Zusammenfassung Human-Computer Interaction

Inhalt

[Human-centered design & interviewing 3](#_Toc534292749)

[Human-Centered Design 4](#_Toc534292750)

[Interviewing 4](#_Toc534292751)

[Contextual inquiry 4](#_Toc534292752)

[Who to interview? 5](#_Toc534292753)

[Problematic interview questions 5](#_Toc534292754)

[Steps of an interview 6](#_Toc534292755)

[Using video in an interview 6](#_Toc534292756)

[Analyzing qualitative data 7](#_Toc534292757)

[Transcription 7](#_Toc534292758)

[Interpretation 8](#_Toc534292759)

[Coding 8](#_Toc534292760)

[Interpretation session 8](#_Toc534292761)

[Summarizing 8](#_Toc534292762)

[Theories about qualitative data analysis 9](#_Toc534292763)

[Ideation and prototyping 10](#_Toc534292764)

[Ideation 10](#_Toc534292765)

[Prototyping 12](#_Toc534292766)

[Storyboard 12](#_Toc534292767)

[Paper prototyping 13](#_Toc534292768)

[Slides prototyping 13](#_Toc534292769)

[Prototyping software 13](#_Toc534292770)

[Using video for prototyping 14](#_Toc534292771)

[Writing the actual software as a prototype 14](#_Toc534292772)

[Making hardware prototypes 14](#_Toc534292773)

[Dimensions of a prototype 14](#_Toc534292774)

[Theories behind ideation & prototyping 15](#_Toc534292775)

[Testing 16](#_Toc534292776)

[Usability Tests 16](#_Toc534292777)

[Expert-based tests 16](#_Toc534292778)

[Heuristic evaluation 16](#_Toc534292779)

[User-based tests 17](#_Toc534292780)

[Automated tests 18](#_Toc534292781)

[Design principles 18](#_Toc534292782)

[Model human processing 18](#_Toc534292783)

[Time 18](#_Toc534292784)

[Errors 18](#_Toc534292785)

[Visual perception 18](#_Toc534292786)

[Interaction styles 18](#_Toc534292787)

[Input devices and interaction techniques 18](#_Toc534292788)

[Survey and experimental research 18](#_Toc534292789)

