Week 2, Part 1

Life cycles, design, requirements, and data

In this part:

- Design life cycles
- Design **process** vs design **approach**
- Data gathering techniques
- How to use the data you gather

What is a life cycle model?

A process for:

Planning, designing, creating, testing, implementing, deploying a system.

Lifecycle Models

Life cycle models show how activities in different stages of the design and production process are related to each other.

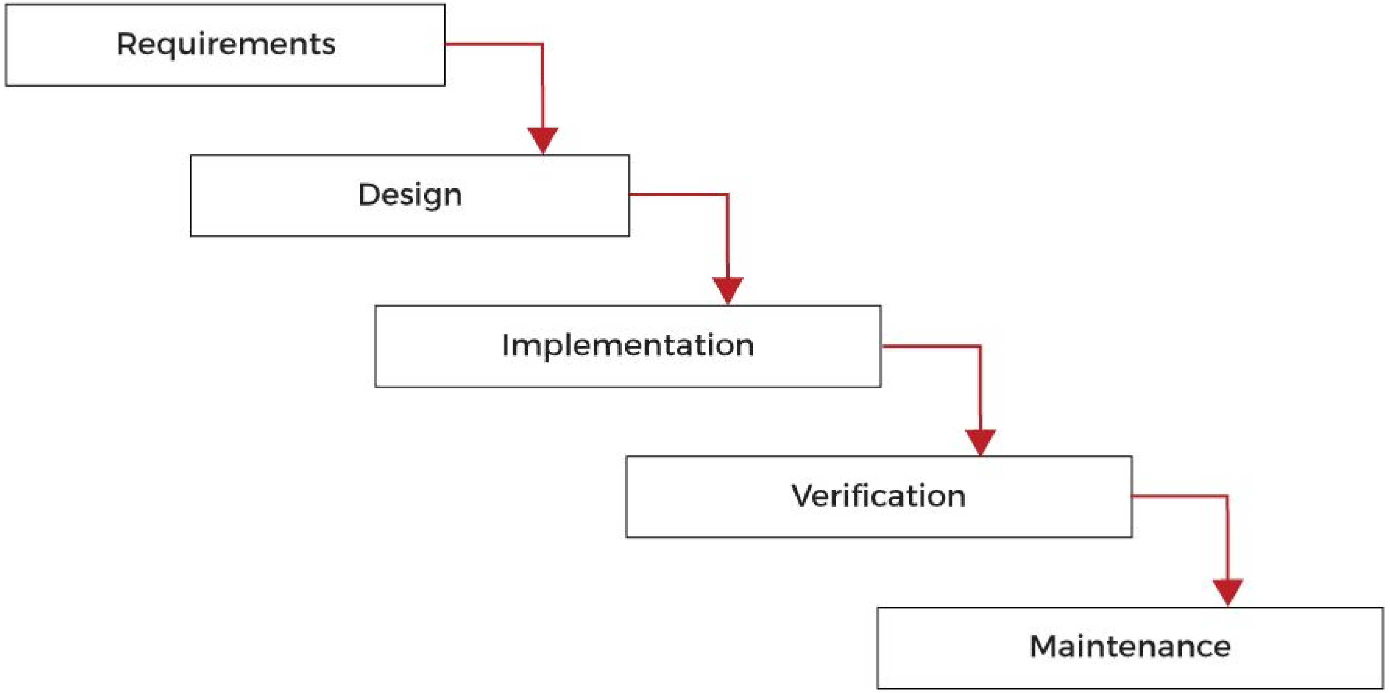
Lifecycle models are:

- **Management tools:** They provide overall view of the development effort
- **Simplified versions of reality:** Generally applicable to software design processes

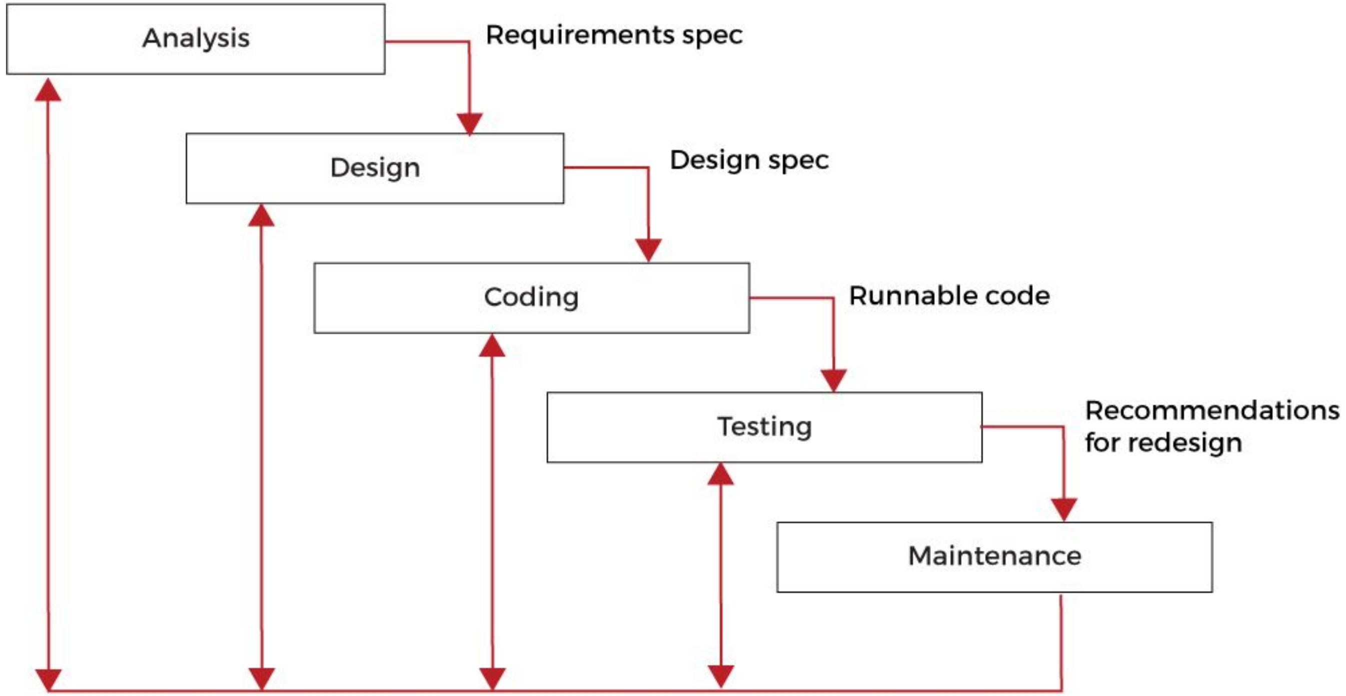
Many lifecycle models and theory about this exist:

- Software engineering: waterfall, spiral, JAD/RAD, agile
- HCI: star, usability engineering

Lifecycle Models: Waterfall



Lifecycle Models: Waterfall



User-centred design.

A set of design **processes** in which the needs, wants, and limitations of the people who use a service or product the primary focus at every stage of the product’s creation.

What does that mean?

characteristics:

- Is a **multi-stage** process
- Requires understanding of how people are likely to use a product
- Requires designers to **test the validity of their assumptions** about the behavior of the people who use the product
- Is **iterative** - the outcome of each stage needs to inform the assumptions on which it was built (it will confirm the requirements, or signal that modifying the original requirements is necessary)

function:

- To ask questions about the people who use a product
- To find out what their tasks and goals are
- To use this data to make decisions about how that product is designed.

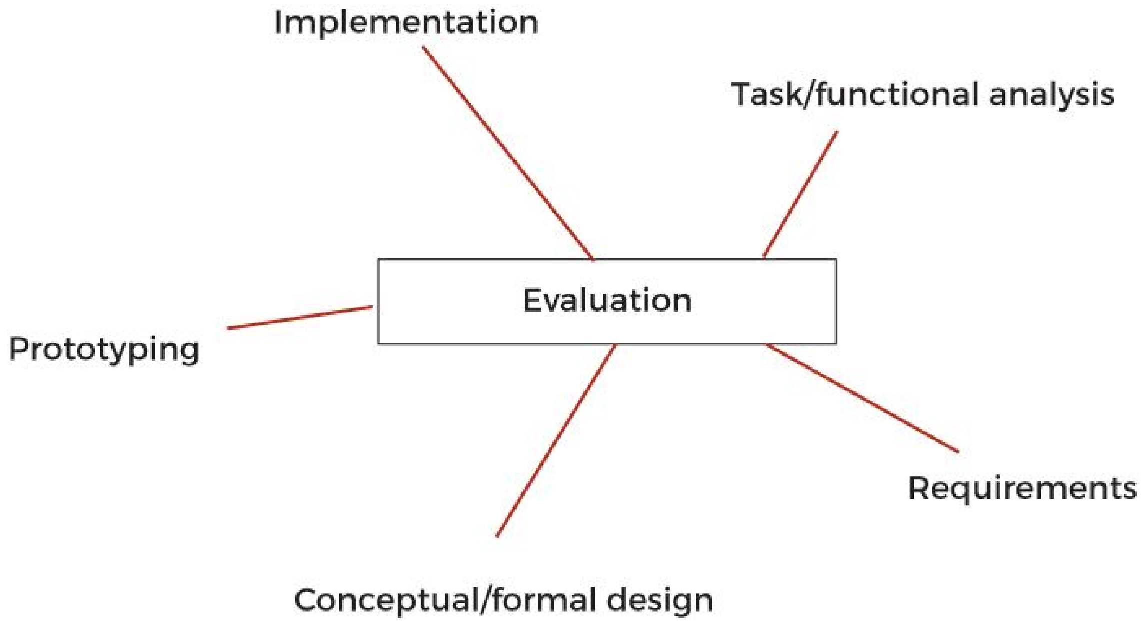
some questions:

- Who uses (or will use) this?
- What is that person's task and/or goal while using this?
- What experience does that person already have with things like this?
- What does the person using this need it to do?
- What information is that person trying to get? How do they understand that information?
- How does the person using this expect it to work?
- What is their environment like when they're using this?
- Are they doing other things - how many things are they thinking about? What information are they working with?

