5 Designing Mobile Interfaces

A mobile interface is different from the one we find in a desktop application, and for this, we have different challenges when designing a mobile one:

- *Tiny screen sizes* which do not offer much space to present information or choices
- *Variable screen widths* which impose us to use it intelligently when designing the app
- *Dynamic physical environments* in which the mobile is used (bright sun, dark cinema, ...)
- *Touch screens* that need links and buttons large enough to hit easily
- *Difficulty of typing text*, hence we have to minimize this action
- *Limited user attention* of which we should take care during design

In order to design usable mobile interfaces we have to rely on already-defined design patterns, invest on the first time UX which tells us a lot, and use prototyping like mockups.
One of the central problems of a User-Centered Design (UCD) process is how to provide designers with the ability to determine the usability consequences of their design decisions. We can follow **Shneiderman's 8 Golden Rules**:

1. *Strive for consistency* in action sequences, terminology, command use...
2. *Enable frequent users to use shortcuts* to perform familiar actions more quickly
3. *Offer informative feedback* for every user action at an appropriate level
4. *Design dialogs to yield closure* i.e. let the user know when they have completed a task
5. *Offer error prevention and simple error handling* i.e. clear instructions to recovery
6. *Permit easy reversal of actions* to relieve anxiety and encourage exploration
7. *Support internal locus of control* (the user is in control of the system)
8. *Reduce short-term memory load* keeping display simple, providing time for learning action sequences...

or **Norman's 7 principles**

1. Use both knowledge in the world and knowledge in the head
2. Simplify the structure of tasks
3. Make things visible
4. Get the mapping right, i.e. make sure that the user can determine the relationships (like larger size of money -¿ larger value)
5. Exploit the power of constraints to make the user perform the right action only
6. Design for error

7. When all else fails, standardize

One way to approach UI design is to learn from examples that have proven to be successful in the past. Design Patterns are solutions to a recurrent problem within a specific app domain. These are the ingredients for usable mobile design:

- **Information Architecture**: the organization and structure of data within an informational space. In other words, how the users will get to information or perform tasks within an app
- **Interface Design**: The design of the visual paradigms from which the user will assess meaning and direction given the information presented to her/him
- **Interaction Design**: The design of how the user can participate with the information present either in a direct or indirect way, meaning how the user will interact with the application to create a more meaningful experience and accomplish her/his goals
- **Information Design**: The visual layout of information presented to the users
- **Mobile Design Patterns**

5.1 Information Architecture

Information Architecture represents the core of the user experience. From a simple mobile website to an iPhone/Android app, the i.a. defines how the information wil be structured. The truly successful mobile products always have a well though and defined info architecture. The first deliverable to define info architecture is the *site map* that represents the relationship of content to other content and provide a map for how the user will travel through the informational space.

