





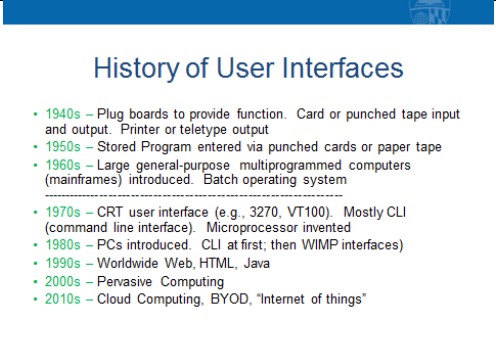
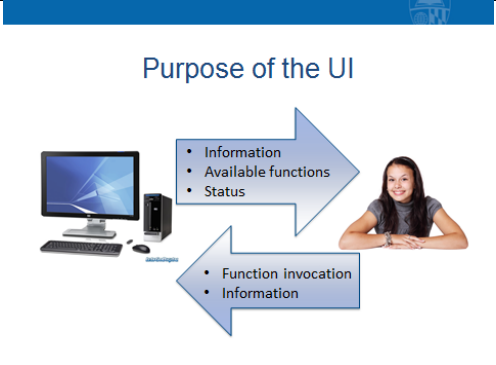
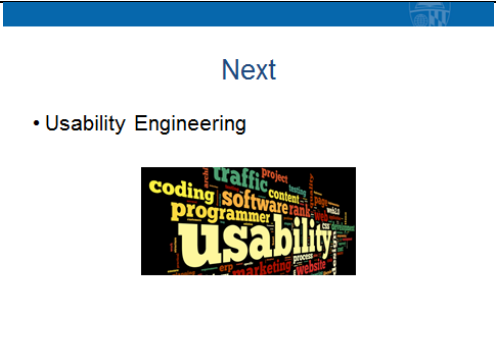







1-1	 <h2>Human-Computer Interaction</h2> 	
2	 <h2>Overview</h2> 	<p>These days, some of the most important non-functional requirements have to do with usability. How easy is it for people to use, learn to use, recall how to use the system?</p> <p>Human-Computer Interaction (HCI) deals with these issues, including modes of operation, usage styles, and human factors.</p> <p>Usability engineering works in parallel with software engineering.</p>
3	 <h2>Objectives</h2> <p>Here is what you should be able to do upon completion of this module:</p> <ul style="list-style-type: none"> <li>• Discuss the history and purpose of user interfaces</li> <li>• Identify Usability and User Experience goals</li> <li>• Perform Usability Analysis</li> <li>• Perform Usability Design</li> <li>• Discuss advantages and disadvantages of four types of user interaction mechanisms</li> <li>• Discuss techniques for handling user errors</li> <li>• Develop a UI prototype and document screen flows and layouts</li> <li>• Apply HCI guidelines and heuristics</li> </ul>	<p>Here are the things you should be able to do after you have completed this module. You should be able to:</p> <ul style="list-style-type: none"> <li>• Discuss the history and purpose of user interfaces</li> <li>• Identify Usability and User Experience goals</li> <li>• Perform Usability Analysis</li> <li>• Perform Usability Design</li> <li>• Discuss advantages and disadvantages of four types of user interaction mechanisms</li> <li>• Discuss techniques for handling user errors</li> <li>• Develop a UI prototype and document screen flows and layouts</li> <li>• Apply HCI guidelines and heuristics</li> </ul>
4	 <h2>Outline</h2> <ul style="list-style-type: none"> <li>• Human-Computer Interaction</li> <li>• Usability engineering</li> <li>• Usability analysis</li> <li>• Usability design</li> <li>• User interface mechanisms</li> <li>• Error handling</li> <li>• Prototyping</li> <li>• HCI guidelines, principles and heuristics</li> </ul>	<p>We start by discussing the importance of the user interface.</p> <p>We will then discuss the field of usability engineering, concentrating on usability requirements and goals.</p> <p>We will see how to perform usability analysis and design.</p> <p>We will then explore the various mechanisms that</p>

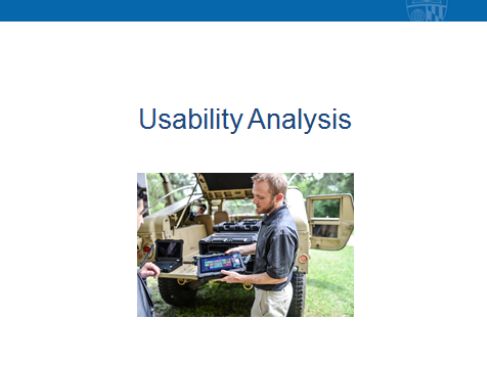

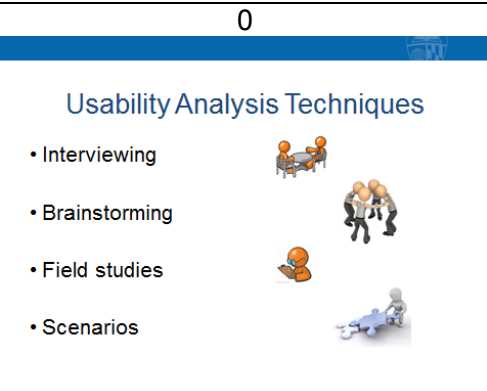

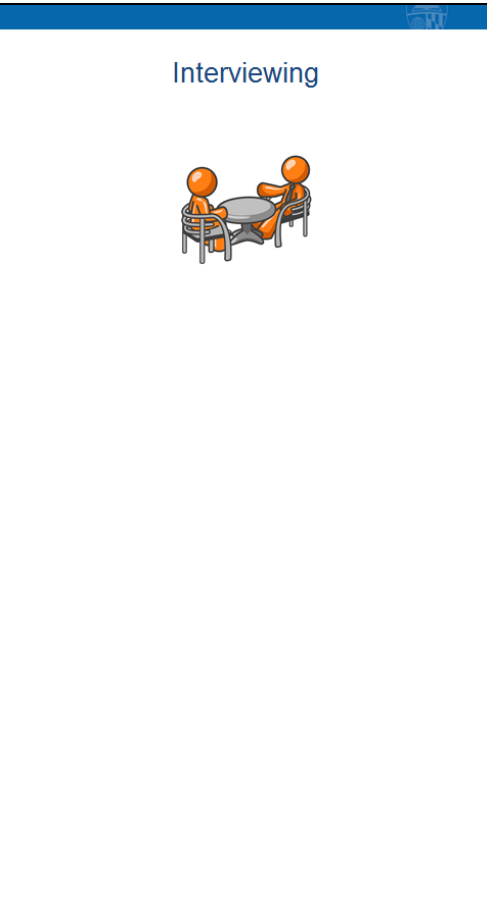

		<p>are commonly used to build user interfaces today.</p> <p>We will consider some guidelines for handling errors.</p> <p>We will talk about ways to produce prototypes of the user interaction.</p> <p>We will conclude with some advice from the experts: HCI guidelines, principles and heuristics.</p>
5	 <p><b>History of User Interfaces</b></p> <ul style="list-style-type: none"> <li>• 1940s – Plug boards to provide function. Card or punched tape input and output. Printer or teletype output</li> <li>• 1950s – Stored Program entered via punched cards or paper tape</li> <li>• 1960s – Large general-purpose multiprogrammed computers (mainframes) introduced. Batch operating system</li> <li>• 1970s – CRT user interface (e.g., 3270, VT100). Mostly CLI (command line interface). Microprocessor invented</li> <li>• 1980s – PCs introduced. CLI at first; then WIMP interfaces)</li> <li>• 1990s – Worldwide Web, HTML, Java</li> <li>• 2000s – Pervasive Computing</li> <li>• 2010s – Cloud Computing, BYOD, "Internet of things"</li> </ul>	<p>The way we interact with computers has evolved from a model where the hardware pretty much dictated the mode of interaction to today, where the user is the most important.</p>
6	 <p><b>Purpose of the UI</b></p> <p>The diagram shows a computer on the left and a woman on the right. A large blue arrow points from the computer to the woman, containing the text: Information, Available functions, Status. A smaller blue arrow points from the woman back to the computer, containing the text: Function invocation, Information.</p>	<p>There are five functions that the user interface provides:</p> <ol style="list-style-type: none"> <li>1. User may invoke functions provided by the system.</li> <li>2. User may input data to the system.</li> <li>3. System may indicate available functions for the user.</li> <li>4. System may display data to the user.</li> <li>5. System may display status information to the user.</li> </ol>
7	 <p><b>Next</b></p> <ul style="list-style-type: none"> <li>• Usability Engineering</li> </ul> 	<p>In the next video, we will discuss Usability Engineering</p>