A design process:

A way of getting from start (idea, brief) **to finish** (product)

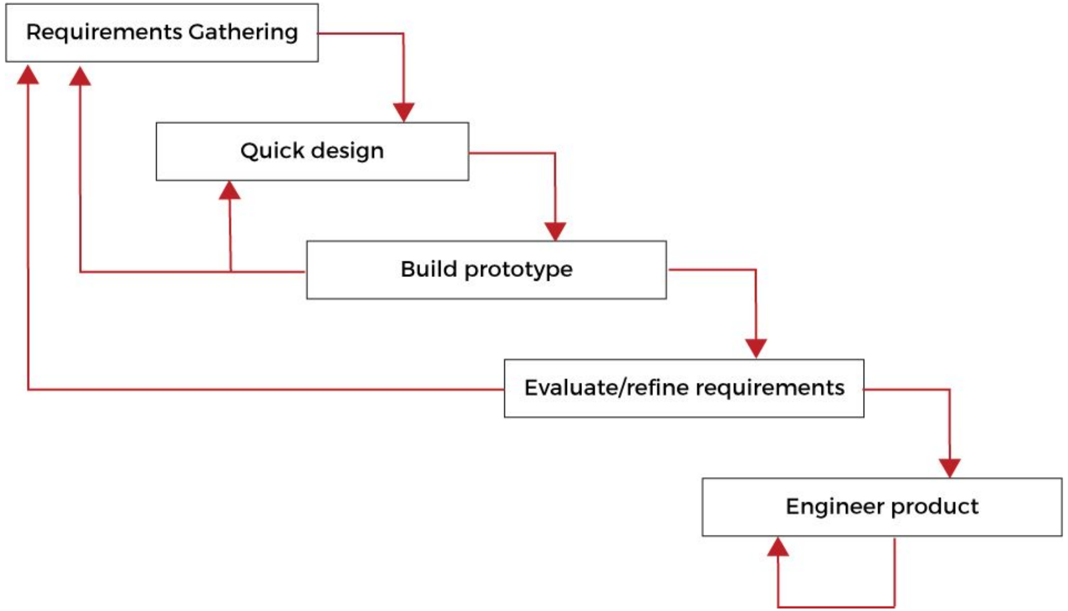
Star life cycle:



It has one major drawback:

We can’t tell what stage of the process we’re at!

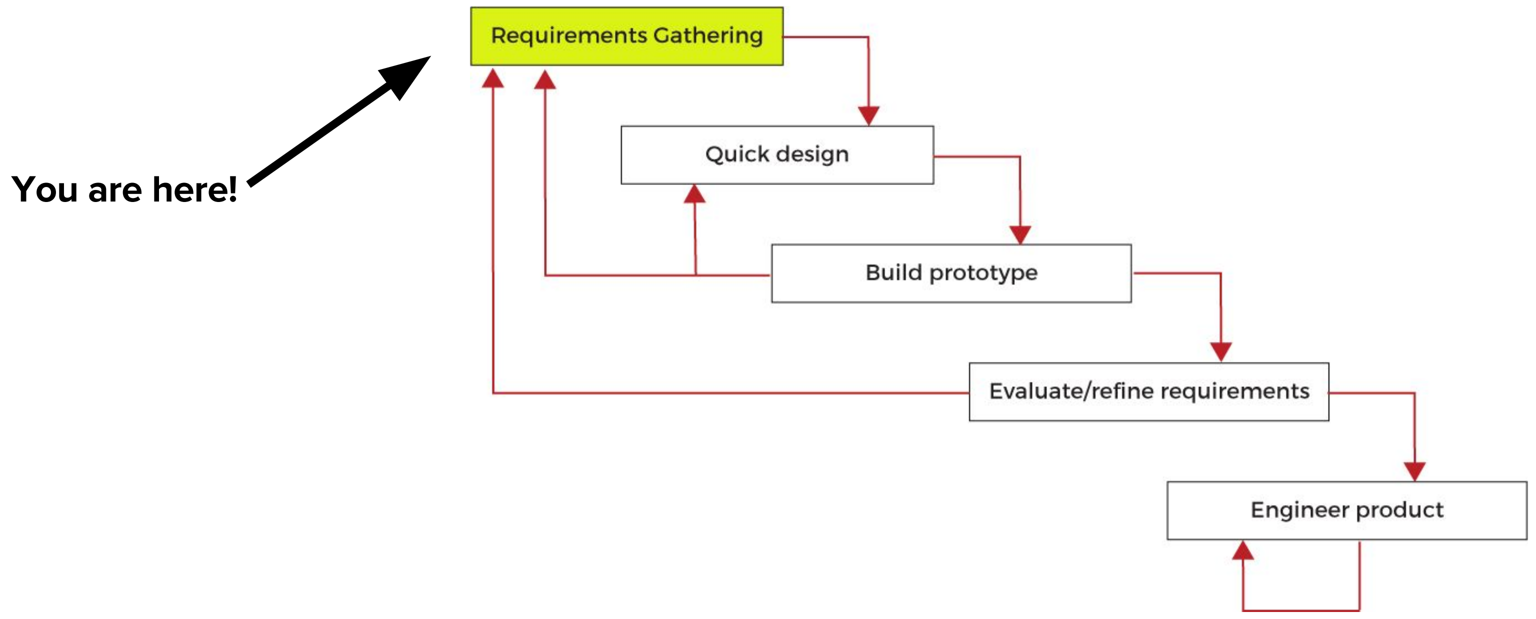
A user-centred design life cycle:



Iteration

Lifecycle Models: Iterative

User centred design lifecycle



A design approach:

The way you go through the design process.

Participatory design

A design **approach and process**

Actively involves stakeholders in the process of design to ensure that the result **meets their needs**

Is rooted in democratic process - was associated with Scandinavian trade unions in the 1960 and 70s (more next week)

Process vs approach

Process: The steps you take

Approach: How you take those steps

Both matter!

Identifying Needs and Requirements:

An iterative process

Aim for identifying needs

Understand as much as possible about people who use the thing, their tasks, habits, context

Aim for establishing requirements

Produce a **stable** set of requirements

How this is done

- Data gathering activities
- Data analysis activities
- Defining the requirements

Identifying needs and requirements:

Why?

Things can go wrong at this stage!

- People can explain their needs ambiguously
- Designers can misunderstand peoples' wishes
- Designers can assume they know what people need; not listening adequately or asking the right questions
- Requirements can be incomplete or unstable
- The result: Implementing something that was not asked for, or something that does not meet the needs of the people using it - and they won't use it!

Most importantly ...

- Costs of fixing errors in the final software product are +/- 100 times higher than finding and fixing them during the requirements analysis phase

Why identify needs and requirements?

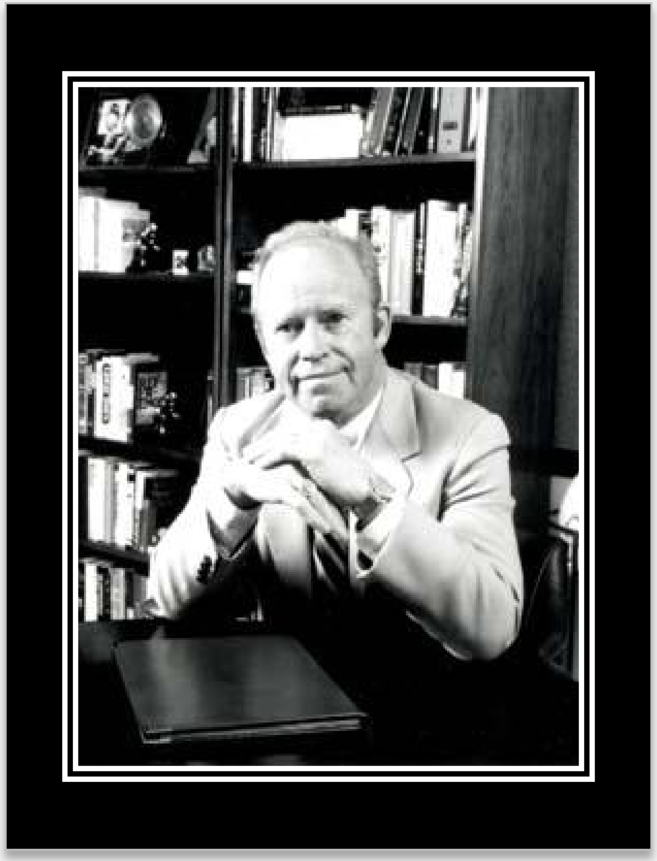
Philip Crosby

"It is always cheaper to do the job right the first time."

"A rule to live by: I won't use anything I can't explain in five minutes."

"No one can remember more than three points."

"Good things only happen when planned; bad things happen on their own."



So what are needs and requirements?

A requirement is …

A statement about what a future product should be, or how it should perform.

Requirements should be …

- Specific
- Clear
- Unambiguous

What kind of requirements are there?

Functional requirements

What should the system **do**? What does it need to **perform**?

Non-functional requirements

What are the **constraints** on the system?

- Data requirements
- Usability
- Environment
- Characteristics of the person using the system

Non-functional requirements:

Data

Non-functional requirements: Data

- What kind of data needs to be stored?
- How will it be stored?
- Who has access?
- How long must the data persist?
- How accurate should the data be?
- What permissions are needed from the person using the system?

