Módulo 3

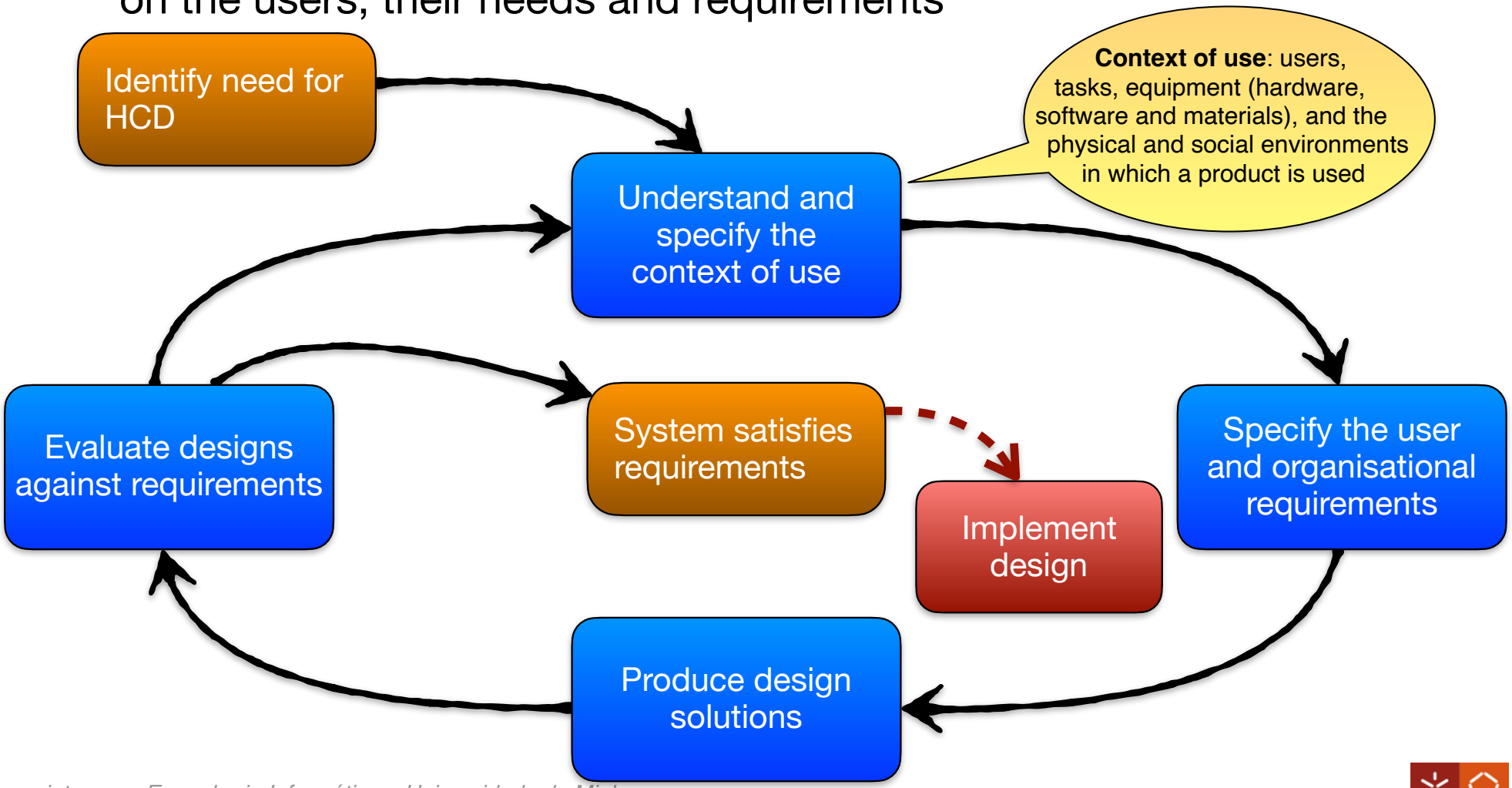
OS UTILIZADORES

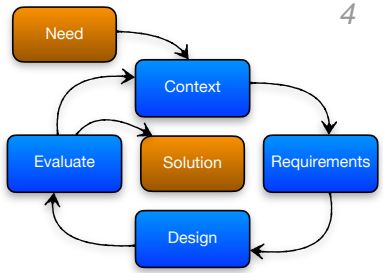
Sumário

- HCD — Concepção centrada no utilizador
- Definição de Usabilidade
- Modelo de Interação de Norman
- Erros do utilizador vs. Erros de utilização
- Keystroke Level Analysis (KLA)

Human-centred design (HCD) ISO 9241-210:2019

- An approach that aims to make systems usable and useful by focusing on the users, their needs and requirements

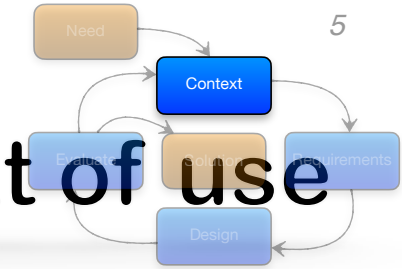




Human-centred design (HCD)

- Key principles:
  1. **The design is based upon an explicit understanding of users, tasks, and environments.**
     - custom-made vs. generic or consumer products
     - appropriate allocation of function between users and technology
  2. **Users are involved throughout design and development.**
     - valuable source of knowledge about the context of use, the tasks, and how users are likely to work with the future product or system
  3. **The design is driven and refined by user-centred evaluation.**
     - feedback from users becomes a critical source of information
  4. **The process is iterative.**
     - preliminary design solutions tested against “real world” scenarios, and the results fed back into progressively refined solutions
  5. **The design addresses the whole user experience.**
     - bringing users into the design process to ensure a specific user experience
  6. **The design team includes multidisciplinary skills and perspectives.**
     - teams do not have to be large but the team should be sufficiently diverse to make appropriate design trade-off decisions
     - Individual team members can cover a number of different skill areas and viewpoints

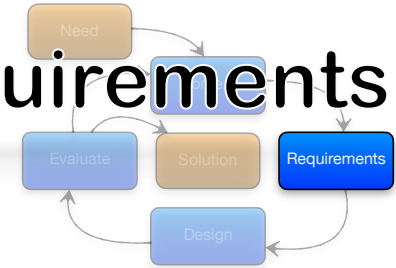
E.g.
end-user; purchaser, manager of user; application domain specialist, business analyst; systems analyst, systems engineer, programmer; marketer, salesperson; user interface designer, visual designer; human factors and ergonomics expert, human-computer interaction specialist; technical author, trainer and support personnel.



