Module 2: Interfaces

Interfaces are the point of connection between people and their tools. Common examples are door handles, light switches, remote controls, steering wheels, keyboards or bank machine displays. Computer applications such as web browsers, document formatting systems or image processing applications provide highly complex graphical interfaces. Design of these interfaces greatly influences their utility, security, and ease or enjoyment of use. In this module you will learn how good design of human-computer interfaces is informed by the fields of computer science and psychology and will use this knowledge to enhance your skills and confidence in navigating both graphical and text based interfaces.

Learning Goals. After this module you should be able to

2.1 explain how tools augment and constrain our power to think and act, define the “myth of human error” and give examples that dispel this myth;

2.2 explain strengths and weaknesses of human-computer interfaces (both graphical and non-graphical) referring to concepts such as familiarity and consistency, mappings and metaphors, feedback, negative transfer, or additional concepts that you identify

2.3 navigate unfamiliar interfaces through effective use of conceptual models, play and systematic exploration, and available resources; and

2.4 use basic features of text based interfaces such as Unix or search engines, with knowledge of the ways that special symbols are interpreted (or misinterpreted) by such interfaces.
2.1 Tools and the myth of human error

If you give people tools, [and they use] their natural ability and their curiosity, they will develop things in ways that will surprise you very much beyond what you might have expected.
— Bill Gates (Microsoft Founder)

“The fewer the tools, the greater the imagination.”
— Ben Okri (Nigerian Author)

“The bad workman blames his tools”
— American Proverb

These quotes present contrasting perspectives on our relationship with tools. Since interfaces are key factors in determining the value and limitations of tools—particularly digital tools—it is useful to start a study of interfaces by considering the ways that tools both augment and constrain our ability to think and act.

2.1.1 Tools augment our ability to think and act

It is hard to argue with the assertion that tools augment our ability to act. That’s really the whole point of tools. The idea that tools augment our thinking is perhaps less obvious. Yet information storage and processing devices can be a boon in jogging our memories, in helping us with mental tasks that don’t come naturally to humans such as mathematical calculations and in opening doors to new information and ideas.

Tools can augment our thinking in more dramatic ways. Advances in science are often the consequence of better tools in which to see our world. For example van Leeuwenhoek, an 18th-century Dutch businessman who advanced the art of lens-making was the first to discover bacteria. Advances in microscopes later made it possible to understand the role of bacteria in disease. Similarly, today’s digital tools for sequencing and analyzing genomes are revolutionizing our understanding of genetic diseases. As one example that builds on the previous module, the ability for biologists to compare and explore phylogenetic trees with thousands of species, using tools such as Tamara Munzner’s TreeJuxtaposer, enables them to think in new ways about the tree of life.

Tools, including digital tools, extend the ways that artists can create powerful works that change the ways we think about ourselves and our world. To quote computer artist Vera
Molnar, “the machine, which is thought to be cold and inhuman, can help to realize what is most subjective, unattainable, and profound in a human being.”

Tools are often used in ways that were never anticipated by the tool designer, as Bill Gates’ quote suggests. The inventors of the World Wide Web aimed to ease communication of scientific ideas; their tools are now fundamentally changing political expression across the world and the role of governments in controlling access to information. (In earlier eras, the invention of communications tools such as the printing press and the telegraph had perhaps even more profound impact on societies.) Intel Corporation, a major manufacturer of computer chips and computer technologies, now employs anthropologists who study the often unexpected ways that people across the world integrate computer and communications technologies in their lives, reflecting cultural priorities and social practices as well as available resources.

2.1.2 Tools constrain our ability to think and act

Tools constrain our ability to act (and think!) when they intrude in unexpected ways in our lives. For example cell phones are great when we want to talk with a friend, but a pain when a call interrupts something else we are doing.

