Human Computer Interaction

The Design Framework, Platform, and Posture

Prof. Andrew D. Bagdanov

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Firenze
andrew.bagdanov AT unifi.it

December 12, 2017

Outline

Overview

Overview

- Today we will see the next steps in turning designs into realized interfaces.
- Last time we finished with a reductive list of design requirements that summarized the elements necessitated by our personas and scenarios.
- Today we will discuss how the design framework provides a skeletal structure for the objects and actions in our product.
- We will then introduce some general principles and patterns of interface design.
- And finally we will discuss some general design considerations involving platform and posture

The Design Framework

The Design Framework

- Rather than jump into implementation, at this point must concern ourselves with the overall structure of the user interface and associated behaviors.
- This is called the Design Framework and it defines the overall structure of user experience, from the arrangement of functional elements on the screen, to interactive behaviors and underlying organizing principles, to the visual and form language used to express data and functionality.
- If we were designing a house, at this point, wed be concerned with what rooms the house should have, how they should be positioned with respect to each other, and roughly how big they should be.
- We would not be worried about the precise measurements of each room, or things like the doorknobs, faucets, and countertops.

The Design Framework

- Form and behavior should be designed in concert with each other.
- The Design Framework is made up of an interaction framework, a visual design framework, and sometimes an industrial design framework.
- We will not say much about industrial design frameworks here, as these mostly concern physical devices.
- At this phase in a project, interaction designers use scenarios and requirements to create rough sketches of screens and behaviors that make up the interaction framework.
- Concurrently, visual designers use visual language studies to develop a visual design framework that is commonly expressed as a detailed rendering of a single screen archetype

The Design Framework

- By taking a top-down approach and rendering our solutions without specific detail in a low-fidelity manner, we ensure that we and our stakeholders stay initially focused on serving persona goals and requirements.
- Revision is a fact of life in design: typically, the process of representing and presenting design solutions helps designers and stakeholders refine their vision and understanding of how the product can best serve human needs.
- The trick is to render the solution only in enough detail to provoke engaged consideration, without spending too much time or effort creating renderings that are certain to be modified or abandoned.
- Storyboards, accompanied by narrative in the form of scenarios, are a highly effective way to explore and discuss design solutions without creating undue overhead and inertia.

Defining the Interaction Framework

- The interaction framework defines not only the high-level structure of screen layouts but also the flow, behavior, and organization of the product.
- The following six steps describe the process of defining the interaction framework:
  1. Define form factor, posture, and input methods
  2. Define functional and data elements
  3. Determine functional groups and hierarchy
  4. Sketch the interaction framework
  5. Construct key path scenarios
  6. Check designs with validation scenarios
- Though the process is broken down into numerically sequenced steps, this is not typically a linear effort, but rather occurs in iterative loops.
- In particular, Steps 35 may be swapped around, depending on the style of the designer.

Defining the Interaction Framework: Step 1

Step 1: Define form factor, posture, and input methods:

- The first step in creating a framework is to define the form factor of the product.
- Is it a Web application that will be viewed on a high-resolution computer screen?
- Is it a kiosk that must be rugged to withstand a public environment while accommodating thousands of distracted, novice users?
- Is it a mobile application that must be used on small screens in all lighting conditions?
- Each of these form factors has clear implications for the design of the product, and answering this question sets the stage for all subsequent design efforts.
- If the answer isn't obvious, look to your personas and scenarios to better understand the ideal usage context and environment.

Defining the Interaction Framework: Step 1

Step 1 (continued): Define form factor, posture, and input methods:

- As you define the form, you should also define the basic posture of the product, and determine the input method(s) for the system.
- A product's posture is related to how much attention a user will devote to interacting with the product (more on posture later).
- This decision should be based upon usage contexts and environments as described in your context scenario(s)
- Input methods will be driven by the form factor and posture, as well as by your persona attitudes, aptitudes, and preferences.
- We must decide which combination of input methods is appropriate for our primary and secondary personas.
- In cases where it may be appropriate to use a combination of input methods (such as the common Web site or desktop application that relies on both mouse and keyboard input), decide upon the primary input method for the product.

Defining the Interaction Framework: Step 2

Step 2: Define functional and data elements:

- Functional and data elements are the representations of functionality and data that are revealed to the user in the interface.
- These are the concrete manifestations of the functional and data requirements identified during the Requirements Definition phase
- Where the requirements were purposely described in general terms, from the perspective of our personas, functional and data elements are described in the language of user-interface representations.
- To be sure that every aspect of the product has clear purpose, it is important to note that these elements must each be defined in response to specific requirements defined earlier.

Defining the Interaction Framework: Step 2

Step 2 (continued): Define functional and data elements:

- Data elements are typically the fundamental subjects of interactive products: photos, e-mail messages, customer records, etc., are the basic units to be referred to, responded to, and acted upon by the people using the product.
- At this point, it is critical to comprehensively catalog the data objects, because functionality is commonly defined in relation to them.
- We are also concerned with the significant attributes of the objects (for example, the sender of an e-mail message or the date a photo was taken).
- We should consider also the relationships between data elements: sometimes a data object may contain other data objects; other times there may be a more associative relationship between objects.