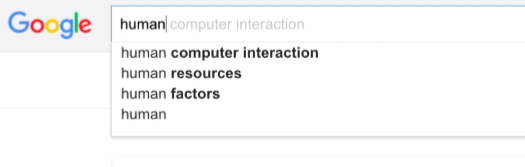
# Human-centered design & interviewing

**HCI…**

* **Saves lives**

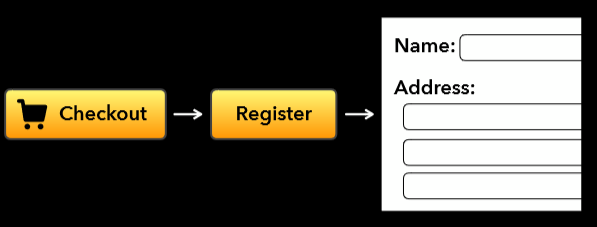


* **Saves time**

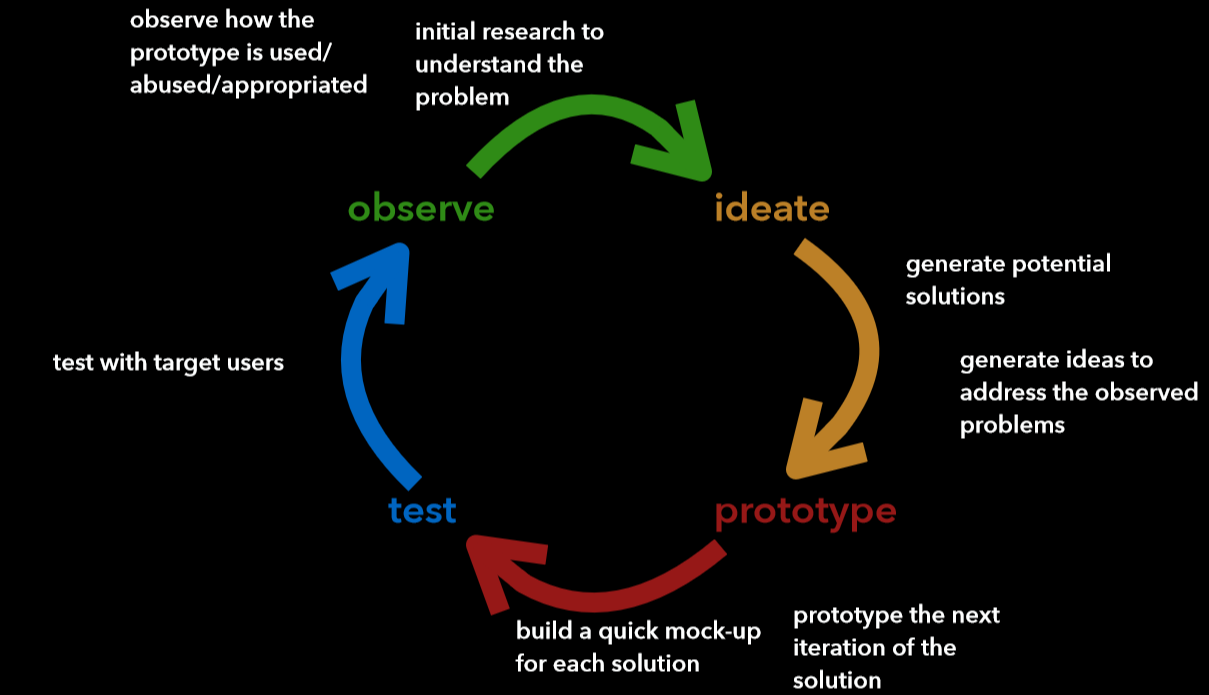


1 second improved per search \* 3.5 Billion Searches per day = 110.915 man-years per day.

* **Sells**



## Human-Centered Design



**Steps of Human-Centered Design**

1. Observe potential user groups to find a problem
2. Generate potential solutions to the problem
3. Prototype different solutions and build different mockups
4. Test the solutions with target users
5. Observe the use of the prototype by the test users and adjust it

**The HCD process ensures that…**

* People’s needs are met
* The resulting product is understandable and usable
* It accomplishes the desired task
* The experience of use is positive and enjoyable

## Interviewing

Interview can be used to discover real problems or to dig deeper into any issues that arise based on the responses from a previous round of interviews.

### Contextual inquiry

Interviews in HCD should be done with contextual inquiry. That means one should go to the user, watch them do the activities one cares about, and talk with them about what they’re doing right then.

The questions should be about what is happening right then and what the interviewer can see. The focus should be on what really happened. Contextual inquiry is grounded on real-life and not on the interviewer’s preconceived ideas.

**Principles of contextual inquiry**

1. **Context**

Gather concrete data about the ongoing experience.

1. **Partnership**

The Interviewer creates a partnership with the users to understand their work. While the user is engrossed in the activity, the interviewer watches the details, patterns and reasons of the user’s activity. This is important because the user might not notice the patterns while they are focusing on the activity. When the interviewer spots a pattern, they interrupt the user to talk about it.

1. **Interpretation**

A good product is built on the designer’s interpretation of the observed facts.

1. **Focus**

If one works in a team, the team’s shared focus and those of the individuals of the group must be defined.

### Who to interview?

1. **Users**: different types of users (current/potential users, novice/expert users)
2. **Stakeholders**: People who don’t use the system but are affected by it
3. **Key informants**: People who know how to do the necessary things. The key informants are repeatedly consulted and provide important insights.

### Problematic interview questions

* **Close-ended questions**: Yes-/No-questions. *i.e. “Do you like Google Search?”*
* **Leading questions**: A question that suggests the answer the interviewer expects or wants to get. *i.e. “Do you like our new design of Google Search?”*
* **Complex questions**: Multiple aspects that must be thought about and could be too complex for the interviewee. *i.e. “What are the strengths and weaknesses of Google Search vs. Bing Search?”*
* **Double barreled questions**: Two questions in one sentence. *i.e. “Should Google Search provide more results and more detailed results?”*
* **Negative questions**: *i.e. “Don’t you like Google search?”*
* **Asking the users to design**: *i.e. “What is a feature that you would like to have from Google Search?”*

### Steps of an interview

1. **Prepare**: Prepare the interview guideline, make sure the recording equipment works, test the interview guideline
2. **Build rapport**
3. **Introduction**: tell the high-level research goals to the interviewee, obtain recording permissions
4. **Have a conversation**
5. **Debrief**: Summarize what you learned, tell the detailed research goals, say thank you
6. **Brain dump**: type down everything one remembered

### Using video in an interview

* Explain how the video is used
* Ask for permission
* Shoot the title page
* Show example of recording
* Record the usage, not the user
* Offer to show the video + cuts

# Analyzing qualitative data

The process of analyzing the data consists of three steps: Transcription, Interpretation & Summarization.

