# Human Error? Bad Design?

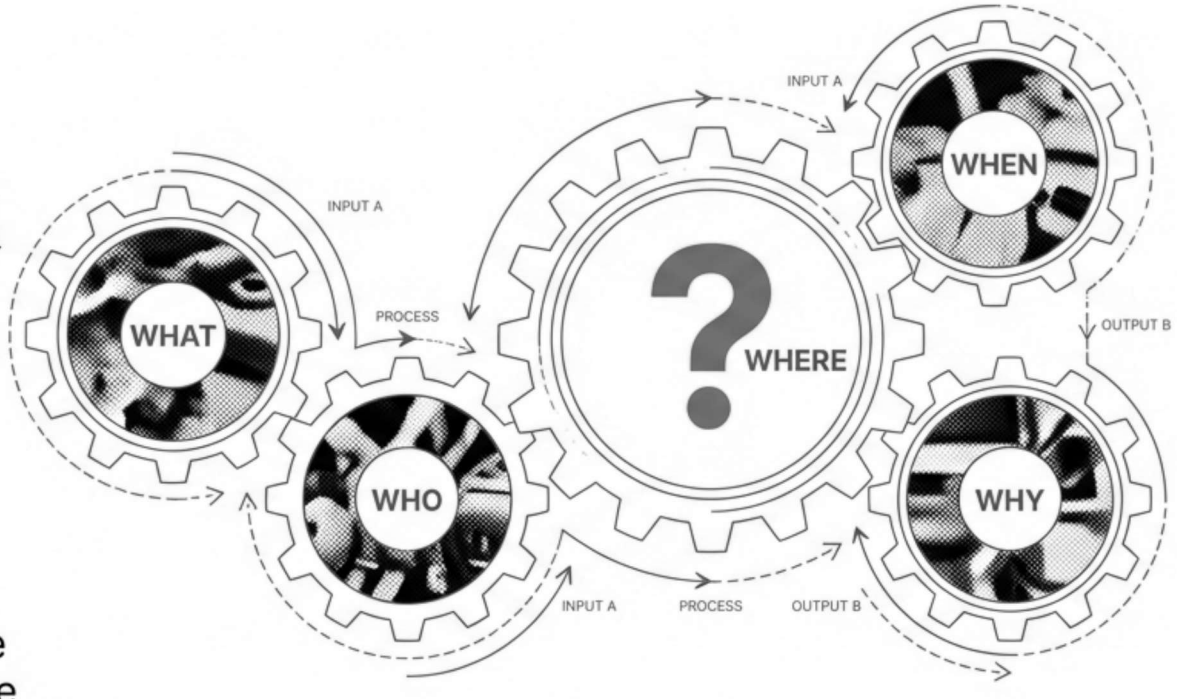
## Understanding Slips, Mistakes, and the Art of Usability

When things go wrong, our instinct is to blame the user. This deck explores why that instinct is wrong, how to diagnose the real root causes of error, and how to use usability testing to design systems that work for actual human brains.

# The Philosophy of Blame

"When an accident is thought to be caused by people, we blame them and continue to do things just as we've always done."

— Don Norman (p. 162)




**Core Concept:** If the system lets you make the error, it is badly designed. We must shift our mindset from blaming the We must shift our mindset from blaming the person to analson to analyzing the process.

# Moving Beyond Symptoms: Root Cause Analysis

**PROBLEM:** Client refuses to pay for leaflets.

↓ Why?

The delivery was late, so leaflets couldn't be used.

↓ Why?

The job took longer than expected.

↓ Why?

We ran out of printer ink.

↓ Why?

Ink was used on a last-minute order.

↓ Why?

We didn't have enough stock and couldn't order in time.

**COUNTER-MEASURE:** Find a supplier with shorter lead times.

## The 5 Whys

A problem-solving technique asking 'Why?' iteratively to move from surface symptoms to structural root causes.

# The Anatomy of Error: Slips vs. Mistakes


## SLIPS (Execution)

The “Whoops” Moment

- **Mechanism:** Unconscious / Autopilot
- **Logic:** Right Goal → Wrong Action
- **Example:** Driving to work instead of the store out of habit.

## MISTAKES (Evaluation)

The “I thought that was right” Moment

- **Mechanism:** Conscious / Thinking
- **Logic:** Wrong Goal → Right Action
- **Example:** Deleting a file intentionally, but misunderstanding what was inside it.