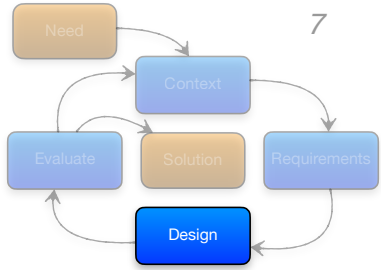
Understand and specify the context of use

- Description of the relevant characteristics of the users, tasks and environment that impact the system design
- The context of use description should
  - specify the range of intended users, tasks and environments in sufficient detail to support design activity;
  - be derived from suitable sources;
  - be confirmed by the users or if they are not available, by those representing their interests in the process;
  - be adequately documented;
  - be made available to the design team at appropriate times and in appropriate forms to support design activities.

Specify the user and organisational requirements



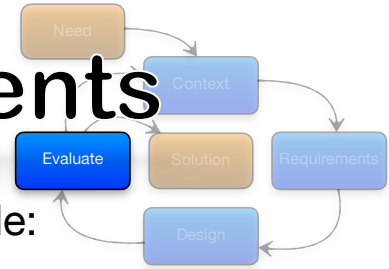
- Complements specification of functional requirements
- This specification should define the “allocation of function”
  - the division of system tasks into those performed by humans and those performed by technology
- The specification of user and organisational requirements should:
  - identify the range of relevant users and other personnel in the design,
  - provide a clear statement of the human-centred design goals,
  - set appropriate priorities for the different requirements,
  - provide measurable criteria against which the emerging design can be tested,
  - be confirmed by the users or those representing their interests in the process,
  - include any statutory or legislative requirements, and
  - be adequately documented.



Produce design solutions

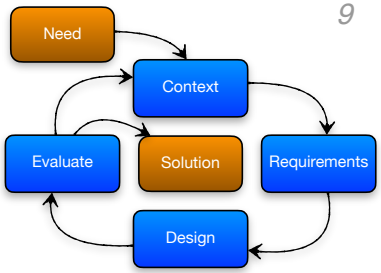
- use existing knowledge to develop design proposals with multi-disciplinary input
  - user interface guidelines, similar products, standards, etc.
- make the design solutions more concrete using prototypes (simulations, mock-ups, etc.)
  - more effective communication with users
  - reduced need and cost of reworking products later in the life cycle
- present the design solutions to users and allow them to perform tasks (or simulate tasks)
  - prototypes are not simply to show users a preview of the design, they are used to collect user feedback (comments, difficulties)
- alter the design in response to the user feedback and iterate this process until the human-centred design goals are met
  - user feedback provides guidance on functional design changes to improve usability
  - feedback can also help to refine the scope and purpose of the interactive system
- manage the iteration of design solutions
  - The results of the above activities should be recorded

Evaluate designs against requirements



- An essential step in HCD, should take place at all stages in the system life cycle:
  - to guide design
    - how well the system meets its goals; potential problems and need for improvements; select design options; elicit further requirements from the users
  - to assess whether objectives have been achieved
    - demonstrate that a design meets the human-centred requirements
    - assess conformity to standards
  - to monitor long-term use of the system
    - collecting user input over a period of time
    - some effects are not recognisable until the system has been used for a period of time
    - there may be effects which result from external factors (e.g. unforeseen changes in working practices)
- It is important to start evaluation as early as possible
  - The longer the process has progressed, the more expensive the introduction of changes is
- expert vs. user-based evaluation

HCD advantages



- Making systems more usable can contribute to:
  - systems that are easier to understand and use, thus **reducing training and support costs**,
  - **improved user satisfaction** and reduced discomfort and stress,
  - **improved productivity** and operational efficiency of users and organisations, and
  - **improved product quality and appeal** to the users — **a competitive advantage**

Usabilidade

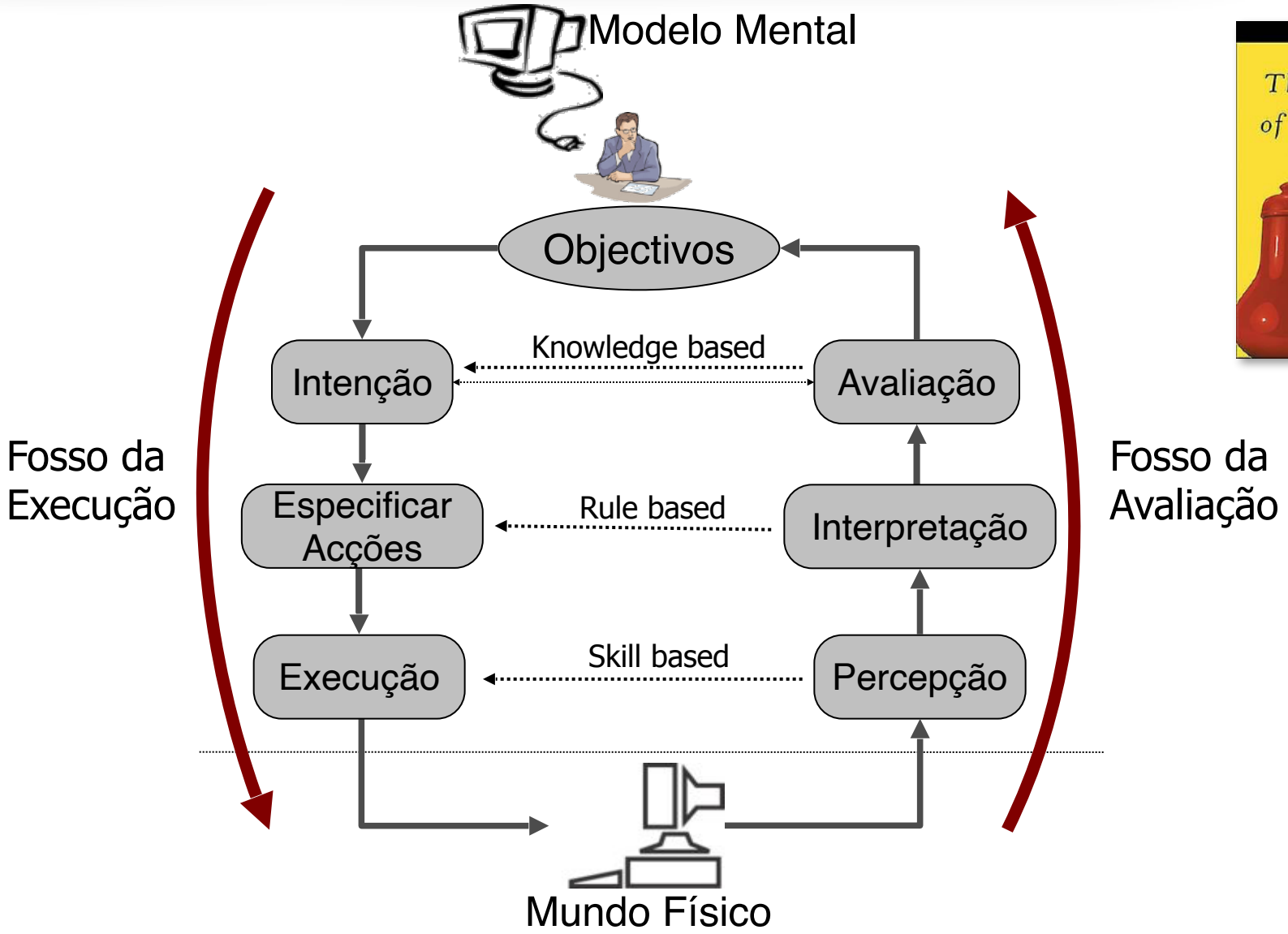
"Extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use." (ISO 9241-11).

