Lesson 9

Interaction Design (CM3055)

“If a picture is worth 1,000 words,
a **prototype** is worth 1,000 meetings.”

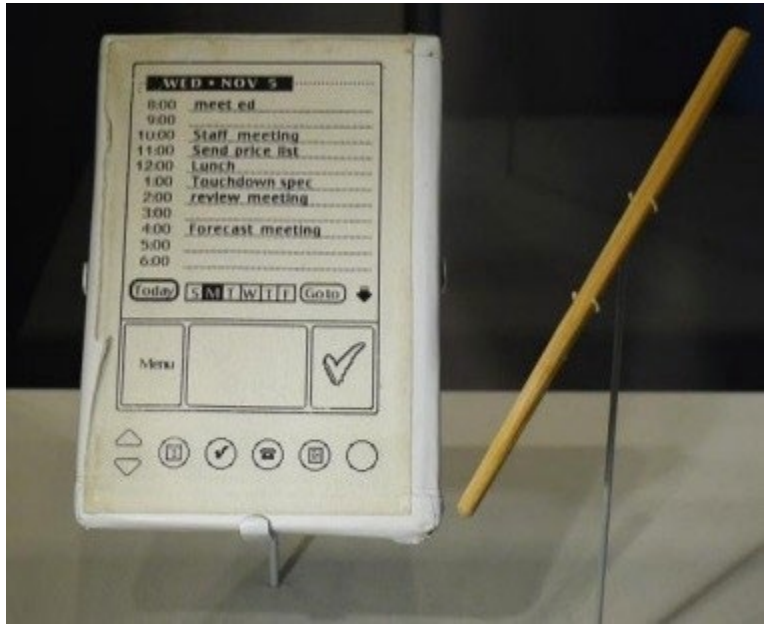
Tom and David Kelley, IDEO

What is a prototype?

- What do you think of when you hear “prototype”?
- What kinds of prototypes have you seen anywhere?
- What are they “for”?

In HCI / interaction design, a prototype can be (among other things):

- a series of screen sketches
- a storyboard, i.e. a cartoon-like series of scenes
- a Powerpoint slide show
- a video simulating the use of a system
- a lump of wood (e.g. PalmPilot)
- a cardboard mock-up
- a piece of software with limited functionality written in the target language or in another language



wooden Palm Pilot

Types of Prototyping

- Rapid (Throwaway) prototyping
- Evolutionary prototyping
- Incremental prototyping
- Extreme prototyping

A prototype

Is a primitive representation or version of a product that a design team or front-end-development team typically creates during the design process.

