CS120 – Introduction to Human-Computer Interaction

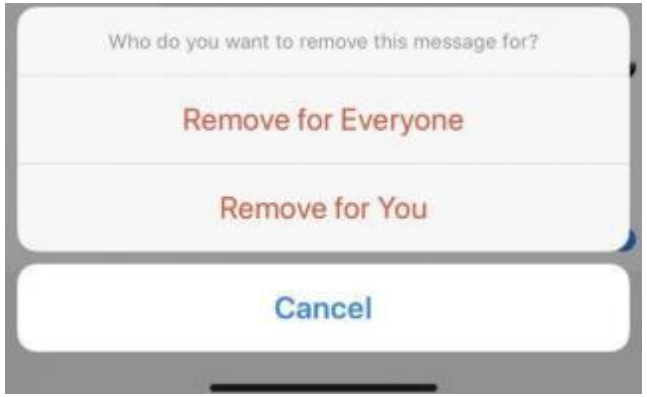
2nd Semester, AY 2025-2026

Analyzing and Synthesizing

Mary Joy P. Canon
Course Professor

Slides from Human Computer Interaction by Luigi De Russis

Hall of Fame or Shame?



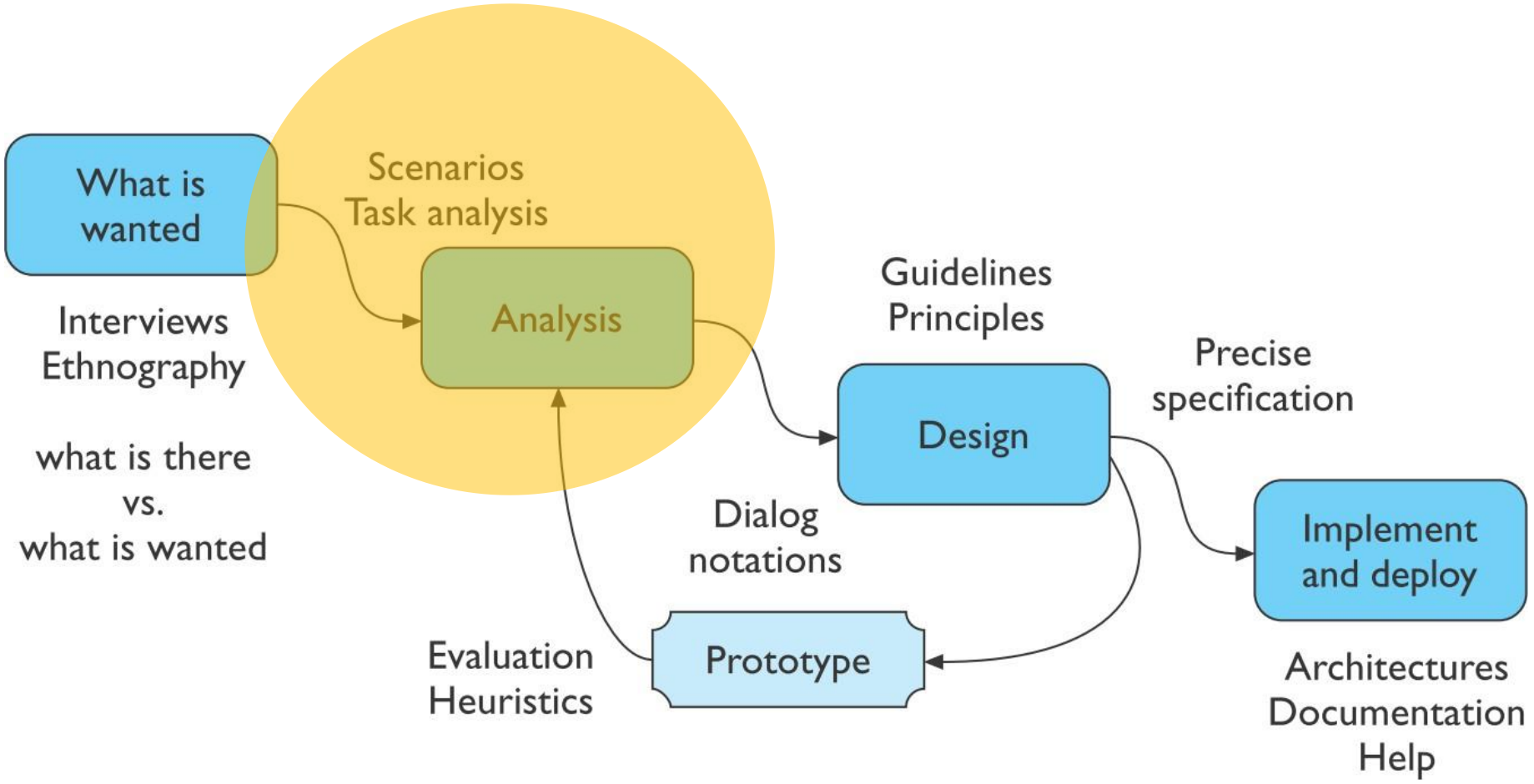
Me: *unsend a message

Friend: What did you unsent?

Me: It was just a typo

Friend:

Human-Centered Design Process



Goals

- Create **design goals**
  - As an intermediate representation before the user interface design
- Make the user needs' analysis **explicit**
- Think about the *interplay* between the activity that someone has and the interface we offer
- **Represent** and synthetize the results of the analysis and the design goals

Task Analysis

How people perform their activities

Task Analysis

- Task Analysis is the study of the way people perform their activities
- Aim is to determine:
  - what they **do** (steps)
  - what things they **use** (artifacts)
  - how well they **succeed** (goals, pain points or workarounds)

Sample Task: To Clean The House (I)

- **Steps**:
  - get the vacuum cleaner out
  - fix the appropriate attachments
  - clean the rooms
  - when the dust bag gets full, empty it
  - put the vacuum cleaner and tools away
- Must know and use different **artifacts**:
  - vacuum cleaners, their attachments, dust bags
  - cupboards, rooms
  - …

Sample Task: To Clean The House (II)

- **Goals:**
  - Here your *point of view* comes in
  - Removing dust? -> narrow goal
  - Tidying up the house after a party?
  - Hosting people for the dinner?
  - Having a satisfying evening? -> wide goal

Sample Task: To Clean The House (III)

- **Pain points**:
  - Narrow version: Why I need to empty the dust bag?
  - Broader version: Why I need a vacuum cleaner to have the house cleaned up?

