MIDTERM REVIEW DECK

CST315 – Fall 2015 Revision

Arizona State University

Some notes from Sommerville 9th edition

OVERVIEW

- This slide deck is a composition of the slides that have been posted for the modules.
- The slides have been trimmed down according to the suggestions posted on BlackBoard.
  - I also cut out some of the general excess information - ten slides for interaction patterns is more than enough!
- Each module now includes an outline slide that explicitly lists the topics and techniques that are covered.
- There are now blue callouts that point out the specific topics shown on the hints from BlackBoard.
  - They are not comprehensive! They are just topics I *guarantee* are in the question pool.

- If you have any questions, please drop us an email.

MO SOFTWARE PROCESS

CST315 – Fall 2015 Revision

Arizona State University

Some notes from Sommerville 9th edition

OUTLINE (FOR EXAM)

- Process Basics
  - See UCD in next module.
- Incremental
- Iterative

- (We are skipping all the specific software processes.)

PRELIMINARY

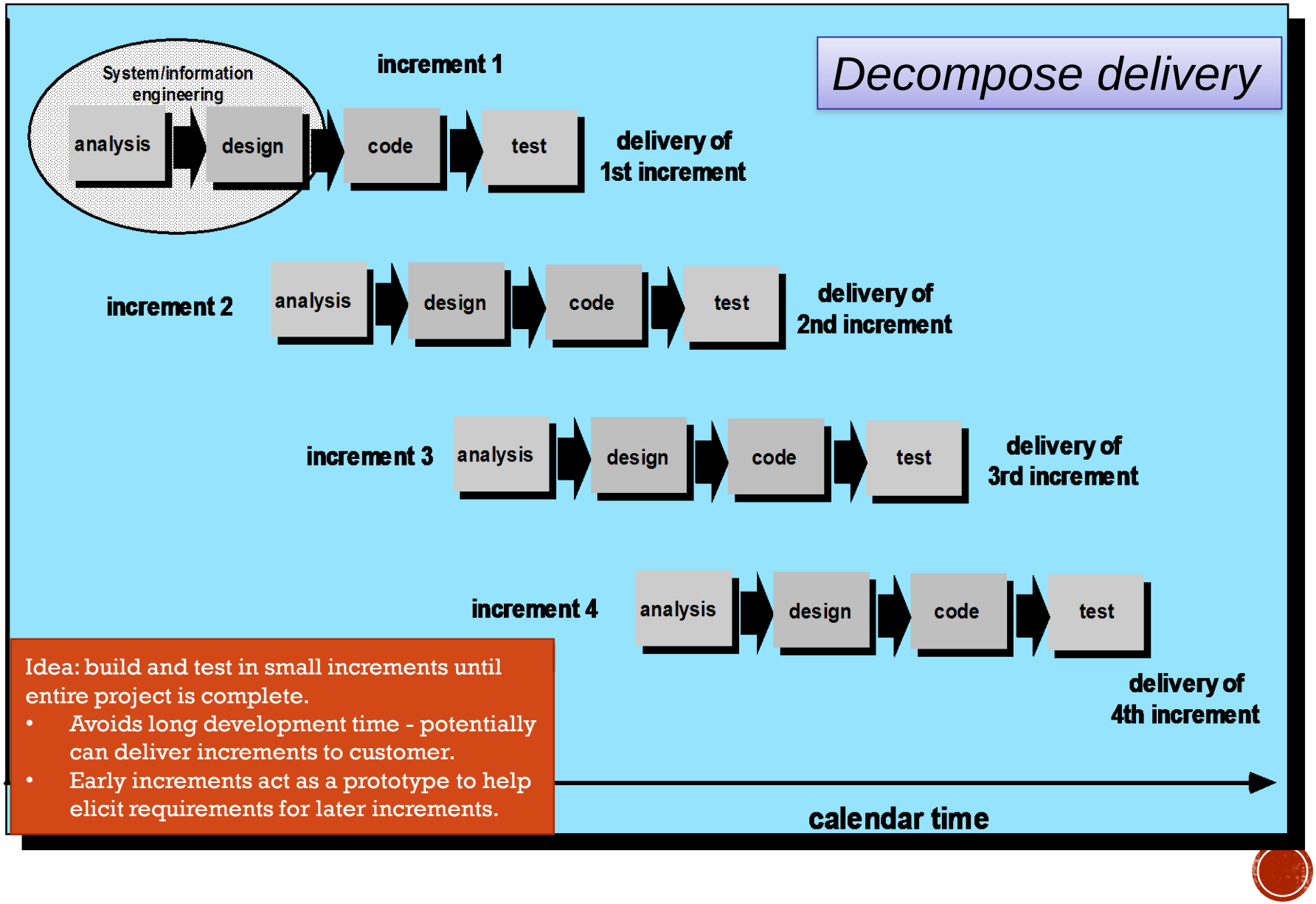
- Why a process?

- Repeatable: everything went right - how do we do it again?

- Permits Analysis: something went wrong, need to find the root of an issue.

- Informs Estimation: can better predict time or budget when adhering to a process.

PROCESS PRINCIPLE: INCREMENTS

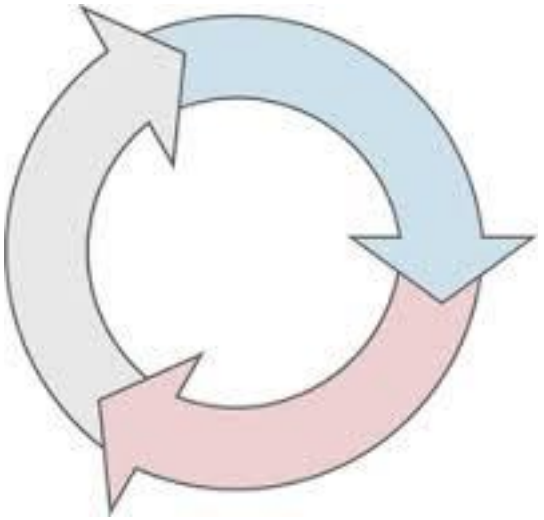


Idea: build and test in small increments until entire project is complete.

- Avoids long development time - potentially can deliver increments to customer.
- Early increments act as a prototype to help elicit requirements for later increments.

PROCESS PRINCIPLE: ITERATION

- Idea: Design, Develop (Prototype), and Test. Then, repeat, refining the project until it meets requirements.
- Assumption:
  - Requirements ALWAYS evolve in the course of a project so process iteration where earlier stages are reworked is always part of the process for large systems
- Iterative process models are considered the current "best practice" for software methodologies
- Approaches that incorporate iteration
  - Synchronize and Stabilize
  - Spiral development
  - RUP
  - XP



ITERATIVE & INCREMENTAL: PROS & CONS

- Pros:
  - Project broken into smaller, more manageable pieces
    - Easier to clarify requirements
    - Small teams may be used to address each increment
- Cons:
  - Poor system architecture
    - "narrow" prioritizing by customer
    - Each increment evolves independently
  - Localizing requirements creep to a particular increment
  - "Big-bang" integration
    - What happens when the increments don't line up?
- When to use:
  - When top-level requirements are properly partitioned
  - When a lot of outsourcing/COTS is used

M1 USER EXPERIENCE DESIGN PROCESSES IN SOFTWARE

CST315 – Fall 2015 Revision

Arizona State University

UX Design Principles

User-Centered Design

Techniques and Tools

Some notes from Sommerville 9th edition

OUTLINE (FOR EXAM)

- UCD Process
- Techniques:
  a) Task Analysis
  b) Paper Prototypes
  c) Wireframes / Storyboards
  d) Prototypes

Mentioned in hints – basically means you need to review all slides labeled UCD in this module.

(We are skipping two techniques: Conceptual Task Model and JAD)

WHAT IS USER CENTERED DESIGN?

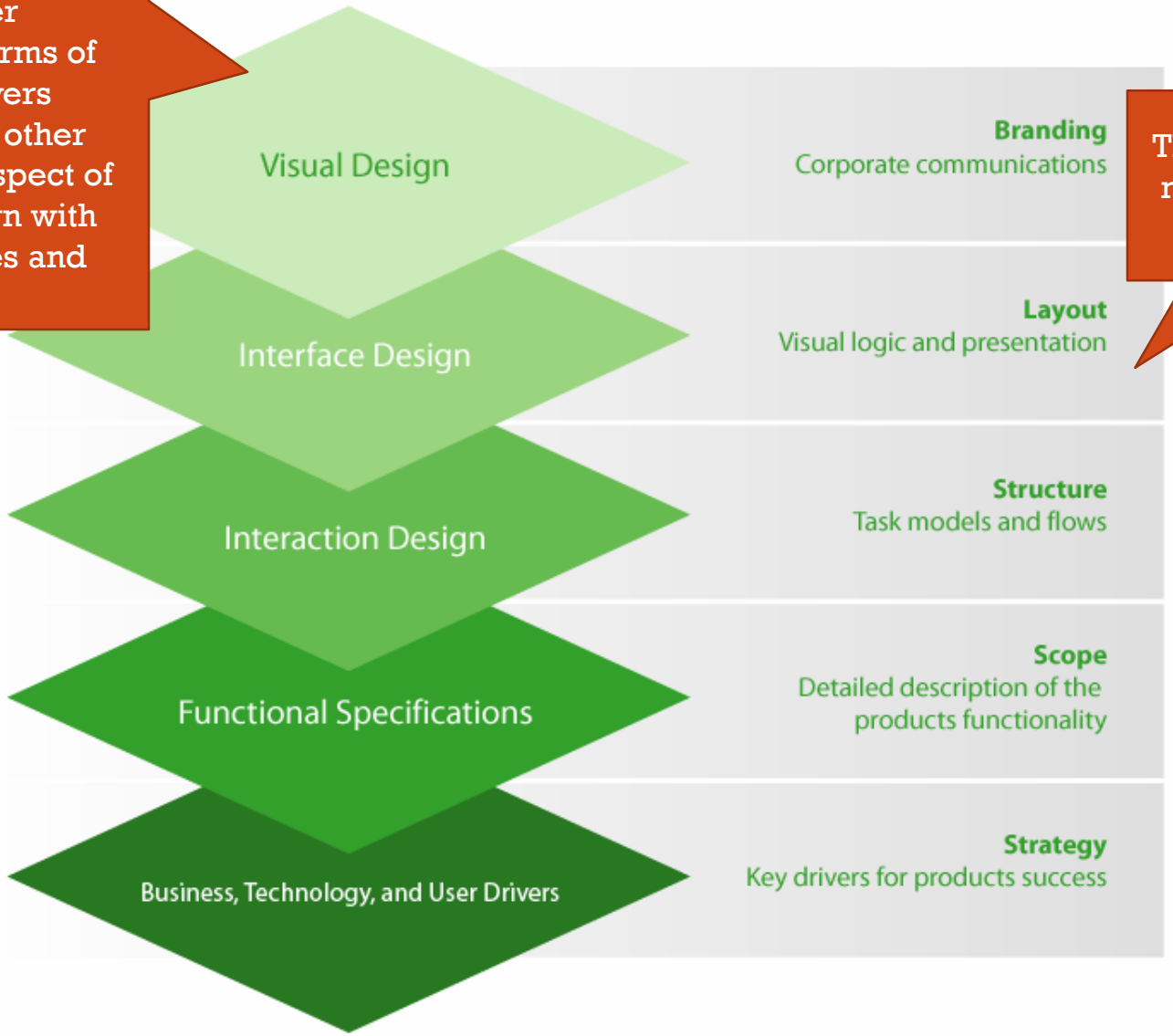
We will take a standards approach in this module – hence the preparation material.

We'll focus on four aspects:

PRINCIPLES Understand Target End-Users Prototype and Test Design with Target End-Users	PROCESS GUIDELINES ISO 13407 ISO 9241-11
STRATEGIC DESIGN Answer the "What" to Design	TACTICAL DESIGN Answer the "How" to Design It

ELEMENTS OF THE USER EXPERIENCE

We can think of a design underlying a user experience in terms of layers, where layers depend on each other and each is an aspect of the overall design with specific purposes and techniques.



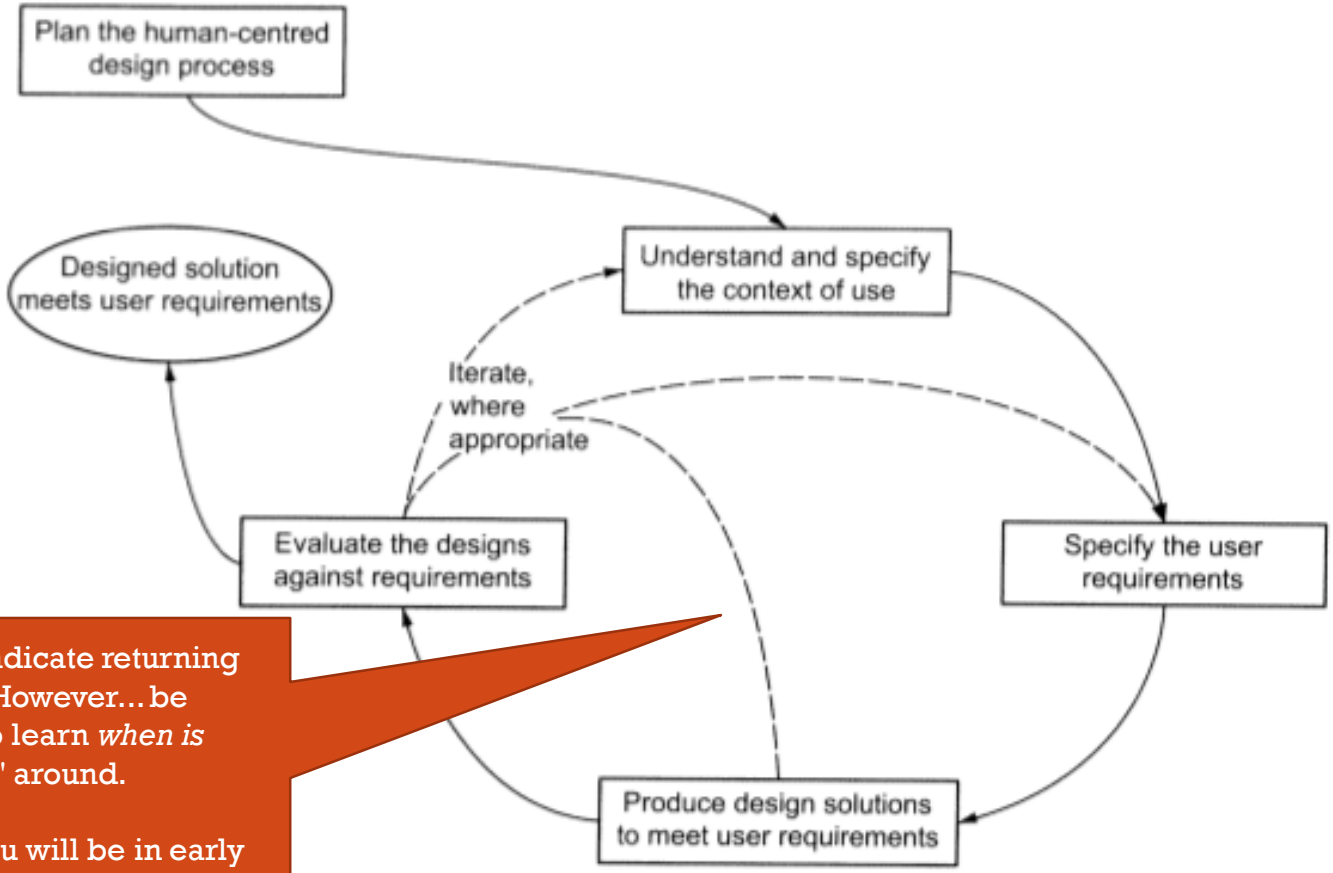
The top 3 will drive much of the next 2 weeks of class.

