Sketching and Prototyping

SWE 632
Fall 2021
Administrivia

- HW2 due today
- HW3 due next week
- Midterm exam in 2 weeks (midterm review next week)
  - Covers all lectures & readings before exam
Iterative Model of User-Centered Design

**Observation**
(Re)Define the Problem
Understand User Needs

**Idea Generation**
Brainstorm what to build

**Prototype**
Build

**Test**
Evaluate what you have built
Iterative Model of User-Centered Design

Idea Generation
Brainstorm what to build

Prototype
Build
Sketching & Storyboards
How do You Brainstorm?
What is a Sketch?

“A conversation between the sketcher or designer and the artifact”
Why Sketch?

- Sketching offers **visual** medium for exploration, offering cognitive scaffolding to externalize cognition.
Being Creative with Sketches

• How do you come up with a great idea?
  • Generate lots of ideas
  • Work through ideas through externalization in sketch
  • Critique the ideas
  • Refine them to make them better

• Sketching offers a low-cost medium for working with early ideas before committing to one

• Design is process of creation & exploration
Sketching vs. Prototyping

Buxton Design Exploration
Sketches

- For design
- Getting the right design
- Experimenting, exploring, being creative
- Goal: Support ideation to find a great design solution

Low-Fidelity Design
Refinement Prototypes

- For UX engineering
- Getting the design right
- Following the UX process
- Goal: Support iterative refinement of a given design
Physical Sketches

• Production tools for sketching:
  • whiteboards, blackboards, cork boards, flip chart easels
  • post it notes
  • duct tape, scotch tape, push pins, staples
  • marking pens, crayons, spray paint
  • scissors, hobby knives, foam core board
  • duct tape
  • bits of cloth, rubber
The Space Remembers

• Covering walls, whiteboards, etc. w/ materials is extremely useful
• Provides fast access for revisiting and remixing old ideas
• Facilitates group discussion of designs
Sketches are Sketchy

- Not mechanically correct and perfectly straight lines
- *Freehand*, open gestures
- Strokes may miss connections
- Resolution & detail **low** enough to suggest it is concept
- Deliberately *ambiguous* & abstract, leaving “holes” for imagination
Rules for Sketching

- Everyone can sketch; you do not have to be artistic
- Most ideas conveyed more effectively with sketch than words.
- Sketches are quick and inexpensive to create; do not inhibit early exploration
- Sketches are disposable; no investment in sketch itself
- Sketches are timely; made in-the-moment, just-in-time
- Sketches are plentiful; entertain large # of ideas w/ multiple sketches of each
Sketches Include Annotations

- Annotations explain what is going on in each part of sketch & how
Sketches part of design exploration

B. Buxton. Sketching User Experiences.

K. Moran, ReDraw Project Sketch
Sketches part of design exploration

Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps

Kevin Moran, Member, IEEE, Carlos Bernal-Cárdenas, Student Member, IEEE, Michael Curcio, Student Member, IEEE; Richard Bonett, Student Member, IEEE, and Denys Poshyvanyk, Member, IEEE

Abstract—It is common practice for developers of user-facing software to transform a mock-up of a graphical user interface (GUI) into code. This process takes place both at an application’s inception and in an evolutionary posture as GUI changes keep pace with evolving features. Unfortunately, this practice is challenging and time-consuming. In this paper, we present an approach that automates this process by enabling automatic prototyping of GUIs via three main steps: detection, classification, and assembly. That is, input components of a GUI are abstracted from a mock-up artist using either computer vision techniques or mock-up analysts. Then, software repositories mining, automated dynamic analysis, and deep convolutional neural networks are utilized to accurately classify GUI components into domain-specific types (e.g., toggle buttons). Finally, a data-driven, K-nearest neighbors algorithm generates a suitable hierarchical GUI structure from which a prototype application can be automatically assembled. We implemented this approach for Android in a simple framework that mitigates the complexity of building a robust toolset. The toolset enables prototype applications that closely mirror mock-ups in terms of visual fidelity while exhibiting reasonable code structure. Important interviews with industrial practitioners illustrate the potential to improve not only development workflows.

Index Terms—GUI, GUI Mining, Prototyping, Machine Learning, Mining Software Repositories.

