Designing Interactive Systems II

Computer Science Graduate Programme SS 2010

Prof. Dr. Jan Borchers
RWTH Aachen University

http://hci.rwth-aachen.de
Today

- Class syllabus
- About our group
- Device technology
Administrivia

- New format: V3/Ü2
- Lecture: Wednesday, 9:00–12:00
- Lab: Monday, 15:30–17:00
- 6 credit points (8 with additional work if needed)
- Final grade:
  - 20% weekly assignments  25% midterm exam
  - 20% final project  35% final exam
- Requires MPO 2010
- Lecture recordings on iTunes U
DIS II Topics: The 1-Page Overview

• Central question:
  How do interactive systems work?
• Device technology
• Window systems
  • Own, existing
• Mobile and Physical Computing
• Interactive Web & Multimedia Systems
• Lab: From X to Mac OS X, Prototyping Environments
The Syllabus In Detail

• What makes a UI tick?
• Technical concepts, software paradigms and technologies behind HCI and user interface development
• Part I: Key concepts of UI systems
  • Window System Architecture Model
• Part II: Comparing seminal window systems
  • Mac, X11, AWT/Swing, Windows, NeXT/OS X, iPhoneOS,…
  • Paradigms & problems, designing future UI systems
  • Overview of UI prototyping tools
The Syllabus In Detail

- Part III: UIs Beyond The Desktop
  - Think beyond today's GUI desktop metaphor
  - UIs for Mobile, Haptics, Physical Computing, Ubicomp, Multimedia
- The Lab
  - Part I: Implementing Simple Reference Window System
  - Part II: Development using several existing GUI toolkits (such as Java/Swing, Interface Builder)
  - Part III: Working with iPhone, Quartz Composer, Arduino, etc.
Lab

- Register for the class in CAMPUS
- Join the L2P class room
- Register as iPhone Developer
  - developer.apple.com/iPhone
- lab sessions on Mondays
DIS 2 Team @ media computing group

• Prof. Dr. Jan Borchers
  • B.Sc. & M.Sc. CS, U Karlsruhe & U London
  • Ph.D. CS, U Linz, U Ulm & TU Darmstadt
  • Assist. Prof. at Stanford & ETH Zurich
  • Full Prof. of CS, RWTH Aachen Univ.
  • Research area:
    Interaction Design for New Media
DIS 2 Team @ media computing group

• Dipl.-Inform. Moritz Wittenhagen
  • Diplom, RWTH Aachen
  • Video Navigation
  • wittenhagen@cs.rwth-aachen.de

• Dipl.-Inform. Florian Heller
  • Diplom, RWTH Aachen
  • Virtual Audio Spaces
  • flo@cs.rwth-aachen.de

• Chatchavan Wacharamanotham, M.Sc.
  • M.Sc., RWTH Aachen
  • Collaboration in meeting, HCI for seniors
  • chat@cs.rwth-aachen.de
How DIS I and DIS II 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

DIS I

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

DIS II

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

DIS I

DIS II

ACM SIGCHI 1992
Some Core Aspects of DIS I Reviewed

• The question developers should be asking (but often forget):
  Not HOW, but WHAT to Design

• Technical viewpoint:
  “How do I build this?”
  • Easy to focus on for us CS folks
  • Important, but do not overlook…:

• User’s viewpoint:
  “What does it do for me?”
  • An excellent system that nobody needs is useless!
Approach: Iterative Design—the DIA Cycle

Forget the waterfall model!

Design

Prototype/Implement

Analyze/Test/Evaluate
Prototyping & Testing

- D: Brainstorm/develop initial project idea
- I: Sketch scenario/storyboard of idea at work
- A: Ask real people (≠ students) about it
  interviews, questionnaire
- D: Rework your feature set & user experience
- I: Paper prototype (crude for a reason)
  Scenario, Storyboard, Post-It Prototype, …
- A: Have users use it to accomplish something
Prototyping & Testing

• D: Refine your feature set and user experience
  Improve information and interaction design

• I: Interactive prototype
  Director, Java, Visual Basic, ...
  to throw away or keep
  Limiting features: Vertical, horizontal, storyboard

• A: Have users “use” it to accomplish a goal
  Intro, atmosphere, tasks, observation, interview, note-taking,...
Prototyping & Testing

- Look for Style Guides for your development environment
  - Macintosh HI Guidelines, CUA, Motif Style Guide,…
- Check your system against “Golden Rules of Interface Design”
  - E.g., Ben Shneiderman: Designing the User Interface
  - Simple, Consistent, Language, Feedback, Errors, Exits, Memory, Help, Shortcuts,…
A Brief History of User Interfaces