Non-functional requirements:

Environment

Non-functional requirements: Environment

- Is the place where this will be used indoors? Outdoors? Crowded? Fast-paced? Cold? Wet? Dirty?
- What are the social characteristics? Collaborative? Secretive?
- What are the organisational characteristics - Where is this person in that structure?
- Technical requirements: Does it have to be compatible with anything?

Non-functional requirements:

User characteristics

Non-functional requirements: User Characteristics

- Who is the person who will be using this?
- What’s their level of technical skill?
- What’s their age, education, language, cultural frame of reference?
- Do they have any physical or mental differences - comprehension, eyesight?
- What do we know about their preferences?

Non-functional requirements:

Usability

Non-functional requirements:
Usability

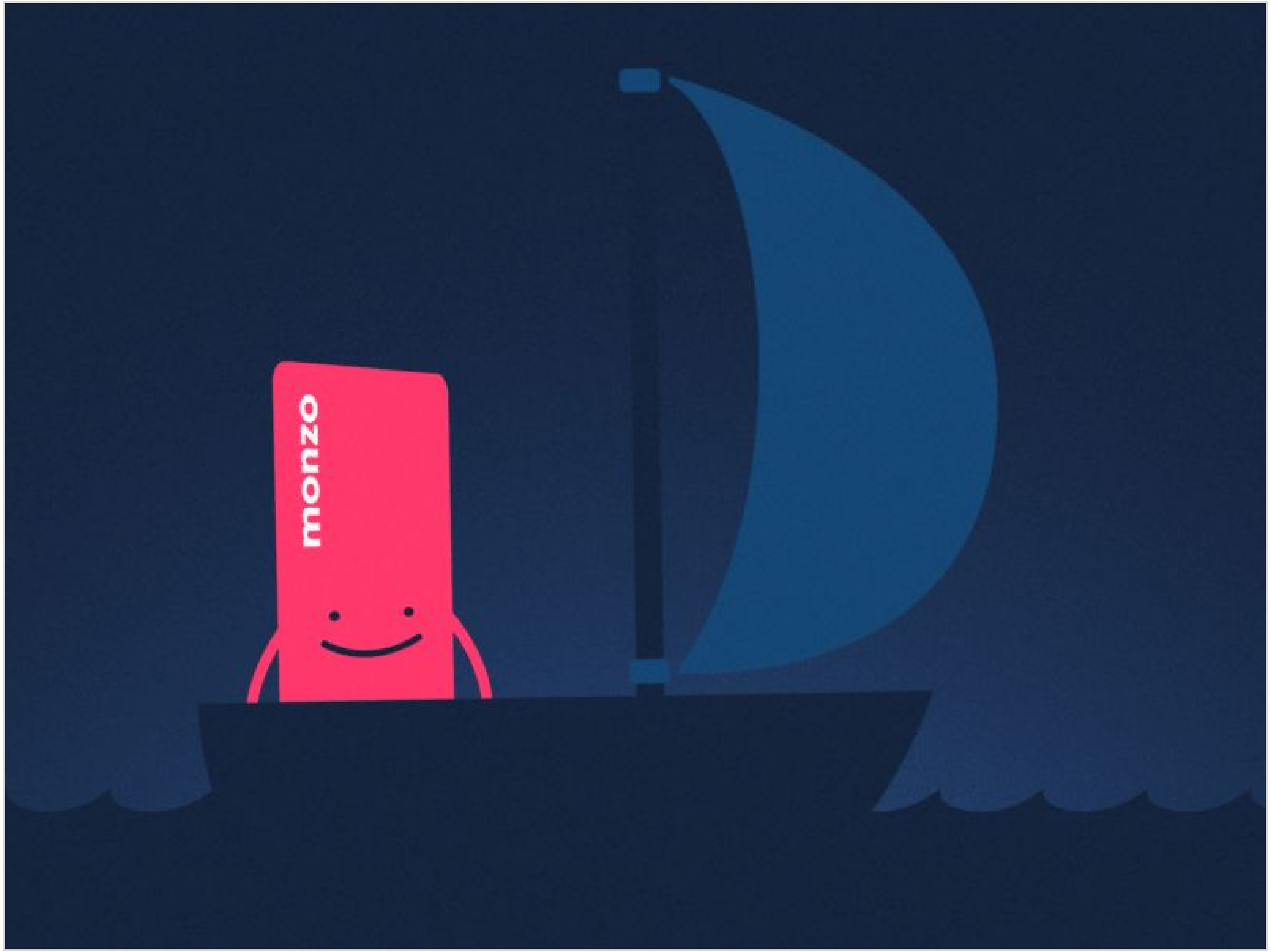
- What does the person using this system think is effective?
- What do they want to achieve?
- What do they have to learn?
- What do they have to remember/memorise?
- How fast do they need to do it?
- Are there any safety issues?

Non-functional requirements:

Experience

Non-functional requirements: Experience

- What does the person who uses this system find pleasing?
- What motivates them?
- How can using this be made more enjoyable?
- How can using this be made easy (and how do they define “easy”)?
- How can the interaction be faster/slower?
- What does this person consider delightful?


monzo

How do we collect needs and requirements?

Data collection: Aims

To collect sufficient relevant and appropriate data so that a set of stable requirements can be produced.

Collect data about the people who will use your product, their tasks, their context of use, their characteristics, and so on.

Data collection: Techniques

- Observation
- Interviews
- Card sorting
- Questionnaires
- Studying documentation
- Focus groups
- Contextual Inquiry
- Scenarios / Use cases
- Researching similar products

Observation

In the field (ethnography)

- Realistic settings/ activities
- Difficult to set up
- The more complex the observation becomes, the more it intrudes
- Problems of privacy/ reliability

In the lab

- Less intrusive
- Not realistic

Observation: Two approaches

Direct

- Ask someone what they’re doing while they’re doing it
  - Great for understanding
  - Time consuming
  - Intrusive

Retrospective

- Ask someone what they did and why after they’re finished
- Can also include actions reconstructed from data logs
- This is a less common approach

Interviews

Attempt to identify participant’s subjective opinions - good technique for exploring issues.

Generates lots and lots of data.

Typically an *interviewer* asks an *interviewee* a set of questions which may be structured in a number of ways

Interview: Types

Structured (closed questions)

- Used for clarifying a specific issue
- Easy data collection, but it’s rigid

Unstructured (open questions)

- Used for eliciting rationale
- Rich data, but generally too much; can stray off target; long.

Semi-Structured (open and closed questions)

- Used for initial eliciting of rationale, as well as in the evaluation phase
- Rich, targeted data.

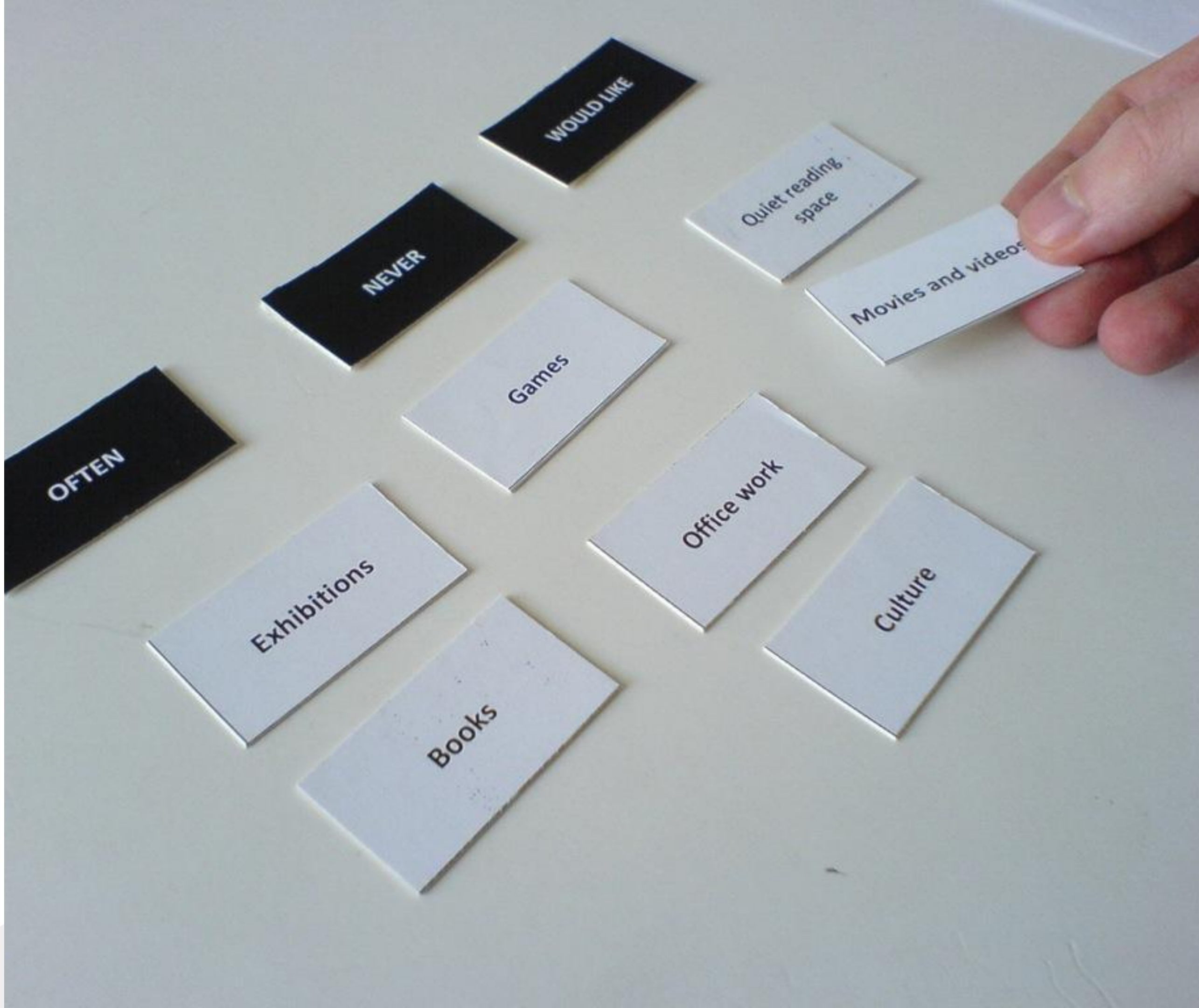
Card sorting

Interviewee is given a set of paper cards, that each have some aspect of the work or environment on them.