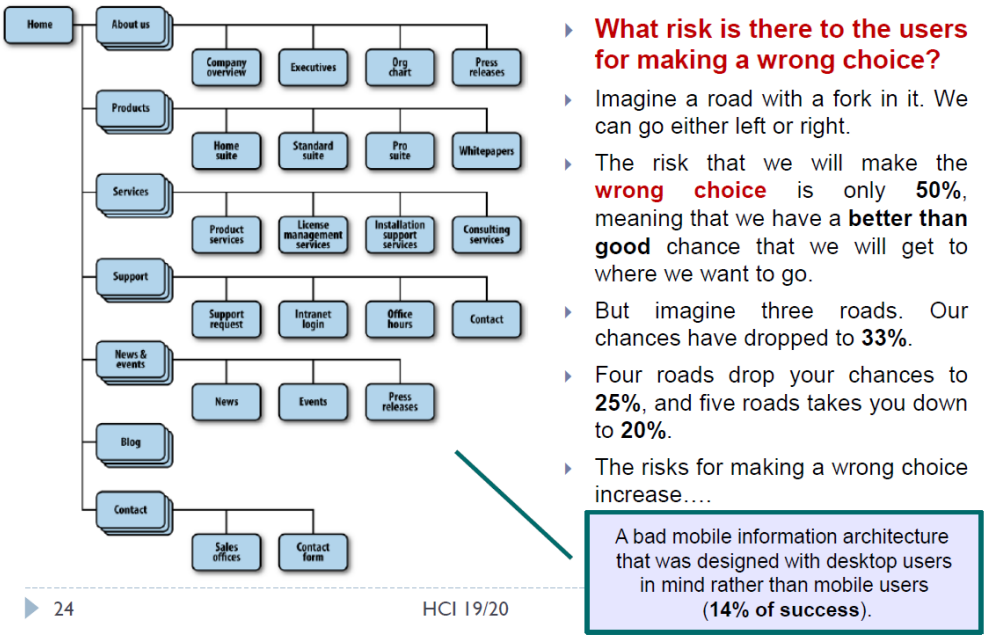

Figure 36: Bad Site Map for Mobile.

- In the mobile context, **tasks are short** and **users have limited time** to perform them.
- **Limit users' options**: A mobile information architecture should provide **5 navigation areas or less**.
  - The risks to make the wrong choice are minor.
- **Suggestion**: Make the path through the information you present **logical** and **easy to predict**.
  - put **markers** to let them where they are.
  - put a **back-button**.
  - *When mobile users select the wrong path, they should immediately click back to where they started and go down another path, eliminating the wrong choices to find the right ones.*

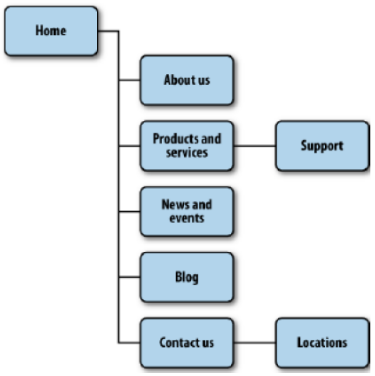

Figure 37: Good Site Map for Mobile.

5.2 Interface Design

Interface design analyzes the visual layout of content presented to a mobile user and how the user assesses meaning and direction from it. The greatest challenge to creating a mobile design that works well on multiple screen sizes is filling the width. a traditional solution is the use of *vertical designs*: the interface design is a cascade of content from top to bottom, similar to a newspaper where the content consumes the majority of the screen.

Figure 38: Vertical Design.

For content-heavy sites and apps, this solution works well. The problem is when it is required to present a large number of tasks or actions. We see some alternatives to accomplish this goal.

- *Axis* is the most basic and common information principle for organizing content and consists of an imaginary line that is used to align a group of elements in an interface. The design feels ordered with them. Axis can be made more apparend if the edges of surrounding elements are well defined (axis reinforcement).

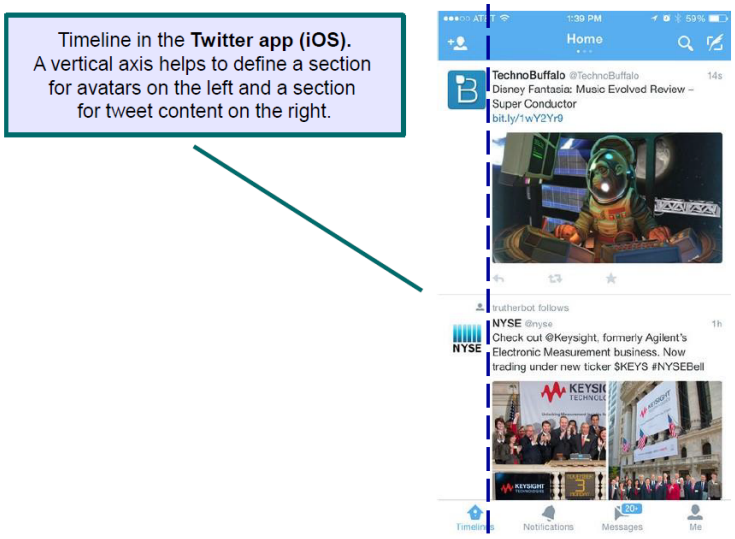

Figure 39: Example of Axis Reinforcement.