Defining the Interaction Framework: Step 2

Step 2 (continued): Define functional and data elements:

- Functional elements are the operations that can be done to the data elements and their representations in the interface.
- Generally speaking, they include tools to act upon data elements and places to put data elements.
- It is common that a single requirement will necessitate multiple interface elements.
- For example, Vivien, our persona from the last lecture, needs to be able to telephone her contacts, which might necessitate these functional elements:
  1. Voice activation (voice data associated with contact)
  2. Assignable quick-dial buttons
  3. Selecting a contact from a list
  4. Selecting a contact from an e-mail header, appointment, or memo
  5. Auto-assignment of a call button in appropriate context (for example, upcoming appointment)

Defining the Interaction Framework: Step 2

Step 2 (continued): Define functional and data elements:

- Defining the functional requirements is one of the most critical phases in design.
- It is here where design first becomes concrete, and it imperative to return to context scenarios, persona goals, and mental models to ensure that your solutions are appropriate.
- In response to any identified user requirement, there are typically quite a number of possible solutions.
- We must ask ourselves which of the solutions is most likely to:
  - Accomplish user goals most efficiently?
  - Best fit our design principles?
  - Fit within technology or cost parameters?
  - Best fit other requirements?

Defining the Interaction Framework: Step 3

Step 3: Determine functional groups and hierarchy:

- When we have a good list of top-level functional and data elements, we begin to group them into functional units and determine their hierarchy.
- Because these elements facilitate specific tasks, the idea is to group elements to best facilitate persona workflow
- Some issues to consider include:
  - Which elements need a large amount of video real estate and which do not?
  - Which elements are containers for other elements?
  - How should containers be arranged to optimize flow?
  - Which elements are used together and which aren't?
  - In what sequence will a set of related elements be used?
  - What interaction patterns and principles apply?
  - How do the personas mental models affect organization?

Defining the Interaction Framework: Step 3

Step 3 (continued): Determine functional groups and hierarchy:

- It is important to organize data and functions into top-level container elements, such as screens, frames, and panes.
- These groupings may change somewhat as we sketch out the interface, but it's still useful to provisionally sort elements into groups.
- Consider which primary screens or views the product requires: initial context scenarios should give a strong feel for what these might be.
- If you know that a user has several end goals and needs where data and functionality don't overlap, it might be reasonable to define separate views to address them.
- On the other hand, if you see a cluster of related needs (for example, to make an appointment, your persona needs to see a calendar and contacts), you might consider defining a view that incorporates all these together

Defining the Interaction Framework: Step 3

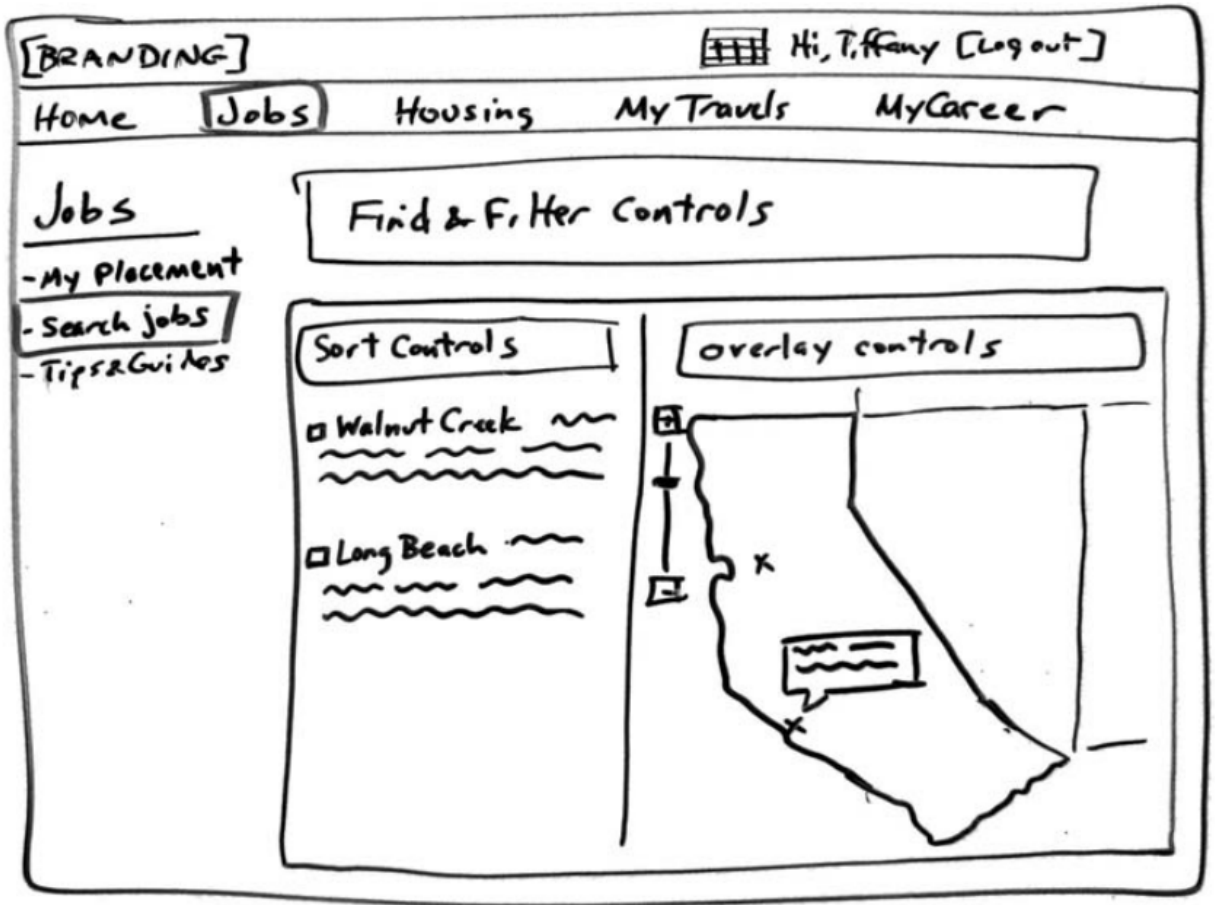
Step 3 (continued): Determine functional groups and hierarchy:

- When grouping functional and data elements, we must consider how they should be arranged given the platform, screen size, form factor, and input methods.
- Containers for objects that must be compared or used together should be adjacent to each other.
- Objects representing steps in a process should be adjacent and ordered sequentially.
- Use of interaction design principles and patterns is extremely helpful at this juncture (more on this in another lecture).