They are asked to sort them into groups.

Card sorting

Gives an idea of the person's values, their perceptions of the context



Card Sorting

Open Card Sorting

Participants are given cards showing content with no pre-established groupings.

They are asked to sort cards into groups that they feel are appropriate and then describe each group.

Closed Card Sorting

Participants are given cards showing content with an established initial set of primary groups.

Participants are asked to place cards into these pre-established primary groups.

Questionnaires

- A series of questions designed to elicit specific information
- Often used in conjunction with other techniques
- We can gather quantitative or qualitative data
- Good for answering specific questions from a large, dispersed group of people

Questionnaire types

Open questions

Respondents are free to write in anything they want

Closed questions

Answers must be chosen from a list, or marked on a scale. Some examples:

- Simple checklist
- Scale
- Rank ordering (indicates preference)

Rating scale

Indicate how much you agree with the following by circling a number from 0 (absolutely disagree) to 5 (totally agree).

I think dogs are great.

5	4	3	2	1
Doggo forever				I am a cat

Likert scale

I think dogs are great.

Totally agree	Agree	Slightly agree	Neutral	Slightly disagree	Disagree	Totally disagree

Questionnaires:

- Can potentially involve a larger group of people, as they’re cheap and time-effective
- Extensive use of questionnaires is a **survey**
- Rule of thumb: Stick to two sides of A4
- An **incentive** can be useful for people to complete them

Study documentation

- Procedures are often documented in manuals
- Good source of data about the steps involved and what's needed to complete a task
- Good for understanding legislation and background information
- A complementary technique - not for use in isolation
- No stakeholders needed for this
- Can be outdated or not applicable

Focus groups

- Essentially a group interview
- Good for identifying terminology and expectations
- Participants must be selected well to represent the target stakeholder group

Contextual inquiry

- Apprenticeship model: Designers work as apprentices to the users
- Good for understanding the task
- Not time effective, can get off target

Researching similar products

- Helps to generate alternatives
- Good for generating ideas
- Helps to prompt requirements
- Can stifle creativity, make you susceptible to trends that aren’t useful

Scenarios

- Scenarios are a situation that you make up in order to test what a system might do or how it might react in a given situation
- Can be used by designers, or by people who will use the product
- Good for testing if expectations are well understood - you may be surprised!

Use cases

- A type of scenario
- A scenario that is focused on intended interaction flow

Choosing data collection methods

- All techniques differ in: the amount of **time**, level of **detail**, and **risk** associated with findings
- Techniques may require specialist knowledge
- Choice of technique is also affected by the nature of your task: Is it sequential steps, or overlapping subtasks? High or low, complex or simple information? Is this task for a layman or a skilled practitioner?

Problems with data gathering

Identifying and involving stakeholders. Who are they? Where are they? Are there barriers (ie, unions/management)? What are the ethical implications (are they children/vulnerable)?

Involving stakeholders. Interviews, workshops and so on all co-opt the people using the product into the development team.

Finding “real” users. This is typically a problem in software engineering but getting better.

Problems with data gathering

Requirements management. Version control, ownership, who decides what's important?

Communication between parties. Within development team, to the person using the product, between users (different people use different terminology, have different wishes/ values ...)

Domain knowledge is distributed. It's difficult for users to articulate, difficult for designers to find, difficult for developers to understand

Availability of key people

Problems with data gathering

Political problems within the organisation

Dominance of certain stakeholders

Economic and business environment changes

Completely unforeseen events: Political crises, sudden technical advancements

Functional vs non-functional: How can these demands be balanced?

Basic Guidelines

- ❖ Focus on identifying the stakeholders’ needs.
- ❖ Involve all the stakeholder groups .
- ❖ Involve more than one representative from each stakeholder group.
- ❖ Use a combination of data gathering techniques.

Basic Guidelines

- ❖ Support the process with prototypes and task descriptions.
- ❖ Run a pilot session.
- ❖ You will need to compromise on the data you collect vs the analysis needed.
- ❖ Consider carefully how to record the data

Data analysis, interpretation, and presentation

- Should take place soon after the data was gathered
- Initial interpretation should be discussed with potential users before deeper analysis - do you think they mean what they actually mean?
- Presentation options: Data flow charts, work flow charts, graphs, diagrams, word clouds ...

Understanding your results

Pulling apart **what your data says** and what **you think it means** is crucial.

Results = the numbers

Interpretation = what you believe the numbers mean

Make sure your interpretations are reasonable and credible. Why does it mean what you think it means? How can you rationalise this conclusion?

Data gathering is critical to successful design.

There are lots of data collection options.

Choose your techniques wisely:

- Consider the kind of data you need to collect
- Consider the time needed to collect and analyse the data
- Consider how data collection might interfere with the interaction you’re trying to study

Week 2, Part 2

Task analysis and task models

In this part:

- Learn to conduct task analysis and construct task models
- Understand the advantages and disadvantages of task analysis

What is task analysis?

Task analysis is:

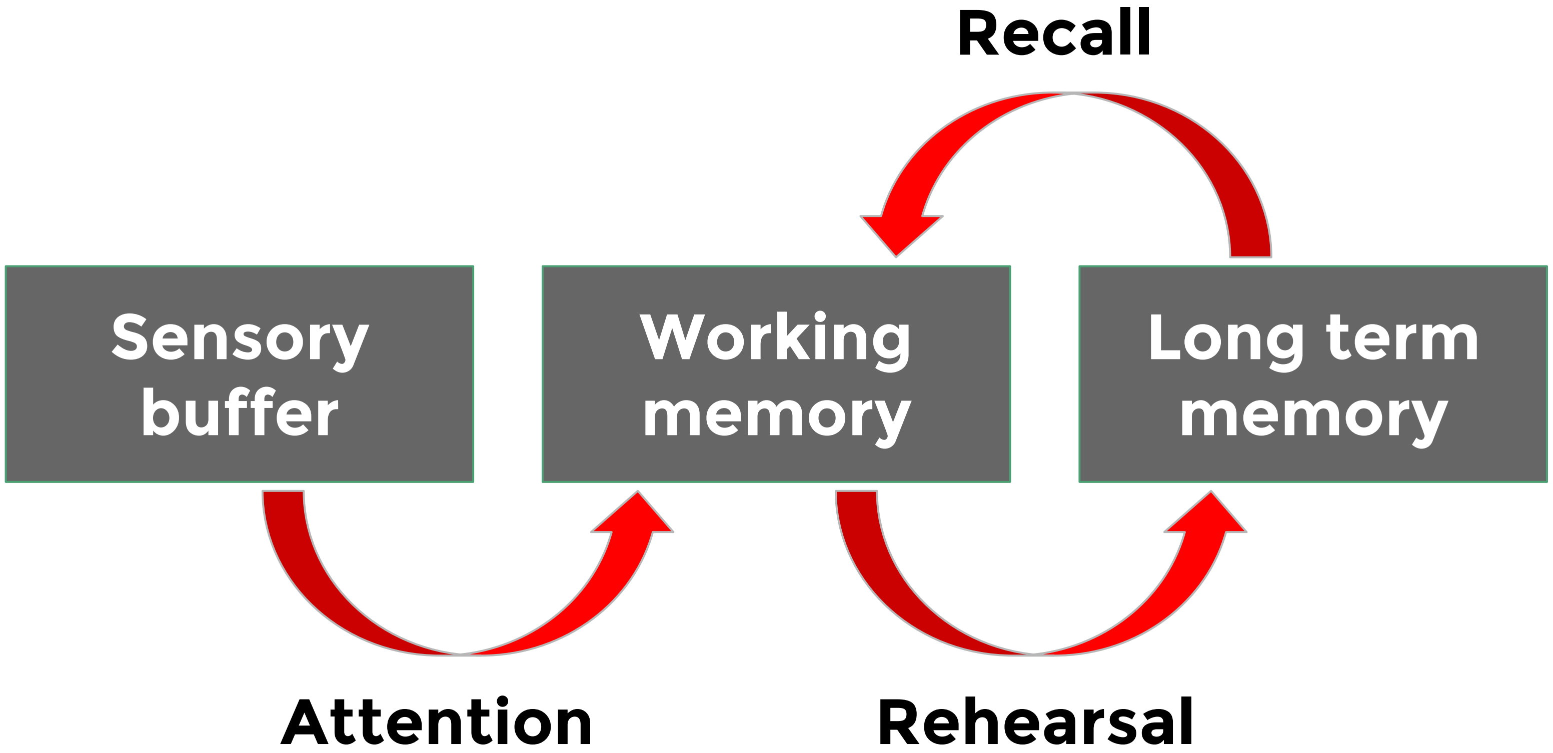
- A hierarchical composition of knowledge
- Based on notions of cognitive psychology
- A method of analysing what people do from a hierarchical perspective

* Cognition will be covered in another lecture!