Another Example of Task (with Steps)

- A person preparing an overhead projector for use would be seen to carry out the following steps:
  1. Plug in to main and switch on supply.
  2. Locate on/off switch on projector.
  3. Discover which way to press the switch.
  4. Press the switch for power.
  5. Put on the slide and orientate correctly.
  6. Align the projector on the screen.
  7. Focus the slide.

What is a Tasks?

- «A **task** is a **goal** together with some ordered set of **actions.**» (Benyon)

Goal

- A state of the application domain that a work system (user+technology) wishes to achieve.
- Specified at particular levels of abstraction.

Task

- A structured set of activities required, used, or believed to be necessary by an agent (human, machine) to achieve a goal using a particular technology.
- The task is broken down into more and more detailed levels of description until it is defined in terms of actions.

Action

- An action is a task that has no problem solving associated with it and which does not include any control structure.
- Actions are 'simple tasks'.

What You Learn with Task Analysis

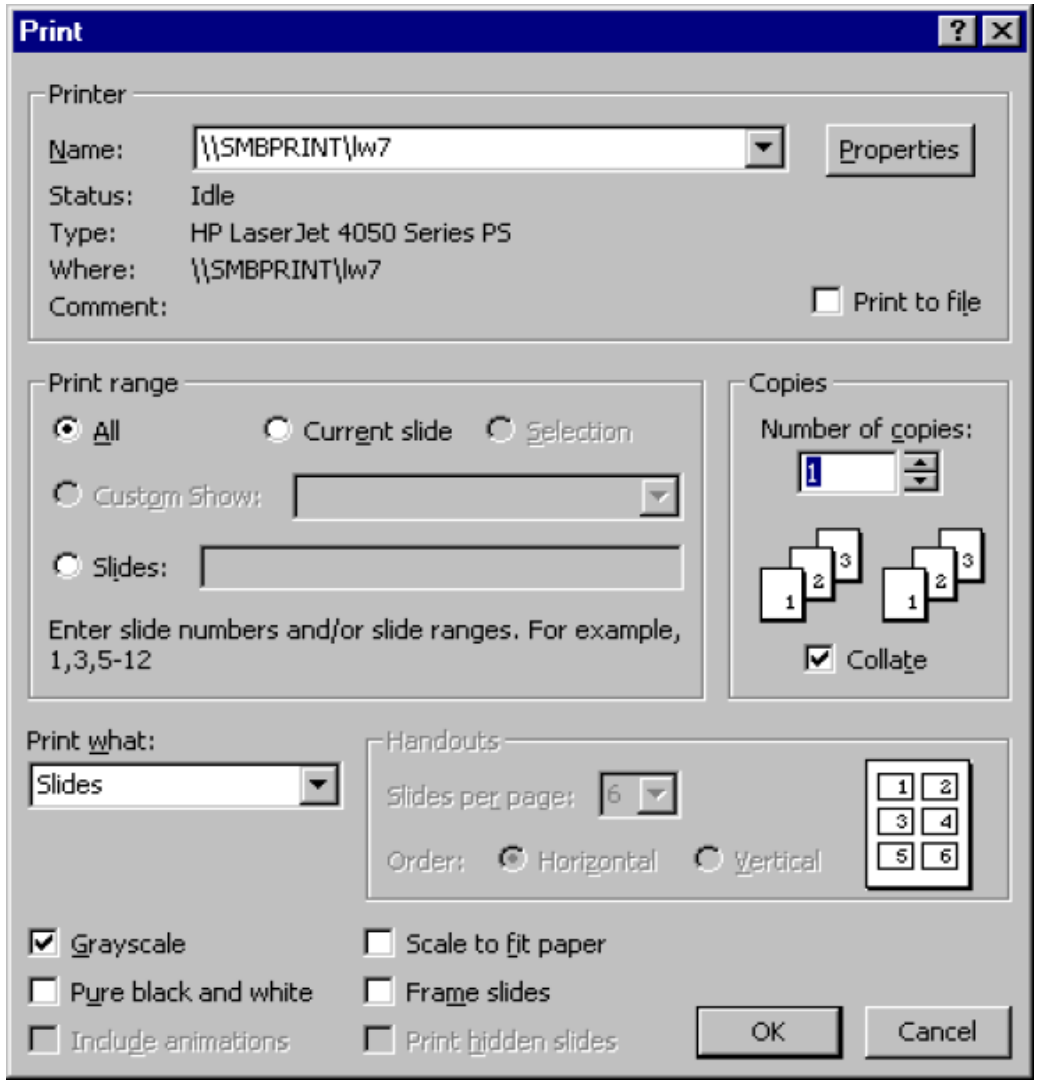
- What your users' goals can be; what they are trying to achieve
- What users actually do to achieve those goals
- What experiences (personal, social, and cultural) users bring to the tasks
- How users are influenced by their physical environment
- How users' previous knowledge and experience influence:
  - How they think about their work
  - The workflow they follow to perform their tasks
  - The pain points they experience to perform the tasks

Why Is It Useful?

- Task analysis is the process of learning about ordinary users by observing them in action to understand in detail **how** they perform their **tasks** and achieve their intended **goals**

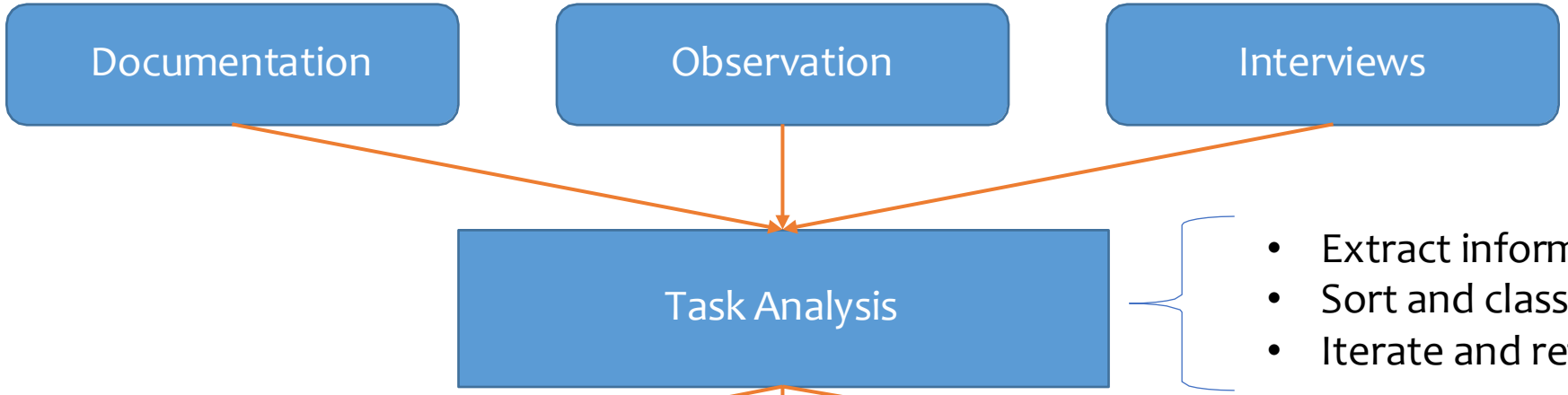
- Tasks analysis helps in:
  - Identifying the tasks that your application **must support**
  - Refining or re-defining your app's navigation or search
  - Application requirements gathering
  - Developing your content strategy and app structure
  - The initial stages of Prototyping
  - Performing usability testing

