[CS3704] Software Engineering

Dr. Chris Brown

Virginia Tech

10/9/2023

Announcements

- Research Discussion Presentations this week
  - SE Processes on Wednesday (10/11)
  - Requirements on Friday (10/13)
  - Sign up if you haven’t already done so!
- HW2 due tonight by 11:59pm!
- PM2 due Friday (10/13) by 11:59pm!

Design

Software Design Overview
Usability Engineering
UI Design Processes

Learning Outcomes

By the end of the course, students should be able to:

- **Understand software engineering processes, methods, and tools used in the software development life cycle (SDLC)**
- **Use techniques and processes to create and analyze requirements for an application**
- **Use techniques and processes to design a software system**
- **Identify processes, methods, and tools related to phases of the SDLC**
- **Explain the differences between software engineering processes**
- **Discuss research questions and current topics related to software engineering**
- **Create and communicate about the requirements and design of a software application**

Warm-up

Stand-up Meeting: Since the last stand-up meeting, discuss:

- **What you did.**
- **What you need to do next.**
- **What is blocking you.**
- **Thoughts on the guest lecture from Ben Nelson last Wednesday.**

Design

Goal: decide the structure of the software and the hardware configurations that support it.

  - The *how* of the project

- How individual classes and software components work together in the software system.
  - Programs can have 1000s of classes/methods
- *Software Artifacts:* design documents, class diagrams (i.e. UML)

Design Engineering

- The process of making decisions about HOW to implement software solutions to meet requirements.
- Encompasses the set of concepts, principles, and practices that lead to the development of high-quality systems.

How Do Developers Design Software?

- **Code**
  - Design-while-coding 😥
- **Iterative and evolutionary design**
  - Diagrams and modeling
    - UML
    - Class diagrams
    - Sequence diagrams
- **Reuse or modify existing design models**
  - High-level: Architectural patterns
  - Low-level: Design patterns

Diagrams

- Drawing and diagramming are essential tasks in software development…

Understanding existing code

Design review

Explain to external members

Ad-hoc meetings (whiteboard)

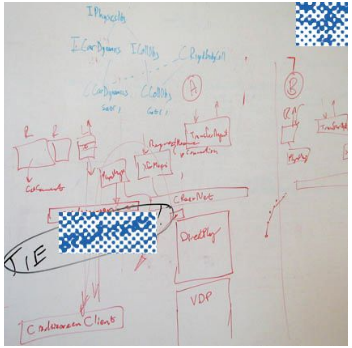
Onboarding

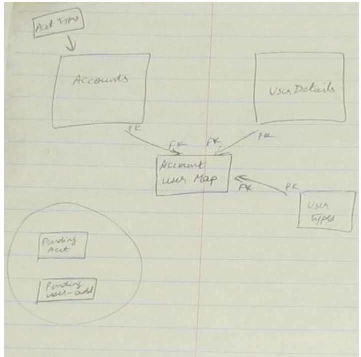
Hallway art

Refactoring plans

Explain to colleagues & newcomers

Documentation





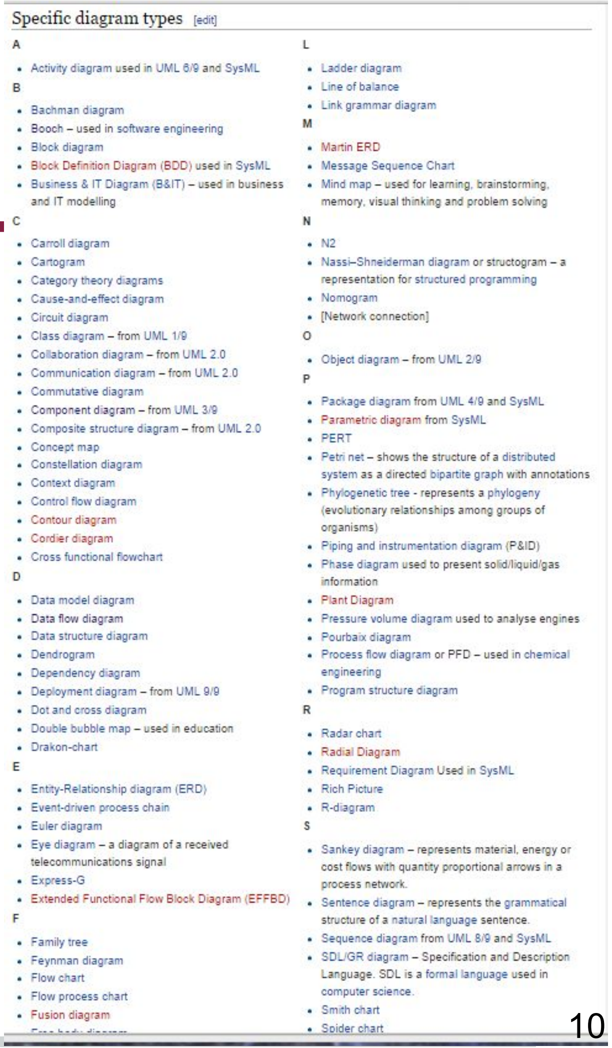
Types of Diagrams

Software Design

- Class Diagrams
- Sequence Diagrams
- State Diagrams

Interface Design

- Wireframes
- Flow Maps/Navigation Maps
- Storyboards
- … *and many more!*



Specific diagram types [edit]

A

- Activity diagram used in UML 6/9 and SysML

B

- Bachman diagram
- Booch – used in software engineering
- Block diagram
- Block Definition Diagram (BDD) used in SysML
- Business & IT Diagram (B&IT) – used in business and IT modelling