# When Autopilot Fails: Action-Based Slips

## Capture Slips

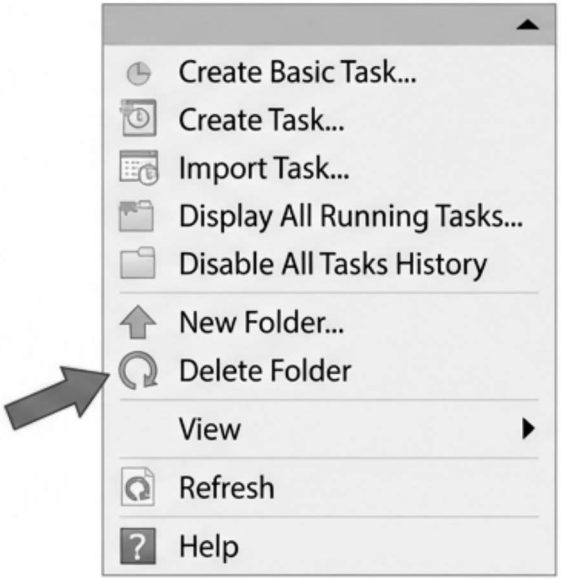
**Definition:** A frequent, practiced activity **takes over an intended action.**

**Cause:** Partial memory lapse + Strong habit.


## Description-Similarity

**Definition: Acting on the wrong object because it looks like the right object.**

**Cause:** Ambiguous visual design.



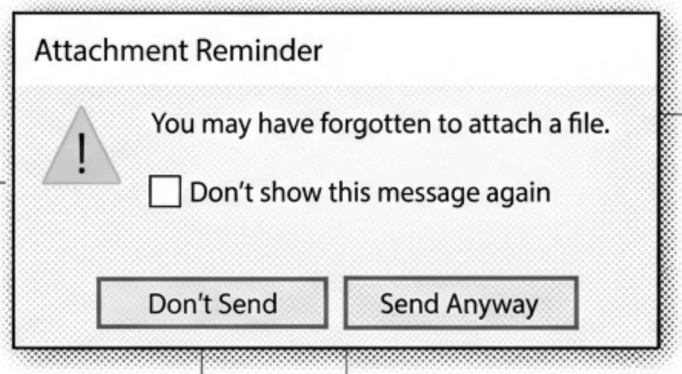
# Context Matters: Mode Errors & Memory Lapses

## Mode Errors

Different states – different meanings.
The same control performs different actions depending on the system's mode.

## Memory-Lapse Slips

Failure to perform all steps in a sequence due to interruption.



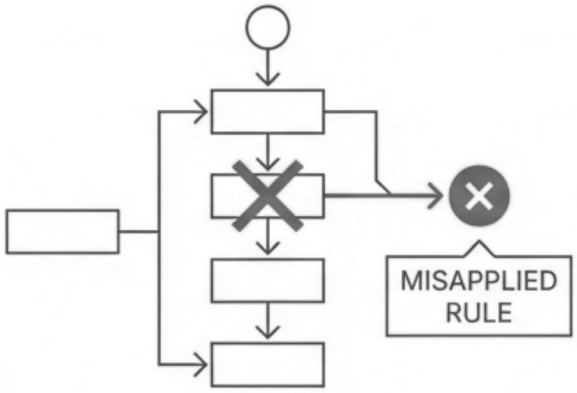
# When the Mental Model is Broken: Mistakes

Mistakes are errors of conscious deliberation. The user thinks they are right, but their plan is flawed.

### Rule-Based Mistakes

Applying the wrong formal procedure or rule to a situation.

**Root:** Misapplication of training.



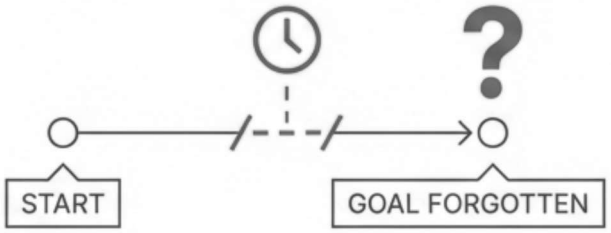

### Knowledge-Based Mistakes

Facing a new situation with no relevant experience or rules.


### Memory-Lapse Mistakes

Forgetting the goal entirely (unlike a slip, where you forget a step).

**Root:** Cognitive overload or long delay between plan and action.



# Designing the Cure

## **Treating Slips** (Interface Focus)

The user has the right goal but fails execution.

### Strategy

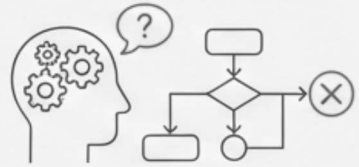
Prevent distractions, improve visual distinctiveness.

### Tactics


- Distinct icons (don't make 'Delete' look like 'Refresh').
- Forcing functions (popups for missing attachments).
- ← visual cues and constraints.