Example



- Tasks are used to plan for the layout of the application window
- Proximity and Boundaries reflect the decomposition of tasks
- Order of tasks is not mandatory

Where It Fits



- Conceptual Manuals (from KBA)
- Procedural (how-to) Manuals (from HTA)

- Focus on system usage rather than system features
- Suggests candidates for automation
- Unconvers user's conceptual model

- Taxonomies → menu layout
- Objects/actions → interface objects
- Task frequency → default choices

Characteristics of Task Analysis

- Task analysis is easier when you have well-defined **workflows** (e.g., planning a trip somewhere)
  - or **repeated activities**, such as scheduling
- Challenge:
  - We **do not** design tasks, but interfaces
  - Tasks and objects do not map 1:1
    - e.g., a web app has multiple tasks
  - People use the same interface and application to achieve slightly different results or do things differently one another

[Some] Techniques for Task Analysis

- **Task decomposition** – Splitting tasks into sub-tasks and their ordering
- **Knowledge-based techniques** – Any information and instructions that users need to know, and how that knowledge is organized
- **Entity-relationship-based analysis** – identify actors, objects, relationships and their actions
- **Ethnography** – Observation of users' behavior in the use context
- **Protocol analysis** – Observation and documentation of actions of the user. This is achieved by authenticating the user's thinking. The user is made to think aloud so that the user's mental logic can be understood.

Hierarchical Task Analysis

A Task decomposition method

Hierarchical Task Analysis (HTA)

- One possible method for Task Decomposition
- Hierarchical Task Analysis is the procedure of **disintegrating tasks into subtasks** that could be analyzed using the logical sequence for execution
- This would help in achieving the goal in the best possible way

> "A hierarchy is an organization of elements that, according to prerequisite relationships, describes the path of experiences a learner must take to achieve any single behavior that appears higher in the hierarchy". (Seels & Glasgow, 1990, p. 94)

Example HTA: How To Clean a House

0. in order to clean the house
   1. get the vacuum cleaner out
   2. fix the appropriate attachment
   3. clean the rooms
      3.1. clean the hall
      3.2. clean the living rooms
      3.3. clean the bedrooms
   4. empty the dust bag
   5. put the vacuum cleaner and attachments away

Plan 0: do 1 – 2 – 3 – 5 in that order.
when the dust bag gets full do 4

Plan 3: do any of 3.1, 3.2 or 3.3 in any order
depending on which rooms need cleaning

- A hierarchy of tasks and sub-tasks
  - Indentation and numbering denote the levels
- A set of **plans** describing in what order and under what conditions subtasks are performed
  - Plans are labeled by the task they describe

Notes

0. in order to clean the house
   1. get the vacuum cleaner out
   2. fix the appropriate attachment
   3. clean the rooms
      3.1. clean the hall
      3.2. clean the living rooms
      3.3. clean the bedrooms
   4. empty the dust bag
   5. put the vacuum cleaner and attachments away

Plan 0: do 1 – 2 – 3 – 5 in that order.
when the dust bag gets full do 4

Plan 3: do any of 3.1, 3.2 or 3.3 in any order
depending on which rooms need cleaning

- Not all tasks are mandatory
  - E.g., task 4 is needed only if the bag is full.
- The order or operations may be free
  - E.g., the rooms may be cleaned in any order
- Could be further refined with additional knowledge or context
  - E.g., Plan 3: do 3.1 every day
    3.2 once a week
    when visitors are due 3.3

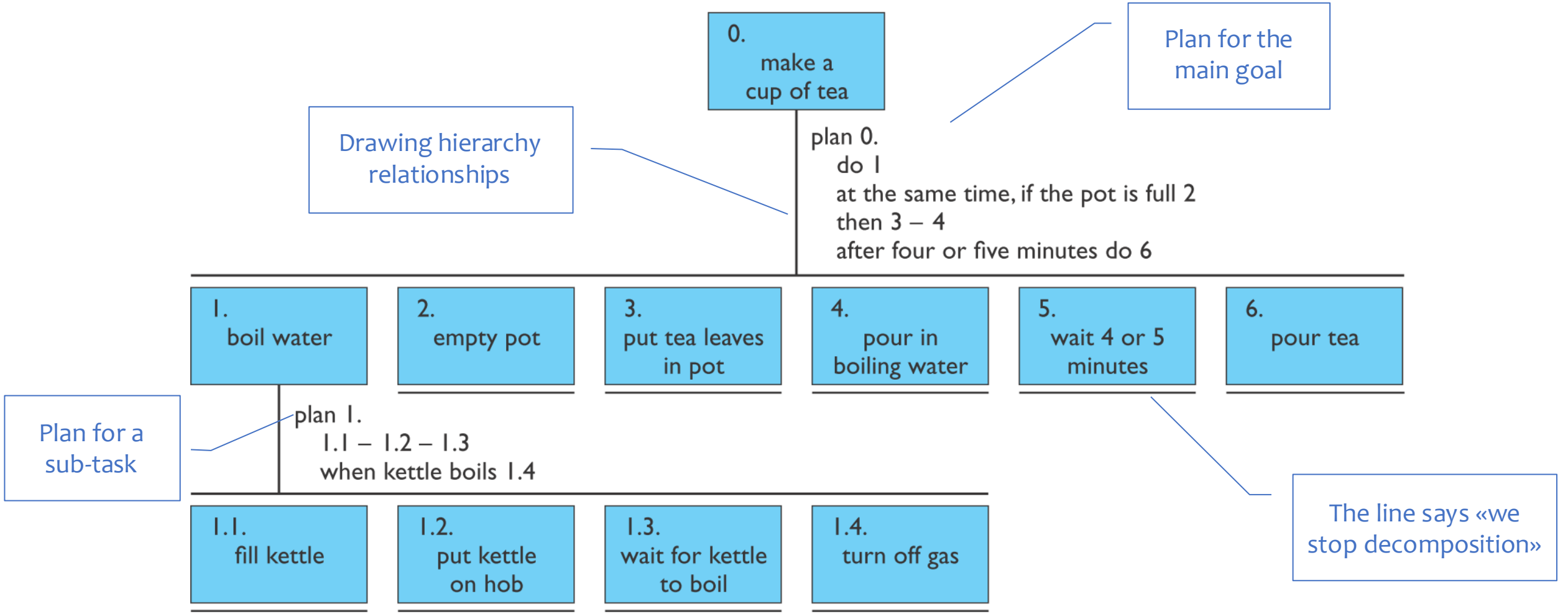
Expanding the Hierarchy (I)

