Human Computer Interaction

INSERT A QUOTE

CITY DATE
QUOTE AUTHOR

Dario Loi, Student, Department of Computer Science
Applied Computer Science & Artificial Intelligence
`loi.1940849@studenti.uniroma1.it`

Contents

1 Need Finding **5**
1.1 Observing . . . 5
1.2 Techniques . . . 5
1.2.1 Guidelines . . . 5
1.2.2 Questionnaires . . . 6

2 Storyboarding **7**
2.1 Task Analysis . . . 7
2.1.1 Drawing The Tasks . . . 7

1 Need Finding

In order to develop a good application, we need to understand the needs of our users, during this first lectures we will focus on a variety of methods and procedures with which to achieve this goal.

1.1 Observing

A good way to understand the needs of our users is to observe them, this allows us to identify existing problems, understand their goals and their needs, observation must be conducted:

- Without inductive bias (we don't pretend that we already have the solution)
- With a mind open to unexpected discoveries
- Knowing that it is observation that allows us to define the problem itself

We want to observe not only the user and their actions, but also the environment, the tools at hand, the context, the social interactions, etc.

Remark 1.1.1. *User perception is often wrong! pay attention to what they* say *and what they* do*, they are often not the same.*

Remark 1.1.2. *Do* not *confuse needs and solutions, as said before, do not try to find a solution to a problem, but try to understand the problem itself, otherwise you risk limiting your creativity.*

1.2 Techniques

In order to gather the information we need, we can go further than just observing, some other techniques are:

- Interviews – Ask people questions
- Questionnaires – Ask people to fill out a form
- Room studies – Observe people in their natural environment
- Pager studies – Ask people to carry a pager and record their activities
- Diary – Ask people to keep a diary of their activities

We focus on *interviews* since they are informal, cheap, and allow us to obtain a lot of unexpected information, unfortunately they are also subjective and time consuming.

User Groups When interviewing, we must realize that different people have different approaches to a product, some might be expert users, some might be casual users, some might not even be users at all, hence we must be careful to understand the different groups of users when interviewing.

1.2.1 Guidelines

In general, there are some good practices to follow when interviewing:

- Prepare the questions first
- Record the interview (or take notes)[1]
- Be polite and friendly
- Encourage the interviewee to elaborate on their answers
- Remain neutral towards the interviewee's positions and opinions

Questions to Beware There are some questions that should be avoided, since they are either too general or too specific, here is a list of some of them:

- Questions with an obvious answer – focus on the *why*
- Questions that contain the answer – try to keep the questions open-ended to allow the interviewee to elaborate

[1] Remember to ask for permission before recording the interview

- Questions that are too general – try to be as practical as possible
- Questions that ask the user to replace the designer – ‘Would you like this function?’ rather than ‘How do you think this function should work?’
- Questions based on weird hypotheticals – *Keep it real.*
- Questions that ask how often something happens – Indicate when exactly it happens.

1.2.2 Questionnaires

As previously explained, there are different types of users, if we want to avoid tailoring interviews toward each group, we can use questionnaires, which are a good way to gather information from a large number of people.

Questionnaires are fast, can be analyzed more easily, and can be distributed to a large number of people, however they are also less flexible.

Questionnaire Design When designing a questionnaire, we must keep in mind that we want to obtain *precise* and *reliable* information, we must take great care into avoiding *bias* and *ambiguity*, especially from the questions themselves.

The professor suggests Google Forms as a good tool to create questionnaires.

2 Storyboarding

2.1 Task Analysis

Up until now, there haven't been any notes since the lectures were largely focused on requirement analysis, we now want to understand a few things:

- Which needs we want to satisfy?
- Why should people use our application?
- What should our application allow to do?

Remark 2.1.1. *You are* Not *the user of your application, hence try to obtain external opinions on your user experience.*

Tasks We want to identify the set of *tasks* supported in our application

Definition 2.1.1. *A task is a sequence of actions that a user performs to satisfy its needs, the task is part of a general* activity

Hence, in this phase, we are mainly interested in the role of the *user interface*, not only on a functionality level, but also in relation to its users, for example:

- The actors involved
- The environment
- The required tasks

We are *not* designing the screens yet, we are just trying to understand the role of the user interface in the application.

2.1.1 Drawing The Tasks

It is optimal to provide a pictorial representation of the flow of the tasks, this helps understanding the way the user approaches the application, when doing so, we are under these constraints:

- Efficiency – We want a quick mock-up
- Comprehensibility – We want it to be understandable
- Communicating the tasks to the designers

The produced drawing is called a **storyboard** and it is a sequence of pictures that represent the flow of the tasks, to its core, it is a **comic**.

The perfect storyboard is:

- Hand drawn
- Clear and simple
- With as few text as possible
- With as few panels as possible

What to draw? The storyboard should provide *snapshots* of the interface at particular points during the user interaction, it should be as simple as possible, but personal notes to clarify the drawings are allowed and encouraged when they don't clutter the drawing.

It is important to not draw the screen, since at this stage in development we don't know what the screen should look like, we just want to understand the flow of the tasks, this allows us to be unconstrained by implementation details.

How Many To Draw? One should draw one storyboard for each task, initially, we want to only consider the *main* tasks, doing *everything* is a bad idea, since it reduces the agility of the design process, we want to focus on the *core* of our application.