Mobile Application Development (Design and)

2nd class

Prof. Stephen Intille
s.intille@neu.edu
Administrivia

• Anyone new?
• Are you checking email in Banner?
• Read the syllabus! (and check back regularly...)
• Readings remainder of the week...
• Wiki:
  – Fill in participant info
  – Pick a date for presentation
  – Add to the FAQ/Tips document
  – Use discussion to ask/answer questions
  – Setup monitoring so you are notified of changes
Questions on reading

1. What is Fudd’s first law of creativity? (hint: idea)

2. When doing lo-fi prototyping, Rettig recommends that in addition to the user, you have three people: an observer, a facilitator, and a ________.

3. If you are skeptical about lo-fi prototyping, Rettig suggest that you do what?
Technical Q&A

- Android SDK version to use
- Environment: Windows, MacOS, Linux
- Porting Java to Blackberry
Project Q&A

• Plug-in for the CITY project
  – Anything that might plausibly help with weight loss, related to diet, exercise, goal setting, stress, communication, etc. Think broadly within the domain.

• Innovative use of motion sensor for encouraging physical activity or less sedentary behavior
  – Includes fitness, exergames, goal setting, ...
  – Have HR monitors as well as Wockets
Assignment Q&A: Due tomorrow

• First design exercise:
  – Hot (and not) critical analysis of mobile health apps
    [link](http://www.ccs.neu.edu/home/intille/teaching/MAD/Design1.htm)

• Team forming:
  – You’ll hear from me via email based on information on the wiki list
Short term plan

• Today
  – Finish up on yesterday’s intro to context-sensitive apps
  – Intro to paper prototyping
  – General introduction to Android

• Tomorrow
  – Looking at code/tools. Putting an app on the Android Market. Q&A.

• Sunday – HelloMAD app due
Primary design challenges

• Short bursty interactions
• User expectations for simplicity
• Interruptions!
• Limited input modality
• Data reliability (and multiple points of desired data access)
• Standing out in a crowd
• Aggressive operating systems
• In the future: security/privacy
• **Cost of data transmission (can be $$$)**
Challenge #1: OS limitations

• Need a phone app to run continuously
  – Some disallow completely (iPhone, WM7)
  – Some allow but aggressively shut off (Android)

• My prediction: market forces will solve this
Challenge #2: Battery life

- Continuous operation of high-sampling sensors (e.g. motion) drains battery
- Phones more efficient, but also have new features, so get status quo
- Impact
  - Critical usability concern for apps
  - Older phones: severe; newer better
Challenge #2: Battery life

• Prediction
  – No silver bullet ... Always a major consideration
  – Workable with newer phones and careful design
  – Does require careful engineering
    • Phones not designed for continuous monitoring
    • Sensors/operating system interaction may create tricky situations
Challenge #3: Expectations

• Design must set expectation that app not perfect
• Impact
  – Example from StepLively:
    Not calories, not steps
    Vague but understandable “points”

• Prediction: requires careful design
Challenge #4: Normal phone use

• Advantage is leveraging phone people have, but must not disrupt that behavior

• Impact
  – StepLively: ≈50%+ of people ... Viable long term?
  – Wockets: Changed entire strategy (dropped original plan to use phone’s motion sensor)

• Prediction: only very few “killer apps” will change how people use/carry phones
Challenge #5: Correction/credit

- People dissatisfied if app misses activity and can’t get “credit”
  - Because of sensor/algorithm error
  - Because person forgot or miscarried devices

- Impact
  - Steplively: Easy (but approximate) way to fix graph
  - Wockets: Text messaging feedback

- Prediction: design challenge
Challenge #6: Behavior

• Deploy apps and see “odd” behavior in data. Tease apart:
  – Bug in code?
  – Subjective reporting errors?
  – Unanticipated/unusual behaviors

• Impact: Need tools to understand

• Prediction: Iterative design using data from remote monitoring tool will increase chances of success
Challenge #7: Evaluation

• Need relatively long term deployments to make convincing case for...
  – Engagement
  – Impact beyond novelty stage
  – Small changes leading to large sustainable changes and health impact

• Most apps tried then dropped. Focus on sustainable use.
Take away #1

• Mobile phones are increasingly capable of sophisticated, real-time information processing using internal sensors

• Most people will have this technology and carry it with them nearly everywhere
Take away #2

- Haven’t yet scratched the surface of behavior-change potential of using accelerometers, GPS, and external sensors that wirelessly communicate with phones + pattern recognition to wireless networks (e.g., Wockets).

