Designing Interactive Systems 2

Lecture 1: Introduction, History, Design Space of Input Devices

Prof. Dr. Jan Borchers
Media Computing Group
RWTH Aachen University

hci.rwth-aachen.de/dis2
Class Syllabus

• Part 1
  Key Concepts

• Part 2
  Usage and Design of UI Toolkits and Design Systems

• Part 3
  UIs Beyond the Desktop

• Part 4
  Prototyping Process
Administrivia

• Format: V3/Ü2

• 6 Credit points

• Class times
  • Lecture on Wednesdays (8:30 — 11:00), Room 2222
  • Lab on Mondays (14:30 — 16:00), Room 2222
Team

Prof. Dr. Jan Borchers
hueber@cs.rwth-aachen.de
E-Mail Subject: [DIS2]

Sebastian Hueber

Anke Brocker
brocker@cs.rwth-aachen.de
E-Mail Subject: [DIS2]
Your Final Grade

35% Final Exam (60 min)
- July, 18th
- or August, 9th

25% Midterm Exam (60 min)
- May, 17th

40% Weekly Assignments

5%
Weekly Assignments

• We have a strict grading policy:

  • **Late submissions** will be graded 5.0 without feedback
  
  • **Team size** is 2 students (3 only by permission). If you hand in a solution without a team partner: 5.0 without feedback

  • If your code does **not compile**: 5.0 without feedback

• For some assignments you will need a **Mac**

  • No Mac? Visit http://www-rbi.informatik.rwth-aachen.de/Poo1+Helpdesk/

• Submission via **Moodle**
Website

- All information about this course can be found online
- hci.rwth-aachen.de/dis2
How DIS1 and DIS2 Cover HCI

**Use and Context**
- U1 Social Organization and Work
- U2 Application Areas
- U3 Human-Machine Fit and Adaptation

**Human**
- H1 Human Information Processing
- H2 Language, Communication and Interaction
- H3 Ergonomics

**Computer**
- C1 Input and Output Devices
- C2 Dialogue Techniques
- C3 Dialogue Genre
- C4 Computer Graphics
- C5 Dialogue Architecture

**Development Process**
- D1 Design Approaches
- D2 Implementation Techniques and Tools
- D3 Evaluation Techniques
- D4 Example Systems and Case Studies

**DIS 1**

**DIS 2**
DIA Cycle

Design

Implement

Analyze
CHAPTER 1

History of User Interface

Programming Paradigms
Batch Processing

- Prepare data on punch cards
- Wait for result as printout offline
**Time-sharing Systems**

- Command-line based interaction
- Shorter turnaround (per-line)
Full-screen textual UIs

- Turnaround per character
- Interaction starts to feel “real-time”
Menu-based Systems

• Discover functionalities instead of memorizing them

• **Threading** becomes important
Graphical User Interface

- Event-based program structure
- Pointing devices in addition to keyboard
CHAPTER 2
Design Space of Input Devices
Design Space of Input Devices

- Card, Mackinlay, Robertson 1991

- Categorization of input devices according to physical, mechanical and spatial properties

- Why?
  - Compare input devices
  - Identify new input modalities
Movement Primitives

Input device \( : = \langle M, \text{In}, S, R, \text{Out}, W \rangle \)
Example

Prof. Dr. Jan Borchers: Designing Interactive Systems 2

19
Compositions

- Merge
- Layout
- Connect
In-Class Exercise

• Plot out the input capabilities of the Ferrari Racing Controller on the Card Design Space of Input Devices.

• The controller consists of a **steering wheel** with 8 **buttons** and a **rotary switch** with 5 states, as well as 2 **pedals**.

• Assume that the steering wheel can only have one full rotation.
## In-Class Exercise

### Designing Interactive Systems 2

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Rotar</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>P</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>dP</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dF</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Inf</td>
</tr>
</tbody>
</table>

- **8 buttons with 2 states each**
- **2 pedals with an infinite number of states each**
- **1 switch with 5 states**
- **1 steering wheel with an infinite number of states**

---

**Layout**

- Buttons (8 in total)
- Pedals
- Rotary switch (5 states)
- Steering wheel
Is This Space Complete?
Testing Points

- **Expressiveness** describes how precisely the meaning is conveyed.

- For input devices, expressiveness suffers if $|In| \neq |Out|$
  
  - $|In| < |Out|$: Cannot specify all legal values
  
  - $|In| > |Out|$: Can specify illegal values
Testing Points

- **Effectiveness** describes how well the intention can be communicated

<table>
<thead>
<tr>
<th>Linear</th>
<th>Rotary</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>rX</td>
</tr>
<tr>
<td>Y</td>
<td>rY</td>
</tr>
<tr>
<td>Z</td>
<td>rZ</td>
</tr>
</tbody>
</table>

- **Linear Devices**
  - Light Pen
  - Touch Panel
  - Tablet
  - Mouse

- **Rotary Devices**
  - Absolute Joystick
  - Trackball
  - Headmouse

<table>
<thead>
<tr>
<th>P</th>
<th>dP</th>
<th>dF</th>
<th>F</th>
<th>R</th>
<th>dR</th>
<th>dT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>10</td>
<td>100</td>
<td>Inf</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>100</td>
<td>Inf</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100</td>
<td>Inf</td>
<td>1</td>
<td>100</td>
<td>Inf</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inf</td>
<td>1</td>
<td>100</td>
<td>Inf</td>
<td>1</td>
</tr>
</tbody>
</table>

---

Prof. Dr. Jan Borchers: Designing Interactive Systems 2
“Will I Get a Seat in This Class?”
CHAPTER 3

Window System Architecture
Window Systems: Basic Tasks

• **Input handling**
  Pass user input to appropriate application

• **Output handling**
  Visualize application output in windows

• **Window management**
  Manage and provide user controls for windows
Window Systems: Requirements

• **Independent** of hardware and operating system

• No noticeable **delays** (few ms) for basic operations, e.g. moving window, redrawing cursor

• **Customizable** look&feel for user preferences

• Input & Output in **parallel**

• **Multimedia** support: Graphics, audio, …

• Support for various **input devices** and modalities
Window Systems: Evaluation Criteria

- **Availability**
  Platforms supported

- **Productivity**
  For application development

- **Parallelism**
  External and internal

- **Performance**
  Usage of resources and latency

- **Graphics model**
  RasterOp vs. vector

- **Appearance**
  Look & Feel, exchangeable?
Window Systems: Evaluation Criteria

- **Extensibility**
  In source code or at runtime

- **Adaptability**
  Localization and customization at runtime

- **Resource sharing**
  E.g., fonts

- **Distribution**
  Over network

- **API**
  Structure and comfort

- **Independence**
  Of application and interaction logic inside programs written for the WS

- **Inter-Application Communication**
  Copy & Paste, Drag & Drop
Window Systems: Conflict

App developer

WS developer

User
Window System Architecture

Apps
User Interface Toolkit
Window Manager
Base Window System
Graphics & Event Library
Hardware

More abstract, user-oriented