1 INTRODUCTION

Most modern user-facing software applications are GUI-driven, and rely on attractive user interfaces (UI) and intuitive user experiences (UX) to attract customers, facilitate the effective completion of computing tasks, and engage users. Software with cumbersome or aesthetically displeasing UIs are far less likely to succeed, particularly as companies look to differentiate their applications from competitors with similar functionality. This phenomenon can be readily observed in mobile application marketplaces such as the App Store [1] or Google Play [2], where many competing applications also known as apps) offering similar functionality (e.g., task managers, weather apps) largely distinguish themselves via UX/UI [3]. Thus, an important step to developing any GUI-based application is drafting and prototyping design mock-ups, which facilitates the in-committing to spending development resources implementing them. After these initial design drafts are created it is critical that they are faithfully translated into code in order for the end user to experience the design and user interface in its intended form. This process (which often involves multiple iterations) has been shown by past work and empirical studies to be challenging, time-consuming, and error prone [4], [5], [6], [7], [8], [9], [10] particularly if the design and implementation are carried out by different teams (which is often the case in industrial settings) [10]. Additionally, UX/UI teams often practice iterative design process, where feedback is collected regarding the effectiveness of GUIs at early stages. Using prototypes would be preferred, as more detailed feedback could be collected; however, current prototypes
Fidelity of Sketches & Mockups

- **Storyboard**: low fidelity (many details left unspecified)
- **Wireframe**: medium fidelity
- **Prototype**: high fidelity (more polished & detailed)
Storyboards
Classic StoryBoards

Storyboard from Studio Ghibli: “My Neighbor Totoro”
Classic Storyboards

Credit Studio Ghibli: “Spirited Away”
Storyboards for UI Design

• Sequence of visual “frames” illustrating interplay between user & envisioned system

• Explains how app fits into a larger context through a single scenario / story

• Bring design to life in graphical clips - freeze frame sketches of user interactions

• “Comic-book” style illustration of a scenario, with actors, screens, interaction, & dialog
Crafting a Storyboard

• Set the stage:
  • Who? What Where? Why? When?
• Show key interactions with application
• Show consequences of taking actions
• May also think about errors
Example Elements of a UI Storyboard

- Hand-sketch pictures annotated with a few words
- Sketch of user activity before or after interacting with the system
- Sketches of devices & screens
- Connections with system (e.g., database connection)
- Physical user actions
- Cognitive user action in “thought balloons”
Example: Ticket Kiosk

Ticket buyer walks up to the kiosk

Sensor detects user & starts immersive process

Displays "Occupied" sign on wraparound case

Detects people with ID card
Example: Ticket Kiosk

Greets buyer and asks for PIN

Buyer selects “Boston symphony at Burruss Hall”

Shows recommendations & most popular categories

Plays music from symphony, shows date & time picker
Frame Transitions

- Transitions between frames particularly important
- What users think, how users choose actions
- Many problems can occur here (e.g., gulfs of execution & evaluation) - we will talk more in a future class!
- Useful to think about how these work, can add thought bubbles to describe
Wireframes & Design Critiques
Wireframes

- Lines & outlines ("wireframes") of boxes & other shapes
- Capturing emerging interaction designs
- Schematic designs to define screen content & visual flow
- Illustrate approximate visual layout, behavior, transitions emerging from task flows
- Deliberate unfinished: do not contain finished graphics, colors, or fonts
Example

<table>
<thead>
<tr>
<th>Work bench</th>
<th>Italy trip</th>
<th>4th of July</th>
</tr>
</thead>
<tbody>
<tr>
<td>Italy trip</td>
<td>Ani's birthday</td>
<td>Graduation</td>
</tr>
<tr>
<td>Camping trip</td>
<td>Pisa set</td>
<td>Rome set</td>
</tr>
<tr>
<td>July set</td>
<td>July beach trip</td>
<td></td>
</tr>
</tbody>
</table>

Cloud 9 Photos [Secondary title here] [Login]

Create new collection

Upload photos

[Photo view preview]

Toolbar with photo manipulation options

Photo versions

<table>
<thead>
<tr>
<th>Italy trip</th>
<th>June 2011</th>
<th>(138 photos)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Img 2356</td>
<td>Img 2357</td>
<td>Img 2358</td>
</tr>
<tr>
<td>Img 2359</td>
<td>Img 2360</td>
<td>Img 2361</td>
</tr>
</tbody>
</table>
Example
Wireframes