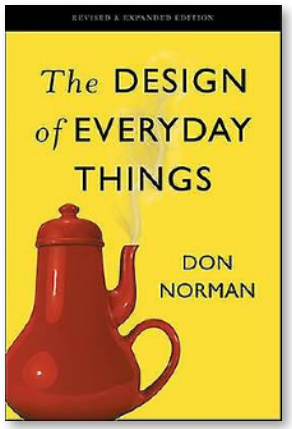
- effectiveness - accuracy and completeness with which users achieve specified goals;
- efficiency - resources used in relation to the results achieve (time, human effort, cost, materials, …)
- satisfaction - extent to which the user's physical, cognitive and emotional responses that result from the use of a system, product or service meet the user's needs and expectations

É necessário compreender

- Os utilizadores
  - A usabilidade deve ser definida em relação a um tipo específico de utilizador — mas existem características gerais dos humanos que podem/devem ser consideradas.
- As actividades que pretendem/devem realizar
  - A usabilidade deve ser definida para tarefas específicas que o sistema deve suportar — no entanto, o sistema acabará muitas vezes por ser utilizado de formas não previstas inicialmente.
- O contexto em que o devem fazer
  - O contexto em que o sistema vai ser utilizado pode influenciar não só a usabilidade do sistema, mas a forma como os testes podem decorrer.

Modelo de Interacção (de Norman)



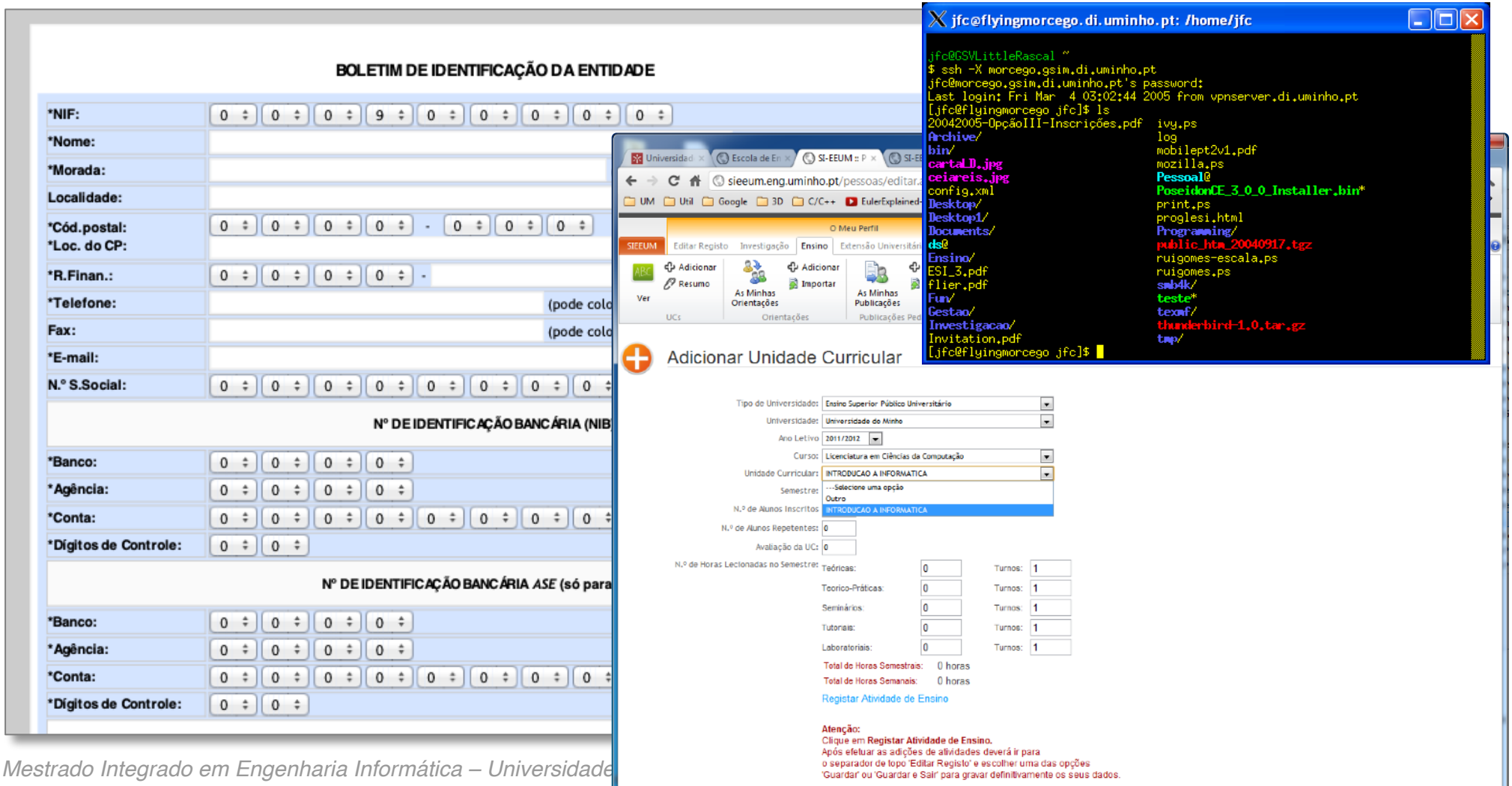


Modelo de Interacção de Norman

- **Fosso da Execução**
  - Esforço que o utilizador tem que realizar para efectuar determinada tarefa.
  - Distância entre os objectivos do utilizador e a forma como pode atingi-los.
  - Atenção à definição das tarefas!
- **Fosso da Avaliação**
  - Esforço que o utilizador tem que realizar para perceber a interface.
  - Distância entre a informação que a interface fornece (e de que forma) e aquela que o utilizador pretende.
  - Atenção ao modo como a informação é apresentada!