- Each task is de-composed in sub-tasks, iteratively and recursively
  - Answer to the question: «what subtasks must be accomplished in order to perform the main task?»
  - The answer will come from direct observation, expert opinion, documentation, …
- Procedural task knowledge elicitation techniques:
  - Observation, re-enactment
  - Ask about procedures and triggers (pre-conditions)
  - "What happens if X goes wrong?"
  - Sorting steps into appropriate orders

Expanding the Hierarchy (II)

- When is this process stopped?
  - Depends on the intended usage of the HTA (design vs. documentation vs. troubleshooting vs. …)
  - Expand only relevant tasks
  - "Simple" tasks should be obvious to the users, and they should not contain hidden risks of failure
  - Motor actions are the lowest level (not always needed)

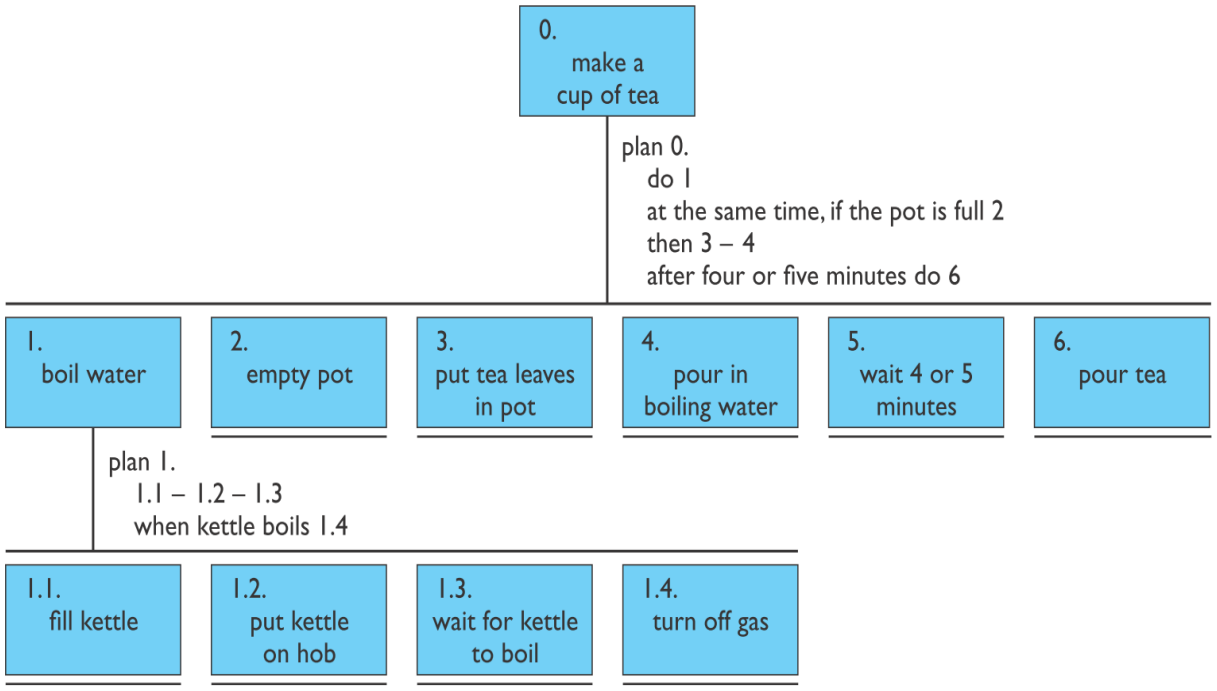
Example


0.
make a
cup of tea
plan 0.
do 1
at the same time, if the pot is full 2
then 3 – 4
after four or five minutes do 6
1.
boil water
2.
empty pot
3.
put tea leaves
in pot
4.
pour in
boiling water
5.
wait 4 or 5
minutes
6.
pour tea
plan 1.
1.1 – 1.2 – 1.3
when kettle boils 1.4
1.1.
fill kettle
1.2.
put kettle
on hob
1.3.
wait for kettle
to boil
1.4.
turn off gas
Plan for the main goal
Drawing hierarchy relationships
Plan for a sub-task
The line says «we stop decomposition»