Types of memory

It is generally agreed that there are three types of memory function:

- sensory buffers
- Short-term memory (“working”) memory
- Long-term memory


Recall
Sensory buffer
Working memory
Long term memory
Attention
Rehearsal

Memory Recall

Information is normally held in an **inactive** state in memory.

To use information it must first be **activated**.

Therefore, knowledge recall from memory is a process of **activating information**.

Memory Recall

Familiar or well-known information is retrieved from long-term memory **more quickly** than unfamiliar or poorly known information.

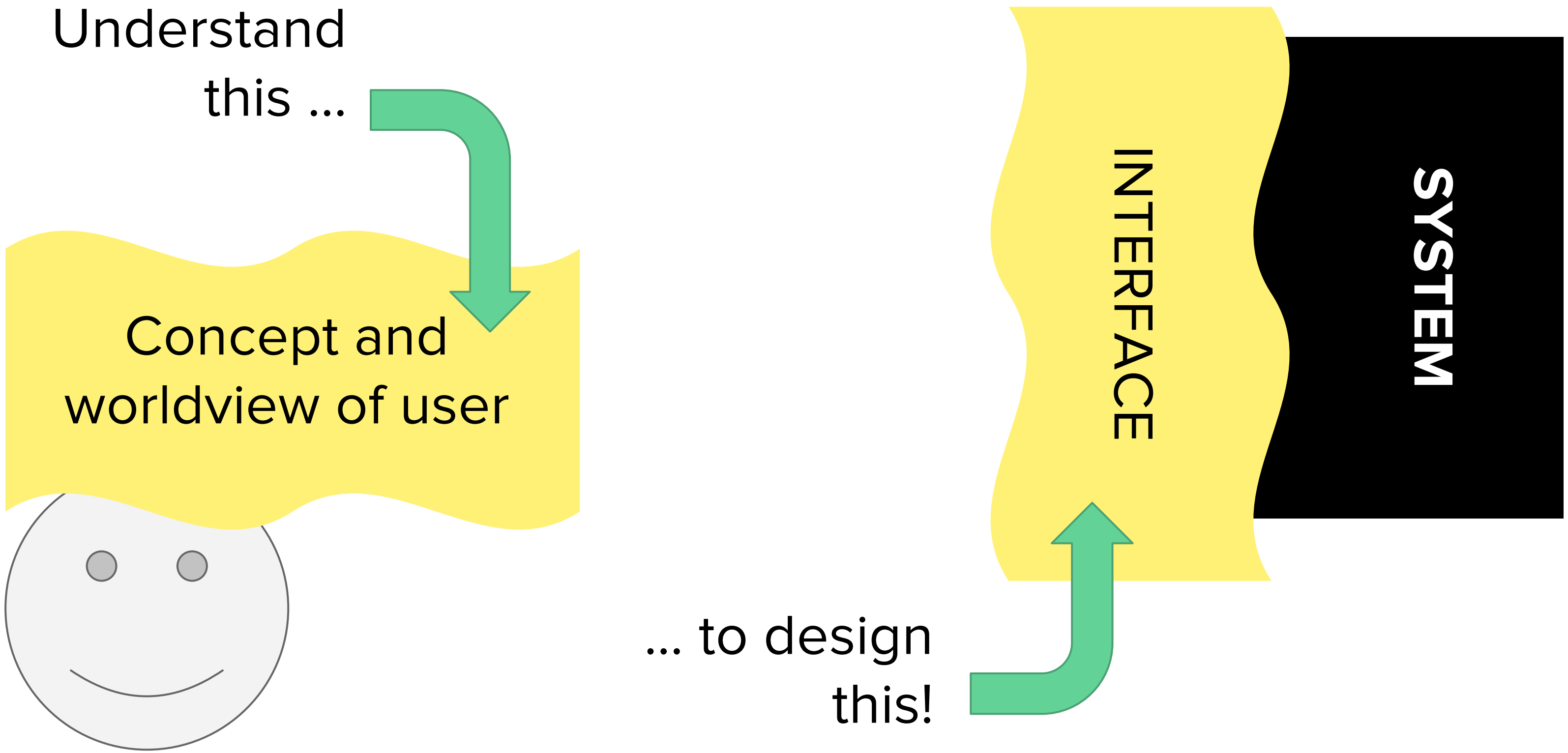
In other words, knowledge is **structured** in memory.

How does this relate to design?

We can’t redesign people.

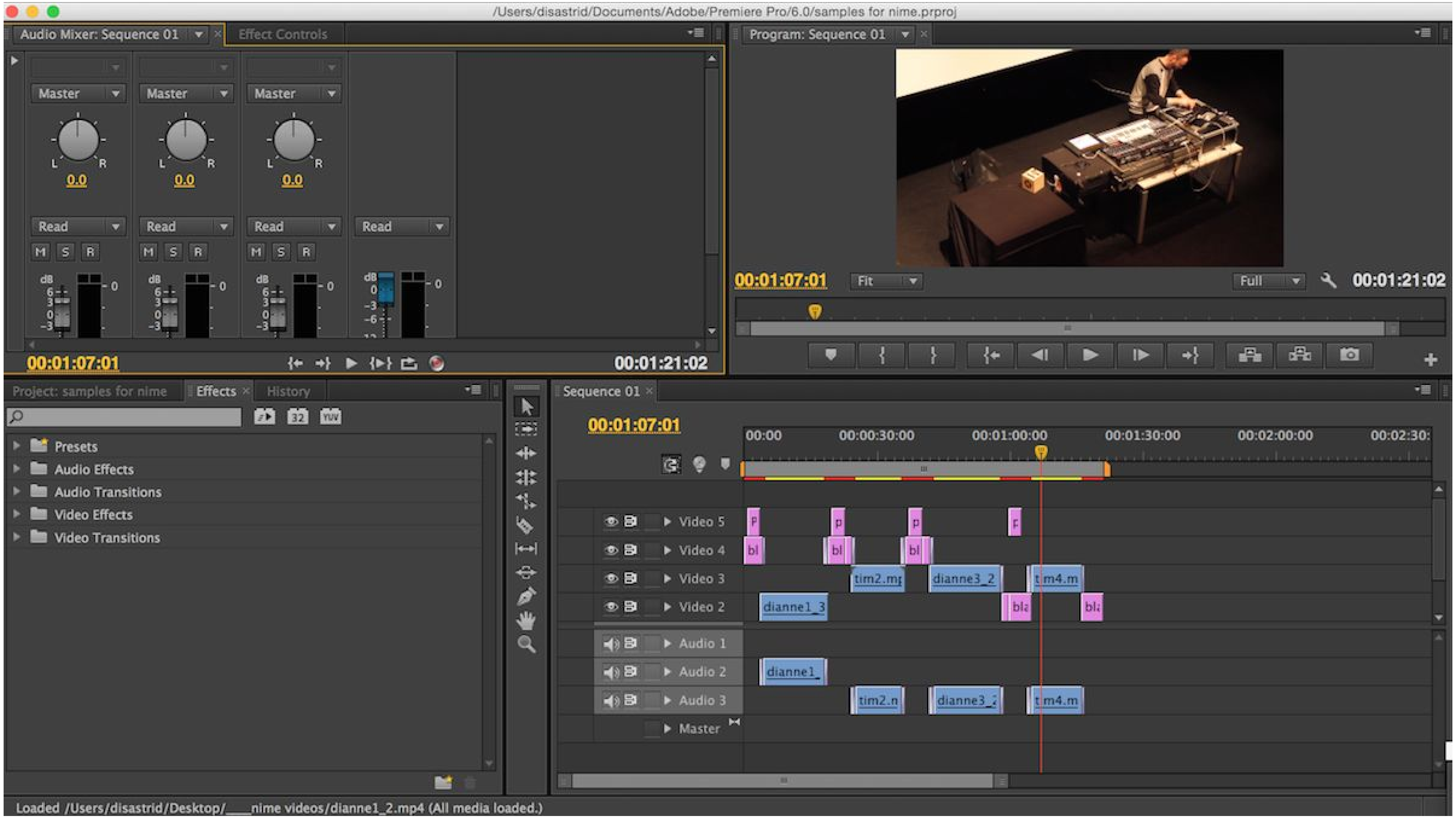
We *can* redesign the interface to support communication.

To do this, we have to understand people’s **limits**.


Understand
this …
Concept and
worldview of user
INTERFACE
SYSTEM
… to design
this!

We also need to understand their **mental models.**

UPPER
LOWER
Timer 1
Touch to begin
Timer 2
Touch to begin
BACK
CANCEL UPPER
START CONTINUE
CANCEL LOWER
Quick Start: Upper Oven, Pure Convection @ 350°


/Users/disastrid/Documents/Adobe/Premiere Pro/6.0/samples for nime.prproj

Loaded /Users/disastrid/Desktop/____nime videos/dianne1_2.mp4 (All media loaded.)

To understand what steps are involved in a task, we use **task analysis.**

Task analysis

Task analysis involves:

- Working out people’s goals
- Figuring out what they are doing to achieve them (tasks)
- Modeling the tasks

Task analysis

Task models are used to inform design, based on ideas of:

- How memory works
- How perception works (for design)

Task analysis: why bother?

We need to understand how people currently perform tasks.

- This informs design – the system must match the user’s tasks

System will fail if:

- It doesn’t do what the users want
- It is inappropriate for the user

What is task analysis?

Task analysis is a process of:

- Finding out who the users are
- Finding out what task they perform
- Creating models of the task
- Creating scenarios of use

… Then try out ideas before building system

Task Analysis: Activities

1. **Find out who the users are.**

- Their background
- Their skills
- Their work habits and preferences
- Any relevant physical characteristics (height, eye sight...)
- What else is relevant?

