Software Requirements

L3. Prototyping

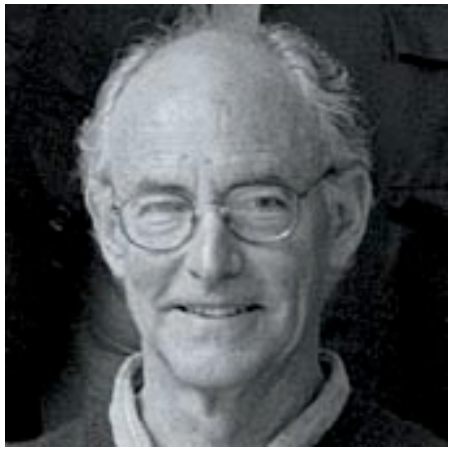
Christoph Stanik

Prof. Walid Maalej

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

What is prototyping?

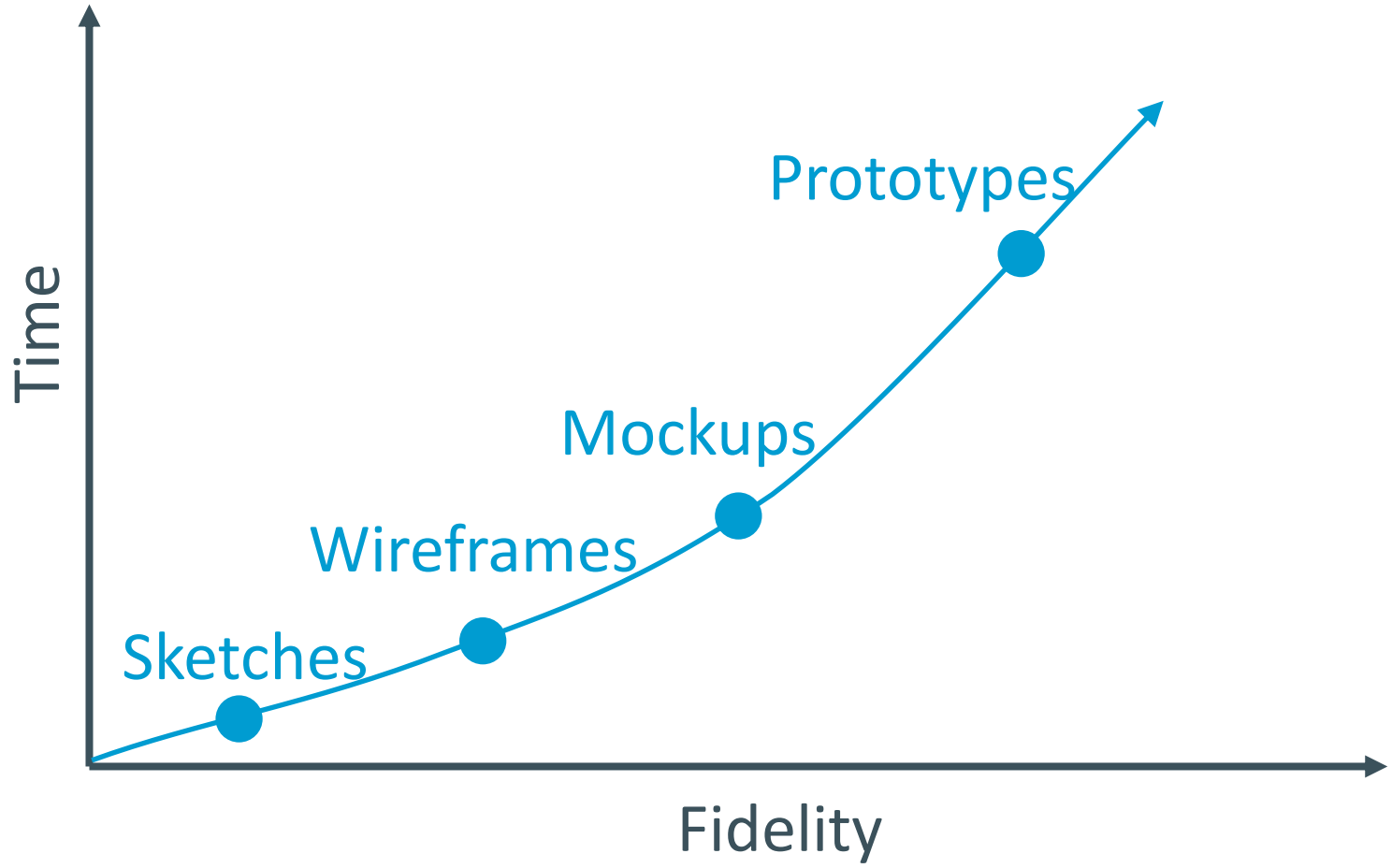
"Prototyping is externalizing and **making concrete** a design idea for the purpose of **evaluation**" – Bill Verplank