C

- Carroll diagram
- Cartogram
- Category theory diagrams
- Cause-and-effect diagram
- Circuit diagram
- Class diagram – from UML 1/9
- Collaboration diagram – from UML 2.0
- Communication diagram – from UML 2.0
- Commutative diagram
- Component diagram – from UML 3/9
- Composite structure diagram – from UML 2.0
- Concept map
- Constellation diagram
- Context diagram
- Control flow diagram
- Contour diagram
- Cordier diagram
- Cross functional flowchart

D

- Data model diagram
- Data flow diagram
- Data structure diagram
- Dendrogram
- Dependency diagram
- Deployment diagram – from UML 9/9
- Dot and cross diagram
- Double bubble map – used in education
- Drakon-chart

E

- Entity-Relationship diagram (ERD)
- Event-driven process chain
- Euler diagram
- Eye diagram – a diagram of a received telecommunications signal
- Express-G
- Extended Functional Flow Block Diagram (EFFBD)

F

- Family tree
- Feynman diagram
- Flow chart
- Flow process chart
- Fusion diagram

L

- Ladder diagram
- Line of balance
- Link grammar diagram

M

- Martin ERD
- Message Sequence Chart
- Mind map – used for learning, brainstorming, memory, visual thinking and problem solving

N

- N2
- Nassi–Shneiderman diagram or structogram – a representation for structured programming
- Nomogram
- [Network connection]

O

- Object diagram – from UML 2/9

P

- Package diagram from UML 4/9 and SysML
- Parametric diagram from SysML
- PERT
- Petri net – shows the structure of a distributed system as a directed bipartite graph with annotations
- Phylogenetic tree - represents a phylogeny (evolutionary relationships among groups of organisms)
- Piping and instrumentation diagram (P&ID)
- Phase diagram used to present solid/liquid/gas information
- Plant Diagram
- Pressure volume diagram used to analyse engines
- Pourbaix diagram
- Process flow diagram or PFD – used in chemical engineering
- Program structure diagram

R

- Radar chart
- Radial Diagram
- Requirement Diagram Used in SysML
- Rich Picture
- R-diagram

S

- Sankey diagram – represents material, energy or cost flows with quantity proportional arrows in a process network.
- Sentence diagram – represents the grammatical structure of a natural language sentence.
- Sequence diagram from UML 8/9 and SysML
- SDL/GR diagram – Specification and Description Language. SDL is a formal language used in computer science.
- Smith chart
- Spider chart

UML

- Unified Modeling Language (UML) is a standard for modeling object-oriented software.
    - Currently on version 2.0
    - Typically depicted with two types of diagrams
        - Structural (i.e. class diagrams)
        - Behavioral (i.e. use case and sequence diagrams)

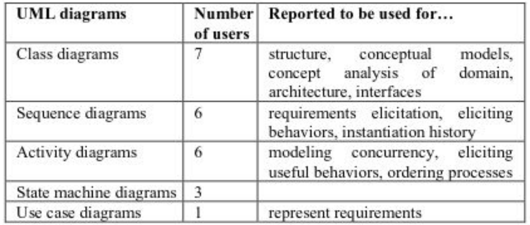
How is UML Really Used?

“UML has been described by some as ‘the lingua franca of software engineering’. Evidence from industry does not necessarily support such endorsements. How exactly is UML being used in industry – if it is? This paper presents a corpus of interviews with 50 professional software engineers in 50 companies and identifies 5 patterns of UML use.” [Petre]

NONE!	**70%**
SELECTIVE	22%
AUTOMATIC CODE GEN	6%
RETROFIT	2%
WHOLE	0%

Of those that reported using it…

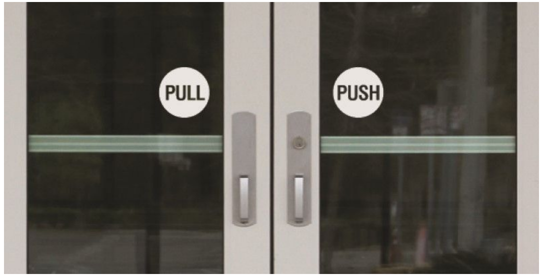
TABLE II. ELEMENTS OF UML USED BY THE 11 ‘SELECTIVE’ USERS.



UML diagrams	Number of users	Reported to be used for…
Class diagrams	7	structure, conceptual models, concept analysis of domain, architecture, interfaces
Sequence diagrams	6	requirements elicitation, eliciting behaviors, instantiation history
Activity diagrams	6	modeling concurrency, eliciting useful behaviors, ordering processes
State machine diagrams	3	
Use case diagrams	1	represent requirements

User Interfaces

- The way users interact with the system
  - Medium between human and computer
- A *very important* part of software design
  - ***All software has a user interface!***
  - User interface (**UI**), User experience (**UX**), Human-computer interaction (**HCI**)
  - **Affordances:** Property of an object that defines its possible uses
    - products should make clear how they should be used



Example Styles of UI

- Direct manipulation
  - the user interacts with objects on the screen

  – E.g., drag a file to a “trash bin”
- Menu selection

  – E.g., select the “delete” on menu for a file
- Form fill-in

  – E.g., fill a file name and click “delete” button

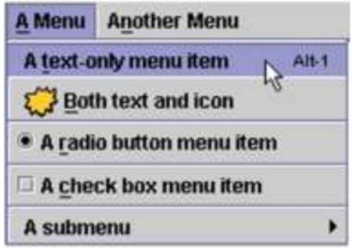
Example Styles of UI (cont.)