Task Analysis: Activities

2. Find out what tasks they perform.

- Talk to them (interviews, questionnaires ...)
- Read about it (studies, manuals, journals ....)
- Observe them in public (take notes)
- Observe them in the lab (get them to think out loud while they perform a task or report what they're doing on video)

Task Analysis: Activities

3. **Find out where the tasks happen.**

- Physical location (office, home, etc)
- What’s the environment like? Is it a high stress environment, like an ambulance or in front of a crowd? Or a low stress environment, like at home on the sofa?)
- Features of the environment (lighting, noise level, are their hands wet, do they need confidentiality?)

Task Analysis: Activities

4. **Model the tasks.**

- Which tasks are important?
- Identify the **goals**
- Identify the **actions** needed to meet the goals
- Identify the **sequential dependencies** (ie, at an ATM the pin needs to be entered before cash can be withdrawn)

Creating a task model:
Task decomposition

- Break high-level (macro) tasks into their constituent (micro) parts (ask, “How is this done?”)
- At the lower level show task flows, decision processes, even screen layouts
- If more sub tasks are found at the lower level, ask “Why is this done?” Can the process be improved? Is this task necessary? Should it be moved up the structure?
- Show the sequencing of activities by ordering from left to right

Creating a task model: Task decomposition

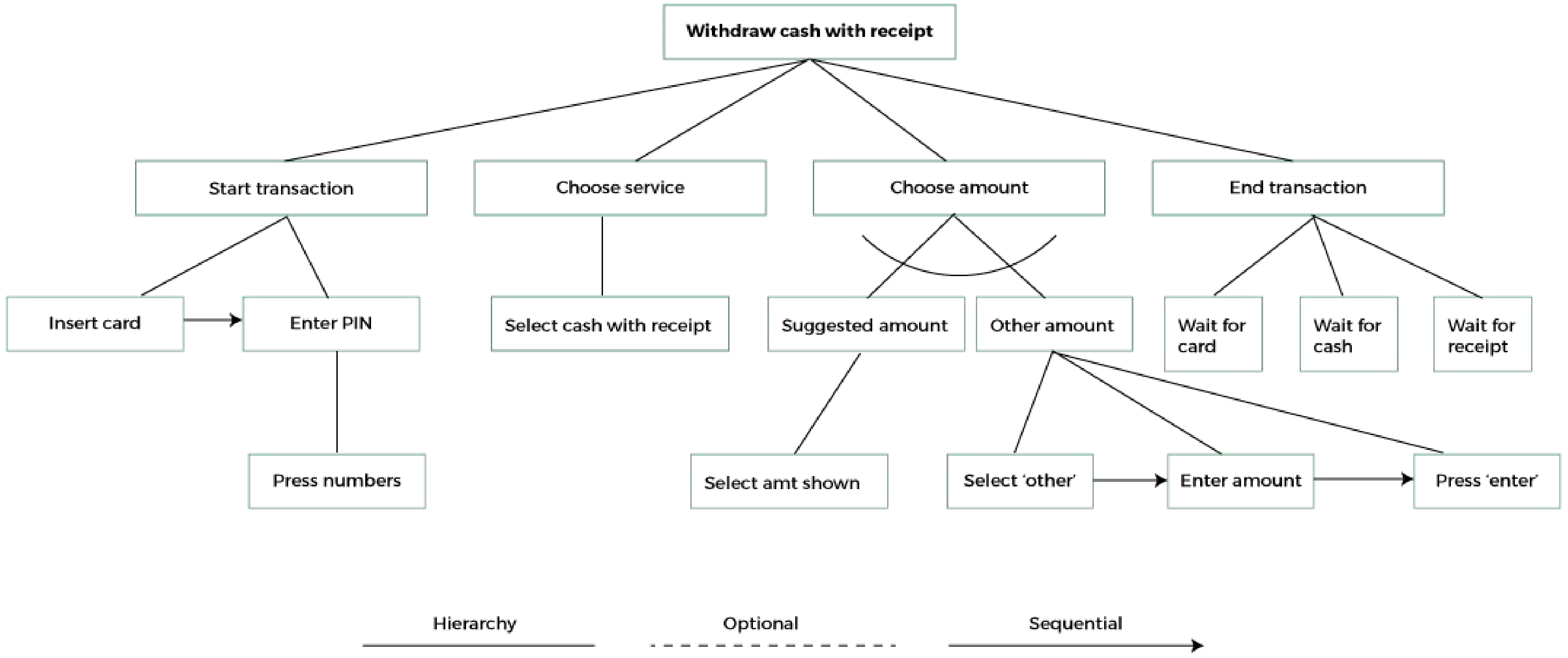
1. Identify the task to be analysed.
2. Break it into sub-tasks.
3. Draw the subtasks as a layered diagram. **Make sure it is complete.**
4. Decide upon the level of detail for decomposition. Are you going too far? Not far enough?
5. **Optional but helpful**: Give the diagram to someone who has not been involved in your discussions, but who knows the task. Do they think it's consistent?

Creating a task model: Task flow diagrams

These diagrams document the details of specific tasks.

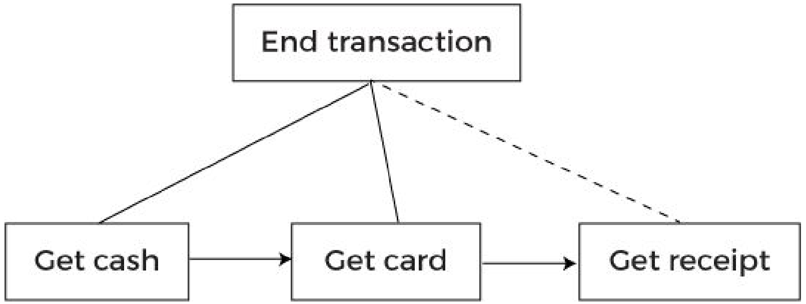
They not only show the specific details of current work processes, but may also highlight areas of certain tasks that need attention, such as:

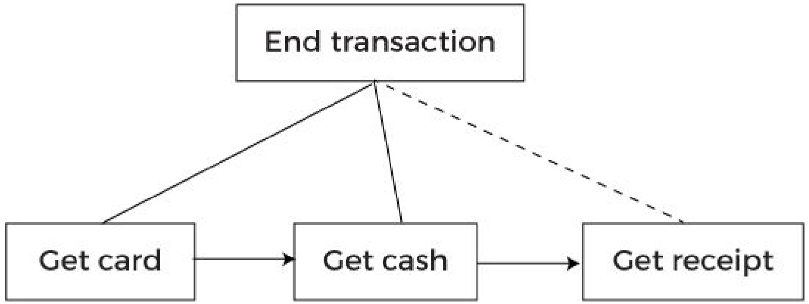
- Those where the task process is poorly understood
- Those where the process is carried out differently by different people
- Those that are inconsistent with the higher level processes in the structure


Withdraw cash with receipt
Start transaction
Choose service
Choose amount
End transaction
Insert card
Enter PIN
Select cash with receipt
Suggested amount
Other amount
Wait for card
Wait for cash
Wait for receipt
Press numbers
Select amt shown
Select 'other'
Enter amount
Press 'enter'
Hierarchy
Optional
Sequential

Why look at the order of tasks?

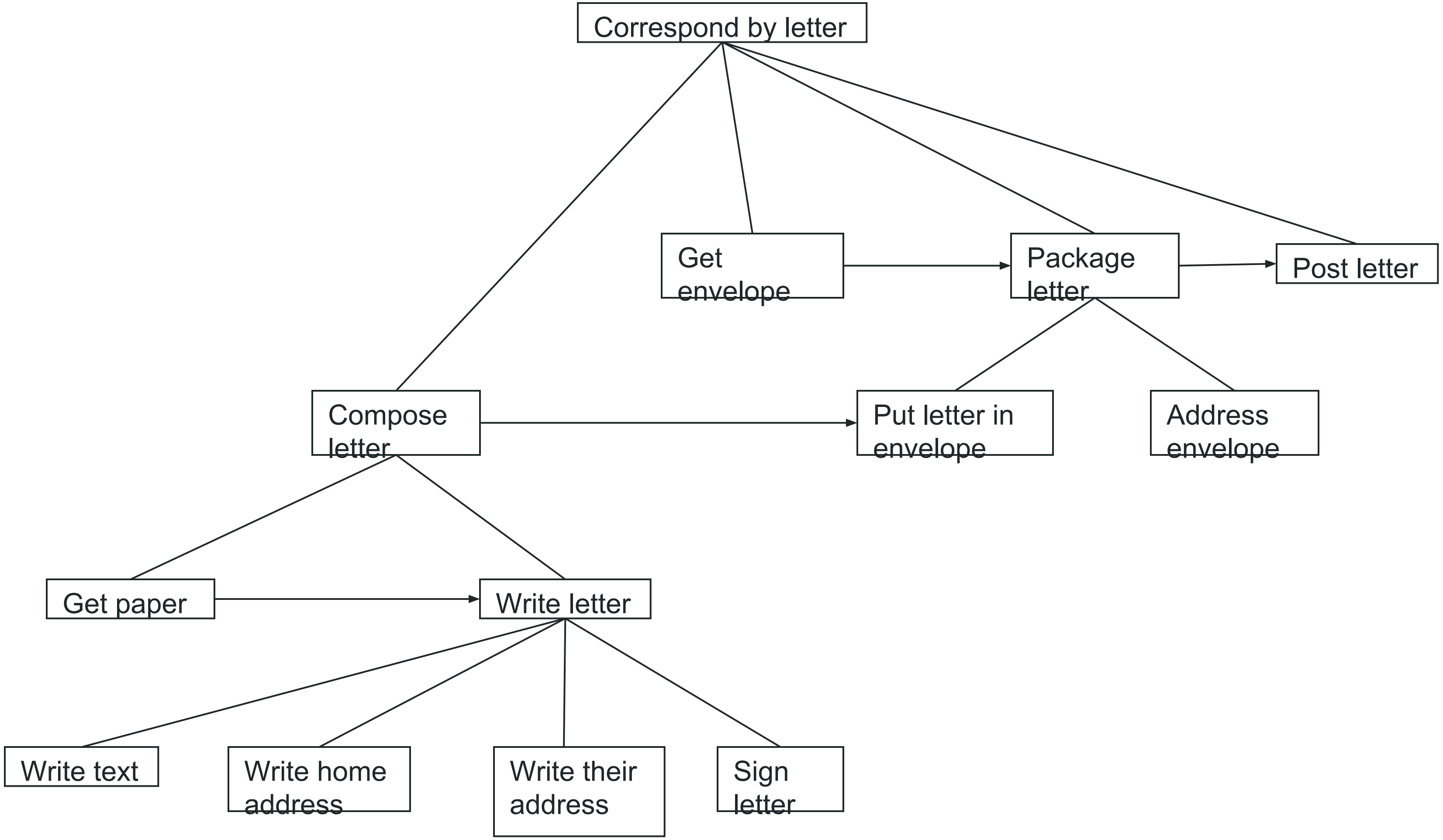
There can be a big difference between two sequences!.