2-1	 <p>Usability Engineering</p>	<p>Usability engineering is a true engineering discipline, sharing the same characteristics as other engineering disciplines we have discussed. It is aimed at producing practical results using a repeatable process based on scientific principles.</p> <p>Much of the underpinnings of usability engineering is the science of psychology. It has to do with the human mind and physiology.</p>
2	 <p>Usability Engineering</p> <ul style="list-style-type: none"> <li>• Usability Requirements Analysis</li> <li>• Usability Design</li> <li>• Usability Testing</li> </ul>	<p>UI Engineering works in parallel with other engineering disciplines: systems engineering, hardware engineering and software engineering.</p> <p>Just as in other engineering disciplines, there is a repeatable process that UI engineers follow.</p> <p>It starts with usability requirements analysis. The main question to be answered here is, “What are the users’ needs?” We investigate this using many of the same techniques that we discussed in our module on software requirements elicitation.</p> <p>The next phase is usability analysis and design. This is analogous to software analysis and design. Usability analysts start with the usability requirements and determine what user interface features need to be built. Usability design is the process of figuring out how those features will be built in the software and hardware.</p> <p>Then there is usability testing. The main question here is, “How successful is the mapping from the implementation view to the user’s view?” In other words, does the way the user interface really work match the way the users think it should work? A lot of the usability testing is based on psychological testing techniques. One kind of usability testing is called alpha testing. Most people know what beta testing is. The name begs the question: is there such a thing as alpha testing? Yes there is. In beta testing the system is tested in the user’s actual environment with real users doing real work. Alpha testing is done in the developer’s location under controlled conditions. Real users are recruited for this testing. They are given tasks to perform and then are observed as they attempt to execute the tasks. If the testers observe that these typical</p>

		users have difficulty using the product, they can modify the user interface and quickly test the changes to see if the test subjects improve their performance.
3	<div data-bbox="368 375 654 413" data-label="Section-Header"> <h3>Usability Requirements</h3> </div> <div data-bbox="350 443 638 623" data-label="Diagram"> <pre> graph LR     Usability((Usability)) --- Productivity     Usability --- Cost     Usability --- Compatibility     Usability --- Reliability     Usability --- Security     Usability --- EasyToLearn[Easy to learn]     Usability --- EfficientToUse[Efficient to use]     Usability --- EasyToRemember[Easy to remember]     Usability --- FewErrors[Few errors]     Usability --- SubjectivelyPleasing[Subjectively pleasing]   </pre> </div>	<p>So, just as for software requirement or hardware requirements, usability requirements have to be testable. They may be specified in their own document, or they may be included in one of the other requirement specification documents. It depends on whether the user interface involves hardware mechanisms or if it's just a software interface such as a graphical user interface. In any event, usability requirements are stated as non-functional requirements. This means that they don't describe what the system does, but they merely describe ways of getting the system to do what it does.</p> <p>Some typical examples of these non-functional usability requirements are these:</p> <ul style="list-style-type: none"> <li>• Effectiveness – can the users be effective when they perform their tasks? Does the interface make the system useful for them?</li> <li>• Efficiency – can the users perform their tasks without unnecessary work? Does it make their tasks easy to perform? In other words, will the users find it easier to use the system with this user interface than with other possible user interface choices?</li> <li>• Safety – does the user interface make it impossible to do some sort of damage: to equipment, to people, to data, to money, to communication, and so forth?</li> <li>• Utility – is the system useful? Or does it get in the way of the users doing their work?</li> <li>• Learnability – how much training is required in order to become effective at using the system? Does classroom training become part of the requirements? Of course, the less training required the better. Consider most apps that are being developed these days. They should be designed so that it is almost obvious how to use them and that no training should be required.</li> <li>• Memorability – this applies to software that is used infrequently. A good example</li> </ul>


		<p>is income tax software. For people who do their own taxes, they use it for a couple of weeks once a year. They don't want to have to spend much time re-learning how to use it each year.</p> <p>Preece, Rogers &amp; Sharp, <i>Interaction Design</i>, John Wiley and Sons, 2002.</p>
4	<p>User Experience Goals</p> 	<p>There are also user experience goals. These are harder to measure, so they may not be stated as non-functional requirements. They are just goals that we would like to aim for.</p> <p>Some examples are these: Examples:</p> <ul style="list-style-type: none"> <li>• Satisfying – does the user get a feeling of satisfaction from using the system?</li> <li>• Enjoyable – is it, in fact, a feeling of enjoyment?</li> <li>• Fun – does the user have fun using the system? This isn't necessarily a goal for all systems, such as business systems, but we don't want them to feel like drudgery, either.</li> <li>• Entertaining – can using the system be a source of entertainment for the user?</li> <li>• Helpful – does the system help the user with the steps involved in performing tasks? Ways in which a system can be helpful are providing a help menu, tool tips, and other on-screen guidance.</li> <li>• Motivating – does the user feel motivated to use the system? This is more than just using the system because they have to or it makes their job easier. It's because they like to use the system, and feel motivated to do so.</li> <li>• Aesthetically pleasing – this is one of those psychological aspects. It has to do with screen layout, use of color, sound, motion, etc.</li> <li>• Supportive of creativity – we will discuss this later in the module. The user should feel as though they are in charge of the system rather than being just a responder to prompts from the system.</li> <li>• Rewarding -- one form of motivation is that the user actually receives rewards from using the system.</li> <li>• Emotionally fulfilling – this is a deeper psychological goal than the others. It is</li> </ul>




		<p>hard to quantify or measure, but I guess you know it when you experience it.</p> <p>Preece, J. Rogers, Y. &amp; Sharp, H. (2007) <i>Interaction Design: Beyond Human-Computer Interaction</i>. 2nd Edition. New York, NY: John Wiley &amp; Sons.</p>
5	 <p>Next</p> <ul style="list-style-type: none"> <li>• Usability analysis and design</li> </ul>	<p>In the next video, we will look at usability analysis and design.</p>