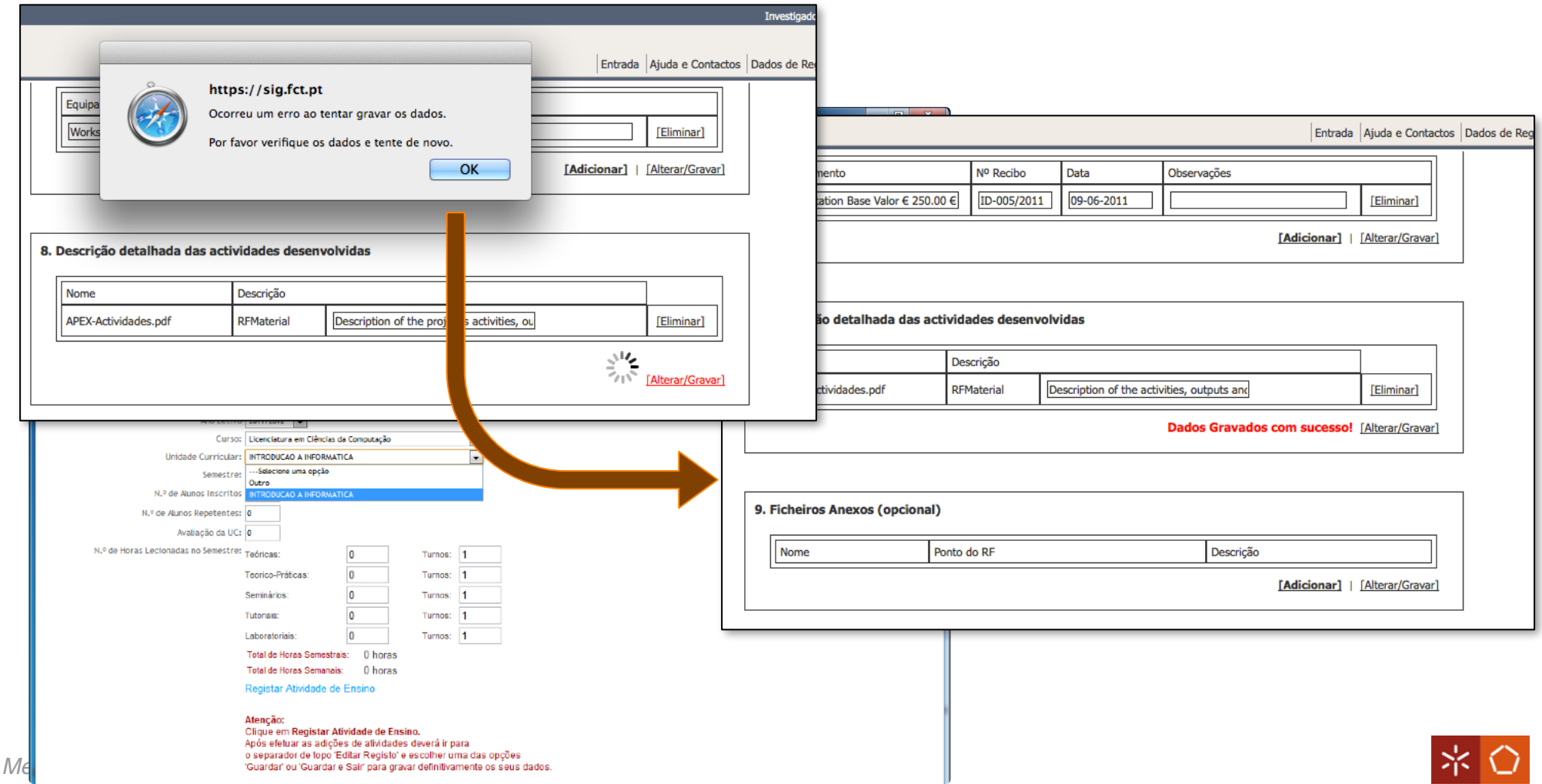
Fosso da Execução

- O esforço necessário para realizar o que se pretende

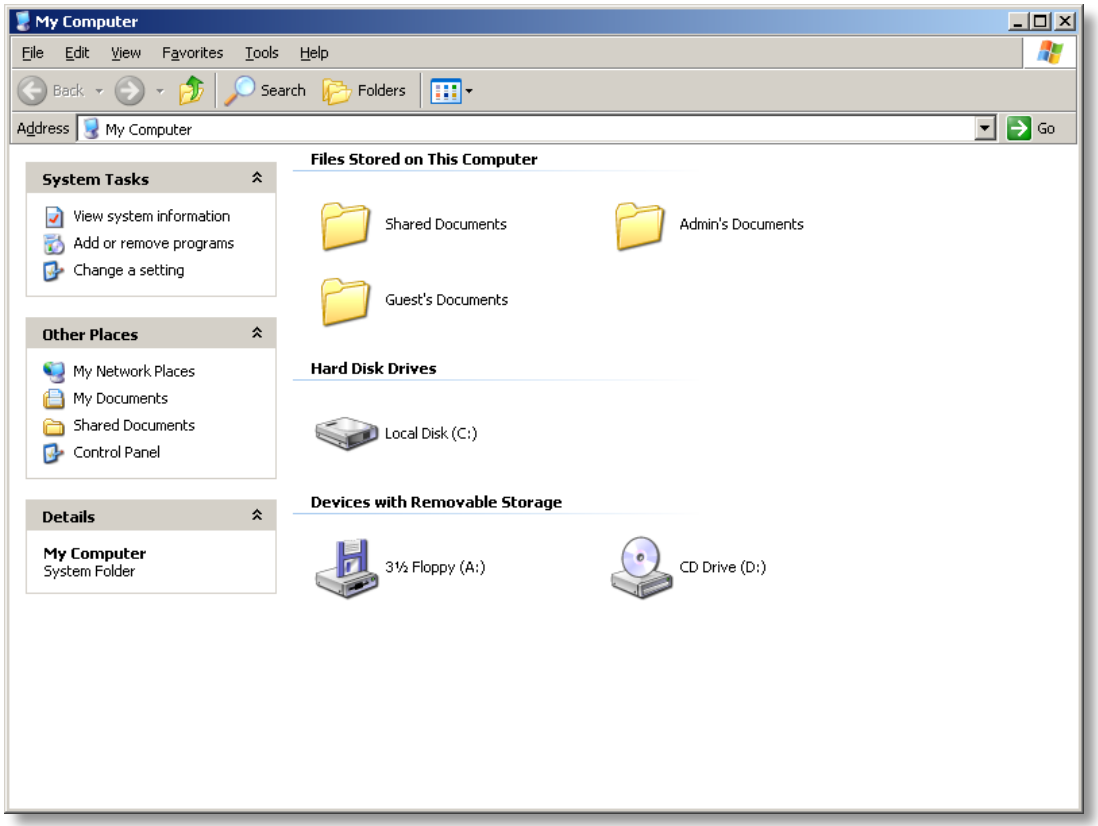


Fosso da Avaliação

- O esforço necessário para perceber o estado do sistema



Outro exemplo



- Necessitamos de métodos e ferramentas que nos permitam desenhar Sistemas Interactivos com qualidade (i.e. usabilidade)

~~User error~~ vs. Use errors

- Reason's error classification
  - **Slips**
    - errors due to actions not carried out as intended
  - **Lapses**
    - errors due to missed actions
  - **Mistakes**
    - errors due to erroneous action plans
- Understanding these types of errors helps create better interfaces that can help prevent or minimize the impact of ~~user~~ **use errors**

