Interação Pessoa-Máquina

Teresa Romão
tir_at_fct.unl.pt
DI/FCT/UNL

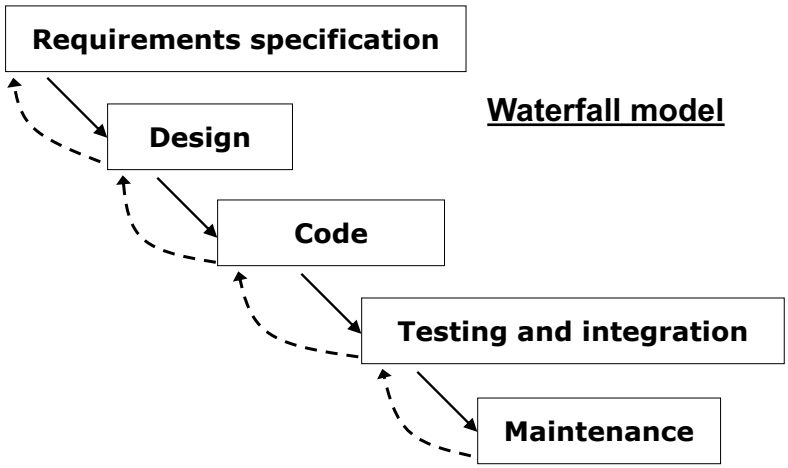
2022/2023

The Design Process

Software life cycle

- 60´s e 70´s – software crisis
  - **Code now, fix it later...** – process needs some structure
- Waterfall model – attempt to organize the development process – **“think first and code second”**.

Software life cycle



Waterfall model

Software life cycle

- Requirements specification
  - designer and customer try to establish **what** should be implemented.
  - involves collecting information concerning the users and the working environment or domain in which the final product will function.
  - working domain aspects include:
    - function that the software should perform
    - details about the environment in which it must operate
    - people it will potentially affect
    - relationship with other existing products which it is updating or replacing.
  - Use a language that both users and developers understand.

User and task analysis

Software life cycle

- Design
  - Establish how the system should be implemented, in order to fulfil the requirements specification.
  - Produce an appropriate detailed description of interface design, in order to be implemented in a programming language.
- Code
  - implementation and testing of the individual modules in a programming language.
- Testing and integration
  - Includes further testing and acceptance tests with users to ensure the system meets the requirements.

Software life cycle

- Maintenance
  - after the system release, maintenance lasts until a complete redesigned new version is produced or the system is phase out completely.
  - comprises:
    - correction of errors discovered after release
    - system review to satisfy requirements that have not been identified earlier
    - ... so, there is feedback between this activity and all others.

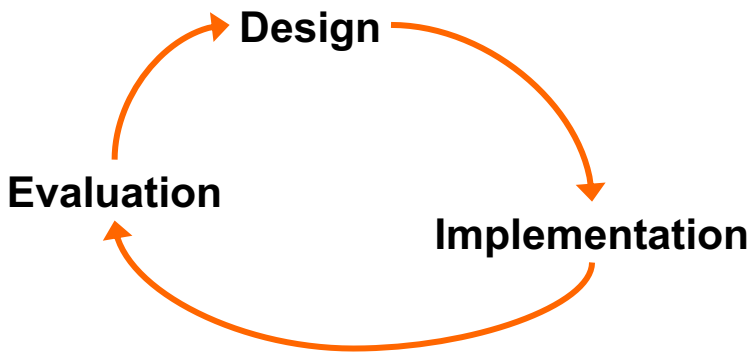
Waterfall model

- Waterfall model is not appropriate for interactive systems interface design:
  - users only participate in the requirements specification and testing.
  - Late detection of errors causes expensive and long lasting rectifications.
  - no support for really iterative processes.

