User Interface Design (Ch. 11)

Cengiz Günay
CS485/540 Software Engineering

Fall 2014, Some slides courtesy of Joan Smith, Roger Pressman, Ian Sommerville, and the Internets
Interface Design

Easy to learn?
Easy to use?
Easy to understand?

These slides are designed to accompany Software Engineering: A Practitioner’s Approach, 7/e (McGraw-Hill, 2009) Slides copyright 2009 by Roger Pressman.
Interface Design

**Typical Design Errors**

- lack of consistency
- too much memorization
- no guidance / help
- no context sensitivity
- poor response
- Arcane/unfriendly
Golden Rules

- Place the user in control
- Reduce the user’s memory load
- Make the interface consistent
Place the User in Control

- Define interaction modes in a way that does not force a user into unnecessary or undesired actions.
- Provide for flexible interaction.
- Allow user interaction to be interruptible and undoable.
- Streamline interaction as skill levels advance and allow the interaction to be customized.
- Hide technical internals from the casual user.
- Design for direct interaction with objects that appear on the screen.
### Find an Existing Value

Search By: | Request ID
---|---
Request ID: | [Enter Request ID]

**Search Results**

<table>
<thead>
<tr>
<th>Request ID</th>
<th>Create Position Status</th>
<th>Department</th>
<th>Description</th>
<th>Position Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0000001</td>
<td>Created</td>
<td>027521</td>
<td>Test 1</td>
<td>00010720</td>
</tr>
<tr>
<td>P0000002</td>
<td>New</td>
<td>011004</td>
<td>Secretary</td>
<td>(blank)</td>
</tr>
<tr>
<td>P0000003</td>
<td>Created</td>
<td>027521</td>
<td>Test 2</td>
<td>00010725</td>
</tr>
<tr>
<td>P0000004</td>
<td>Created</td>
<td>053301</td>
<td>Director</td>
<td>00010723</td>
</tr>
<tr>
<td>P0000005</td>
<td>Created</td>
<td>027521</td>
<td>Test Director</td>
<td>00010721</td>
</tr>
<tr>
<td>P0000006</td>
<td>Created</td>
<td>027521</td>
<td>Test approval / position creation</td>
<td>00010724</td>
</tr>
<tr>
<td>P0000007</td>
<td>Created</td>
<td>027521</td>
<td>test</td>
<td>00010726</td>
</tr>
<tr>
<td>P0000008</td>
<td>Created</td>
<td>027521</td>
<td>Test grey-out logic</td>
<td>00010727</td>
</tr>
<tr>
<td>P0000009</td>
<td>Pending</td>
<td>031101</td>
<td>Other Job</td>
<td>(blank)</td>
</tr>
<tr>
<td>P0000010</td>
<td>Pending</td>
<td>027521</td>
<td>Al-Test Error Msks</td>
<td>(blank)</td>
</tr>
<tr>
<td>P0000011</td>
<td>Created</td>
<td>026491</td>
<td>Assistant Director</td>
<td>00010739</td>
</tr>
</tbody>
</table>

View All | First  | 1-34 of 34 | Last
Reduce the User’s Memory Load

- Reduce demand on short-term memory.
- Establish meaningful defaults.
- Define shortcuts that are intuitive.
- The visual layout of the interface should be based on a real world metaphor.
- Disclose information in a progressive fashion.
Make the Interface Consistent

- Allow the user to put the current task into a meaningful context.
- Maintain consistency across a family of applications.
- If past interactive models have created user expectations, do not make changes unless there is a compelling reason to do so.
Interface Design Steps

- Using information developed during interface analysis, define interface objects and actions (operations).
- Define events (user actions) that will cause the state of the user interface to change. Model this behavior.
- Depict each interface state as it will actually look to the end-user.
- Indicate how the user interprets the state of the system from information provided through the interface.
Activity Diagram

Supplements the use case by providing a graphical representation of the flow of interaction within a specific scenario.
Swimlane Diagrams

Allows the modeler to represent the flow of activities described by the use-case and at the same time indicate which actor (if there are multiple actors involved in a specific use-case) or analysis class has responsibility for the action described by an activity rectangle:

- **Actors**
- **Timing**
- **Flow**
User Analysis

- Are users trained professionals, technician, clerical, or manufacturing workers?
- What level of formal education does the average user have?
- Are the users capable of learning from written materials or have they expressed a desire for classroom training?
- Are users expert typists or keyboard phobic?
- What is the age range of the user community?
- Will the users be represented predominately by one gender?
- How are users compensated for the work they perform?
- Do users work normal office hours or do they work until the job is done?
- Is the software to be an integral part of the work users do or will it be used only occasionally?
- What is the primary spoken language among users?
- What are the consequences if a user makes a mistake using the system?
- Are users experts in the subject matter that is addressed by the system?
- Do users want to know about the technology the sits behind the interface?
Design Issues

- Response time
- Help facilities
- Error handling
- Menu and command labeling
- Application accessibility
- Internationalization
Usability Testing

- Define tasks
  - user stories
  - use cases
- Define users
  - Testers
  - Regular users
- Test and collect measurements
  - Time to complete task
  - Common errors
  - Complaints
  - Bugs
Facebook in 1997...
WebApp Interface Design

- **Where am I?** The interface should
  - provide an indication of the WebApp that has been accessed
  - inform the user of her location in the content hierarchy.
- **What can I do now?** The interface should always help the user understand his current options
  - what functions are available?
  - what links are live?
  - what content is relevant?
- **Where have I been, where am I going?** The interface must facilitate navigation.
  - Provide a “map” (implemented in a way that is easy to understand) of where the user has been and what paths may be taken to move elsewhere within the WebApp.
Effective WebApp Interfaces

- Bruce Tognozzi [TOG01] suggests…
  - Effective interfaces are visually apparent and forgiving, instilling in their users a sense of control. Users quickly see the breadth of their options, grasp how to achieve their goals, and do their work.
  - Effective interfaces do not concern the user with the inner workings of the system. Work is carefully and continuously saved, with full option for the user to undo any activity at any time.
  - Effective applications and services perform a maximum of work, while requiring a minimum of information from users.
Mapping User Objectives

List of user objectives

objective #1
objective #2
objective #3
objective #4
objective #5
objective #n

Navigation menu

graphic, logo, and company name

Menu bar
major functions

Home page text copy

graphic

These slides are designed to accompany Software Engineering: A Practitioner’s Approach, 7/e (McGraw-Hill, 2009) Slides copyright 2009 by Roger Pressman.
Aesthetic Design

- Don’t be afraid of white space.
- Emphasize content.
- Organize layout elements from top-left to bottom right.
- Group navigation, content, and function geographically within the page.
- Don’t extend your real estate with the scrolling bar.
- Consider resolution and browser window size when designing layout.
Design Evaluation Cycle

- preliminary design
- build prototype #1 interface
- user evaluates interface
- design modifications are made
- build prototype # n interface
- interface design is complete
- evaluation is studied by designer
12 Techniques for UI Design

From: “12 Useful Techniques For Good User Interface Design” by Dmitry Fadeyev

1. Highlight important changes
2. Enable keyboard shortcuts in your Web application
3. Upgrade options from the account page
4. Advertise features of the application
5. Use color-coded lists
6. Offer personalization options
7. Display help messages that attract the eye
8. Design feedback messages carefully
9. Use tabbed navigation
10. Darken background under modal windows
11. Lightboxes and Slideshows
12. Short sign-up forms