Reason's error classification

- Slips — errors due to actions not carried out as intended
  - The user intends to do one thing but ends up doing something else
  - Causes:
    - automatic or habitual behaviour
    - user distracted / in a hurry
  - Example:

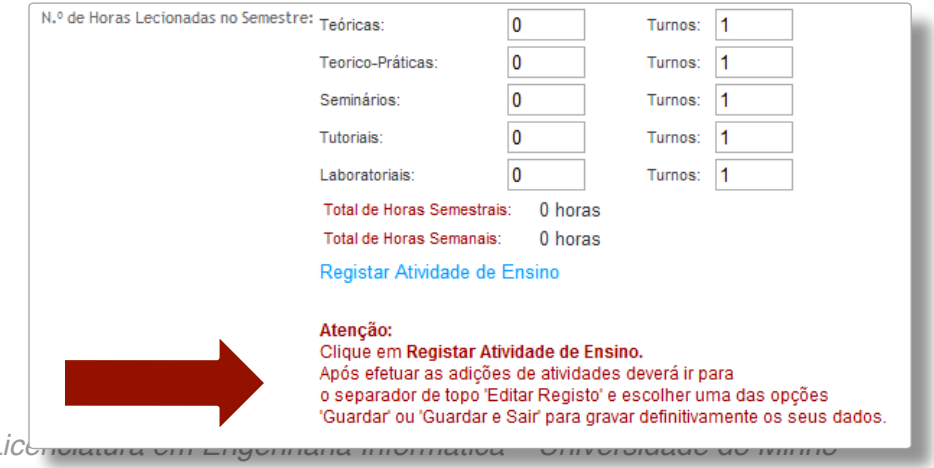
Reason's error classification

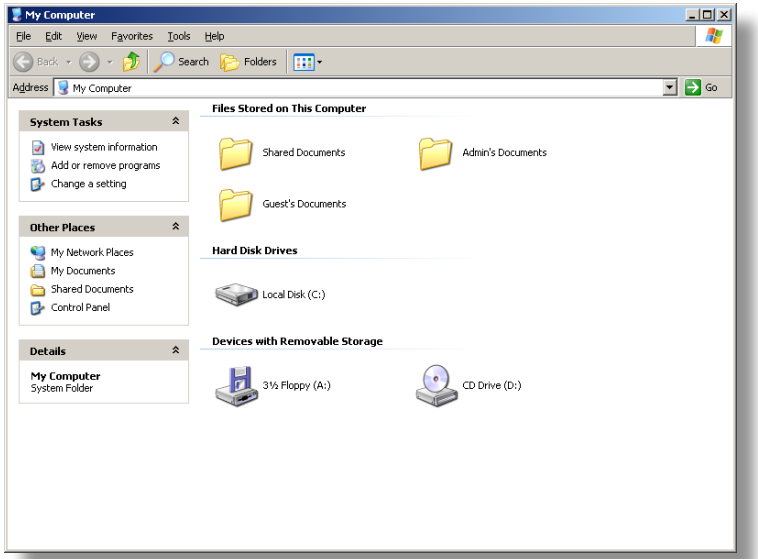
- Lapses - errors due to missed actions
  - The user misses (does not perform) relevant actions
  - Causes:
    - temporary failure of concentration, memory, or judgement.
  - Example:
    - Post-completion error — missing a final step in a procedure, after the goal is achieved.

https://wiki.ucl.ac.uk/display/UCLICACS/Human+Error

Reason's error classification

- Mistakes — errors due to erroneous action plans
  - The user has a goal in mind but selects an inappropriate or incorrect action (plan) to achieve that goal
  - Causes:
    - user lacks the necessary knowledge or understanding of the system or task
  - Examples:





Helping prevent/minimize the impact of errors

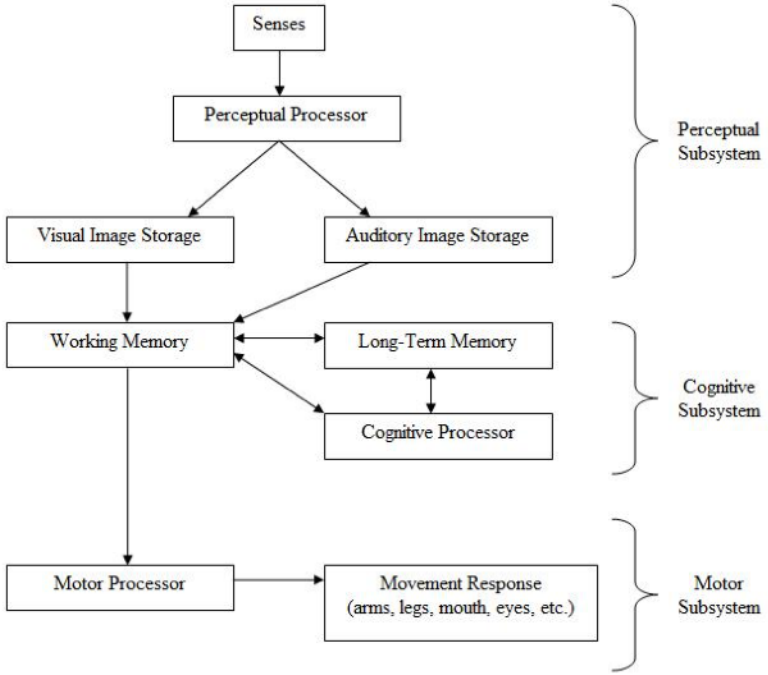
- **Slips**
  - Reduce the number of steps required to complete a task
  - Provide clear and distinct visual cues
  - Implemente undo/redo functionality
  - Include confirmation prompts
- **Lapses**
  - Provide clear and visible reminders
  - Simplify the user interface
  - Provide help and support
  - Provide defaults
- **Mistakes**
  - Simplify the user interface
  - Provide clear and concise instructions
  - Use consistent and familiar design patterns:
  - Use feedback and error prevention mechanisms