- Key opportunity: “just-in-time” tailored feedback base on behavior.
Take away #3

• There are some challenges. Examples:
  – Dealing with noisy sensors to automatically detect certain types of physical behaviors
  – Unanticipated and variable end-user behavior that impacts user interface system design
  – Power-management issues on mobile devices
  – Effective remote management and interpretation of data and subject behavior as a study is running
Take away #4

• There are some design and technical challenges, but they can be overcome with creative approaches.

• Mobile devices with real-time feedback create novel (and engaging?) options that can’t be achieved without the technology ... Should be explored!
So we have our domain...
Now what?

1. Start thinking about design

2. Start learning Android
Typical user interface design

- Observation
- Model tasks
- Simplify/refine/stress-test the task models
- Lo-fidelity prototyping (paper)
- Test in context
- Iterate
- Eventually...
  - High fidelity prototyping
  - Test in context
  - Iterate (entire process)
Observation
Model the tasks

[Diagram showing various task nodes connected with lines: Family Members, Home, Email, Voice Mail, Whiteboards, Fridge Magnets, PDA, Post-it Notes, External, Work, Friends, Unknown, and Users.]
Simplify/refine/stress-test tasks

• Gotchas
  – Missing what’s truly important to user
  – Interruptions
  – Influence of environment/context
  – Boredom/lack of novelty
  – Dealing with problems created by
    • Environment
    • Other people
    • Technological limitations
Paper prototyping
Paper prototyping
Test in context (or try)
Test in context (or try)
High-fidelity prototyping
Iterate!

Central interface

MOBILE MESSAGES for object-oriented messages.
Iteration

At every stage!

Diagram from J. Landay
Benefits of paper prototyping

• Focus on interaction
• Fast, fast, fast and affordable
• Tests BIG IDEAS early
• Teams can participate
• Helps designer address three design truisms:
  – Fudd’s first law of creativity:
    “To get a good idea, get lots of ideas”
  – “Know your user”
  – “You aren’t your user”
Challenges of paper proto.

• Mobile app use involves quick interactions
• Context is critical and hard to simulate
• Slow simulation of interface behavior
• Some graphical user interface behavior is difficult to simulate with paper
• Non-visual cues may be difficult to simulate (audio, tactile)
Formative design goals

• Get through as many design iterations as possible
• Create deadlines for ideas: paper prototype early and often ... don’t wait for perfection
• Worry less about the underlying theory and more about instantiation of that theory in the interface: What is the experience for the user?
Running a session

- Subject
- Facilitator
- Computer
- Observer

- Conduct dry runs with team to debug design first
Facilitator

• Only person speaking
• Hand person explicit, written tasks
• Don’t explain (despite compelling urge)
  – What do you think it will do?
  – What are you thinking right now?
  – What questions are on your mind?
  – Are you confused about what you are seeing?
Computer

- Organizes the paper model
- Efficiently selects parts
- Works on the fly to modify if necessary
Observer

• Takes notes
• Things to note:
  – Hesitation
  – Finger pointing
  – Confusion
• Record 1 per index card
  – Observation
  – Solution (if something comes to mind)
Session organization

• Sessions 1-2 hours
• Schedule 2 hours between sessions for revisions
• 1-2 participants. Friends can be particularly informative.
Set the stage

- Begin with greetings, introductions, and refreshments
- Assure people that the test is confidential, the results will remain anonymous, etc.
- Stress that this is not a test. There are no right answers. They can be most helpful by being completely honest about their impressions.
- People often say things like, “Am I flunking the test? Am I getting it right?” Possible answer:
  - “Don’t worry, the question is whether or not we are flunking. The interface is on trial, not you. If you fail to understand something or can’t complete one of the tasks, that’s a sign of trouble.”
Focus on BIG IDEAS

• Make paper prototype messy
  – Avoid perfection
  – Avoid color
  – Don’t use a ruler

• Show participant change on the fly in first ten minutes
Goal: realistic feedback (despite setting)

- Get to know the person
- Then modify preplanned (realistic) scenarios to fit
- Conduct interview with a friend and ask them: what would s/he really do?
Learn about simplicity

• Listen – never explain
• Listen for the “aha” moments
  – “What do you think it will do?”
  – [I think it will …]
  – “Ok, press it and see what it does.”
  – [Oh, so it does ______]
Interact, don’t ask

• It’s not about showing the interface, it is about using it
• Asking if they like it gets typical responses
  – People say and do different things
  – Sometimes people love something they can’t use and vise versa
• Construct models, not illustrations
Logistical tips

• Photocopier simplifies construction
• Fan-folding a piece of paper can simulate a long, scrollable page
• Transparencies can be helpful
• Assemble construction kit
Construction kit

- White, unlined, heavy paper that is bigger than letter size (11 by 17 inches)
- Index cards for construction material and note taking
- Various adhesives. Tape: clear, colored, double-backed, pin strip tape
- Glue sticks, and most importantly, Post-It glue-a stick
- Rolls of white correction tape
- Various markers-colored pens and pencils, highlighters, and thick markers, pastels
- Lots of sticky note pads of various sires and colors
- Transparencies
- Scissors, x-acto knives, metal straightedges (band aids)
- Other possibilities found in art stores:
  - Rub on texture
  - Modeling clay (if using physical devices)
  - Sound clicker
Tips for mobile paper proto.