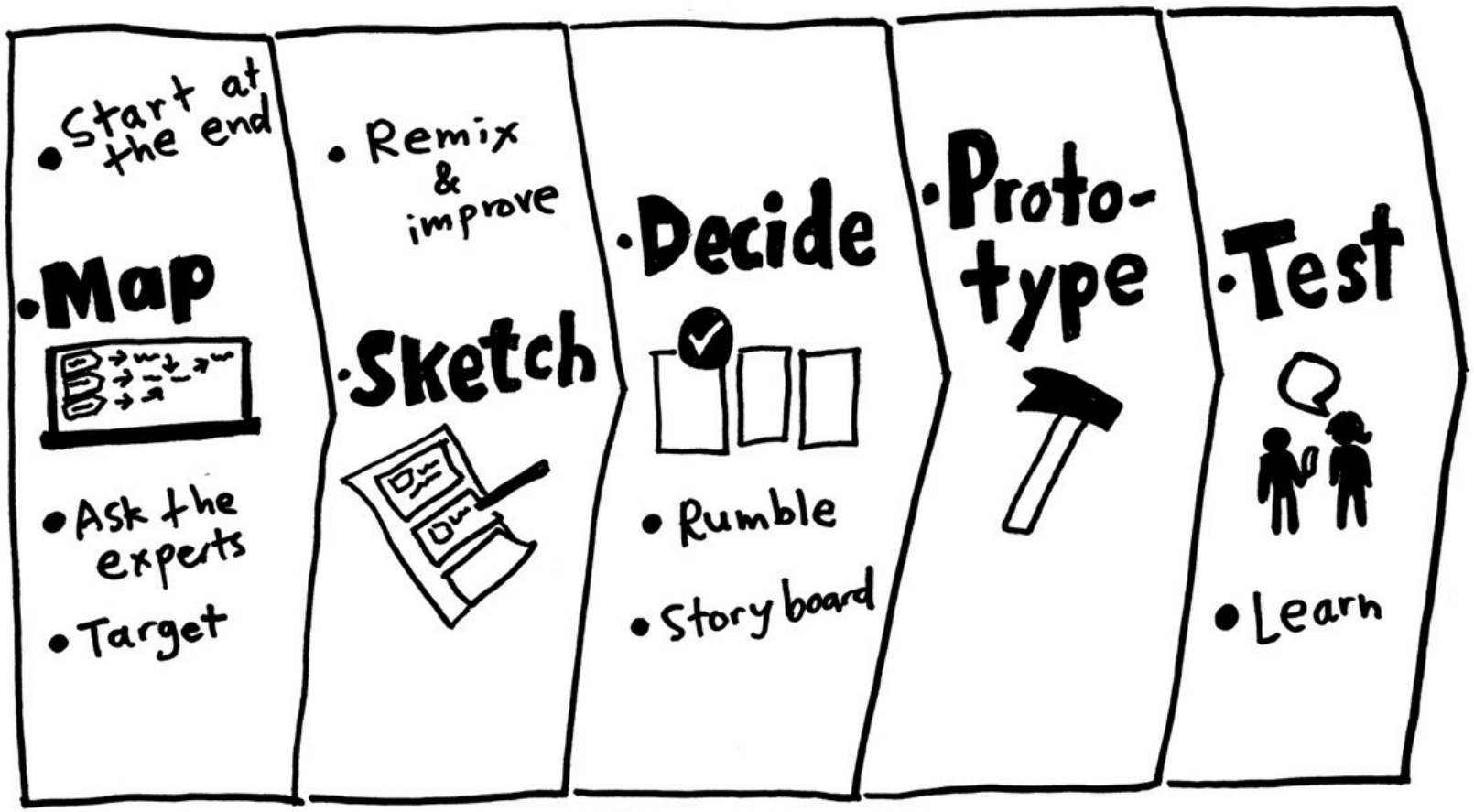
Goal

is to test the flow of a design solution and gather feedback on it - from both internal and external parties - before constructing the final product.

State of a prototype is fluid > the team revises the design iteratively based on user feedback.

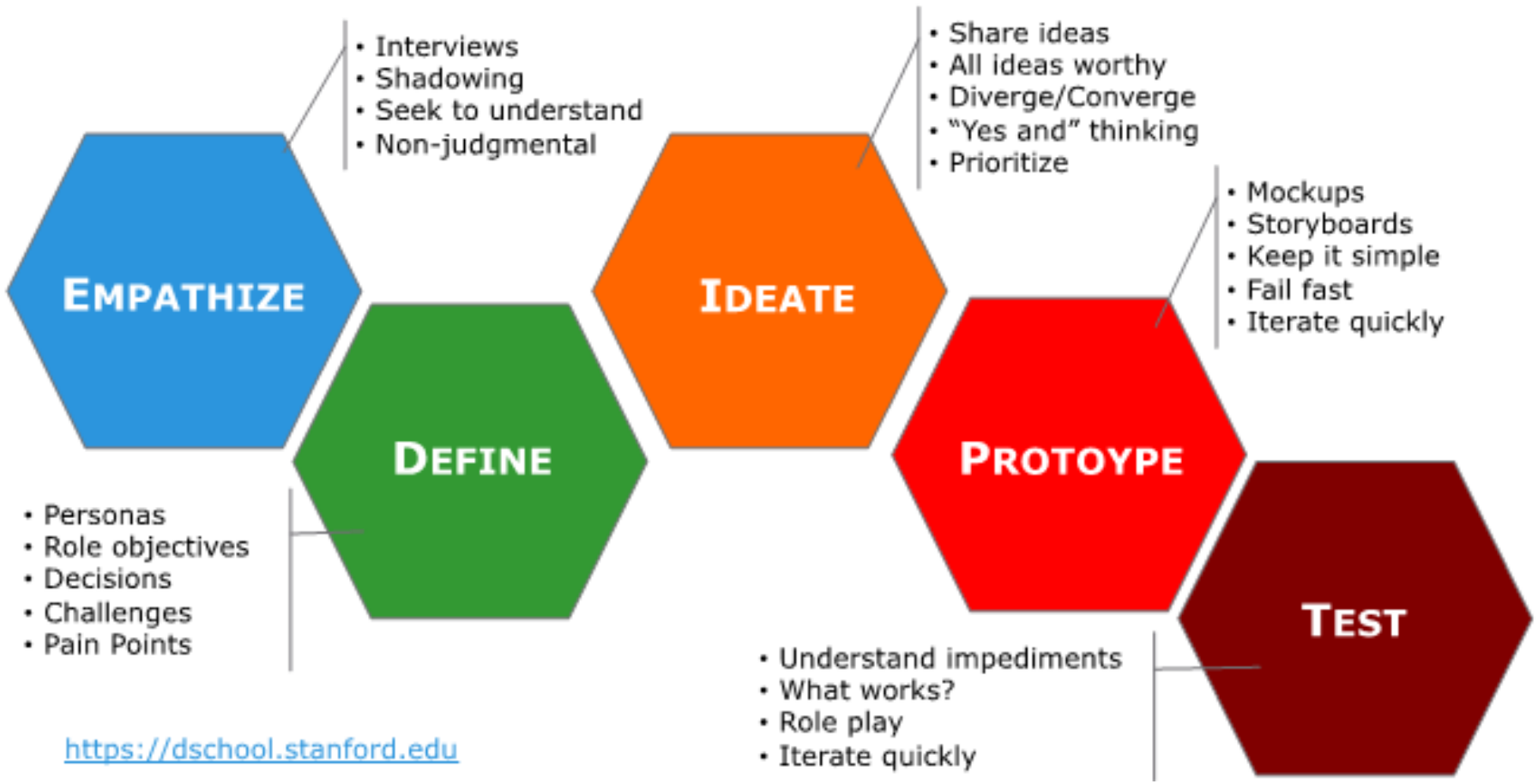
*"They **slow us down to speed us up**. By taking the time to prototype our ideas, we **avoid costly mistakes** such as becoming too complex too early and sticking with a weak idea for too long."*