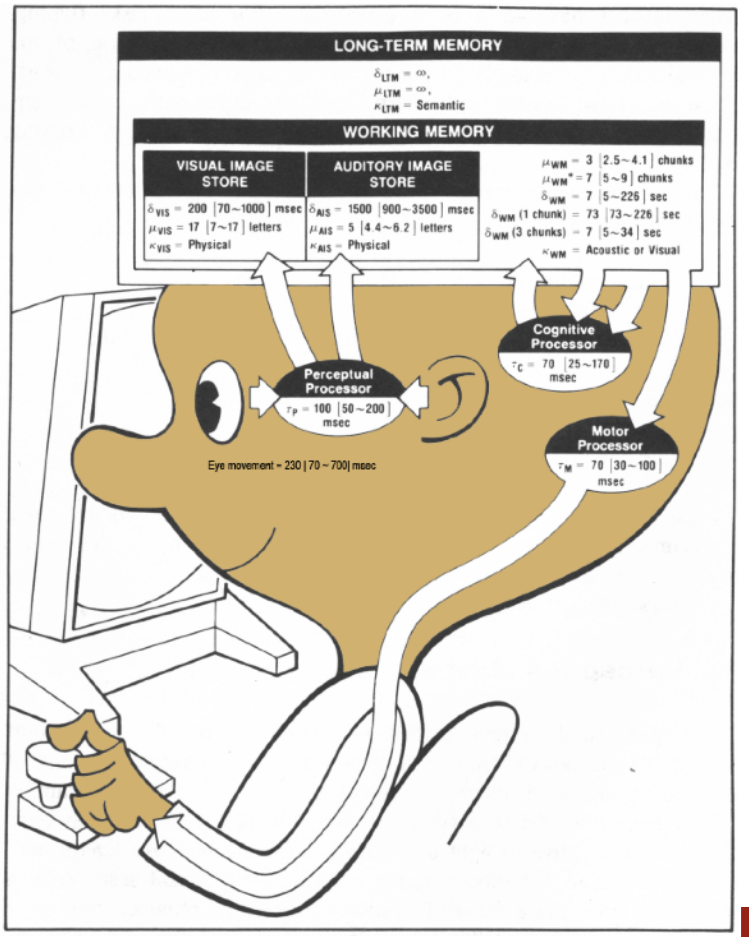
Understanding/Modelling the User

- **Empirical laws**
  - Predict human performance
- **Cognitive Models**
  - Capture the decision and/or execution process that enables the achievement of a given goal
  - Example: GOMS
    - Hierarchies of goals and tasks
    - Assumes we solve problems in a top-down "divide and conquer" strategy
- **Cognitive Architectures**
  - The user as an information processing machine
  - Example: Model Human Processor
- Tools based on these approaches

Arquitecturas cognitivas

- Capturam os resultados da Psicologia Cognitiva num modelo computacional
  - Como se estrutura a nossa mente
  - Como os diferentes componentes trabalham em conjunto
- **Model Human Processor (MHP)**





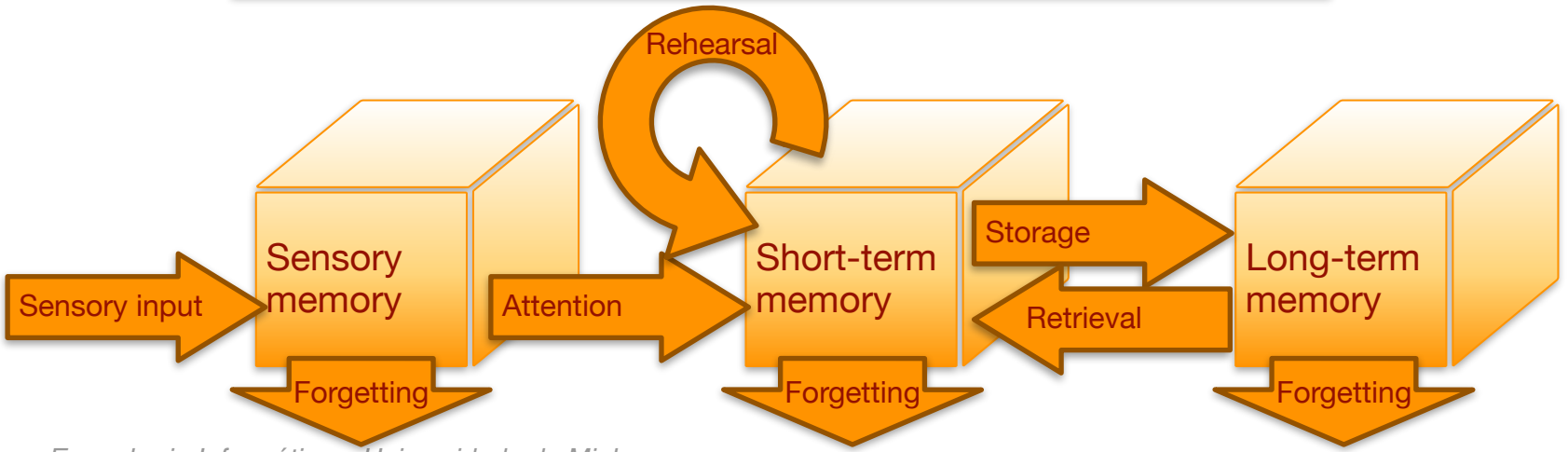
Card, Moran and Newell

Empirical laws

- Derived from observations/study of human behaviour
- As they can only be formulated for very concrete situations, empirical laws normally apply to isolated actions
- Qualitative predictions of human performance
- Some laws:
  - **Miller's Law** — short-term memory
  - **Hick-Hyman law** — decision making
  - **Fitts Law** — movement to hit a target

Miller's law (of short-term memory)

- Most people can only hold about seven pieces of information in their short-term memory at a time — the magic number: 7 ± 2
  - The exact number can vary greatly from person to person and with the type of information involved
- If more information is received than can be processed we have *information overload*
  - We can increase capacity by "chunking" (aggregating) information into meaningful groups — e.g. phone numbers (253604447 vs. 253 60 44 47)
- Note: This law should not be applied to decide the number of items in a menu
  - the selection task does not require people to hold the options in memory.



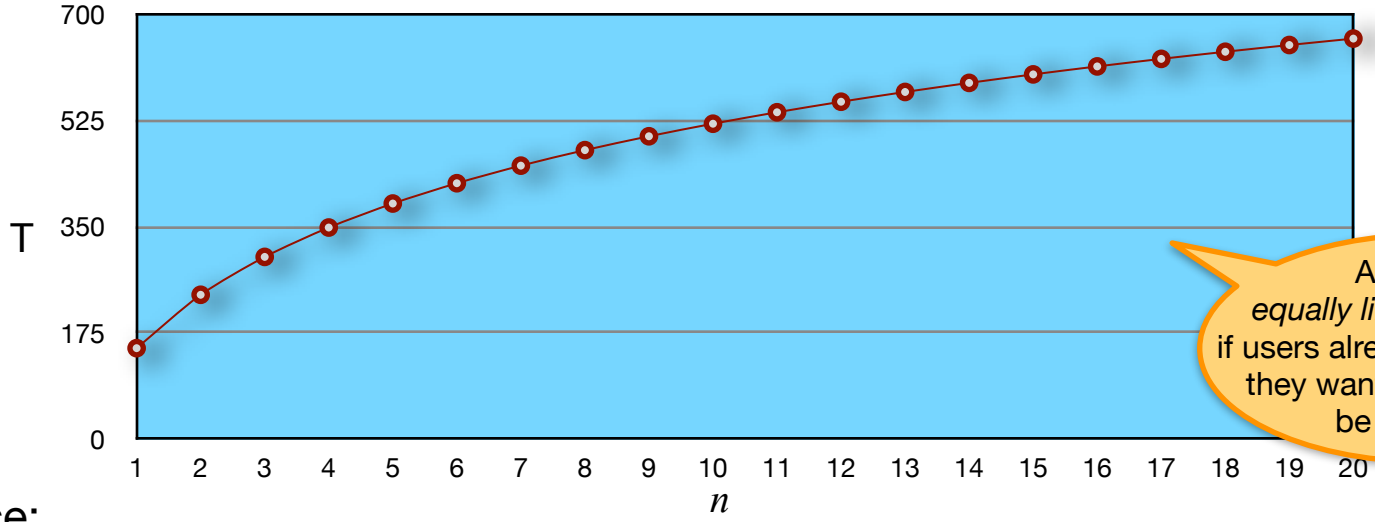
Hick-Hyman law

People tend to group alternatives into categories and eliminate about half of them at a time (instead of considering each alternative in isolation)