Educational resources aim to help us learn and think. How does the shift to digital technologies support or impede these aims? As noted earlier by a student in this class, digital notes currently constrain our ability to add individualized mark-ups to course notes, compared with paper-based notes. Digital tools still often hinder face-to-face collaboration, in part because of the way they are currently designed: while several people with pens can use a whiteboard to share ideas, computer applications typically support input only from one user at a time. At the same time, digital materials are easier to edit and distribute, and on-line forums provide ways to share and collaborate across long distances.

Number systems are valuable conceptual tools that we use constantly. The Roman numeral system was a valuable tool for keeping track of time and much more in Roman society. But try multiplying numbers represented in Roman numerals and you’ll gain a better appreciation of the Hindu-Arabic number system we use today!

Ben Okri’s quote at the start of this section speaks to an ultimate constraint of tools: limiting our imaginations. Do you agree?
2.1.3 The myth of human error

Tools constrain us especially when they lead to undesirable or unintended consequences. When faulty use of technology leads to problems, the person using the tool is often blamed, as the American proverb quoted at the start of the section suggests. The myth of human error is that people’s mistakes or poor judgement cause errors, rather than poor design of the technology or the use of technology in a challenging environment.

In fact, many so-called human errors are actually errors in design.

Redesign of a tool’s interface can significantly reduce the rate at which problems arise. This is particularly true in situations that are time-critical, safety-critical, or involve multi-tasking. For example, a review of changing practices in medicine noted that “computerized physician order entry—in which physicians enter prescriptions online, where they can be checked for problems—was shown in a randomized trial to decrease the frequency of serious medication errors by 55%.” Human factors analysis has significantly informed the engineering of safety-critical systems for the greater part of a century. It is well established that accidents result from poor design of digital interfaces which provide information to system operators. For example, the spatial layout and organization of information is correlated with the speed at which an operator can infer critical information and thereby take action to prevent an accident. Good ergonomic design of tools is also important in reducing injuries.

Effective design of complex tools and their interfaces, then, is informed by an understanding both of human physiology and psychology as well as engineering principles. For example, cognitive scientists are learning about the factors that influence whether or how people attend to and notice changes in their visual range. UBC’s Ron Rensink, who holds a joint appointment in Psychology and Computer Science, has researched the cognitive factors that explain why people “fail to see” changes in their visual view. His work is influencing the design of digital interfaces and signals in cars with the goal of increasing safety and reducing certain types of accidents.

Exercises:

2.1.1 Door handles and doorknobs are interfaces that we use every day. Find a few and consider how they augment or constrain your actions. Which can you use when both

---

of your hands are full? Which help you know which way a door opens? Which can be used in other ways, such as as a hook or hanger? Which provide security mechanisms? In what other ways might doorknobs be useful? See http://www.brighthandle.com/ for some creative ideas.

2.2 Sizing up user interfaces

Even in applications which are not safety-critical, interfaces should work well for real people working under real constraints. A simple example: unlike earlier systems, computer systems today routinely provide ways for people to undo actions and to think twice before deleting files and folders. It is also possible to retrieve deleted data from the Trash Can. Work on files is periodically autosaved so that some work can be retrieved after an unexpected computer failure.

Don Norman’s essay “Why is Everything So Difficult to Use?” (Chapter 8, The Invisible Computer) provides historical perspective on the tendency for tools to be complex and difficult to use. Computer applications such as document formatting or image processing applications are inevitably complex because they provide a broad range of features that traditionally were the exclusive domain of trained professionals. These tools aim both to be usable in basic ways by novices, while also being of use to experts. As designers of digital tools gain experience in their craft, principles for design and evaluation of the quality of interfaces are emerging. Following are just a few principles that are widely cited. There is not always universal agreement about these principles and you can likely add others to the list.

