BIT 2nd Year

Semester 3

IT 3405

User Interface Design
Chapter 9 - Developing Effective Prototype Interfaces

Intended learning outcomes

- Identify the importance of prototyping for acceptable design
- Describe different prototypes and cost required to develop them
- Learn the steps in paper prototyping
- Use a tool to develop a prototype for a given description

Sub Topics

9.1. Overview of prototyping

9.2. Types of prototyping

9.3. Paper prototyping

9.4. Tools for prototyping

9.5. Developing a working prototype

9.1. Overview of prototyping

Definitions of Prototype

- **Prototyping** is the process of quickly putting together **a working model** (a prototype) in order to test various aspects of a design, illustrate ideas or features and gather early user feedback.

- A type of development in which emphasis is placed on developing prototypes early in the development process to permit early feedback and analysis in support of the development process.

Prototyping – Make it clear

- Prototyping is a means of exploring ideas before the real implementation to verify proposed solution

- All designers (crafts people) work on some prototypes
  - Architects create models out of paper or cardboard, or with virtual reality tools
  - Aeronautic engineers use wind tunnels
  - Bridge builders create stress models
  - Software and Web designers create mock-ups of how users will interact with their designs

- **The best reason to prototype is to save time and resources**

Why prototyping

- Get feedback on our design faster
- Experiment with alternative designs
- Fix problems before code is written
- Keep the design centered on the customer
- ……

Why Prototype? - explained

- Traditional software development: you can't test until you implement and Implementation is expensive
- Mistakes are unavoidable in design. Sometimes these mistakes are based on the wrong identification of requirements. However, they are reported as design errors. It is expensive to change the design once implementation is completed.
- design errors, unless they are really bad, are left in the product (as "features")
- User Interface Prototyping is a technique to minimize such possible mistakes that combined with the usability studies and prototypes guide the developers to the right direction.

Need for prototyping

- Enable to explore the problem space with the stakeholders to clearly identify requirements.
- Validate the possible solution or identify alternative solutions by exploring the solution space.
- A vehicle for you to communicate the possible User Interface design(s) of the system.
- Recognize the a potential foundation from which to continue developing the system.
- Validate the requirements identified or assumed in the analysis stage.

Reasons for developing Prototypes

- Some objectives or reasons on examining the needs of a particular project,
  - Minimizing resources for development
  - explore a new interfaces or models
  - Make modifications to one part of the existing design
  - Investigate a new technology

- **Prototype Goal/Objective:**
  - It's important to be clear about reasons for developing the prototype before implementing it

- **Clear goal guides to develop an effective prototype. Three categories of prototype goals are possible**

Categories of prototype goals

1. **Proof of concept. (POC)**
   - Clear some ambiguities or disagreements about the future direction of a project.
   - to prove that an idea or new approach has merit or value.
   - illustrates that an idea works, express its qualities in a visual and interactive way, and/or motivate team members to think about the problem from another perspective.
2. **Design exploration**
   - If you design interactive things, the only way to explore how something will be used.
   - Help the designer to identify how something could work or look
   - Experiment different approaches to solve specific problems

3. **Technical exploration**
   - Analyze or evaluate different technologies. E.g. HTML, Jscript, SQL, DHTML, Win32, or specific coding approaches within each technology that have different tradeoffs.
   - an exploration into what technology will work well to support a certain UI or web features.

Breaking the implementation paradox

- **Build** a prototype of the basic functionality, especially the interface
- **Test** the prototype, which will uncover design errors
- **Correct** the errors
- **Repeat** until you have a clean design

- Prototyping is
  - A major tool for improving usability
  - Heavily used in the industry

Advantages & Disadvantages of Prototyping

Advantages	Disadvantages
Users can try the system and provide constructive feedback during development	Each iteration builds on the previous iteration and further refines the solution. This makes it difficult to reject the initial solution as inappropriate and start over.
An operational prototype can be produced in weeks	Formal end-of-phase reviews do not occur. Thus, its is very difficult to contain the scope of the prototype.
Users become more positive about implementing the system as they see a solution emerging that will meet their needs	System documentation is often absent or incomplete, since the primary focus is on development of the prototype.
Prototyping enables early detection of errors	System backup and recovery, performance, and security issues can be overlooked.

9.2. Types of prototyping

Types of Prototyping

There are two major types of prototyping:

1. Throwaway Prototyping

- refers to the creation of a prototype that will eventually be discarded rather than becoming part of the finally delivered software.
- Main objective is to show the user how it may work in the real system.

2. Evolutionary Prototyping

- The initial prototype is presented to the user. Users provide feedback and suggestions for improvements.
- The developer who then presents a more refined version of the prototype. The user once more provides feedback. The process is repeated.
- This prototype will become a part of final software delivered.

Throw Away Prototype

Main Benefits	Throwaway prototyping can significantly reduce risk.
Keys to Success	Choose the prototyping language that enables quick prototyping and commit to throwing the prototype away.
When to Use	It can be used at any time on a project by any of the project's personnel. Individual project participants can realize some benefit by prototyping risky areas within their individual areas of responsibility.
Main Risks	The main risks of throwaway prototyping are not throwing it away and inefficient use of prototyping time.

Throw Away Prototype …

- Throw Away Prototype is developed from the initial requirements but is not used for the final project.
- Not an alternative for written specification of the requirements
- Some developers believe that this type is a waste of time because it will not be a part of final product
- Whether the prototype is discarded or kept for production, it is important to use an easy to use language.
- The most obvious reason for using Throwaway Prototyping is that it can be done quickly.
- If the users can get quick feedback on their requirements, they may be able to refine them early in the development of the software.

Throw Away Prototype …

- Making changes early in the development lifecycle is extremely cost effective since there is nothing at that point to redo.
- If a project is changed after a considerable work has been done then small changes could require large efforts to implement since software systems have many dependencies.
- Speed is crucial in implementing a throwaway prototype, since with a limited budget of time and money little can be expanded on a prototype that will be discarded.
- Another strength of Throwaway Prototyping is its ability to construct interfaces that the users can test.

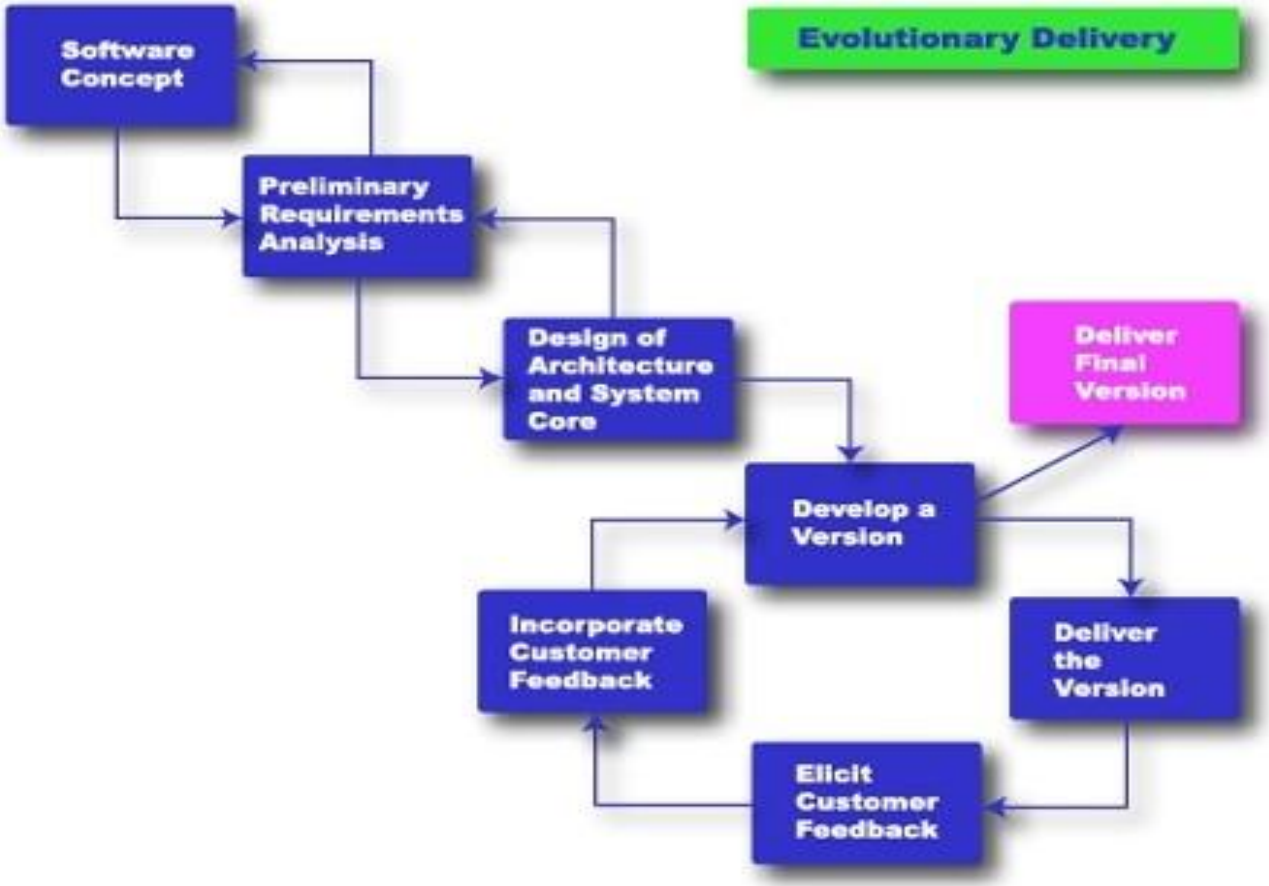
Advantages & Disadvantages - Throwaway prototype

Advantages	Disadvantages
Significantly reduce project risk	The prototype actually does nothing, its just presentational.
Has a short project timeline	Only for a limited purpose
Easier and faster to develop the interface	Starting become a thing of the past. Not getting used as much now.