## Transcription

**Purpose**: Providing a representation of data that is easy to navigate and analyze, exposing yourself with the responses with more distance

**Types of transcription**

* Full transcription: transcribe everything that is done and said during the interview
* Partial transcription: take field notes on when interesting parts took place and only transcribe these. Note that there is already a certain amount of judging taking place during the interview when deciding on what is important, which will influence the partial transcription.

**Data sources**: Audio, video, screen recording, …

**Transcription software**

There are a lot of software/ methods that can help when transcribing.

* A **foot pedal** with which the audio or video recording can be controlled. This saves a lot of time as you don’t have to switch key-bindings on the keyboard to switch between the media-player and the software used to transcribe.
* **Adding timestamps**: makes it easier to find additional context when looking at the transcript and wanting to find said passage in the recording
* **Playback speed control (0.5x-2x)**: Can be used if the interviewee talks too slow/fast to type down.
* **Support audio and video formats**: a good transcription software should be able to play different types of audio and video formats, so that the user is not limited in what he can use.

## Interpretation

The interpretation consists of coding and an interpretation session.

### Coding

**Purpose**: To reduce the complexity of free-form of the dataset into a finite set of codes for further analysis.

**Coding procedure**

1. Read through the transcript in one first pass (note down the first impression of the data)
2. For each chunk of the transcript
   1. Read it
   2. Assign one or multiple codes that represent that chunk
3. Iteratively refine the set of code and how they are assigned
   1. At the beginning you may need to go back and change the assigned codes in previous chunks
   2. After coding 20-30% of data, the code set usually stabilizes
   3. Discuss with other coders to refine the understanding of codes
   4. Take notes of ideas and questions that emerges

**Assigning codes**

The code lists can be determined either by using pre-defined codes based on existing theoretical frameworks or taxonomy (i.e. “Normans” taxonomy of errors) or through open coding. The latter means finding a pattern in the data and designating a code for it.

**Asking questions while coding**: While coding one should ask sensitizing questions to better understand the meaning and theoretical questions to make connections between concepts and categories.

### Interpretation session

There are different methods that can be used to interpret the data that is now present.

* Building a shared understanding of the data through a partial reenactment of the interview session
* Interviewer recaps what happened
* Asking for clarification questions if things are unclear
* Note down any new insights or ideas
* Interpretation session can be done together with the coding step

## Summarizing

The part of the summarizing is best done with an **affinity diagram**. Affinity diagramming is a bottom-up analysis process. An affinity diagram is done by sorting similar quotes into groups. This gives a general overview over multiple interviews.

## Theories about qualitative data analysis

**Research philosophies**

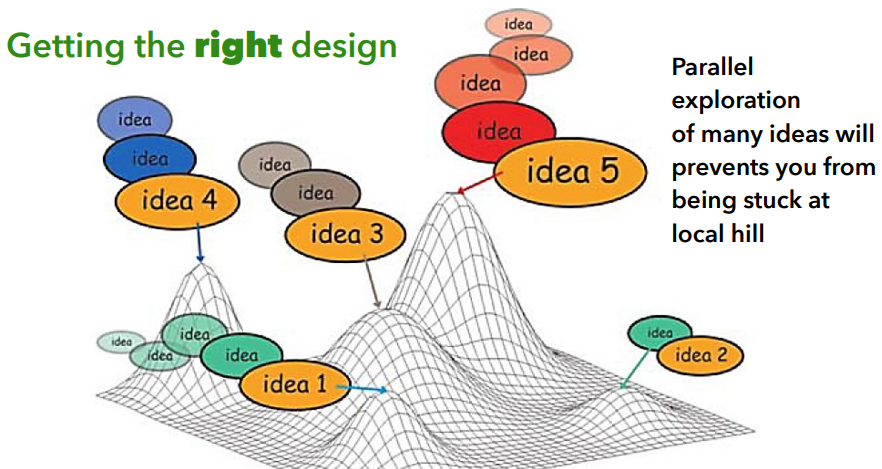
* **Positivism**: assume that there is an objective truth and this truth can be measured by human sense together with assistance of scientific instruments.
* **Interpretativism**: assumes that theoretical beliefs of researchers cannot fully be removed from their inquiry. This has the benefit that rich nuances that are unanticipated can be captured. This is very useful to generate research questions.