Defining the Interaction Framework: Step 4

Step 4: Sketch the interaction framework

- We should start out by subdividing each view into rough rectangular areas corresponding to panes, control components, and other top-level containers.
- The interface sketch should be extremely simple: boxes representing each functional element with names and descriptions of the relationships.



Defining the Interaction Framework: Step 4

Step 4 (continued): Sketch the interaction framework

- We need to look at the entire, top-level framework first: if not, we can get distracted by the details of a particular area of the interface.
- At this high-level "rectangle phase" its very easy to explore a variety of ways of presenting information and functionality and to perform radical reorganizations.
- It's often useful to try several arrangements on for size, running through validation scenarios (see Step 6), before landing on the best solution.
- Spending too much time and effort on intricate details early in the design process discourages designers from changing course to what might be a superior solution.
- It's easier to discard our work and try another approach when we don't have a lot of effort invested.

Defining the Interaction Framework: Step 4

Step 4 (continued): Sketch the interaction framework

- Sketching the framework is an iterative process that is best performed with a small, collaborative group.
- Working at a whiteboard promotes collaboration and discussion and everything is easy to erase and redraw.
- Once the sketches reach a reasonable level of detail, it becomes useful to start rendering in a computer-based tool (more on GUI sketching/prototyping tools in a later lecture).
- The key here is to find the tool that is most comfortable for you, so you can work quickly, roughly, and at a high level.
- It useful to render Framework illustrations in a visual style that suggests the sketchiness of the proposed solutions.

Defining the Interaction Framework: Step 5

Step 5: Construct key path scenarios

- A key path scenario describes how the persona interacts with the product, using the vocabulary of the interaction framework.
- These scenarios depict the primary pathways through the interface that the persona takes with the greatest frequency.
- Their focus is at the task level: for example, in an e-mail application, key path activities include viewing and composing mail, not configuring a new mail server.
- These scenarios typically evolve from the context scenarios, but here we specifically describe the persona's interaction with the various functional and data elements that make up the nascent interaction framework.

Defining the Interaction Framework: Step 5

Step 5 (continued): Construct key path scenarios

- As we add more and more detail to the interaction framework, we iterate the key path scenarios to reflect this detail in greater specificity around user actions and product responses.
- Unlike the goal-oriented context scenarios, key path scenarios are more task oriented, focusing on task details broadly described and hinted at in the context scenarios.
- This doesn't mean that we can ignore goals – goals and persona needs are the constant metric throughout the design process.
- However, key path scenarios must describe in exacting detail the precise behavior of each major interaction and provide a walkthrough of each major pathway.

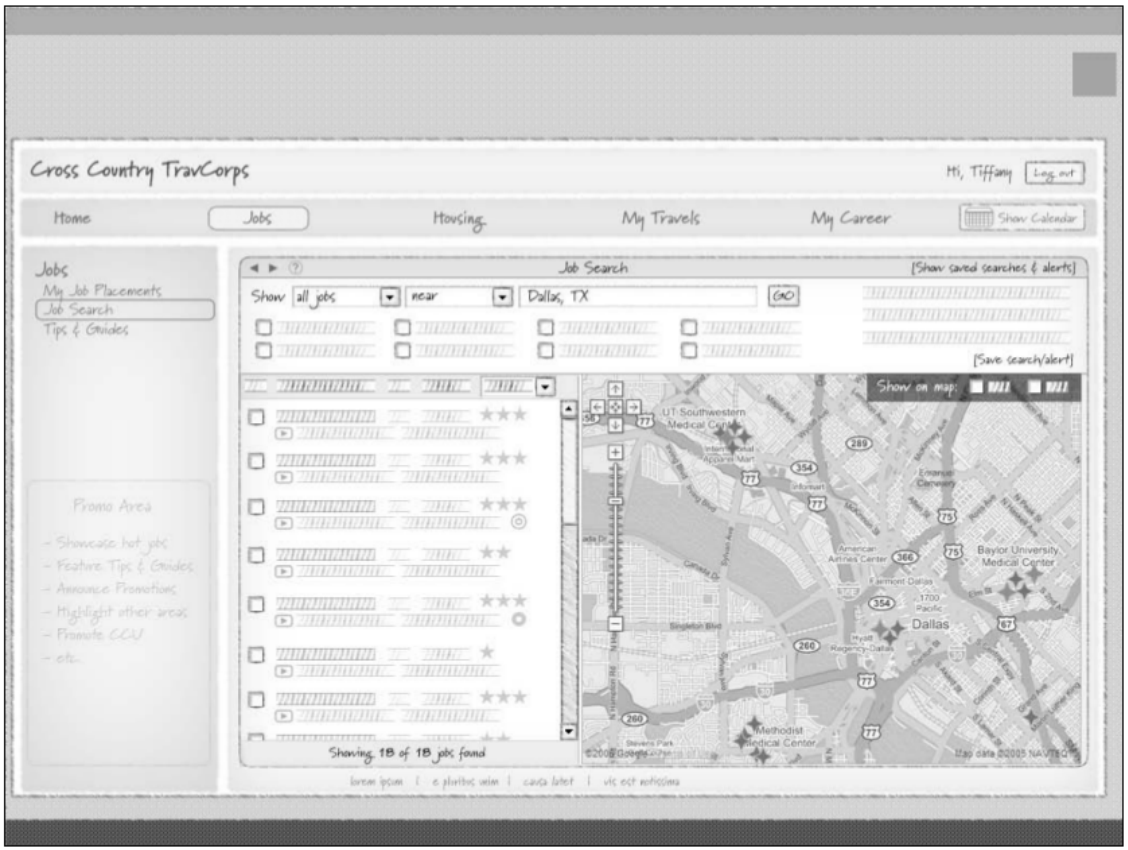
Defining the Interaction Framework: Step 5

Step 5 (continued): Construct key path scenarios

- Storyboarding is useful in this phase: a sequence of low-fidelity sketches accompanied by the narrative of each key path scenario allows us to richly portray how a solution helps personas accomplish their goals.
- It is used to plan and evaluate ideas without having to deal with the cost and labor of building an actual interface.
- Each interaction between the user and the product can be portrayed on one or more frames or slides.
- Advancing through them provides a reality check for the coherence and flow of the interactions.