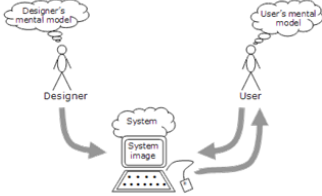
3-1	 <p>Usability Analysis</p> 	<p>In this video we will discuss the usability analysis process.</p> <p>It starts with analyzing usability requirements to determine the best way for users to interact with the system.</p> <p>Then these interactions are designed in terms of software and hardware.</p> <p>Later on, the user interface is implemented as part of the code development process.</p>
2	 <p>Usability Analysis Techniques</p> <ul style="list-style-type: none"> <li>• Interviewing</li> <li>• Brainstorming</li> <li>• Field studies</li> <li>• Scenarios</li> </ul> 	<p>There are a number of possible techniques for analyzing the usability requirements for a system and specifying the appropriate modes of user interaction. We will discuss four such techniques here.</p>
3	 <p>Interviewing</p> 	<p>The first one is interviewing. The goal of an interview is to gain an understanding of the problem from the point of view of the interviewee. It's important to interview the right stakeholders as well as to choose the right interviewer.</p> <p>The stakeholders that can provide the most beneficial information are probably the eventual users of the system. There may be many different types of users, and they may have different perspectives. We should try to interview enough of them so we get a complete picture of how the system is expected to be used. Other people that could be interviewed are the sponsors of the system. They can tell you what their expectations are. If the system is replacing an existing system, it is helpful to interview people who use that system to find out what improvements they would like to see. The danger here is that their perception of the user interaction is biased by the current system, and it may be hard for them to think outside the box. This is especially true if the existing system is a manual system that will be replaced by a computerized system.</p> <p>It is also important to choose the right interviewer.</p>

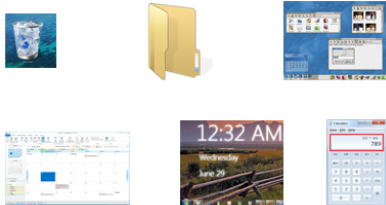
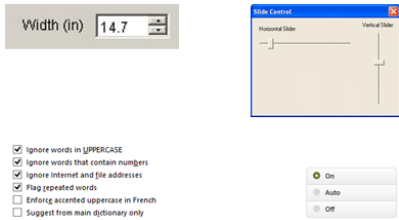

		<p>This person should have enough knowledge about the domain that they can understand the answers to the questions and ask good follow-up questions, but not enough knowledge that they might unconsciously bias the results of the interview. We are trying to find out what the person being interviewed thinks, and not the interviewer. You can see this if you watch people doing interviews on TV. A really inexperienced interviewer will have a list of questions to ask and won't deviate from the list regardless of what the person says. The interviewer doesn't really add any value. The good interviewers do more listening than talking. They might ask an introductory question, but the rest of their questions are based on what the interviewee says. Believe it or not, one of the best interviewers on TV has been Larry King of CNN. He is famous for his style of interview. For example, if he is interviewing an author of a book, he won't have read the book in advance. You'd think that a good interviewer would have at least read a summary of the book or the staff would have briefed the interviewer on the book. But not Larry. The effect is that he knows how to ask the right questions to get the person to talk, listens to the answers, and asks intelligent follow-up questions. He is putting himself in the place of the viewer, who also probably hasn't read the book, either. So he is asking the kind of questions the viewer might ask.</p> <p>You don't want to rely on just one or two interviews. You need to listen to enough people so that you get a complete picture, and individual bias does not sway the conclusions too much.</p>
4	<div>Brainstorming</div> 	<p>Another approach is brainstorming. This technique is useful when no one has much of an idea of how things should work. Brainstorming is a group dynamic approach to coming up with ideas that would not likely have presented themselves if the people were working by themselves.</p> <p>Brainstorming has two phases. During phase 1, we pose a problem to be solved. Then each of the members of the group will suggest solutions to the problem. The ideas are recorded so that everyone can see them. There is no discussion or evaluation of the ideas during this phase. The goal is to collect as many ideas as possible. The</p>









		<p>reason that we record the ideas so that everyone can see them is that something that is suggested by one person might provide the spark for an idea by someone else. Without that spark, that person might never have thought of the idea.</p> <p>After some amount of time the process will seem to slow down. At that point, we go to phase 2. During phase 2, the ideas are organized into categories. Similar ideas are combined. Really zany ideas might be discarded. There might be some kind of evaluation process or even voting to identify the best ideas.</p> <p>Studies have shown over and over again that groups doing brainstorming almost always out-perform individuals working separately at coming up with the best answers to the problem.</p>
5	<div>Field Studies</div> 	<p>Another approach to requirements analysis is performing field studies. This technique is useful when we are replacing an existing system, particularly if we are automating a manual process. We send out a team of analysts to study how things are done currently. The analysts will document the business processes. Many times, the people working in the business are not even aware of the business process until they are shown the business process model by the analysts. In fact, the people have been working so long in the business that the processes have become very inefficient, without them really knowing it.</p> <p>Once we have a model of the business process, we can identify opportunities for automating parts of the process. This will lead to an understanding of the requirements for user interactions to the systems doing the automating.</p> <p>We have to be careful if we are replacing one automated system with another. You don't want to make really radical changes to the interactions, but you do want to take advantage of newer ways of interacting with systems. For example, we might want to replace a command line user interface with a point and click interface. Have you ever noticed how many systems still use monochrome text based interaction technology that was in use back in the 1970's? I'm thinking of a lot of government systems. Or the airline</p>








		<p>reservation system. (What is it that the agents are typing into the system whenever I want to make a change to my ticket?) But much of the time the users have gotten so used to using the old outdated system that they have a hard time (at first) getting used to the new way of operating. You can expect initial dissatisfaction with the new interaction style.</p>
6	 <p>Scenarios</p> 	<p>The last technique is to generate operational scenarios. These are models of how the users will interact with the system. These scenarios are like prototypes of the system. They provide us with ways to study various options or ideas. The prototypes may be at varying levels of fidelity.</p> <p>We will discuss prototyping in more detail later in this module.</p>
7	 <p>Next</p> <ul style="list-style-type: none"> <li>• Usability Design</li> </ul>	<p>In the next video, we will talk about usability design.</p>

4-1	 <h2 style="text-align: center;">Usability Design</h2>	<p>The topic of this video is Usability design. This is the process of figuring out how to bridge the gap between how the user wants to use the system and the capabilities provided by the system implementers.</p>
2	<h2 style="text-align: center;">Mapping Between Models</h2> 	<p>The user's <i>mental model</i> is based on the user's goals for using the system. It may be based on experience with similar systems, perhaps even a prior version of the system under development. It may be based on similar systems that the users may be familiar with. The mental model may be just based on the way the user accomplishes things in real life. The system interface should imitate the real world.</p> <p>The <i>designer's model</i> is described in terms of mechanisms offered by the system implementation. Of course the system won't be able to provide any functionality that is not implemented in the code. The interface must provide access to those functions and information that are provided by the software. The job of usability design is to build a bridge between the capabilities provided by the software to the needs of the users. This bridge may be built in either direction, depending on what came first. If the software hasn't been built yet, the user needs are used as inputs to the software requirements. The bridge is built from the user's model to the designer's model. On the other hand, if the software already exists and our job is to provide an interface to make it easy for the users to access the capabilities of the software then the bridge is built the other direction.</p> <p>Of course the best approach would be to create both models as well as the bridge at the same time.</p> <p>Sometimes this mapping between the user's model and the designer's model is accomplished through the use of metaphors</p>

3	<h3>Metaphors</h3> <p>"All the world's a stage, and all the men and women merely players..."</p> <ul style="list-style-type: none"> <li>Shakespeare, <i>As You Like It</i>, Act II, Scene VII</li> </ul>	<p>Here is an example of a metaphor. One type of metaphor is called a simile. A simile is an analogy or comparison. It usually contains the word, <i>like</i>. For example, we might say that the interface to an audio player app is <i>like</i> a CD player. It has buttons that look just like those on a CD player, and they function in an analogous way.</p>
4	<h3>Desktop Metaphors</h3> 	<p>Here are some familiar desktop metaphors. The computer screen is like a desktop with familiar objects such as calendars, clocks, file folders, and so forth. Their purpose is obvious and they don't take a lot of training to be able to use them.</p>
5	<h3>UI Metaphors</h3> 	<p>Graphical user interfaces, or GUIs, have adopted some metaphors as well. Here are some examples: the little up-down arrows are called spin buttons. There are slider controls, check boxes and radio buttons. These are ubiquitous and their behavior is obvious.</p>
6	<h3>Skeuomorphs</h3> 	<p>Sometimes user interface designers go out of their way to emphasize the metaphor by artistically designing the interaction mechanisms to look realistic. This is referred to as skeuomorphism.</p>