Prototype

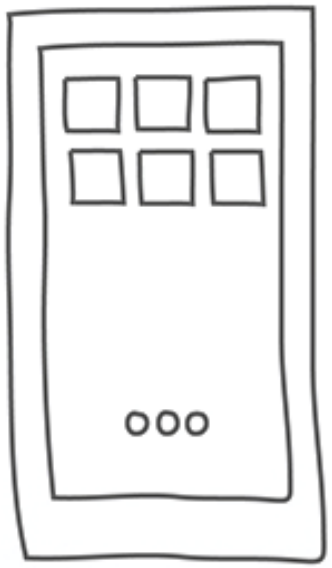
- is a model of a real system
- hides details to reduce complexity
- is either functional, structural, or behavioral

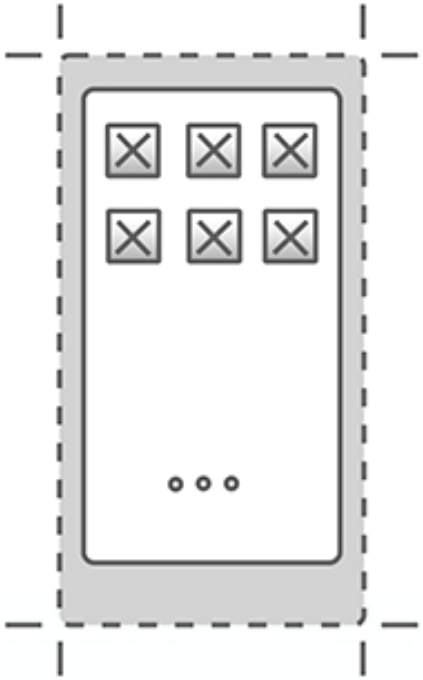
Prototyping stages



Fidelity (low vs. high) defines the prototype's **closeness** to the actual product.
We typically evaluate fidelity in terms of **content**, **interactivity**, and **design**.