Waterfall Model

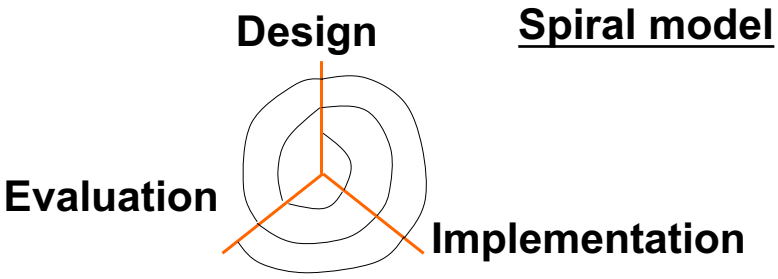
- Wrong: follow the waterfall model, produce a bad interface and release the system.
  - Each iteration corresponds to a version of the system
    - Errors detected during evaluation are corrected in the next version.
  - Clients are used to evaluate the system's usability:
    - If they don't like, they don't buy the next version!

Iterative design



worst case in the waterfall model?

Iterative design



Iterative design

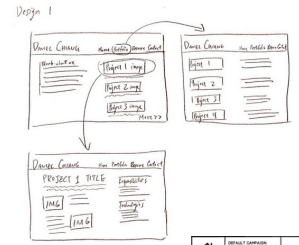
- Spiral model
  - Several iteration
    - Cost, accuracy and correctness increase in each iteration.
  - First iteration may be done in paper: low cost, ...and far from what it will look like.

Iterative design

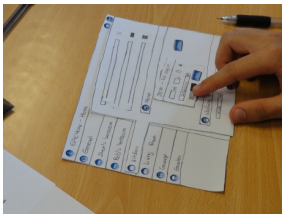
- First iterations – low cost prototypes
  - Parallel design: development and testing of several prototypes to explore multiple alternatives.
- Subsequent iterations (after eliminating the highest risks)
  - creation of more elaborated prototypes
- Every prototype is evaluated
  - users are involved in every iteration
- More iterations → better interfaces
- Only better interfaces survive and reach the market.

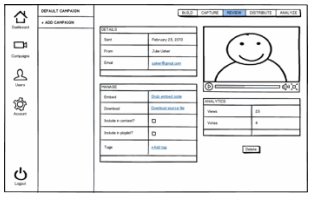
Iterative design

Sketches



Paper prototype





Computer Mockup

User and task analysis

User-Centered Design

- Science finds, Industry applies, Man conforms - Slogan of World Fair Chicago 1933.
- People propose, Science studies, Technology conforms - Slogan de Donald Norman.

User-Centered Design

- The design is based upon users':
  - needs,
  - abilities,
  - context,
  - work,
  - task.

User and task analysis

User-centered design – first steps

- User analysis: Who are the users?
- Task analysis: What does the user need to do?

User and task analysis

- Collect data about the users and tasks (characteristics and needs) ...
- ... represent the data, in order to make interpretation easier and guide the design.

User and task analysis

	User and task analysis	System analysis
Main focus	People (users)	Computer
Goal	Collect data for UI design	Collect data for the software and data structure design
Results	List of tasks and users' profiles	Functional specification and architecture

Based on Gonçalves, 2017

Know the user

- Identify characteristics of the target user population:
  - age, gender, ethnicity
  - Education
  - Physical abilities
  - General computer experience
  - Domain experience
  - Application experience
  - Work environment and social context
  - Communication patterns

Know the user

- Many applications have several kinds of users.
- Need to analyse all kinds of users.
- Example: Self-service supermarket checkout
  - Clients
  - Assistants
- Describe what the real users are rather than what you want them to be.

