A Toolkit for Managing User Attention in Peripheral Displays

Tara Matthews, Anind K. Dey*, Jennifer Mankoff*, Scott Carter, Tye Rattenbury

EECS Department, UC Berkeley
*Intel-Berkeley Research
*currently of HCII, Carnegie Mellon

Problematic Scenario

• Bus schedule
  – Requires a task switch to use
  – Doesn’t afford continuous monitoring

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BUS</td>
<td>LAB</td>
<td>DEPT</td>
<td>BUS</td>
</tr>
<tr>
<td>6S</td>
<td>601a</td>
<td>616a</td>
<td>605a</td>
</tr>
<tr>
<td>6S</td>
<td>703a</td>
<td>716a</td>
<td>710a</td>
</tr>
<tr>
<td>6S</td>
<td>635a</td>
<td>646a</td>
<td>630a</td>
</tr>
<tr>
<td>6S</td>
<td>650a</td>
<td>665a</td>
<td>650a</td>
</tr>
<tr>
<td>6S</td>
<td>1035a</td>
<td>1046a</td>
<td>1030a</td>
</tr>
<tr>
<td>6S</td>
<td>1135a</td>
<td>1146a</td>
<td>1130a</td>
</tr>
<tr>
<td>6S</td>
<td>1235p</td>
<td>1246p</td>
<td>1230p</td>
</tr>
<tr>
<td>6S</td>
<td>1335p</td>
<td>1346p</td>
<td>1330p</td>
</tr>
</tbody>
</table>

Solution: Peripheral Displays

• Provide awareness with min attention
• Separate from primary task
• Bus LED Display
  + Get bus info at a glance
  + Dynamically manages attention it attracts
  – Hard to accomplish

Why is creating PDs hard?

• Need to abstract info to be glance-able
• Need mechanisms for dynamically managing attention PDs attract:
  – Deciding attention levels to attract (notification levels)
  – Displaying info appropriately (transitions)
• Our contribution:
  – Peripheral Display Toolkit (PTK) supports these key issues in PD creation
Example PTK Applications

- Remote Activity
  - Social Guitar
  - Audio Monitor
  - Motion Monitor
  - Remote Awareness Display
- Bus Displays
  - Bus Mobile
  - Bus LED
- Instant Messenger Status

Orb showing remote activity

Example PTK Applications

- Remote Activity
  - Social Guitar
  - Audio Monitor
  - Motion Monitor
  - Remote Awareness Display
- Bus Displays
  - Bus Mobile
  - Bus LED
- Instant Messenger Status

Orb showing remote activity
Example PTK Applications
+ Manage attention with abstraction, notification levels, & transitions
+ Glance-able
+ Few lines of code
+ Simplified code design & code re-use

Simple Example: Motion Monitor
- no/low motion
- medium motion
- high motion

Outline
- PTK: Motivation & Examples
- Managing attention
  – Abstraction
  – Notification levels
  – Transitions
- Toolkit Architecture & Library
- Conclusion

Managing User Attention
- Attention is a crucial issue in design
  – Significant impact on success
  – More info can be monitored w/o overburdening
- Attention management is difficult
  – Used to designing apps that attract full attention
  – Attention attracted should match info urgency
- Literature surveys led to key characteristics of attention management
Attention Management Characteristics
From survey of peripheral displays:
1. Abstraction
   • enables lower-attention info consumption
2. Notification levels
   • help determine how much attention to attract based on urgency of info
3. Transitions
   • enable display of info to match urgency

Abstraction
• Enables lower-attention consumption of info
• Extracting features or reducing fidelity of info
• Motion Monitor: replaces video images with general motion amount (high, medium, none)

Notification Levels
• Indicate the urgency of info
  – Low levels = non-urgent info
  – High levels = more urgent info
• Motion Monitor: doesn’t show urgent info, so uses low notification levels

More on Notification Levels
• Attention is (roughly) scalar – PD designers must decide where on the scale displayed info should fall
More on Notification Levels

- **Attention Type**
  1. None
  2. Inattention

- **Notification Level**
  1. Ignore
  2. Change blind
More on Notification Levels

Attention Type
1. None
2. Inattention
3. Divided
4. Focused

Notification Level
1. Ignore
2. Change blind
3. Make aware
4. Interrupt

0% 100% attention awareness unconscious
More on Notification Levels

<table>
<thead>
<tr>
<th>Attention Type</th>
<th>Notification Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Ignore</td>
</tr>
<tr>
<td>Inattention</td>
<td>Change blind</td>
</tr>
<tr>
<td>Divided</td>
<td>Make aware</td>
</tr>
<tr>
<td>Focused</td>
<td>interrupt</td>
</tr>
<tr>
<td>Focused + interaction</td>
<td>Demand action</td>
</tr>
</tbody>
</table>