Tasks as Explanation of Goals

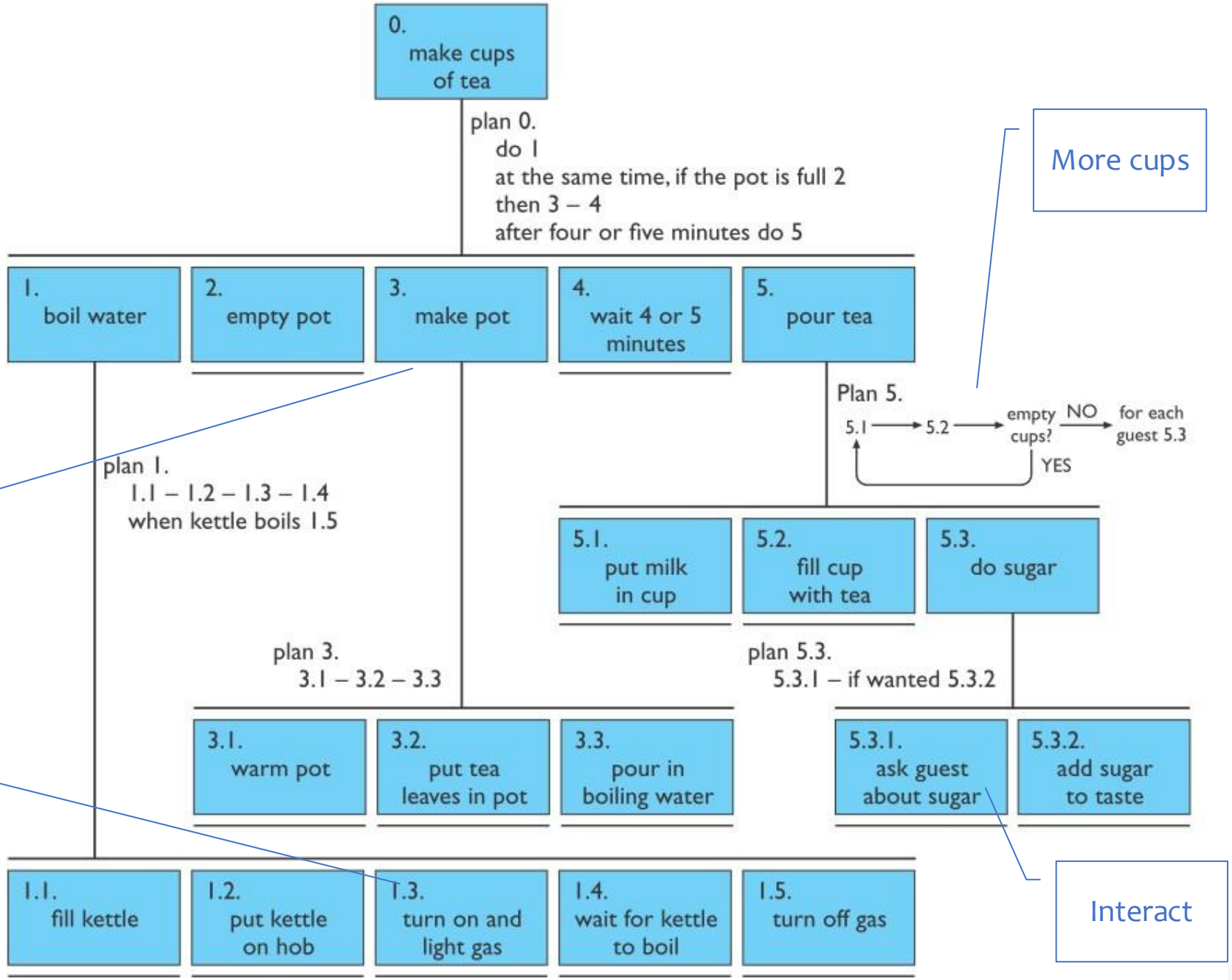
- Imagine asking the user the question:
  - **What are you doing now?**
- For the same action, the answer may be:
  - Typing ctrl-B
  - Making a word bold
  - Emphasizing a word
  - Editing a document
  - Writing a letter
  - Preparing a legal case

Refining the HTA



- Checking matched actions
  - Turn "off" without turning "on"?
- Restructuring
  - "Make pot" might be a meaningful task and group related actions
- Balancing complexity
  - Is "pour tea" simpler than "make pot"?
- Generalizing
  - If we want to make one *or more* cups?

Modified HTA



Main Constructs To Define Plans

- **Fixed sequence** - 1.1 then 1.2 then 1.3
- **Optional tasks** - if the pot is full 2
- **Wait for events** - when kettle boils 1.4
- **Cycles** - do 5.1 5.2 while there are still empty cups
- **Time-sharing** - do 1; at the same time ...
- **Discretionary** - do any of 3.1, 3.2 or 3.3 in any order
- **Mixtures** - most plans involve several of the above

Knowledge-based Analysis

Knowledge-based Analysis (KBA)

- Aim to understand knowledge required for a task
  - Provide training material, how-to manuals
  - Take advantage of common knowledge across tasks
  - Organize information and Navigation in the application
- Focus on:
  - Objects used in task
  - Actions performed
- Knowledge-based analysis in Human-Computer Interaction (HCI) focuses on understanding the **knowledge users need to operate a system efficiently** and the **cognitive processes involved as they interact** with a user interface. This form of analysis is critical in designing systems that are intuitive and meet the users' cognitive abilities and expectations.

2 main categories of Knowledge-based Analysis

1. Declarative Knowledge: This is the factual knowledge about the system and its functions, such as knowing what each menu item in a software application does.

2. Procedural Knowledge: This involves understanding the processes required to perform tasks using the system, like knowing the steps to save a file in a software application.

How to conduct KBA

1. **User Interviews**
   - Conducting interviews with users to understand their background, experiences, and familiarity with similar systems. This helps identify what users know and what they need to learn.
2. **Surveys**
   - Distributing surveys to gather broad data on the user base's knowledge levels and identifying common gaps or misconceptions that need addressing in the design.
3. **Task Analysis**
   - Breaking down tasks to identify what knowledge is required at each step and how users apply their knowledge to solve problems or complete tasks.
4. **Cognitive Walkthroughs**
   - Simulating the cognitive process a new user would go through when using the system for the first time. This helps identify areas where the system does not align with user expectations or common knowledge practices.
5. **Usability Testing**
   - Observing users as they interact with the system to see how effectively they use their existing knowledge and identify areas where additional information or training is needed.

Capturing Knowledge

- Use taxonomies
  - Represent levels of abstraction
  - Organization (grouping) depends on purpose