Evolutionary Prototype

- Also known as **breadboard prototyping.**
- Evolutionary prototyping is considered to be the most **fundamental** form of prototyping.
- Evolutionary prototyping main concept is to build a **robust prototype** and constantly improve it.
- Functional **high fidelity** system.
- Objective to deliver a working system to the end user.
- Evolutionary prototyping acknowledges that we do not understand all the requirements and builds only those that are well understood.
- Allows the development team to add features, or make changes that couldn't be conceived during the requirements and design phase.
- Helps the developer to develop part by part of the system considering the usability aspects of the system.

Evolutionary Delivery



- According to Steve McConnell, "evolutionary delivery is a lifecycle model that straddles the ground between evolutionary prototyping and staged delivery."

Advantages & Disadvantages - Evolutionary Prototype

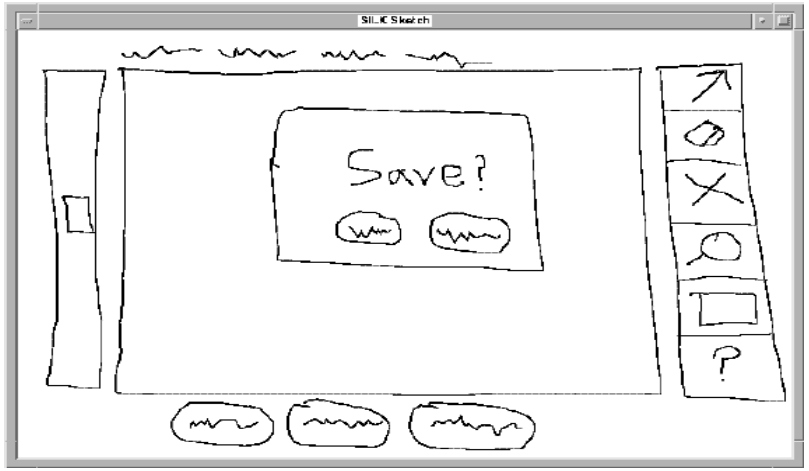
Advantages	Disadvantages
You are always looking for new ways to improve the system.	This method can be used to avoid documenting the requirements of the system.
This model increases the chance of having the client satisfied with the working system.	Management is required
The model can be used even when the requirements are not defined.	Long term maintenance can be expensive
Quicker delivery of the system	Uncertain design idea's
	Information can be lost through so many improvement changes

Fidelity in Prototyping

- Fidelity refers to the level of detail

1. **High fidelity**
   - prototypes look like the final product

2. **Low fidelity**
   - artists renditions with many details missing



Prototyping Approaches

1. Low-fidelity Prototyping

2. High-Fidelity Prototyping

Low-fidelity Prototyping

- Low-fidelity prototyping generally considers the limited functionality and interaction.
- They are constructed to depict concepts, design alternatives and screen layouts. They are intended to demonstrate general look and feel of the interface.
- They are created to educate , communicate and inform, but not to train, test or serve as a basis for which to code.
- Low fidelity prototyping is used early in the design cycle to show general conceptual approaches without much investment in development.

Problems with Low-fidelity Prototypes

- Doesn’t map well to what will actual fit on the screen (realism)
- Couldn’t hold in your hand -- different ergonomics from intended device
- Timing in real-time hard to do
- Difficult to simulate some things (e.g., highlighting)
- Writing on paper not the same as writing on the intended device (realism)
- Appearance unrealistic
- Dynamic widgets hard to simulate (pop-ups)
- Some items had to be static!
- Dragging hard to simulate

High-Fidelity Prototyping

- High-fidelity prototypes represent the core functionality of the products user interface.
- High fidelity prototypes are fully interactive systems. Users can enter data in entry fields, respond to messages, select icon to open windows and interact with user interface as if it were a real system.
- They trade-off speed for accuracy.
- Building high fidelity prototypes consume resources and have high cost.

When to use Hi-Fi Prototyping

- For testing user interface issues
- For demonstrating product to potential customers (internal or external)
- To clarify the specification
- Represent core functionality of product's user interface
- Can be so realistic that user can't tell them from product.
- Fully interactive
  - possible to enter data
  - use widgets
- Simulate functionality of final product
- Can be horizontal or vertical or both

Creation of Hi-Fi Prototypes

- Trade off speed for accuracy
- More difficult and time consuming to create than low-fi prototypes
- Often vertical, due to need to test certain features
- Address issues related to navigation and flow
- Serve as a living specification

Comparison of two prototyping efforts

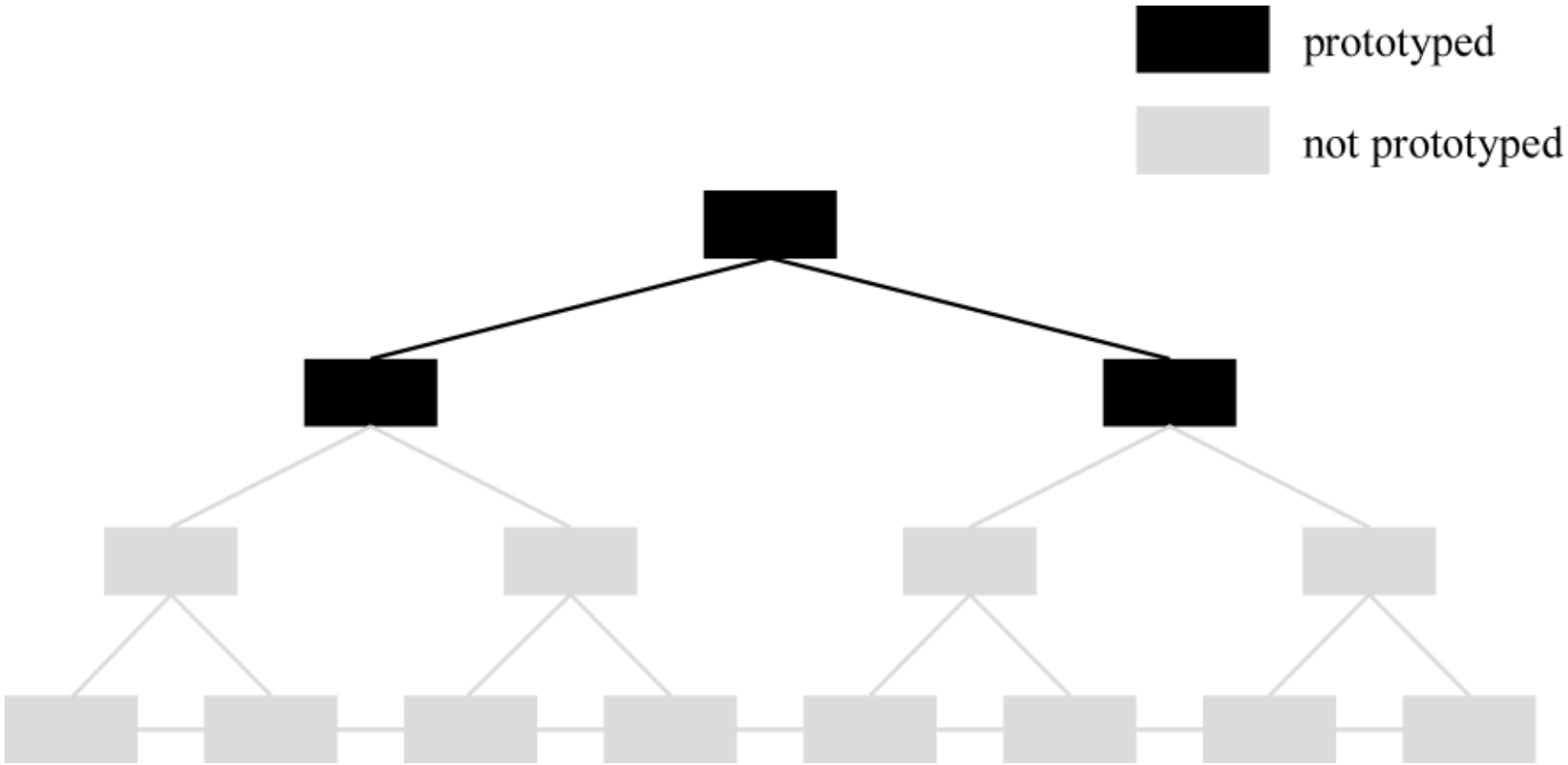
Type	Advantages	Disadvantages
Low-fidelity Prototypes	• Lower Development cost • Evaluate multiple design concept • Useful communication device • Address screen layout issues • Address screen layout issues • Useful for identifying requirements	• Limited error checking • Facilitator driven • Limited utility after requirements established • Navigational and flow limitations • Poor detailed specifications to code
High-fidelity Prototypes	• Complete functionality • Fully interactive • User Driven • Clear definition of navigation • Look and feel of final product • Use for explorations and tests	• More expensive to develop • Time-consuming to create • Not effective for requirements gathering