- Can be used to step through a particular scenario
- Focus on key screens rather than every screen
- Tools can help
  - Can be made clickable
  - Can use stencils & templates; copy & edit similar screens
Creating a Wireframe - (1)

- What are the key interactions needed to support design?
- What widgets support these interactions?
- What are the best ways to lay them out?
- How do these relate to conceptual design & user’s mental model?
Creating a Wireframe - (2)

• What are all of the items: toolbars, scrollbars, windows, …?
• Are there too many widgets on the screen?
• What happens when data is larger than available space? Will entire page scroll, or individual panel?
• How much detail of items to show?
Example Tool - Balsamiq
Design Critiques

• Stylized meeting for getting feedback on design sketches & prototypes
• Solicit feedback from peers
• History: studio art education

http://www.flickr.com/photos/pjchmiel/2972140234/
Designer: Frame the Discussion

• State *explicitly*: What would you like comments on?
  • Overall idea?
  • Usability?
  • Specific interaction design?
  • Visual design?
• Take a *dispassionate* stance (this is hard!)
  • Show alternatives where possible
Critic: How to Avoid Deaf Ears

• Comments about the design, not the designer

• Point out positive aspects - be specific
  • Not: “I like this, but…”
  • “The layout effectively communicate the hierarchical nature of the data. However…”

• Ask for alternatives instead of offering solutions
  • Not: “You should really change X”
  • Instead “Have you considered alternatives for X?”
Prototyping
Prototyping

• How do you know your system design is right before you invest the time to build it?

• Answer: prototyping!

  • Evaluation performed before investing resources in building finished product

  • Early version of system constructed much faster & with less expense used to evaluate & refine design ideas
Types of Prototypes

- Which details do you leave out?
  - **Horizontal**: *broad* in features, less depth
    - Explore overall concept of app, but not specific workflows
  - **Vertical**: lots of *depth*, but only for a few features
    - Enables testing limited range of features w/ realistic user evals
  - **T**: most of UI realized at low depth, few parts realized in depth
    - Combination of vertical & horizontal
  - **Local**: focused prototype on *specific* interaction detail
Interactivity of Prototypes

• Scripted, click through prototypes
  • Prototype w/ clickable links to move between screens
  • Live action storyboard of screens
  • Simulates real task flow, but w/ static content
• Fully-implemented prototypes
  • Usually expensive to implement actual system
  • But can build key piece of system first to evaluate
Wizard of Oz

• Goal: *simulate* actual system w/ out building it
  • Want user to interact *as if* they were interacting w/ real system
  • Helps explore how users would interact w/ novel interaction if it were to exist
• Example: natural command line (Good et al 1984)
  • Users typed in commands to interact w/ computer
  • Commands intercepted by hidden human who interpreted commands & executed them
Paper Prototypes

• *Low fidelity* prototype w/ paper mockups

• **Goal:** get feedback from users early w/ very low cost interactive prototype of envisioned interaction design
Paper Prototyping (1)

- Set a realistic deadline
- Gather set of paper prototyping materials
- Work **fast** & do not color within the lines
- Reuse existing sketches & mockups
- Make underlying paper mockups of key screens
Paper Prototyping (2)

- Use **paper cutouts** & tape onto full-size transparencies as “interaction sheets” for moving parts, making modular by including only a small amount
- Do not write or mark on interaction sheets
- Be *creative*
- *Reuse* at every level
- Cut corners wherever possible (trade accuracy against efficiency)
- Make a “this feature not implemented” message
Paper Prototyping (3)

- Include “decoy” user interface objects not needed for expected tasks
- Accommodate data value entry by users with blank transparencies
- Organize materials to manage complex task threads
- Pilot test thoroughly
10 Minute Break
In Class Activity
Group activity

• In groups of 2:
  • The venture capitalist from Lecture 3 who invested $5M in your new consumer product would like an update! They'd like to see how your app would work in one specific scenario, and how this would help better meet user needs.
  • Start with a specific set of user needs and develop a key scenario illustrating a benefit of your app.
  • Build a series of at least 5 wireframe “pages” supporting one scenario for the app.

• Deliverables
  • Few sentences describing the purpose of the app.
  • Few sentences describing the scenario for the app: what is the user's goal.
  • At least 5 wireframe pages describing what the app looks like at each step, with annotations below describing the user's goal.
  • Few sentences explaining why this design is better than current approach users might use.