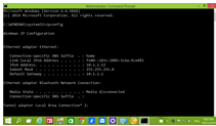
<p>7</p>	<p>Cognitive Distance</p> 	<p>Cognitive Distance -- The decisions made by the user are guided by the user's <i>conceptual model</i>. How does the user interpret:</p> <ul style="list-style-type: none"> <li>▪ What operations are available?</li> <li>▪ What the system state is?</li> <li>▪ Whether the last operation was successful?</li> <li>▪ What information is being displayed?</li> <li>▪ How to navigate to another part of the dialog?</li> <li>▪ What to do next?</li> </ul> <p>The <i>designer's model</i> is conveyed and supported by the user interface as implemented in the code.</p> <p>Hopefully there is congruence between the user's mental model and the HCI designer's model. The degree of congruence between the two models is called the cognitive distance.</p> <p>The closer the mapping between the user's mental model and the designer's model, the easier it is for the user to identify and achieve the system goals. That's why it is most desirable to develop the user's model and the designer's model, as well as the mapping between the two, at the same time.</p>
<p>8</p>	<p>Usability Design</p> <ul style="list-style-type: none"> <li>• Activity design </li> <li>• Information design </li> <li>• Interaction design </li> </ul>	<p>The usability design process involves describing three aspects of the interaction. We can use scenarios or prototypes to explore these aspects of the design.</p> <p>Activity scenarios – show how the users carry out their goals. We can use UML use cases or activity diagrams for these models.</p> <p>Information scenarios – show how information is exchanged between the users and the system. Screen layout prototypes are useful here.</p> <p>Interaction scenarios – show the details of how the user and the system interact. Once again, a prototype, either low fidelity or high fidelity, can be used here.</p>

9	 <p>UI Components</p> <ul style="list-style-type: none"> <li>• Input controls</li> <li>• Navigational controls</li> <li>• Informational components</li> </ul>	<p>There are several aspects of the user interface that can be prototyped.</p> <p><b>Input Controls:</b> These are ways to allow the user to present information to the system. These controls include buttons, text fields, checkboxes, radio buttons, dropdown lists, list boxes, toggles, date fields, file selectors, and so forth.</p> <p><b>Navigational Components</b> provide ways for the user to control the action or the presentation of information. These controls include: breadcrumbs, sliders, search fields, pagination controls, tags, icons, and so forth.</p> <p><b>Informational Components</b> are mechanisms that enable the system to present information to the user. They include: tooltips, icons, progress bar, notifications, alerts, message boxes, modal windows, and so on.</p>
10	 <p>Next</p> <ul style="list-style-type: none"> <li>• Interface mechanisms</li> </ul>	<p>In the next video we will compare some of these basic user interface mechanisms</p> <p>Some of the material for this video was derived from Mary Beth Rosson and John Carroll (2002) <i>Usability Engineering</i></p>


5-1	 <p>User Interface Mechanisms</p>	<p>In this video we will start by exploring input and output mechanisms. Then we will discuss various interaction styles.</p>
2	 <ul style="list-style-type: none"> <li>• Keyboard </li> <li>• Pointing device (stylus, mouse, trackball) </li> <li>• Microphone </li> <li>• Video camera </li> <li>• Special devices </li> </ul>	<p>Input mechanisms give the user the capability of sending messages to the system to either control it or send information to it.</p> <p>One of the earliest and most common input mechanism is the keyboard. It resembles the old typewriter, down to the layout of the keys. The same skills that were needed for typing were now useful for keyboarding on a computer.</p> <p>Later on, as graphical user interfaces began to appear, some sort of device was needed to point to locations on the computer screen. Various pointing devices have been used but the most common devices in use today are the mouse and the stylus. The advantage of a stylus or finger on a touch screen is that there is practically no hand-eye coordination learning curve unlike with a mouse.</p> <p>Microphone input allows the user to use voice commands to control things or provide input. Voice commands are especially popular with small devices such as mobile phones, where using a keyboard is inconvenient.</p> <p>Video cameras can be used as input devices to detect motion and gestures. Devices such as the Microsoft Kinect give us this capability for more than just videogames.</p> <p>And of course there are dozens of special devices that are used for input. Video game manufactures have popularized a lot of these devices. Also, there are devices for people with special needs to help make computers more accessible.</p>


3	<div data-bbox="267 195 760 226" data-label="Image"></div> <div data-bbox="394 243 638 275" data-label="Section-Header"> <h3>Output Mechanisms</h3> </div> <ul data-bbox="300 302 508 451" style="list-style-type: none"> <li>• Text or graphics</li> <li>• Auditory</li> <li>• Video and animation</li> <li>• Haptic</li> <li>• Special devices</li> </ul>	<p>Output mechanisms provide ways for the system to provide information to the user. They also allow system status to be communicated to the user.</p> <p>Of course, text or graphics on a computer screen is the most popular form of output mechanism.</p> <p>We have had auditory output for quite a while. But this has mostly been in the form of beeps and other sounds, mostly to indicate status. Think “You’ve got mail.” But increasingly we are seeing auditory output used for conveying information. Simulated voice can replace text on the screen. A good example is the turn by turn direction in a GPS navigation system. Again, this is particularly useful when the user is unable to see a screen, as when driving, or for people with special needs.</p> <p>Video and animation can be used to convey information or status. A really simple example of this is the animation that you see when the system is busy. The mouse pointer might become an animated gif, or the system might display a moving progress indicator.</p> <p>Haptic feedback uses the sense of touch. A lot of mobile devices will provide a tactile buzz when the user touches icons or keys on the virtual keyboard. This feedback replaces the tactile feedback you would normally get when using a real keyboard or a mouse on a computer.</p> <p>Again, there are many special devices that have been invented for situations in which the more common devices such as screens or even audio output is not appropriate.</p>
4	<div data-bbox="267 1434 760 1465" data-label="Image"></div> <div data-bbox="410 1486 621 1518" data-label="Section-Header"> <h3>Interaction Styles</h3> </div> <ul data-bbox="300 1545 500 1724" style="list-style-type: none"> <li>• Command line</li> <li>• Menus</li> <li>• Fill-in forms</li> <li>• Direct manipulation</li> <li>• Auditory commands</li> <li>• Gestures</li> </ul>	<p>Now, let’s talk about various styles of interacting with systems.</p> <ul data-bbox="836 1507 1141 1713" style="list-style-type: none"> <li>• Command line</li> <li>• Menus</li> <li>• Form fill-in</li> <li>• Direct manipulation</li> <li>• Auditory commands</li> <li>• Gestures</li> </ul>



<div data-bbox="315 203 769 231">5</div> <div data-bbox="315 231 769 1491"> <h2 data-bbox="315 231 769 273">Command Line Interfaces (CLI)</h2>  </div>	<div data-bbox="769 203 1304 1491"> <p>The earliest form of interaction style was the command line interface, or CLI. This is still in use today for some system level tasks.</p> <p>The user must type commands on the command line.</p> <ul style="list-style-type: none"> <li>User must <i>recall</i> the command.</li> <li>User must <i>recall</i> the syntax.</li> <li>User must <i>recall</i> the options.</li> </ul> <p>Commands refer to the system objects and actions.</p> <p>Cognitive distance depends on how well the commands correspond to the user goals.</p> <p>The advantages of a command line interface are</p> <ul style="list-style-type: none"> <li>Flexible and supports user initiative.</li> <li>Appeals to power users.</li> <li>Easy for user to create macros.</li> </ul> <p>The downsides of command line interfaces include</p> <ul style="list-style-type: none"> <li>Poor error handling</li> <li>Substantial memorization and recall are required</li> </ul> <p>Some guidelines to follow when designing command line interfaces:</p> <ul style="list-style-type: none"> <li>Command words should be consistent with user goals.</li> <li>Consistent command and argument syntax</li> <li>Symmetric commands. <ul style="list-style-type: none"> <li>Symmetric: TAKE/RELEASE, FORWARD/BACKWARD.</li> <li>Non-symmetric: GRAB/UNHOOK, GO/BACK.</li> </ul> </li> <li>Consistent rule for command truncation.</li> <li>Allow both full command and truncated command.</li> <li>Consistent rule for keyboard shortcuts.</li> </ul> </div>
---	---