(Done in DIS I to understand the new interaction metaphors, reviewed here to understand the new programming paradigms)

- Batch-processing
  - No interactive capabilities
  - All user input specified in advance (punch cards, ...)
  - All system output collected at end of program run (printouts, ...)
  - Applications have no user interface component distinguishable from File I/O
  - Job Control Languages (example: IBM3090–JCL, anyone?): specify job and parameters
A Brief History of User Interfaces

• Command-Line Systems
  • Command-line based interaction with simple terminal
  • Shorter turnaround (per-line), but similar program structure
  • Applications read arguments from the command line, return results
  • Example: still visible in Unix commands

• Full-screen textual interfaces
  • Shorter turnaround (per-character)
  • Interaction starts to feel “real-time” (e.g. vi)
  • Applications receive UI input and react immediately in main “loop” (threading becomes important)
A Brief History of User Interfaces

- Menu-based systems
  - Discover “Read & Select” over “Memorize & Type” advantage
  - Still text-based!
  - Example: VisiCalc
  - → Applications have explicit UI component
  - But: choices are limited to a particular menu item at a time (hierarchical selection)
  - → Application still “in control”
A Brief History of User Interfaces

- **Graphical User Interface Systems**
  - From character generator to bitmap display (Alto/Star/Lisa..)
  - Pointing devices in addition to keyboard
  - → Event-based program structure
    - Most dramatic paradigm shift for application development
    - User is “in control”
    - Application only reacts to user (or system) events
    - Callback paradigm
- **Event handling**
  - Initially application-explicit
  - Later system-implicit
Design Space of Input Devices

• Card, Mackinlay, Robertson 1991
• Goal: Understand input device design space
  • Insight in space, grouping, performance reasoning, new design ideas
• Idea: Characterize input devices according to physical/mechanical/spatial properties
• Morphological approach
  • device designs = points in parameterized design space
  • combine primitive moves and composition operators
Primitive Movements

- Input device maps physical world to application logic
- Input device := \langle M, In, S, R, Out, W \rangle
  - Manipulation operator
  - Input domain
  - Device State
  - Resolution function In->Out
  - Output domain
  - Additional work properties

\begin{table}[h]
\begin{tabular}{|c|c|}
\hline
P, dP & R, dR \\
\hline
F, dF & T, dT \\
\hline
\end{tabular}
\end{table}
Radio Example

Manipulation
Input
State
Resolution fn.
Output
Works

Application

- $R_z \in [0, 270]$ dB
- $l(r) \in [0, 270]$
- $s(r) \in [0, 45, 90]$
- $dR_z$ Real
- $I(r)$ Real
- $P_x \in [0, 5]$ Hz

Jan Borchers
Composition

• Merge
  • Result = Cartesian product
  • E.g., mouse coordinates: $X \oplus Y = \{(x, y)\}$

• Layout
  • Spatial collocation
  • E.g., mouse (x, y) & buttons
  • How different from merge?

• Connect
  • Chaining
  • E.g., mouse output & cursor
  • Virtual devices
Complete space := {all possible combinations of primitives and composition operators}

Mouse = one point!
In-Class Group Exercise: SpaceBall

- Place the SpaceBall into the design space
  - Ball mounted on a plate with 12 buttons
  - Detects precise amount of pushing and twisting in all directions without moving
  - Auto-zeroes physically
Is This Space Complete?

- No – it focuses on mechanical movement
  - Voice
  - Other senses (touch, smell, ...)
- But: Already proposes new devices
  - Put circles into the diagram and connect them
Testing Points

• Evaluate mappings according to
  • Expressiveness (conveys meaning exactly)
  • Effectiveness (felicity)
• Visual displays easily express unintended meanings
• For input devices, expressiveness suffers if $|\text{In}| \neq |\text{Out}|$
  • $|\text{In}| < |\text{Out}|$: Cannot specify all legal values
  • $|\text{In}| > |\text{Out}|$: Can specify illegal values
Effectiveness

• How well can the intention be communicated?
• Various figures of merit possible
  • Performance-related
    - Device bandwidth (influences time to select target, ergonomics and cognitive load)
    - Precision
    - Error (% missed, final distance, statistical derivatives)
    - Learning time
    - Mounting / grasping time
  • Pragmatic
    - Device footprint, subjective preferences, cost,...
Example: Device Footprint

- Circle size := device footprint
  - Black: with 12" monitor
  - White: with 19" monitor

- What do we see?
  - Tablet, mouse expensive
  - Worse with larger displays

- But:
  - Mouse Acceleration alleviates this (model of C:D ratio?)
  - Higher resolution mice
Assignments

• Register in CAMPUS by Monday 12:00
• For next class, read:
• See the L2P course room for all materials