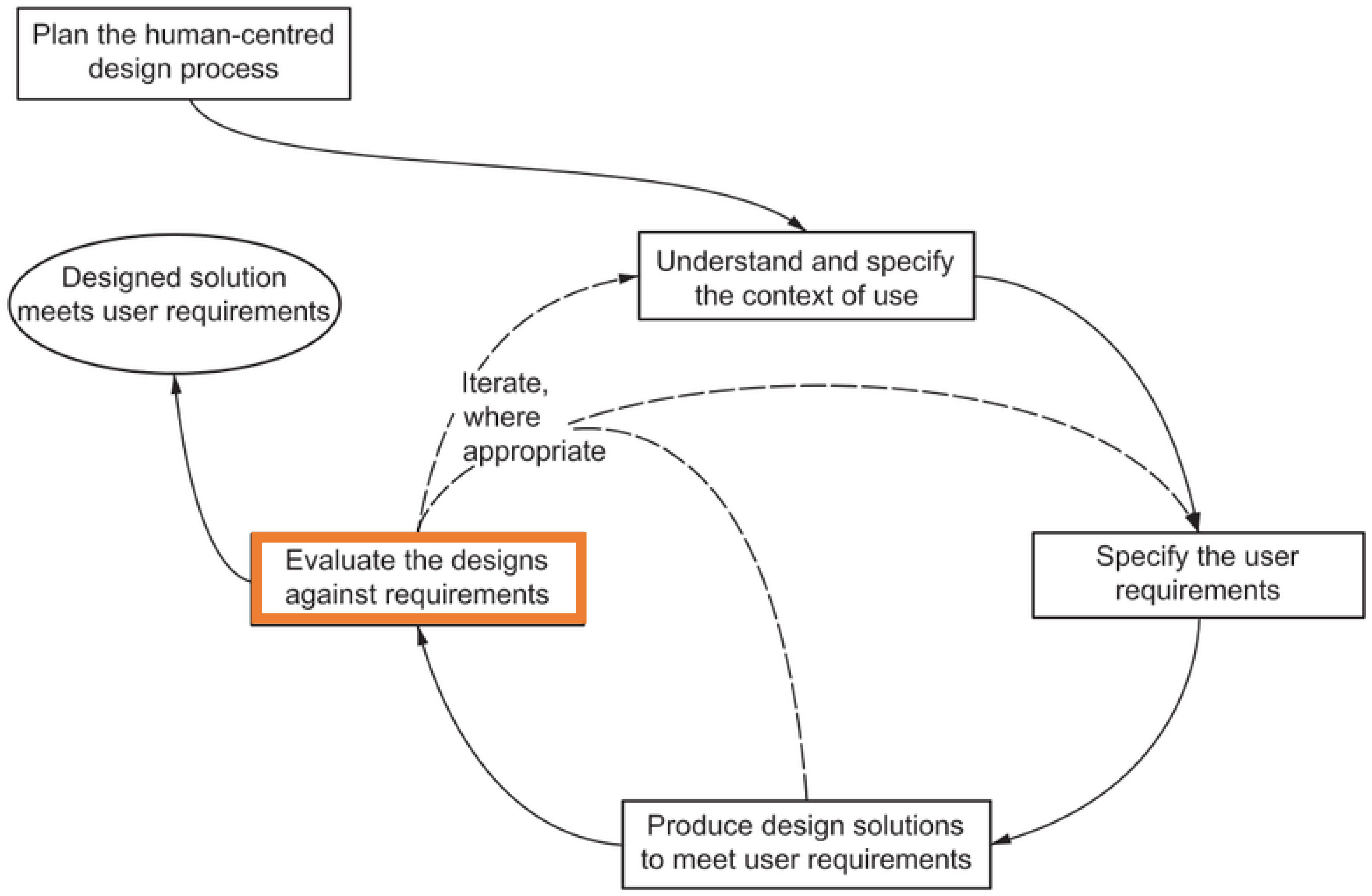
Usability Engineering – Evaluation: Inspection

FACULTY

OF COMPUTER SCIENCE

USABILITY: PROCESS




Plan the human-centred design process
Understand and specify the context of use
Specify the user requirements
Produce design solutions to meet user requirements
Evaluate the designs against requirements
Designed solution meets user requirements
Iterate, where appropriate

