Part III
Software Design Perspectives

This third part introduces five additional software design topics. Four of these five topics are covered independent of each other, allowing the learner to cover these topics in any order they choose. The design patterns topic is the exception; the design patterns chapters refer to chapters on HCI and secure design. A brief description of the topics and chapters is below.

- Human–computer Interaction Design
  Read Chaps. 17, 18 (OOD) and 19 (SD) to learn about a text-based user interface. Read Chaps. 17, 20, 21 (OOD), and 22 (SD) to learn about a graphical-based user interface.
- Quality Assurance
  Chapter 23 covers important quality assurance topics. To help you create better designs, you should read this chapter!
- Secure Design
  Chapter 24 covers important security design principles, while Chaps. 25 (OOD) and 26 (SD) apply these principles to the case studies.
- Design Patterns
  Chapter 27 covers design patterns, while Chaps. 28 (OOD) and 29 (SD) apply design patterns to the case studies. The OOD design patterns chapter refers to designs found in Chaps. 15, 18, 21, and 25. The SD design patterns chapter refers to designs found in Chaps. 16, 19, 22, and 26.
- Persistent Data Storage Design
  Chapter 30 covers data modeling and Chap. 31 introduces XML and relational databases, two common persistent storage technologies. Chapters 32 (OOD) and 33 (SD) apply these concepts to the case studies.

Introduction to Human–Computer Interaction (HCI) Design

17

The objective of this chapter is to introduce design concepts associated with human–computer interaction (HCI).

17.1 Preconditions

The following should be true prior to starting this chapter.

- You understand the six characteristics of a good software design: simplicity, coupling, cohesion, information hiding, performance, and security.
- You understand the notion of abstraction as a design process. Abstraction is the ability to generalize a concept by removing details that are not needed to properly convey the design perspective being emphasized.
- You have created and/or modified models that described an object-oriented or structured software design.
- You understand how to apply the Model–View–Controller architectural pattern to modify an existing design into these three components.
- You understand how to apply the Model–View–Controller architectural pattern to create a high-level design that accurately reflects the domain requirements while satisfying the design constraints inherent in MVC.

D. P. Voorhees, *Guide to Efficient Software Design*, Texts in Computer Science,
https://doi.org/10.1007/978-3-030-28501-2_17

17.2 Concepts and Context

Designing a user interface is hard, for several reasons. First, evaluating a user interface is subjective. For example, which of the following search web-page designs do you like better? [1]

- www.dogpile.com
- www.duckduckgo.com
- www.botw.org

The *dogpile* search page gives you some choices about the type of content you would like to search for—web, images, video, news, and shopping. The *duckduckgo* search page provides only a text box for entry of a search term—no search choices are provided. In contrast, the *botw* search page shows a directory structure on its search page. Do you like the minimalist design of duckduckgo? Or do you like the search choices presented on the dogpile page? Or do you like the option of searching through a directory structure as provided by *botw*?

Second, the term *user friendly* is vague but yet is often used to describe a good user interface. A person may look at a web page and conclude that *they really like this user interface because it is user friendly*. Another person may look at the same web page and conclude that *this user interface is not user friendly*. This term is used in a subjective manner and supports the popular saying: *beauty is in the eye of the beholder*.

Finally, different HCI technologies make it difficult to compare designs across platforms. Designing a user interface for a smartphone is different from designing a web page which is different from designing a software application that will run on a personal computer. Each of these types of applications uses different technologies, some of which (e.g., touch screen devices) have a significant impact on the HCI design. Even within the same set of technologies, there are differences. For example, designing a web page that is used for data entry (e.g., a web page form) is different from designing a web page that displays content to the user.

17.2.1 HCI Evaluation Criteria

What we need is a way to assess an HCI design using criteria that provides structure and consistency to this process. The following four criteria provide a way to assess HCI designs while also highlighting design aspects that are important for designers to think about as they develop a user interface [1].

- Efficiency—How does the HCI affect the productivity of the user?
- Learnability—How easy is it for a user to learn the HCI?

[1] The comparison of the following three web pages is based on their appearance on August 11, 2019.

- User satisfaction—How satisfied is the user with the HCI?
- Utility—Does the HCI provide useful and timely information?

17.2.1.1 Efficiency

Some questions to ponder that pertain to the efficiency of an HCI design.

1. What type of input device must be used to enter data?
   When entering text-based data, a keyboard is the traditional device but pointing devices and touch screen technology offer other options. When selecting an option from a list, a pointing device is the traditional device but keyboard (via function keys, hot keys, or arrow keys) and touch screen technology offer other options.
2. Within one screen/page, how much information is related to content versus data entry versus navigation?
   In the context of this question, content refers to static information while data entry and navigation require some form of user action/input.
3. What type of input device must be used to navigate?
   When the screen/page includes both text-based data entry and navigation options, hand movements between the keyboard and pointing device will slow down the user. In this scenario, a pointing device (e.g., mouse) tends to be slower than a function key or hot key that initiates the same action.
4. How much information needs to be displayed on one screen/page?
   Scrolling the screen/page because all of the information cannot be viewed (at the same time) on the display device will slow down the user. However, having multiple screens/pages so that scrolling is not necessary requires additional navigation actions from the user.