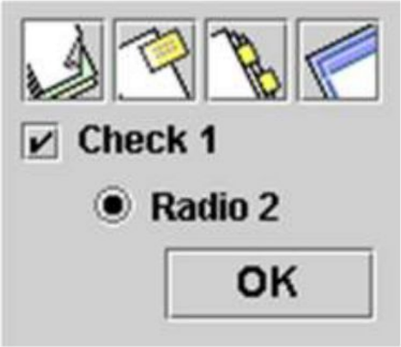
- Command language
  – Type in delete command with the filename as a parameter
- Natural language
  - Type in natural language description, which will be parsed and executed

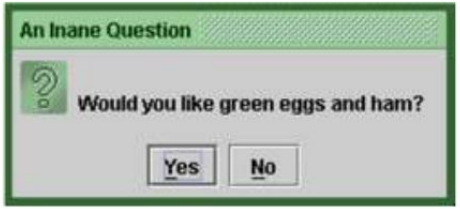
  – E.g., “delete the file named xxx”

UI Design Components

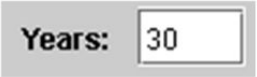
- When should we use:
  - A button?
  - A check box?
  - A radio button?
  - A text field?
  - A list?
  - A combo box?
  - A menu?
  - A dialog box?
  - Other..?

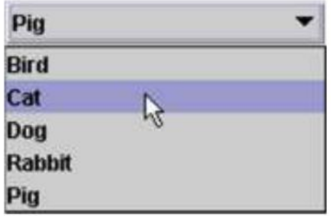


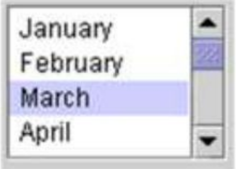


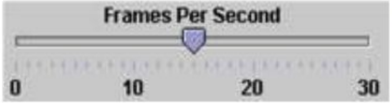


Would you like green eggs and ham?


Years: 30





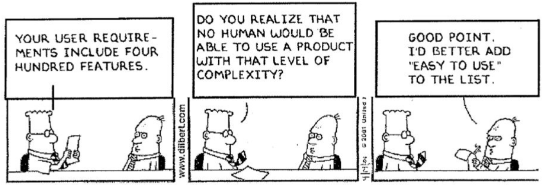


What is UI Design?

- Following a set of interface design principles, design identifies interface objects and actions and then creates a screen layout that forms the basis for a user interface prototype
- **Goal:** To make user interfaces easy to understand, learn, and use.

Usability

- ***Usability:*** How well users can use a system’s functionality
- Dimensions of usability
  - Learnability: is it easy to learn?
  - Efficiency: once learned, is it fast to use?
  - Visibility: is the state of the system visible?
  - Errors: are errors few and recoverable?
  - Satisfaction: is it enjoyable to use?


YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.
DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?
GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.
www.dilbert.com
© 2001 United

Typical Interface Design Errors

- lack of consistency
- too much memorization
- no guidance / help
- no context sensitivity
- poor response
- arcane/unfriendly
- …

TODO: Discuss an experience using poorly designed software. What made the design unusable and which of these (or other) UI design errors existed? (also in HW3)

Three Golden Rules

1. Place the user in control
2. Reduce the user’s memory load
3. Make the interface consistent

[Mandel]

Place the User in Control

- Define interaction modes which do not force users into unnecessary actions
- Provide flexible interaction
  - E.g., keyboard commands and mouse clicks
- Allow user interaction to be interruptible or undoable
  - E.g., Automatic save, undo, redo

Place the User in Control (cont.)

- Allow for streamline interaction as skill levels advance (customization)
- Hide technical details from user

"What I really would like is a system that reads my mind"

Reduce the User’s Memory Load

- Reduce demand on short-term memory
  - E.g., autofill, single sign-on
- Establish meaningful defaults
- Define intuitive shortcuts
  - E.g., “alt-P” to “print”
- Base visual layout on real-world metaphors
  - Whenever possible
  - E.g., email as ✉️
- Disclose information in stages
  - Use a hierarchy for choices

> *“The more a user has to remember, the more error-prone the interaction will be”*

Make the Interface Consistent

- Allow understanding of current task in context – Window titles, graphical icons, consistent color usage, forward, backward
- Maintain consistency across a family of SW products
- If users have expectations from past interactions, try not to make changes.

“Things that look the same should act the same”

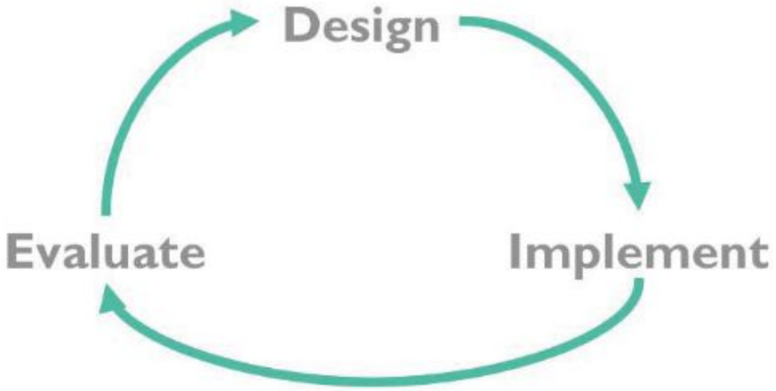
Usability Heuristics

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help users recognize, diagnose, and recover from errors
10. Help and documentation

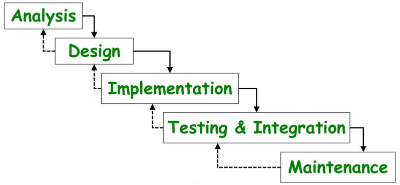
[Nielsen]

Usability Engineering

- Designing usable software is an iterative process.
  - Often will not get it right the first time

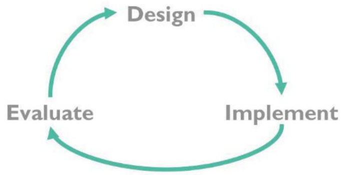


Usability Engineering (cont.)