<p>6</p>	<div data-bbox="462 241 558 275" data-label="Section-Header"> <h2>Menus</h2> </div> <div data-bbox="435 331 583 497" data-label="Image"> </div>
----------	---

		<ul style="list-style-type: none"> <li>Assistance is convenient. You can put defaults in some of the fields, you can show examples of information in case the user doesn't know what to put into a particular field</li> </ul> <p>The main drawback is that the form consumes screen space. This can be a problem if the screen is not very large, such as on a mobile phone.</p> <p>Some guidelines for designing fill-in forms include:</p> <ul style="list-style-type: none"> <li>Comprehensible instructions: <ul style="list-style-type: none"> <li>Use familiar terminology.</li> <li>Be brief.</li> </ul> </li> <li>Logical grouping and sequencing of fields: <ul style="list-style-type: none"> <li>Alignment.</li> <li>Blocking.</li> </ul> </li> <li>Visually appealing layout of the form.</li> <li>Familiar field labels.</li> <li>Consistent terminology and abbreviations.</li> <li>Visible space and boundaries for data entry fields.</li> <li>Convenient cursor movement: <ul style="list-style-type: none"> <li>TAB key.</li> <li>Arrow keys.</li> </ul> </li> <li>User error correction and editing.</li> <li>Explanatory messages for fields.</li> <li>Save contents of fields.</li> <li>Automatically fill fields: <ul style="list-style-type: none"> <li>On new form with some fields duplicated.</li> <li>On the same form when user returns to it.</li> </ul> </li> <li>Automatically fill in (some) fields with reasonable default values.</li> </ul>
8	<div> <div></div> <div>Direct Manipulation</div> <div>  </div> </div>	<p>In a direct manipulation interface, objects on the screen represent things in the application. Operations that you perform on objects on the screen correspond to operations on the things in the application. For example, moving an object from one place to another on the screen results in some information being modified inside the application.</p> <p>The advantages of direct manipulation interfaces include:</p> <ul style="list-style-type: none"> <li>Easy to learn.</li> <li>Easy to retain.</li> <li>Errors can be avoided.</li> <li>Encourages exploration.</li> <li>High subjective satisfaction.</li> </ul>

		<p>The disadvantages include:</p> <ul style="list-style-type: none"> <li>▪ Requires graphics display and pointing devices.</li> <li>▪ Some actions may be cumbersome.</li> <li>▪ May be difficult for those with impairments.</li> </ul> <p>Some guidelines for developing direct-manipulation interfaces include:</p> <ul style="list-style-type: none"> <li>▪ Minimize movement. -- Fitts' Law says that the time to move the pointing device and hit a target is directly proportional to the distance to the target and inversely proportional to the size of the target</li> <li>▪ Consider Affordances – UI mechanisms should look like they will do what they are designed to do. The user shouldn't have to guess what a control will do. This means that we should minimize the cognitive distance between the user's mental model and the designer's model. This will have an impact on things like graphic design and labeling.</li> </ul> <p>Because of the flexibility of the UI, the user may have many options of what to do next.</p> <ul style="list-style-type: none"> <li>▪ Enhances the user's impression of being in control.</li> <li>▪ Allows for multitasking.</li> <li>▪ But may become confusing for novices.</li> </ul>
9	<p>Auditory Commands</p> 	<p>In situations where the user is unable to use his hands, control of the system may be accomplished through voice commands. This is a more recent development, mainly because of the hardware and software resources required for voice recognition.</p> <p>The advantages are obvious. In addition to the hands-free aspect, there is emotional satisfaction of being able to talk to someone, enough so that we sometimes forget that we are talking to a machine.</p> <p>Voice response capabilities are not perfect, yet. If we haven't trained the system to recognize our voice, it sometimes interprets things wrong, leading to sometimes humorous or even embarrassing results.</p>

10	<div data-bbox="451 243 573 275" data-label="Section-Header"> <h2>Gestures</h2> </div> <div data-bbox="399 327 630 474" data-label="Image"> </div>	<p>As touch screens or touchpad interface devices are becoming more commonplace, we see the use of gesture inputs becoming more popular. It can be easier to use your fingers to resize a window than it is to use the mouse or keyboard to do the same thing.</p> <p>We will probably see more use of body gestures as input mechanisms in the near future. Microsoft's Kinect system offers some capability here. In fact, with the proper software, any video camera can be used to detect gestures.</p>
11	<div data-bbox="483 648 545 678" data-label="Section-Header"> <h2>Next</h2> </div> <div data-bbox="297 705 444 730" data-label="List-Group"> <ul style="list-style-type: none"> <li>• Error handling</li> </ul> </div> <div data-bbox="406 762 621 926" data-label="Image"> </div>	<p>In the next video, we will talk about handling user errors.</p> <p>Some of the material for this video was derived from Shneiderman, <i>Designing the User Interface: Strategies for Effective Human-Computer Interaction</i></p>

6-1	<div data-bbox="417 296 604 329" data-label="Section-Header"> <h2>Error Handling</h2> </div> <div data-bbox="435 373 587 491" data-label="Image"> </div>	<p>In this video we will discuss how to handle user errors.</p> <p>People will always make errors.</p> <ul style="list-style-type: none"> <li>• They're only human, after all.</li> </ul>
2	<div data-bbox="446 562 574 590" data-label="Text"> <p>++++++</p> </div> <div data-bbox="454 640 568 676" data-label="Section-Header"> <h2>Severity</h2> </div> <div data-bbox="367 716 487 890" data-label="List-Group"> <ul style="list-style-type: none"> <li>• Useful</li> <li>• Annoying</li> <li>• Dangerous</li> </ul> </div> <div data-bbox="524 686 652 751" data-label="Image"> </div> <div data-bbox="548 760 630 932" data-label="Image"> </div> <div data-bbox="548 840 630 932" data-label="Image"> </div>	<p>We could classify errors by their Severity.</p> <p>The system will notify the user when an error has been made. There are three levels of error messages.</p> <p>Useful error messages tell us when we have goofed and help us correct our mistakes. A good example is a spell checker in a word processing program. We're usually grateful that we got the error message. If we hadn't received the error we may not have spotted the typo ourselves, so these kinds of error messages can actually save us a lot of work.</p> <p>Annoying messages are those that we receive when we already probably know we did something wrong. Did you ever push a button, and two microseconds after you pushed it you realized you pushed the wrong button? It's too late to change your mind. We get the annoying error message anyway.</p> <p>We really need to be warned when we are about to do something dangerous. For example loss of data, loss of money, loss of secrets, injury to people, or in the worst case, loss of life. We need to be warned in the event that we try to do a dangerous action. Probably through a multiple levels of warning, depending on the severity of the risk.</p>

### Types of Errors

- Mistakes
  - E.g., mixing up i.e. and e.g.
- Slips
  - E.g., typos ("Teh quick brown fox...")



Psychologists tell us that there are two types of errors that people make.

Mistakes are where you have the wrong mental model. You think you are doing the right thing, but you are mistaken.