- Tim Brown, CEO & President of IDEO



Design Sprint

Stanford d.school Design Thinking Process



https://dschool.stanford.edu

Integral part of the design process

Visualization - help UX designers show stakeholders how the final product would look and function.

Feedback - generate feedback from team members as well as test groups of users.

Prototype to **Empathise**, **Define**, **Ideate**, and **Test**

Why prototype in general?

- Evaluation and feedback are central to interaction design
- Developers can test feasibility of ideas with team, users
- Stakeholders can see, hold, interact with a prototype more easily than a document or a drawing
- Team members and users can communicate effectively
- To validate existing / other requirements
- It encourages reflection: very important aspect of design
- Prototypes answer questions, and support designers in choosing between alternatives

What to Prototype and Why

- Prototyping reduces uncertainty
  - It can be a major tool for ***risk management***
  - Apply on whatever you might be uncertain about!
- Prototyping technical issues
  - E.g. run-time issues
- Prototyping to establish requirements
  - Users “see” functionality
- Prototyping for usability concerns

When and at What Level

- For software, you might prototype at various times in the lifecycle
  - Different goals, different techniques

- Conceptual Design
- Interaction Design
- Screen Design

Advantages of prototyping

1. Have a **solid foundation from which to ideate towards improvements** - giving all stakeholders a clear picture of the potential benefits, risks and costs associated with where a prototype might lead.

2. Can **adapt changes early** - thereby avoiding commitment to a single, falsely-ideal version, getting stuck on local maxima of UX and later incurring heavy costs due to oversights.

3. Show the prototype to your users so they can give you their **feedback** to help pinpoint which elements/variants work best and whether an overhaul is required.

4. Have a **tool to experiment** with associated parts of the users' needs and problems - therefore, you can get insights into less-obvious areas of the users' world (e.g., you notice them using it for additional purposes or spot unforeseen accessibility issues such as challenges to mobile use).

5. Provide **a sense of ownership** to all concerned stakeholders- therefore fostering emotional investment in the product's ultimate success.

6. Improve time-to-market by **minimizing the number of errors** to correct before product release.

Prototyping: Conceptual Design

- Early in development
- Explore high-level issues
  - Different conceptual models
  - Interaction styles
  - User needs and characteristics
  - Usability goals
- High-level representations
  - Far from final code or GUIs