Know the user

- Techniques
  - Questionnaires
  - Interview
  - Observation
- Obstacles
  - Some users are hard to reach
  - Users speak another language

Example: Self-service supermarket checkout

- Who are the users?
  - Shoppers
  - Age: 10-80 years
  - Different physical abilities: height, strength,...
  - No training: arrive and use
  - Knowledge of food, but not about supermarket stock management.
  - Can ask each other for help.
- Main user classes:
  - Clients
  - Assistants

Users description

Table 5 ATM user groups (adapted from Stone, 2001)

User characteristic	ATM customer characteristics, by group		
	Teens/Young adults	*Young adults to middle age*	*Middle age to senior citizens*
Age	12 to 25.	25 to 50.	50 to 80+.
Sex	Both male and female.	Both male and female.	Both male and female.
Physical limitations	May be fully able-bodied, or may have some physical limitations in relation to, for example, hearing or sight. Will be of varying heights.	May be fully able-bodied, or may have some physical limitations in relation to, for example, hearing or sight. Will be of varying heights.	May be fully able-bodied, or may have some physical limitations in relation to, for example, hearing or sight; mobility, or use of hands. Will be of varying heights.
Educational background	May have minimal or no educational qualifications.	May have only minimal educational qualifications.	May have only minimal educational qualifications.
Computer/IT use.	Probably have some prior experience of computer or IT use.	May have little or no prior experience of computer or IT use.	May have little or no prior experience of computer or IT use.
Motivation	Probably very motivated to use the ATM, especially in relation to their banking habits.	Could be very motivated to use the ATM, especially if they can do their banking quickly and avoid queuing in a bank.	Could be very motivated to use the ATM, but would probably prefer to stand in a queue in the bank.
Attitude	Attitudes to use may vary, depending on the services the automated teller offers and the reliability of the technology itself.	Attitudes to use may vary, depending on the services the automated teller offers and the reliability of the technology itself.	Attitudes to use may vary, depending on the services the automated teller offers and the reliability of the technology itself.

Task analysis

- Identify the users' goals and study the way users perform their jobs.
  - What users do?
  - Why they do it?
  - How they do it?
  - What they must know?
  - What tools they use?
- The new system/interface may change the current processing ("How?")
- Understanding "how" and "why" allows for a deeper knowledge about the tasks.

Task analysis

- Study of the way people perform tasks with existing systems.
- Users can help you learn:
  - What is related to their job performance
  - What instruments do they use
  - What they actually do
- You show the possibilities of technology
  - Build a relationship and convey an idea of what is possible
  - Users can comment if ideas make sense

Task analysis

- The general method for Task Analysis is:
  - Observe / Ask
  - collect unstructured lists of words and actions
  - organize using notation or diagrams

Task analysis

- Identify the individual tasks the system should perform
- Each task represent a goal (what?, not how?)
- Top-down approach: start with the overall goal of the system and decompose it hierarchically into tasks
- Overall goal: self-service checkout
  - Tasks:
    - Register products
    - Pack
    - Pay

Task analysis

- What needs to be done?
  - Goal
- What must be done to make it possible?
  - Pre-conditions
    - Tasks on which this task depends
    - Information the user needs to know
- What steps are involved in doing the task?
  - Sub-tasks
  - Sub-tasks may be decomposed recursively.

Example: Self-service supermarket checkout

- Goal
  - Register products
- Pre-conditions
  - All the desired products are in the cart
- Sub-tasks
  - Register pre-packaged product
  - Register loose product

Example: Self-service supermarket checkout

- Where is the task performed?
  - Supermarket exit, standing up
- How often is the task performed?
  - once a week
- What are its time or resource constraints?
  - 3 minutes
- How is the task learned?
  - Try it
  - Watching others
  - Assistant demo
- What can go wrong? (exceptions, errors, emergencies)
  - Bar code is missing or unreadable
- Who else is involved in the task?

Task analysis

- Collecting information techniques:
  - Direct observation of users performing tasks
  - Interviews with users
  - Contextual inquiry
  - Participatory design
  - Expert advice
  - Documentation analysis
  - Logging

Example: Self-service supermarket checkout

- Observe store cashiers checking out products to understand the supermarket checkout task.
- Interview shoppers to better understand their goals.