ISO 9241 DEFINITION OF USABILITY

- **Usability**: The extent to which a product can be used by specified users to achieve specified goals with *effectiveness*, *efficiency* and *satisfaction* in a specified *context of use*.
  - *Effectiveness*: the accuracy and completeness with which users achieve specified goals
  - *Efficiency*: the resources expended in relation to the accuracy and completeness with which users achieve goals
  - *Satisfaction*: freedom from discomfort, and positive attitude to the use of the product
  - *Context* of use: characteristics of the users, tasks and the organizational and physical environments
- ISO 9241 also defines a *Goal* as an intended outcome, and *Tasks* as activities to achieve a goal
  - Example (Jokela et al.): "90 % users achieve the goal (Es) in less than 1 minute (Ey) with an average satisfaction rating '6' (S) when users are novice ones (U), and they want to have a desired sum of cash withdrawn (G) with any bank machine (Cx)."

Es = effectiveness U = user
Ey = efficiency G = user goal
S = satisfaction Et = environment

ISO 13407 UCD PROCESS



The dashed lines indicate returning to a previous step. However… be careful! You need to learn *when is appropriate* to 'skip' around.

For your project, you will be in early iterations, where everything is still evolving, and will almost certainly need to execute all of the steps.

PLANNING IN UCD

1.
Plan

- UCD values information gleaned from collaboration:
  - What activities will you perform at steps 2-5?
  - When will you perform these activities?
  - Who will conduct the process?

Don't forget, we should also have some 'buy-in' from the team on the use of UCD.

- From the spec, the plan should identify:
  - a) the human-centred design process activities for steps 2-5;
  - b) procedures for integrating these activities with other system development activities, e.g. analysis, design, testing;
  - c) the individuals and the organization(s) responsible for the human-centred design activities and the range of skills and viewpoints they provide;
  - d) effective procedures for establishing feedback and communication on human-centred design activities as they affect other design activities, and methods for documenting these activities;
  - e) appropriate milestones for human-centred activities integrated into the overall design and development process;
  - f) suitable timescales to allow feedback, and possible design changes, to be incorporated into the project schedule.

UNDERSTAND THE CONTEXT OF USE

Context is very important – without it we could choose to do **anything**!

- The context of use description should
  - a) specify the range of intended users, tasks and environments in sufficient detail to support design activity;
  - b) be derived from suitable sources;
  - c) be confirmed by the users or if they are not available, by those representing their interests in the process;
  - d) be adequately documented;
  - e) be made available to the design team at appropriate times and in appropriate forms to support design activities.

- The specification suggests that the context of use document (wiki, knowledge base) should be a *working document* that is iterated upon as you gain more understanding.

DEFINE USER AND BUSINESS REQUIREMENTS

The specification's guidance on this step may not be useful.
User-centric requirements can be identified through a breakdown:

Users

- Who are they?
- What are their work environments like?
- How experienced are they with the technologies?
- What are their mental models and vocabulary?
- What are their personal characteristics?
- What are their cultural differences?
- What are their motivational differences?
- *Techniques we might use: Personas, Scenarios*

Goals

- The main thing users are trying to achieve
- *Techniques we might use: Interviews, Questionnaires, Reverse Engineer*

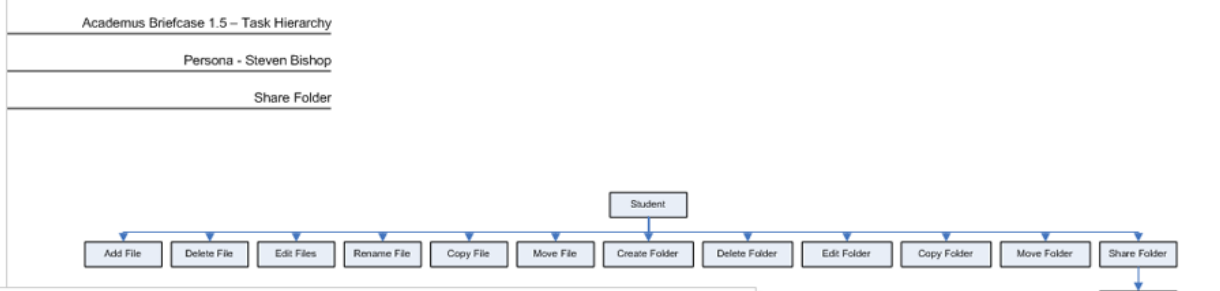
Tasks

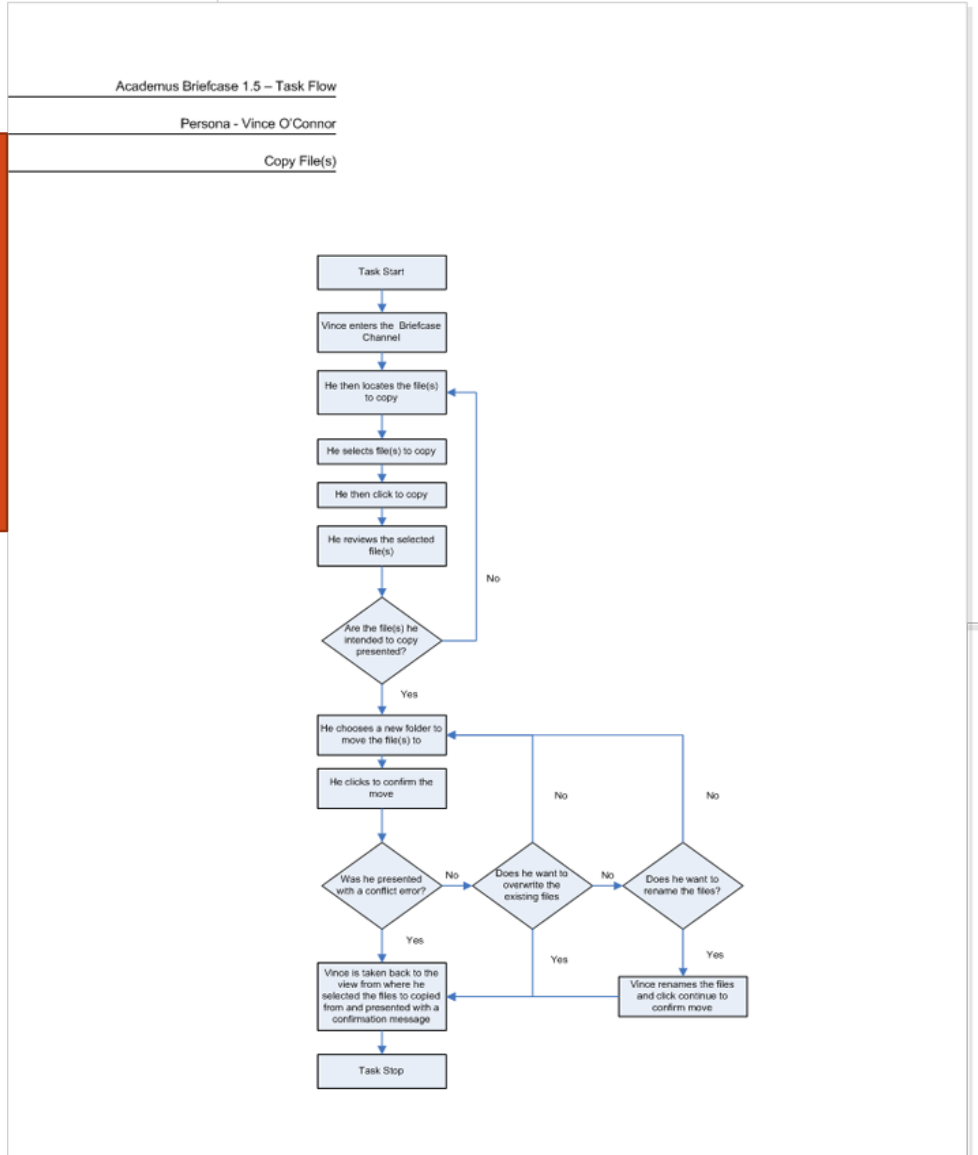
- How users do things to achieve their Goals
- Business process analysis, Job analysis, Task sequences,
- *Techniques we might use: Observation, Task hierarchies, Research*

EXAMPLE : TASK ANALYSIS

We can also breakdown a user's personal problem solving 'workflow' as a flowchart style diagram.

More on this later when we talk about Hierarchical Task Networks (HTNs).





3.

Specify User Requirements

PRODUCE DESIGN SOLUTIONS

4.
Produce Design Solutions

- From the spec, this step involves the following activities:

  a) *use existing knowledge* to develop design proposals with multi-disciplinary input;

  b) make the design solutions *more concrete* using simulations, models, mock-ups, etc.;

  c) present the design solutions to users and *allow them to perform tasks*

  d) alter the design in *response to the user feedback* and iterate this process until the human-centred design goals are met;

  e) manage the *iteration* of design solutions.

- *According to the spec, the benefits of prototyping (step b above):*

  a) make design decisions *more explicit* (this enables members of the design team to communicate with each other early in the process);

  b) allow designers to *explore several design concepts* before they settle on one;

  c) make it possible to *incorporate user feedback* into the design *early*

  d) make it possible to *evaluate* several iterations of a design & alternative designs

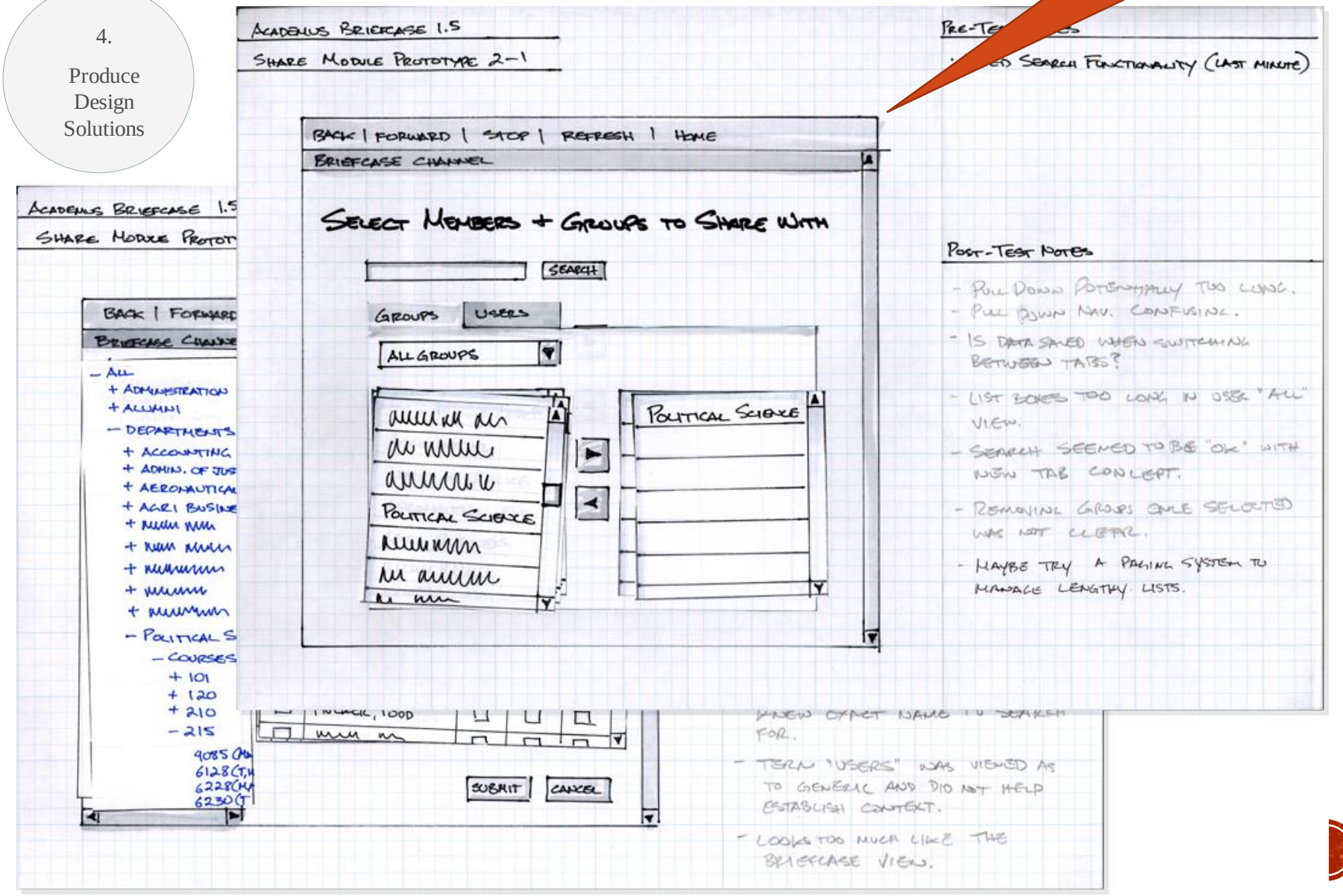
  e) *improve the quality and completeness* of the functional design specification.

TECHNIQUE: PAPER PROTOTYPES

Quickly sketching out a design on paper is the simplest way to build a prototype!

4.

Produce Design Solutions

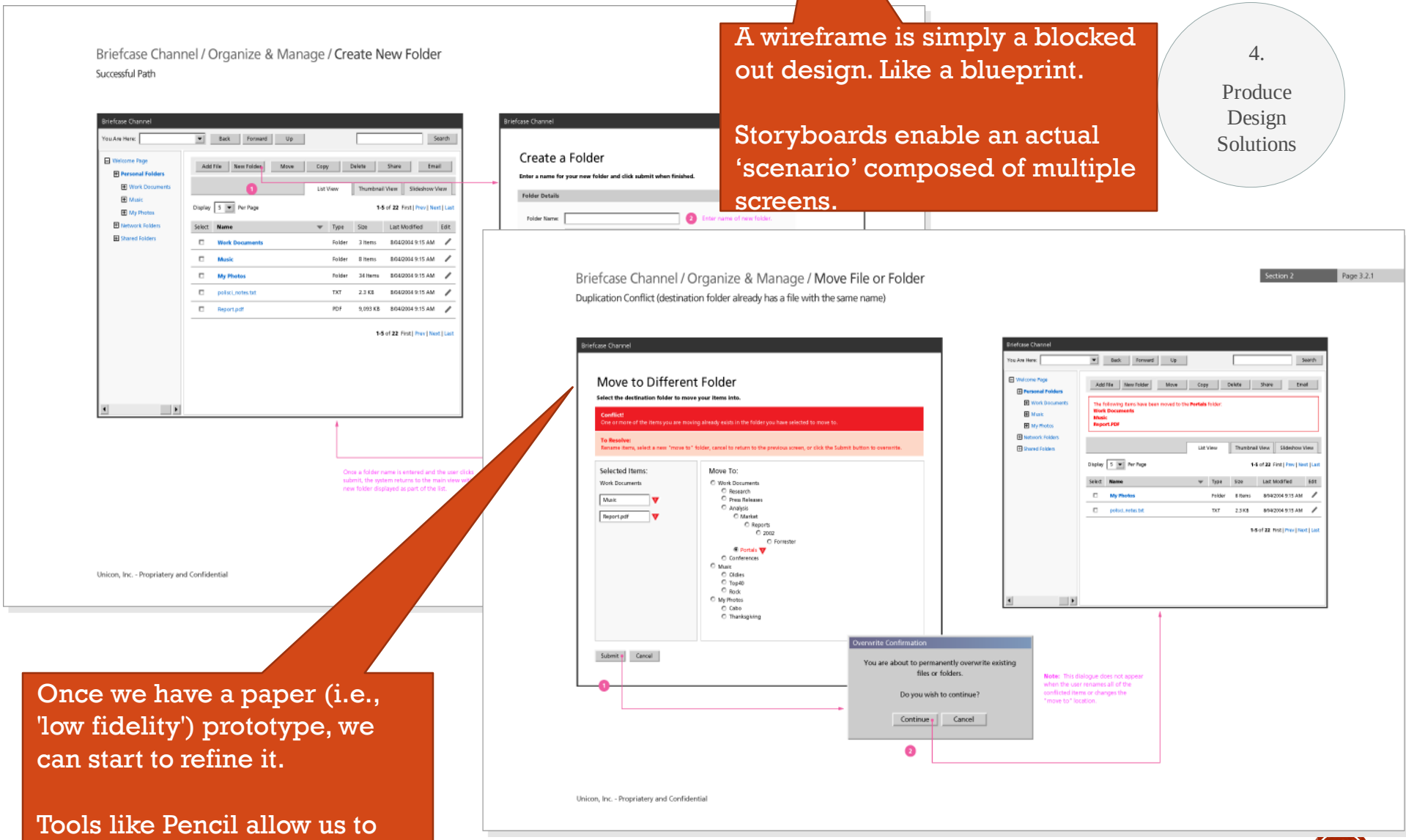


TECHNIQUE: WIREFRAMES / STORYBOARDS

A wireframe is simply a blocked out design. Like a blueprint.