**Why do we need these methods?**

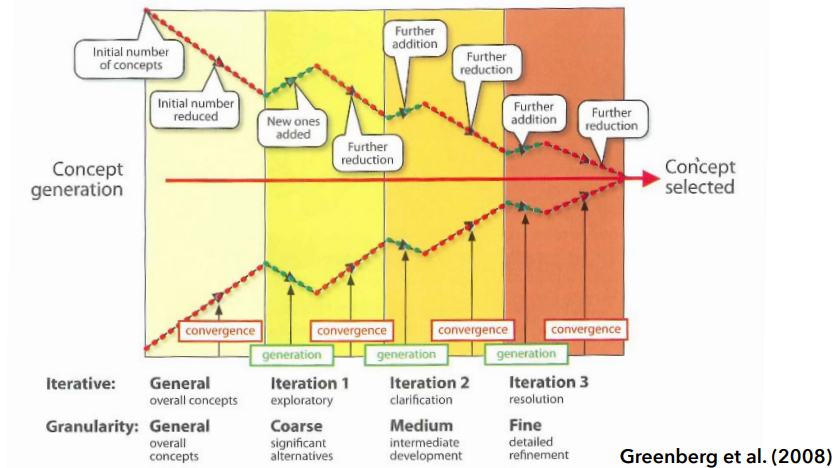
* **Curse of knowledge**: The difficulty of better-informed people to think about issues from the perspective of lesser-informed people.
* **Confirmation bias**: The tendency to search for, interpret, focus on and remember information in a way that confirms one’s preconceptions.
* **Primacy effect** and **recency effect**: The first event and the last event are easier to recall than others.

# Ideation and prototyping

## Ideation



1: Generating different ideas that may vary in how optimal they are.



2: During the process of finding a final solution there can be more ideas coming up as new problems may arise.

**Principles of brainstorming**

* Go for quantity
* Put criticism on hold, no idea is wrong
* Welcome wild ideas
* Combine and improve ideas

## Prototyping

Prototyping is a concrete, cheap and low-tech implementation of the ideas. It can free up one’s head for more ideas and aids the communication in the team and with the users. Prototypes can reveal first mistakes and problems, which makes them very important.

### Storyboard

A storyboard is an illustration of the whole user experience. It shows the context beyond the user surface and is useful for communicating the overall interaction flow.

**Questions a storyboard should answer:**

* Where does the interaction take place?
* What is the problem?
* What is the task that people are trying to do?
* Which people are present and what are their actions?
* What kind of objects or digital devices do they use?
* What is the possible input and output for each digital system?
* How do the actions of people and/or devices solve the problem?

**Camera angles**



**Strengths**

* Includes the context of the use
* Tells one coherent story of interactions, including the user’s motivation prior to the interaction as well as what the user does after the interaction

**Weaknesses**

* Can show only one path of the interaction
* Static and difficult to change on-the-fly (i.e. while discussing with customers)

**When to use**: Early in the design process to check your understanding of the current work practice with users. Or to make ideas about the usage scenario more concrete

### Paper prototyping

**Strengths**

* Very cheap to create
* Can be changed on-the-fly
* Elicits high-level feedback without nitpicking details

**Weaknesses**

* Interaction must be driven by the designer in a face-to-face session

**When to use**: In brainstorming session or in early tests with users

### Slides prototyping

**Strengths**

* Allows reusing of user-interfaces
* Users can interact with the prototype by themselves
* Can be distributed digitally

**Weaknesses**

* Detailed look-and-feel may mislead the users to focus their comments on visual design

**When to use**: Mid-project, when the high-level interaction is already clear and focus on testing out the visual design of the software.

### Prototyping software

**Strengths**

* Can quickly use and adjust standard user interface components
* Users can interact with the prototype by themselves
* Can be distributed digitally

**Weaknesses**

* Being restricted by standard user interface components
* Cannot be changed on-the-fly (comparing with paper prototyping)

**When to use**: In the project that is restricted to standard user interface components, this can be use early in the design process.

### Using video for prototyping

**Strengths**

* Allows prototyping user interfaces in unconventional form factors
* Shows one coherent interaction story
* Shows the context of use

**Weaknesses**

* Not interactive
* Cannot be changed on-the-fly

**When to use**: For showing overall interaction concepts, especially in non-conventional platforms that are difficult to prototype by other means.