When we encounter something linear, such as an axis, we naturally follow the line in a direction; lines in fact encourage *movement* and interactions.

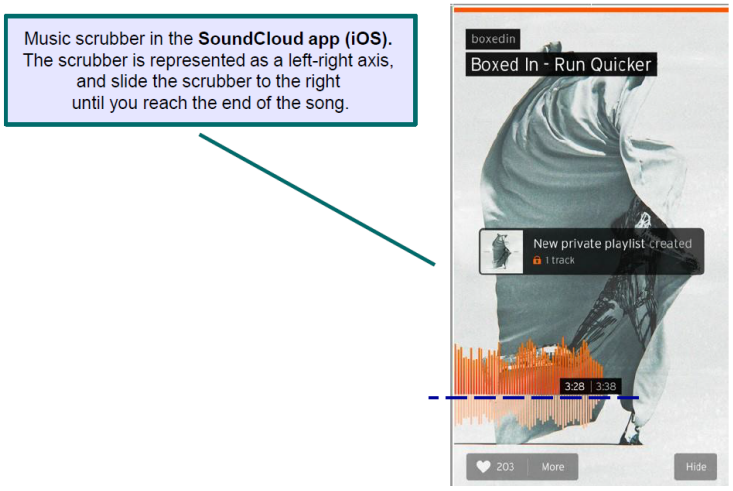

Figure 40: Example of Axis Movement.

We call *Infinite Axis* when an axis endpoint is undefined, i.e. we can slide through the axis indefinitely until we reach something interesting or we get tired of such interaction (social networks).

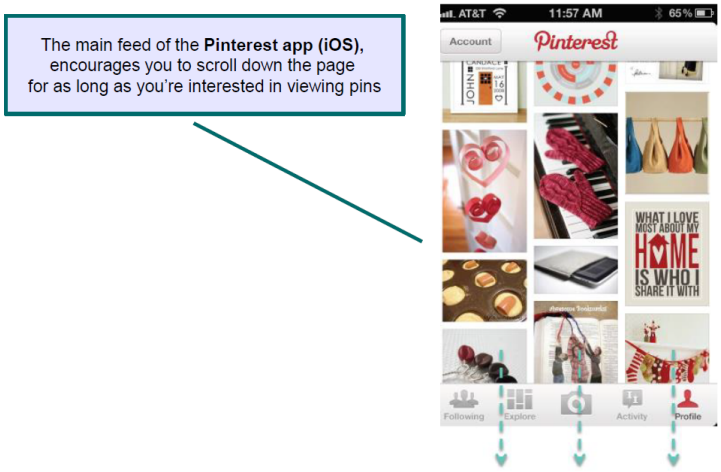

Figure 41: Example of Infinite Axis.

- *Simmetry* helps to make the design feel armonious and easy to read, both top-bottom and left-right. Think about Android's image gallery.

- *Hierarchy* is obtained when an element appears more important in comparison to others in a design. We have different hierarchies:
  - by Size (trivial)
  - by Shape (rounded Instagram profile picture wrt the rest)
  - by Placement (end of an axis is more hierarchical than points through the axis)
- *Rhythm* is the movement created by a repeated pattern of forms, like the one that we can see in Instagram's feed with images and videos. The same pattern repeats over and over, hence the user knows exactly where to look in the rhythm to get a specific info.
- *Break* in a repeated pattern gets more hierarchical and usually divides in sections a rhythm (think about "who to follow" and "who you searched" when we see the list of people in instagram search!).

5.3 Interaction Design

Interaction Design investigates the way people interact with their mobile devices. The interaction is any direct or indirect communication between a user and his/her mobile device. Direct is with user feedback, indirect is with sensors or similar stuff. When we perform direct one, we can do it through single or multi touch, and also through physical buttons like a pad or so.

