Chapter XI

Design of the Man-Machine Interface: Menus, Inputs, and Outputs

This chapter deals with the design of the interfaces between the users and the system. First, it describes a method for the design of menus trees—for the entire system as well as for its subsystems. Then it describes how to design the inputs and outputs/reports of the systems. As a result, new classes of menus, forms, and reports are added to the class diagram.

Designing the Menus Tree Interface

A menus interface enables the user to choose the desired options from lists of available options presented on screen. The lists of options, that is, menus, may appear in various forms, for example, text, icons, buttons, and so forth. The advantage of a menus interface is that even naïve, inexperienced users are able to operate it and find what they are looking for. If the menus are organized in a hierarchy, as a tree of menus, the user may start the search from the root menu which proposes the main options; the user then can successively make selections.
As we know, a menus tree consists of a hierarchy of menus. The root (or main) menu contains the primary options, which reflect the main issues or services which the system provides. The user selects an appropriate option from this menu, and as a result the system displays a secondary (or child) menu which includes the suboptions, that is, more specific issues/services. This process may go on—depending on the “depth” of the menus tree in the direction explored by the user—until at some point the user selects an option which does not lead to another child menu, but rather activates the desired operation. Hence, any menu may contain two types of items: “selection” and “trigger.” A “selection” item causes the display of a submenu; a “trigger” item causes the activation of an operation, that is, a transaction of the application. It is likely that a menu at the top of the menus tree (particularly the root menu) will include mostly “selection” items, while menus at lower levels will include more “trigger” items. Note that a menus tree is not necessarily balanced, that is, the depth from the root menu to the lowest level menu may vary, depending on the breakdown of issues/services provided by the system. This issue will be elaborated later on.

As said, there are many ways and forms of presenting menus on screen. In other words, a menu item may appear to the user in various forms. For example, as a text box consisting of one or more words which describes briefly the option, or an icon which portrays the option, or both. A “user friendly” system would enable the user to get more explanations about the options (“help”) which can be invoked by the user (e.g., by hitting the respective menu item). Menus may be presented in lines or columns. For example, a root menu may consist of text boxes displayed on the top line of the screen; once a “selection” item is selected, the submenu may “pull down” in a column under the selected option; the selection of a “selection” item from this menu may cause a submenu to “pop up” at a certain location on screen, and so forth. Besides varieties of forms of presentation, there are issues of aesthetics of menus. This includes proper use or colors, size of text and icons, location of items on screen, and so forth. All these aspects of menus design will not be discussed in this chapter. We concentrate on the **functional design** of the menus interface, namely the various menu items which

down the tree until the desired option is found and selected. This kind of interface is suitable for a variety of users, especially occasional or inexperienced ones, who are not familiar with the information system (IS) and its capabilities. A menus interface also enables enforcing an authorization policy: Certain users may be given access only to the options which they are allowed to perform. However, a menus tree may be too tedious and time consuming for experienced, routine users, who already know what they are looking for and would like to get direct access to the desired options. For them, a “direct manipulation” interface is needed, for example, special function keys or shortcuts which directly activate the desired operations. In this chapter we concentrate on the design of a menus tree interface.