### Writing the actual software as a prototype

**Strengths**

* Allows users to interact with the prototype in a fine granularity
* Parts of the implementation can be used in real software

**Weaknesses**

* Expensive to create
* Testers may focus their feedback on nitpick visual design details
* Testers may be afraid to criticize the prototype

**When to use**: Late in the project after the design has been refined by other low-cost techniques.

### Making hardware prototypes

Creating the form of the object with a different material, that doesn’t have all the functions.

### Dimensions of a prototype

**Look**

* Visual fidelity of the prototype (i.e. font, color, graphics)
* More polished look doesn’t have to be better
* Rough prototypes tend to encourage users and stakeholders to respond in more creative manner

**Breadth**

* How many % of product’s functionality is in the prototype?
* A prototype needs sufficient breadth to cover the test task, not more

**Depth**

* How fleshed out are the functionalities?
* Greater depth leads to more exploration that the user can do, better chance of catching usability issues, but takes more time in preparation of the prototype

**Interaction**

* How much do input and output of the prototype reflect those of the product?

## Theories behind ideation & prototyping

**Parallel exploration**

Exploring in different directions makes sure that ideas, that could be implemented in different prototypes, aren’t ignored.

**Bounded rationality** (Graphic 1)

While solving problems, people make rational decisions, but they lack the ability of knowing all the potential solutions. The rationality of the people is limited by the tractability of the decision problem, the cognitive limitations of their minds and the available time to make the decision.

The Design-process is therefore an attempt to change a situation from something suboptimal to something optimal within their bounds of rationality.

**Lateral thinking** (Graphic 1)

Theory that there are two types of thinking: vertical & lateral thinking.

**Vertical thinking**: solving a problem with a logical process and in the manner, that is the most expected and most logical. This often leads to a predictable outcome.

**Lateral thinking**: purposefully looking at a situation from an unexpected perspective. This type of thinking usually has a playful attitude and may be driven by provocation. The attempt is to surprise, shock, or disrupt the situation with an unexpected outcome.

**Doing and thinking** (Graphic 2)

Designers see constraints around the problem they are solving. Now they generate a solution that may create new constraints that have to be solved. This can generate a circle of solving a problem and generating new ones.

# Testing

## Usability Tests

**Definition**: Usability testing involves representative users attempting representative tasks in representative environments, on early prototypes or working versions of computer interfaces.

This has the goal of finding interface flaws and features that work well.

The purposes of usability tests are separated depending on when in the creation process they are used.

**Formative tests** (early in the design process, usually on prototypes): used to explore early design concepts and discover problems. This usually gets qualitative results, as it involves only a few users.

**Summative tests** (used in the late stages): used to evaluate the effectiveness of the specific design choices. These tests generate quantitative results and involve a sizeable number of users or task repetitions. Quantitative tests measure how many tasks are performed correctly and how much time was needed for them.

**Types of usability tests**: Expert-based tests, user-based tests and automated tests.

## Expert-based tests

Structured inspection by UI experts.

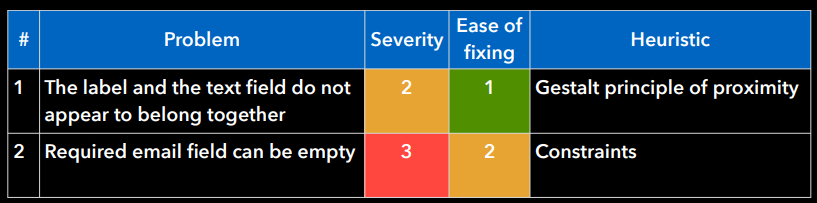
### Heuristic evaluation

**Definition**: Evaluating a UI against a set of design principles with the goal of finding fine-grained usability problems at low cost (before involving users).

The evaluation is done by an HCI or a domain expert. The best evaluators are both HCI & domain experts. This method is best applied to prototypes that are mature in the look and the depth dimensions.