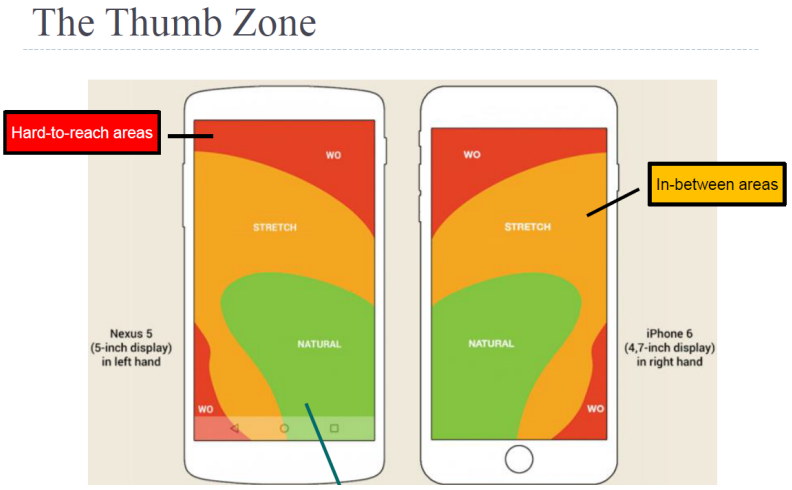

Figure 42: Thumb Zone.

We have to place relevant content within the thumb's reach. Important content needs to be aimed towards the thumb, while secondary content (everything that is not oftenly accessed) can be placed in the stretch. Better not to use hard to reach areas at all.

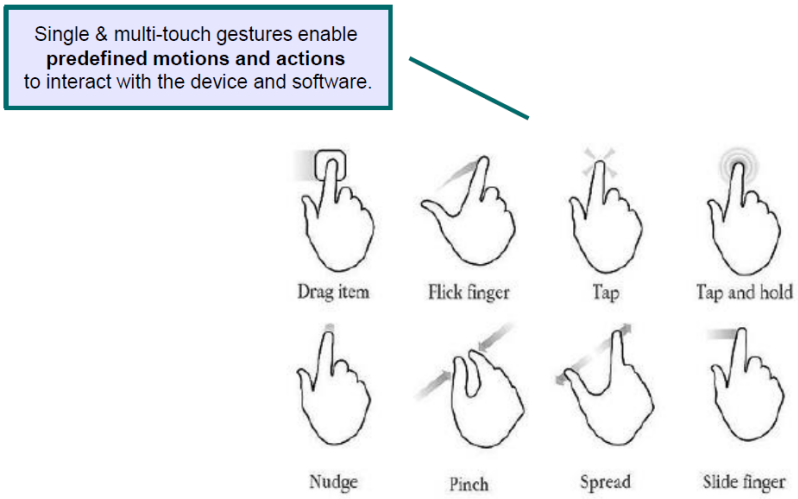

Figure 43: Touch interaction and gestures.

Gestures have to be embraced with attention. Users expect gestures to work the same, regardless of the app they are currently running. We have to make sure that users can reach them!

5.4 Information Design

It concerns the visual layout of info presented to the users. Three main aspects to consider:

- Design for "fat fingers". Make your links and buttons large enough to hit easily
- Design for distracted users.
- Think about colors and typography. And think about motion.

Users react differently to colors, since they evoke different emotions. A predefined color palette has to be chosen when designing an app.

Figure 44: Colors emotions.

Remember also to follow readability guidelines, i.e. readable font and space between letters and lines, organizing the text in short paragraphs.