Storyboards enable an actual 'scenario' composed of multiple screens.

4. Produce Design Solutions



Once we have a paper (i.e., 'low fidelity') prototype, we can start to refine it.

Tools like Pencil allow us to construct electronic prototypes with interactivity.

STORYBOARDING / PROTOTYPES

4.

Produce Design Solutions

- Modality:
  - 2-way
  - Participants are end users
  - Provides a structure for individual or group interaction
    - Storyboarding more conducive to small group interaction
- How-to:
  - Develop functionality based on vague requirements
    - Throw-away code!
  - Present to end user for direct feedback
  - Robustness of prototype needs only to be "sufficient to facilitate effective user feedback"
  - Technology base is chosen based on RAD, not based on the non-functional requirements

The code for your prototype is **not** intended to be used in the final product!

We will explore Storyboarding and Prototyping in detail in Interaction Design.

STORYBOARDING / PROTOTYPES

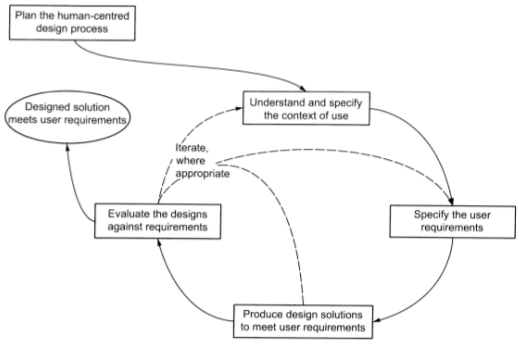
- Pros:
  - Making the solution "visible" provides you a precise means of agreeing on things with the user
  - May also facilitate your design and test cases
- Cons:
  - Cost to develop (need a RAD framework)
  - May pigeon-hole user into early requirements commitments
  - May pigeon-hole developers into early design commitments
  - Throw-away solution becomes a Big Ball Of Mud (BBOM)

DESIGN EVALUATION (OR, TEST PROTOTYPE)

- Evaluation (in any context), is only effective when there is a clearly defined goal for the evaluation
  - Consider two examples
    1. SER215/SER216: software testing classes last year
    2. How you are evaluated (graded)

5. Evaluate Designs

- But it is not just *summative evaluation* (go or no-go)
  - We use the process to find out more about our design ideas
  - From this perspective, we may elicit *formative* feedback
    - We aren't ready early on to know if we are "right"
    - We are still gathering information
    - So we may demonstrate several design ideas in one cycle
    - As we prune out bad ideas, we try to then "get it right"



M2 INTRODUCTION TO USER MODELING

CST315 – Fall 2015 Revision

Arizona State University

Personas

Use Case Actors

Observation / Ethnography

OUTLINE (FOR EXAM)

Why modeling?

Techniques:

a) Personas
b) Observation

(We are skipping one technique: Use Case Actors.)

WHY MODELING?

- Provides a layer of abstraction that aims to capture generality.
- Provides a standardized/documented method to represent user behaviors and goals.

- Works toward the UCD ideal while minimizing physical user involvement until prototype stage.

USER MODELING

These are the types of questions you'll want to consider as you build personas.

2. Understand Context of Use

Users

- Who are they?
- What are their work environments like?
- How experienced are they with the technologies?
- What are their mental models and vocabulary?
- What are their personal characteristics?
- What are their cultural differences?
- What are their motivational differences?
- *Techniques we might use: Personas, Scenarios*

Goals

- The main thing users are trying to achieve
- *Techniques we might use: Interviews, Questionnaires, Use Case Actors, Reverse Engineer (e.g., logs)*

Tasks

- How users do things to achieve their Goals
- Represent using Business process analysis, Job analysis, Task sequences
- *Techniques we might use: Observation, Task hierarchies, Research*

USER RESEARCH / PERSONAS

These aren't quite personas… there aren't any goals.

Clark Richards
University of Califo
VP Academic / Stu

Overview
Clark is a senior business leader who found himself in the academic market after successfully starting and selling a high-end copier / fax mail order business.

With a background in operations, Clark oversees the entire range of campus and student services.

He works very closely with the CIO who is both a colleague and good friend. Clark is reasonably knowledgeable about high-level technology issues but rarely takes a hands-on role in making technical recommendations.

The thing that drives Clark is quality education, prestige, and making sure that the academics, students, and administration are given the tools they need to succeed. He's very much a perfectionist.

Technical Profile
Moderate Computer User
T1 at work, no connection at home
PC at work only
17" LCD monitor
Mainly uses MSIE

Demographics
56 Years Old
Male
Caucasian
Divorced
3 Kids (All girls – 2
$130,000 Annual $

Psychographics
At the peak of his p
Very sophisticated
Has a healthy inte
show-boat with his
Enjoys fine dining,
fishing
Works roughly bet

Juanita Flores
Mesa Community College
Political Science Instructor (Tenured)

Overview
Juanita is considered to have a strong and diplomatic personality by her colleagues. Though she is not totally familiar with Course Management solutions, she would make a good candidate for heading up the research and consensus building amongst her peers.

She is very well organized and takes pride in balancing her career goals with her family goals.

Technical Profile
Moderate Computer User (Still uses an offline PIM)
T1 at work, 256k DSL at home
PC at work, Mac at home
17" LCD monitor
Uses AOL at home

Demographics
45 Years Old
Female
Hispanic
Married
2 Kids (Boy 16 / Girl 14)
$65,000 Annual Salary

Psychographics
Family oriented
Involved heavily with her kids education
Member of the local PTA
A true working mother – comes home and makes dinner for the family
Has very little time to herself
Late night activities may include grading homework, reading, research, and yoga

In many domains, understanding the user's technical skillset is very important.

DESIGNING A PERSONA

- Personas are *goal-directed*

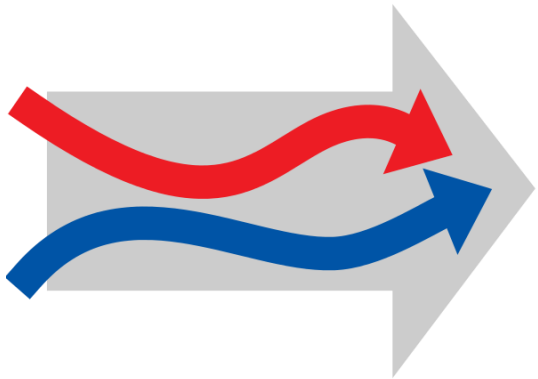
  "A precise description of our user and what he wants to accomplish"

  - Personas are only useful in a purposeful context
  - Goals are not tasks; it is not about what the user wants to do, it is about what s/he wants to achieve

- Personas are *not real people*, they represent them
  - Life-like description motivates the designer
    - Forces the designer to have an individual in mind, not some vague "end user"
  - Not "made up" but *discovered*
    - This means you have to conduct research to come up with them
    - There is no "recipe"; no definitive process, only guidelines

Use the questions from earlier.

PERSONAS: GUIDELINES



- *A "how-to"?* Not really!
  - it isn't a prescriptive process
- Useful Techniques
  - Be *goal-directed*; make sure the goals and objectives of the persona are clear
  - *Ethnography* – "observe in their habitat" *(stay tuned…)*
  - *Contextual inquiry* – like ethnography but more active; e.g. you may collaborate with the end user on a task
  - *Independent* verus *specific to the domain* – unlike "business objects" in UML, a persona is specific to the design problem at hand, and they do not overlap
  - Your reading recommends starting with *3-12 personas*
    - Primary, secondary, supplemental, served, and negative
    - At least *one primary*, but not too many either

PERSONAS: GOAL TYPES

1. Personal – simple, universal, and personal
   - e.g. *"complete 3 sales follow-up calls today"*
2. Corporate – Goals of the organization w.r.t. the system transferred to the end user
   - e.g. "*increase online sales by 20%*"
   - These are the ones we usually think of as *requirements*
3. Practical – bridge the objectives of the organization with the objectives of the individual
   - i.e. bridge between 1 & 2 above
   - e.g. *"if we meet our sales target I may get a bonus"*
4. False – Not relevant to the end user
   - e.g. "user-friendly", "use responsive web design", etc.
   - Represent design or implementation decisions, not end user needs

Mentioned in hints!

You don't always need a picture for your personas but they help to make it more concrete.

PERSONA: EXAMPLE

Figure 1: The team created detailed descriptions of each persona, including a stock photo and a biography. The text to the right represents a summary of Rhonda's persona description (each description was about a full page in the actual deliverable).

Persona Example: Rhonda Wilson, RN
Nurse Unit Coordinator

Rhonda is a 36-year-old RN who has worked at several skilled nursing facilities. She started out in acute care but moved to long-term care so she could have more autonomy. Rhonda was promoted to Unit Coordinator four years ago because she is very competent and generally well organized.

Rhonda is entirely overwhelmed and is drowning in paper, even more so than the average nurse. She often misses eating dinner with her boyfriend because she has to work late, filling out forms and reports.

Rhonda's goals are to:

- **Spend time on patient care and staff supervision, not paperwork.**
- **Be proactive.** Rhonda needs to understand trends in order to solve problems before they happen, instead of just reacting to crises.
- **Know that things are being done right.** Rhonda supervises the unit because she's good at what she does. If nurses aren't following procedure or documenting things, she wants to know right away.

From Calde et al. 2002

DETERMINING TASKS: OBSERVATION

- Also called "ethnography"
- Modality:
  - 1-way communication
  - Real-time vs. video capture vs. event capture (log)
  - Staged environment versus real environment
- How-to:
  - Determine modality – will you observe live or capture via video or some other technology
  - Review organizational & regulatory policies, NDAs, etc.
  - Prepare a debriefing memo
  - Determine a recording format and method
  - Embed into environment with minimal intrusion

Must be careful not to change the context of use too much.

OBSERVATION

- Pros:
  - Observing how customer works allows you to see how the technology benefits.
  - Removes the 0^{th}-order interpreter – the user
- Cons:
  - Time-consuming
  - Observee will not behave "naturally" (Hawthorne effect)
  - Disruption to the workplace
- To Note:
  - Ethnography is a well-known elicitation technique in research circles, and may be suitable for inception
  - Requirements elicitation using ethnography is often too time-consuming, too disruptive – simply too awkward

OBSERVATION

- To Note (cont):
  - A variant on embedded real-time observation is a staged observation for HCI evaluation
    - Observe a user interacting with the system
      - Video
      - Event tracking (mouse clicks, screen visit sequence, etc.)
    - Not really "ethnography"
    - Part of "User-centered design" when designing a HCI
  - Another variant on Observation is *Apprenticing*
    - The users train the investigator on how to perform the job
    - Investigator then performs in that role for some time to learn first-hand the issues for end users
    - Time-consuming but very effective

SUMMARY

- You want to be able to write down *Users*, *Goals*, and *Tasks*
  - Understanding who your users are, and giving them a name or identity, defines *scope* and *accountability* in your approach
  - *Goals* are forms of *requirements*; it tells you what your users want to accomplish
  - Humans *interact* with software products
    - They could receive *information* (we will discuss information modeling later)
    - Usually they *interact* with the system (do something) to achieve a Goal

M3 INTERACTION DESIGN (IxD)

CST315 – Fall 2015 Revision

Arizona State University

Some notes from Nathan Pearson and Sommerville 9th edition.

OUTLINE (FOR EXAM)

- IxD Dimensions
- IxD Principles
- IxD Process (maybe can skip)
- Techniques:
  a) Scenarios
  b) Hierarchical Task Networks (maybe can skip)
  c) IxD Patterns
  d) IxD Best Practices
  e) IxD Bad Practices

DIMENSIONS TO IxD

We can think of IxD as having five independent factors, or, dimensions. A dimension is a property of design that we should consider.

1D Words

From Smith & Silver (via Wikipedia).

- Dimension **defines the interactions**. Words embody *semantics* that users use to interact.

2D Visual Representations

- The visual representations are the **things that the user interacts with** in the interface. These may include but not limited to "typography, diagrams, icons, and other graphics"

3D Physical objects or space

- The **space with which the user interacts** is the third dimension of interaction design. It defines the space or objects "with which or within which users interact with"

4D Time

- The **time with which the user interacts with the interface**. Some examples of this are "content that changes over time such as sound, video, or animation"

5D Behavior

- Defines the **users actions reaction** to the interface and **how they respond to it**.
- (This is essential; the IxD must create an expectation of how to interact, yet be flexible enough to account for mis-interaction.)

IXD PRINCIPLES

- Know your Users! (review User Modeling)
- Factors from the Textbook:
  - People have limited short-term memory (Miller: +/- 7 items)
  - Tolerance – accounting for and anticipating human error
  - Physical manipulation limitations
    - There is also a field that emerged based on cognitive aspects, or information processing *modalities* (audio vs. written vs. video learners)
  - Individualized interaction preferences (mouse/icon vs. hotkey)
- IxD Principles
  - *User familiarity* – based on terms & concepts from users' experience
  - *Consistency* – perform similar operations similarly
  - *Minimal Surprise* - do not surprise users w/ behavior of the system
  - *Recoverability* – allow users to recover from errors
  - *User Guidance* – feedback and context-sensitive help (hmm…)
  - *User Diversity* – appropriate interactions for different types of users

Mentioned in hints!

PROCESS: IxD PROCESS

This diagram comes straight from your text. As an exercise, try identifying the four 'regions' in the diagram and find which align them with the UCD steps.

- How does this process compare to the UCD?

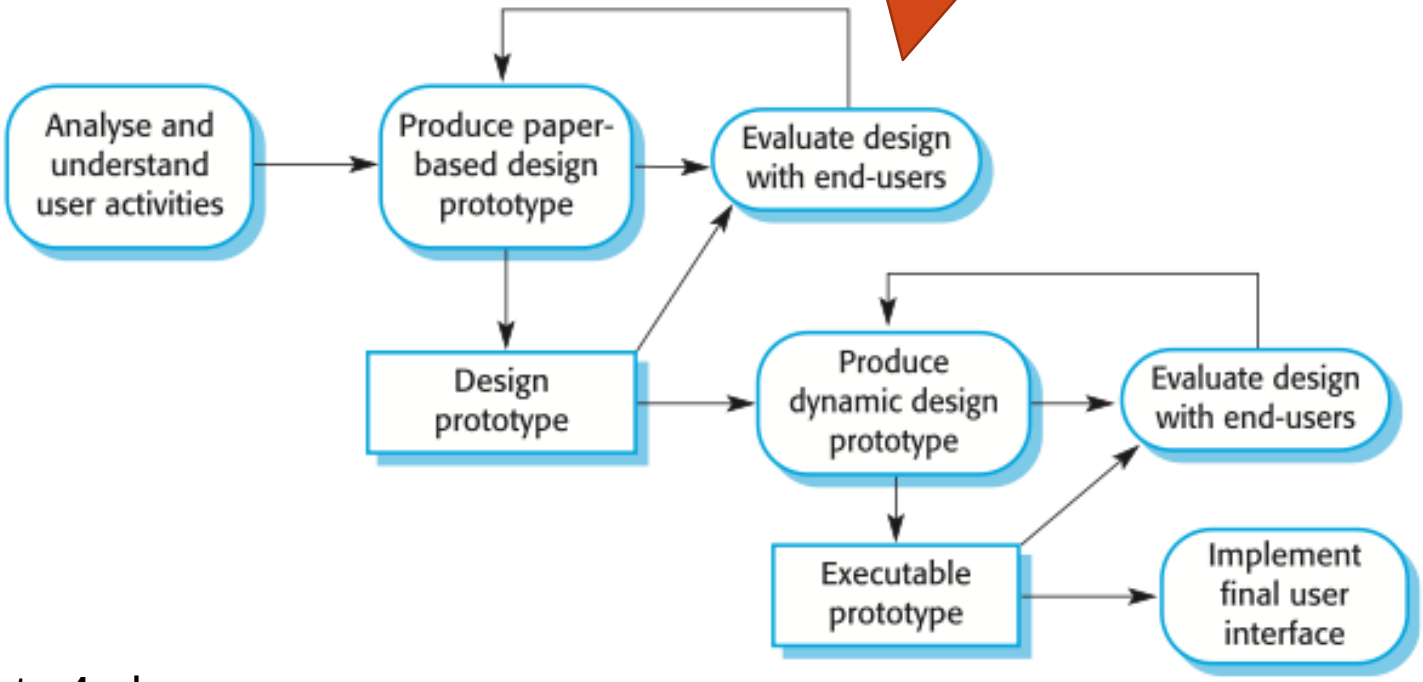


Figure 29.13
The interaction design process

The diagram suggests 4 phases:

1. Understand user activities
2. Produce design prototype (low-fidelity)
3. Produce executable prototype (high-fidelity)
4. Implement

We will now introduce two techniques (scenarios and HTNs), for understanding user tasks.