Step 5 (continued): Construct key path scenarios

- Because creative human activities are rarely a linear process, the steps in the Framework phase shouldn't be thought of as a simple sequence.
- It is common to move back and forth between steps and to repeat the whole process several times to refine the design through iteration of key path scenario definition.



Defining the Interaction Framework: Step 6

UNIVERSITÀ DEGLI STUDI FIRENZE

Step 6: Check designs with validation scenarios

- After storyboarding key path scenarios and adjusting the interaction framework until the scenario flows smoothly, it is time to shift focus to less important interactions.
- These validation scenarios are not typically developed in as much detail as key path scenarios, instead this phase consists of asking a series of "what if . . ." questions.
- The goal here is to poke holes in the design and adjust it as needed (or throw it out and start over).

Step 6: Check designs with validation scenarios

- There are three major categories of validation scenarios that should be addressed in the following order:
  1. Key path variant scenarios are alternate or less-traveled interactions that split off from key pathways at some point along the persona's decision tree. Returning to Vivien: an example of a key path variant would be if Vivien decided to respond to Frank by e-mail in Step 2 instead of calling him.
  2. Necessary use scenarios include actions that must be performed, but only infrequently. Purging databases, configuring, and making other exceptional requests might fall into this category. Necessary use interactions demand pedagogy since users may forget how to access the function or how to perform tasks related to it. Vivien: if the phone was sold second-hand, requiring the removal of all personal information associated with the original owner.

Step 6 (continued): Check designs with validation scenarios

- And the final type of validation scenario:
  3. Edge case use scenarios, as the name implies, describe atypical situations that the product must nevertheless be able to handle. Programmers focus on edge cases because they often represent sources of system instability and bugs, but they should never be the focus of the design effort. Vivien: an example of an edge case scenario would be if Vivien tried to add two different contacts the same name. This is not something she is likely to want to do, but something the phone should handle if she does.

Defining the Visual Design Framework

- As the interaction framework establishes an overall structure for product behavior, a parallel process focused on the visual and industrial design is also necessary.
- This process is similar to designing the interaction framework, in that the solution is first considered at a high level and then narrows to an increasingly granular focus.
- Defining the visual design framework typically follows this process:
  1. Develop visual language studies
  2. Apply chosen visual style to screen archetype

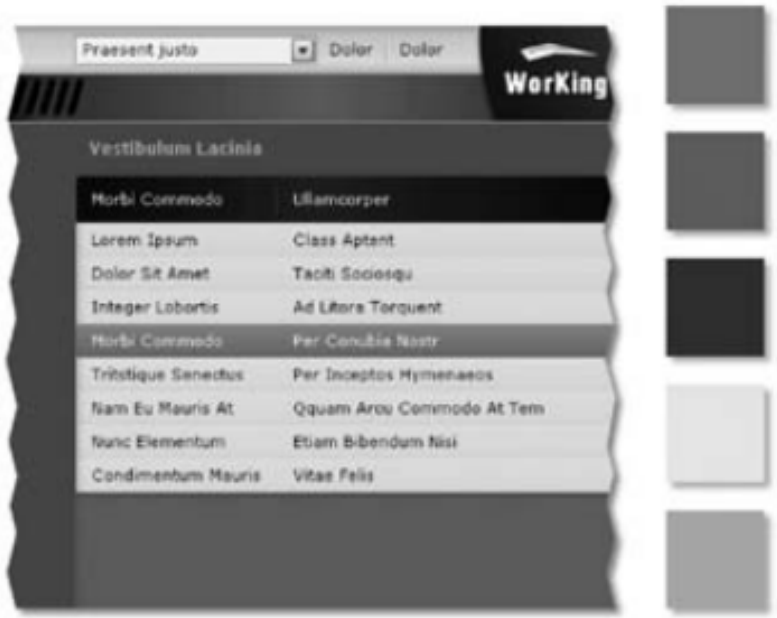
Step 1: Develop visual language studies

- The first step in defining a visual design framework is to explore a variety of visual treatments through visual language studies.
- These studies include color, type, and widget treatments, and any "material" properties of the interface (for example, does it feel like glass or paper?).
- These studies should show these aspects abstractly and independently of the interaction design: we must avoid running the risk of distracting our stakeholders with highly rendered versions of rough interaction designs.

Defining the Visual Design Framework: Step 1

Step 1: Develop visual language studies

- The first step in defining a visual design framework is to explore a variety of visual treatments through visual language studies.
- These studies include color, type, and widget treatments, and any "material" properties of the interface (for example, does it feel like glass or paper?).
- These studies should show these aspects abstractly and independently of the interaction design: we must avoid running the risk of distracting our stakeholders with highly rendered versions of rough interaction designs.



Defining the Visual Design Framework: Step 1

Step 1: Develop visual language studies

- The first step in defining a visual design framework is to explore a variety of visual treatments through visual language studies.
- These studies include color, type, and widget treatments, and any "material" properties of the interface (for example, does it feel like glass or paper?).
- These studies should show these aspects abstractly and independently of the interaction design: we must avoid running the risk of distracting our stakeholders with highly rendered versions of rough interaction designs.