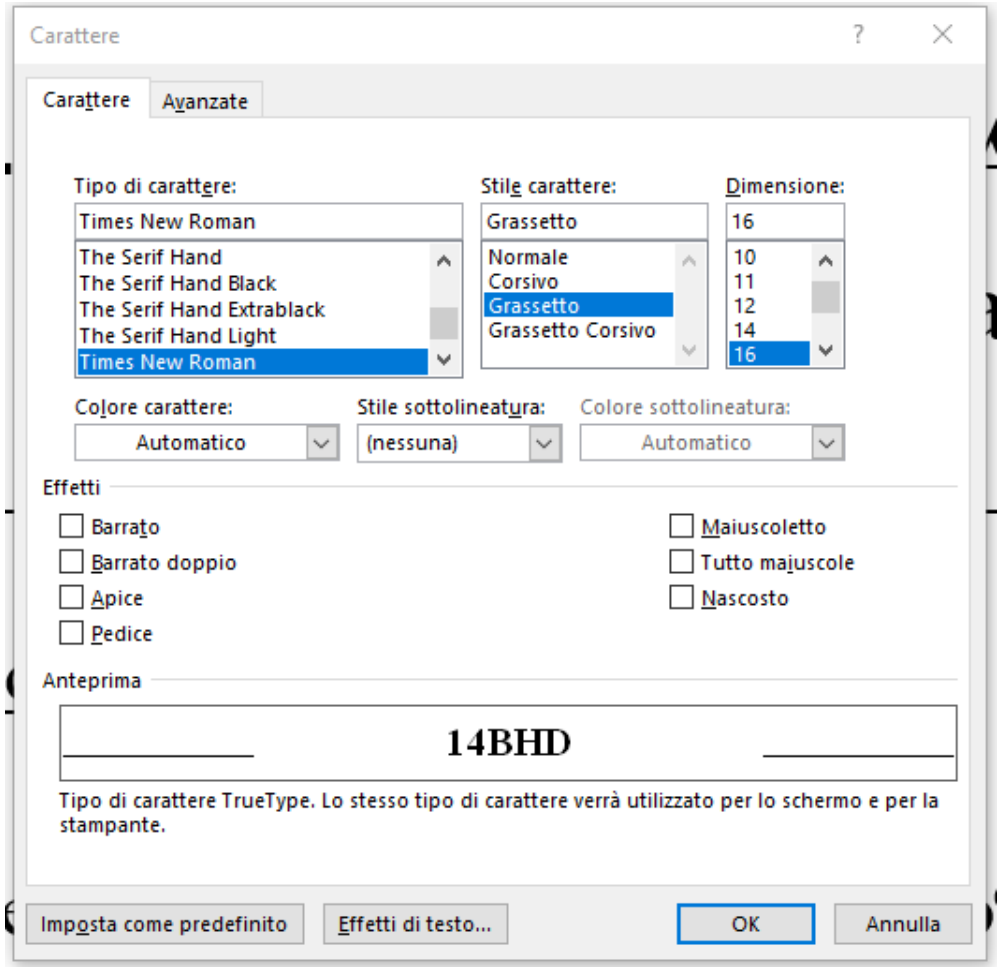
Example: Taxonomy of Car Controls

- motor controls
  - steering — *steering wheel, indicators*
  - engine/speed
    - direct — *ignition, accelerator, foot brake*
    - gearing — *clutch, gear stick*
  - lights
    - external — *headlights, hazard lights*
    - internal — *courtesy light*
  - wash/wipe
    - wipers — *front wipers, rear wipers*
    - washers — *front washers, rear washers*
  - heating — *temperature control, air direction, fan, rear screen heater*
  - parking — *hand brake, door lock*
  - radio
    - numerous!

How To Generate Concepts?

- Declarative knowledge elicitation techniques:
  - Established convention, existing documentation
  - Asking users to list objects; card sorting
  - Structured interviews, listing nouns and verbs
- Group concepts according to general-specific relationships

From Concept Taxonomies To User Interfaces



- A Typeface is described by Font, Style, Size, Color
- Different “effects” may be applied
- Moving in the dialog window will explore different related concepts in the taxonomy

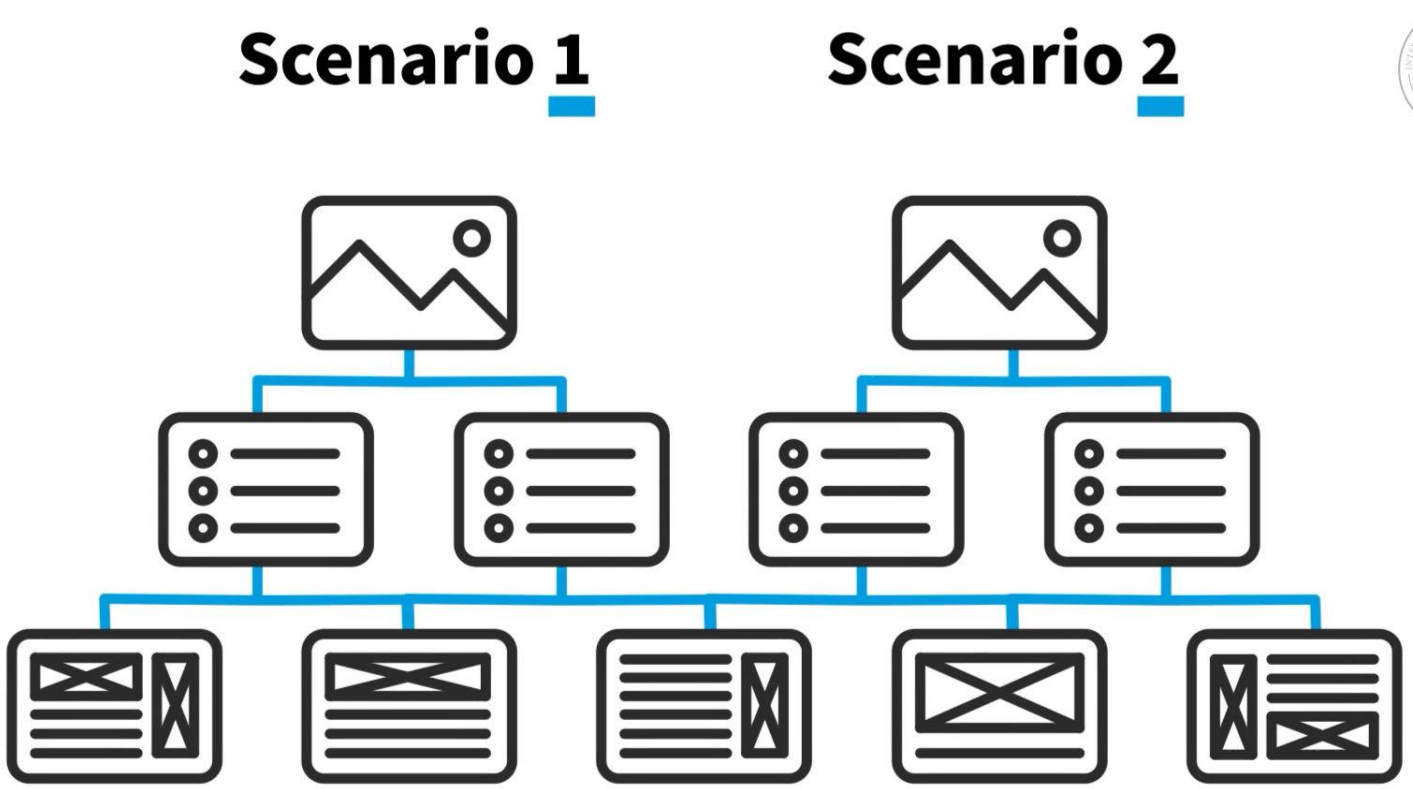
Scenarios

Possible sequences of actions for reaching user goals

Scenario

- Scenarios are stories for design: rich stories of interaction
- Description of how the user engages the interactive system to solve a specific task/goal
- Formats:
  - Written summary, Use Case
  - Graphical sketches (→ Storyboard)
  - Flowcharts, Transition Diagram…

Individual scenarios are usually describing a particular path through an interactive system.