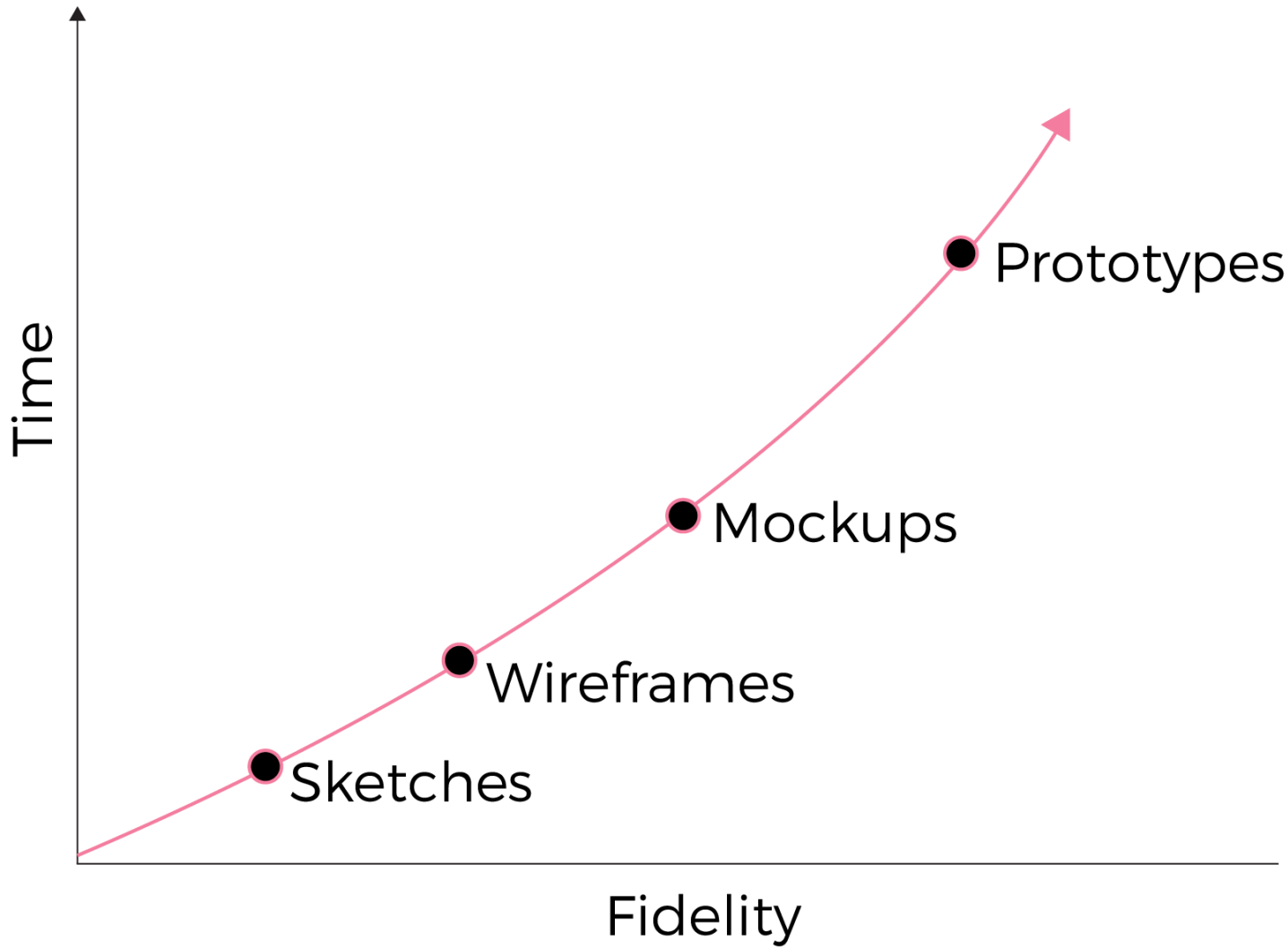
Prototyping: Interaction Design

- Later in development
- Focus on user work-flows
  - Tasks and scenarios identified
- Might focus at the screen (or page) level.
  - identify screens, pages, activities
  - Organize these in groups
  - Define flows or transitions between them
- Involve users in evaluation
- Representations
  - Still probably not much like final code or GUIs

Prototyping: Screen Design

- Before development
- Define and refine screens (pages)
  - Blue-prints for final physical design
- User evaluation
  - Both achieving tasks and navigation, and other usability criteria (as we've studied)
- Representations
  - Low-fidelity or high-fidelity prototypes

Low, mid, high, mixed fidelity prototype



Fidelity-versus-time chart

Low-fidelity Prototyping

- Uses a medium which is unlike the final medium, e.g. paper, cardboard
- Is quick, cheap and easily changed
- Examples:
  - sketches of screens, task sequences, etc
  - ‘Post-it’ notes
  - storyboards

Storyboards

- Often used with scenarios, bringing more detail, and a chance to role play
- It is a series of sketches showing how a user might progress through a task using the device
- Used early in design

Sketching

- Sketching is important to low-fidelity prototyping

- Don’t be inhibited about drawing ability. Practice simple symbols
  - Can use post-its, photo-copied widgets, etc.