- **Aim for familiarity and consistency.** When icons and symbols are used in consistent ways across many applications, users can transfer experience in the use of one tool to another. Examples of common usage in graphical user interfaces include:
  - The icon which indicates the position of the mouse on the screen. A pointed finger suggests that the data object underneath the icon can be clicked, for example.
  - Color and symbols used in text menus, such as the following:
    - black followed by a filled △ symbol; holding the mouse over such a menu item causes a new menu to open,
    - black followed by ellipses (…); clicking on such a menu item opens up a new box or menu with further options,
    - black but not either of the above cases; clicking on such a menu item causes an action, or
grey; clicking on such a menu item causes no action given the current mode of the application.

– Use of universal “play” and “pause” symbols in a digital movie or audio players.

• **Use well-chosen mappings and metaphors.** Controls on a tool should “map” in an intuitive way to functions and be organized in a spatially meaningful way. Examples of metaphors in digital applications which derive from physical artifacts of our everyday lives include files and folders for data organization, the trashcan for deleting files, erasers and paint brushes in paint applications, and so-called “radio buttons” in online surveys. Metaphors need to be used with care: users may transfer aspects of their understanding of the physical artifact to the digital artifact, causing confusion. For example, many new users assume that when a file is put in the trashcan, it can no longer be retrieved. When associations from one context are transferred to another but no longer apply in the new context, the result is **negative transfer.**

• **Provide useful feedback.** When tasks take time to complete, signals indicating that the task is indeed progressing are helpful. Examples include bars that indicate the percentage of the task completed, or an “egg-timer” icon to indicate that a task is in progress. Audio feedback can also be valuable; for example, when a virtual button or key is selected a clicking sound provides confirmation that the signal has been received. Other examples are signals that a web page is actively being loaded, or that the battery is low.

• **Manage complexity.** Hierarchically organized menus help avoid a cluttered display while enabling a user to find needed tools in an intuitive fashion. Tools typically provide other mechanisms to help the user control which parts of the system they can view on the screen at a given time. For example, on Google Maps it is possible to close the part of the screen used to provide directions if it is not of interest. A help page provides a way to find features and tools that are not readily apparent (however, these pages need to be indexed with words that users are likely to use in their search).

**Exercises:**

2.2.1 Before doing this exercise, read Don Norman’s essay “Why is Everything So Difficult to Use?”, Chapter 8 in his book “The Invisible Computer”.

• Don Norman’s thinking about factors that make tools difficult or easy to use has changed over the years, reflecting his own experiences and those of the digital design community. Describe in 1-2 sentences an example where Norman’s thinking has changed.
• Suppose you are a technology reviewer for your favourite magazine. Based on Norman’s article, list three criteria you might use when evaluating new digital technologies in your reviews.

• Using one or two of the criteria you chose in the previous part of the exercise, describe something you like about the user interface provided by Google Maps (http://maps.google.com/) and something that you think could be better.

• Do you think Don Norman overstates the case that digital technologies are difficult to use? Briefly explain your answer.

2.3 From concepts to practice: navigating unfamiliar graphical interfaces

In the class labs you will likely use graphical interfaces that are unfamiliar to you, such as that provided by Scratch and GIMP. Consistency across interfaces and the use of effective metaphors provide useful starting points when learning the features of a new application. Following are some other suggestions that may also be useful.

• Take the time to experiment with features and options of an application that are not familiar to you. When feasible, try to predict what’s going to happen when you take a particular action.

• Be aware of the ways that data types and data properties influence the behaviour of an application. For example, if file permissions are not set properly, files cannot be viewed on the World Wide Web.

Be aware also of the ways that the mode, or current state of the application, may influence the behaviour of an application. As an example, it is common in many application to select one or more data items. Typically, actions performed on data will affect only those data items that are selected and if no data items are selected some actions are not possible. For example, when a file is selected in a folder, it can be copied; the copy action does nothing (and is presented in grey in a menu) when no files are selected.