Defining the Visual Design Framework: Step 1

Step 1: Develop visual language studies

- Visual language studies should relate to the experience goals of the personas, as well as any experience or brand keywords that were developed in the Requirements Definition phase.
- Brand guidelines commonly consist of a document explaining how a brand identity should be visually and textually conveyed.
- It's also important to consider environmental factors and persona aptitudes when devising visual styles:
  - Screens that must be visible under bright lights or from a distance require high contrast and more saturated colors
  - The elderly and other visually-impaired users require larger and more readable type faces.
- Very important: *never show a design approach that youre not happy with – stakeholders just might like it*.

Defining the Visual Design Framework: Step 2

Step 2: Apply chosen visual style to screen archetype

- The next step is to apply one or two selected visual styles to key screens.
- We should coordinate our visual and interaction design efforts so this step is performed close to the end of the interaction framework.
- This further refines the visual style so that it reflects key behaviors and information.
- By making the design more concrete, you can better assess the feasibility of the proposed solution without the overhead of updating numerous screens for each minor change.

Refining Form and Behavior

- When a stable framework definition is reached, the remaining pieces of the design begin to smoothly fall into place.
- Each iteration of the key path scenarios adds detail that strengthens the overall coherence and flow of the product.
- At this stage, a transition is made into the Refinement phase, where the design is translated into a final, concrete form.
- In this phase, principles and patterns remain important in giving the design a formal and behavioral finish (more on this later).

Refining Form and Behavior

- The Refinement phase is marked by the translation of the sketched storyboards to full-resolution screens depicting the user interface level.
- The basic process of design refinement follows the same steps we used to develop the design framework, this time at deeper and deeper levels of detail.
- After following Steps 2–6 at the view and pane levels, we use scenarios to motivate and address the more granular components of the product (e.g. dialogs and menus).

Refining Form and Behavior

- While the end product of the design process can be any one of a variety of outputs, we typically create a printed form and behavior specification.
- This document includes screen renderings sufficiently detailed for a programmer to code from, as well as detailed storyboards to illustrate behaviors over time.
- It can also be valuable to produce an interactive prototype or mockup of all major key path interactions.
- Prototypes alone are rarely sufficient to communicate underlying patterns, principles, and rationale, which are vital concepts to communicate to programmers.
- It requires constant attention to ensure that design vision is faithfully translated from the design document to a final product.

Design Validation and Usability Testing

- In the course of an interaction design project, it's useful go beyond your personas and validation scenarios to put solutions in front of actual users.
- This should be done once the solution is detailed enough, and with enough time to make design alterations based upon findings.
- It is a way to identify major problems with the interaction framework like problems with button labels or activity order.
- There are many ways to validate designs with users, from informal feedback sessions to see what the user thinks, or more rigorous usability testing where users complete a predetermined set of tasks.
- Usability testing is a means to evaluate, not to create – it is not an alternative to interaction design, and it will never be the source great ideas that make compelling products.

Summative versus Formative Testing

- In his 1993 book *Usability Engineering*, Jakob Nielsen distinguished between summative evaluations, which are tests of completed products, and formative evaluations, conducted during design as part of an iterative process.
- Summative evaluations are used in product comparisons, to identify problems prior to a redesign, and to investigate the causes of product returns and requests for training and support.
- Formative evaluations are quick, qualitative tests conducted during the design process – generally during the Refinement phase.
- A formative evaluation opens a window to the user's mind, allowing designers to see how their target audience responds to information and tools.
- Unlike summative evaluations, formative evaluations are conducted in the service of design, during the design process.

Conducting Formative Usability Tests

- There are a wide variety of perspectives on how to conduct and interpret usability tests.
- A good reference for usability testing methods is Carolyn Snyders *Paper Prototyping*.
- In brief, some essential components to successful formative usability tests are:
  - Test late enough in the process that there is a substantially concrete design to test, and early enough to allow adjustments in the design and implementation
  - Recruit participants from the target population (use personas)
  - Ask participants to perform explicitly defined tasks while thinking aloud
  - Have participants interact directly with a low-tech prototype
  - Moderate the sessions to identify issues and explore their causes
  - Minimize bias by using a moderator who has not previously been involved in the project
  - Focus on participant behaviors and their rationale
  - Involve designers throughout the study process

Principles and Patterns of Good Design

Interaction Design Principles

- Interaction design principles are generally applicable guidelines that address issues of behavior, form, and content.
- They encourage the design of product behaviors that support the needs and goals of users, and create positive experiences with the products we design.
- These principles are a set of rules based upon our values as designers and our experiences in trying to live up to those values.
- Design Principles are applied throughout the design process, helping us to translate tasks and requirements from scenarios into formalized structures and behaviors in the interface.

Interaction Design Principles

- Design principles operate at several levels of granularity, ranging from the general practice of interaction design down to the specifics of interface design.
- The lines between these categories are fuzzy, but interaction design principles can be generally grouped into the following categories:
  - Design values describe imperatives for the effective and ethical practice of design. These principles inform and motivate lower-level principles.
  - Conceptual principles help define what a product is and how it fits into the broad context of use required by its users.
  - Behavioral principles describe how a product should behave, in general, and in specific situations.
  - Interface-level principles describe effective strategies for the visual communication of behavior and information.