Rettig's "Prototyping for Tiny Fingers"

- "To get a good idea, get lots of ideas."
- Problems with hi-fi prototyping:
  - too easy to focus on "fit and finish"
  - developers resist changing software
  - SW prototype sets expectations
  - Bug in SW prototype kills an evaluation

http://fpl.cs.depaul.edu/jriely/360/extras/prototyping-for-tiny-fingers.pdf

Rettig's "Prototyping for Tiny Fingers"

http://fpl.cs.depaul.edu/jriely/360/extras/prototyping-for-tiny-fingers.pdf

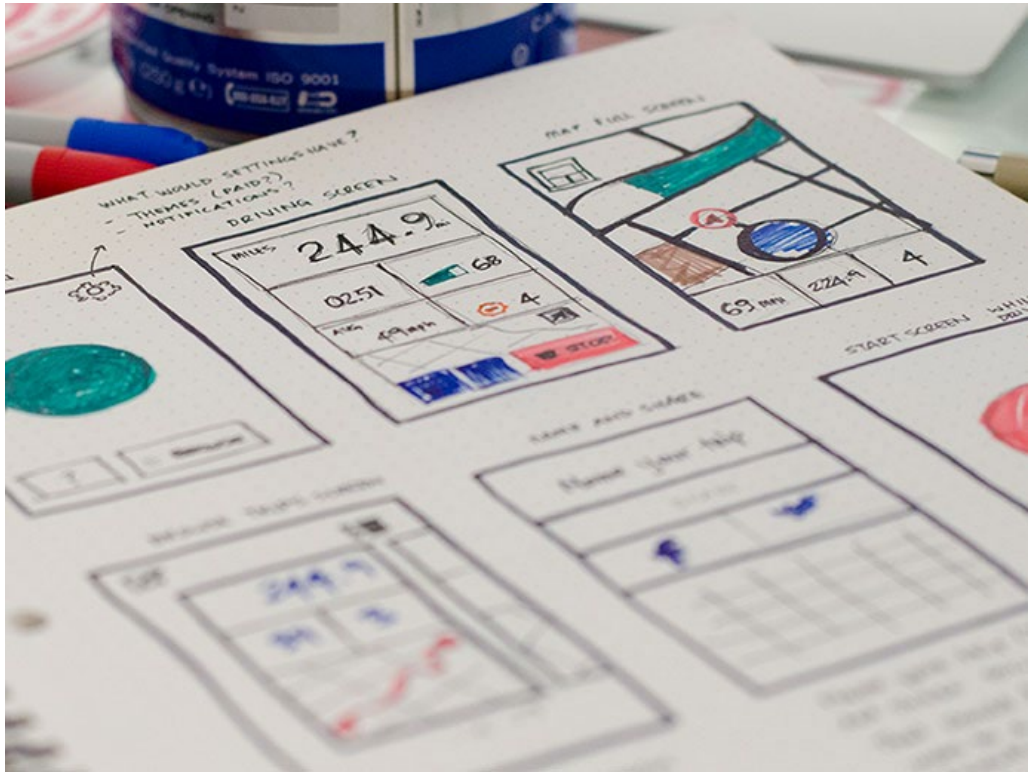
Storyboards

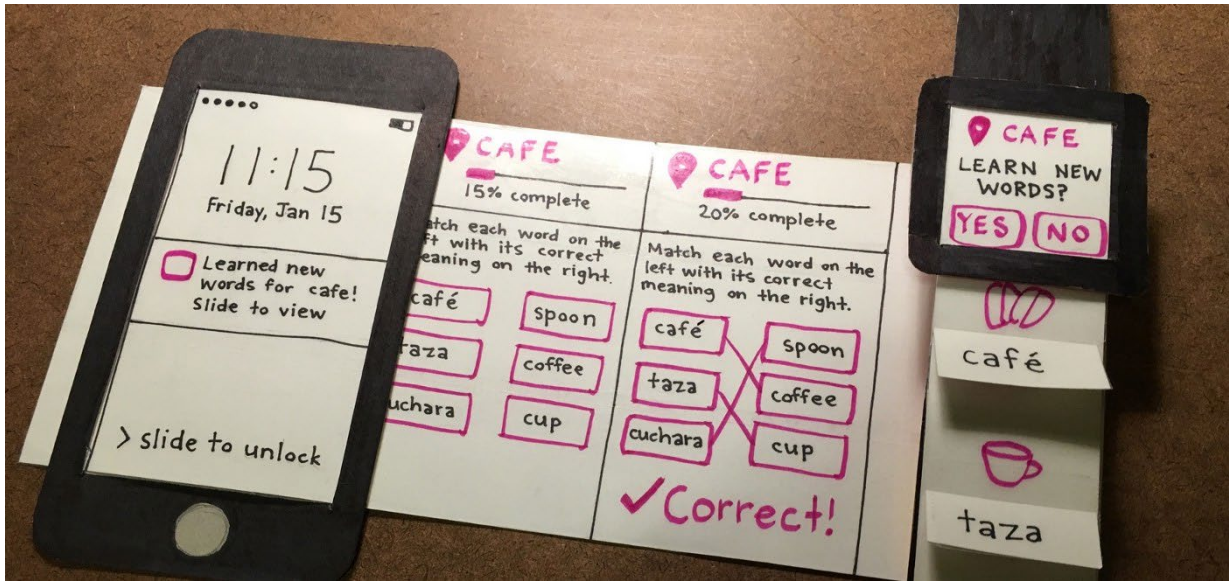
- Storyboards are:
  - a low fidelity visual representation where
  - steps or actions represented by panels, like a comic book
- Goals are to
  - flesh out the scenarios in an interaction design
  - effectively communicate with users or stakeholders