Observation

- Real environment (animals in a zoo) versus controlled environment (video).
- Passive (watch and hear - record) versus active (ask)
- Encourage the user to think aloud
- Capture what the users say and do
- Describe the observation to someone who have never witnessed the task

Observation

- Questions to ask:
  - Why do you do this? (goal)
  - How do you do this? (sub-task)
  - What must be done before doing this? (sequence, pre-conditions)
  - What fails when you do this?(exceptions)
- Look for the weaknesses in the current system
  - Goals not accomplished, wasted time, user irritation
- At the end: “What else should I ask?”

Observation

- Dangers (direct observation):
  - Duplicate bad existing procedures
  - Failing to capture good existing procedures

  Know: Why users do what they do (not just what they do!)

Interviews with the users

- **Structured**
  - Follow an interview plan
  - Be specific
  - Efficient
  - Needs preparation
- **Non structured**
  - Open talk
  - Inefficient
- **Semi-structured**
  - Start with a plan of questions and end up in an open talk
  - Balanced
  - Often appropriate
- Record interviews (when appropriate and with consent)

Interviews with the users

- **Plan your questions:**
  - How do you perform task X?
  - Why do you perform task X?
  - When (what conditions) do you perform task X?
  - What do you do before you perform task x?
  - What information do you need for…?
  - Who are the persons you need to communicate for …?
  - What do you use for…?
  - What happen after performing…?
  - What is the result of…?
  - What are the consequences of not doing…?

Contextual inquiry

- Contextual inquiry
  - Combines interviewing and observation in the user's actual work environment, discussing actual work products.
  - Fosters strong collaboration between the designers and the user.
  - Be concrete
  - Establish a master-apprentice relationship
    - User shows how and explains
    - Interviewer watches and asks questions.

Participatory design

- Instead of guessing, designers should have access to a pool of representative users.
- Include representative users directly in the design team.

Participatory design

- Periodical refresh of the pool of users who participate in large projects
  - users become less representative as they understand the proposed system structure
- Changing users representative involves spending time explaining the project to new users.

Expert advise

- Experts describe tasks as they should be executed ...
- ...not necessary, how they are actually executed.

Documentation analysis

- Describe how it should be done...
- ...instead of how it is done.
- Try to understand why it is not done “by the book”.

Logging

- Keystrokes / mouse clicks
- Transactions logs
- Location
  - Mobile phones
  - RFID
  - GPS

Study concurrent products

- Search for good and bad ideas:
  - Funcionalities
  - Interaction styles

Task description

- After collecting the information, we need to organize and represent it in order to guide the design
- Even the simplest task may become quite complex to describe
  - For example, sending an email.
    Easy, right?

Task description

- Sending an email:
  - Click New Email button
  - Click inside the “to:” field
  - Type recipient’s email address
  - Click inside the subject field
  - Type the subject of the email
  - Click inside the body field
  - Type email, including a greeting and closing sentence.
  - Add signature
  - Double-check email for correct spelling and grammar
  - Click Send button

Task description

1. Identify the task to analyse
2. Break down the task into subtasks
3. Identify steps in subtasks

Task description

- Sending an email:
  - Click New Email button
  - Enter recipient
    - Click inside the “to:” field
    - Type recipient’s email address
  - Define the subject
    - Click inside the subject field
    - Type the subject of the email
  - Write message
    - Click inside the body field
    - Type email, including a greeting and closing sentence.
    - Add signature
  - Double-check email for correct spelling and grammar
  - Click Send button

Task analysis

- "Hierarchical task analysis"- HTA
  - Hierarchical decomposition of tasks
  - Specification plan describing in what order and under what conditions subtasks are performed.
  - Start point: user goal.

Task analysis

- Hierarchical task analysis (HTA) can be textually or graphically represented.