Example Scenario in User Design

Scenario : Booking a Flight Online

Persona: Julia, a 29-year-old project manager who frequently travels for work.
Goal: To book a round-trip flight from New York to San Francisco.
Scenario:
•Julia logs into her preferred airline's booking app on her smartphone.
•She enters her departure city (New York) and destination (San Francisco).
•She selects her travel dates and specifies that she prefers morning flights.
•The system displays available flights with details including price, duration, and layover information.
•Julia filters the results to show only direct flights and sorts them by price.
•She selects the most affordable flight and chooses her preferred seat using an interactive seat map.
•Julia reviews the flight details, enters her payment information, and confirms the booking.
•She receives an email confirmation with her ticket and a calendar invite for her travel dates.

Level of Details In Scenarios

1. Stories

- From needfinding
- Used for understanding what people do and what they want

2. Conceptual (abstract) **Scenarios**

- Used for generating ideas and specifying requirements
- Abstracts tasks from stories
- No reference to technology
- May lead to different concrete scenarios

3. Concrete Scenarios

- Used for envisioning ideas and evaluation
- One possible solution to a Conceptual Scenario (may try many alternatives)
- Shows how technologies are used in the user context
- Key design features are included

4. Use Cases

- Used for specification and implementation (→ software engineering)

Example Level Details of Scenarios

1. **Stories**

Emma is a sophomore at a large university who struggles with managing her time, especially during midterms and finals. She often feels overwhelmed trying to balance study sessions, group projects, and exams. She wishes there was an easier way to organize her academic life, perhaps through some sort of digital planner that understands her course load and deadlines.

2. **Conceptual (Abstract Scenarios)**

Imagine a system where students can input their course schedules, including assignment deadlines and exam dates, at the beginning of the semester. The system could automatically generate a personalized study schedule that adjusts as deadlines approach or priorities change.

Example Level Details of Scenarios

3. Concrete Scenarios

Emma downloads the 'StudySmart' app to her smartphone and sets up her profile. She enters her semester courses, noting down each course's major deadlines. The app displays a weekly calendar view that includes scheduled study times, reminds her of upcoming deadlines, and suggests the best times to work on assignments based on her past study habits. One week before her statistics exam, the app automatically adjusts her study times to focus more on that subject.

4. Use Cases

Use Case Name: Update Study Schedule
Actor: Student (User)
Preconditions: User is logged into the 'StudySmart' app, and the semester courses and initial deadlines are already entered.
Main Flow:

- User selects 'Update Schedule' from the menu.
- User adds a new assignment with a due date.
- The system recalculates the study schedule based on the new information.
- The system displays the updated schedule with adjusted study sessions.
- User receives a notification confirming the schedule update.

Postconditions: The study schedule reflects the new assignment's due date, and the user is notified.

Storyboards

Comic book – like representation of user scenarios, with emphasis of how the system supports users in the development of the task

Storyboard

- "A graphical depiction of the outward appearance of the intended system, without any accompanying system functionality"
- A hand-drawn comic that features the execution of a task (like a concrete scenario)
- With a few panels (sequence of sketches) it conveys what a person may accomplish
  - Always include people
- They communicate **flow**, showing what happens **at key points** in time
- No artistic skills are required
  - Not about "nice pictures"
  - About communicating ideas

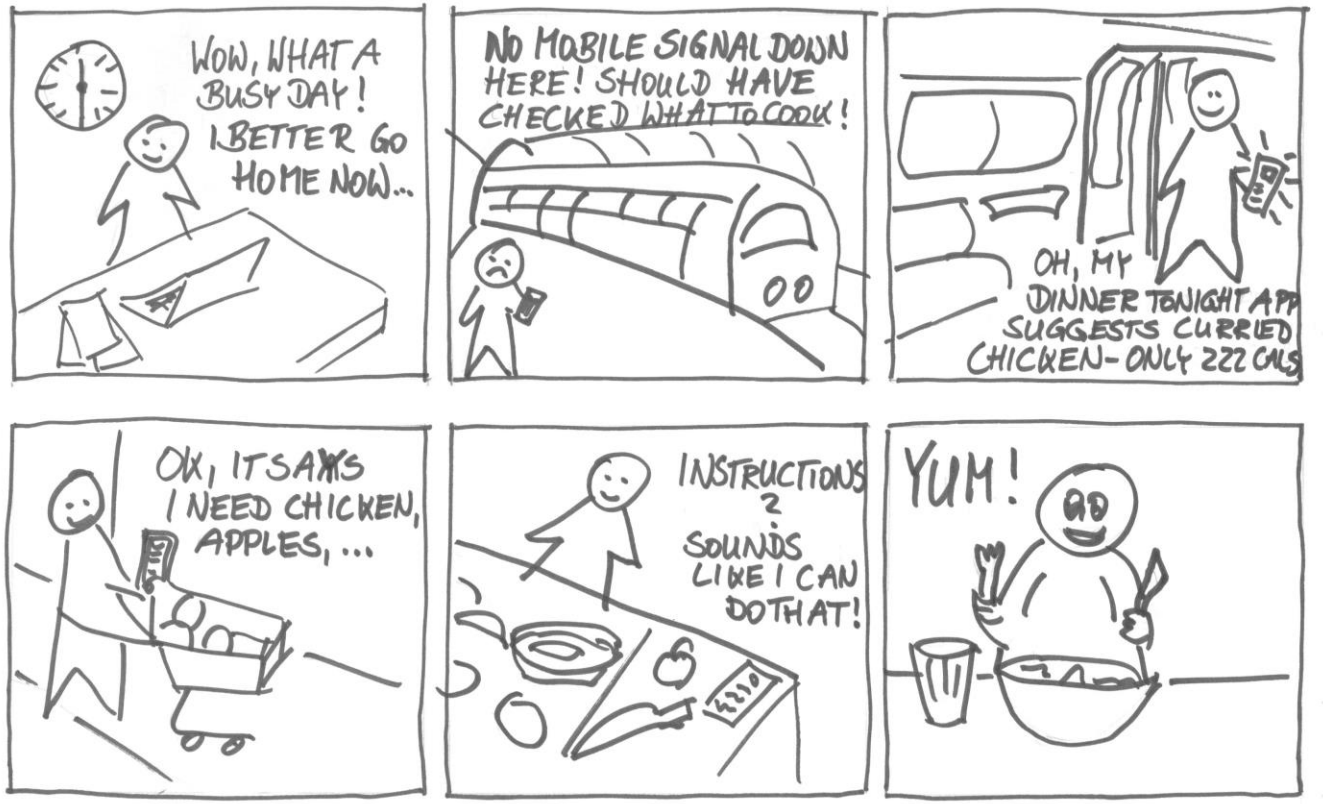
What To Find In a Storyboard

- Illustrate a goal (for the task)
- How a task unfolds (people interacting among themselves and with devices)
  - Repeated for all significant steps
- At the end, how they accomplish their goals (satisfactory outcome)