USABILITY EVALUATION

- Reference point: User requirements
- Examination and evaluation of the system by users and/or experts
- System assessment: absolute judgment vs. system optimization: identification of strengths and weaknesses
- Types of evaluation:
    - Summative vs. formative
    - Comparative: Product A vs. B
    - Evaluative: Product A - Attribute x
    - Analytical: Product A - Problems
    - Empirical vs. analytical

UE: ANALYTICAL VS. EMPIRICAL

- Analytical:
  - **Model-based**: e.g., GOMS
  - Usability **Inspection** (UI): e.g., Heuristic Evaluation, Cognitive Walkthrough
- Empirical:
  - **Questionnaires**: e.g., SUS, PSSUQ, AttrakDiff
  - Usability **Tests**

MODEL-BASED: KLM

- Video about GOMS & KLM on YouTube: https://www.youtube.com/watch?v=Ocz1hV34flk
- KLM = Keystroke-level model
- GOMS variant
- Required times on average in experimental conditions:
    - K (Keystroke): Pressing a key: **t_k = 0.28s**
    - P (Pointing): Pointing to a screen position: **t_P = 1.1s**
    - H (Homing): Switching between keyboard and mouse: **t_H = 0.4s**
    - M (Mental preparation): Mentally preparing for a subsequent operation: **t_M = 1.35s**
    - R(t) (response time)

EXAMPLE KLM

Which of the methods M1 or M2 is faster?

M1: *Switch to the mouse, move the mouse cursor to the file, click on the file, drag it to the trash can, and release it, switch to the keyboard.*

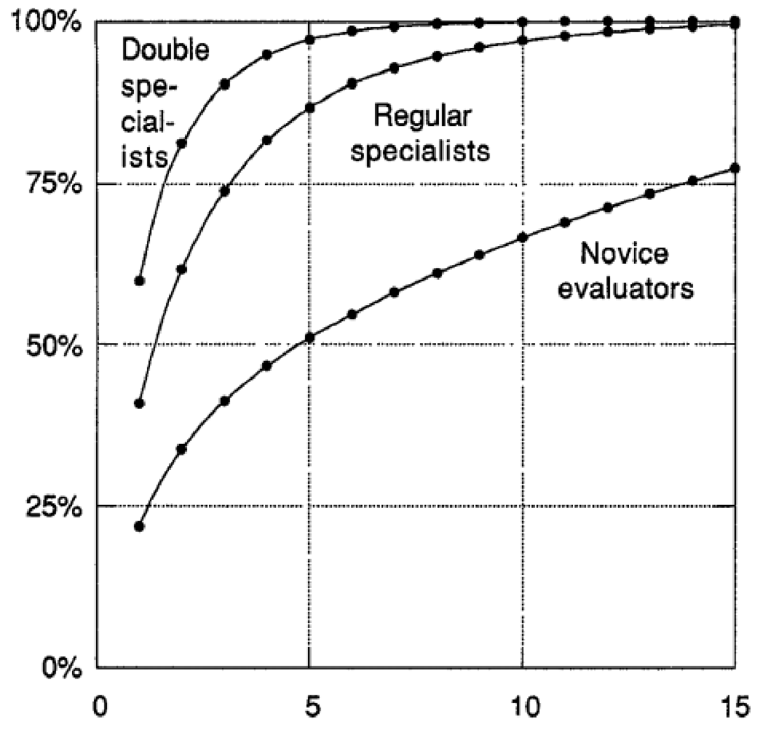
M2: *Switch to the mouse, select the file, switch to the keyboard, press the Delete key.*

$t_{M1} = t_H + t_P + t_K + t_P + t_H = 0.4 + 1.1 + 0.28 + 1.1 + 0.4 = 3.28s$

$t_{M2} = t_H + t_P + t_H + t_K = 0.4 + 1.1 + 0.4 + 0.28 = 2.18s$

UI: NUMBER OF EVALUATORS

- Nielsen and Molich (1990)
  - 1 usability specialist discovers 38% of usability problems
  - 10 lay evaluators: approximately 2/3
  - 5 double experts: > 95%
- $G = N (1 - (1 - p)^i)$
  - G: discovered usability problems
  - N: total number of usability problems
  - p: proportion of problems that an evaluator finds
  - i: number of evaluators
- Recommended number: about 3 to 10 evaluators



METHODS: EVALUATION

- **Discount Usability Inspection**: Heuristic Evaluation, Cognitive Walkthrough
- Questionnaires
- (DIY) Usability Tests
- Big Data: Automated Evaluation
- Context-oriented methods