Hierachical task analysis (HTA)

0. Make a cup of tea
- **1.** Boil water
  - **1.1** Fill kettle
  - **1.2** Put kettle on stove
  - **1.3** Wait for water to boil
  - **1.4** Turn off stove
- **2.** Put tea leaves in pot
- **3.** Pour in boiling water
- **4.** Wait 4/5 minutes
- **5.** Remove tea leaves

Plan 0

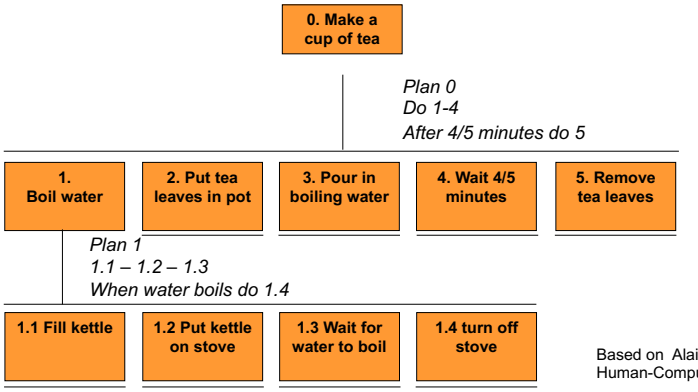
Do 1- 4
After 4/5 minutes do 5

Plan 1

1.1 – 1.2 – 1.3
When water boils do 1.4

Task analysis

- Hierarchical task analysis (HTA) can be textually or graphically represented.



Based on Alain Dix et. al, Human-Computer Interaction, 2004

Task analysis

- After the first approach to the task's description: verify errors and omissions.
- Possible approach: consult an expert.
  - Omission: warm pot.
- Examine sub-tasks
  - 1.4 turn off stove. When was it turn on? Implicitly in 1.2.
- Balance the hierarchy (may not be necessary or desirable!)

Task analysis

0. Make a cup of tea
- **1.** Boil water
  - **1.1** Fill kettle
  - **1.2** Put kettle on stove
  - **1.3 Turn on stove**
  - **1.4** Wait for water to boil
  - **1.5** Turn off stove
- **2. Make pot**
  - **2.1 Warm pot**
  - **2.2** Put tea leaves in pot
- **3** Pour in boiling water
- **4.** Wait 4/5 minutes
- **5.** Remove tea leaves