## **Treating Mistakes** (System Focus)

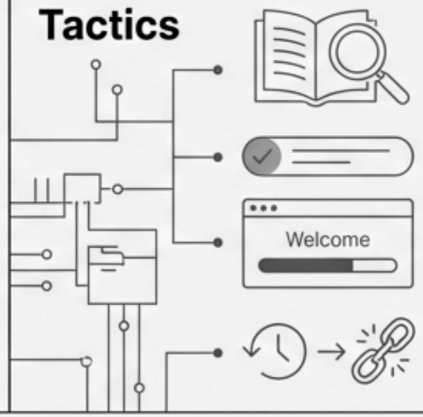
The user has the wrong goal or mental model.



### Strategy

Improve system clarity and feedback.

### Tactics

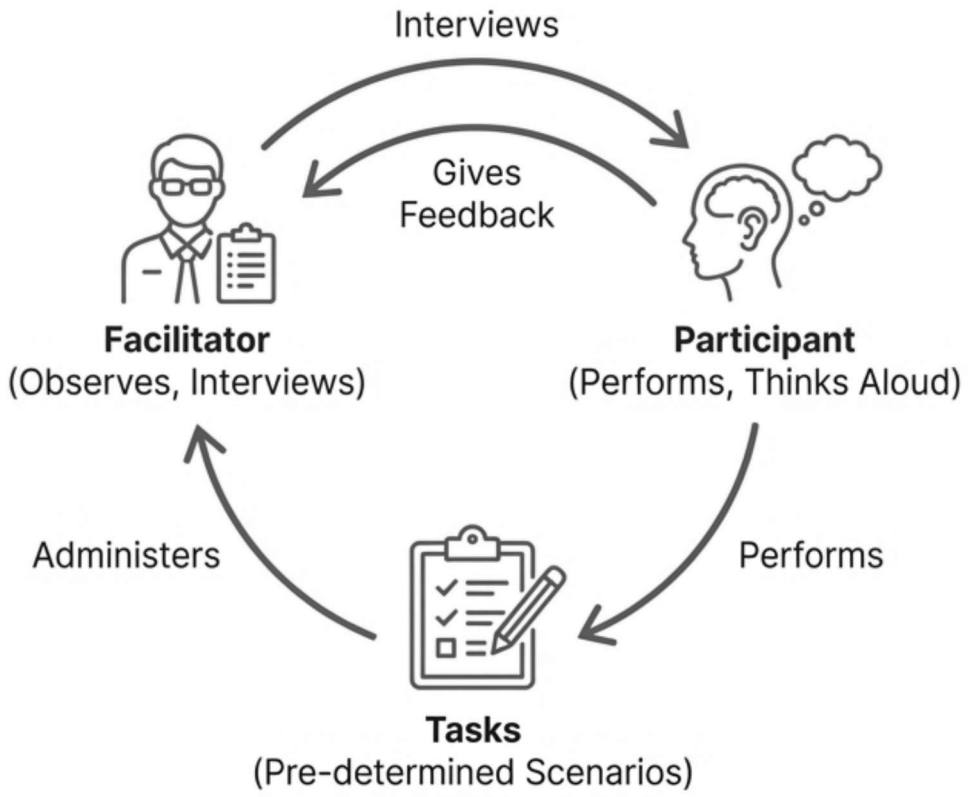



System status

- Better help documentation.
- Clearer feedback on system state.
- Educational onboarding.
- 'Undo' functionality to recover from bad decisions.

Clinical Swiss Editorial

# The Reality Check: Usability Testing




**Definition**

A method of testing functionality by observing real users as they attempt to complete specific tasks.

**Key Metrics**

**Efficiency** (Time), **Satisfaction** (Enjoyment), **Error Rates**

# Planning a Valid Test

## The Plan

**Scenarios:**
Define realistic goals. Give the user a destination (e.g., “Find a flight to Boston”), not a turn-by-turn instruction.

**Metrics:**
Decide what to measure. Quantitative (Time on task, completion rate) vs. Qualitative (Satisfaction).

**Recruiting:**
Recruit scrappily. You do not need a perfect demographic sample to find usability bugs.

# 3

## Users Per Round

### Users Per Round Is Enough.

You will uncover most major issues with just 3 users. It is better to run small, frequent tests than one massive study.

Clinical Swiss Editorial

# Protocols & Execution

How to observe without contaminating the data.

### Listen, Don't Lead

Don't explain the UI.
Don't judge the user.
Don't interrupt their struggle.

### Think Aloud

Encourage the user to vocalize their thought process as they work.

### Answer with Questions

If a user asks "What does this do?", ask back "What do you think it does?"