Analysis
Design
Implementation
Testing & Integration
Maintenance

- Waterfall processes are bad for usability engineering and user interface design!
  - UI design is risky
    - Often will not get it right the first time
  - Users are often not involved until acceptance testing
    - No feedback until the end
  - UI flaws often cause changes in requirements
    - Throw away carefully-written and tested code, or deploy with a bad UI

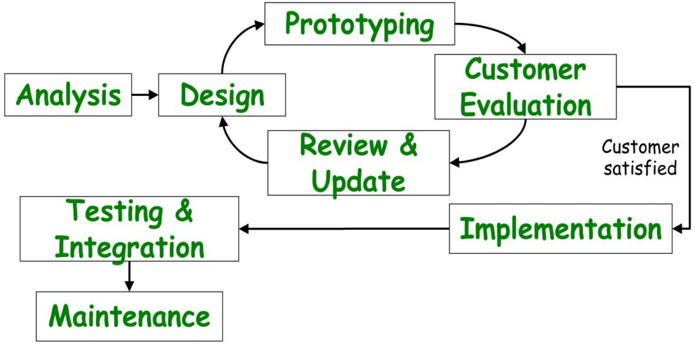
Iterative Design of UIs

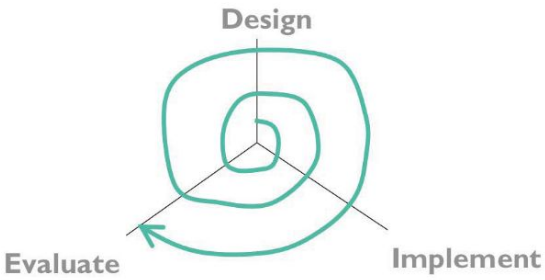


Iterative Design the Wrong Way:

- Every iteration corresponds to a release
  - Feedback (complaints) feed into the next version design for the next iteration.
    - Evaluation of prototypes
    - Learn from past mistakes
- Using (paying) users to evaluate usability
  - They won’t like it
  - They won’t buy version 2

Iterative Design of UIs





- Build room for several iterations in UI design processes.
    - Making early iterations as cheap as possible
- Early prototypes can detect usability issues
- **Parallel Design** is feasible: build and test multiple prototypes
- Later iterations use richer implementation after UI risk is mitigated
- More iterations generally means better UI
    - Mature iterations deployed to users

User-Centered Design

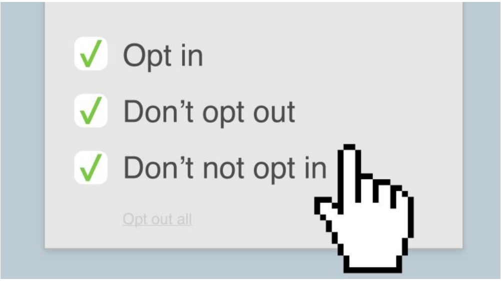
- Iterative Design
  - Using rapid prototyping
- Early focus on users and tasks
  - User analysis: who are the users
  - Task analysis: what do they need to do
  - Involve users as evaluators, consultants, and even designers
- Constant evaluation
  - Users involved in every iteration

Universal Design

- A school of thought that seeks to design for ***all*** users, as much as possible
  - As opposed to designing for the *typical* user.
- Should not involve “dumbing down” interfaces, but make design better for everyone.
- Guidelines for Universal/Accesible Design
  - Section 508 rules: https://www.section508.gov/
  - World Wide Web Content (WC3) Accessibility Initiative: https://www.w3.org/TR/WAI-WEBCONTENT/

Dark Patterns

- Dark patterns are user interface designs that deceive users into making unintended decisions.
  - Examples: hidden costs and fees, confusing language, publicly sharing information unknowingly, confirmshaming,...