Plan 0

Do 1- 3
At the same time 2
Then 3-4
After 4/5 minutes do 5

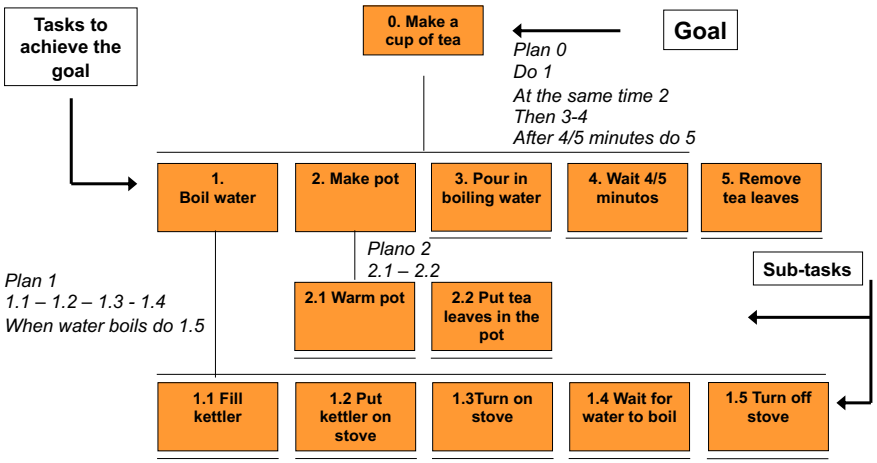
Plan 1

1.1 – 1.2 – 1.3 –1.4
When water boils do 1.5

Plan 2

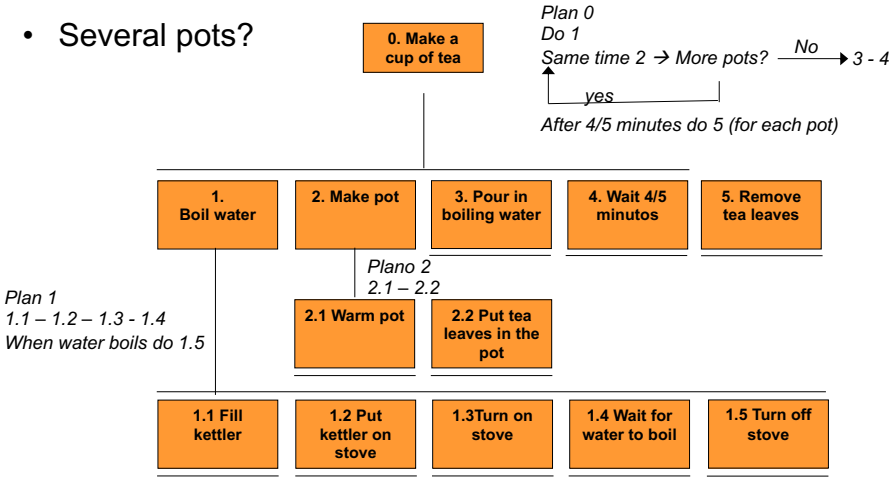
2.1 – 2.2 – 2.3

Task analysis



Task analysis

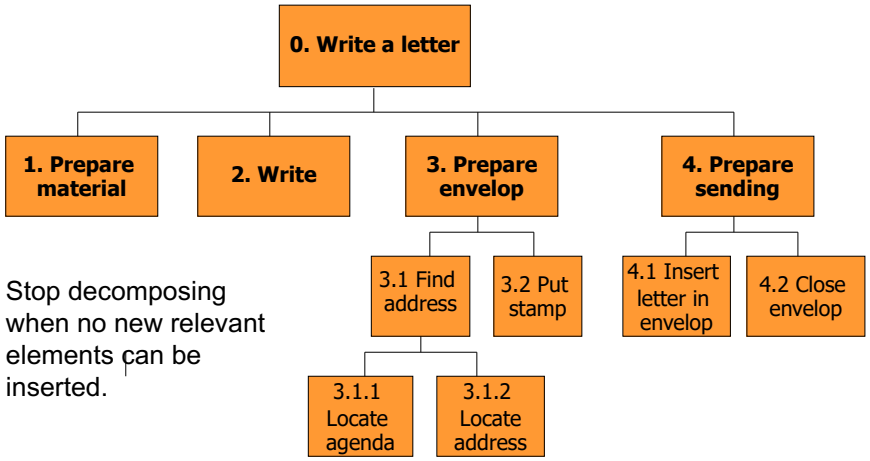
- Several pots?



Task analysis

- Types of tasks
  - Fixed sequence (plan 2)
  - Optional tasks (add sugar as task 6)
  - Waiting for events (plan 0 e 1)
  - Cycles (plan 0)
  - Time-sharing (task 1 and 2 can be done at the same time)

Task analysis



Stop decomposing when no new relevant elements can be inserted.

Task analysis

- User X
  - Gather paper, pen, envelop and stamp
  - Write letter
  - Fill in envelop
  - Stick the stamp
  - Insert letter in envelop
  - Close envelop
- And user Z?

- User Y
  - Gather paper, pen, envelop and stamp
  - Fill in envelop
  - Write letter
  - Insert letter in envelope
  - Stick the stamp
  - Close envelop

Task scenarios

- Based on narratives that describe:
  - Actors
  - Objectives
  - Tools
  - Thoughts/Actions/events (sequence) to achieve the goals

Task scenarios

- Informal description narrative
- Uses the user vocabulary
- Repetitive references to an object or behaviour may suggests its importance or relevance in the context.
- Scenarios describing the actual situation may help to define new scenarios.
- Provide test cases

Task scenarios

- Task scenario describing the use of a library catalogue:
  - "Say I want to find a book by George Jeffries. I don't remember the title but I know it was published before 1995. I go to the catalog and enter my user password. I don't understand why I have to do this, since I can't get into the library to use the catalog without passing through security gates. However, once my password has been confirmed, I am given a choice of searching by author or by date, but not the combination of author and date. I tend to choose the author option because the date search usually identifies too many entries. After about 30 seconds the catalog returns saying that there are no entries for George Jeffries and showing me the list of entries closest to the one I've sought. When I see the list, I realize that in fact I got the author's first name wrong and it's Gregory, not George. I choose the entry I want and the system displays the location to tell me where to find the book."