UNDERSTANDING USER TASKS: SCENARIOS

- If you don't understand *what the users want to do with a system*, you have no realistic prospect of designing an effective interface.
- *Scenarios* where you describe typical *episodes of use*, are one way of describing these analyses.
  - User analyses have to be described in terms that users and other designers can understand.
  - Complements, doesn't replace, our use of personas earlier.

Try identifying possible personas in this scenario.

Example Scenario:

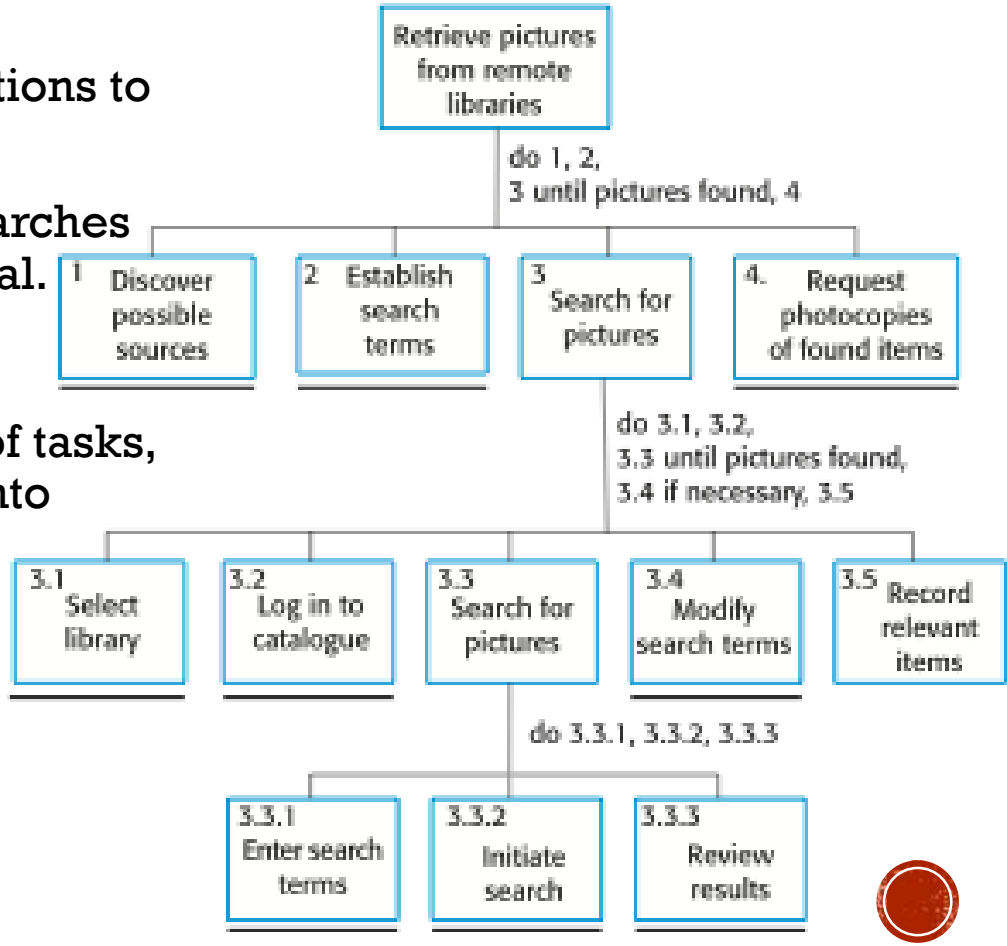
Jane is a student of Religious Studies and is working on an essay on Indian architecture and how it has been influenced by religious practices. To help her understand this, she would like to access some pictures of details on notable buildings but can't find anything in her local library.

She approaches the subject librarian to discuss her needs and he suggests some search terms that might be used. He also suggests some libraries in New Delhi and London that might have this material so they log on to the library catalogues and do some searching using these terms. They find some source material and place a request for photocopies of the pictures with architectural detail to be posted directly to Jane.

UNDERSTANDING USER TASKS: HTNs

Breaking down the scenario:

- Users may not be aware of appropriate search terms so need a way of helping them choose terms.
- Users have to be able to select collections to search.
- Users need to be able to carry out searches and request copies of relevant material.



One way to express this is a hierarchy of tasks, that shows how a task is decomposed into smaller, and more specific, sub-tasks.

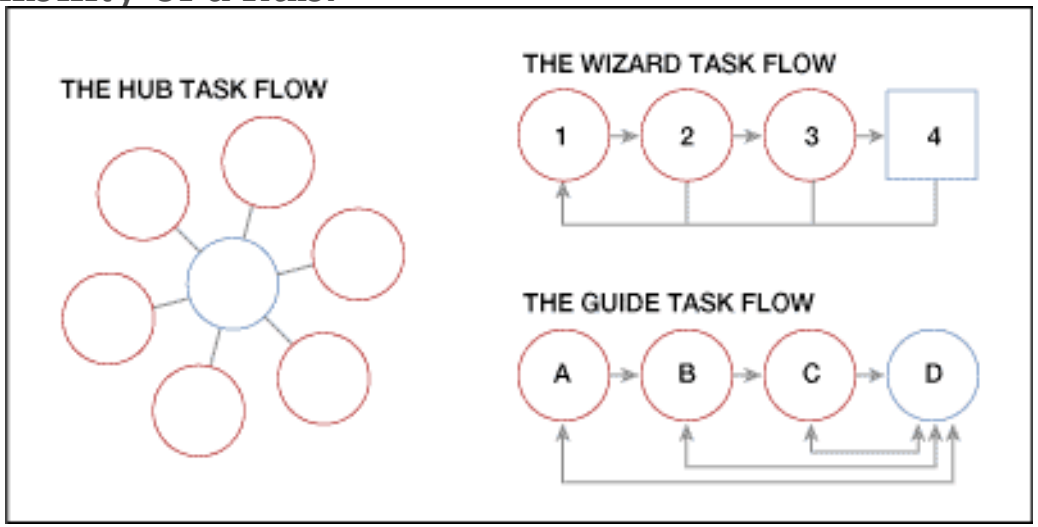
Hierarchical Task Networks (HTNs) can be used to establish both a **decomposition** structure and a sequence of alternatives structure.

Mentioned in hints!

IxD PATTERNS SUMMARY

- **Hubs, Wizards, and Guides**

- **Hubs:** You go, you come back. Hubs are ideal for situations that use multiple, discreet, single-page forms.
- **Wizards:** Step one, two, three. Wizards are appropriate for multi-page procedures or operations that must be completed in a prescribed order.
- **Guides:** This way please! Guides are useful for complex, multi-part sequences that seek to combine the navigational guidance of a wizard with the navigational flexibility of a hub.



IxD BEST PRACTICES

We'll cover these four techniques over the next several slides.

- **Miscellaneous Interaction Techniques**
  - Progressive Disclosure
  - Use of Metaphors
  - Modeling Interaction Styles
  - Using Modalities

- **Progressive Disclosure**
  - Basic idea: Move complex and less frequently used options out of the main user interface and into secondary screens.

- Benefits
  - Prevents user from getting overwhelmed by the UI.
  - Allows users to recognize what to do as they go along.
  - Increases ease of use of the product and reduces chance for user error.

- Risks
  - Repeat user may not require progressive disclosure (depends on the task).
    - Assumes the designer understands the most popular, common or important task for the user.

IxD BEST PRACTICES

It is easy for a user to get the sense that an icon of a trashcan has to do with deleting/removing something.

- **UI Metaphors**
- UI Metaphors are product oriented conceptual models that map over to user oriented mental models. They embed common semantics that we can leverage.
- Examples of Metaphors:
  - Inbox
  - Typewriter
  - Canvas, brushes, and pallet
  - Desktop
  - Shopping cart
  - Etc.

IxD BEST PRACTICES: MODELING INTERACTION STYLES

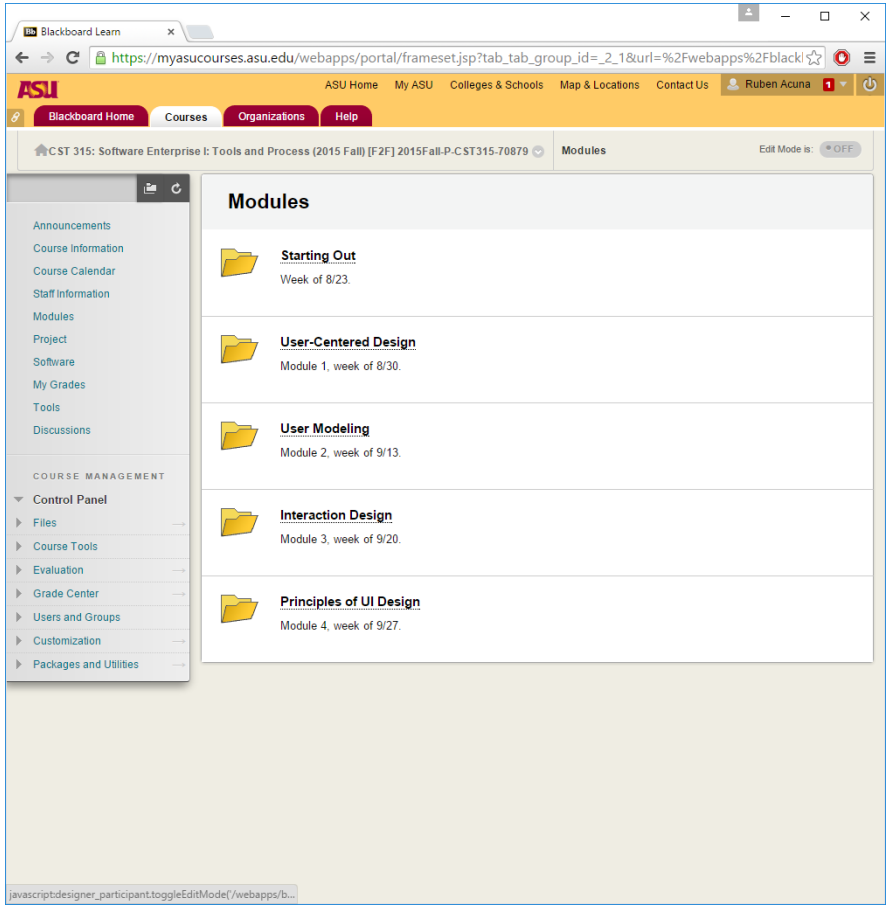
This table is from your textbook – you should already be familiar with it.

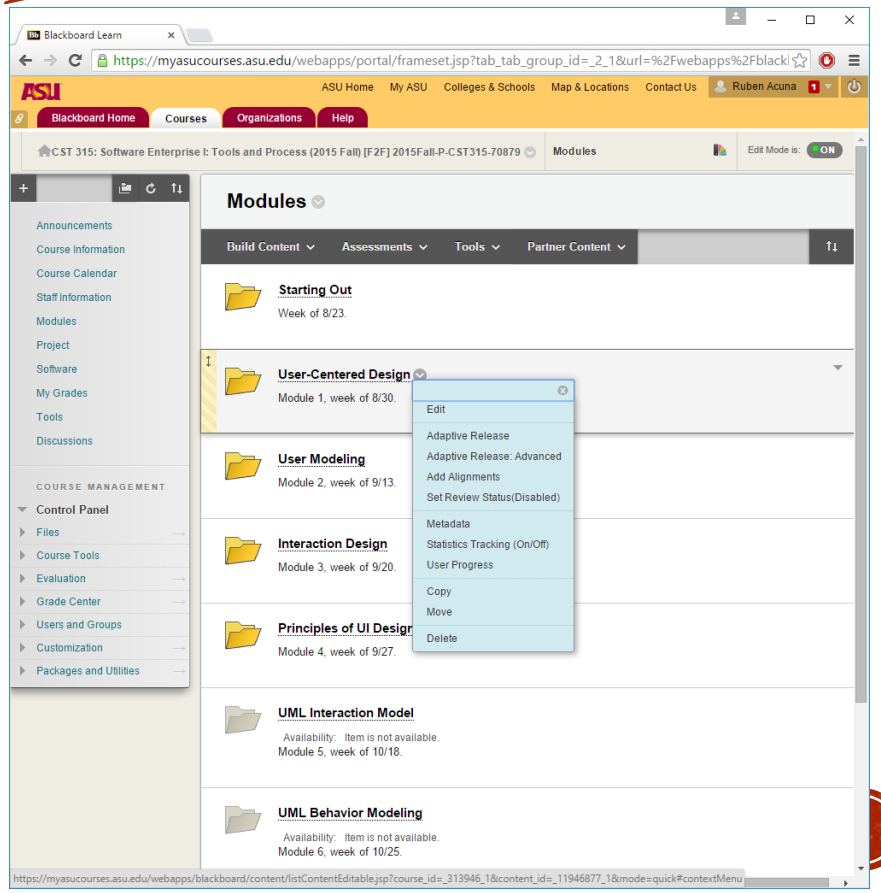
Interaction style	Main advantages	Main disadvantages	Application examples
Direct manipulation	Fast and intuitive interaction Easy to learn	May be hard to implement. Only suitable where there is a visual metaphor for tasks and objects.	Video games CAD systems
Menu selection	Avoids user error Little typing required	Slow for experienced users. Can become complex if many menu options.	Most general-purpose systems
Form fill-in	Simple data entry Easy to learn Checkable	Takes up a lot of screen space. Causes problems where user options do not match the form fields.	Stock control, Personal loan processing
Command language	Powerful and flexible	Hard to learn. Poor error management.	Operating systems, Command and control systems
Natural language	Accessible to casual users Easily extended	Requires more typing. Natural language understanding systems are unreliable.	Information retrieval systems

IxD BEST PRACTICES

- **Modalities – modes of interaction.**
  - A View mode = no changes to stored data.
  - Edit mode = changes to stored data.