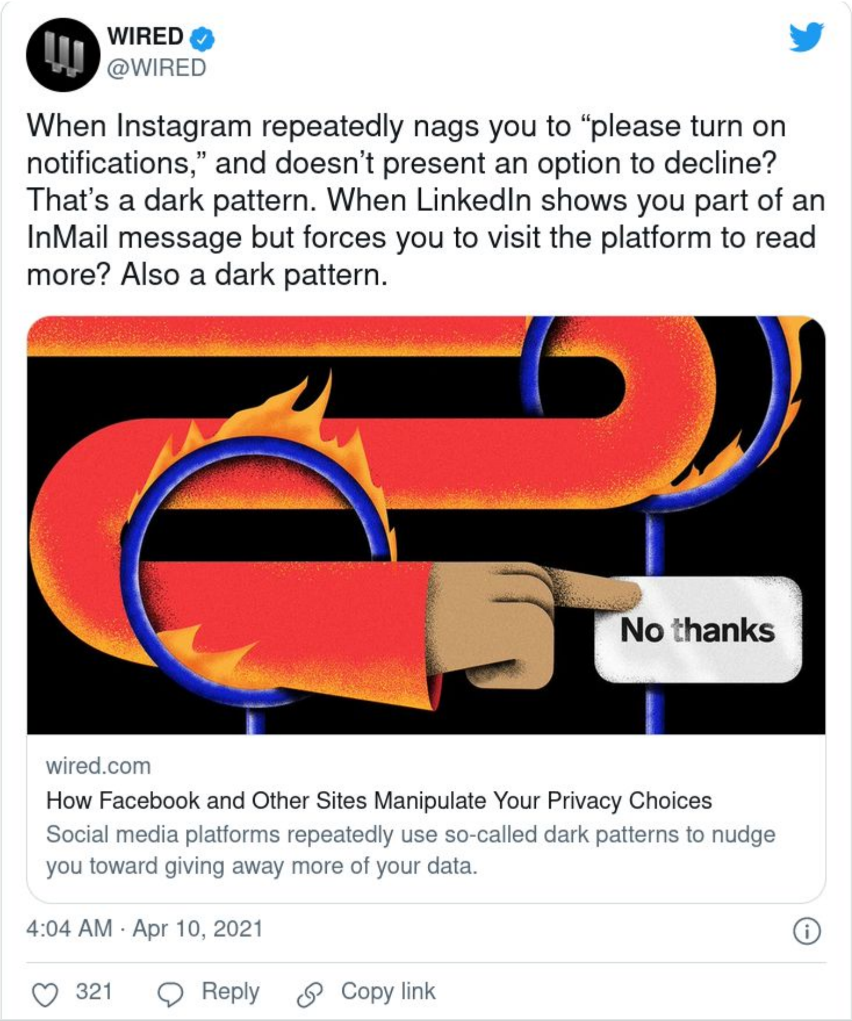


☑ Opt in

☑ Don’t opt out

☑ Don’t not opt in

Opt out all



WIRED
@WIRED

When Instagram repeatedly nags you to “please turn on notifications,” and doesn’t present an option to decline? That’s a dark pattern. When LinkedIn shows you part of an InMail message but forces you to visit the platform to read more? Also a dark pattern.

wired.com

How Facebook and Other Sites Manipulate Your Privacy Choices

Social media platforms repeatedly use so-called dark patterns to nudge you toward giving away more of your data.

4:04 AM · Apr 10, 2021

How to Design a UI?

- Understand what users need
  – Types of users
  – Tasks users will perform with the system
- Use cases
  – Design task is similar to design of rest of the system
  – Offer interactions that “fit” user requirements

Types of Users

- Novice
  - Have little knowledge about usage
  - Use small vocabulary of familiar terms
  - Give informative feedback
- Knowledgeable intermittent users
  - Know task but may forget specific details
- Frequent users
  - Want to accomplish tasks rapidly with as few keystrokes or clicks as possible

Help Facility

- How does user request help?
- How is help presented?
  - Separate window, 1-2 line suggestion at a fixed screen location, pointer to document,...
- How does user return to normal mode?
- Is help flat or structured?

UI Error Handling

What happens when a user interacts with your system incorrectly?

- Describe the problem in the language user can understand, in non-judgmental manner
- Provide constructive advice for recovery
- Indicate any negative consequences
- Message associated with visual or audio cue

How to Design UIs? (cont.)

DISCOVER

- Bodystorming
- Cognitive walkthrough
- Contextual inquiry
- Design studio
- Dot voting
- Heuristic analysis
- KJ method
- Metrics definition
- Stakeholder and user interviews

DECIDE

- Affinity diagramming
- Comparative analysis
- Content audit
- Design principles
- Journey mapping
- Mental modeling
- Personas*
- Site mapping
- Storyboarding*
- Style tiles
- Task flow analysis*
- User scenarios

MAKE

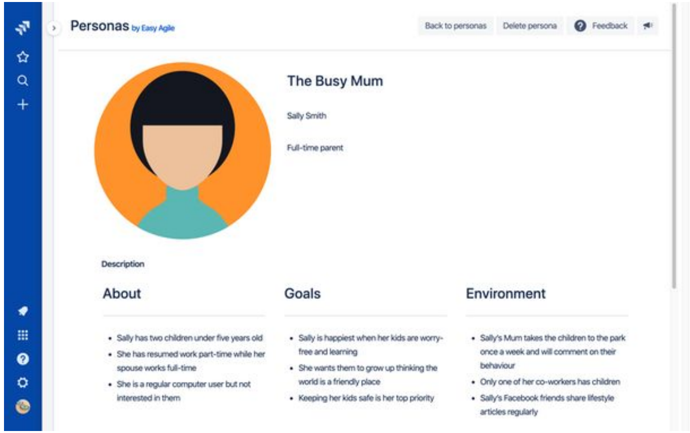
- Design pattern library
- Prototyping*
- Wireframing*

VALIDATE

- Card sorting
- Multivariate testing
- Usability testing
- Visual preference testing

Decide: Personas

A **persona** is arch-user type which represents a segment of a user population, and allows role-play during task planning and UX design.