- The time (T) it takes to make a decision is a function of the number (n) of choices available

$$T = k \cdot \log_2(n+1) \quad [k \approx 150\text{msec}]$$

- The more alternatives a person is presented with, the longer it takes to make a decision:



Assumes *equally likely choices* — if users already know what they want to do, time will be shorter

- Relevance:
  - Reducing the number of choices can make decision-making easier and more efficient — **avoid information saturation**
  - More relevant for small lists (e.g. menu vs contacts list)

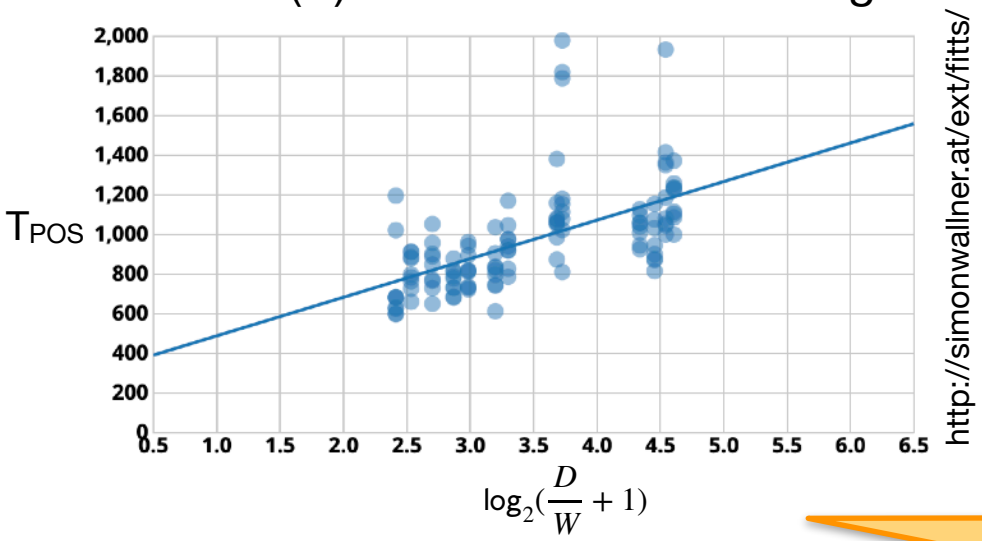
Fitts Law

a and b must be determined. a — the time the time it takes to prepare for and initiate the movement; b — the difficulty of the task (the rate at which time increases as the target gets smaller/ the distance larger)

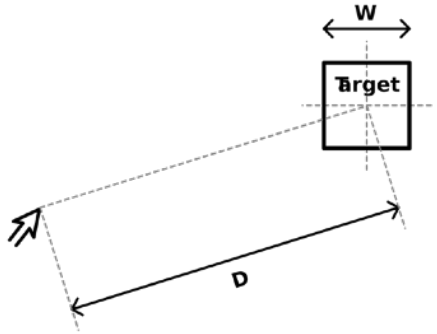
- The time (T_{POS}) required to move a pointer (e.g. the mouse) to a target is a function of the distance to the target divided by its size

$$T_{POS} = a + b \cdot \log_2(\frac{D}{W} + 1)$$

  - the greater the distance (D) and the smaller the target width (W), the more time is required:



http://simonwallner.at/ext/fitts/



Difficulty index (ID) as defined by ISO 9241-9 (Shannon formulation). Originally was: $\log_2(2D/W)$

- Relevance:
  - User interface elements must have an adequate size
  - User interface commands must be logically grouped

Keystroke Level Analysis (KLA)

- Enables the analysis of performance in the execution of known tasks, at the physical level of the device.
- Based on empirical knowledge of the human psychomotor system.
- Useful to compare the expected performance of alternative operation methods.
  - Each method is divided into operations (cf. GOMS)
  - Each type of operator has an associated execution time (cf. Fitts, etc.)
  - Heuristics are used to introduce "mental preparation" operations (pauses)
  - The times of the operators are added up

KLA

Windows, **I**cons, **M**enus and **P**ointer
(base paradigm for **G**raphical **U**ser **I**nterfaces — **GUI**)

- Operators for WIMP interfaces

Code	Operation		Time
K	Key press and release (keyboard)	Best Typist (135 wpm)	0.08 seconds
		Good Typist (90 wpm)	0.12 seconds
		Poor Typist (40 wpm)	0.28 seconds
		Average Skilled Typist (55 wpm)	0.20 seconds
		Average Non-secretary Typist (40 wpm)	0.28 seconds
		Typing Random Letters	0.50 seconds
		Typing Complex Codes	0.75 seconds
		Worst Typist (unfamiliar with keyboard)	1.20 seconds
P	Point the mouse to an object on screen		1.10 seconds
B	Button press or release (mouse)		0.10 seconds
H	Hand from keyboard to mouse or vice versa		0.40 seconds
M	Mental preparation		1.20 seconds
T(n)	Type string of characters		n × K seconds
W(t)	User waiting for the system to respond		

KLA

- Rules for placing Ms
  - The most complex component of the method

Begin with a method encoding that includes all physical operators and response operations. Use Rule 0 to place candidate Ms, and then cycle through Rules 1 to 4 for each M to see whether it should be deleted.	
Rule 0	Insert Ms in front of all Ks that are not part of argument strings proper (e.g., text strings or numbers). Place Ms in front of all Ps that select commands (not arguments).
Rule 1	If an operator following an M is fully anticipated in the operator just previous to M, then delete the M (e.g., PMK -> PK).
Rule 2	If a string of MKs belong to a cognitive unit (e.g., the name of a command), then delete all Ms but the first.
Rule 3	If a K is a redundant terminator (e.g., the terminator of a command immediately following the terminator of its argument), then delete the M in front of the K.
Rule 4	If a K terminates a constant string (e.g., a command name), then delete the M in front of the K; but if the K terminates a variable string (e.g., an argument string) then keep the M.

KLM – Example

- Delete file by dragging to trash bin vs. adding delete option in Edit menu
  - Will the new proposed solution save time?
- It is assumed that:
  - a single file is to be deleted;
  - file and trash bin are visible;
  - hand starts and ends at keyboard;
  - user is average non-secretary typist.