Behavioral and interface-level principles minimi

- One of the primary purposes principles serve is to optimize the experience of the user when engaging with a product.
- In the case of productivity tools and other non-entertainment-oriented products, this optimization means minimizing work.
- In this case, work means:
  - Cognitive work: Comprehension of product behaviors, as well as text and organizational structures
  - Memory work: Recall of product behaviors, command vectors, passwords, names and locations of data objects and controls, and other relationships between objects
  - Visual work: Figuring out where the eye should start on a screen, finding one object among many, decoding layouts, and differentiating among visually coded interface elements (such as list items with different colors)
  - Physical work: Keystrokes, mouse movements, gestures (click, drag, double-click), switching between input modes, and number of clicks required to navigate

Design Values

- Principles are rules that govern action, and are typically based at their core on a set of values and beliefs.
- The following set of values was developed by Robert Reimann, Hugh Dubberly, Kim Goodwin, David Fore, and Jonathan Korman to apply to any design discipline that aims to serve the needs of humans.
- As designers, we should create design solutions that are:
  - Ethical [considerate, helpful]: Do no harm, improve human situations.
  - Purposeful [useful, usable]: Help users achieve their goals and aspirations, accommodate user contexts and capacities
  - Pragmatic [viable, feasible]: Help commissioning organizations achieve their goals, accommodate business and technical requirements
  - Elegant [efficient, artful, affective]: Represent the simplest complete solution, possess internal coherence, appropriately accommodate and stimulate cognition and emotion

Design Values: Ethical Interaction Design

- Interaction designers are faced with ethical questions when they are asked to design a system that has fundamental effects on the lives of people.
- The products we design should do no harm, including:
  - Interpersonal harm (loss of dignity, insult, humiliation)
  - Psychological harm (confusion, discomfort, frustration, coercion, boredom)
  - Physical harm (pain, injury, deprivation, death, compromised safety)
  - Environmental harm (pollution, elimination of biodiversity)
  - Social and societal harm (exploitation, creation, or perpetuation of injustice)
- Avoiding the first two types of harm requires a deep understanding of the user audience.
- Avoiding physical harm means understanding ergonomic principles and appropriate use of interface elements so as to minimize work.

Design Values: Ethical Interaction Design

- Our designs should also improve human situations by:
  - Increasing understanding (individual, social, cultural)
  - Increasing efficiency/effectiveness of individuals and groups
  - Improving communication between individuals and groups
  - Reducing sociocultural tensions between individuals and groups
  - Improving equity (financial, social, legal)
  - Balancing cultural diversity with social cohesion
- As designers we should keep these broad issues at the back of our minds.
- Opportunities to do good should always be considered, even if they are slightly outside the box.

Design Values: Purposeful Interaction Design

- The primary theme of this part of the course is purposeful design based on an understanding of user goals and motivations.
- Part of purposefulness, however, is not only understanding user goals but also understanding their limitations.
- User research and personas serve well in this regard – the behavior patterns we observe and communicate should describe your users' strengths as well as their weaknesses.
- Interaction Design helps designers to create products that support users where they are weak and empower them where they are strong.

Design Values: Pragmatic Interaction Design

- Design specifications alone are of no use to anyone: a design must get built to be of value.
- It is critical that business goals and technical issues and requirements be taken into account in the course of design.
- There must be an active dialog among the business, engineering, and design groups about where there are firm boundaries.
- Programmers often state that a proposed design is impossible when what they mean is that it is impossible given the current schedule.
- Marketing organizations may create business plans based upon aggregated and statistical data without fully understanding how individual users likely to behave.
- Design works best when there is a relationship of mutual trust among Design, Business, and Engineering.

Design Values: Elegant Interaction Design

- Elegance is defined in the dictionary as both "gracefulness and restrained beauty of style" and "scientific precision, neatness, and simplicity."
- Elegance in interaction incorporates both of these ideals:
  - Represent the simplest complete solution. One of the classic elements of good design is economy of form: using less to accomplish more. In interface design, this means using only the screens and widgets necessary to accomplish the task. Less is more in good design, and designers should endeavor to solve design problems with the fewest additions of form and behavior.
  - Possess internal coherence. Good design has the feeling of a unified whole, in which all parts are in balance and harmony. Products that are poorly designed often look and feel like they are cobbled together. The Persona-Scenario-Requirement process, in which product concepts are conceived of as a whole and iteratively refined to detail, provides an ideal environment for creating internally coherent designs.

Interaction Design Patterns

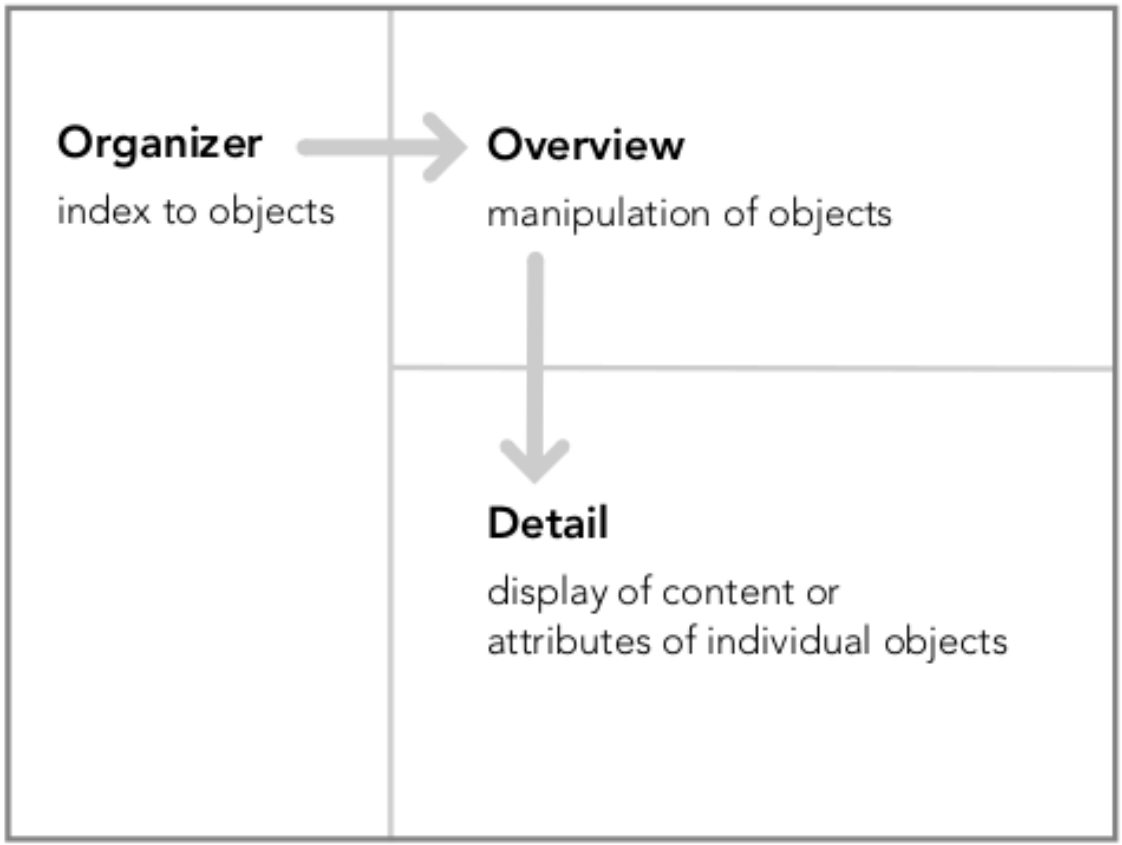
- Design patterns are a means of capturing useful design solutions and generalizing them to address similar problems.
- This effort to formalize design knowledge and record best practices can serve several vital purposes:
  - Reduce design time and effort on new projects
  - Improve the quality of design solutions
  - Facilitate communication between designers and programmers
  - Educate designers