From Interaction Design: Beyond Human-Computer Interaction, 2nd Edition; Sharp, Rogers, Preece. 2007.

Task scenarios

Caller:	(Dials 233-888-8888.)
Operator:	Irish National Olympic Committee. Can I help you?
Caller:	I want to leave a message for my son, Michael.
Operator:	Is he from Ireland?
Caller:	Yes.
Operator:	How do you spell his name?
Caller:	K-E-L-L-Y
Operator:	Thank you. Please hold for about 30 seconds while I connect you to the Olympic Message System.
Operator:	Are you ready?
Caller:	Yes.
OMS:	When you have completed your message, hang up and it will be automatically Sent to Michael Kelly. Begin talking when you are ready.
Caller:	"Michael, your Mother and I will be hoping you win. Good luck." (Caller hangs up.).

Example of a Parent Leaving
a Voice Message for an Olympian (from Gould, 1987)

Task scenarios for usability testing

- A narrative that describes the action that you ask the participant to take on the tested interface.
- Need to **provide context** so users engage with the interface and pretend to perform the tasks as if they were at home or in the office.

- Example:

  You're planning a vacation to Katmandu, April 10 − April 24. You need to buy both flights and hotel. Use application X to find the best deals.

Task scenarios for usability testing

- Do not give clues nor describe the steps
- Avoid terms used in the interface

- Example:

  User goal: Check grades.

  Poor task scenario: You want to check the results of your exams. Go to the website X, sign in, click on Courses -> Grades.

  Better task scenario: Look up the results of your exams in website X.

Task scenarios for usability testing

- Avoiding clues does not mean being vague
- Example:

  Poor task scenario: Make an appointment with your doctor.

  Better task scenario: Make an appointment for next Wednesday at 4pm with your doctor, Dr. Philips.

Aplications (task analysis)

- User guide production
- Requirements specification
- Detailed interface design
  - Lists of object/action suggest interface elements
  - Sequences of actions guide the dialog design.

Example: IDEO

https://www.youtube.com/watch?v=GYkb6vfKMI4

Example: IDEO

https://www.youtube.com/watch?v=M66ZU2PCIcM

Generating ideas

- First, think about and write down your individual ideas
- Then, brainstorm: come together to discuss and build upon each other's ideas.
- Get everything on the board

Generating ideas

- Some IDEO tips on better brainstorming: (https://www.ideou.com/pages/brainstorming)
  1. Defer judgment
  2. Encourage wild ideas
  3. Build on ideas of others
  4. Stay focused on the topic
  5. One conversation at a time
  6. Be visual
  7. Go for quantity

Summary

- Define the necessary data
- Collect data using the different methods and techniques
- Represent tasks and sub-tasks
- Use these data as the basis for design

- Be efficient!

References

- Dix, Alan, Finlay, Janet, Abowd, Gregory, Beale, Russel. Human-Computer Interaction (3rd Edition), Prentice Hall, 2004.
- Gonçalves, D., Fonseca, M.J., and Campos, P., Introdução ao Design de Interfaces. FCA, 2017.
- Preece, Rogers and Sharp, Interaction Design, Wiley, 2002.

Assignment : Read and analyse

- John Gould et al., The 1984 Olympic Message System: a test of behavioral principles of system design. *Communications of ACM*, v.30 n.9, 1987. (http:/doi.acm.org10.1145/30401.30402)
  You tube video: https://www.youtube.com/watch?v=W6UYpXc4czM&feature=related