KLM – Example

Begin with a method encoding that includes all physical operators and response operations. Use Rule 0 to place candidate Ms, and then cycle through Rules 1 to 4 for each M to see whether it should be deleted.	
Rule 0	Insert Ms in front of all Ks that are not part of argument strings proper (e.g., text strings or numbers). Place Ms in front of all Ps that select commands (not arguments).
Rule 1	If an operator following an M is fully anticipated in the operator just previous to M, then delete the M (e.g., PMK -> PK).
Rule 2	If a string of MKs belong to a cognitive unit (e.g., the name of a command), then delete all Ms but the first.
Rule 3	If a K is a redundant terminator (e.g., the terminator of a command immediately following the terminator of its argument), then delete the M in front of the K.
Rule 4	If a K terminates a constant string (e.g., a command name), then delete the M in front of the K; but if the K terminates a variable string (e.g., an argument string) then keep the M.

- Alternative 1 - Drag file to trash bin
  1. move hand to mouse (**H**)
  2. point to file icon (**P**)
  3. press and hold mouse button (**B**)
  4. drag file icon to trash bin icon (**P**)
  5. release mouse button (**B**)
  6. move hand to keyboard (**H**)

- HPBPBH / HMPBMPBH
- Tempo total: 3,2s /

KLM – Example

Begin with a method encoding that includes all physical operators and response operations. Use Rule 0 to place candidate Ms, and then cycle through Rules 1 to 4 for each M to see whether it should be deleted.	
Rule 0	Insert Ms in front of all Ks that are not part of argument strings proper (e.g., text strings or numbers). Place Ms in front of all Ps that select commands (not arguments).
Rule 1	If an operator following an M is fully anticipated in the operator just previous to M, then delete the M (e.g., PMK -> PK).
Rule 2	If a string of MKs belong to a cognitive unit (e.g., the name of a command), then delete all Ms but the first.
Rule 3	If a K is a redundant terminator (e.g., the terminator of a command immediately following the terminator of its argument), then delete the M in front of the K.
Rule 4	If a K terminates a constant string (e.g., a command name), then delete the M in front of the K; but if the K terminates a variable string (e.g., an argument string) then keep the M.

- Alternative 1 - Drag file to trash bin
  1. move hand to mouse (**H**)
  2. point to file icon (**P**)
  3. press and hold mouse button (**B**)
  4. drag file icon to trash bin icon (**P**)
  5. release mouse button (**B**)
  6. move hand to keyboard

Rule 1
(assuming that dragging is fully anticipated, once the file icon is clicked)

- HPBPBH / HMPBPBH
- Tempo total: 3,2s / 4,4s

Online calculator

KLA – Example

Begin with a method encoding that includes all physical operators and response operations. Use Rule 0 to place candidate Ms, and then cycle through Rules 1 to 4 for each M to see whether it should be deleted.	
Rule 0	Insert Ms in front of all Ks that are not part of argument strings proper (e.g., text strings or numbers). Place Ms in front of all Ps that select commands (not arguments).
Rule 1	If an operator following an M is fully anticipated in the operator just previous to M, then delete the M (e.g., PMK -> PK).
Rule 2	If a string of MKs belong to a cognitive unit (e.g., the name of a command), then delete all Ms but the first.
Rule 3	If a K is a redundant terminator (e.g., the terminator of a command immediately following the terminator of its argument), then delete the M in front of the K.
Rule 4	If a K terminates a constant string (e.g., a command name), then delete the M in front of the K; but if the K terminates a variable string (e.g., an argument string) then keep the M.

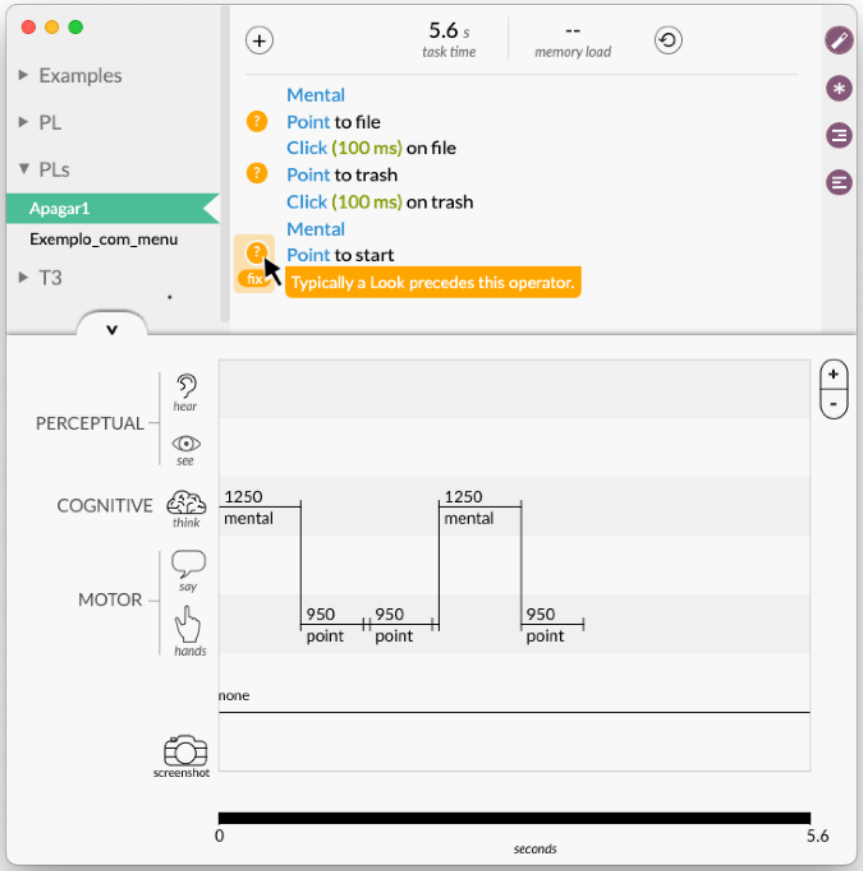
- Alternative 2 - delete file via menu bar
  1. move hand to mouse (**H**)
  2. point to file icon (**P**)
  3. click mouse button (**BB**)
  4. point to file menu (**P**)
  5. press and hold mouse button (**B**)
  6. point to DELETE item (**P**)
  7. release mouse button (**B**)
  8. move hand to keyboard (**H**)

- HPBBPBPBH / HMPBBMPBPBH
- Tempo total: 4,5s / 6,9s

Under the presented assumptions, the new option will take longer

Tools — Cogulator

- A human performance calculator for estimating:
  - **task time**
  - working memory load
  - mental workload
- Based on GOMS
  - supports different variations
- Highly configurable
- Aims at use of use — GUI interface
- Applications in Cognitive Psychology, Artificial Intelligence and Human-Computer Interaction
- Developed by the MITRE Corporation



Mote…

"We must design for the way people behave, not for how we would wish them to behave."

Don Norman, *Living with Complexity*