Application-Agnostic Interaction & Visualization Techniques

Pierre Dragicevic
Context-specific Design

- Taking into account the context of use leads to **optimal usability**

- The context of use includes:
  - The type of **task**
  - The type of **user**: specific skills or disabilities
  - The type of **environment**: desktop, mobile, car,...
Context-specific Design

- Example: choosing input devices

**Task**

**User**

**Environnement**
Context-specific Design

- Example: choosing input devices

[Hinkley et al. 94]
An Utilitarian Perspective

\[ \text{Utility} = \text{usability} \]
An Utilitarian Perspective

Utility = usability 
\times \text{number of users} 
\times \text{time they spend on the task}
An Utilitarian Perspective

Utility = usability
x number of users
x time they spend on the task

Common personal computing tasks:
- Data management
- Time & task management
- Communication
- ...

Generic design
Generic Design

- Users
- Environments / Platforms
- Tasks / Applications

Point design
Generic Design

Tasks / Applications

Environments / Platforms

Users

Generic design
Generic Design

Users

Environments / Platforms

Tasks / Applications
Generic Design

Tasks / Applications

Environments / Platforms

Users
Generic Design
Generic Design

Application-Agnostic Techniques

Tasks / Applications
Application-Agnostic Techniques

Make **few assumptions** about the target application

- Potential applicability: the technique can be used in lots of **prospective** applications
  Examples: Drag & Drop, Menus,...

- Effective applicability: the technique can directly benefit lots of **existing** applications.
Application-Agnostic Techniques

Make **few assumptions** about the target application

- Potential applicability: the technique can be used in lots of **prospective** applications
  Examples: Drag & Drop, Menus,…

- Effective applicability: the technique can directly benefit lots of **existing** applications.
Application-Agnostic Techniques

Three types:

1. **Window-level**: application are windows

2. **Widget-level**: applications are widget assemblies (menus, buttons, etc.)

3. **Device-level**: applications communicate with physical devices
Window-Level Techniques

= an application is a window
Window-Level Techniques

New windows management techniques
Widget-Level Techniques

= an application is a set of widgets
Widget-Level Techniques

= an application is a set of widgets
Widget-Level Techniques

GUI inspection (Accessibility APIs)

Type: Button
Name: New document
Description: Creates a new document
Command: new_doc()
Location: (75, 21) – (122, 35)

...
Widget-Level Techniques

Examples of Assistive Technologies

- **Generic Voice control**
- **Screen readers:**

```
Unlike other screen readers, VoiceOver is built right in to Mac OS X. You can even access VoiceOver at any time, by simply pressing command+F5. The VoiceOver cursor launches, allowing you to
```
Widget-Level Techniques

Other examples

Semantic pointing [Blanch et al. 2004]

Visual space

Motor space
Widget-Level Techniques

Other examples

**Phidgets / WidgetTaps** [Greenberg & Boyle 2002]
Widget-Level Techniques

Other examples

Widget Drag & Drop
Widget-Level Techniques

Limits:

- No standard accessibility API
- GUIs are more than sets of widgets
Widget-Level Techniques

Limits:
- No standard accessibility API
- GUIs are more than sets of widgets
Widget-Level Techniques

Limits:
- No standard accessibility API
- GUIs are more than sets of widgets
Device-Level Techniques

Three types:

1. **Window-level**: applications are windows

2. **Widget-level**: applications are widget assemblies (menus, buttons, etc.)

3. **Device-level**: applications communicate with physical devices
Device-Level Techniques
Device-Level Techniques

- Screen Magnifiers
Device-Level Techniques

- **Sticky keys**

  Use StickyKeys if you want to use Shift, Ctrl, or Alt key by pressing one key at a time.

  - [ ] Use StickyKeys
  - Settings...
Device-Level Techniques

- Pointing Filters
Device-Level Techniques

Pointing Filters

Mouse cursor → Filter → Filtered cursor
Device-Level Techniques

Pointing Filters

Smoothing Filter
Device-Level Techniques

Pointing Filters

Inertia Filter
Device-Level Techniques

Pointing Filters

- Grids
- 2nd Order Control
- Mouse Macros
- etc.
Device-Level Techniques

Mnemonic Rendering

[Bezerianos, Dragicevic & Balakrishnan 2006]
Change Blindness Demo

See, e.g., [Rensink 2002]
Change Blindness Demo

See, e.g., [Rensink 2002]
Digital Displays

- Increase of dynamic visual information
- Often hidden or unseen
- Visual changes or transitions lost
Dealing with Change

- Structuration of visual information
- Histories
- Notification
- Situation Awareness designs [Banbury 04, Endsley 03]
- Groupware designs [Tam 05]

- All Application-Dependent
Pixel Visibility

Unshown

Unseen
Mnemonic Rendering

Buffering
Mnemonic Rendering

Flashback

Persistence
Mnemonic Rendering

Flashback

Persistence
Mnemonic Rendering

- On a macroscopic scale:
Detecting Visibility

- Pixels not shown

→ Software detection
Detecting Visibility

- Pixels not shown
- Pixels not seen
  → Activity detection
  → Head/eye tracking

Mnemonic Wall
Mnemonic Rendering: An Image-Based Approach for Exposing Hidden Changes in Dynamic Displays

A. Bezerianos, P. Dragicevic, R. Balakrishnan
Dynamic Graphics Project Lab
University of Toronto
www.dgp.toronto.edu
Preliminary Testings

- On a layout matching game
  - 4 participants for Mnemonic Desktop
  - 6 participants for Mnemonic Wall
Preliminary Testings

- Participants used restitutions
- Mnemonic rendering preferred to side by side comparison

- Persistence
  - ↓ Bad at explaining changes
  - ↑ Provided peripheral awareness in Mnemonic Wall

- Flashback & Combination
  - Both equally preferred in Mnemonic Desktop
  - Combination preferred by Mnemonic Wall users
Mnemonic Rendering

- No transient information is lost
- No memorization and comparison
- Application and data agnostic
- Implemented at the window manager level
- Use of visibility
- Implicit interaction
Phosphor vs. Mnemonic Rendering

- Iconic illustrations of changes

*Phosphor* [Baudisch et al. 2006]
Phosphor vs. Mnemonic Rendering

- Showing Widget Changes

**Phosphor** [Baudisch et al. 2006]  

**Mnemonic Rendering**
Conclusion

- Context-Specific Design $\rightarrow$ Usability
- Generic Design $\rightarrow$ Utility

- Application-Agnostic Techniques
  - Window-Level
  - Widget-Level
  - Device-Level
Conclusion

We only covered one aspect of generic design
Conclusion

The Quality/Generality trade-off

- Better quality
- Wider applicability
Conclusion

Thank you.

More info:
http://www.dgp.toronto.edu/~dragice
Device-Level Techniques

Pointing Filters

Speech-Operated Cursor

Noise...
release
up