Transitions

- Enable display of info to match urgency
- Depend on notification level
  - Ignore: no changes
  - Change blind: fading, slow motion, tiny updates
  - Make aware: discrete, abrupt updates
  - Interrupt: flashing, beeping, vibration
  - Demand action: interrupt until user does action
- Motion Monitor: varies pulse speeds for different notification levels

Outline

- PTK: Motivation & Examples
- Managing attention
  - Abstraction
  - Notification levels
  - Transitions
- Toolkit Architecture & Library
- Conclusion

PTK Architecture

1. Support for managing impact on human attention using abstraction, notification levels, and transitions
2. Simplified code design and code re-use
3. Library of common PD components
Motion Monitor Architecture

Input-Side

Camera Input → PTK Discovery Server → Motion Monitor → Notification Map → Transl. Orb Widget

Output-Side

Motion Monitor → Notification Map → Transl. Orb Widget

notification level = change blind
Motion Monitor Architecture

Input-Side
- Camera Input
- Discovery Server
- PTK Discovery Server

Output-Side
- Motion Monitor
- Notification Map
- Trans
- Orb Widget

Motion Monitor = medium
Notification level = change blind

Remote Awareness Display

- Count # people, How much motion?
- Phone ring? Voices?

Motion Monitor → Remote Awareness Display

Input-Side
- Camera Input
- Discovery Server

Output-Side
- Motion Monitor
- Exact Match (Motion)
- Trans
- Orb Widget
**Motion Monitor → Remote Awareness Display**

- **Input-Side**
  - Camera Input
  - Audio Input
- **Discovery Server**
  - Motion Monitor
  - # People Abstractor
- **Output-Side**
  - PTK Discovery Server

**Remote Awareness Display**

- **Input-Side**
  - Camera Input
  - Audio Input
- **Discovery Server**
  - Motion Monitor
  - # People Abstractor
- **Output-Side**
  - PTK Discovery Server

54 – total lines of code

**General Architecture Diagram**

- **Input-Side**
  - Input 1
  - Input 2
  - Input N
- **Discovery Server**
  - PTK Discovery Server
  - Peripheral Display 1
  - Peripheral Display 2
- **Output-Side**
  - Motion Monitor
  - # People Abstractor

**Library Components**

- **Input**
  - audio, camera, Phidgets, Context Toolkit, online calendars, news, stocks, weather, Web page parser, serial port communication
**Library Components**

- **Input**
  - audio, camera, Phidgets, Context Toolkit, online calendars, news, stocks, weather, Web page parser, serial port communication
- **Output**
  - ticker text, Ambient Orb, Phidgets

- **Abstractors**
  - motion, people counting, voices, phone ringing

- **Notification**
  - exact match, threshold, contains, degree of change

**Transitions**

- PTK provides architectural support for implementing as animations
  - Timing
  - Number of steps
  - Flashing (single, repeated)
  - Tied to output
    - Depend on output modality
    - Example transitions: color changing and pulsing (orb); fading, motion, & flashing (ticker text)
Future Work & Conclusion

- PTK supports the creation of PDs
- Focuses on managing user attention
- To do this, uses abstraction, notification levels, transitions
- Validated with 7 example applications
  - Supports attention management
  - Facilitates code re-use
- Future work
  - Incorporate interruptibility and local context sensing
  - Cross-modality animations
  - PD evaluation support

Questions?

- For more information & downloads:
  www.cs.berkeley.edu/projects/io/ptk/
  tmatthew@cs.berkeley.edu

Thanks to...
NSF, Ed De Guzman, Jeff Heer, Scott Klemmer, Berkeley & CMU

What about user context?

- PDs must 2 sets of attention issues:
  - User context
    - interruptibility, primary activity, focus of attention
  - Attention management
    - balancing info urgency and user attention
- PTK focuses on attention management
  - User context invasive to sense, not practical to assume PDs have access
  - Management issues are independent of user context
- Context input can be considered on a library level (in notification maps)

Related Work

- AROMA: architectural support for abstraction
- Context Toolkit: infrastructure for context input and some abstraction
- Phidgets: library of hardware widgets for input and output
- Real World Interfaces Toolkit: library of X10 output widgets
- iStuff: infrastructure for distributed inputs and outputs