Types of Interaction Design Patterns

- Like most other design patterns, interaction design patterns can be hierarchically organized from the system level down to the level of individual interface widgets.
- Like principles, they can be applied at different levels of organization:
  - Postural patterns can be applied at the conceptual level and help determine the overall product stance in relation to the user. An example of a postural pattern is "transient" which means that a person only uses it for brief periods of time service of a larger goal being achieved elsewhere.
  - Structural patterns solve problems that relate to the arrangement of information and functional elements on the screen. They consist of views, panes, and element groupings.
  - Behavioral patterns solve wide-ranging problems relating to specific interactions with functional or data elements. What most people think of as widget behaviors fall into this category.

Types of Interaction Design Patterns

- Building up a mental catalog of patterns is one of the most critical aspects of the education of an interaction designer.
- One of the most commonly used high-level structural patterns is apparent in Microsoft Outlook:



Platform and Posture: Desktop

Platform

- One of the the first questions to answer as we begin to design an interactive product is, "What platform and posture are appropriate?"
- The platform is the combination of hardware and software that enables the product to function (deployment and infrastructure).
- There has been an explosion in platform diversity: desktop, Web applications, kiosks, in-vehicle systems, handhelds (such as cameras, phones), home entertainment systems (game consoles, TV set-top boxes/tuners, and stereo/home theater systems), and professional devices (such as medical and scientific instruments).
- Platform is not an entirely well-defined concept: it is shorthand to describe a number of important product features, such as the physical form, display size and resolution, input methods, network connectivity, operating system.
- Choosing the right platform is a balancing act, where we must find the sweet spot that best supports the needs and context of your personas and fits within project constraints.

Posture

- A product's posture is its behavioral stance – the way it presents itself to users.
- This decision, too, must be based upon an understanding of likely usage contexts and environments.
- Most people have a predominant behavioral stance that fits their working role on the job: the soldier is wary and alert; the toll collector is bored and disinterested; the service representative is upbeat and helpful.
- A program too may be bold or timid, colorful or drab, but it should be so for a specific, goal-directed reason.
- Its manner shouldnt result from the personal preference of its designer or programmer.
- Programs whose appearance and behavior conflict with their purposes will seem jarring and inappropriate, like a clown at a wedding.

Posture

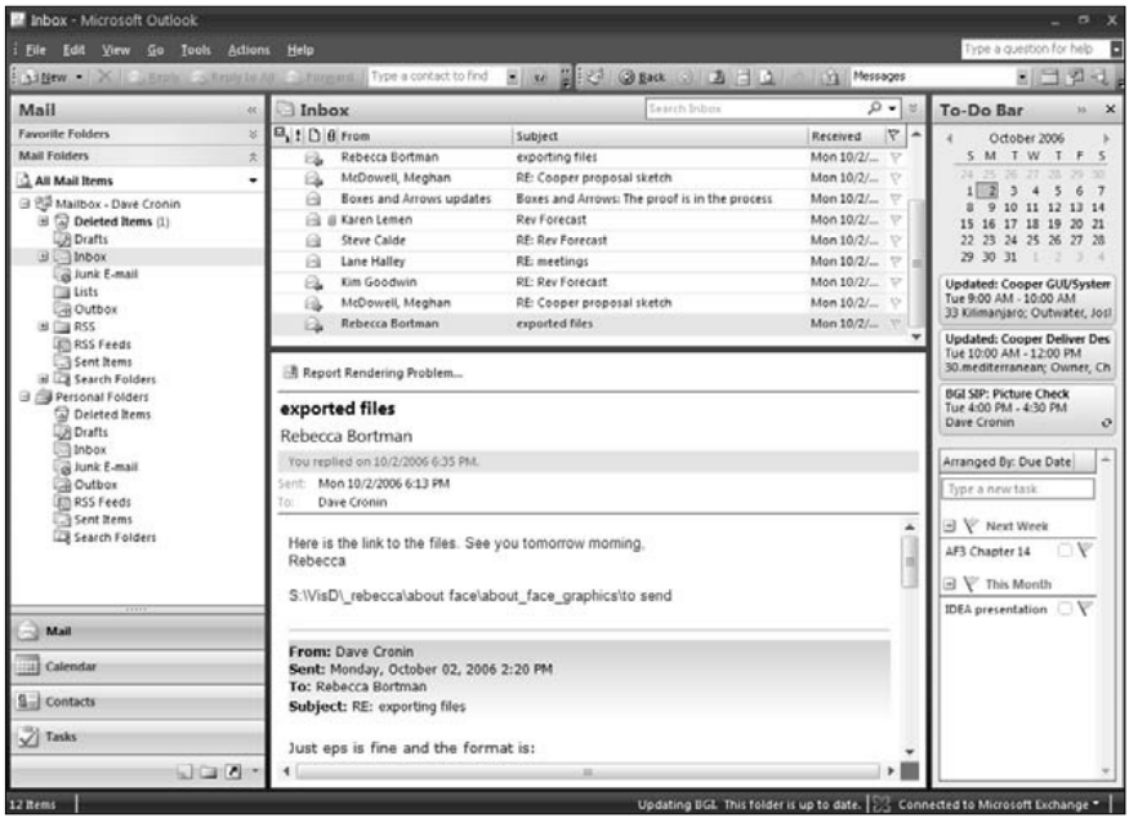
- The posture of your interface dictates many important guidelines for the rest of the design, but posture is not simply a black-and-white issue.
- Just as a person may present herself in a number of slightly different ways depending on the context, some products may exhibit characteristics of a number of different postures.
- When reading e-mail on a Blackberry during a train ride, a user may devote concentrated attention to interactions with the device
- The same user will have significantly less attention to devote if she is using it to look up an address while running to a meeting.
- Similarly, while a word processor should generally be optimized for concentrated, devoted, and frequent user attention, there are transient and infrequently used tools in it too.