Scope of prototypes

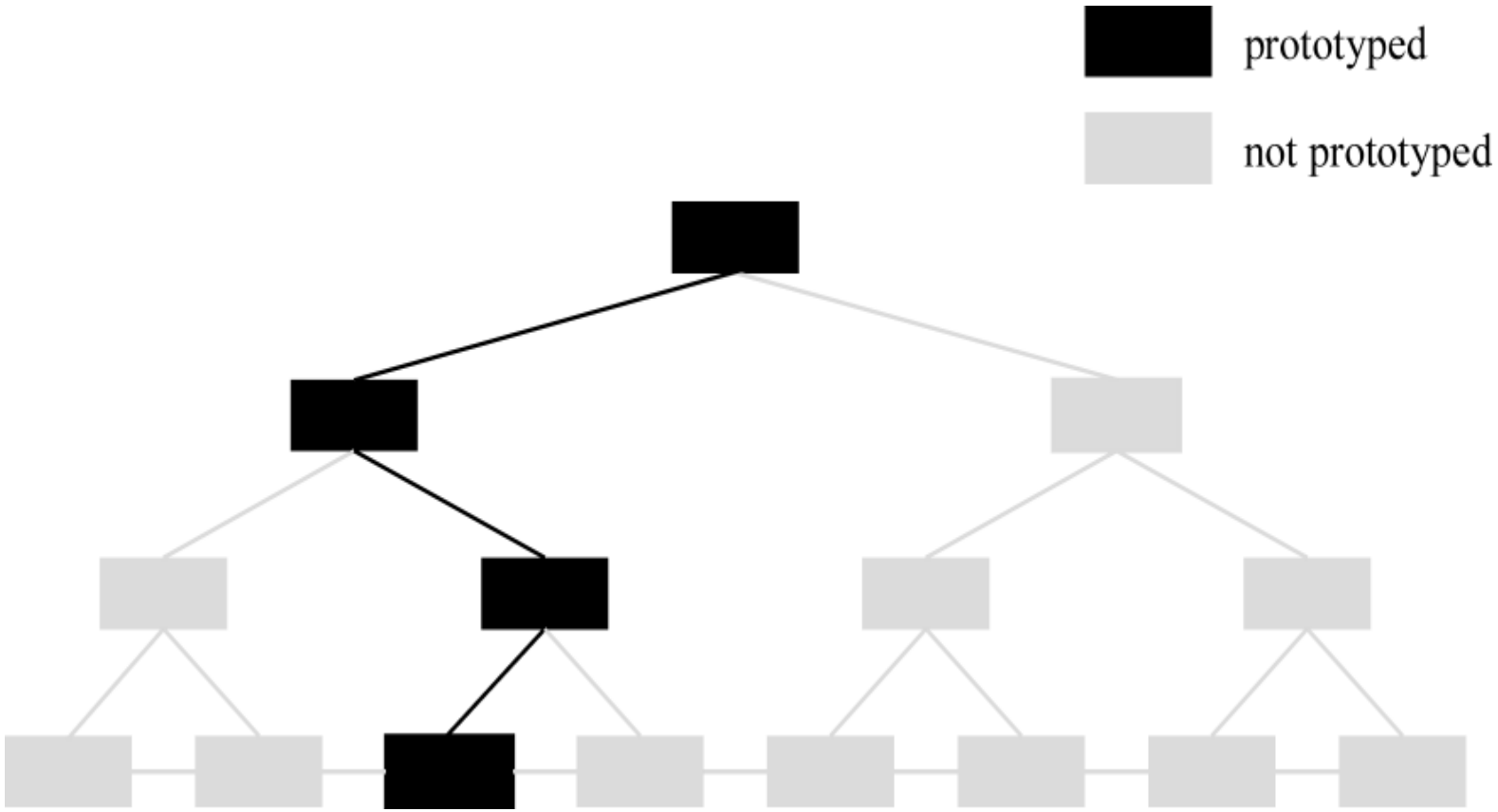
1. Horizontal prototype

2. Vertical prototype

Horizontal prototype: broad but only top-level



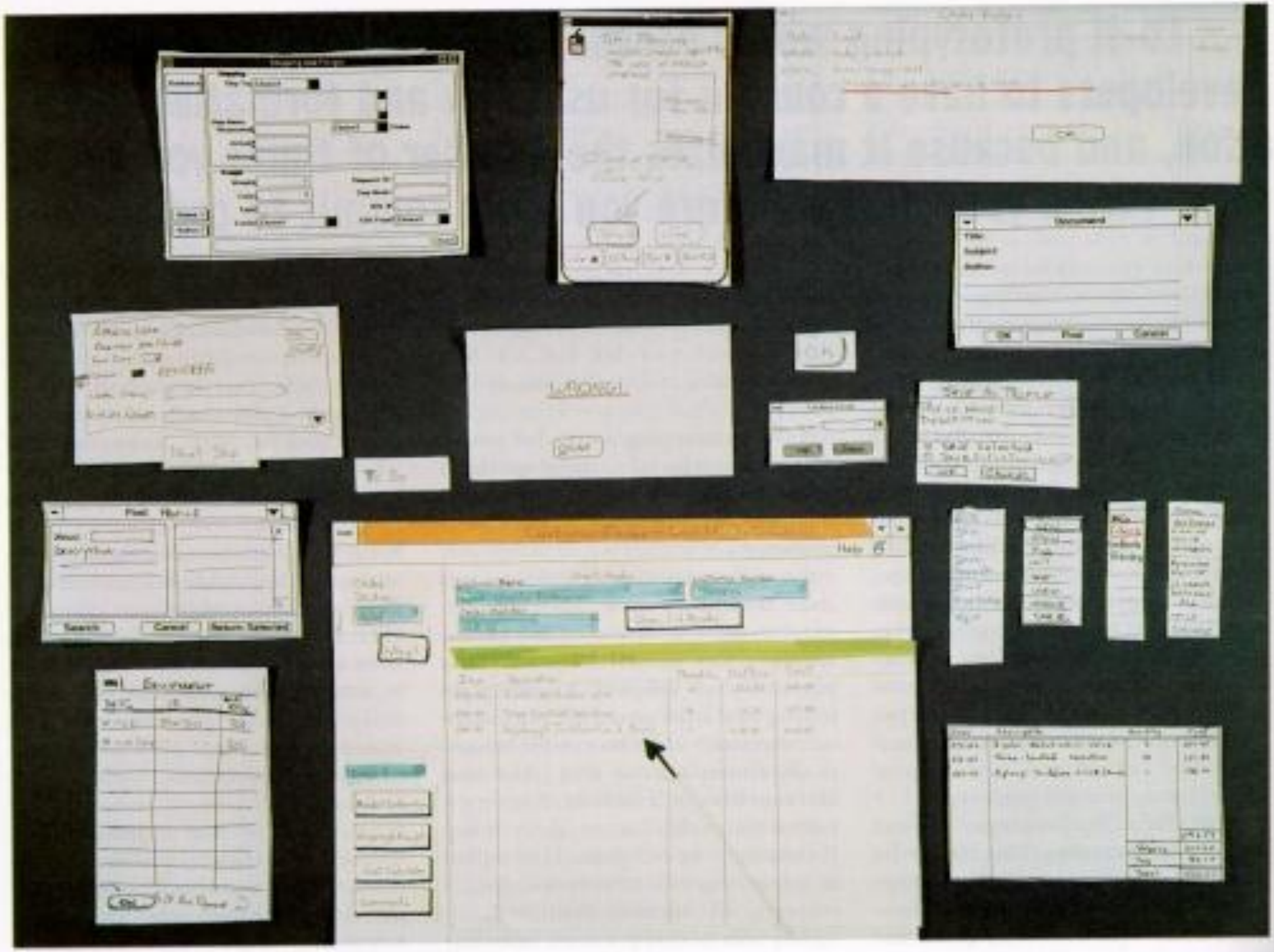
Vertical prototype: deep, but only some functions



9.3. Paper prototyping

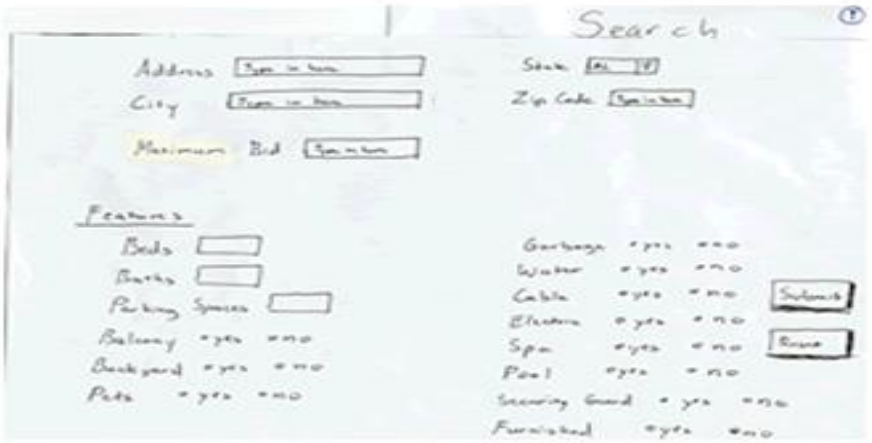
Paper Prototypes

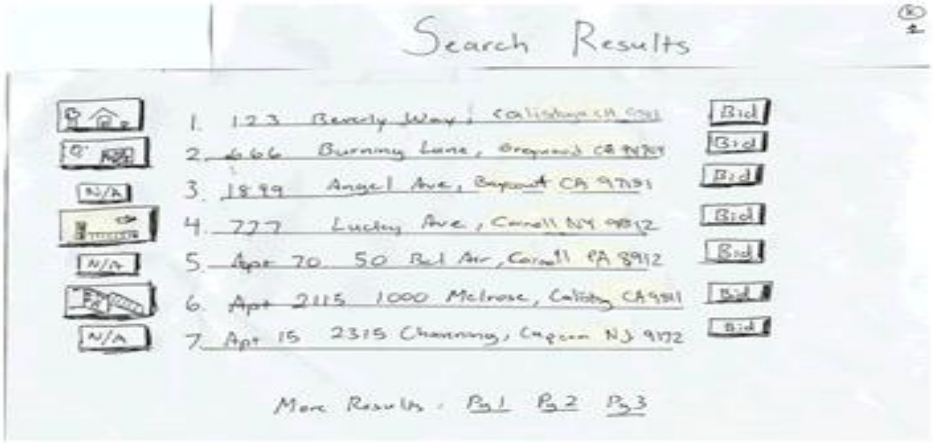
- It is a **low fidelity Throwaway Prototype.**
- Developed using paper and pencils.
- *Paper prototyping is a variation of usability testing where representative users perform realistic tasks by interacting with a paper version of the interface that is manipulated by a person 'playing computer,' who doesn't explain how the interface is intended to work.*