Principles and Variations

- (As usual in HCI) storyboards should be “real” and “representational” rather than “abstract” or “complete”
- Used in different ways at different phases
  - Early: focus on user tasks, work-flow, context, etc.
  - Later: lo-fi drawing of screens, menus, etc.
- Principles:
  - Describe a scenario -- focused on interaction
  - Contains explanations, notes, etc.

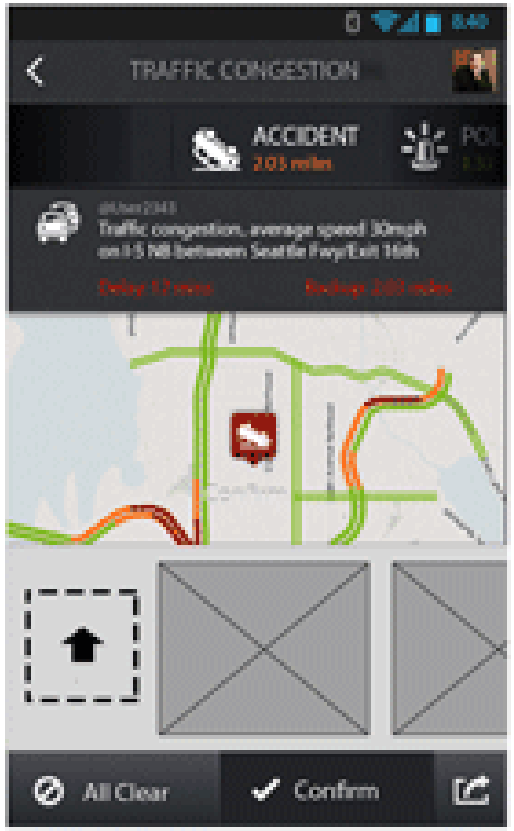
- Lo-fi interface sketches



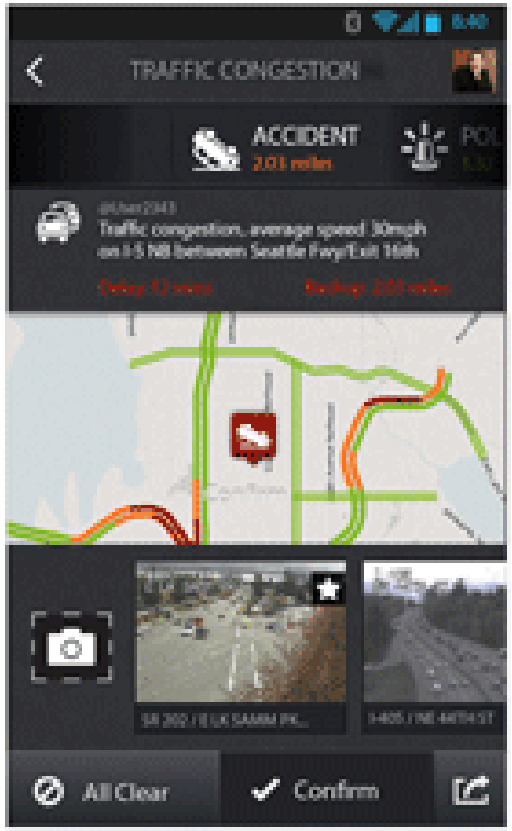


SKETCH

LOW-FI



HI-FI



source: jfarny.com

High-fidelity Prototyping

- Uses materials that you would expect to be in the final product.
- Prototype looks more like the final system than a low-fidelity version.
- For a high-fidelity software prototype common environments include Macromedia Director, Visual Basic, and Smalltalk.
- Danger that users think they have a full system…….see compromises

High-fidelity Prototyping

- Interactive prototypes
- Digital prototypes
- Coded prototypes

- **Benefits**
  - More realistic
  - Closer to final product
    - Good for developers and users
  - Can collect metrics

- **Limitations**
  - More expensive, less rapid
  - Reluctance to change
  - See Rettig’s list

Compromises in prototyping

- All prototypes involve compromises
- For software-based prototyping maybe there is a slow response? sketchy icons? limited functionality?
- Two common types of compromise
  - **‘horizontal’:** provide a wide range of functions, but with little detail
  - **‘vertical’:** provide a lot of detail for only a few functions
- Compromises in prototypes must not be ignored. Product needs engineering