Postures for Desktop Software

- Desktop software here refers to applications that run on a modern PC.
- When defining the platform of your product, clearly you must go beyond the term "desktop" to consider what the appropriate operating system, database, and user interface technology are for your product.
- In many organizations, platform decisions are unfortunately still made well in advance of the interaction designer's involvement.
- It is important to inform management that platform choices will be more effective if made after interaction designers complete their work.
- Desktop applications fit into four categories of posture: sovereign, transient, and daemonic.
- Because each describes a different set of behavioral attributes, each also describes a different type of user interaction.

Designing for Desktop: Sovereign Posture

- Programs that monopolize user attention for long periods of time are sovereign posture applications.
- Sovereign applications offer a large set of related functions and features, and users tend to keep them up and running continuously, occupying the full screen.
- Good examples of this type of application are word processors, spreadsheets, and e-mail applications.



Designing for Desktop: Sovereign Posture

- Some things to keep in mind:
  - Users of sovereign applications are typically intermediates. Because people typically devote time and attention to using sovereign applications, they often have a vested interest in getting up the learning curve to become intermediate users. From the designers point of view, this often means that the program should be optimized for use by perpetual intermediates and not be aimed primarily at beginners (or experts).
  - Be generous with screen real estate. Because a user's interaction with a sovereign application dominates his session at the computer, the application should take as much screen real estate as possible. Optimize sovereign applications for full-screen use.
  - Use a minimal visual style. Users will stare at a sovereign application for long periods. Keep the color palette narrow and conservative. Toolbars and auxiliary controls such as screen-splitters, rulers, and scrollbars can be smaller and more closely spaced than normal.

Designing for Desktop: Sovereign Posture

- Some more things to keep in mind:
  - Provide rich visual feedback. Sovereign applications are great platforms for creating an environment rich in visual feedback for users. The status bar at the bottom of the screen, the ends of the space normally occupied by scrollbars, and the title bar can be filled with visual status indications.
  - Provide rich input modalities. Sovereign applications similarly benefit from rich input. Every frequently used aspect of the application should be controllable in several ways. We can make more aggressive demands on fine motor skills with direct-manipulation idioms.

Designing for Desktop: Transient Posture

- A product with a transient posture comes and goes, presenting a single function with a constrained set of accompanying controls.
- The application is invoked when needed, appears, performs its job, and then quickly leaves, letting the user continue her normal activity (usually with a sovereign application)
- The defining characteristic of a transient application is its temporary nature.
- Due to it's transience, the user interface should be obvious, presenting its controls clearly and boldly with no possibility of confusion or mistakes.

Designing for Desktop: Transient Posture

- Things to keep in mind for transient applications:
  - Make things bright and clear. Atransient application must conserve the total amount of screen real estate it consumes, the controls on its surface can be proportionally larger than those on a sovereign application. Bolder graphics help the user to orient himself more quickly when the application pops up.
  - KISS: Keep It Simple, Stupid: After summoning a transient application, all needed information and facilities needs should on the surface of the application's single window. Keep the focus of attention on that window and never force him into supporting subwindows or dialog boxes to take care of the main function of the application.
  - Remember user choices. The best way to help users with both transient and sovereign apps is to give applications a memory. If a transient application remembers where it was the last time it was used, the chances are excellent that the same size and placement will be appropriate next time too.

Designing for Desktop: Daemonic Posture

- Programs that do not normally interact with the user are daemonic posture applications.
- These applications serve quietly and invisibly in the background, performing possibly vital tasks without the need for human intervention.
- A printer driver or network connection are excellent examples.
- Our discussion of daemonic posture is brief, due to the nature of these applications.
- Mostly we can think of applications of daemonic posture as transient applications that occasionally need our attention.

Summary

Summary

- Today we saw the next steps in turning designs into realized interfaces.
- We discussed how the design framework provides a skeletal structure for the objects and actions in our product.
- We then discussed general principles and patterns of interface design.
- Finally, we discussed general considerations for platform and posture.

Homework

Homework

Nothing until next lesson