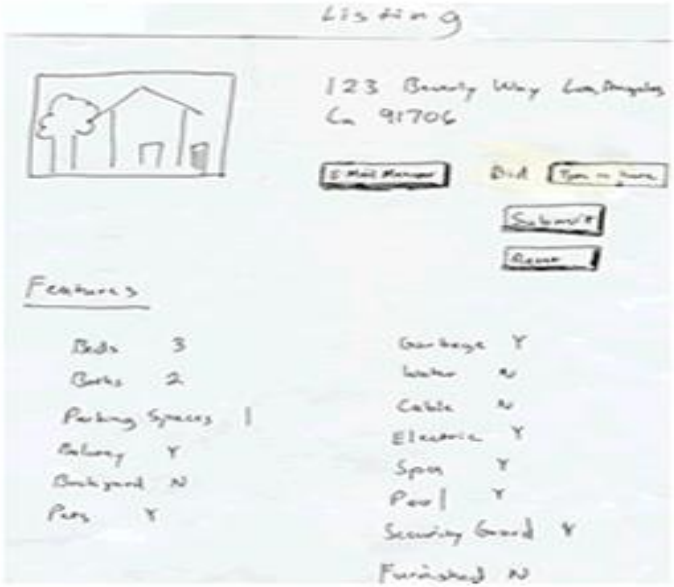


Paper Prototypes (Sketches)

Rentaliy for Renters-Searching for Housing



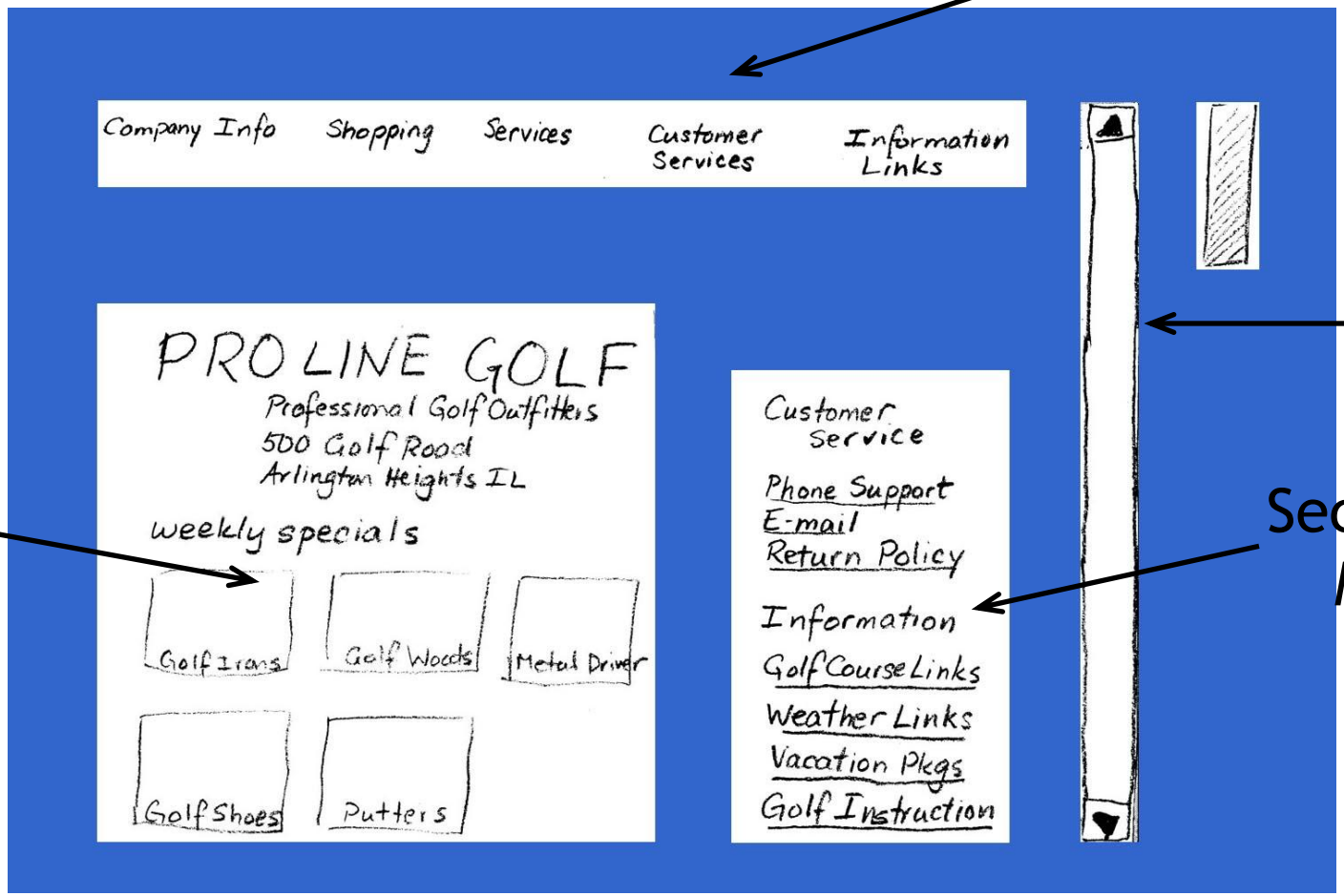




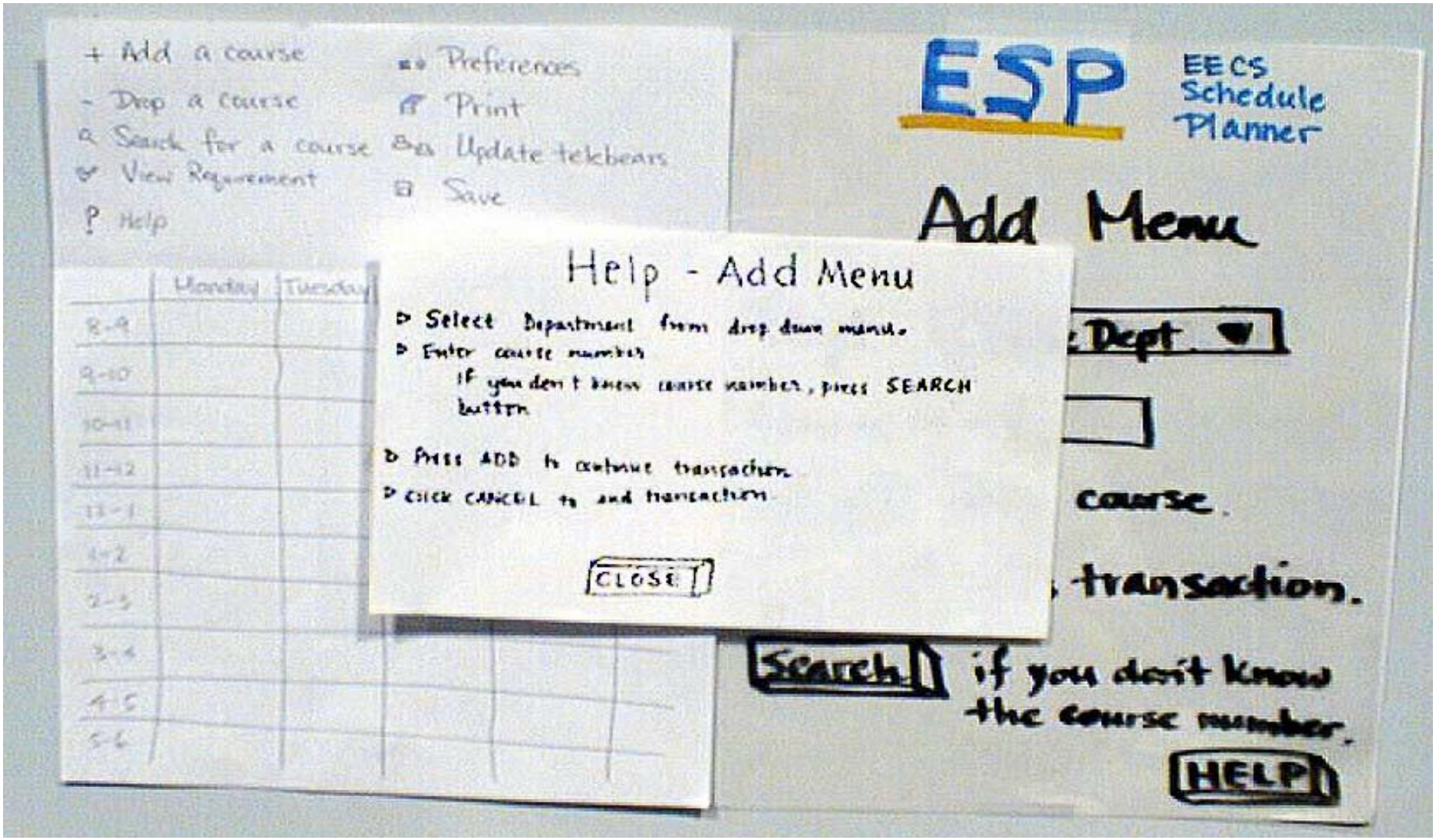
The Basic Materials for Paper Prototyping

- Large, heavy, white paper (11 x 17)
- 5x8 in. index cards
- Tape, stick glue, correction tape
- Pens & markers (many colors & sizes)
- Overhead transparencies
- Scissors, X-acto knives, etc.