Prototyping stages

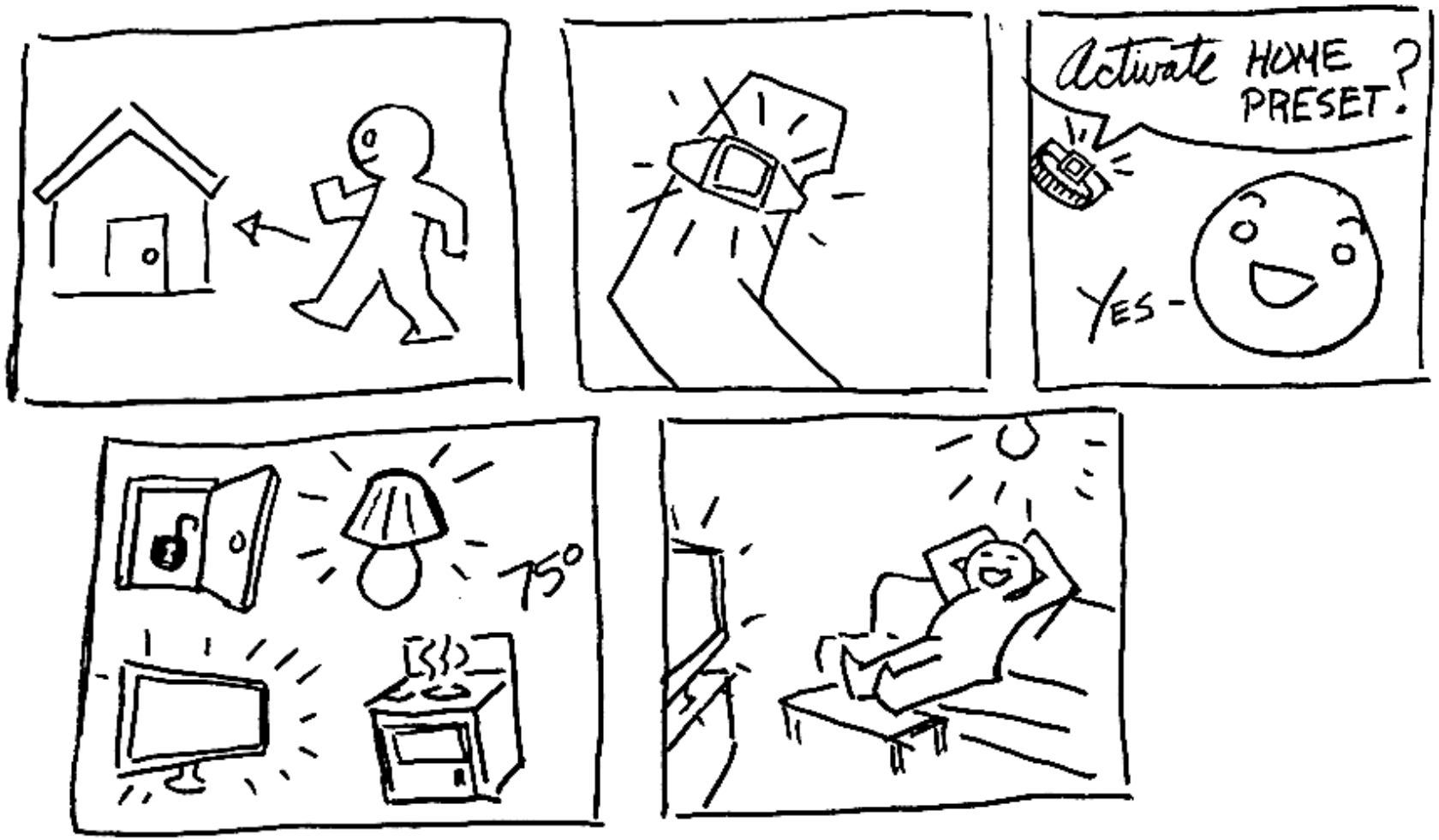




Sketch ▶ **Wireframe** ▶ **Mockup** ▶ **Prototype**

Sketch	Wireframe	Mockup	Prototype
Drawing a basic concept	Refining the concept further	Creating visual graphic and content	Making a detailed visual representation of a product