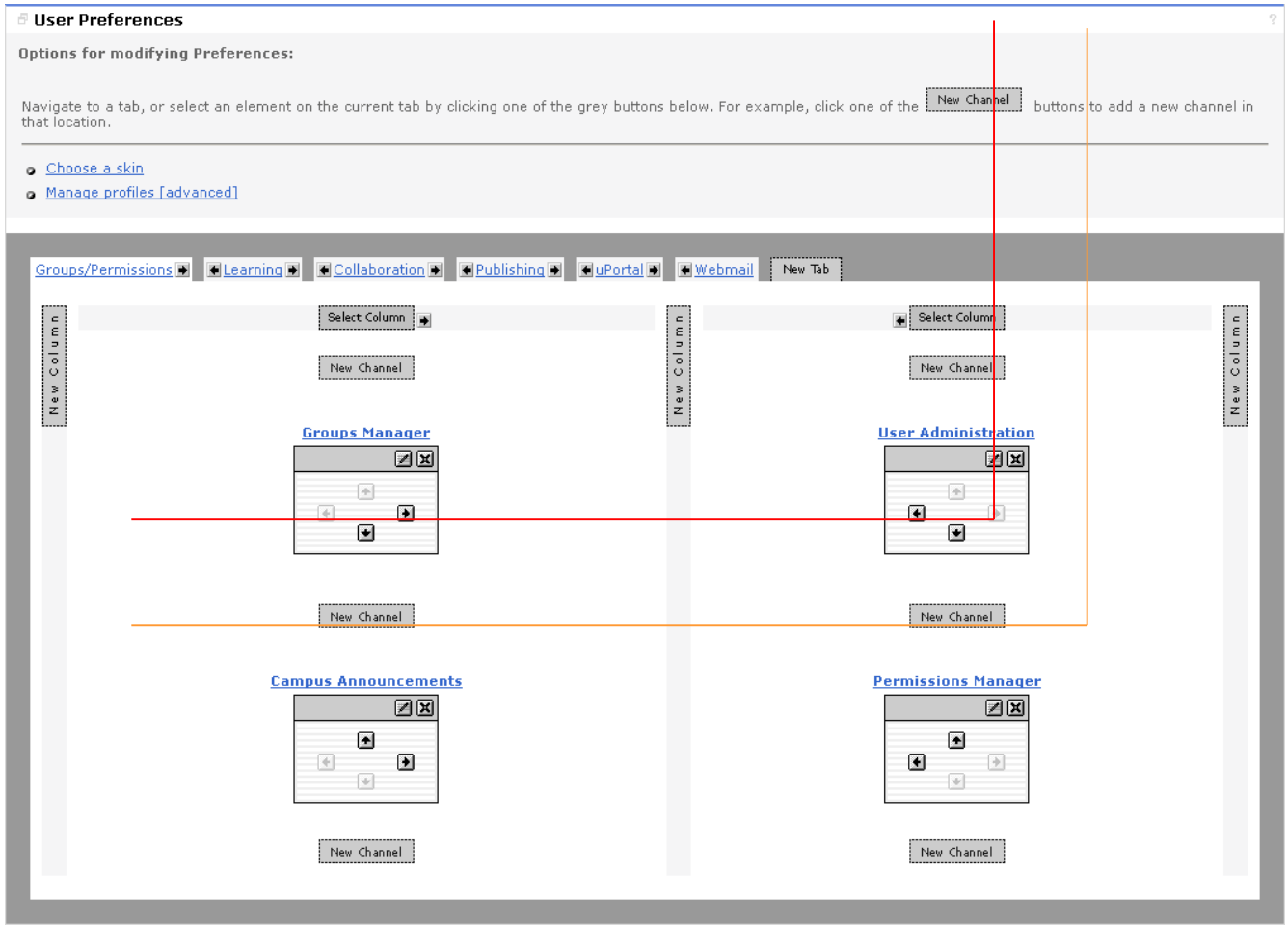
BlackBoard has two basic modes: a view mode for students, and a edit mode for teachers. Based on the mode different functionality is available (e.g., editing the name of a folder).





IxD BAD PRACTICES

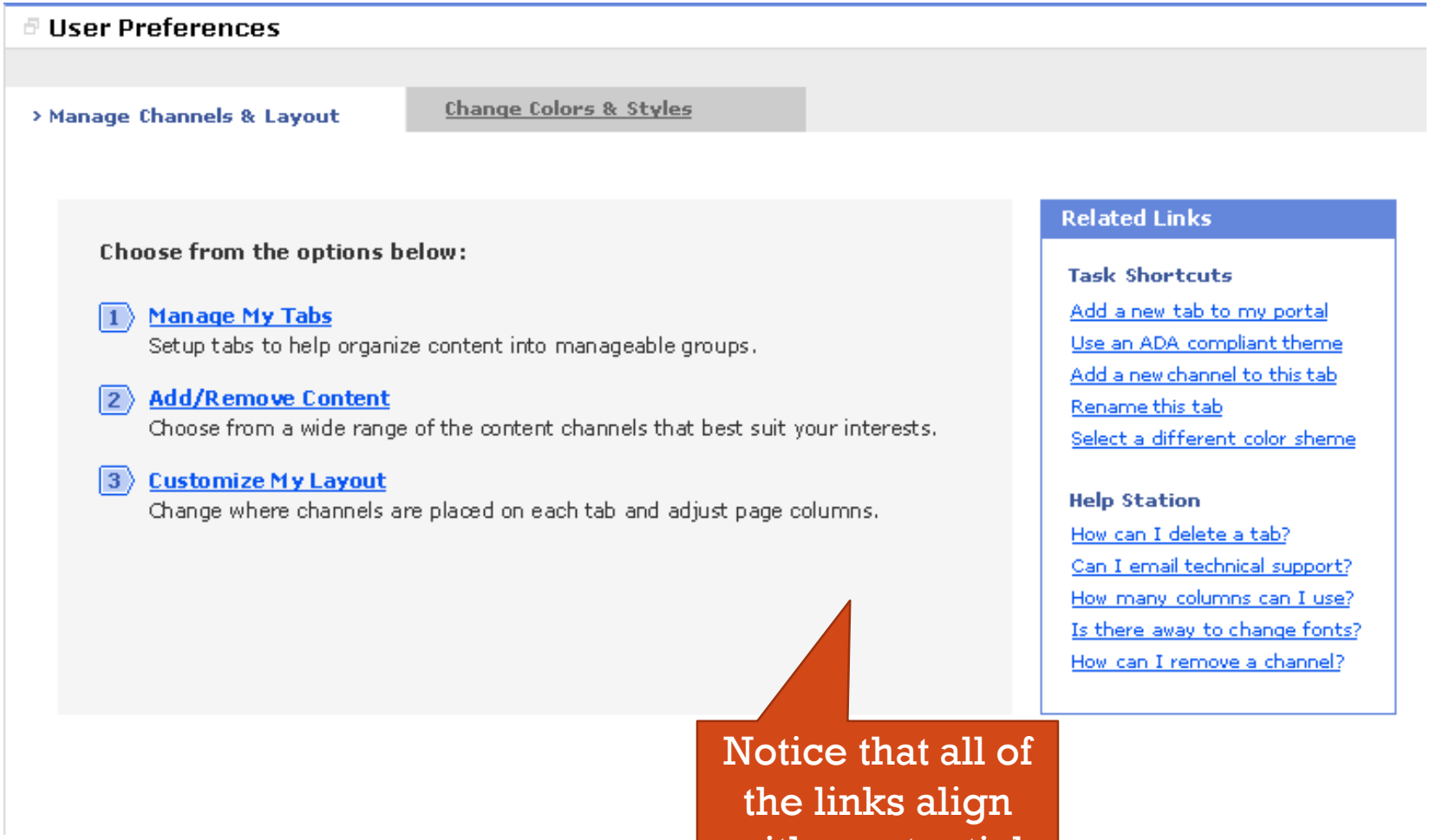
800 x 600 1024 x 768



- No progressive disclosure
- Poor system feedback
- No task flow pattern
- Overly literal translation
- Unconventional and ambiguous buttons
- Inconsistent controls & operations (ambiguous modes)
- Overly redundant controls
- Ineffective use of white space, awkward dimensions
- Relies on recall, rather than recognition
- Poor / no access to help

IxD BEST PRACTICES

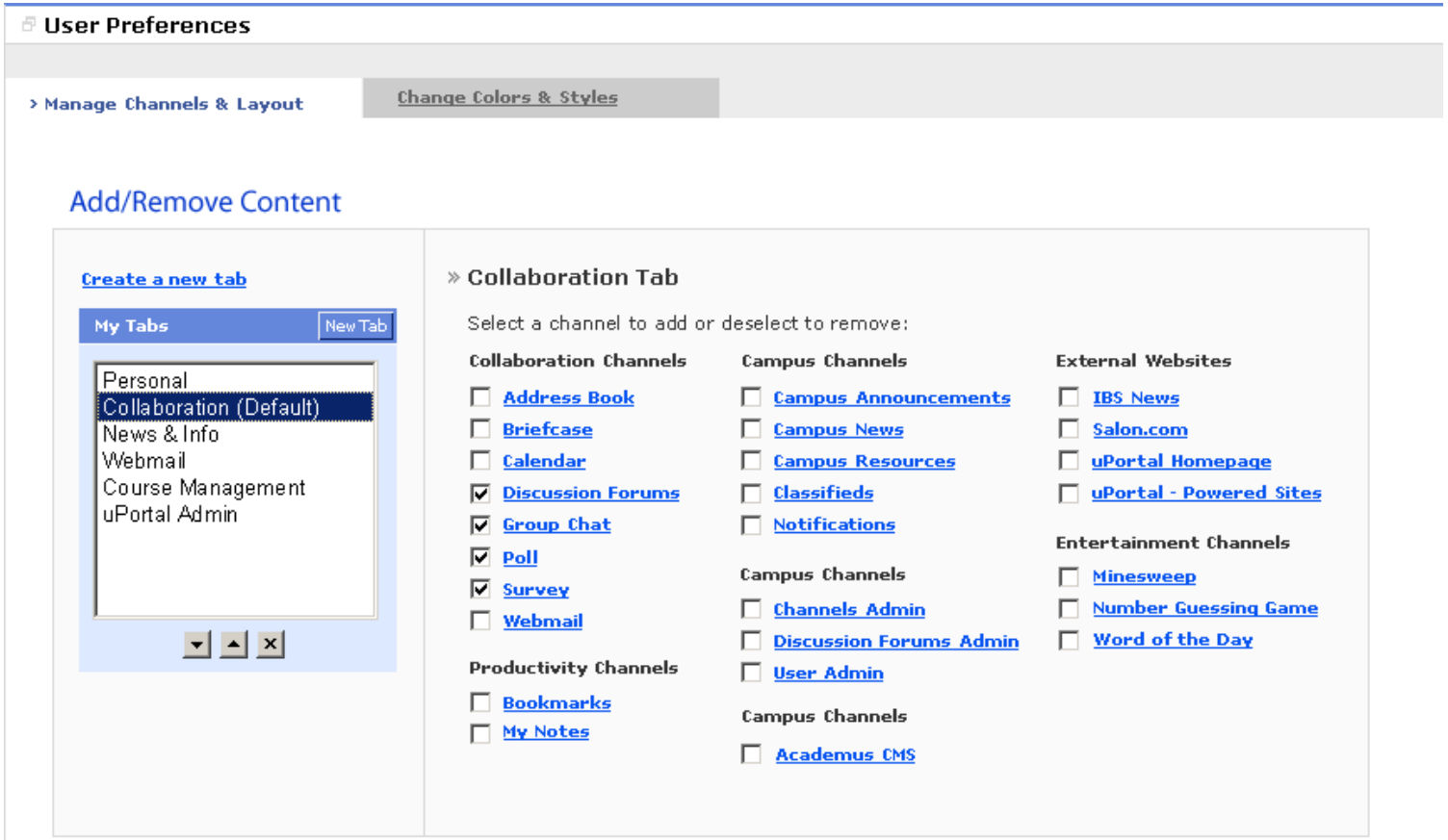
- **Example of Preferences**



Goal and Task oriented

IxD BEST PRACTICES

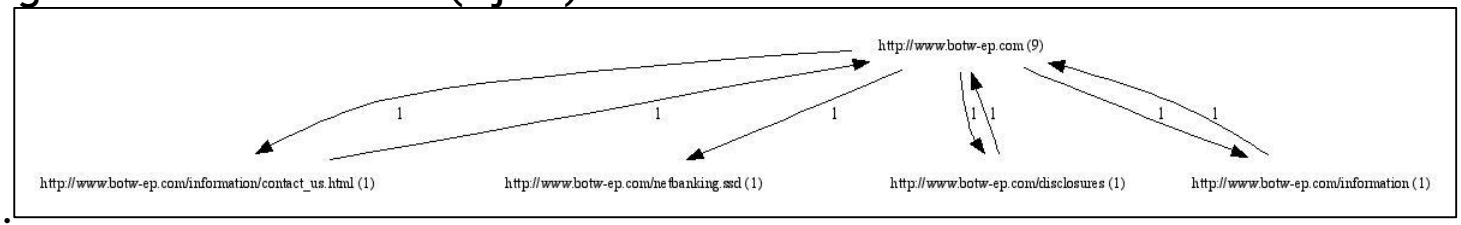
- **Example of Preferences**



Use workflow patterns like a guide.

IxD IN THE CONTEXT OF UCD

- Some aspects of IxD blend in with UCD:
- Storyboarding
  - Our storyboards should provide a *flow* to our application
  - Any interaction that "changes screens" must be planned carefully
    - This is changing somewhat with web (AJAX) & mobile

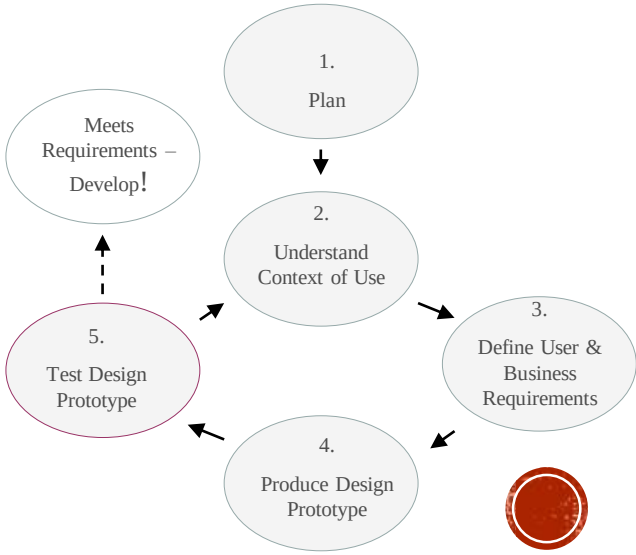


Clickstream Analysis

- The analyzing of user interactions ("clicks") to see if s/he is using the system/app in the manner designed
  - What path is the user taking? Is it the intended path?
  - How many false/exploratory actions? Is the user frustrated?
- This is also changing ("swipestream analysis"?)

Refinement

- We discussed how UCD has iterations as much for refinement as for exploration of alternatives
- Interaction models certainly require refinement.



IxD SUMMARY

- IxD is a subset of overall UI Design and UX
  - Defines a *planned set of interactions* to help the end user accomplish a *goal*
  - A significant UCD aspect is the *validation* that the end user uses the design in the manner intended
    - If s/he does not, is there frustration?
  - The nature of IxD is changing
    - our computing devices are changing in both physical interaction and form factor
    - And they are ubiquitous and used in "bursts"
      - Count how many times you take out your phone per day

- IxD comes before UI Graphical Design Modeling
  - That is, you map out interactions to achieve goals before you worry about the bells and whistles
  - IxD generally concerns itself with *macro-interactions*, not *micro* (which widget will I use to accomplish low-level task T?)

