Software Design & architecture

S41

User Interface Design

User interface design is an essential part of the software design process. User interface design should ensure that interaction between the human and the machine provides for effective operation and control of the machine. For software to achieve its full potential, the user interface should be designed to match the skills, experience, and expectations of its anticipated users.

4.1 General User Interface Design Principles

- *Learnability*. The software should be easy to learn so that the user can rapidly start working with the software.
- *User familiarity*. The interface should use terms and concepts drawn from the experiences of the people who will use the software.
- *Consistency*. The interface should be consistent so that comparable operations are activated in the same way.
- *Minimal surprise*. The behavior of software should not surprise users.
- *Recoverability*. The interface should provide mechanisms allowing users to recover from errors.
- *User guidance*. The interface should give meaningful feedback when errors occur and provide context-related help to users.
- *User diversity*. The interface should provide appropriate interaction mechanisms for diverse types of users and for users with different capabilities (blind, poor eyesight, deaf, colorblind, etc.).

4.2 User Interface Design Issues

- User interface design should solve two key issues:
- How should the user interact with the software?
- How should information from the software be presented to the user?
- User interface design must integrate user interaction and information presentation. User interface design should consider a compromise between the most appropriate styles of interaction and presentation for the software, the background and experience of the software users, and the available devices.

4.3 The Design of User Interaction Modalities

- User interaction involves issuing commands and providing associated data to the software. User interaction styles can be classified into the following primary styles:
- Question-answer. The interaction is essentially restricted to a single question-answer exchange between the user and the software. The user issues a question to the software, and the software returns the answer to the question.
- Direct manipulation. Users interact with objects on the computer screen. Direct manipulation often includes a pointing device (such as a mouse, trackball, or a finger on touch screens) that manipulates an object and invokes actions that specify what is to be done with that object.
- Menu selection. The user selects a command from a menu list of commands.
- Form fill-in. The user fills in the fields of a form. Sometimes fields include menus, in which case the form has action buttons for the user to initiate action.
- Command language. The user issues a command and provides related parameters to direct the software what to do.
- Natural language. The user issues a command in natural language. That is, the natural language is a front end to a command language and is parsed and translated into software commands.

4.4 The Design of Information Presentation

Information presentation may be textual or graphical in nature. A good design keeps the information presentation separate from the information itself. The MVC (Model-View-Controller) approach is an effective way to keep information presentation separating from the information being presented. Software Design Software engineers also consider software response time and feedback in the design of information presentation. Response time is generally measured from the point at which a user executes a certain control action until the software responds with a response. An indication of progress is desirable while the software is preparing the response. Feedback can be provided by restating the user's input while processing is being completed. Abstract visualizations can be used when large amounts of information are to be presented. According to the style of information presentation, designers can also use color to enhance the interface. There are several important guidelines:

- Limit the number of colors used.
- Use color change to show the change of software status.
- Use color-coding to support the user's task.
- Use color-coding in a thoughtful and consistent way.

- Use colors to facilitate access for people with color blindness or color deficiency (e.g., use the change of color saturation and color brightness, try to avoid blue and red combinations).
- Don't depend on color alone to convey important information to users with different capabilities (blindness, poor eyesight, colorblindness, etc.).

4.5 User Interface Design Process

User interface design is an iterative process; interface prototypes are often used to determine the features, organization, and look of the software user interface. This process includes three core activities:

- User analysis. In this phase, the designer analyzes the users' tasks, the working environment, other software, and how users interact with other people.
- Software prototyping. Developing prototype software help users to guide the evolution of the interface.
- Interface evaluation. Designers can observe users' experiences with the evolving interface.

4.6 Localization and Internationalization

User interface design often needs to consider internationalization and localization, which are means of adapting software to the different languages, regional differences, and the technical requirements of a target market. Internationalization is the process of designing a software application so that it can be adapted to various languages and regions without major engineering changes. Localization is the process of adapting internationalized software for a specific region or language by adding locale-specific components and translating the text. Localization and internationalization should consider factors such as symbols, numbers, currency, time, and measurement units.

4.7 Metaphors and Conceptual Models

User interface designers can use metaphors and conceptual models to set up mappings between the software and some reference system known to the users in the real world, which can help the users to more readily learn and use the interface. For example, the operation "delete file" can be made into a metaphor using the icon of a trash can. When designing a user interface, software engineers should be careful to not use more than one metaphor for each concept. Metaphors also present potential problems with respect to internationalization, since not all metaphors are meaningful or are applied in the same way within all cultures.