- You push the wrong button on purpose, thinking it is the right one to push.
- A lot of people mix up e.g. and i.e. "e.g. stands for exempli gratia, which is Latin for "for the sake of example." I.e. stands for "id est," which is Latin for "that is."
- I had a friend who had just gotten a new digital camera. On the menu was the command, "format memory card." He didn't know what this meant. He thought it would put his pictures in some sort of formatted order, so he selected the action. Of course, what it did was to format the memory on the card, erasing all of his pictures. Fortunately he was able to send it in and have his pictures restored, but it cost him about \$80.
- These types of errors may be prevented through training and education.
- They can also be prevented with better warnings on the actions themselves.
- Perhaps "format" was not the right word for the action. Obviously not everyone knows that formatting a disk erases everything on it. In fact, I would say that most people don't even know why this is called "formatting the disk."
- Causal analysis may be helpful in spotting the areas where users are error prone.







Slips are where you have the right mental model; you just messed up in execution.



- You pushed the wrong button because you missed with the mouse.
- You knew how to spell the word, but your fingers just pushed the wrong buttons (sometimes this is called a finger check).
- Fortunately a lot of these kinds of errors can be caught by software. Spelling and grammar checkers, for example, are getting pretty good at spotting slips in typed text.

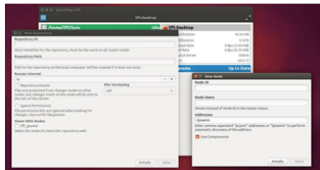
4	<div data-bbox="267 195 760 220" data-label="Image"></div> <h3 data-bbox="418 247 609 277">Handling Errors</h3> <ul data-bbox="300 300 581 499" style="list-style-type: none"> <li>• Do nothing <ul style="list-style-type: none"> <li>◦ Or make warning sound</li> </ul> </li> <li>• Disable <ul style="list-style-type: none"> <li>◦ Grey out the inactive options</li> </ul> </li> <li>• Issue confirmation prompt <ul style="list-style-type: none"> <li>◦ "Do you really want to do this?"</li> </ul> </li> <li>• Provide "undo" capability <ul style="list-style-type: none"> <li>◦ And a "redo"</li> </ul> </li> </ul>	<p data-bbox="787 199 1421 262">The best way to handle errors is to prevent them from occurring in the first place.</p> <ul data-bbox="836 266 1437 1291" style="list-style-type: none"> <li>▪ For example, we could block inappropriate commands. <ul style="list-style-type: none"> <li>▪ Nothing happens when you attempt to execute the command, or, better yet, some sort of warning sounds. A beep is enough to tell the user that they can't do what they just attempted to do.</li> </ul> </li> <li>▪ A better way of preventing the error would be to disable the inappropriate commands. <ul style="list-style-type: none"> <li>▪ They still show up on the interface, but they are greyed out to indicate that they are inoperable.</li> </ul> </li> <li>▪ Where you aren't sure whether the command is inappropriate or not, we would leave it up to the user to decide. We would issue a confirmation prompt. <ul style="list-style-type: none"> <li>▪ Something like, "Do you really want to do this?"</li> </ul> </li> <li>▪ Alternatively, we could provide an undo capability. The user would have to decide after the fact that he made an error and would be able to rectify the situation. <ul style="list-style-type: none"> <li>▪ We should also provide a re-do option, in case the user decides that yes, in fact, he did choose the right action.</li> <li>▪ The confirmation prompt should be used if the operation requested cannot be undone.</li> </ul> </li> </ul>
5	<div data-bbox="267 1297 760 1323" data-label="Image"></div> <h3 data-bbox="354 1354 673 1383">Error Message Guidelines</h3> <ul data-bbox="300 1411 646 1591" style="list-style-type: none"> <li>• Be specific</li> <li>• Provide constructive guidance</li> <li>• Be positive</li> <li>• Be consistent</li> <li>• Consider experience and skill level</li> <li>• Consider culture</li> </ul> <div data-bbox="636 1428 711 1501" data-label="Image"></div>	<h3 data-bbox="787 1306 1128 1335">Error Message Guidelines</h3> <ul data-bbox="836 1339 1437 1894" style="list-style-type: none"> <li>▪ Be specific. <ul style="list-style-type: none"> <li>▪ Tell user what the problem was.</li> </ul> </li> <li>▪ Provide constructive guidance. <ul style="list-style-type: none"> <li>▪ Give user advice about how to fix problem.</li> </ul> </li> <li>▪ Be positive. <ul style="list-style-type: none"> <li>▪ Rather than condemning.</li> <li>▪ Avoid words such as ILLEGAL, ERROR, INVALID, or BAD.</li> <li>▪ Particularly for people who may not be comfortable with computers (yes, there still are some out there).</li> <li>▪ These are emotionally charged words.</li> <li>▪ You can get arrested for doing illegal things.</li> </ul> </li> </ul>

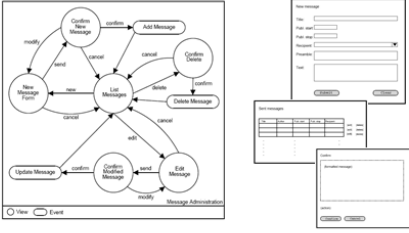





		<ul style="list-style-type: none"> <li>▪ Use consistent visual format and placement of the error messages. Use consistent grammatical form, terminology, and abbreviations.</li> <li>▪ If the form requires scrolling over more than one screen, sometimes the user may not see the error message. Consider some kind of a modal popup dialog for the error message. A modal window requires a confirmation action, such as a mouse click, to dismiss it.</li> <li>▪ Consider the user's experience and skill level. The kind of message you might display for a novice might be very annoying for an experienced user. Perhaps the user could be given the option to set the level of error messages.</li> <li>▪ Consider the culture of the region where the system will be deployed. There are many cultural differences between countries in Asia, Europe, the Americas and Africa. We need to be sensitive to cultural norms.</li> </ul>
6	<div>  </div> <p>Next</p> <ul style="list-style-type: none"> <li>• Prototypes</li> </ul> 	<p>In the next video, we will re-visit a topic we have introduced earlier: prototyping.</p>



7-1	 <p>Prototyping</p> 	<p>The subject of this video is prototyping. We will describe what prototypes are, the types of prototypes, the pros and cons of prototyping, and some tips and suggestions for prototyping the user interface,</p>
2	 <p>Purpose</p> 	<p>A prototype is a concrete, but partial implementation of a design.</p> <ul style="list-style-type: none"> <li>Its basic purpose is to reduce <i>risk</i> in software development. <ul style="list-style-type: none"> <li>The risk of us developing the wrong thing.</li> <li>Or the risk of the users not knowing what they want.</li> </ul> </li> </ul> <p>Prototypes expose misunderstandings.</p> <ul style="list-style-type: none"> <li>There are user misunderstandings. This is where the users and customers: <ul style="list-style-type: none"> <li>Don't know what they are getting</li> <li>Or they don't even know what they want.</li> </ul> </li> <li>Prototypes also prevent developer misunderstandings. If the developers don't know what customer wants, they will develop what they <i>think</i> the customer wants (or just what they want to develop). Without a prototype, they are just guessing. (And they usually guess wrong.)</li> </ul>
3	 <p>Benefits</p> 	<p>Here are some benefits of prototypes:</p> <ul style="list-style-type: none"> <li>Expose missing system capabilities.</li> <li>Expose confusing services.</li> <li>Early availability (through evolutionary prototyping).</li> <li>Basis for specification of software requirements.</li> <li>Support user training.</li> <li>Support system testing.</li> </ul>