The Busy Mum

Sally Smith

Full-time parent

Description

About

- Sally has two children under five years old
- She has resumed work part-time while her spouse works full-time
- She is a regular computer user but not interested in them

Goals

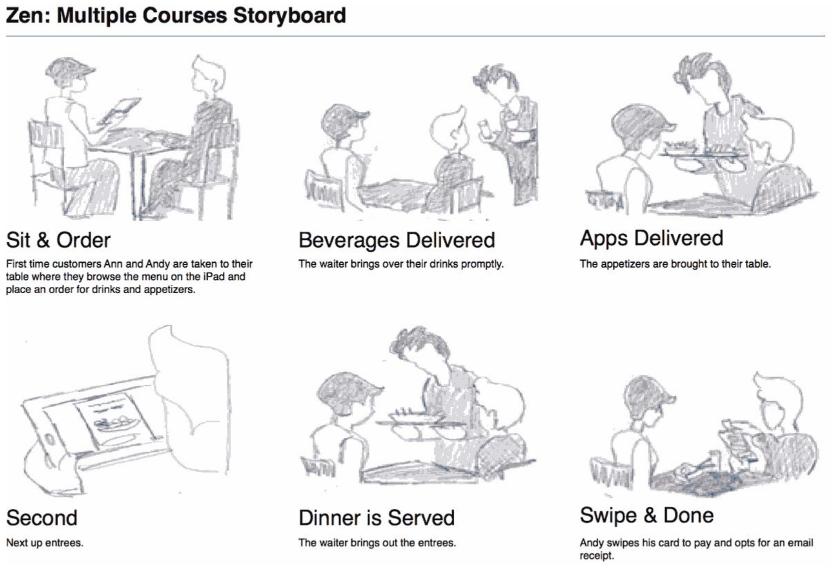
- Sally is happiest when her kids are worry-free and learning
- She wants them to grow up thinking the world is a friendly place
- Keeping her kids safe is her top priority

Environment

- Sally's Mum takes the children to the park once a week and will comment on their behaviour
- Only one of her co-workers has children
- Sally's Facebook friends share lifestyle articles regularly

Decide: Storyboarding

A **storyboard** illustrates the *timeline* of user performing a task as a sequence of frames.



Zen: Multiple Courses Storyboard

Sit & Order

First time customers Ann and Andy are taken to their table where they browse the menu on the iPad and place an order for drinks and appetizers.

Beverages Delivered

The waiter brings over their drinks promptly.

Apps Delivered

The appetizers are brought to their table.

Second

Next up entrees.

Dinner is Served

The waiter brings out the entrees.

Swipe & Done

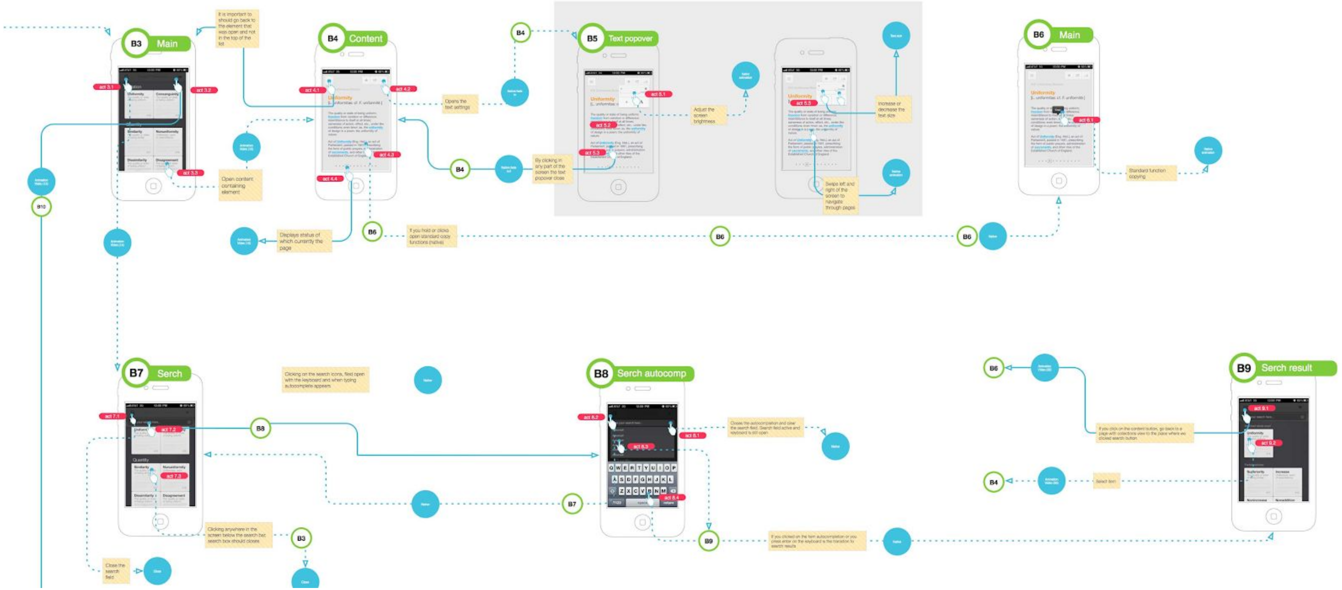
Andy swipes his card to pay and opts for an email receipt.

Useful tools:

- Storyboarder
- Plot
- FrameForge
- Miro
- Canva
- StoryboardThat
- ...