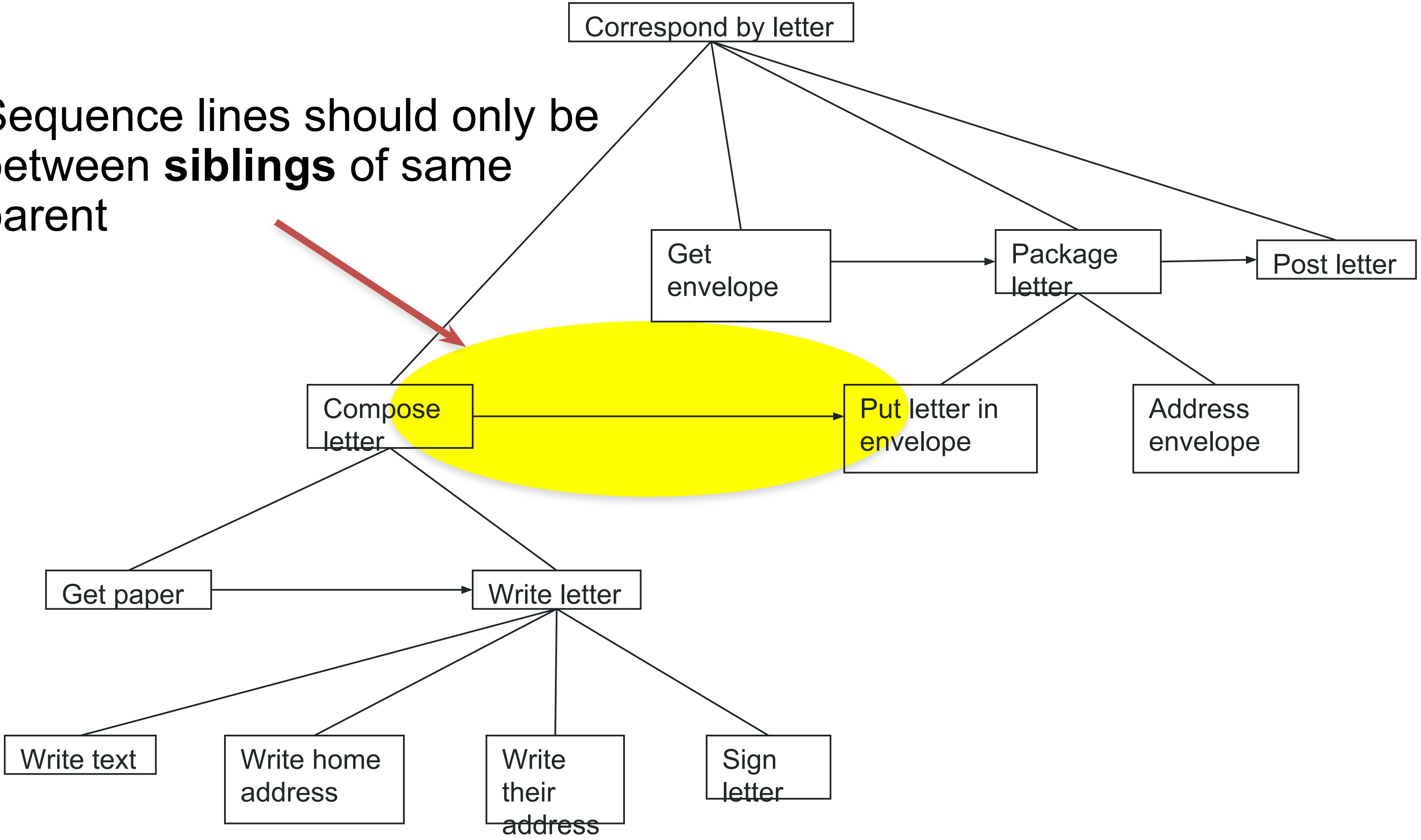


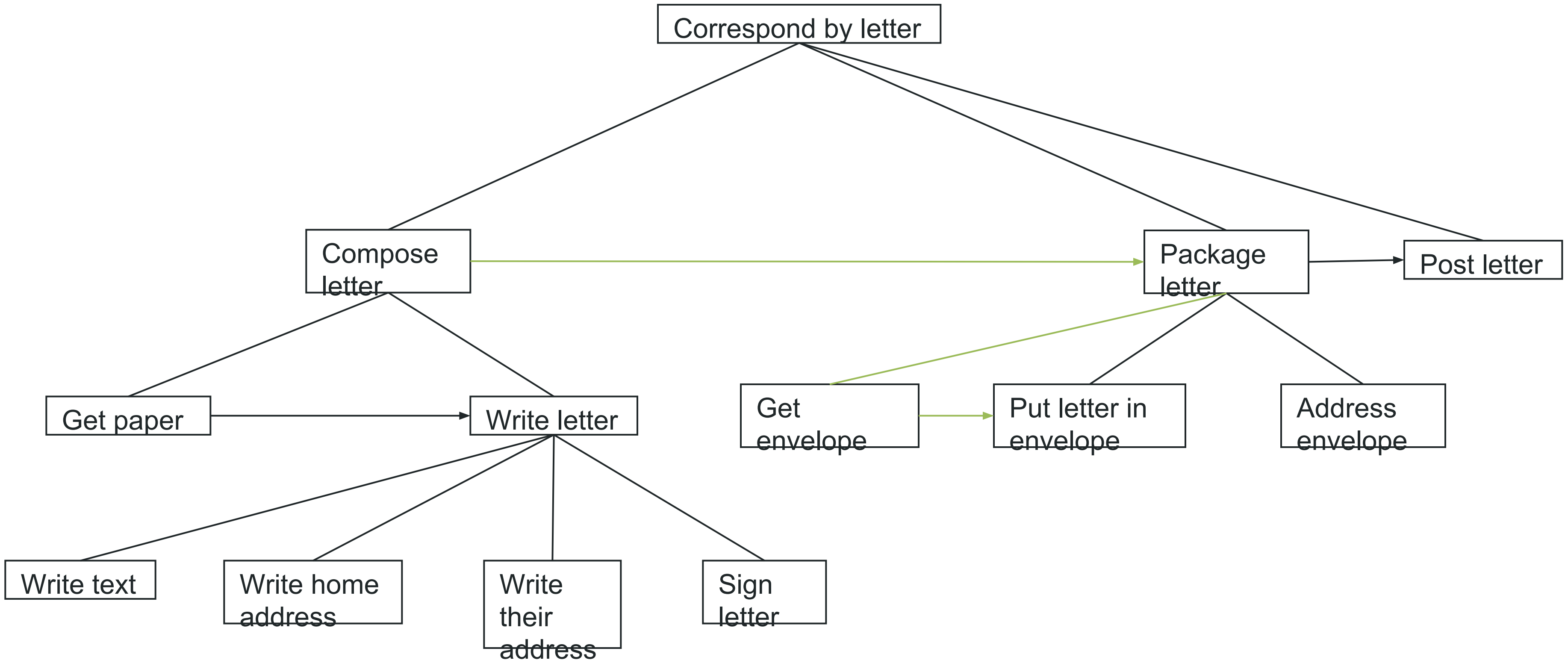
Relation Between Ideal Task and Task Realisation