- Classic Usability Tests
- User Modeling

USABILITY INSPECTION

Usability inspection is the generic name for a set of **methods** that are all **based on having evaluators inspect a user interface**. Typically, usability inspection is aimed at finding usability problems in the design, though some methods also address issues like the severity of the usability problems and the overall usability of an entire system. Many inspection methods lend themselves to the inspection of user interface specifications that have not necessarily been implemented yet, meaning that inspection can be performed early in the usability engineering lifecycle. (Nielsen, 1994)

HEURISTIC EVALUATION

- **Evaluators** review usability-relevant aspects of a product
- **Goal**: Identification of **problems**; not necessarily generating solution proposals
- **Problem**: Characteristics of a product that could affect the efficiency or effectiveness of interaction or user satisfaction (cf. definition)
- Located in the realm of **Discount Usability Engineering**
- Usability **experts** find more problems than laypeople
- **Double experts**: Usability experts with domain knowledge achieve the highest rate

HE: PROCESS

- Description of the **target audience**.
- Creation of one or more **usage scenarios** (e.g., tasks; see Key-Path scenarios).
- Embedding in a **context scenario with persona**.
- Selection and determination of the **heuristics**.
- Each expert conducts the **evaluation** individually and documents usability issues.
- Classification and **prioritization** of usability problems (e.g., "catastrophic," "major problem," "minor problem," and "cosmetic problem").
- Compilation of a problem list/**report** (organized by heuristics/problem categories or functional groups or information processing model, etc.).

HE PROCESS: SEVERITY RATINGS

Problem evaluation along three dimensions:

- **Problem Frequency**: Does the problem occur in many or few interaction situations?
- **Problem Impact**: To what extent does it affect task performance?
- **Persistence**: Is the problem easily avoidable once it is known?

Nielsen (1994) suggests the following **scale**:

0. I don't agree that this is a usability problem at all.

1. Only a **cosmetic issue** - doesn't need to be addressed unless additional time is available.

2. **Minor usability problem** - fixing it has low priority.

3. **Major usability problem** - should be fixed with high priority.

4. **Usability catastrophe** - must be fixed before product launch.

HE: CLASSIC HEURISTICS

1. Visibility of system status

The system should always inform the user about what is happening, providing appropriate **feedback** within a reasonable time frame.

2. Match between system and the real world

The system should speak **the user's language**, using words, phrases, and concepts that are familiar to the user, rather than system-oriented terms. Follow real-world conventions so that information appears in a natural and logical order.

3. User control and freedom

Users often choose software functions accidentally. They need a clearly marked "**emergency exit**" to be able to leave an unintended state without having to go through an extensive dialogue. Support **undo** and **redo** actions.

HE: CLASSIC HEURISTICS

4. Consistency and standards

Users should not have to think about whether different terms, situations, or actions mean the same thing. Follow platform conventions.

5. Error prevention

Better than good error messages is a careful design that prevents problems from occurring in the first place.

6. Recognition rather than recall

Make objects, actions, and options visible. The user should not be forced to memorize information from one part of the dialogue to another. Instructions for system usage should be visible or easily retrievable whenever appropriate.

7. Flexibility and efficiency of use

Shortcuts that are not visible to the novice user can speed up interaction for experienced users, making the system suitable for both beginners and experts. Allow users to customize frequent actions to their needs.

HE: CLASSIC HEURISTICS

8. Aesthetic and minimalist design

Dialogues should not contain information that is irrelevant or rarely needed. Every extra piece of information in a dialogue competes with relevant information and reduces its relative visibility.

9. Support for error recognition, understanding, and recovery

Error messages should be presented in **plain language** (no codes). They should accurately describe the problem and provide constructive suggestions for resolving it.

10. Help and documentation

While it is preferable for the system to be usable without documentation, it may be necessary to provide help and information. Any such information should be easy to search, focus on the user's tasks, and provide concrete steps for execution. The documentation should not be overly extensive.