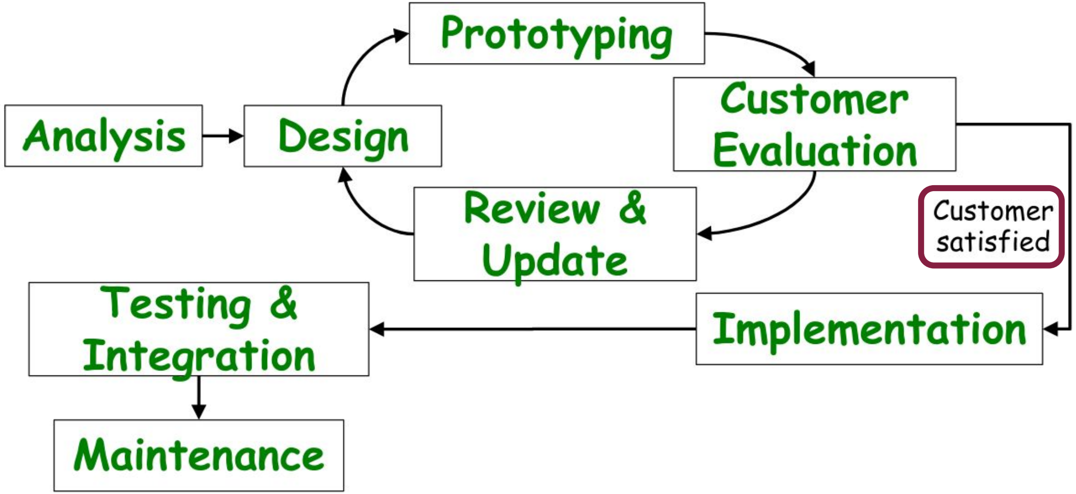
Decide: Task Flow Analysis

A **flow map** describes the *wayfinding* activity of a user and transitions between UI states.



B3 Main
act 3.1
act 3.2
act 3.3
It is important to should go back to the element that was open and not in the top of the list
Open content containing element
B10
B4 Content
act 4.1
act 4.2
act 4.3
act 4.4
Opens the text settings
Displays status of which currently the page
B6
If you hold or clicks open standard copy functions (native)
B4
By clicking in any part of the screen the text popover close
B5 Text popover
act 5.1
act 5.2
act 5.3
Adjust the screen brightness
Increase or decrease the text size
Swipe left and right of the screen to navigate through pages
B6 Main
act 6.1
Standard function copying
B7 Serch
act 7.1
act 7.2
act 7.3
Clicking on the search icons, field open with the keyboard and when typing autocomplete appears
B8
Clicking anywhere in the screen below the search bar, search box should closes
B3
Close the search field
B8 Serch autocomp
act 8.1
act 8.2
act 8.3
act 8.4
Closes the autocompletion and clear the search field. Search field active and keyboard is still open
B7
B9
If you clicked on the item autocompletion or you press enter on the keyboard is the transition to search results
B9 Serch result
act 9.1
act 9.2
If you click on the content button, go back to a page with collections view to the place where we clicked search button
Select item

Make: Prototyping



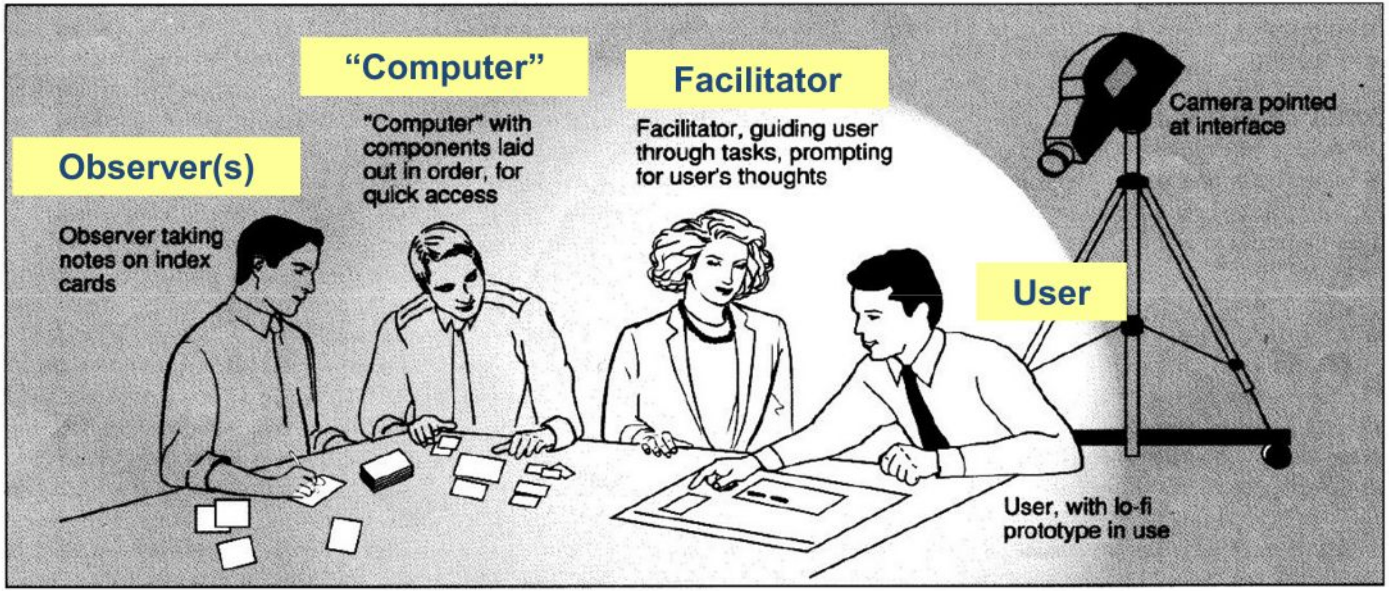
Useful tools:

- Figma
- Miro
- App Builder
- Adobe XD
- Balsamic
- Usability Hub
- ...

Prototype involves clients, design quickly and iteratively before implementation.