6 User-Centered Design (UCD)

The primary aim of the process of design and implementation of an interactive system should be to maximize the usability of the system. It is therefore important for us to understand:

- The characteristics, methods and tools of a process of design and implementation that can maximize usability
- The characteristics of usability
- How to measure and/or evaluate the usability of an interactive system (another class on evaluation)

UCD is also referred to as the user-centered methodology or human-centered design or human-centered methodology, and is an approach to design that grounds the process in information about the people who will use the product. UCD intends to ensure that the user is at the center during the design process in order to realize products that meet usability requirements.
Usability is the property that reflects the ease of use of a computer system. It is defined by 5 principles: learnability, efficiency, memorability, errors and satisfaction. It is important in improving the software quality. The use context also improves the same way. Let's see some advantages of usability:

- Allows you to focus on user needs and organization
- Increases productivity
- Improve the quality of products
- Improve quality of life
- Allows compliance e.g. with ISO standards, European directives on displays

We observed that a common definition of usability is: "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use."
Here are some characteristics of UCD:

1. Know your users
2. Actively involve users early and continuously
3. Rapid and frequent iteration of designs with usability assessments
4. Multidisciplinary team

According to the ISO 13407 standard, there are four essential user-centered design activities which should be undertaken to incorporate usability requirements into the software development process:

1. understand and specify the context of use: the quality of use of a system depends very much upon the context in which a system will be used. In some cases contextual information may already be known, although, where a new product or system is to be introduced, then it will be necessary to collect the relevant contextual information. At the end, the following aspects are understood:

- the characteristics of the intended users
- the tasks the users will perform , and allocation of activities between users and system
- constraints and characteristics of the socio-organizational and technological environment in which the users will use the system

the result of this initial activity are embodied in a document which desscribes the context of use for the proposed software.

2. specify the user and organizational requirements: building on the context of use description obtained previously, an explicit statement of the user-centered requirements for the new software should be formulated, with identification of relevant users' range, prevision of a clear statement of design goals...
3. produce design and prototypes: the key goal is to simulate the design solutions using paper or computer based mockups, with which we can explore design solutions and prototypes in general and then later presenting them to a representative sample of users. We have to involve users early in order to explore and refine design choices in light of feedback, and iterate design solutions until the design/usability goals or requirements are met.
4. carry out assessments/evaluations: this is a crucial phase in which we develop an evaluation plan. Three steps:
   - identify anomalies and defects most relevant at this stage
   - select the best solution for the system in light of the requirements at this stage
   - report the results and recommendation for refining the design at this stage; the refine design becomes the design in the next stage

   This evaluation process is iterated at each stage until design/usability goals or requirements are met.

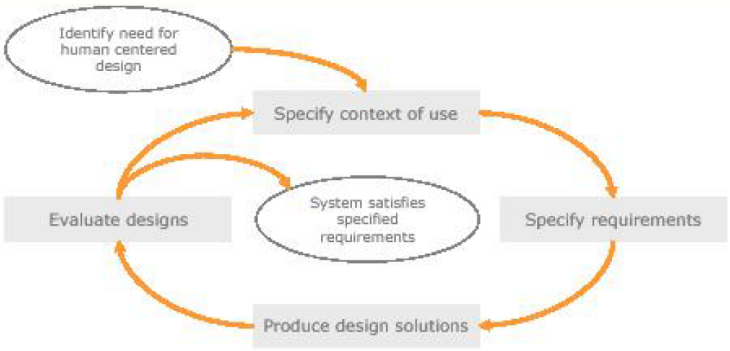

Figure 45: UCD Activities.

There are many methods which can be used to achieve the goals of UCD: requirements analysis, task analysis and evaluation techniques.

7 Interaction Design Basics

Interaction design is not just about the design of interactive systems. Interaction design also includes the design of artifacts (e.g.tools/resources such as office equipment). Interaction design is about the design of interaction itself. However, i.d. is not just abut the artifact that is produced, it's about understanding and choosing how the artifact is going to affect the way people work or live. It is also about documentation, manuals, tutorials etc.

Design is achieving goals within constraints. Goals are the purpose of the design we want to do, while constraints are materials, standards, platforms, cost, time... we have to find the tradeoff. The golden rule of design is: *understand your materials*, that in HCI means understanding computers, humans and their interaction.

The central message or core of interaction design is the user. We have to know their background, computer experience and so on. For doing it, we use Personas, Scenarios, Questionnaires, Interviews.