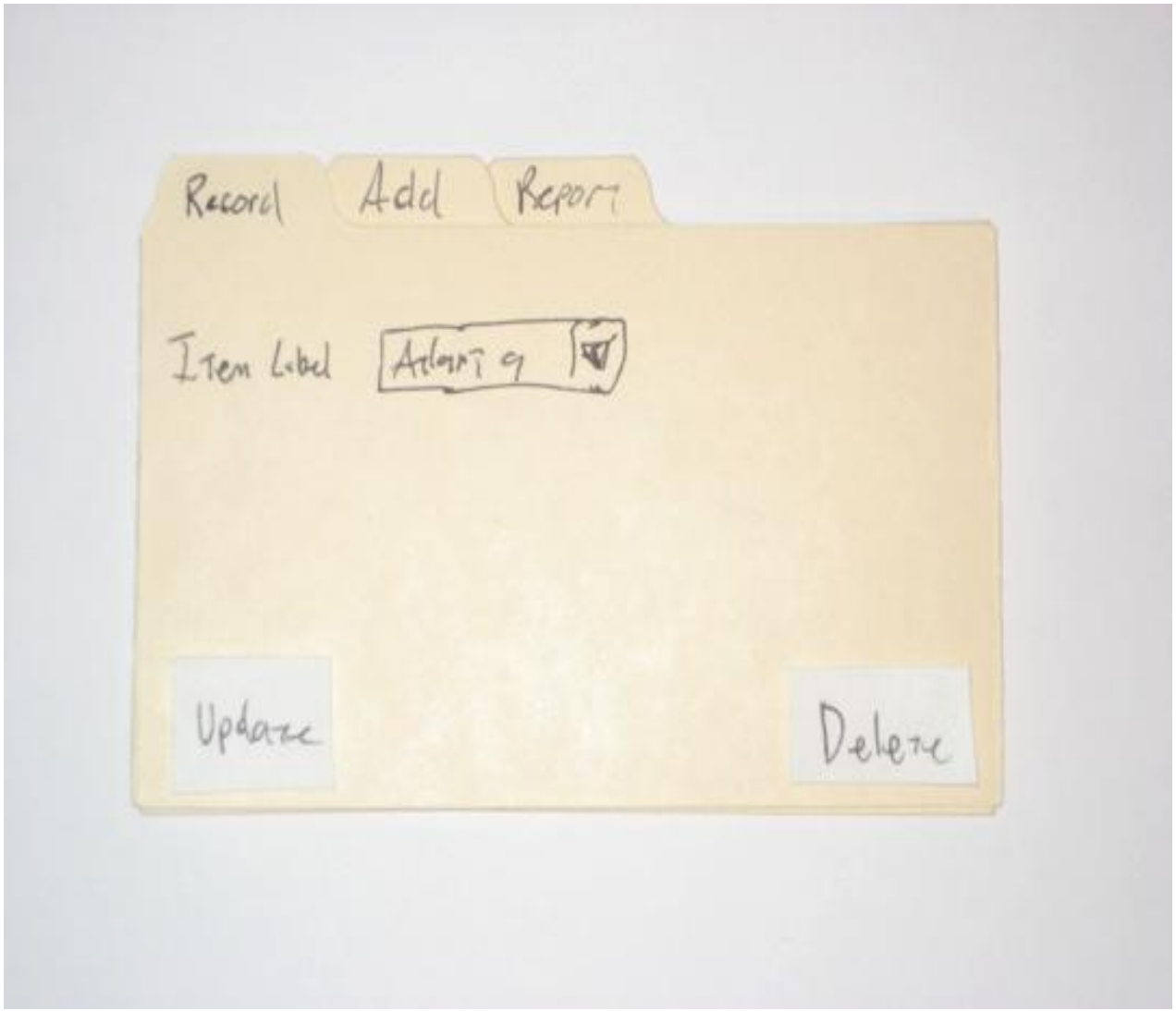
Elements of a paper prototype



Examples – Sketching the prototypes

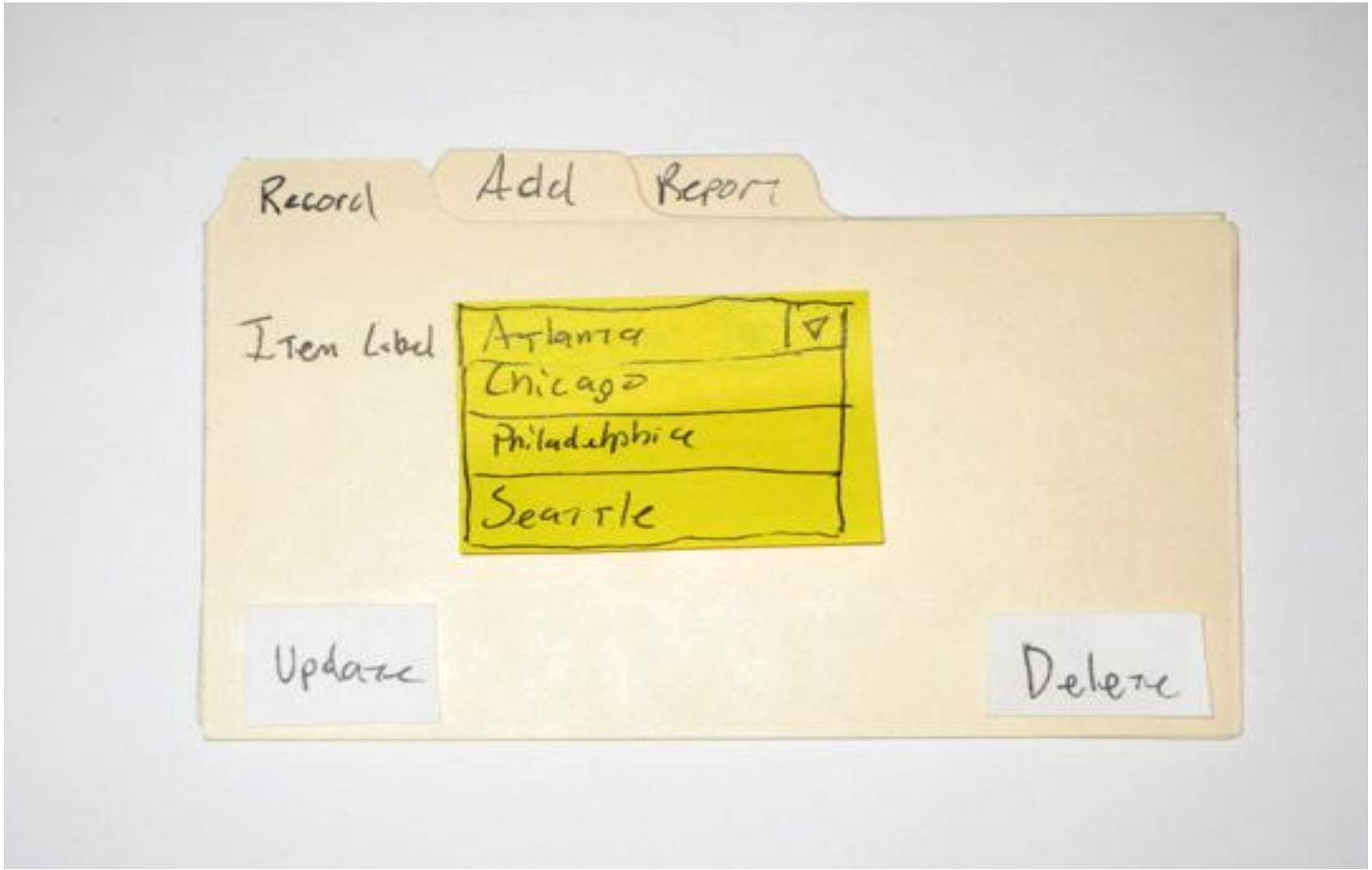


Examples - Paper Prototyping …

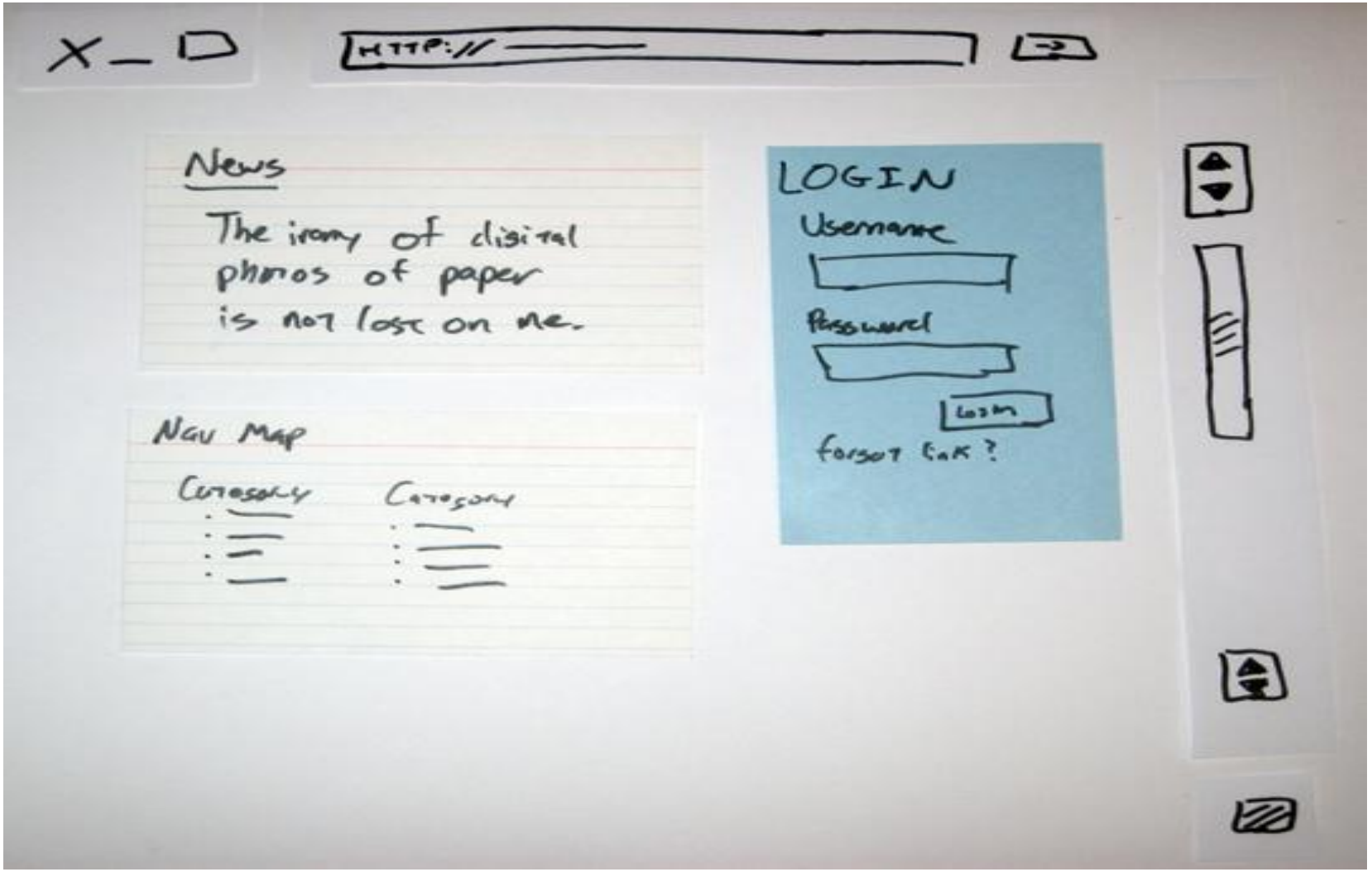


- Traditional user interface widgets can be modeled easily with paper.
- Here is a set "tabs" showing a couple of buttons and what happens when you "click" on a drop-down menu:

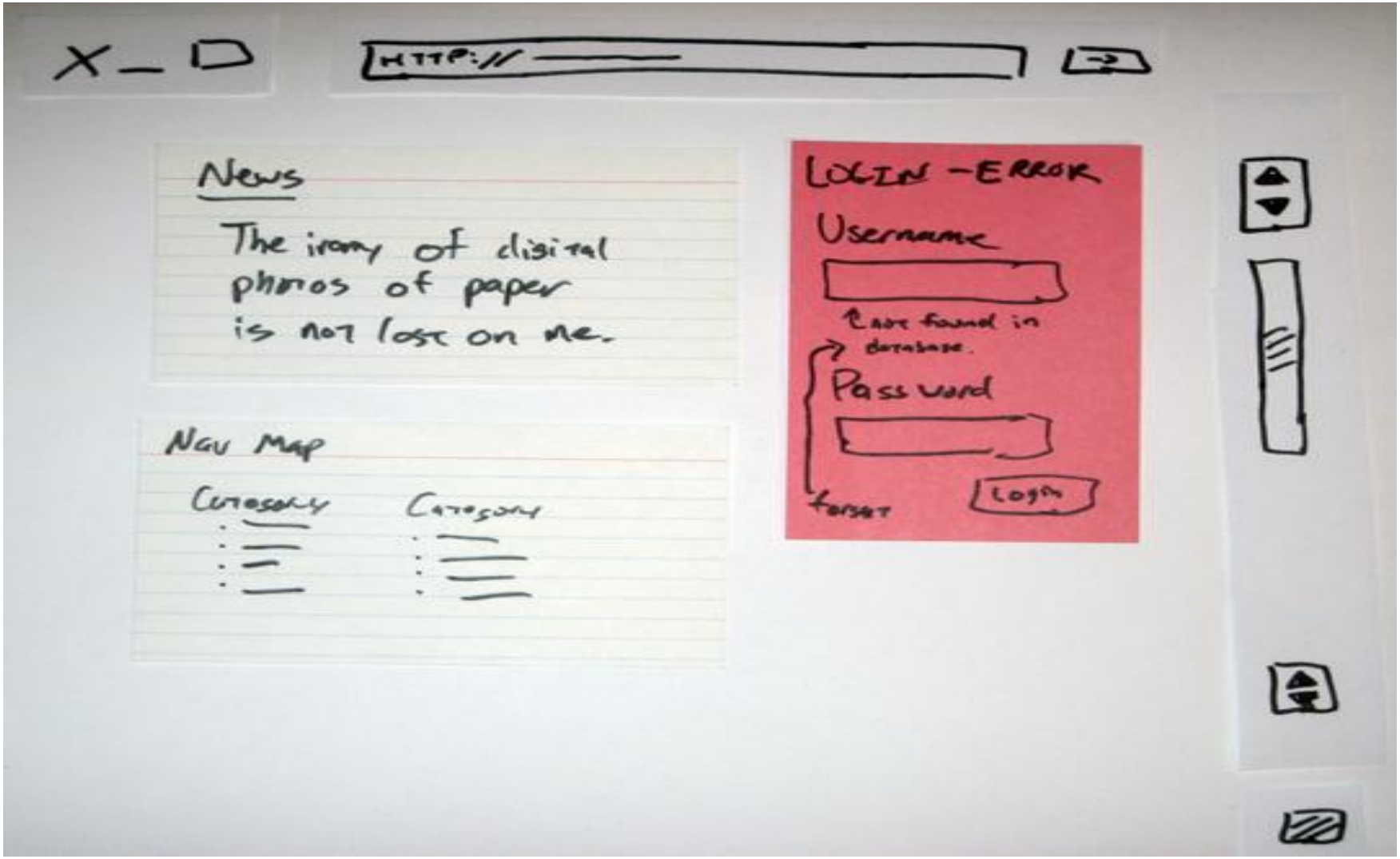
Examples - Paper Prototyping



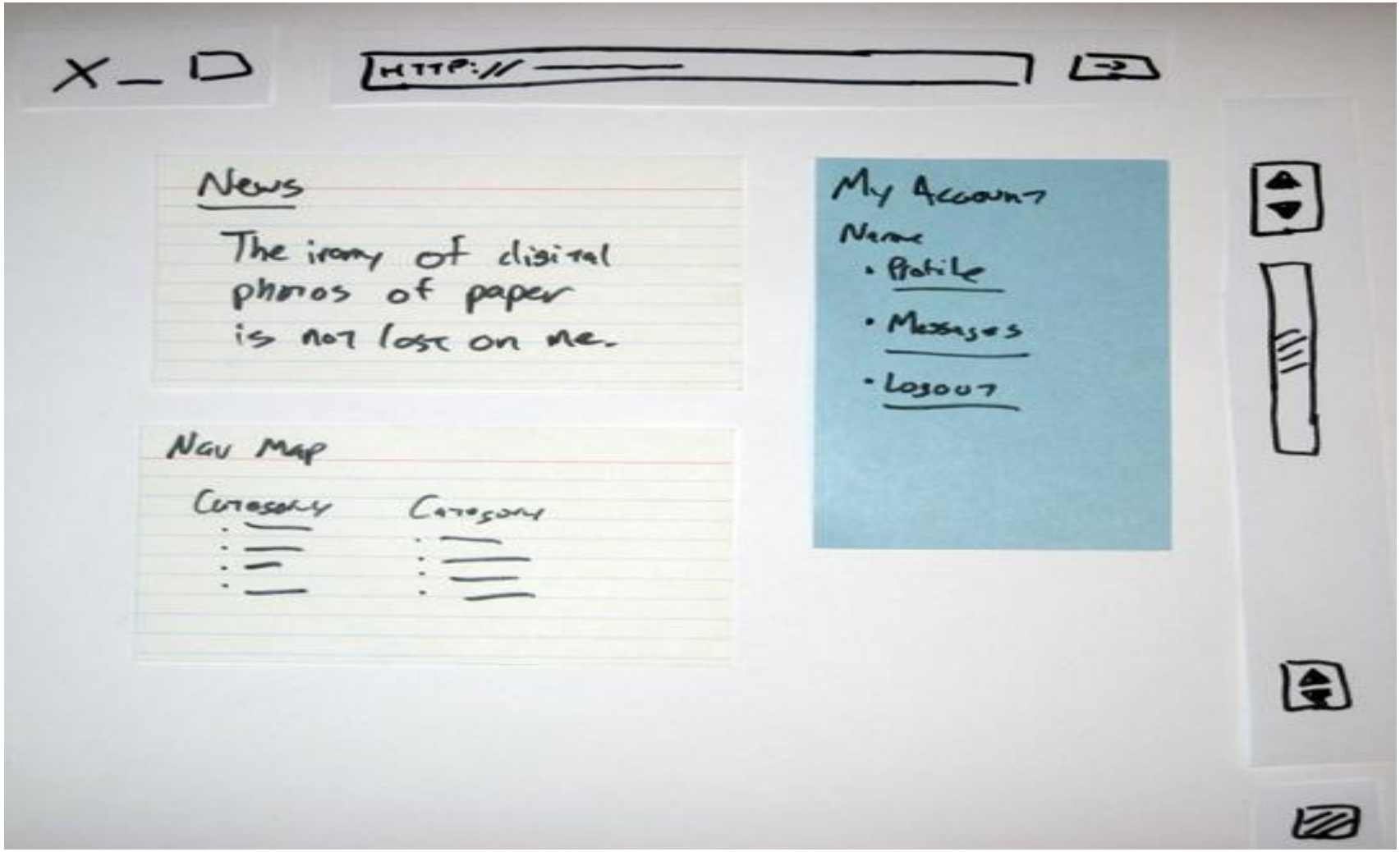
Examples - Paper Prototyping

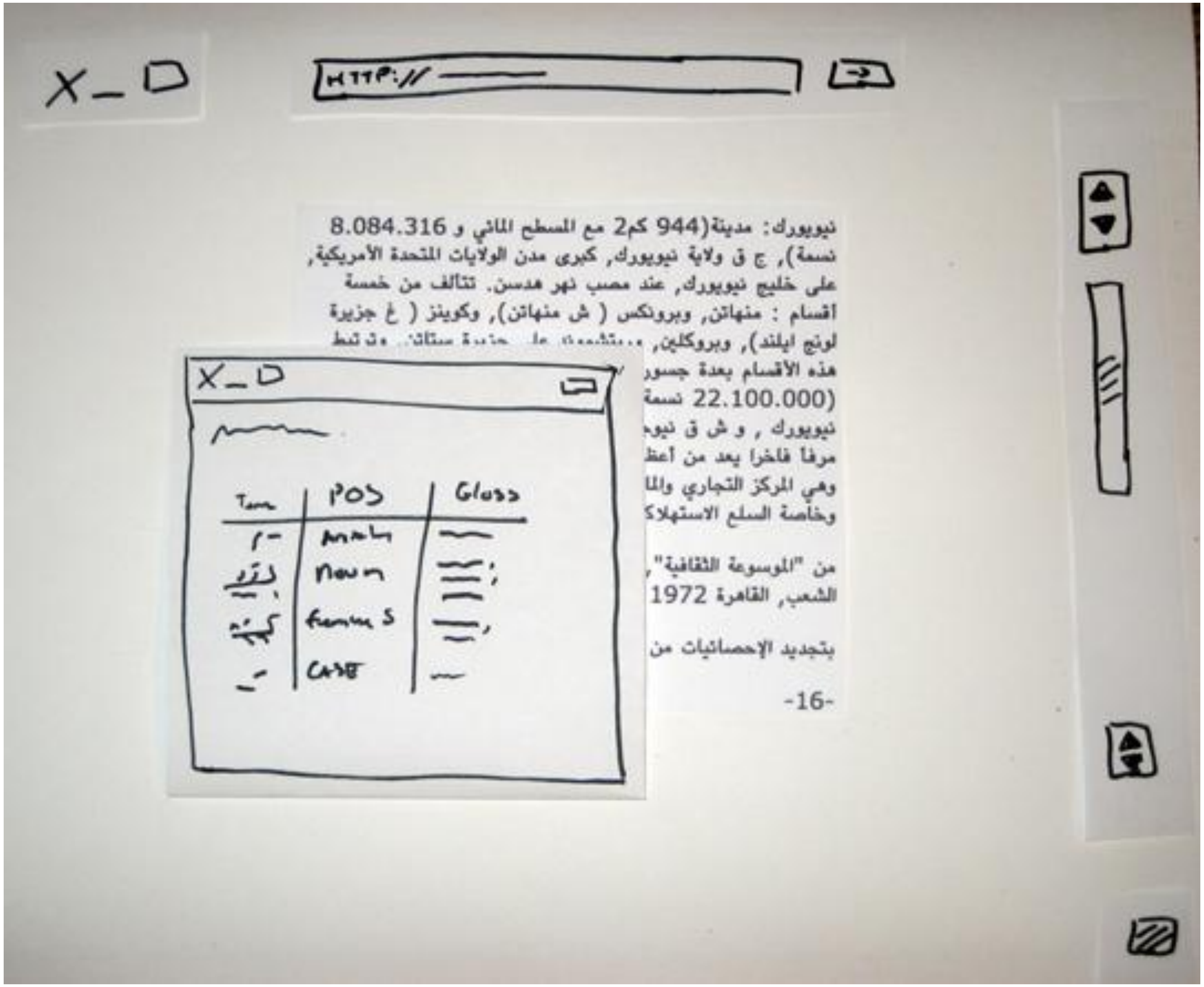


Examples - Paper Prototyping



Examples - Paper Prototyping





- It is possible to demonstrate problems such as a pop-up windows that block key elements of your interface and potential alternatives to the pop-up:

Advantages & Disadvantages - Paper Prototyping

Advantages	Disadvantages
support brainstorming	do not evolve easily
do not require specification of details	lack support for “design memory”
designers feel comfortable sketching	force manual translation to electronic format
	Limited end-user interaction

9.4.
Tools for prototyping

Tools for Prototyping

- Tools are used to quickly create prototypes simulating key aspects

- Different types of tools
  - General Purpose Tools – e.g. Powerpoint
  - Wireframing (sketiching) –e.g.