Make: Prototyping (cont.)

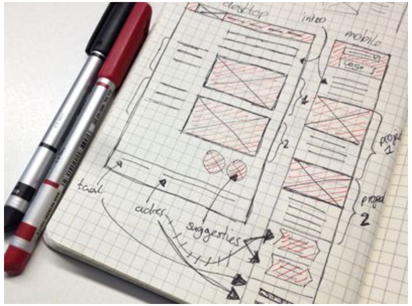
- In the past, companies adopted a process known as “paper prototyping”.

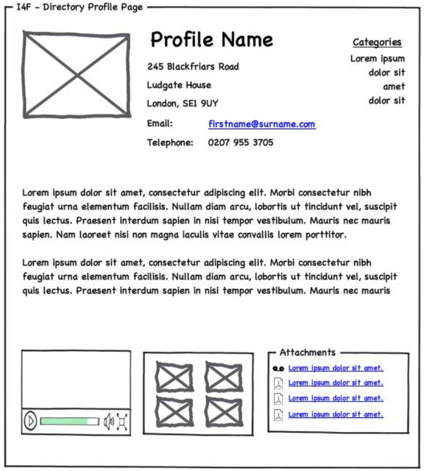

“Computer”
"Computer" with components laid out in order, for quick access
Facilitator
Facilitator, guiding user through tasks, prompting for user's thoughts
Camera pointed at interface
Observer(s)
Observer taking notes on index cards
User
User, with lo-fi prototype in use

Make: Wireframes

A **wireframe** is a view schematic that captures all layout and content decisions of that view.

- How will you allocate space for particular content?
- Where will content live?
- What actions can you do?




I4F - Directory Profile Page

Profile Name

245 Blackfriars Road
Ludgate House
London, SE1 9UY

Email: firstname@surname.com
Telephone: 0207 955 3705

Categories
Lorem ipsum
dolor sit
amet
dolor sit

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur nibh feugiat urna elementum facilisis. Nullam diam arcu, lobortis ut tincidunt vel, suscipit quis lectus. Praesent interdum sapien in nisi tempor vestibulum. Mauris nec mauris sapien. Nam laoreet nisi non magna iaculis vitae convallis lorem porttitor.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi consectetur nibh feugiat urna elementum facilisis. Nullam diam arcu, lobortis ut tincidunt vel, suscipit quis lectus. Praesent interdum sapien in nisi tempor vestibulum. Mauris nec mauris

Attachments
Lorem ipsum dolor sit amet.
Lorem ipsum dolor sit amet.
Lorem ipsum dolor sit amet.
Lorem ipsum dolor sit amet.

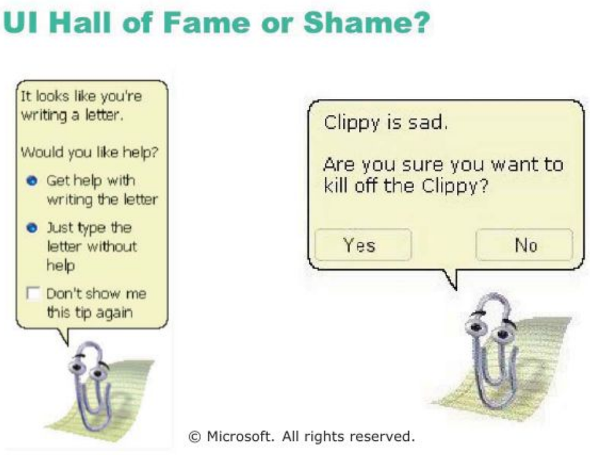
created with Balsamiq Mockups - www.balsamiq.com

The User Interface is Important

- UIs strongly affect the perception of software
  - Usable software sells better
  - Unusable software is abandoned
- Costly to get the user interface wrong
  - Users’ time is not cheap
  - Design it correctly now, or pay for it later

But User Interface Design is Hard…

- Many different types of users…
  - Different expectations…
  - Different experiences…
  - Different goals…
  - Different abilities…
- “Good” design is relative
- Evolving technology/interaction norms
- Increased digital literacy
- Perception can be superficial
- The user is always right
- …but the user is not always right either
- You are not the user!



UI Hall of Fame or Shame?

It looks like you're writing a letter.

Would you like help?

● Get help with writing the letter

● Just type the letter without help

□ Don't show me this tip again

Clippy is sad.

Are you sure you want to kill off the Clippy?

© Microsoft. All rights reserved.

Usability is only one attribute

- Software designers have a lot of other factors to worry about:
    - *Usability*
    - Functionality
    - Cost
    - Performance
    - Size
    - Reliability
    - Security
    - Accessibility…
- Many design decisions in software involve tradeoffs between different attributes.

Next Time…

- **Presentation Discussions (10/11, 10/13)**
- High-Level Design (10/16)

- **HW2 due tonight (11:59pm)**
- **PM2 due Friday (11:59pm)**
- **HW3 due next Friday (10/20 at 11:59pm)**

References

- RS Pressman. *“Software engineering: a practitioner's approach”.*
- Robert Miller. *“User Interface Design and Implementation”.* MIT Open Courseware
- Jakob Nielsen. *“Designing Web Usability”*.
- Jakob Nielsen. *“Finding usability problems through heuristic evaluation”*.
- Theo Mandel. *“Golden Rules of User Interface Design”.* <https://theomandel.com/resources/golden-rules-of-user-interface-design/>
- Na Meng and Barbara Ryder
- Chris Parnin
- Sarah Heckman