<p>4</p>	<p>Low Fidelity (lo-fi) Prototyping</p> 	<p>There are low fidelity prototypes and high fidelity prototypes.</p> <p>Low fidelity prototypes are sometimes called paper prototypes, because they are just sketches on paper.</p> <p>The analysts would talk through the scenarios using these prototypes. The prototypes would show what would be displayed on the screen at various points in the interaction.</p> <p>They can be used to show ways in which the user would interact with the system (for example, buttons or menus).</p> <p>Some of the advantages of lo-fi prototypes are that they are</p> <ul style="list-style-type: none"> <li>• Quick and easy to create or modify.</li> <li>• They use low cost materials.</li> <li>• Their roughness actually helps to control premature commitment and invites participation.</li> <li>• They allow for quick feedback, and permits more cycles of evaluation and revision.</li> </ul> <p>The main drawback of low fidelity prototypes is that they are only sketches, and some people may have a hard time imagining the real interface when looking at them. They don't have the sort of eyeball appeal that high fidelity prototypes might have.</p>
<p>5</p>	<p>Storyboards</p> 	<p>A good example of a low fidelity prototype is a storyboard. Storyboards have been used by all sorts of people who want to plan out sequences of events in advance.</p> <p>For example, Alfred Hitchcock and Steven Spielberg are noted for creating storyboards for their movies. Most television commercials are storyboarded.</p> <p>You can see a nice video of Ridley Scott talking about how he used storyboards to direct one of the most famous TV commercials ever, the Apple Macintosh “1984” commercial, which ran only one time, during the 1984 Super bowl. The link is in the student notes.</p>

		<p>Here's the Ridley Scott interview:  <a href="https://www.youtube.com/watch?v=Llxog089IkA">https://www.youtube.com/watch?v=Llxog089IkA</a></p> <p>here's the commercial itself:  <a href="https://www.youtube.com/watch?v=axSnW-ygU5g">https://www.youtube.com/watch?v=axSnW-ygU5g</a></p> <p>A storyboard is a sequence of sketches illustrating key points in a scenario. They are usually sketched by hand. A narrative description of the action typically accompanies each frame. For movies and commercials, directions about camera angles and movement, lighting, and actor movement can also be annotated on the sketches.</p> <p>The main thing about a storyboard is that it depicts a single specific scenario, which is what makes it useful for planning out scenes in movies and TV commercials. But for user interfaces, there may be multiple scenarios for a particular interaction. Thus you would need multiple storyboards.</p>
6	<p>High Fidelity (hi-fi) Prototyping</p> 	<p>At the other extreme you would have high fidelity prototypes. These are actually mock-ups of the actual user interface. The mock-up looks and works exactly like the final system, except it wouldn't do anything. All of the buttons and menus work so you can navigate through the screens but all of the outputs are all simulated.</p> <p>The mock-up should be written using a rapid prototyping language.</p> <p>One benefit of an executable prototype is that it is, well, executable. It is very useful for testing out ideas for interaction styles.</p> <p>Another big benefit is that changes to the interface would be easy and quick to make. If the mock-up is being reviewed with stakeholders, any suggested changes could be made in a matter of seconds. It wouldn't take a great deal of time to perfect the interface, as long as everyone can agree on the changes.</p> <p>A high fidelity prototype can be used to test out some of the psychological usability principles we discussed earlier, such as use of color, motion, sound, screen design, and so forth.</p> <p>High fidelity prototypes tend to impress clients</p>

		<p>and managers who are not deeply involved in the requirements process. They give the illusion that something is being accomplished.</p> <p>An executable prototype can give the documentation writers a head start in documenting the system.</p> <p>The main down side of these hi fidelity prototypes is that customers or end users could easily confuse the prototype with the finished system, since they look exactly the same. So you want to make sure that whomever you show the prototype to knows that it is only a mock-up and not the finished system.</p> <p>If you don't have the proper tools, executable prototypes can be expensive to produce and may, in fact, slow down the process of usability design.</p>
7	<p style="text-align: center;"><b>Screen Flows and Layouts</b></p>  <p style="text-align: center;"><small><a href="http://fornc.darkeye.net/deliverables/deliverable3.3.pdf">http://fornc.darkeye.net/deliverables/deliverable3.3.pdf</a></small></p>	<p>A useful technique for modeling user interaction is called Screen flows and Layouts. The layout of each of the screens is sketched out. Then we draw the flow diagram as a directed graph in which the nodes represent the screens, and the arrows represent the effect of navigation actions (e.g., button or menu selection). It shows all possible flows from screen to screen (sort of like a roadmap).</p> <p>The screen flow diagram can thus describe multiple scenarios. Each scenario would be only one path through the screen flow. There is no standard UML notation for screen flow diagrams, but you could possibly adopt the UML state transition diagram for this purpose. We will cover state transition diagrams in a later module of this course.</p>
8	<p style="text-align: center;"><b>Other Prototyping Techniques</b></p> <div style="display: flex; align-items: flex-start;">  <ul style="list-style-type: none"> <li>• System mock-ups <ul style="list-style-type: none"> <li>◦ Fabricated devices with simulated controls</li> </ul> </li> <li>• Wizard of Oz <ul style="list-style-type: none"> <li>◦ Human behind the scenes simulates system actions</li> </ul> </li> <li>• Video prototype <ul style="list-style-type: none"> <li>◦ Great for selling</li> </ul> </li> <li>• Computer animation <ul style="list-style-type: none"> <li>◦ Automatic simulation</li> </ul> </li> </ul> </div>	<p>Here are some other prototyping techniques.</p> <ul style="list-style-type: none"> <li>▪ Mock-ups <ul style="list-style-type: none"> <li>▪ Where we would use fabricated devices with simulated controls. Astronauts trained in shuttle flight simulators before going out into space.</li> </ul> </li> <li>▪ Wizard of Oz – “Pay no attention to that man behind the curtain.” <ul style="list-style-type: none"> <li>▪ We have a human behind the scenes simulate system actions and display appropriate output on the interface.</li> </ul> </li> </ul>

		<ul style="list-style-type: none"> <li>▪ Video prototypes <ul style="list-style-type: none"> <li>▪ Which are recorded walkthroughs. These are great for selling.</li> </ul> </li> <li>▪ Computer animations <ul style="list-style-type: none"> <li>▪ Are similar to video prototypes, but they are just software prototypes that run automatically.</li> </ul> </li> </ul>
9	 <h3>Scott Ambler's Prototyping Tips</h3> <ul style="list-style-type: none"> <li>• Work with the real users</li> <li>• Get your stakeholders to work with the prototype</li> <li>• Understand the underlying business</li> <li>• Only prototype features that you can actually build</li> <li>• You cannot make everything simple</li> <li>• It's about what you need</li> <li>• Get an interface expert to help you design it</li> <li>• Explain what a prototype is</li> <li>• Consistency is critical</li> <li>• Avoid implementation decisions as long as possible</li> <li>• Small details can make or break your user interface</li> </ul> <p><a href="http://www.ambysoft.com/essays/userInterfacePrototyping.html">http://www.ambysoft.com/essays/userInterfacePrototyping.html</a></p>	<p>Scott Ambler is a Canadian software engineering consultant. He wrote down some tips for folks doing prototyping, based on his own experiences. You can see more about him on his website, <a href="http://www.ambysoft.com">ambysoft.com</a>. I am not going to read through the list of tips here. I recommend that you visit the website shown in the link and read his tips and techniques in full. It's only one page long, so it won't take you very long.</p>
10	 <h3>Next</h3> <ul style="list-style-type: none"> <li>• Principles, guidelines and heuristics</li> </ul>	<p>The last video in this module is all about usability principles, guidelines and heuristics.</p>