17.2.1.2 Learnability

Some questions to consider that pertain to the learnability of an HCI design.

1. Does the design use standard user interface objects and metaphors?
   With the creation of many different operating systems come many different user interface designs. The notion that a user interface standard exists and is used by all of the companies that produce an operating system is a pipe dream. Given this, there are some pseudo-standards that HCI designers should use. For example, the appearance and behavior of a text box for entry of text-based data or the appearance and behavior of a drop-down list should conform to the pseudo-standard that exists across all of the operating systems (and their mostly proprietary user interface platforms).
2. Does the user interface look different for different types of users?
   Some users of an application may have more authority and thus can see more information. A simple example of this is a payroll application at a company. A payroll clerk will not have the ability to see as much data as the payroll manager. Do these different views into the information get presented in a similar

way? That is, a payroll clerk gets promoted to payroll manager. Does this person need to relearn the entire system or is it a simple matter that some additional fields/information is now displayed to the individual?
3. Does the user interface require instructions on most screens/pages?
   Excessive amounts of instruction on a screen/page suggest that the HCI design is not easy to learn and use.
4. Is the user interface consistent across all of its screens/pages?
   Viewing and entering data, and navigating around an application, should be consistent across all of its screens/pages. For large organizations that have lots of custom software, every attempt should be made to make the user interface consistent across all of these applications.

17.2.1.3 User Satisfaction

Some questions to contemplate that pertain to the user satisfaction of an HCI design.

1. How do we balance design choices when a group of users with similar needs have different opinions about what they like and don't like in an HCI?
   A compromise between different opinions may produce the best design, or it may be worse than an HCI design that conforms to a majority (or minority) opinion.
2. How do we make design choices when the users are unknown?
   We may be designing version one of a software product. No users exist, so a user-centered design approach may not work.

17.2.1.4 Utility

Some questions to think about when considering the utility of an HCI design.

1. Does all of the information displayed pertain to the purpose of the screen/page?
   Information unrelated to the purpose of the screen/page may create too much clutter; the user cannot quickly find the pertinent information.
2. Do the series of screens/pages for an application have a logical flow to their use? Does the series of screens/pages correspond to the process steps being automated?
   Any mismatch between the HCI design and the underlying process being automated will make the application less useful.

17.2.1.5 A Simple Example

Let's say that a software application needs to (1) obtain a person's name and (2) display a list of people's names. Let's also assume that these two HCI needs would be designed into two separate screens/pages.

For the data entry of a person's name:

- Should they enter their full name into one text box? If yes, does the software expect this data to be entered in a specific format (e.g., *first middle last* versus *last, first middle*)?
- Should they enter their first name, middle name, and last name in separate text boxes? If yes, how should we sequence these three text boxes on the screen/page (e.g., *first middle last* versus *last first middle*)? Should the three text boxes be sequenced left-to-right or top-to-bottom within the screen/page?

For the display of a list of people's names:

- What format should be used to display the list of names (e.g., *first middle last* versus *last, first middle*)?
- Should the user have the option of sorting the list by first name, last name, or middle name?

For both the data entry and display:

- Does this application need to work in other countries/cultures? Instead of using *first middle last* as components of a name the application may need to use other terminology. Examples include *family-name given-name*, *surname given-name*, and the reverse of either of these examples.

17.2.2 Software and HCI Design Goals

When designing a software solution, the goal is to help a person or organization solve a problem. How well the software achieves this goal is largely dependent on the HCI design. Why? Because a user understands the capabilities and limitations of the software through the lens of its user interface. That is, the users' perception is that *the user interface is the software*.

Given the importance of the HCI design, what should someone tasked with developing a user interface think about as they consider HCI alternatives? An HCI designer should know the user, prevent user errors, optimize user abilities, and be consistent [2]. Each of these is discussed in more detail below.

17.2.2.1 Know the User

Knowing the user is challenging since this implies an understanding of the user's:

- True needs,
- Common activities, and
- Level of expertise in the problem domain and in information processing.

Understanding a user's true needs is hard since there may be different types of users where each type of user has slightly different needs. Effectively communicating user

needs may be challenging given the international nature of many software products, the differences in native languages, and the terminology being used may be specific to a business or industry. When the HCI designer has a different native language from the user, or does not understand the terminology being used by the business or industry, miscommunication of user needs will inevitably occur.

Understanding the user's common activities may be difficult when there are users that need to use the software for different purposes. How does the HCI design enable these different uses without adding unnecessary layers of complexity?