M4 USER INTERFACE DESIGN

CST315 – Fall 2015 Revision

Arizona State University

Notes from Nathan Pearson and Sommerville 9th edition

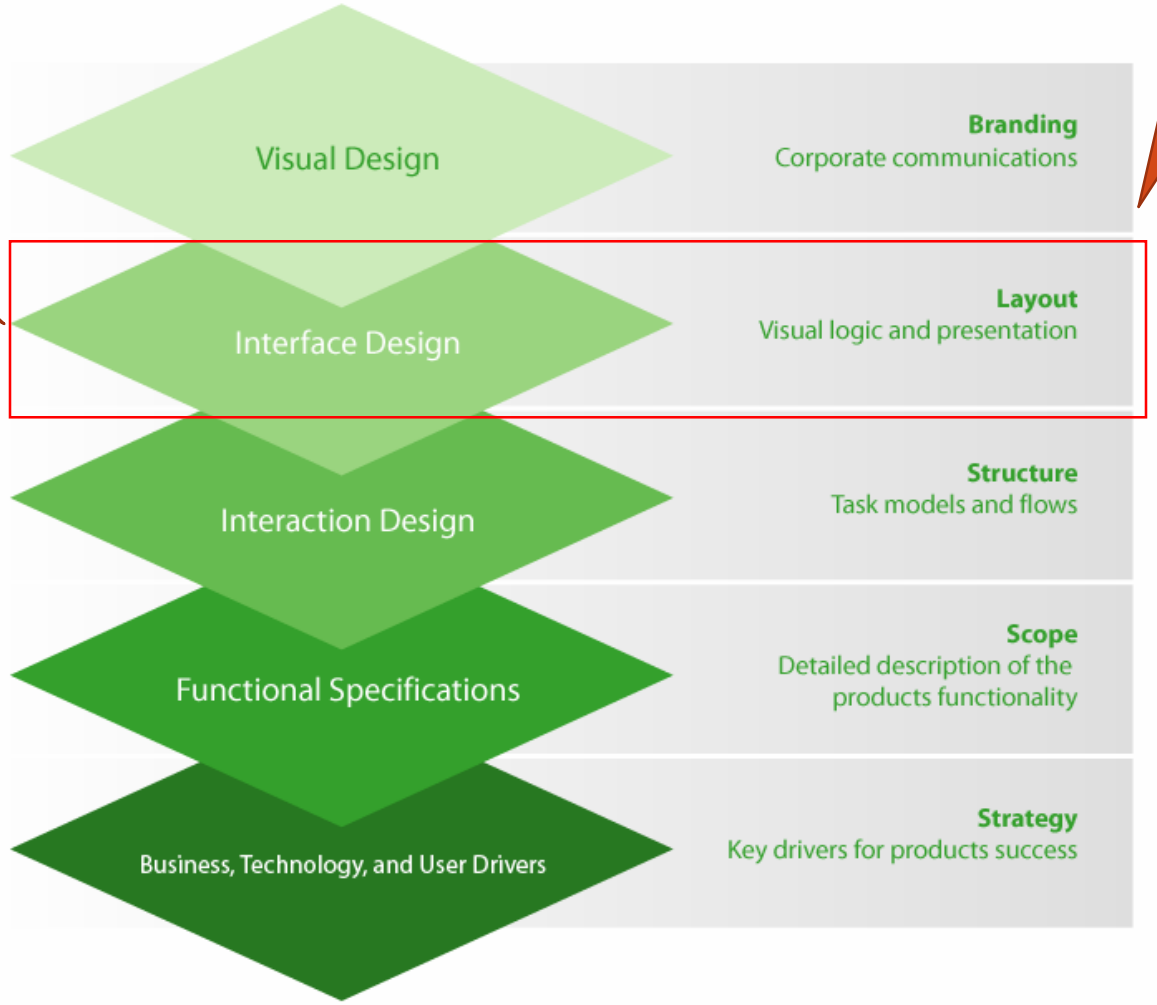
OUTLINE (FOR EXAM)

- Human Factors in Interface Design
- UI Design Principles
- Layout Management
- Usability Heuristics
- Modularity in Design
- Information presentation
  - Visual Information-Seeking Mantra
  - Information display factors
  - Analogue or digital presentation?
  - Error messages

INTERACTION DESIGN

We won't go into visual design, as it borders graphic design and this course focuses on technical aspects.

Our focus in this module is on interface layouts and components.



HUMAN FACTORS IN INTERFACE DESIGN

Mentioned in hints!

There are four basic considerations:

- Limited short-term memory
  - People can instantaneously remember about 7 items of information. If you present more than this, they are more liable to make mistakes.
- People make mistakes
  - When people make mistakes and systems go wrong, inappropriate alarms and messages can increase stress and hence the likelihood of more mistakes.
- People are different
  - People have a wide range of physical capabilities. Designers should not just design for their own capabilities.
- People have different interaction preferences
  - Some like pictures, some like text.

UI DESIGN PRINCIPLES

Principles

- Designers should be aware of people's physical and mental limitations (e.g. limited short-term memory) and should recognise that people make mistakes.
- UI design principles underlie interface designs although not all principles are applicable to all designs.
- User interfaces should be designed to match the skills, experience and expectations of its anticipated users.

Remember that your interface is the gateway to the functionality offered by a system. If users cannot effectively use the interface, it does not matter if your system is useful or technically impressive.

Impacts

- System users often judge a system by its interface rather than its functionality.
- A poorly designed interface can cause a user to make catastrophic errors.
- Poor user interface design is the reason why so many software systems are never used.

LAYOUT MANAGEMENT

- How widgets and information are laid out on the display has a significant impact on user effectiveness
  - Consistency – within your design, repeat style patterns. Ex: use the same font, colors, or design elements, across your entire design.
  - Convention – follow patterns that other systems use. Ex: If most systems have menu bars at the top, then you probably shouldn't put yours at the bottom.
  - Proximity/Alignment – Items that are related should be placed together so they become one visual unit. Helps organize information and reduce clutter. Ex: put all of the links to different pages in the same menu bar – do not scatter them.
  - Contrast – if some elements of a page have different behavior, that you should contrast those elements so they stand out. Ex: if you have links in the middle of text block, make them a difference color (e.g. blue).

LAYOUT MANAGEMENT

- New devices and practices have caused some changes
  - Layouts have become minimalistic and goal-oriented
  - Responsive design – the practice of creating a common layout even on smaller form factors (phones)
    - Have you seen a non-responsive site (m.asu.edu)?
  - But some things haven't changed
    - Layouts are still "box-ey"
- Technologies
  - Development platforms have APIs (Windows, Qt, Java)
  - Alternative: **declarative** (primarily XML-based technologies)
    - See http://en.wikipedia.org/wiki/User_interface_markup_language
    - E.g. XUL, Android, DOT, XAML

Mentioned in hints!

Remember SER221? The basic idea is to define a system in terms of what a 'correct' solution looks like, rather than give specific (imperative) instructions to achieve it.

Instead of placing an element at a specific position, we would declare it as being 'top-right' or 'bottom-right'. That is, give a general rule that would work for many cases (screen resolutions).

USABILITY HEURISTICS (BASED ON CONVENTIONS AND STANDARDS)

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency in design
- Error prevention (or recovery)
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Access to help and documentation
- Etc…

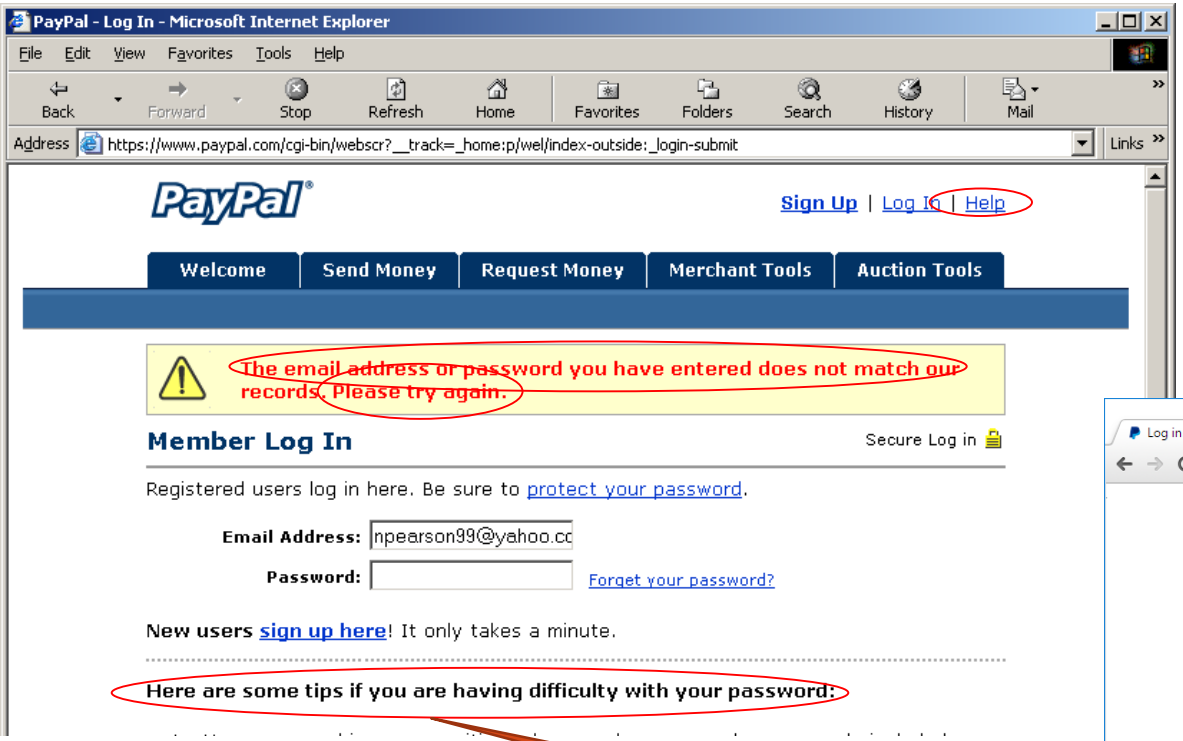
Helpful Resources: Jakob Nielson, Bruce Tognazzini

Some examples of good and bad interfaces follow.

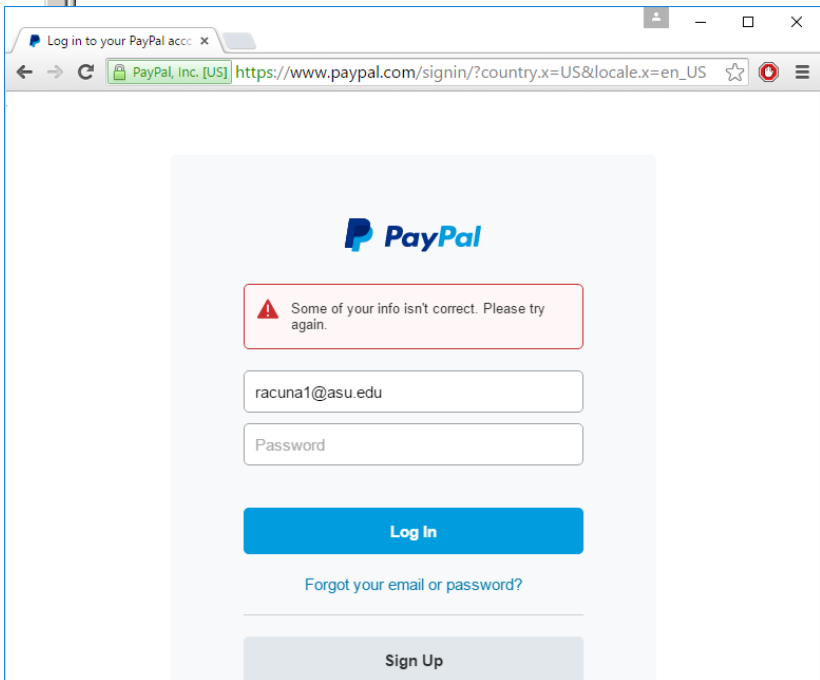
EXAMPLE OF GOOD USABILITY HEURISTICS: VISIBILITY OF SYSTEM STATUS

The system should indicate what is happening, with appropriate feedback within reasonable time.

- Help users recognize, diagnose, and recover from errors. Tell them what is wrong, and do so in common language.
- Match between system and the real world. The system should use words, phrases and concepts that are familiar to the user - don't start talking about null pointers!
- Access to help and documentation. Usually on demand, and searchable.



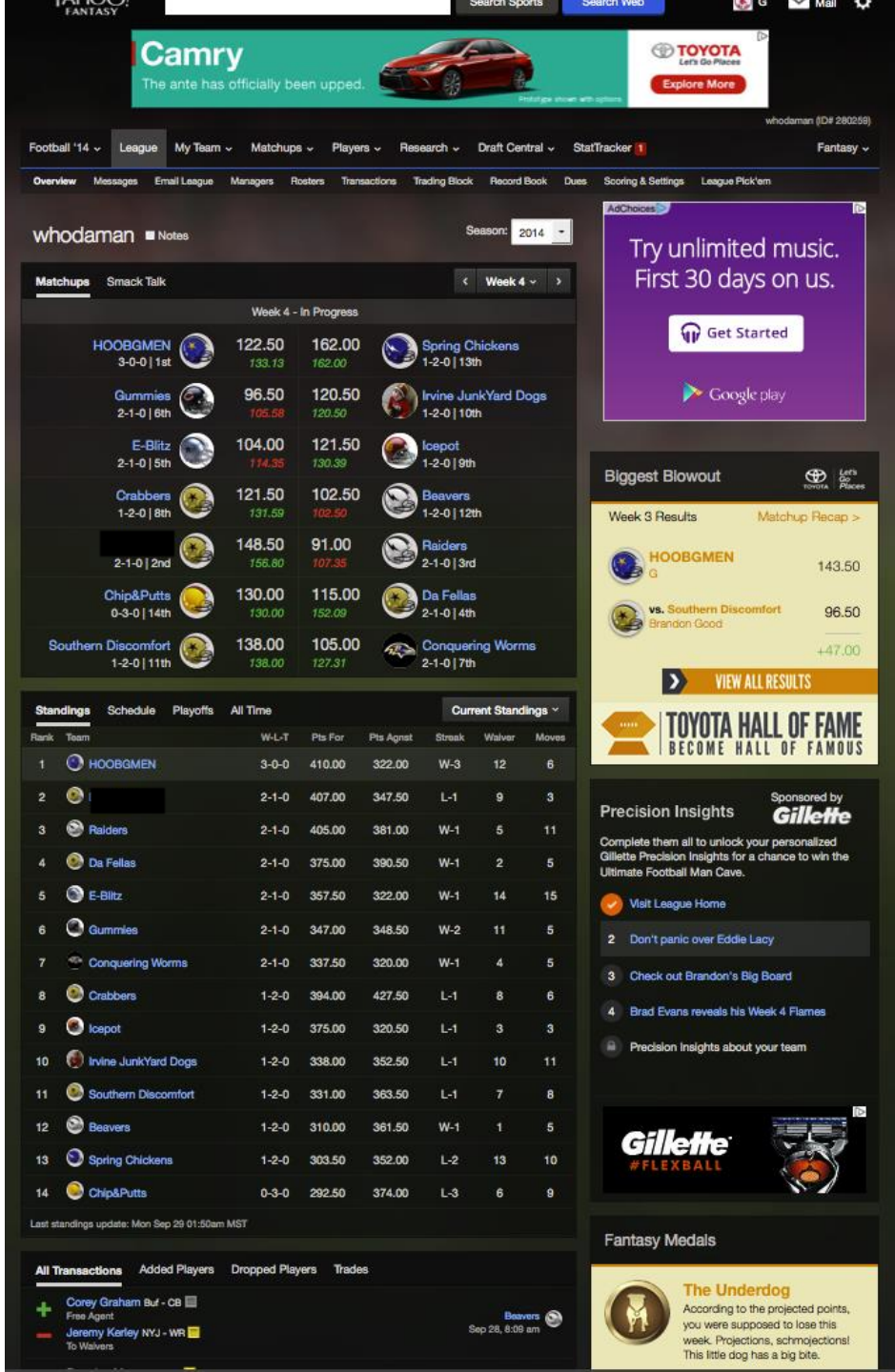
Best practice: suggest how to resolve the issue.