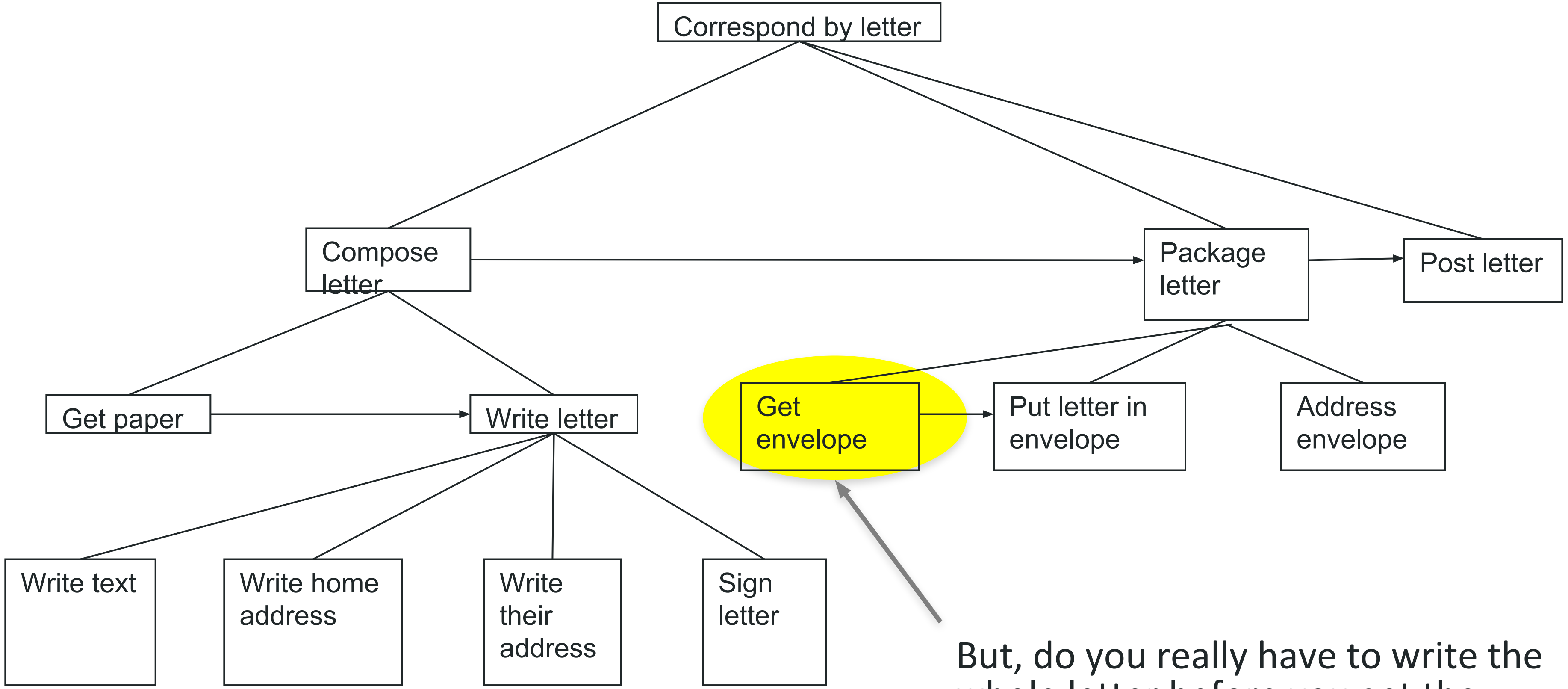
What’s the process of sending a letter?


Correspond by letter
Get envelope
Package letter
Post letter
Compose letter
Put letter in envelope
Address envelope
Get paper
Write letter
Write text
Write home address
Write their address
Sign letter

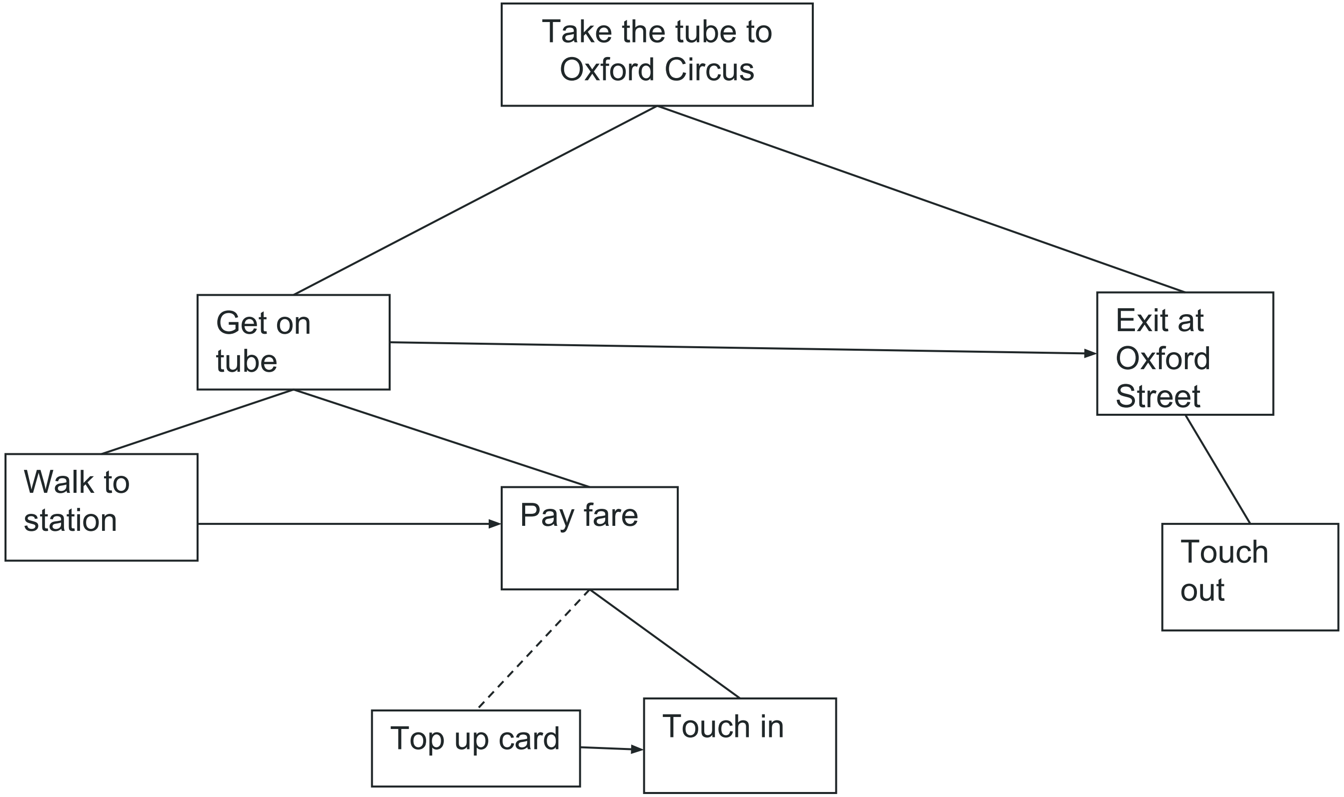
Sequence lines should only be between **siblings** of same parent




Correspond by letter
Compose letter
Package letter
Post letter
Get paper
Write letter
Get envelope
Put letter in envelope
Address envelope
Write text
Write home address
Write their address
Sign letter



But, do you really have to write the whole letter before you get the envelope?



Where is it incomplete? Are there mistakes?

Task model advantages:

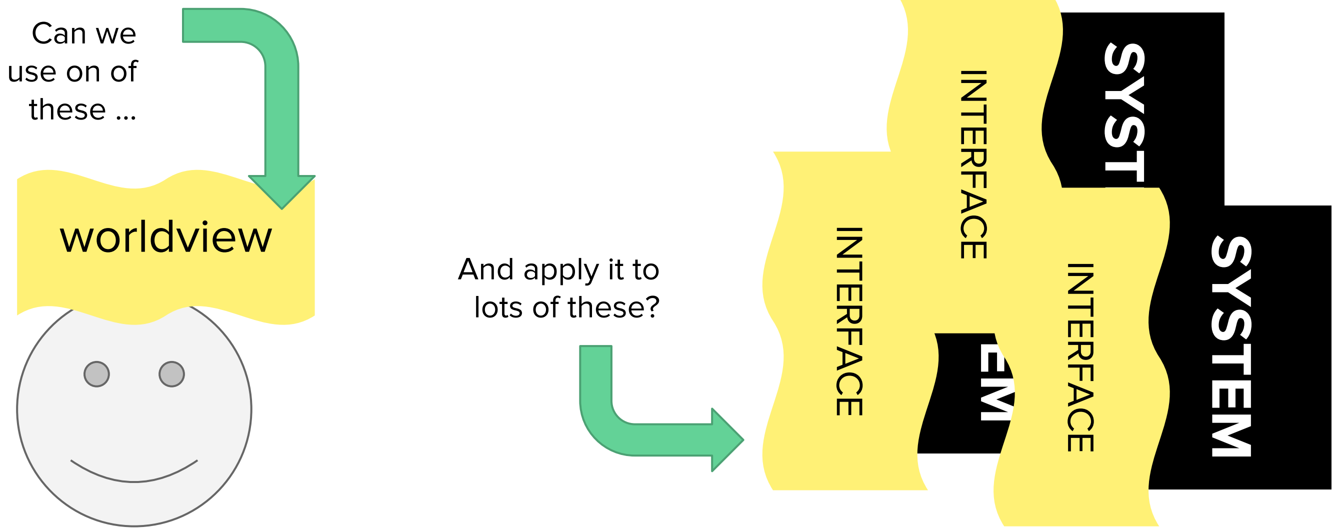
+ They allow us to really dig into tasks and understand what’s involved
+ They’re great for processes that will never vary (ideal tasks)

Task model disadvantages:

- They don’t allow for overlapping tasks, or multitasking
- They assume the task is done the same way every time

Generalisation

Once we understand the process and the mindset of the person using the device, how far can we apply it?



Critiques of generalisation

- Doesn’t scale well or predictably
- Tends to concentrate on how things are done already
- Doesn’t allow for:
    - Modelling overlapping tasks
    - Modelling interruptions
    - Learning, thereby changing how the task is done

In summary:

- The steps taken to complete something are important! We have to understand what we’re asking a person to do.
- Task analysis models the structure of tasks.
- They are hierarchical, and show a sequence of steps
- Task models deal with *idealised* tasks

In summary:

- Task models allow for very fine-grained analysis
- They can be good for identifying problems
  - ie, sequences that are too long, tasks that are complicated and have to be designed well
- But they’re not very good for overlap, or tasks that can vary in sequence