  - UI Animation (advanced wireframing) –e.g MS Visio
  - UI builders – e.g. Visual Basic, Dreamweaver

- Scripting language of the tool helps to improve the value of tool.

Why use Prototyping Tools (rather than writing the real code)?

- Faster
- Easier to incorporate testing changes
- Multiple UIs for same application
- Consistent user interfaces
- Easier to involve variety of specialists
- Separation of UI code from app. code
  - easier to change and maintain
- More reliable

Basic Prototyping Tools vs. UI Builders

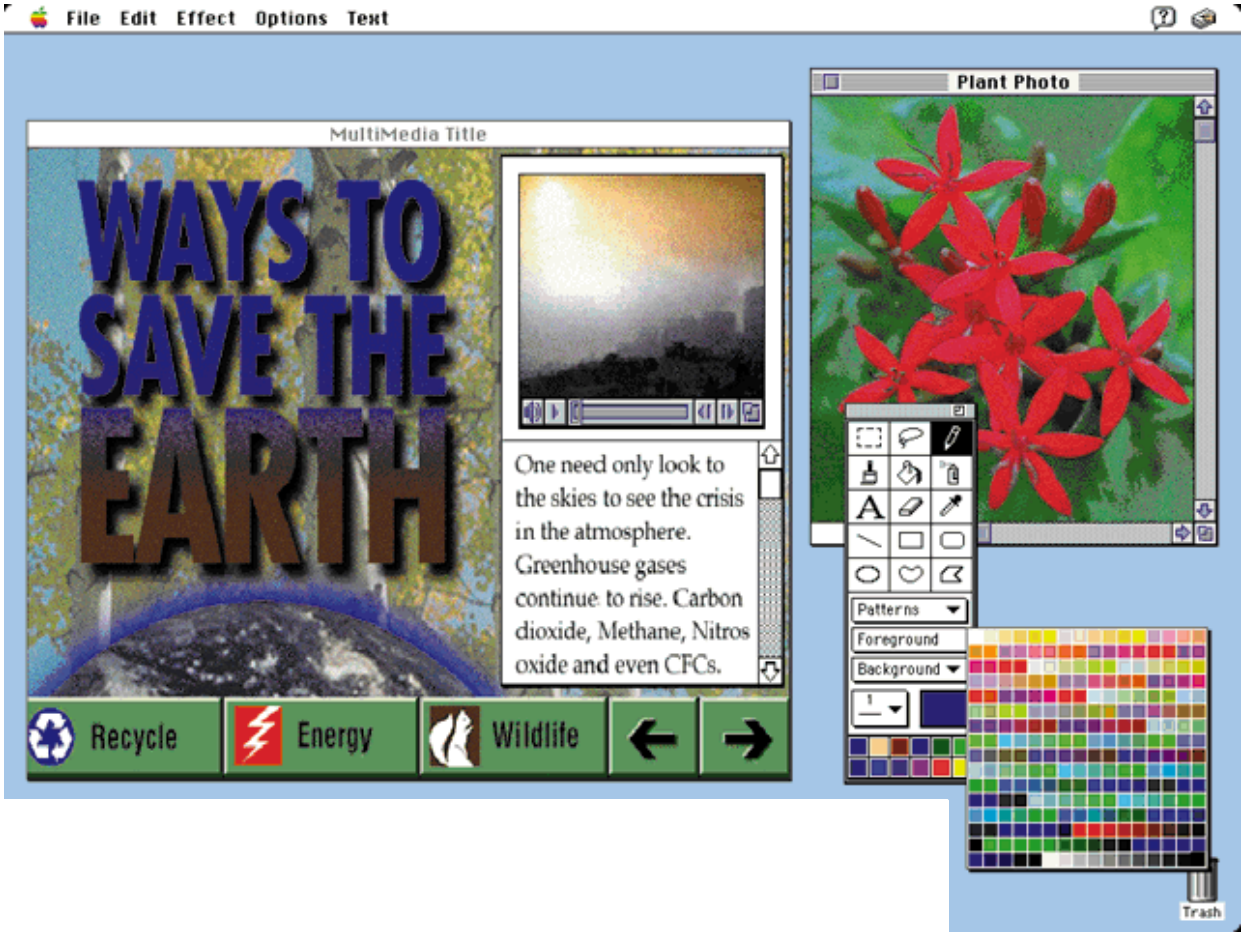
- **Prototyping tools**
  - Visio, Dreamweaver, JustinMind etc.
  - Lay out the design of the system
- **UI builders/toolkits**
  - Create the code that underlies the UI in a real application
    - examples: visual basic, tcl/TK, Java GUI builders (visual café),
- **Sometimes the difference is due to types of prototyping**
  - Throwaway prototype
  - Evolutionary prototyping