• Recognize when interfaces are poorly designed and keep the myth of human error in mind. When there is no explanation for unexpected behaviour of interfaces or you cannot find a mechanism to perform your desired action, blaming yourself is probably not a useful response! If you can, take the time to report your problem to the tool developers or offer suggestions on what would make the tool better. And don’t hesitate
to consult others who might provide workarounds or solutions to the problems you are having.

- When all else fails, a fallback option that works in surprisingly many scenarios is to quit and restart the application or the computer.

### 2.4 Text based interfaces

Long before today’s era of the graphical user interface and the mouse, text based interfaces were the primary means for interacting with computers. We still use text based interfaces for many purposes, with keyboards as the primary input device. For example, you’ll use some Unix commands in your labs (see Section 2.4.1 below) and use text when using a search engine such as Google. You also use text-based passwords and ID’s when logging in to protected websites.

Some text based interfaces (TBI’s) may seem primitive or unintuitive at first and there may a learning curve to absorb words and symbols with special meaning. But TBI’s also offer some nice advantages to users. Text can be quick to type and the hassle of constantly moving one’s hand from keyboard to mouse is avoided. People tend to be good at storing commonly-used textual commands in “motor memory”. A keyboard is can be easier to use for people who have visual impairments or who have difficulty in controlling a mouse.

There are two principles to keep in mind when using TBI’s. First, you’ll often need to use special symbols (sometimes called meta characters) and language to convey meaning. When things don’t work as expected, the problem is often rooted in the way special symbols play a role. When analyzing what’s wrong, ask whether special symbols might be causing the problem.

Second, TBI’s often offer elegant ways to combine or generalize basic commands. Knowing how this is possible (i.e., having a conceptual model) can help to greatly expand the range of things you can do with just a few simple commands. In this section we use two examples of TBI’s to illustrate these two principles. You will find them useful in other situations too.

#### 2.4.1 Command Line Interfaces in Unix

Conceptually, properties and organization of files on Unix and Linux operating systems are just as in windows-based systems that you are familiar with. Folders are typically referred to as directories, however. So, directories can contain both files and other directories, and directories are organized hierarchically. You can expect to be able to perform much the
same actions on files and directories using the Unix interface as you do using a windows-based GUI. Keeping this conceptual model in mind should be helpful if you explore the capabilities of Unix on your own.

In the lab you will use the Unix operating system to set your file properties so that your web pages are publically available. Three activities, and associated commands, that will be useful to you are as follows⁴.

- **Navigating directories.** Recall that folders are called directories in Unix speak. Upon logging in to a Unix system, you will have access to files in your home ("root") directory. If you have a directory, say “classes”, in your root directory, you can move to that directory by typing `cd classes`.

- **Checking file and directory properties.** To see what files and directories are in your “current” directory, type `ls`. You can add options to this command. For example, if you type `ls -l` you are adding the “l” option which specifies that the files be listed in long format. You can then see additional information about the files and directories, such as how permissions are set. Permission settings determine who else can view your file, who owns (and can therefore modify) the file or directory, and the size of the file or directory (in bytes).

- **Changing file and directory permissions.** In the HTML lab, when you need to make your web page publically accessible you will need to change access permissions on the .html files and the directory in which they reside. For example, to make a file called index.html in your current directory publically readable, type `chmod a+r index.html`. The “a+r” signifies that all users can read the file.

- **Copying a file.** If you have a file called index.html in your current directory you can make a copy of the file, say called old-index.html, by typing `cp index.html old-index.html`

An important point about command line interfaces is that spaces between “words” (such as commands or file names) serve an important function: to separate different parts of the command. Without the spaces, the system would not be able to interpret your command and perform the action you want. Let’s take the Unix file copy command in the last bulleted

---

⁴For a brief introduction to Unix, see [unix:http://www.math.utah.edu/lab/unix/unix-tutorial.html] and [http://hpc.ilri.cgiar.org/documents/unixtut/index.html]. When logged into a Unix or Linux system you can also use the “man” command to find out more about a command. For example “man cp” will tell you a lot about the copy command. The `man` command stands for manual.
item above as an example. Suppose that instead of typing the command as above, you typed instead:

```
cp index .html old-index.html
```

That is, you’ve inserted a space between the name and extension of the first file. The command will no longer work to copy `index.html` to `old-index.html` because `index .html` (with the space) is not considered to be a single file name.