Understanding the user's level of expertise, whether this is related to the problem domain or in general information processing, may result in the need to design different types of HCIs to address these different levels of expertise.

17.2.2.2 Prevent User Errors

Whenever possible, a user interface should be designed to minimize possible user mistakes. For example, there may be a situation where there are only a few valid options the user may choose from. In this case, presenting these choices in a way that prevents the user from making an erroneous selection would be a good design choice.

For a text-based user interface, there may be a situation where there are only a few valid options. In this case, presenting a menu of choices and allowing the user to enter a letter or digit related to the menu choice being selected should minimize the possibility of the user making a mistake. Similarly for a graphical-based user interface, a small group of radio buttons or a drop-down list with fixed choices should minimize the possibility of user error.

When a user does make a mistake, the way in which the error message is displayed and how the error message is worded are additional areas where a design choice is important. An error message should guide the user to the source of the error without requiring lots of explanation.

17.2.2.3 Optimize User Abilities

Not all users of a software application are alike. Some may be more familiar with software systems than others. A power user is someone that has significant experience and knowledge using software systems. Designing a user interface that meets the needs of power users as well as individuals that have no experience using software systems or knowledge of the application domain is very challenging. For this reason, an HCI design should provide multiple ways to accomplish a task whenever this is reasonable. For example, shortcuts may be provided for power users while a wizard that provides a step-by-step walk through of a process could be used by less experienced users. Providing a help facility within a software application should be done so a user may obtain help based on their current context without making this facility too intrusive.

17.2.2.4 Be Consistent

Users get used to things appearing and working in certain ways. The use of metaphors (e.g., a trash can for deleting data) should be used in a consistent and common-sense manner. Colors should be used consistently to indicate similar types of information or processing. For example, reporting errors to a user should be done using a consistent color scheme. When multiple screens must be used within a software system, placing content that appears on each of these screens in the same relative location gives the user a quick way to find this content regardless of which screen they are using. Error messages should appear in the same relative location when multiple screens are being used. Having error messages displayed via a pop-up window in some cases and displayed within a screen in other cases may cause confusion.

17.2.3 HCI Design Steps

An overview of the design steps in developing a user interface design is now described [2].

- Gather user needs related to HCI.
- Develop and test HCI prototypes.
- Select HCI prototypes for further design and implementation.

The first two steps may be done with users in an interactive fashion. When combining the first two steps, these sessions should focus on the key interactions between the user and the software. When a large number of screens/pages are needed in the HCI design, particular attention should be spent discussing ways to navigate between these screens/pages. Depending on the types of prototyping tools you use, the selected prototypes may be translated quickly into a more detailed HCI design and then into an implementation. Another option is to simply use pencil and paper (or markers and whiteboard) to create various design layouts.

17.3 Post-conditions

The following should have been learned when completing this chapter.

- Evaluating an HCI design should include thinking about the design using at least four criteria:

1. Efficiency—How does the HCI affect the productivity of the user?
2. Learnability—How easy is it for a user to learn the HCI?
3. User satisfaction—How satisfied is the user with the HCI?
4. Utility—Does the HCI provide useful and timely information?

- An HCI designer should know the user, prevent user errors, optimize user abilities, and be consistent.
- User participation in the HCI design process is critical to developing a user interface that satisfies the four criteria listed above.

Exercises

Discussion Questions

1. Some people will prefer to use a website that offers many different types of content on a single page, while others may prefer using a website that contains information on a single topic. Can the four HCI evaluation criteria help explain these different preferences? If so, how?
2. Have you used a software application that reports errors in different ways? Has this inconsistency led to confusion about how you use the application? If so, explain.
3. Two HCI designs exist—one displays information while the other allows a user to enter lots of data. What differences might there be in the use of the four HCI evaluation criteria in designing these two types of HCI's?

Hands-on Exercises

1. Use an existing code solution that you've developed, evaluate its HCI design using the four evaluation criteria. How good or bad is your design?
2. Continue Hands-on Exercise 3 from Chaps. 12 or 13 by developing an HCI design and then evaluating your design using the HCI criteria described in this chapter. You may sketch possible HCI designs using paper and pencil or use an HCI prototyping tool if one is available. The list below serves as a reminder of the application domain you may have chosen for this exercise. Refer to the Hands-on Exercises in Chaps. 12 or 13 for details on each domain.

    - Airline reservation and seat assignment,
    - Automated teller machine (ATM),
    - Bus transportation system,
    - Course-class enrollment,
    - Digital library,

- Inventory and distribution control,
- Online retail shopping cart,
- Personal calendar, and
- Travel itinerary.

References

1. The Joint Task Force on Computing Curricula (2008) Computer science curriculum 2008: an interim report of CS 2001. ACM and IEEE Computer Society, New York
2. The Joint Task Force on Computing Curricula (2013) Computer science curricula 2013: curriculum guidelines for undergraduate degree programs in computer science. ACM and IEEE Computer Society, New York