Human Computer Interactions
Individual requirements

Christophe Pincemaille

October 30, 2008

Abstract

In Human-Computer Interactions (HCI), we have to do a group project to elaborate a user interface. We decided to make it on OS[1] window-managers. The objective of this paper is to make some proposals, in order to build up requirements for the group project.

Introduction

The goal is to customize a part of an OS window manager, in order to it to be more user-friendly and easy to use. The main orientation around which are done these specifications is the idea that the more the environment will look as the reality, the easier it will be for people to use it. The important thing is: it is important that there is a terminal which enable command line, in order for the advanced users to be able to have a huge power of expression and not being limited by the GUI.

1 multiple virtual desktops

In some OS (Linux, Mac OS...), there are multiple virtual desktops. This seems very interesting to improve the usability of the system. In general, advanced users know that there are several desktops, know how to switch between them, how the shortcuts are, how to change the number and parameters of the desktops, how to organize their windows and applications among the desktops... The problem might be for novice to know how all that stuff work, how to easily switch of desktop...

- the problem of knowing the things can be solved by putting, by default, icons on a menu bar of the OS representing the desktops ; that is a required feature;
- to switch between the desktops, there could be a graphical effect which makes each desktop becoming a table of a real desktops which draws in 3D; it is a core function as a lot of things will stick on it;

[1] Operating Systems

- the trash (placed in the menu bar) can grow and go on the right of the 3D-image of the desktop, to appear like a real trash ; that could be considered as an optional but interesting feature.

2 installation and uninstallation of software

Installation and uninstallation of software is often something that will influence a lot the usability and the ease of a system. There are several systems which exists: step-by-step installation as on Windows, command-line from a repository like on Linux, drag-and-drop of the application like on Mac OS X.

We can think of a system, window-manager based, which makes easy and user-friendly the system usability concerning installation and uninstallation of software.

- the concept of the application that can be uninstalled by being drop into the bin is interesting and should be kept, and is important ;
- the Linux concept of the repository in which are a huge quantity of programs, whose links are accessible in local is interesting, but could be optional as it is hard to do ;
- the previous concept should be a little and minimal local window, simply accessible, in which are links to the applications ;
- the application is downloaded by clicking the link, and installed by being automatically dropped into a folder of applications.
- installations and uninstallation should be possible by command line as well.

3 Access to programs

Access to programs is another point that can make a system simple or hard to use.

- the programs should not be accessible, by default, through a huge quantity of menus ;
- programs shortcuts should be displayed on a tray ;
- this tray which should be resizable by cursor-dragging: when putting the cursor on the limit of the tray, a different arrows appear that shows it is resizable ;
- though, if the user wants to set up new bar with a menus and put its applications shortcuts in it, it should be able to do it ;
- a simple click should open the program – a double-click would seem artificial and hard to do to a novice user ;

- the information that would be expected by a simple click should be displayed when putting the cursor on the program's icon ;
- when launching an application, a feed-back should be given to the user. Examples of means to give this feed-back:
  - by the icon of the tray (like in Mac OS) ;
  - by something in the cursor (like in Linux) ;
  - by another mean (to be found).
- when a program is launched, its icon on the tray should have another appearance (icon lighter, darker, change of colors, or even a different icon – not too far from the original one) ;
- it should be possible to define keyboard shortcuts to call applications (cf. keyboard shortcuts on part 5) ;
- programs should also be accessible through a terminal, by typing its name, like in UNIX/Linux environments.

4 widgets

What we name *widgets* are little applications which give little services (like Mac OS dashboard, like widgets Windows Vista or KDE4): give the weather, data about the computer, the hour, the date...

- The widget should appear when to 3D desk appears: the clock on the top, the post-its on a "wall", the whether on a screen near to the desktop...
- The widget should be abled to be placed to another location by click-and-drop.
- Widget should be able to be added or removed easily. For example :
  - widget could be put on the trash, which remove them unless they are put out of the trash ;
  - the way to put widget on the trash could be by drag-and-drop, which is the closest way to the reality ;
  - on the 3D-image of the desktop, there could be a drawer, from which can be extracted absent widgets. Thus, the widget could be installed by drag-and-drop from this drawer to the desk.
  - this drawer can be, by default, linked to a repository of default widget, or can be supplied by the user

5 Other elements

Defined keyboard shortcuts

The user should be able to define keyboard shortcuts to make some events happen, like opening the applications. That should be defined through the administration panel of the considerate thing, for example, the application panel to define keyboard shortcuts that open applications. But all the shortcuts should be stored in a central database, in order to prevent problems with one shortcut used to call more than one event.