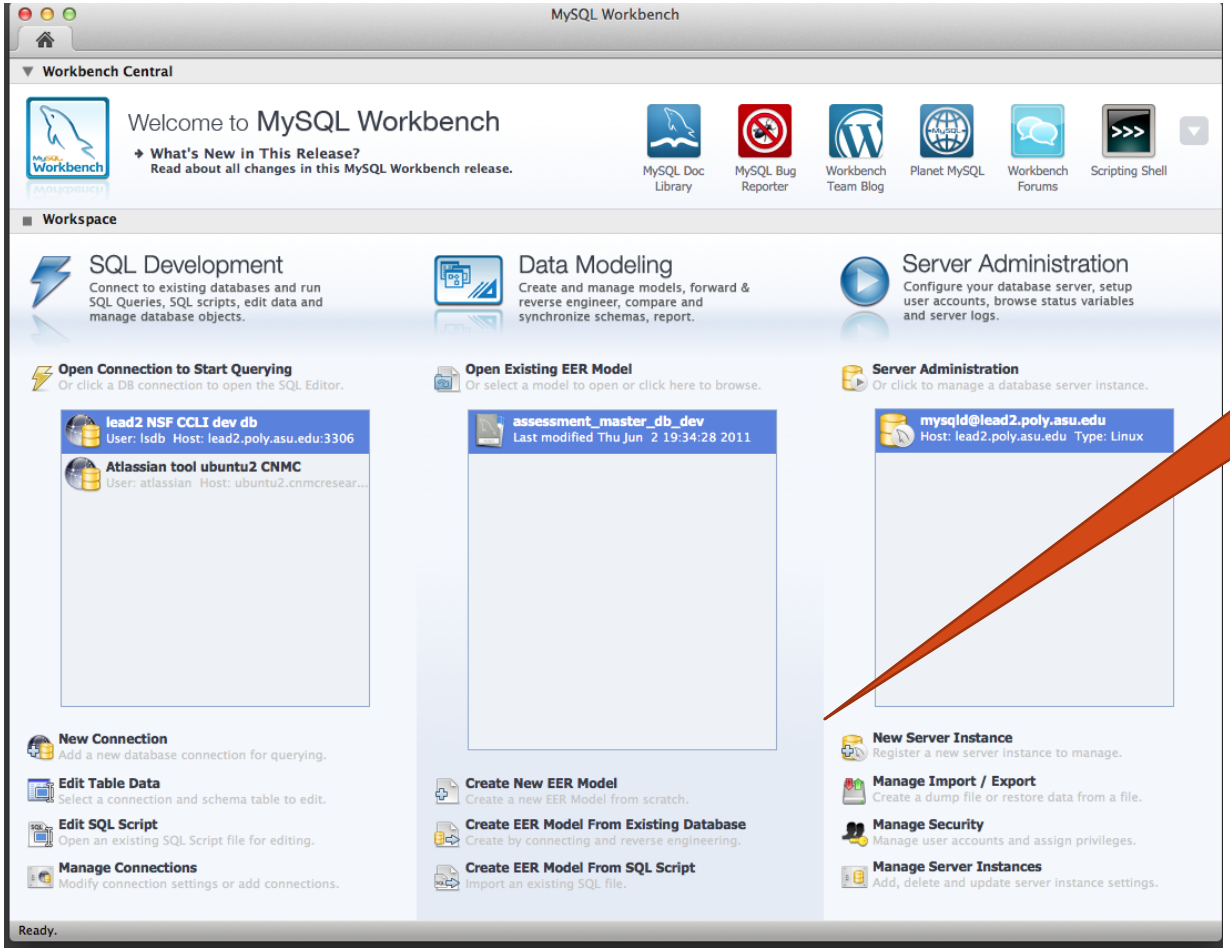
- Above: PayPal 2012 login screen
- Right: PayPal 2015 login screen
- What has changed?

COMMON ERRORS - CLUTTER & VISUAL NOISE

Where is the content???

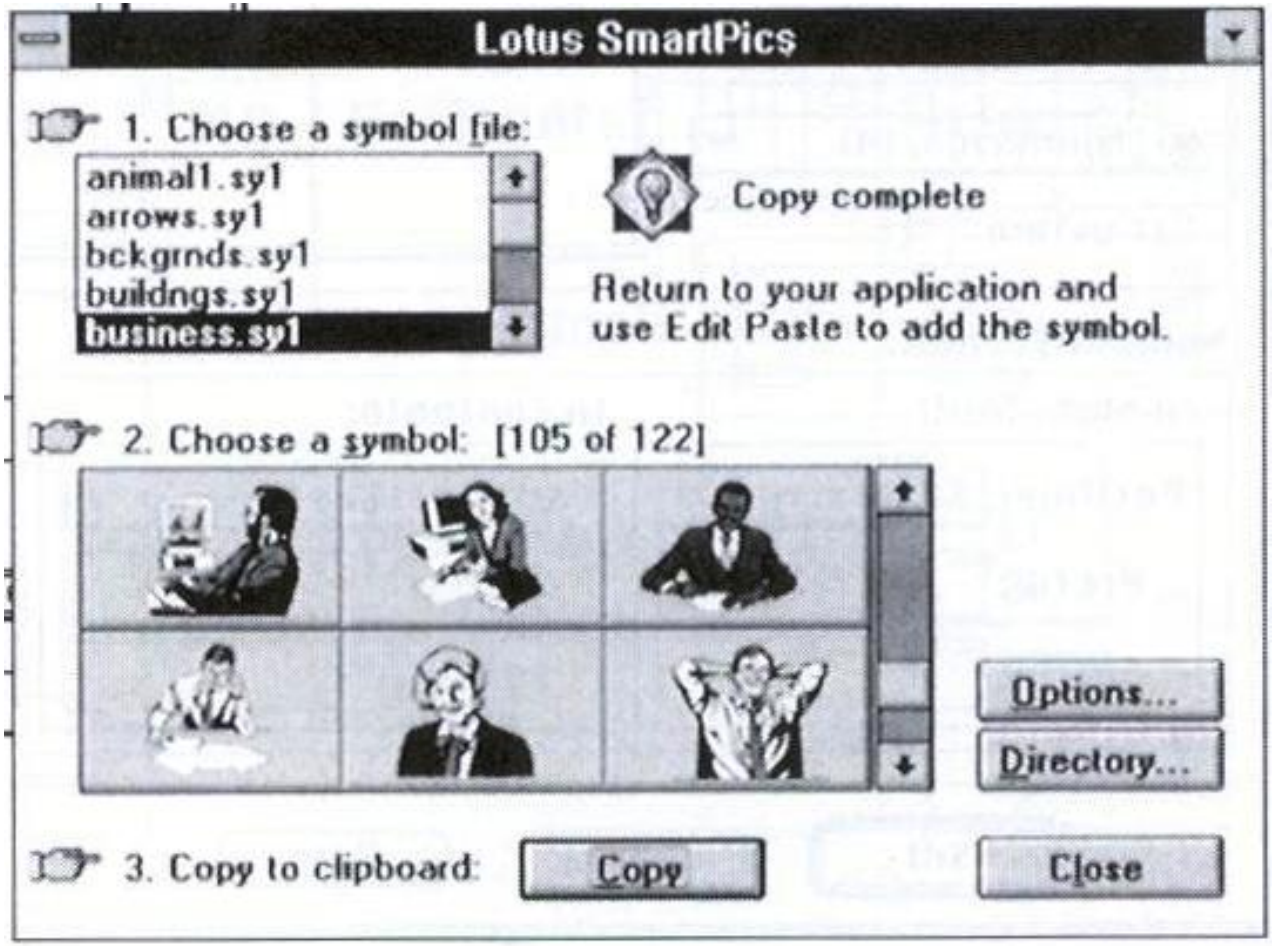


COMMON ERRORS – USING EXPLICIT STRUCTURE AS A CRUTCH



Extra elements (line divisions) have been added to organize the different regions of an interface. Sometime can be appropriate, sometimes indicates larger clutter issues.

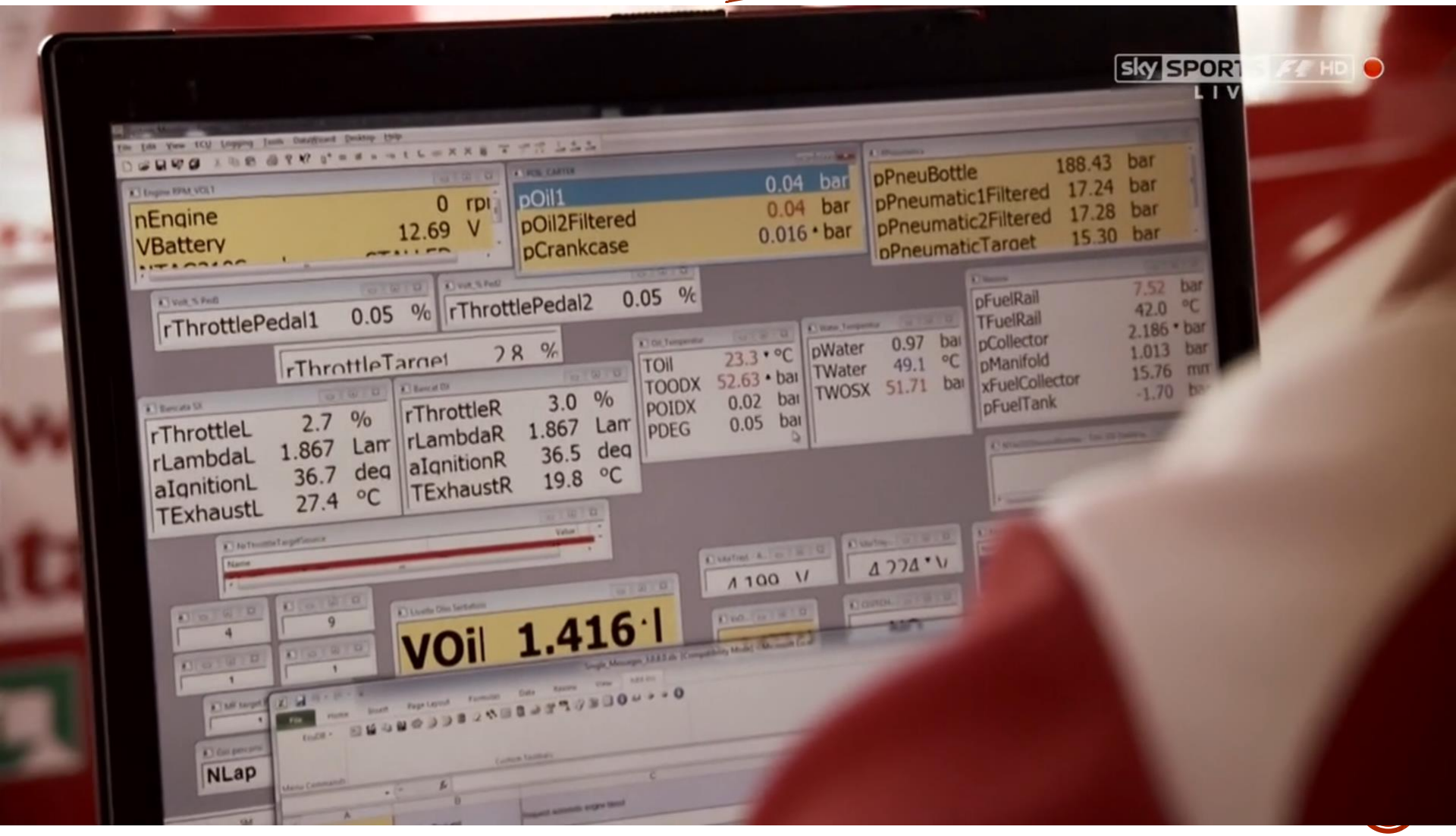
COMMON ERRORS – BELABORING THE OBVIOUS



Too many unnecessary details, too much redundancy.

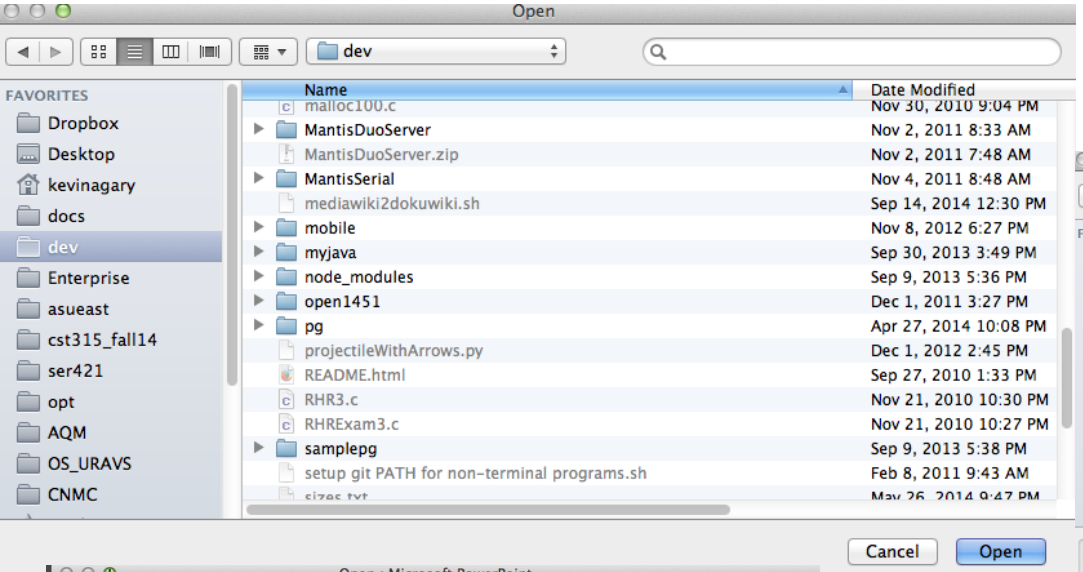
COMMON ERRORS – POOR USE OF SPATIAL LOGIC

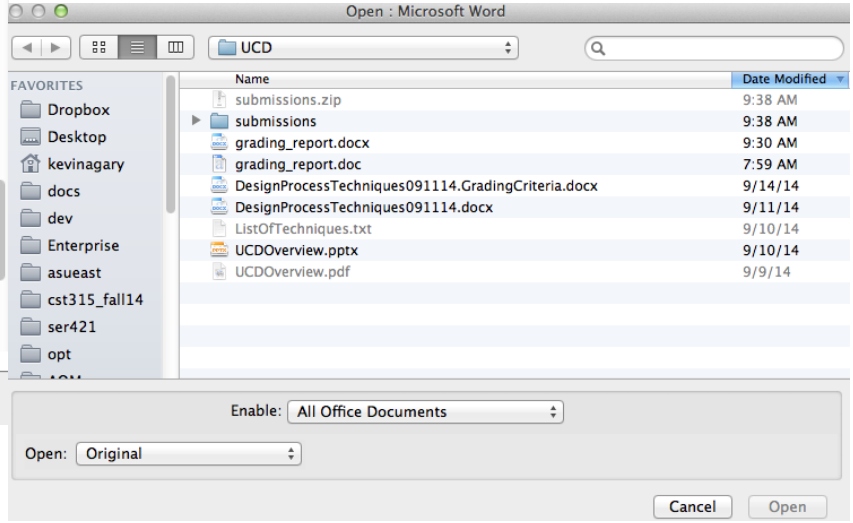
Remember Proximity/Alignment from layout management a few slides ago?