- Storyboards are **all about tasks**

Example

This storyboard illustrates how the app had already downloaded the daily recipe to the user's smartphone, so he could look it up and check the shopping list while on the underground, before shopping for ingredients and making a healthy meal.



http://alexmevissen.com/2014/07/16/storyboarding/

Example

This storyboard illustrates how the app can show the user that a home cooked meal can be quicker than ordering food delivery, using left over ingredients in the fridge.

http://alexmevissen.com/2014/07/16/storyboarding/

Storyboards Should Convey…

- Setting
  - People involved
  - Environment
  - Task being accomplished
- Sequence
  - What steps are involved?
    - Not the detailed UI
    - What role the UI plays in helping users achieve their goal?
  - What leads someone to use the system?
    - The "trigger" for the task
  - What task is being illustrated?
- Satisfaction
  - What's the motivation for the user?
    - The end point to reach after all the steps
  - What's the end result?
  - What need are you "satisfying"?

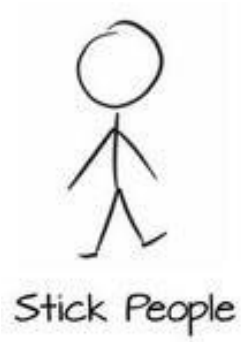
Handling Dynamicity In Storyboards

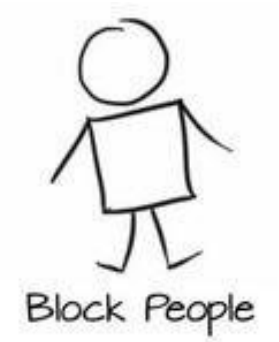
- Traditional storyboarding
  - “Comic book” conventions: actors, speech bubbles, background
  - Notes attached to each scene explaining what is happening
- Scored storyboards
  - When the user interface is highly dynamic, or contains specific media elements
  - Add specific annotations focusing on movement, colors, sounds, …
- Text-only storyboards
  - When the interaction behavior is too complex to compact into an illustration, use a longer text description

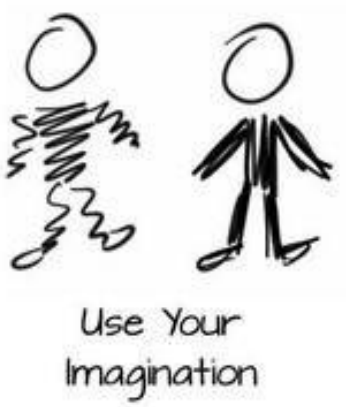
Why Hand-drawn?

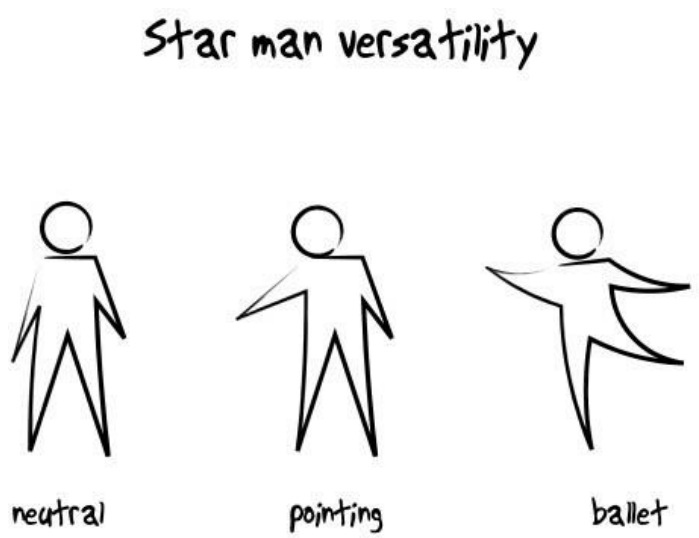
- Quick
  - No need to spend time in graphics tools (they would "push" you to focus on details, too
  - Able to experiment different scenarios
- Imprecise
  - Users feel free to express more comments and suggestions w.r.t. a more "polished" version
  - Focus on the content (the graphics is obviously ignored)
  - No distraction by fonts, colors, icons, …

~~Drawing~~ Sketching People









Benefits of Storyboards

- Emphasize how an interface accomplishes a task
- Focus the conversation and feedback on user tasks
- Get everyone on same page about the app's goals
- Avoid nitpicking about user interface details (buttons, etc.)

References

- Alan Dix, Janet Finlay, Gregory Abowd, Russell Beale: Human Computer Interaction, 3rd Edition, Chapter 15 “Task Analysis”
- David Benyon: Designing Interactive Systems, Chapter 11 “Task Analysis”
- http://www.usabilitybok.org/task-analysis
- https://www.usability.gov/how-to-and-tools/methods/task-analysis.html

Acknowledgements

- Some icons from https://icons8.com

- Some material by
  - http://www.inf.ed.ac.uk/teaching/courses/hci/0708/lecs/tasks.pdf
  - https://www.tutorialspoint.com/human_computer_interface/design_process_and_task_analysis.htm
  - https://www.slideshare.net/alanjohndix/hci-3e-ch-15-task-analysis

- Thanks to Fulvio Corno, past teacher of the course, for his work on some of these slides

License

- These slides are distributed under a Creative Commons license **"Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)"**

- **You are free to:**
  - **Share** — copy and redistribute the material in any medium or format
  - **Adapt** — remix, transform, and build upon the material
  - The licensor cannot revoke these freedoms as long as you follow the license terms.

- **Under the following terms:**
  - **Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **NonCommercial** — You may not use the material for commercial purposes.
  - **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

- https://creativecommons.org/licenses/by-nc-sa/4.0/