## Testing Environments

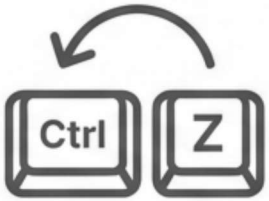
- **Lab Testing:** Controlled, formal. 
- **Café Testing:** Guerrilla, scrappy, real-world context.
- **The Cycle:** Test -> Fix Low Hanging Fruit -> Retest.

# The Prevention Toolkit: Nielsen's Heuristics (1-5)

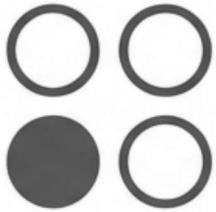
**Visibility of System Status**

**Match Between System & Real World**

**User Control and Freedom**




**Consistency and Standards**



**Error Prevention**

## Visibility of System Status

Users should always know what is going on through appropriate feedback within reasonable time.

## Match Between System & Real World

Speak the user's language with words, phrases, and concepts familiar to the user, not system-oriented terms.

## User Control and Freedom

Provide a clearly marked 'emergency exit' to leave unwanted states without extended dialogue (Undo/Redo).

## Consistency and Standards

Users should not have to wonder whether different words, situations, or actions mean the same thing.

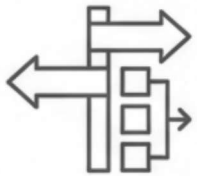
## Error Prevention

Good design prevents problems from occurring in the first place.

Clinical Swiss Editorial

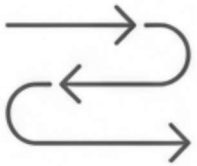
# The Prevention Toolkit: Nielsen's Heuristics (6-10)

### Recognition Rather than Recall



Minimize the user's memory load by making objects, actions, and options visible.

### Flexibility and Efficiency of Use



Accelerators—unseen by the novice—may often speed up the interaction for the expert user.

### Aesthetic and Minimalist Design

Dialogues should not contain information which is irrelevant or rarely needed.

### Help Users Recognize, Diagnose, and Recover

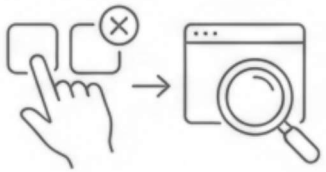
Error messages should be expressed in plain language (no codes), precisely indicate the problem, and suggest a solution.

### Help and Documentation

Any help documentation should be easy to search, focused on the user's task, and not too large.

# Summary: From Diagnosis to Resolution

## Diagnosis

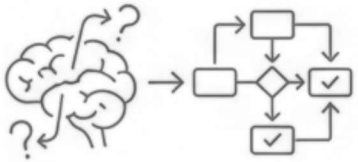


**Slip**

"I didn't mean to do that."

(Action Error)

→ **Fix:** Check environmental distractions and interface cues.



**Mistake**

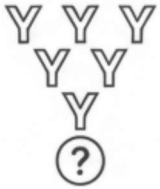
"I misunderstood the goal."

(Thinking Error)

→ **Fix:** Check system clarity and feedback.

## Method

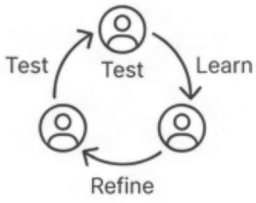
- **5 Whys:**
  Drill down to the root cause.



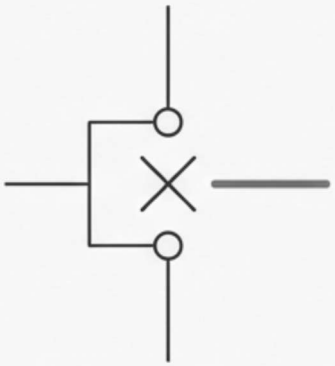
- **Heuristics:**
  Apply Nielsen's 10 principles during design.

- **Usability Testing:**
  Observe real users with scenario-based tasks.

- **Iterate:**
  Test early, test often, with small groups (3 users).

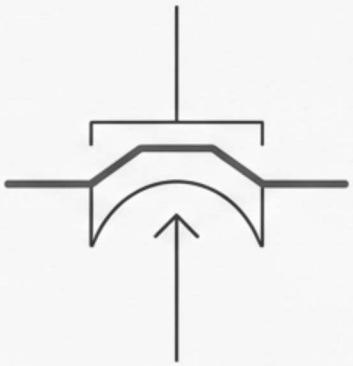


# Problem vs. Opportunity



PROBLEM: An issue preventing the achievement of goals.

**OPPORTUNITY:** An initiative that assists in reaching goals if implemented appropriately.



Every error you diagnose is an opportunity to refine the process. Don’t just fix the slip; improve the system that allowed it.