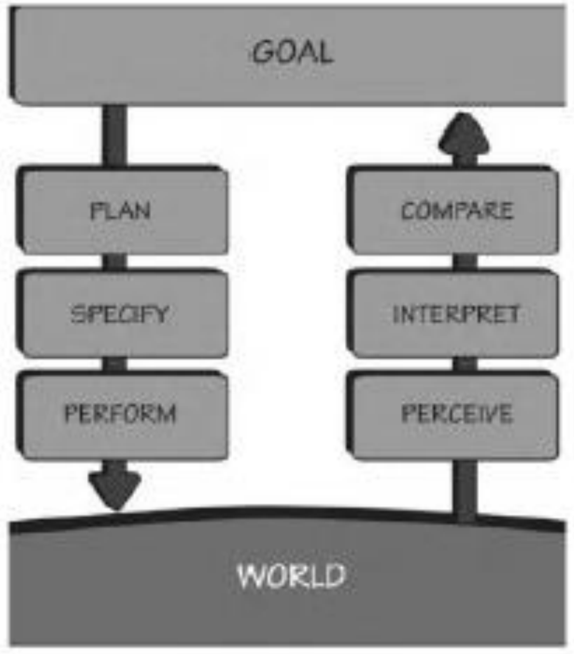
HE: ALTERNATIVE HEURISTICS

- Deriving heuristics from Don Norman's concepts - design of everyday things: Affordances, Signifiers, Constraints, Mappings, Feedback
  → **Design principles**

- **Seven Stages of Action**

- Deriving heuristics from the stages of visual perception, particularly the Gestalt principles.

- Deriving heuristics from the model of selective attention SEEV (Selective Encoding, Effortful Engagement of Attention, Voluntary Disengagement) by Wickens.



HE: ADVANTAGES

- No **user** required
- **Cost-effective**
- Relatively easy to **learn** - low implementation barriers
- Minimal **planning** and preparation effort
- Can be used early in the **development process**
- Well suited for **agile** methods and processes

HE: DISADVANTAGES

- **Poorly scalable** for complex interfaces
- **No solution proposals**

- Identification of relatively **minor problems** (controversial)
- Identifies **fewer problems** than usability tests (controversial)
- **False alarms** (controversial)

HE ON YOUTUBE

https://www.youtube.com/watch?v=J09MeSfOTJE

COGNITIVE WALKTHROUGH

- **Expert** method
- (User) target group: rather **inexperienced** users
- Starting point: **correct action sequences** (key-path scenarios)
- Provides **reasons** for problems
- Takes into account the **user's mental processes**
- Problem areas:
  - Mismatch between developer's and user's **concepts** (cf. Don Norman, DoET: Consistency of conceptual models)
  - Naming of **controls/elements**
  - **Feedback**

CW: PROCESS

The execution of the Cognitive Walkthrough can be divided into the following **steps**:

1. Definition of the **input**
2. **Examination** of the action sequences for each task
3. **Logging** critical information
4. **Revision** of the interface

CW-1: INPUT

- Description of **user characteristics**
- Selection of sample tasks (**scenarios**)
- Detailed **action sequences**
- Description or implementation of the **interface**

CW-2: ACTION SEQUENCES

Guiding Questions

- Will the user recognize that the correct action is available? ("*Perception*")
- Will the user attempt to achieve the desired effect? ("*Mental model*")
- Will the user establish a connection between the correct action and the desired effect? ("*Understanding*")
- If the correct action has been performed, will the user recognize the progress? ("*Feedback*")

CW-3: DOCUMENTATION

- **Information** and knowledge **required** for specific actions
- Potentially **error-prone actions**
- **Reasons** for potential errors
- No **design alternatives**

CW-4: REVISION

- The user does not attempt to achieve **the desired effect**:
  - Eliminate the action by having the system perform it automatically or combining it with another action.
  - Direct the user to the correct action to be performed.
  - Modify the interface to make it clearer why the action should be performed.
- The user does not **recognize** that the correct action is available:
  - Assign the action to a more obvious control element.
- The user fails to establish a **connection** between the correct action and the desired effect:
  - Modify the design/labeling of the control elements.
- The user does not receive **feedback** on their action:
  - Provide feedback on what is happening and the result of the action.

CW, HE VS. USABILITY TESTS

Usability inspection

- Not a substitute for usability **tests**
- Applicable even for **simplest prototypes**
- Fewer **problems** compared to usability tests
- Less **effort** required

Heuristic Evaluation and Cognitive Walkthrough

- Different **perspective**: CW focuses on **user** characteristics, HE focuses on **interface** characteristics
- HE has a broader approach; CW focuses on key tasks → Combination

CW ON YOUTUBE

https://www.youtube.com/watch?v=QcBPSmDW-Dc

1. Will users be trying to produce the right outcome?

EXERCISE: USABILITY INSPECTION

Search for an app on your smartphone that you want to evaluate.

Try out Heuristic Evaluation and/or Cognitive Walkthrough.