Possible Problems with Prototyping

- Pressure to enhance prototype to become delivered system
  - From client
  - From management
    - Both see code, see almost-working “system”
- Why not use the prototype?
- Prototype built for quick updates, so...
  - No design, so hard to maintain
  - Ugly code, no error checking
  - Wrong environment

And then… Construction

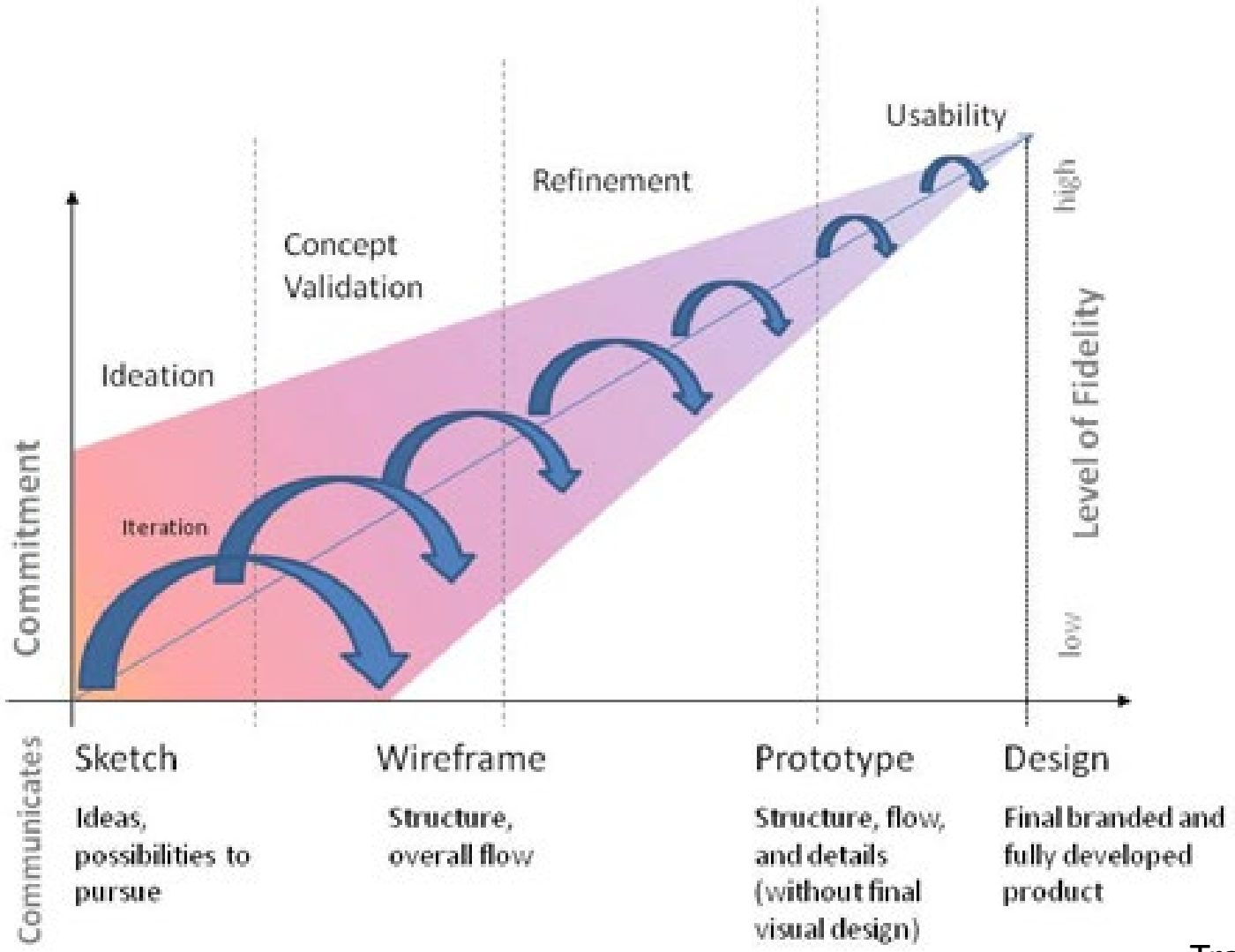
- Taking the prototypes (or learning from them) and creating a final product
- Quality must be attended to: usability (of course), reliability, robustness, maintainability, integrity, portability, efficiency, etc
- Product must be engineered

  Evolutionary prototyping
  'Throw-away' prototyping

“Goldilocks” quality - just the right amount of detail.

Prototype looked too fake > test participants will go into feedback mode instead of reaction mode; what we want is to get honest reactions.

Prototype too detailed and too complex > we risk not finishing the prototype in time.