8 Cognitive Models

Cognitive models are models represented by goal and task hierarchies. They model aspects of user: understanding, knowledge, intentions and processing. Common categorizations are: competence vs performance, computational flavor, no clear divide (we choose one).
Goals - Intentions, Tasks - Actions
For goal hierarchies we can follow different schemes:

- *GOMS*: Goals, Operators, Methods and Selection. Operators are basic actions performed by user, methods are decompositio of a goal into subgoals/operators and selection means of choosing between competing methods.

```
GOAL: CLOSE-WINDOW
.   [select GOAL: USE-MENU-METHOD
            .   MOVE-MOUSE-TO-FILE-MENU
            .   PULL-DOWN-FILE-MENU
            .   CLICK-OVER-CLOSE-OPTION
            GOAL: USE-CTRL-W-METHOD
            .   PRESS-CONTROL-W-KEYS]

For a particular user:

   Rule 1: Select USE-MENU-METHOD unless another
           rule applies
   Rule 2: If the application is GAME,
           select CTRL-W-METHOD
```

Figure 46: GOMS Example.

- *Cognitive Complexity Theory*: CCT comprises two parallel descriptions: user production rules and device generalised transition networks. Production rules are of the form: *if condition then action*. Transition networks are covered under dialogue models.

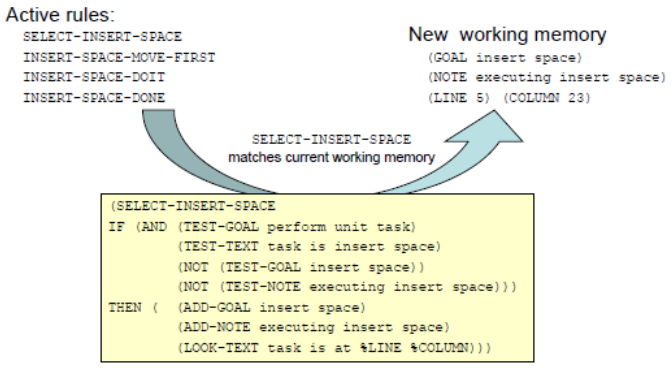

Figure 47: CCT Example.

- *HTA*: they will be explained in detail in next chapters.

Linguistic notations are needed to understand the user's behavior and cognitive difficulty based on analysis of language between user and system. There are two main types:

- *Backus-Naur Form (BNF)*: it is a very common notation from computer science and is a purely syntactic view of the dialogue. There are two components: *terminals* (lowest level of user behavior like mouse actions) and *nonterminals* (ordering of terminals and higher level of abstraction like menu selection). The basic syntax of BNF is nonterminal ::= expression. An expression contains terminals and nonterminals combined in sequence (+) or as alternatives (|).

```
draw line     ::=  select line + choose points + last point
select line   ::=  pos mouse + CLICK MOUSE
choose points ::=  choose one  |  choose one + choose points
choose one    ::=  pos mouse + CLICK MOUSE
last point    ::=  pos mouse + DBL CLICK MOUSE
pos mouse     ::=  NULL  |  MOVE MOUSE+ pos mouse
```

Figure 48: BNF Example.

Measurement in BNF can be done through number of + and | operators. There are some complications, like: same syntax for different semantics, no reflection of user's perception and minimal consistency checking.

- *Task Action Grammar (TAG)*: this method makes consistency more explicit. We have parameterised grammar rules. Nonterminals are modified to include additional semantic features.

In BNF, three UNIX commands would be described as:

```
copy  ::= cp + filename + filename | cp + filenames + directory
move  ::= mv + filename + filename | mv + filenames + directory
link  ::= ln + filename + filename | ln + filenames + directory
```

No BNF measure could distinguish between this and a less consistent grammar in which

```
link ::= ln + filename + filename  |  ln + directory + filenames
```

Figure 49: Another BNF.

consistency of argument order made explicit using a parameter, or semantic feature for file operations

Feature Possible values

Op = copy; move; link

Rules

file-op[Op] ::= command[Op] + filename + filename
| command[Op] + filenames + directory
command[Op = copy] ::= cp
command[Op = move] ::= mv
command[Op = link] ::= ln

Figure 50: TAG.

