Outline—User Interface Design

1. Why UI design now?

2. Rules of User Interface Design,
   (a) Basic Design/Analysis
   (b) Facilitating Users’ Learning/Adoption
   (c) Minimizing the User’s “Mental Load”
   (d) Handling Errors/Exceptions

3. Conclusion
The Rules by Group

1. *Initial Design Analysis Rules*
   (a) Introduce through experience.
   (b) Use the user's pre-existing model.
   (c) Design Never Ceases.
   (d) Keep Logs of use.

2. *Facilitating User's Learning/Adoption*
   (a) Avoid Acausality.
   (b) Put the user in Control.
   (c) Group Dialogs when Reasonable (Parallel/Sequential Tradeoff)

3. *Minimizing "Mental Load"
   (a) Seek Uniformity & Consistency
   (b) Provide Help: Everywhere and Contextual.
   (c) Make the State always observable.

4. *Handling Errors/Exceptions*
   (a) Validate All Data on Entry
   (b) Provide Reset: Everywhere and Contextual.
Rules of GUI Design—Framework

When applicable and possible, we use this framework:

1. The Basic Principle of the Rule
   • Why this rule?

2. A Good example: the application of this rule
   • What can we learn from this example?

3. Bad Example: breaking the rule
   • How can we avoid this kind of goof?
Group 1: Initial Design/Analysis Rules

1. Introduce through experience.

2. Design Never Ceases.
   - **Reasons**: Have to get user’s input early and often. Rapid Prototyping—high-level languages and tools.

3. Use the user’s pre-existing model.
   - **Reasons**: The computer is the new “kid on the block”. Has to confirm to old way of doing things.
   - **Good**: http://www.mwave.com (main page).
   - **Bad**: http://www.macconnection.com (Main Page).

4. Keep Logs of use.
   - **Reasons**: Identify/Diagnose/fix user’s problems, Provide Useful feedback to users.
   - **Good**: http://www.amazon.com (Recommendations)
Group 2: Facilitate Learning/Adoption

1. Avoid acausality. Always give relevant, comprehensible, feedback.
   - **Reason**: Give user simple, unsuperstitious view of system.
   - **Good**: Copying “gauges” on Macintosh/Wintel
   - **Bad**: http://chat.yahoo.com (it knows me?!?)
   - **Bad**: `rm a.*` versus `rm a. *`

2. Put the user in Control.
   - **Reason**: Give user all options needed to achieve goals.
   - **Good**: http://www.amazon.com: main & kids page

3. Group Dialogs when Reasonable (Parallel/Sequential Tradeoff)
   - **Reason**: Save user time & thinking.
   - **Good**: Get a) personal info b) specific purchases c) shipping info d) charging info. http://www.amazon.com
   - **Bad**: Most others.
Minimizing User’s Mental Load

1. Seek Uniformity & Consistency
   - **Reason**: User sees & does the same thing everywhere.
   - **Bad**: Unix: `cp, mv, ln` versus `chgrp, chown`

   - **Reason**: User gets just the needed help.
   - **Good**: Macintosh Balloon Help.
   - **Bad**: www.travelocity.com

3. Make the State always observable.
   - **Reason**: User knows location, and can select destination.
   - **Good**: http://www.macconnection.com
   - **Bad**: Most web sites
Handling Errors/Exception

1. Validate All Data on Entry
   - **Reason**: Don’t diddle the user around.
   - **Good**: http://www.amazon.com
   - **Bad**: Amazon, and http://yahoo.flifo.com

2. Provide reasonable reset: everywhere, contextual
   - **Reason**: Save user’s time and energy.
UI Design & Use Case Analysis

For each of user’s goals–

- Walk through the corresponding use case, with a candidate design of the user interface.

- For each step, ask yourself if these conditions are satisfied.

- You can't always satisfy them; if not, convince yourself why not, and document it.

- After complete a candidate design of all the UI for all the use cases, look for commonalities so that all the interfaces can have the same look and feel.