The special status of the space becomes a problem when a file name has a space in it. Suppose you have a file called `WMST201 Notes` in your directory and you want to make a copy called `Old WMST201 Notes`. You can’t just type

```
cp WMST201 Notes Old WMST201 Notes
```

because the spaces are interpreted as separating command and file names. Instead, you can type:

```
cp WMST201 Notes Old\ WMST201 Notes
```

The special `\`’s ensure that the spaces are no longer treated as separators but rather are part of the file name.

Exercises:

2.4.1 The `cp` command can also be used to copy a file into a directory. See if you can figure out how to do that. Then, experiment and use available resources to learn how the command

```
cp index .html old-index.html
```

would be actually interpreted in Unix. In your own words explain how the addition of a space changes the meaning of the command.

2.4.2 In Unix, sophisticated commands can be composed from simpler ones in elegant ways. For example, Unix has a “more” command that is used to scroll through text. The `more` command can be combined with the “`ls -l`” command, making it easier to see the list of properties of files in a large directory. Specifically, the list output by the first command can be “piped” into the second command as follows:

```
ls -l | more
```

The result provides users with the ability to use the scrolling function provided by the `more` command when viewing properties of files in a directory. This is particularly useful when there are many files in the directory and the list does not fit on one
screen. The symbol “|” is called the pipe symbol in Unix. Learn more about the pipe command and effective ways to use it.

2.4.2 Text Based Search and Boolean Operators

Search engines rely on text based interfaces to facilitate searches for images, movies and music (e.g., via their titles) as well searches of text. When searching a large data base, Boolean operators help to narrow or broaden the search in useful ways. Let’s consider Google search on the World Wide Web (http://www.google.ca) as an example.\footnote{For more information on advanced search options using Google, see http://www.google.ca/intl/en/help/features.html}

1. In a typical Google search we enter more than word. Each word used in a search tends to limit the scope of the search. This is because Google’s search engine seeks data sources that match all of the words, or as many as possible. For example, compare the difference when you search using the single word “UBC” versus when you search with the two words “UBC Okanagan”. In the latter case you are getting results that match both “UBC” AND “Okanagan”. (You don’t need to type the “AND”, this is implicit in Google search.)

2. Sometimes, however, we’d like our choice of words to broaden rather than limit a search. For example, suppose we are interested in results that match either UBC Vancouver or UBC Okanagan, or both. Google supports this; just type UBC Okanagan OR UBC Vancouver. In this case, the inclusion of the word OR broadens (rather than limits) the search. Compare the results when you type “UBC Vancouver UBC Okanagan” versus “UBC Vancouver OR UBC Okanagan in the search box.

3. Alternatively, we may want to limit the search by excluding results that contain certain words. For example, want to find articles about UBC Vancouver, you might want to restrict the search to articles that have the words “UBC Vancouver” but NOT the word “Okanagan” Google search supports this too; type “UBC Vancouver -Okanagan”. Similarly, typing “Saturn -car” results in hits for the planet Saturn at the top of the list, while “Saturn” alone results in many car dealership hits.

In summary, Google supports simple searches that can be described in terms of the Boolean operators AND, OR, and NOT. The AND operator is implicitly represented when multiple words are used in a search and the NOT operator is represented by the ”−” symbol. Advanced search options on many web sites support such searches. The power catalogue search provided by the Vancouver Public Library\footnote{see http://ipac3.vpl.ca/ipac20/ipac.jsp?profile=pac&menu=search&submenu=power} provides AND, OR, and NOT operators.