Old Prototyping Tools

- **Director**
  - most commonly used by designers
  - intended for multimedia -> lacks widgets
  - good for non-widget UIs or the “insides” of app
- **HyperCard**
  - metaphor: card transitions on button clicks
  - comes with widget set
  - drawing & animation more limited
- **Both have “scripting” languages**

HyperCard

- Tool palettes



General Purpose tool - PowerPoint

- **pros**
  - can be used as a kind of storyboard
  - decent drawing package
  - animation features are easy to use and fairly flexible
  - the notes page feature can comment on storyboard

- **cons**
  - Not much interaction facilities

    e.g. if users clicks on button A, bring up window Wa, otherwise bring up window Wb.
  - No direct scripting language (possible to use embedded objects)

Macromedia/Adobe Dreamweaver

- Activity: Discuss how Dreamweaver could be used as a prototyping tool, its benefits and limitations

UI Builders

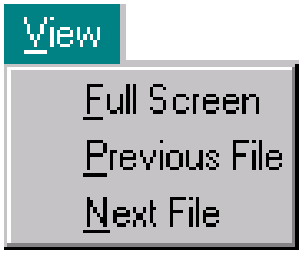
- **Visual Basic, Java, ……..**
  - lots of widgets (AKA controls)
  - simple language
  - slower than other UI builders
  - widgets sets
  - easily connect to code via “callbacks”
- **Programming ability**
  - prototyping tools usually don’t require much
  - UI builders usually do
- **Performance**
  - prototyping tools produce slow programs
  - UI builders depend on underlying language
- **Widgets**
  - prototyping tools may not have complete set
  - UI builders have widget set common to platform

Some Widgets used in prototypes

- Buttons (several types)

- Scroll bars and sliders

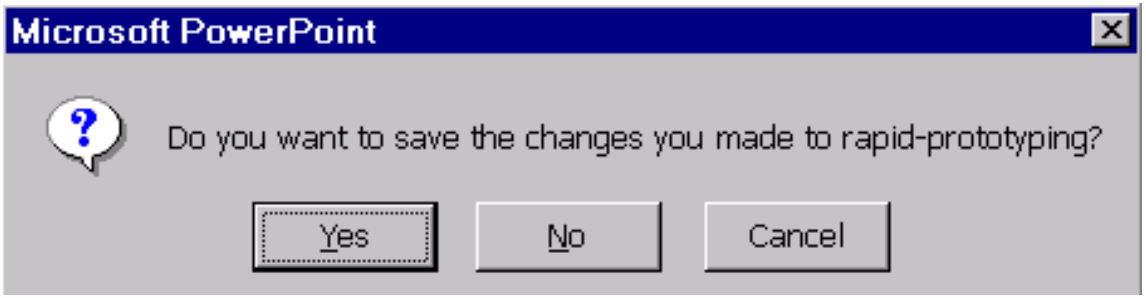
- Pulldown menus



Some Widgets used in prototypes …

- Palettes

- Dialog boxes



- Windows

New Tools for prototyping

- **Justinmind**
  - http://www.justinmind.com/overview
  - http://www.youtube.com/watch?v=dJgCHAJ0r9E
- **Similar tools**
  - http://blog.profitbricks.com/top-29-mockup-and-wireframing-tools-for-developers/
  - http://www.sitepoint.com/tools-prototyping-wireframing/

9.5. Developing a working prototype

Developing a Working Prototype

1. Work with real users
2. Collaborative Development
3. Domain understanding
4. Down to earth
5. KIS - Keep It Simple
6. Follow WYSIWYG
7. Walkthrough: Get an expert opinions
8. Prototype is not the real system
9. Consistency across the prototype
10. Working Metaphor

Developing a working prototype

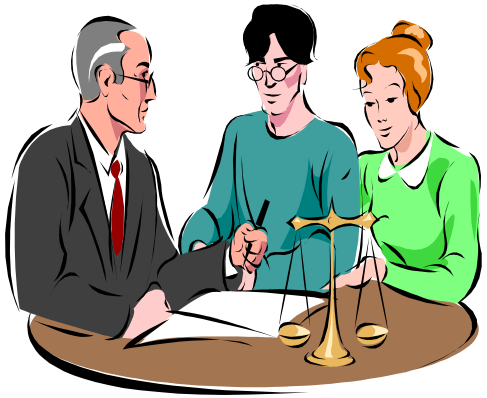
1. Work with real users

- Prototyping is not practical if a sample of end users is unable to participate to evaluate the prototypes
- Both developers and evaluators (representative users) should have some experience in prototyping concept and process, (otherwise it may lead to some misunderstandings)
- Committeemen from both parties (developers and users) are important when developing and evaluating prototypes

Developing a working prototype

2. Collaborative Development

- Get your users stakeholders to work with the prototype.
- If it is possible, invite some candidate users to work with you when drafting or developing the prototype
- In this way, users can quickly determine whether the system will meet their needs.
- A good approach is to ask them to work through some use case scenarios using the prototype as if it were the real system.



Developing a working prototype

3. Domain understanding

- Without a proper understanding about the background, such as underlying business, types of users and real infrastructure, it is hard to develop a prototype that will support to achieve goals/objectives
- Semantics of requirements depends on the working environment
- Design and develop the prototype that supports the culture of business and users in the organization
- Once again, active stakeholder participation is critical to your success.

Developing a working prototype

4. Down to earth

- Consider prototype features that you can actually build.
- Christmas wish lists are for kids.
- If you cannot possibly deliver a particular functionality, do not prototype it.

Developing a working prototype

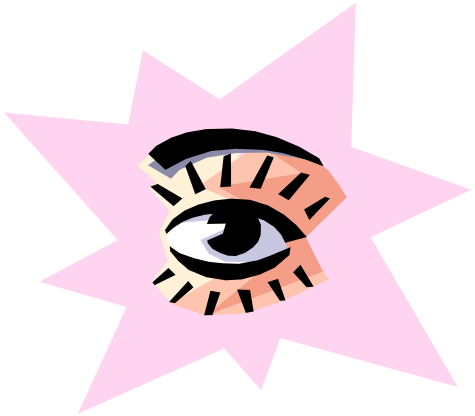
5. KIS - Keep It Simple

- Not only the final system, your prototype too should be simple but You cannot make everything simple.
- Sometimes your software will be difficult to use because the problem it addresses is inherently difficult.
- Your goal is to make your user interface as easy as possible to use, not simplistic.

Developing a working prototype

6. Follow WYSIWYG

- Follow the concept “WYSIWYG”, “What You See Is What You Get,” and utilize it to make sure from tentative users that
- WYSIWYN, "What You See Is What You Need.”
- Interface is the facilitator who guides the user to complete the work within the shortest possible way.
- Hence, interface should make some interest /motivate users to get what they want



Developing a working prototype

7. Walkthrough: Get an expert opinions

- Get an interface expert to help on prototype design
- End users are not experts in the domain and they present things based on their views and requirements.
- It is better to obtain the assistance from User Interface specialist, a walkthrough, to make sure that you do things in the correct way

Developing a working prototype

8. Prototype is not the real system

- Explain what a prototype is.
- It is natural to have misunderstandings between end users and development team, after going through the prototype.
- They may believe the work is completed or the work is a simpler thing than explained by developers.
- users may not realize more work is left to do on the system after going through the prototype.
- To avoid this problem, point out that your prototype is like a Styrofoam model that architects build to describe the design of a house. Nobody would expect to live in a Styrofoam model, so why would anyone expect to use a system prototype to get his job done?

Developing a working prototype

9. Consistency

- Consistency across the prototype
- Yes it is a common rule in the usability of any system
- Inconsistent user interfaces lead to less usable software,
- At the same time, it will make more programming, and higher user support and high training cost

Developing a working prototype

10. Working Metaphor

- Use your user language, not the technical terminology to present your prototype
- Development team may be more familiar with technical words when presenting things but it may not be something understandable to users
- You need to craft your prototype by mapping concepts in user working environment to interface elements when designing any interface
- Avoid implementation technology/ terminology to present things.
- e.g. “Database” is more understandable than saying “mysql database”

Advantages & Disadvantages - Developing a working prototype

Advantages	Disadvantages
Can be used to test every detail of the final product before the product is built (E.g. MoS testing rooms)	Users may be unfamiliar with the technique.
Results in higher user satisfaction	Management may think that the project is nearly finished if the prototype is “too good,” or that the prototype can be converted into the final product.
Users are better at evaluating an existing (vs described) system	
It brings the users into the process early	