8-1	 <p>Guidelines, Principles and Heuristics</p>	<p>In this video we will look at some advice from two world famous experts in usability design.</p>
2	<p>Shneiderman's 8 Golden Rules</p> <ol style="list-style-type: none"> <li>1. Strive for consistency</li> <li>2. Enable frequent users to use shortcuts</li> <li>3. Offer informative feedback</li> <li>4. Design dialog to yield closure</li> <li>5. Offer simple error handling</li> <li>6. Permit easy reversal of actions</li> <li>7. Support internal locus of control</li> <li>8. Reduce short-term memory load</li> </ol>  <p><a href="https://www.cs.umd.edu/users/ben/goldenrules.html">https://www.cs.umd.edu/users/ben/goldenrules.html</a></p>	<p>Ben Shneiderman teaches at the University of Maryland.</p> <p>He calls these the 8 golden rules. I always wondered why he called them that. There are eight of them, I know that. They are rules, well OK. The golden part, I guess, comes from what is known as the golden rule: "do unto others as you would like them to do unto you." I guess this applies to building user interfaces, too.</p> <p>One common theme in Shneiderman's 8 golden rules and Nielsen's heuristics is the importance of consistency. Consistent look and feel. Consistent terminology, layout, mechanisms, etc.</p> <p>Sometimes with graphical user interfaces, frequent users get frustrated with the number of mouse movements and clicks required for particular tasks. There ought to be a way to provide shortcuts for these folks.</p> <p>The users should have the feeling that they are in charge of the dialog.</p> <p>We have discussed error handling earlier. This is a big topic in Shneiderman's book.</p> <p>The last point about short-term memory has to do with something called the "magic number 7 plus or minus 2." -- George Miller was a cognitive psychologist who wrote a landmark paper in 1956 reporting on his experiments with human subjects. Perhaps you remember this if you took a psychology class. He would present the subject with a number of items for a very short amount of time. Then he would ask them to recall the objects they'd seen. Most people could remember up to about 5 objects without many problems. Almost no one could recall all of the</p>

		<p>objects when he presented them with more than about 9 objects. Miller concluded that there was a short-term working memory in the brain that could hold about 7 plus or minus 2 pieces of information.</p> <p>What does this have to do with user interfaces? Well the user can't focus on more than about 7 things at one time. Thus, we are advised to limit the number of things in the user interface to 7 plus or minus two: the number of menus, the number of items in a menu, the number of buttons, the number of windows, and so forth. Now if you look at a lot of software, there are certainly more than seven items in a menu, but if you look closely, they are grouped. There are no more than seven groups, and there are no more than seven items in any group. If the designers run out of these limits, they use cascading menus or dialog boxes.</p> <p>You can use the link shown here to read the full descriptions of the rules. It is only one page long.</p>
3	<div> <div>  <h3>Nielsen's Heuristics</h3> <ol style="list-style-type: none"> <li>1. Visibility of system status</li> <li>2. Match between system and the real world</li> <li>3. User control and freedom</li> <li>4. Consistency and standards</li> <li>5. Error prevention</li> <li>6. Recognition rather than recall</li> <li>7. Flexibility and efficiency of use</li> <li>8. Aesthetic and minimalist design</li> <li>9. Help users recognize, diagnose, and recover from errors</li> <li>10. Help and documentation</li> </ol> <p><a href="http://www.nngroup.com/articles/ten-usability-heuristics/">http://www.nngroup.com/articles/ten-usability-heuristics/</a></p> </div> <div>  </div> </div>	<p>Jacob Nielsen is a Danish computer scientist. He was one of the earliest researchers to develop user interface guidelines.</p> <p>A number of Nielsen's ten heuristics are very similar to Shneiderman's golden rules, which I guess means they are good advice.</p> <p>Nielsen has a few items that didn't appear on Shneiderman's list. One is the match between the system and the real world. This is something that we stressed earlier in this module. Recognition rather than recall means that menus are better than command line interfaces.</p> <p>The last item about help and documentation is very important, as we all know. We all hear the advice that interfaces should be user friendly. I would suggest rewording this to say that interfaces should be user considerate. This goes back to the idea of the golden rule. Think about what you are doing to or for the user and be considerate.</p> <p>You can use the link shown to read the full descriptions of these heuristics. It is only a few pages long.</p>



4	<div data-bbox="673 195 722 220" data-label="Image"></div> <h3 data-bbox="443 247 586 275">References</h3> <ul data-bbox="300 298 730 506" style="list-style-type: none"> <li>• Preece, J. Rogers, Y. &amp; Sharp, H. (2007) <i>Interaction Design: Beyond Human-Computer Interaction</i>. 2nd Edition. New York, NY: John Wiley &amp; Sons.</li> <li>• Shneiderman, B &amp; Plaisant, C. (2010) <i>Designing the User Interface: Strategies for Effective Human-Computer Interaction</i>, 5th edition. Addison-Wesley.</li> <li>• Mary Beth Rosson and John Carroll (2002) <i>Usability Engineering</i>, Morgan Kaufmann.</li> <li>• U.S Dept. of Health and Human Services, <i>The Research-Based Web Design &amp; Usability Guidelines</i>. Washington: U.S. Government Printing Office.</li> <li>• <a href="http://www.usability.gov">www.usability.gov</a>.</li> <li>• <a href="http://www.webpagesthatsuck.com/">www.webpagesthatsuck.com/</a>.</li> <li>• <a href="http://asktog.com/atc/principles-of-interaction-design/">http://asktog.com/atc/principles-of-interaction-design/</a></li> </ul>	<p>Here are a number of references on the subject of usability. I have used a number of them for the notes in this module.</p> <p>The US Government document is from a big effort by the Department of Health and Human Services to standardize interfaces in their websites. The document is available as a pdf for free. There is a companion website <a href="http://usability.gov">usability.gov</a>, which has a lot of good material in it.</p> <p>The site <a href="http://webpagesthatsuck.com">webpages that suck</a> is a fun place to visit to see what not to do when building a website.</p> <p>The asktog site is run by Bruce Tognazzini, another famous usability expert. Bruce was responsible for a lot of the usability decisions on the Apple Macintosh. He now works with Jacob Nielsen and Donald Norman, another famous usability expert, at a company called the Nielsen Norman Group. Tog has been working on this collection of good advice for many years. It's definitely worth a look.</p>
5	<div data-bbox="673 961 722 987" data-label="Image"></div> <h3 data-bbox="427 1018 602 1045">End of Module</h3> <ul data-bbox="300 1068 722 1276" style="list-style-type: none"> <li>• You should now be able to: <ul style="list-style-type: none"> <li>◦ Discuss the history and purpose of user interfaces</li> <li>◦ Identify Usability and User Experience goals</li> <li>◦ Perform Usability Analysis</li> <li>◦ Perform Usability Design</li> <li>◦ Discuss advantages and disadvantages of four types of user interaction mechanisms</li> <li>◦ Discuss techniques for handling user errors</li> <li>◦ Develop a UI prototype and document screen flows and layouts</li> <li>◦ Apply HCI guidelines and heuristics</li> </ul> </li> </ul>	<p>Well, we've come to the end of the module.</p> <p>Here's what you should now be able to do.</p> <ul style="list-style-type: none"> <li>• Discuss the history and purpose of user interfaces</li> <li>• Identify Usability and User Experience goals</li> <li>• Perform Usability Analysis</li> <li>• Perform Usability Design</li> <li>• Discuss advantages and disadvantages of four types of user interaction mechanisms</li> <li>• Discuss techniques for handling user errors</li> <li>• Develop a UI prototype and document screen flows and layouts</li> <li>• Apply HCI guidelines and heuristics</li> </ul> <p>If you haven't started already, you should participate in the discussion for this module.</p> <p>There is a quiz that you need to take.</p> <p>You can work with your team to use the concepts from this module to develop the user experience for your project.</p>