Among other uses of TAG, we find user's existing knowledge and congruence between features and commands, modelled as derived rules.

Physical and device models are based on empirical knowledge of human motor system. User's task: acquisition then execution. It is complementary with goal hierarchies. Most important type:

- *Keystroke Level Model (KLM)*: it is the lowest level of original GOMS. It has six execution phase operators, and times are empirically determined.

– Physical motor:	K - keystroking P - pointing H - homing D - drawing
– Mental	M - mental preparation
– System	R - response

Texecute = TK + TP + TH + TD + TM + TR

Figure 51: KLM Operators and Time.

```
GOAL: ICONISE-WINDOW
[select
   GOAL: USE-CLOSE-METHOD
   .     MOVE-MOUSE-TO- FILE-MENU
   .     PULL-DOWN-FILE-MENU
   .     CLICK-OVER-CLOSE-OPTION
   GOAL: USE-CTRL-W-METHOD
         PRESS-CONTROL-W-KEY]
```

- compare alternatives:
  - USE-CTRL-W-METHOD vs.
  - USE-CLOSE-METHOD
- assume hand starts on mouse

USE-CTRL-W-METHOD		USE-CLOSE-METHOD	
H[to kbd]	0.40	P[to menu]	1.1
M	1.35	B[LEFT down]	0.1
K[ctrlW key]	0.28	M	1.35
		P[to option]	1.1
		B[LEFT up]	0.1
Total	**2.03 s**	**Total**	**3.75 s**

Figure 52: KLM Example.

Architectural Models: all of these cognitive models make assumptions about the architecture of the human mind, like with LTM/STM, problem spaces, interacting cognitive subsystems...

Display-based interaction: most cognitive models do not deal with user observation and perception. Some techniques have been extended to handle system output (e.g. BNF with sensing terminals, display-TAG,...), but problems persist.

9 Requirements Gathering

A *requirement* is something the product must do or a quality it must have. However, users do not mention some requirements because they assume that they are obvious. Good interviews and observations will help to reveal them. Some of them arise only when models are constructed or prototypes are reviewed. We can divide them into:

- *functional requirements*, what the system must do;
- *non-functional requirements*, qualities the system must have.

Ethnography is an ethnographic technique where the evaluator visits the normal workplace of the users. The evaluator should be as unobtrusive as possible so as to allow the user to work normally. Usually, there are no videos in order to respect this not-intrusion criterion. Attention is paid to how tasks are actually done, as opposed to the way they are thought to be done. The purpose of ethnography is to give designers and evaluators an understanding of the context in which a technology is used.

Interviews should be thematic yet open-ended and discursive to allow the participant to direct the process somewhat. Extracts of interviews can be made to highlight details of particular interest.

- *Structured interviews* have a specific, predetermined agenda;
- *Unstructured interviews* are used during the earlier stages of design and the goal is to gather as much infos as possible concerning the user's experience. It is done with no structure, no agenda.

Focus groups are a technique for collecting data from a range of users. A moderator is required to lead the group, but the session should be as fluid as possible whilst staying on topic. All participants should contribute and care should be taken to cover a broad range of topics and not allow one person to dominate. The collected data may be difficult to organize, but audio recording should help.

Questionnaires need very careful design and piloting and are composed by closed questions and opened ones. Obviously, the first ones are easier to analyze. Sensitive questions have to be asked in a series of separated closed questions so that the sensitive answer is not directly disclosed. Questionnaires require testing before release and statistical verification and analysis, and therefore can be unwieldy.

PACT Analysis is People, Activities, Context, Technologies analysis of a system.

Storyboards are series of scenes/frames from the user experience point of view, usually based on scenarios. They can be used in requirement collection and with mock-ups and they can be time consuming. Storyboards may not accurately reflect actual process to be implemented and should be refined during the design process.