The Sketch to Design Continuum

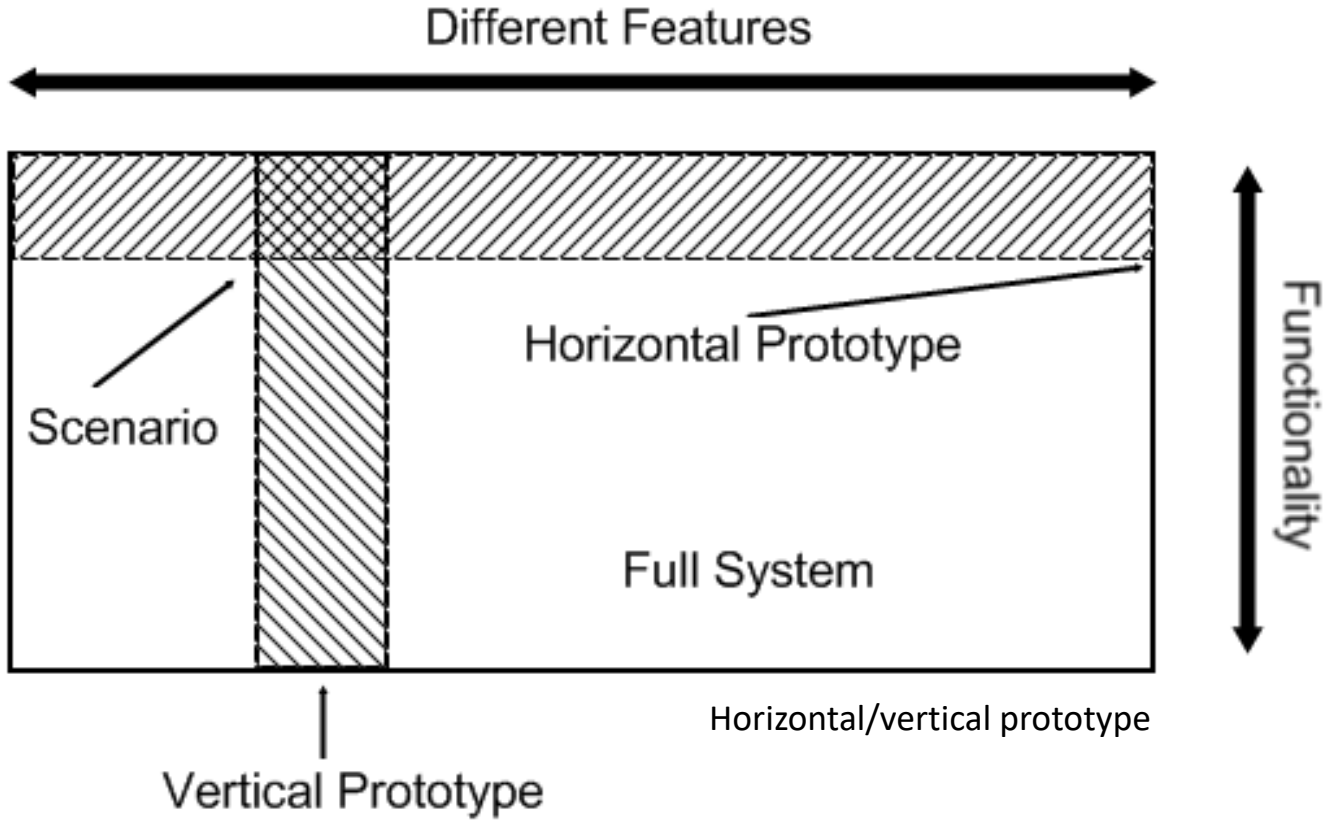
Differences by **visually showing the continuum from sketch to design** by depicting the relationships between what the design is trying to communicate, the amount of iteration between phases, and the fidelity of the design.

Tracy Lepore

Dimensions of Prototype Fidelity by

Carolyn Snyder

1. Breadth
2. Depth
3. Visual
4. Interaction
5. Content



Horizontal/vertical prototype

Area of test	Dimension needed			
	Breadth	Depth	Look	Interaction
Concept and terminology	x	x		
Navigation, work flow, task flow	x	x		
Content	x	x	*sometimes*	
Documentation, help	x	x	*sometimes*	
Requirements, functionality	x	x	*sometimes*	*sometimes*
Screen layout	x		x	
Brand	x	x	x	
Colors, fonts, and graphic elements			x	
Widgets and controls				x
Response time, performance metrics				x
Real world use	All of them plus context of use			

Categories of interface testing and prototype dimensions

The Palm Pilot Story

https://albertosavoia.medium.com/the-palm-pilot-story-1a3424d2ffe4

Thank you!