• Use a real phone and place paper on screen
• Use font guide so you are realistic about font sizes
• Consider the reading glasses challenge
• Avoid components that need a stylus (use finger-sized buttons)
• Minimize text entry ... painful
• Screens should be understood at a glance
Challenges in project domain

• Physiology
  – Delayed gratification (exercise, dietary control)
  – Instant gratification (tempting food)

• Timeline
  – Novelty ... how to exploit, not suffer, from it?
  – When content is required, how can it be kept fresh without requiring a huge amount of manual generation?
1. Start thinking about design
   **Fudd + Paper Prototyping = Simplicity, Usability & Success!**

2. Start learning Android
Android basics: what is it?

• Open source OS and development platform
  – In theory, you can change anything
  – In practice....

• Hardware reference design

• Linux OS kernel

• Open-source libraries for app development
  – E.g., SQLite, Webkit, OpenGL, media manager
Android basics: what is it?

• Dalvik virtual machine
  – Not Java ME
  – Use Java, but Dalvik compiles for small and efficient and multiple VMs

• Application framework
  – Window manager, location manager, telephony, sensors

• UI framework to run and launch apps

• Preinstalled apps
Android basics: what is it?

• SDK and tools
• Wild west of app stores: the Market
System architecture

- Applications
  - Home
  - Contacts
  - Phone
  - Browser
  - ...

- Application Framework
  - Activity Manager
  - Window Manager
  - Content Providers
  - View System
  - Package Manager
  - Telephony Manager
  - Resource Manager
  - Location Manager
  - Notification Manager

- Libraries
  - Surface Manager
  - Media Framework
  - SQLite
  - OpenGL ES
  - FreeType
  - WebKit
  - SGL
  - SSL
  - libc

- Android Runtime
  - Core Libraries
  - Dalvik Virtual Machine

- Linux Kernel
  - Display Driver
  - Camera Driver
  - Flash Memory Driver
  - Binder (IPC) Driver
  - Keypad Driver
  - WiFi Driver
  - Audio Drivers
  - Power Management
Notables

- “Open philosophy”
- Application framework that encourages reuse of application components
- Access to (much) hardware (sometimes even without bugs!)
- Release and fix mentality (or it seems like it, sometimes)
- Great standard apps: Google Maps, location services, quick search box
- Background services
Notables

• SQLite (with app data sandboxed)
• Shared data and interprocess communication
  – Notifications (via UI)
  – Intents
  – Content providers (managed access to app private data)
• All apps created equal
• Widgets and livewallpaper and livefolders
Notables

• 2d canvas drawing
• 3d OpenGL
• Support for popular media formats: MPEG4, H.264, AAC, MP3, JPG, PNG, GIF
• Native Development Kit (NDK) (C++ tinkering under the hood)
• OS optimization of memory and process management
New way of thinking #1

• Expect...
  – Limited processing power
  – Limited RAM
  – Limited permanent storage capacity
  – Small screen and low resolution
  – High cost of data transfer
  – Slow data transfer rates with high latency
  – Unreliable data connections

• Moore’s law less impactful...
New way of thinking #2

• Application framework that encourages reuse of application components
• “Screens” are Activities that are chained with lightweight exchange of data between them
• OS can handle stack of Activities if you want it to (e.g., back)
New way of thinking #3

• OS Manages process lifetime *(app assassin)*
  – App responsiveness
  – Setting priority to interaction

• You MUST
  – Ensure that your app is ready for swift death
  – Yet, it must remain response and/or restart in the background
  – Must come to the foreground quickly
**Types of applications**

- **Foreground**
  - Useful when being used.
  - Suspended otherwise
- **Background**
  - Apart from when being configured, spends most of lifetime hidden (e.g., call screening app)
- **Intermittent**
  - Some interaction but mostly in the background (e.g., media player)
- **Widget**
  - Home screen status update
Good behavior

- Is well behaved
- Switches seamlessly from background to foreground
- Is polite (e.g., stealing focus)
- Presents a consistent user interface
- Is responsive
Behavior police: process assassin

- Two conditions monitored
  - Must respond to any user action (e.g., key press) within 5s
  - A BroadcastReceiver must return from its OnReceive handler within 10s

- Worst case, not goal! Users notice .5s