**Procedure**

1. Select a set of design principles as heuristic
2. First-pass: freely use the UI to get an overall impression
3. Second-pass: focus on specific elements & identify issues and take notes of issues
4. End: Classify issues based on severity
5. Discuss with the development team about the ease of fixing them



## User-based tests

**Steps of a user-based test**

1. Select representative users
2. Select a location
3. Decide what task the users should perform
4. Decide what type of data to collect
5. Briefing before the test session
   1. Clarify the purpose “testing the interface, not the user”
   2. Obtain permission to record data
6. Test session
7. Debriefing

**How many users?:** Depends on how large and how complex the system is, how many problems there are and where/which order the are in (a small problem may block the tester from seeing a big one). One should use as many users as possible.

**Locations**

*Lab*: Can limit disturbances and can setup extensive recording equipment. This is costly for the users (can lead to low number of users) and it may be inappropriate for users with disabilities.

*User’s workplace or home*: Users feel more comfortable and less hassle and represents a more realistic context of use. It is more time-consuming as it requires a lot of travel and setting up/ tearing down recording equipment.

*Remote*: Makes it easy to access a large number of participants but makes it difficult to pick-up nonverbal and interpersonal cues and to provide instructions when things go wrong.

**Requirements**: Make sure the task list is as clear as possible and make sure to protect the user’s privacy.

**Interventions**: Decide in advance what to do when there is an interface barrier that does not allow the participant to continue in the interface.

**Example:**

**Think-aloud test**: Ask the user to say what they think out loud during the interactions. This has the negative effect that the interaction may be unnatural and slows down the usage.

**Retrospective think-aloud**: Videotape the interaction and ask users to think-aloud during the replay. This has the negative effect of the imperfect reconstruction of memory.

**Constructive interaction**: One experienced user teaches a new user to use the system. The social dynamics might distort the result.

**Example: Wizard of Oz**

Assessing interactions for technologically sophisticated product before implementing it. The system is controlled by a human.

**A/B Testing**: Parallel-Testing

## Automated tests

**Example: AXE**

AXE is an accessibility testing tool for HTML-based user interfaces that is available as a plug-in for web browsers. It evaluates the website against 72 accessibility rules.

# Design principles

## Conceptual model

A good **conceptual model** in the head of the user leads to more discoverability. The user can **discover** what the application does, how it works and what operations are possible.

A good conceptual model comes from the five design principles **affordance**, **signifier**, **feedback**, **mapping** and **constraint**.

The **user’s conceptual model** is a highly simplified explanation of how a product works. It goes into as many details as is necessary to use the product and does not contain any unnecessary details. Different users have different models and each individual model evolves over time as he uses (or forgets how to use) the product.

The user’s simplified model is based on several assumptions. If the application doesn’t satisfy these assumptions, then the model breaks and problems while using the product come up.

It is important to remember that the user’s model is not the same as the designer’s model. Successful communication of conceptual models allows users to predict the effect of their actions.

## System image

The **system image** is what the user perceives about the system. The system image is mediated by the *context* that the product is used in. *Additional materials* (i.e. training course, manuals, …) may help communicate the system image. A well-designed product should not expect users to read or see external materials.

An incoherent, incomplete and/or contradicting system image can lead to an erroneous conceptual model which will then cause usability problems.

## Affordance

**Definition**: Affordance is a relationship between a physical object and a person that encourages certain actions. Certain properties of an object and certain capabilities of a person can afford a certain action. Affordances can change according to the user’s ability.

**Example**: A chair has the correct properties for a human being to be able to sit on it. It doesn’t on the other hand have the correct shape and size to be used to cut a piece of paper.

Some affordances may be invisible or ambiguous (mehrdeutig) so that a signifier is added to indicate affordance.

## Signifier

**Definition**: A signifier is any perceivable indicator (visual or audible) that communicates appropriate behavior to a person. It is a mean of signaling the presence of the corresponding affordance.

Some signifiers may be redundant or may contradict the affordance. Bad signifiers can mislead the users, which can cause usability problems.

When the signifiers contradict with affordance, the affordance usually wins. Digital products need careful signifier design because the physical affordance stays the same.

## Feedback

Feedback are ways a product communicates the results of an action to the user. Sometimes this feedback is redundant, but it can still be useful as backups or for quicker user responses.

Feedback can be:

* Immediate / delayed
* Informative / ambiguous
* Consistent / misleading
* Ambient / obvious / obtrusive / distracting

## Mapping

Mapping is the relationship between the controllers (input) and the thing or actions that are being controlled (output/feedback).

Mapping can be done by either spatial correspondence or cultural convention. Especially in the second one the cultural conventions of the different areas that the product is for must be looked at and the product changed accordingly.

## Constraint

There are two types of constraints: physical and cultural/semantic/logical constraints.