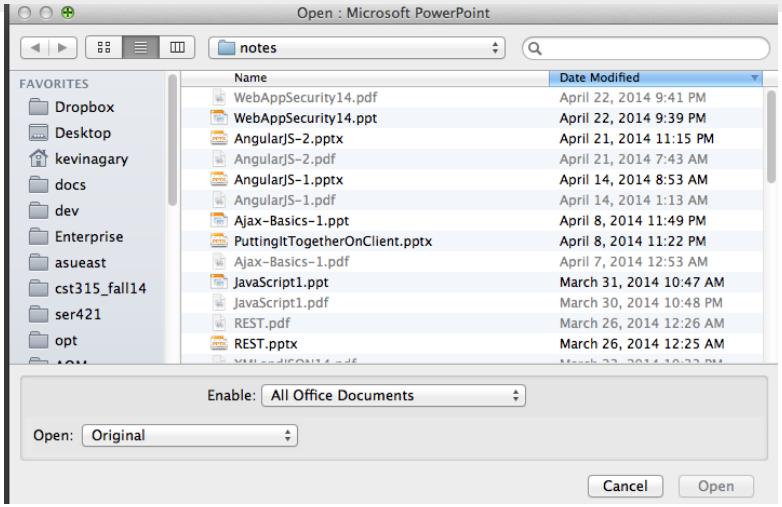


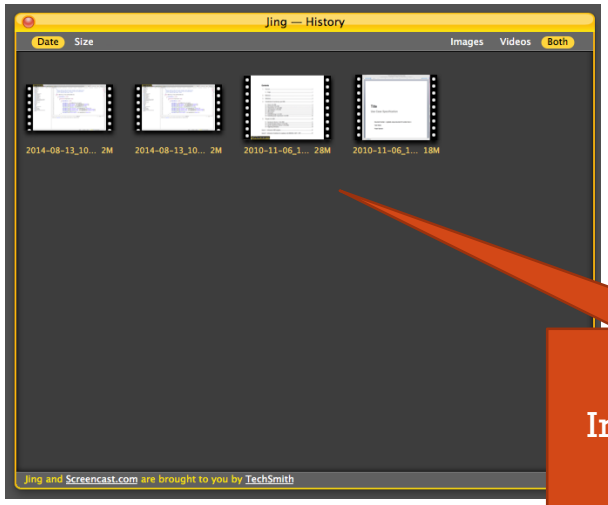
GOOD UI: MODULARITY IN DESIGN

Not only are these windows built using common modules, but the modules have a familiar arrangement.









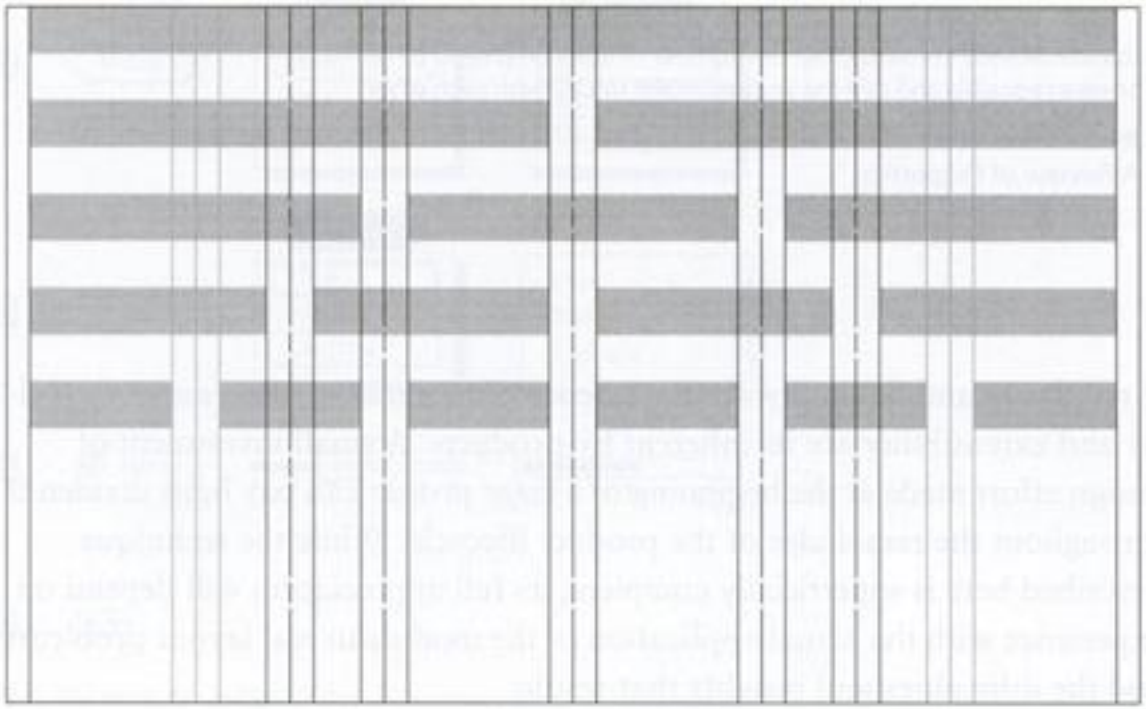
Functional and Esthetic Excellence

In comparison to the other three, how usable is this UI?

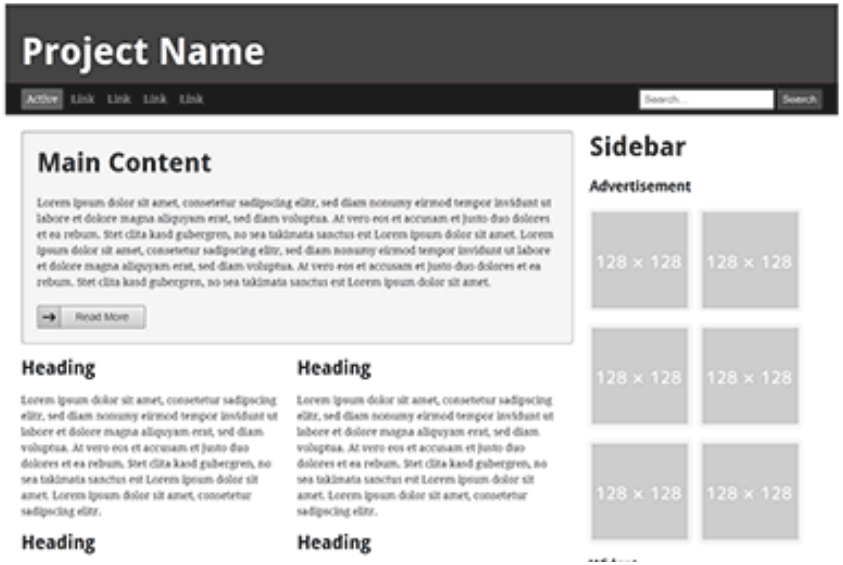
Simple repetition enhances apparent consistency

MODULARITY IN DESIGN

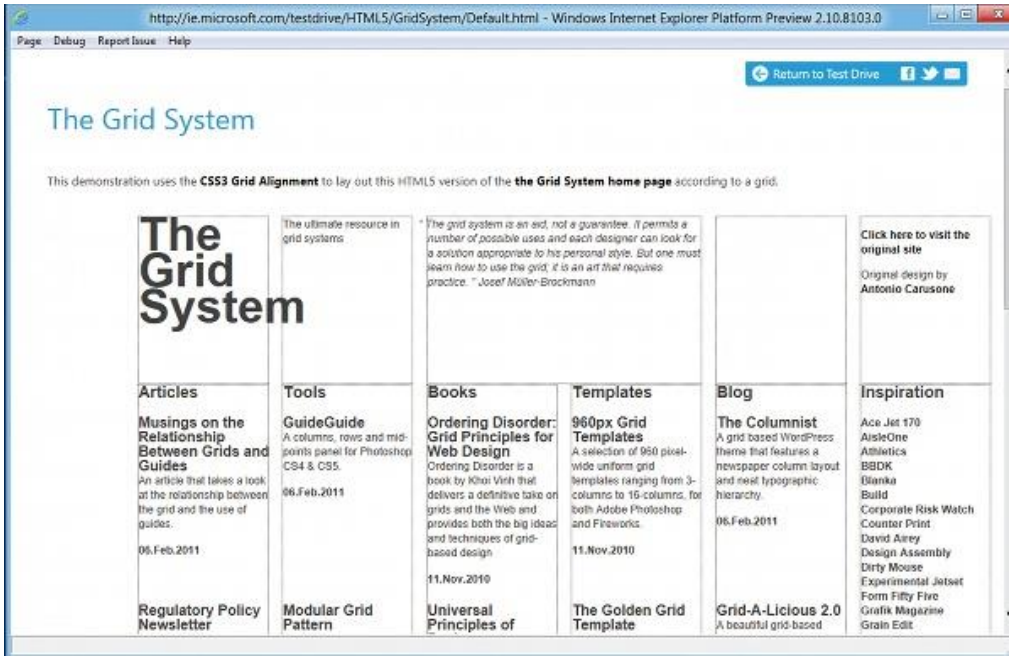
- One familiar way to arrange modules/components is grids or columns.
- Grid system supports two, three, four, and six column layouts in any graphical user interface.
- Useful from the standpoint that we can programmatically resize (reflow) while maintaining some consistency.



MODULARITY IN DESIGN



Two Column Grid (Nested Flex) Layout



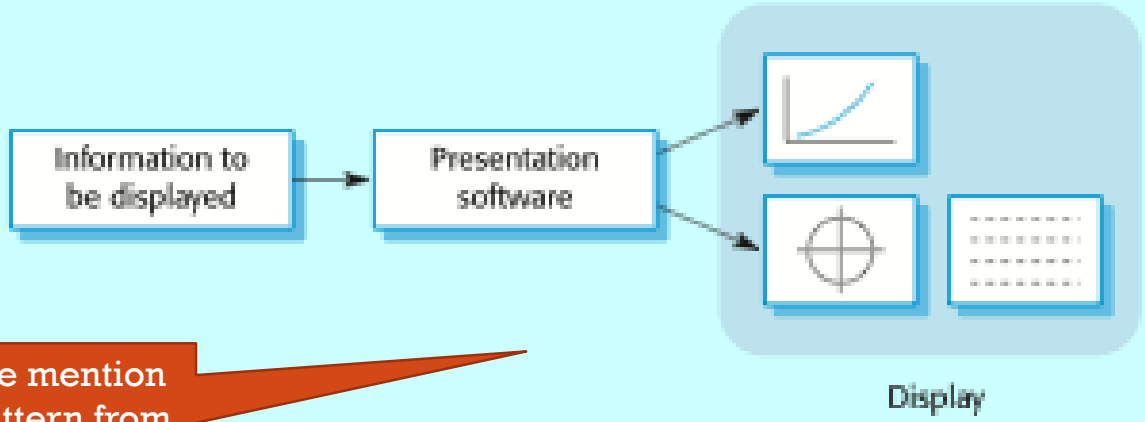
Six Column Grid Layout

- Modern layout technologies and design approaches (like *flexboxes*) rely heavily on grids but provide flexible, customizable configurations.
- Programming support for grid layout often has a declarative nature. For example, flexboxes let us specify rules like cell order, minimum size, maximum size, etc., which the layout engine uses to decide where to exactly place elements.

Helpful Resource: Darrell Sano and Kevin Mullet - Designing Visual Interfaces: Communication Oriented Techniques

INFORMATION PRESENTATION

- Information presentation is concerned with presenting system information to system users.
- The information may be presented directly (e.g. text in a word processor) or may be transformed in some way for presentation (e.g. in some graphical form).
- In general, we want to think of a layer approach that involves a separation between data representation and presentation:



Remember the mention of the MVC pattern from the reading? How does it compare to this figure?

Mentioned in hints!

VISUAL INFORMATION-SEEKING MANTRA

A first step to designing information-centric UIs is to consider the following pattern:

"Overview first, zoom and filter,
then details-on-demand."

[Shneiderman, 1996]

INFORMATION DISPLAY FACTORS

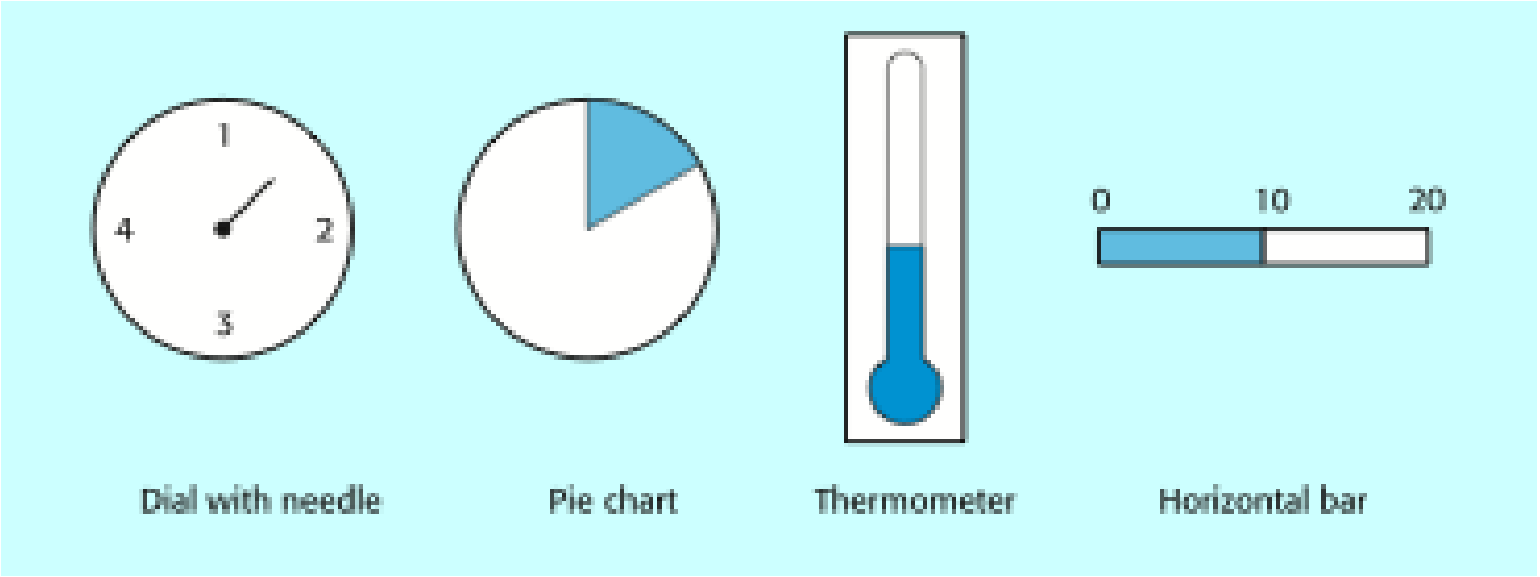
- Is the user interested in precise info or data relationships?
- How quickly do information values change?
  Must the change be indicated immediately?
- Must the user take some action in response to a change?
- Is there a direct manipulation interface?
- Is the info textual or numeric? Are relative values important?

There are two basic types of information:

- Static information
  - Initialized at the beginning of, and does not change during, a session
  - May be either numeric or textual.
- Dynamic information
  - Changes during a session and the changes must be communicated to the system user. (A consideration: how long might a session be? what if a terminal is left open over night?)
  - May be either numeric or textual.

ANALOGUE OR DIGITAL PRESENTATION?

- Digital presentation (i.e., displaying digits)
  - Compact - takes up little screen space;
  - Precise values can be communicated.
- Analogue presentation (i.e., displaying figure)
  - Easier to get an 'at a glance' impression of a value;
  - Possible to show relative values;
  - Easier to see exceptional data values.



ERROR MESSAGES

- Error message design is critically important.
  - Poor error messages can mean that a user rejects rather than accepts a system.
- Messages should be polite, concise, consistent and constructive.
- The background and experience of users should be the determining factor in message design. (notice the UCD theme!)
- While this is the *theory,* in many cases errors are conveyed poorly
  - Users often to blame when they are not
  - No corrective action given, user is at a dead-end
  - Over-notification, "everything I do is wrong"
    - Perhaps the user is not the primary stakeholder
    - User ignores warnings – then something bad happens!

Example: skipping the Windows UAC prompt.

DESIGN FACTORS IN MESSAGE WORDING

Know your users!

Factor	Description
Context	Wherever possible, the information generated by the system should reflect the current user context. As far as is possible, the system should be aware of what the user is doing and should generate messages and indicators that are relevant to their current activity.
Experience	As users become familiar with a system they become irritated by long, 'meaningful' messages. However, beginners find it difficult to understand short terse statements of a problem. You should provide both types of message and allow the user to control message conciseness. This is a form of *progressive disclosure*.
Skill level	Information presentation should be tailored to the user's skills as well as their experience. Information for the different classes of user may be expressed in different ways depending on the terminology that is familiar to the reader.
Style	Messages should be positive rather than negative. They should use the active rather than the passive mode of address. They should never be insulting or try to be funny.
Culture	Wherever possible, the designer of information presentation should be familiar with the culture of the country where the system is sold. There are distinct cultural differences between Europe, Asia and America. A suitable presentation for one culture might be unacceptable in another.