**Physical constraints** limit what *can* be done.

**Cultural** / **semantic** / **logical** **constraints** limit what *should* be done.

**Forcing functions**

Sometimes it is necessary that the user is forced to fulfill a desired behavior. This can be enforced by disrupting efficient or automatized performance of a task. This has the result that the user is slowed down, and his attention is brought to the necessary task.

There are three types of ways to force the user to do a desired behavior:

* Interlock
* Lock-in
* Lock-out

**Interlock** forces the user to perform actions in a specific sequence.

**Lock-in** forces the user to continue the current activity (preventing from stopping the task prematurely)

**Lock-out** forces the user to stop the current activity (preventing from entering a dangerous state)

Using constraints too frequently could create a bad habit.

## Using the design principles

Observe how existing user interfaces apply or neglect design principles, and their consequence on users’ conceptual models

Ideate by using these principles to come up with design characteristics that foster preferable conceptual models

Prototype these characteristics to test their effectiveness with users.

# Model human processing

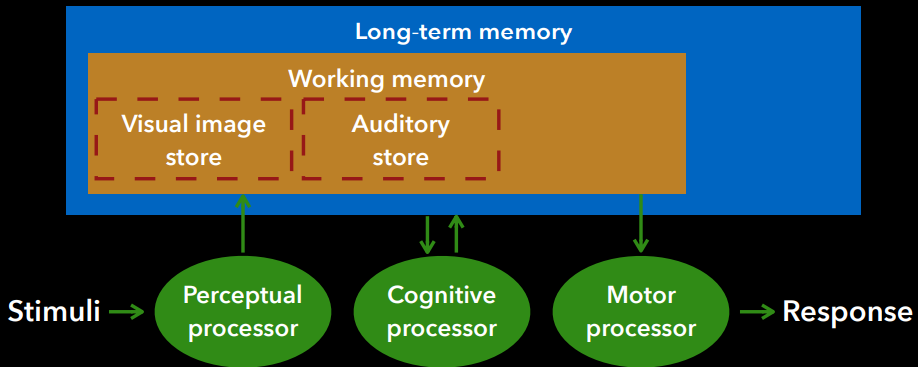
Model human processing can be used to predict the time it takes for the users to react to information and how much information they can store.

## Psychology of HCI

Model human responses based on the information from the stimuli is useful for predicting which cognitive processes are involved and how long a user will take to carry out simple tasks.

There are three processors:

* Perceptual processor
* Cognitive processor
* Motor processor



## Eye movement

**Saccades**: series of jumps made by eyes that last 20-35ms but vary by distance moved.

**Fixations**: periods that they eyes are relatively still that last 200-250ms.

We perceive the world in discrete pieces and our brain knits these pieces together.

## Time estimates of the processors

* **Perceptual processor**: avg: 100ms (range: 50-200ms)🡪 Two events occurring within 100ms are combined as one.
* **Cognitive processor**: avg: 70ms (range: 25-170ms)
* **Motor processor**: avg: 70ms (range: 30-100ms)

**Motor control**

*Open loop control*: signal from the motor processor executes the muscles in one-way manner. (70ms)

*Closed-loop control*: the motor signal is controlled by the perceived feedback. (240ms)

## Memory capacity

**Working memory**

**Chunking**: grouping individual pieces of information into larger units (chunks)

**Capacity**: 7 (+/- 2) chunks, recent research says 4 chunks

**Half life**: the duration in which there is a 50% probability of forgetting the entire chunk

* 1 chunk: 73s (after 73s one can either recall the chunk entirely or forget it entirely)
* 3 chunks: 7s

**Long-term memory**

Infinite capacity and half life; Stored as a network of associations. In order for things to be stored in the long-term memory there needs to be repetitions.

One recalls information by reconstruction. It’s an imperfect reconstruction. Random recalls are slow, when relevant concepts are activated the recall is faster.

**Limitations of the Model Human Processor**

**Assumptions**: Only perceived information & those inside the head matter. The user’s full attention is on the product.

**Real world**: People offload memory & computation to external objects, environment and other people. There are distractions in the real world.

The Model should be used as an estimate of the best case.

## Knowledge

Knowledge of the user consists of the one the world provides and the one that is in the users head. These two combined determine how the user interacts with the product.

# Time

# Errors

# Visual perception

# Interaction styles

# Input devices and interaction techniques

# Survey and experimental research