Sketches



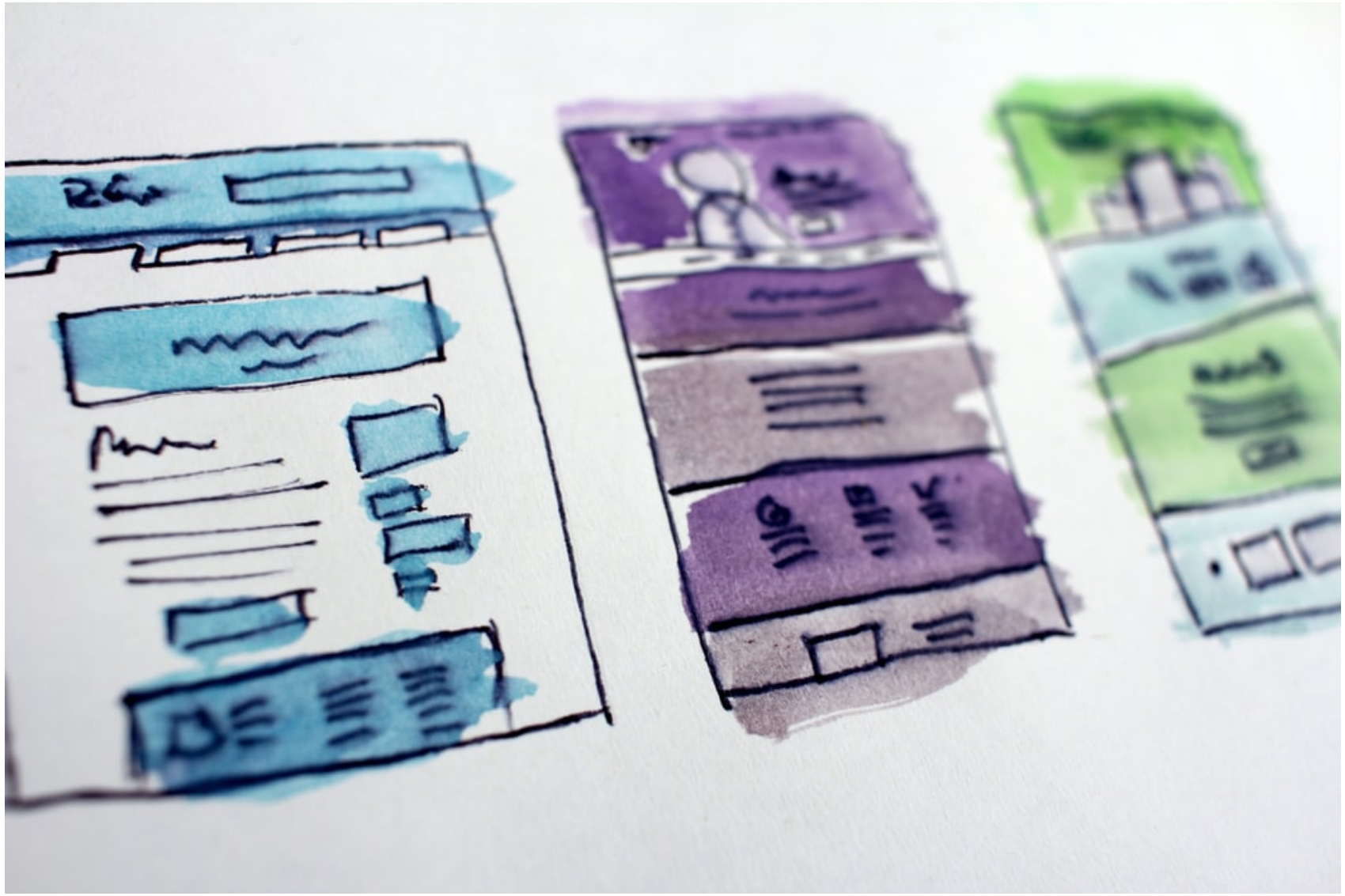
Hand drawn

Communicate a story

Highlight the context of use

Very **low fidelity** but very **fast results**

Sketches / Wireframes



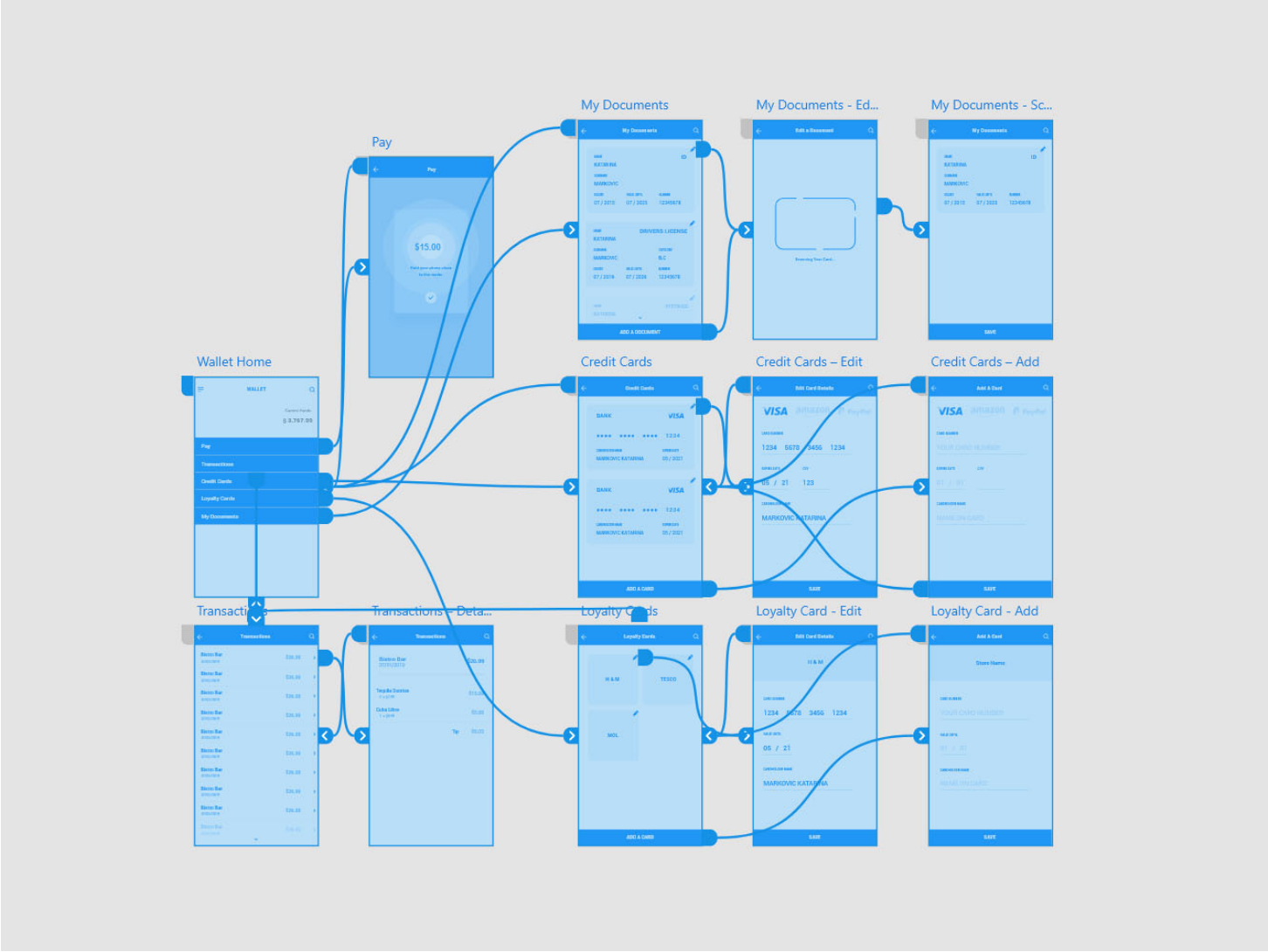
Hand drawn

Static representation of ideas

Rather for internal discussions

low fidelity but **fast results**

Wireframes



Tool-created

Representing the UI flow (navigation)

Contain placeholders for content

low fidelity but **fast results**

Mockups

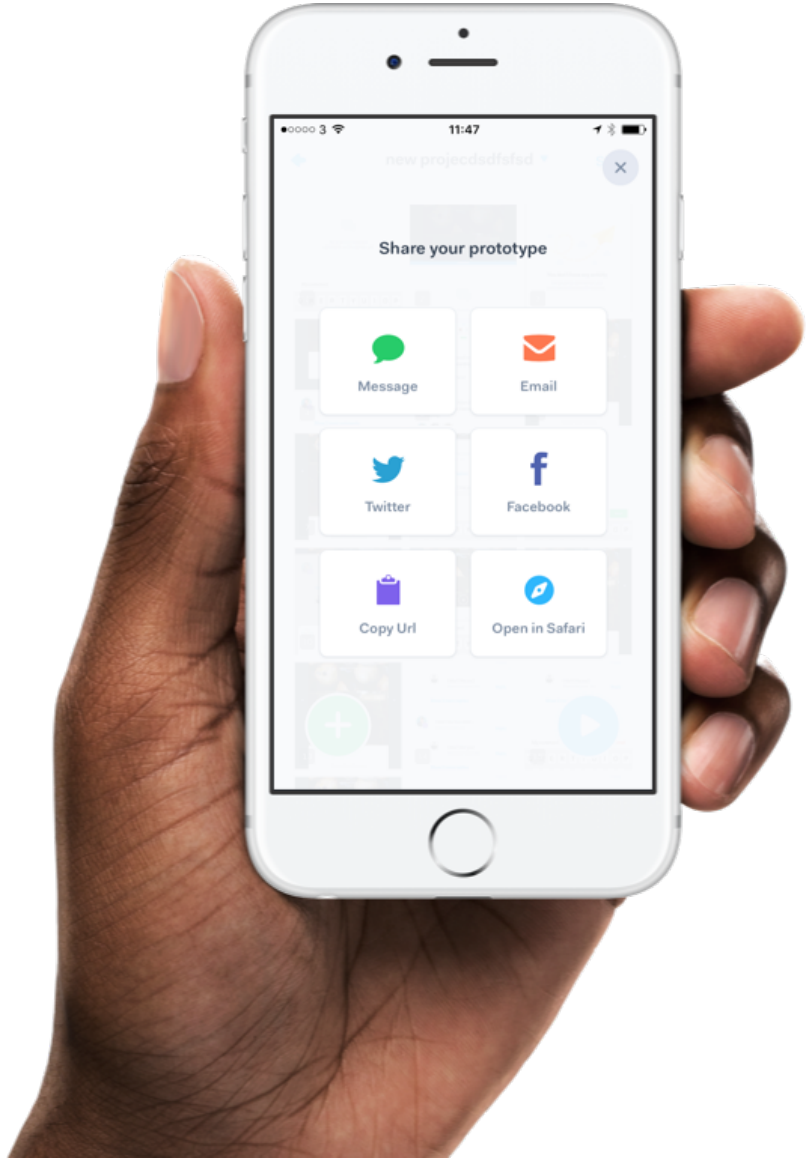
Tool-created

High quality
“pixel perfect” UI

Communication
with clients and
stakeholders

Mid to **high fidelity**
but **slower results**

Prototype



Tool-created

Fully interactive

Looks and acts like the final product

high fidelity but **slow results**

Why Prototyping?

Deal with complexity of software

Close gaps in understanding of the requirements

Make the requirements more concrete

Take a tentative step into the solution space (solution domain)

Reduce risk of customer misunderstanding and dissatisfaction

Bring use cases “to life”

Help gather early feedback from stakeholders

Prototypes as a requirements (improvement) tool

- Clarify, complete, and validate requirements
- Point out errors and problems with requirements
- Uncover overlooked requirements
- Assess the accuracy and quality of the requirements

A design exploration tool

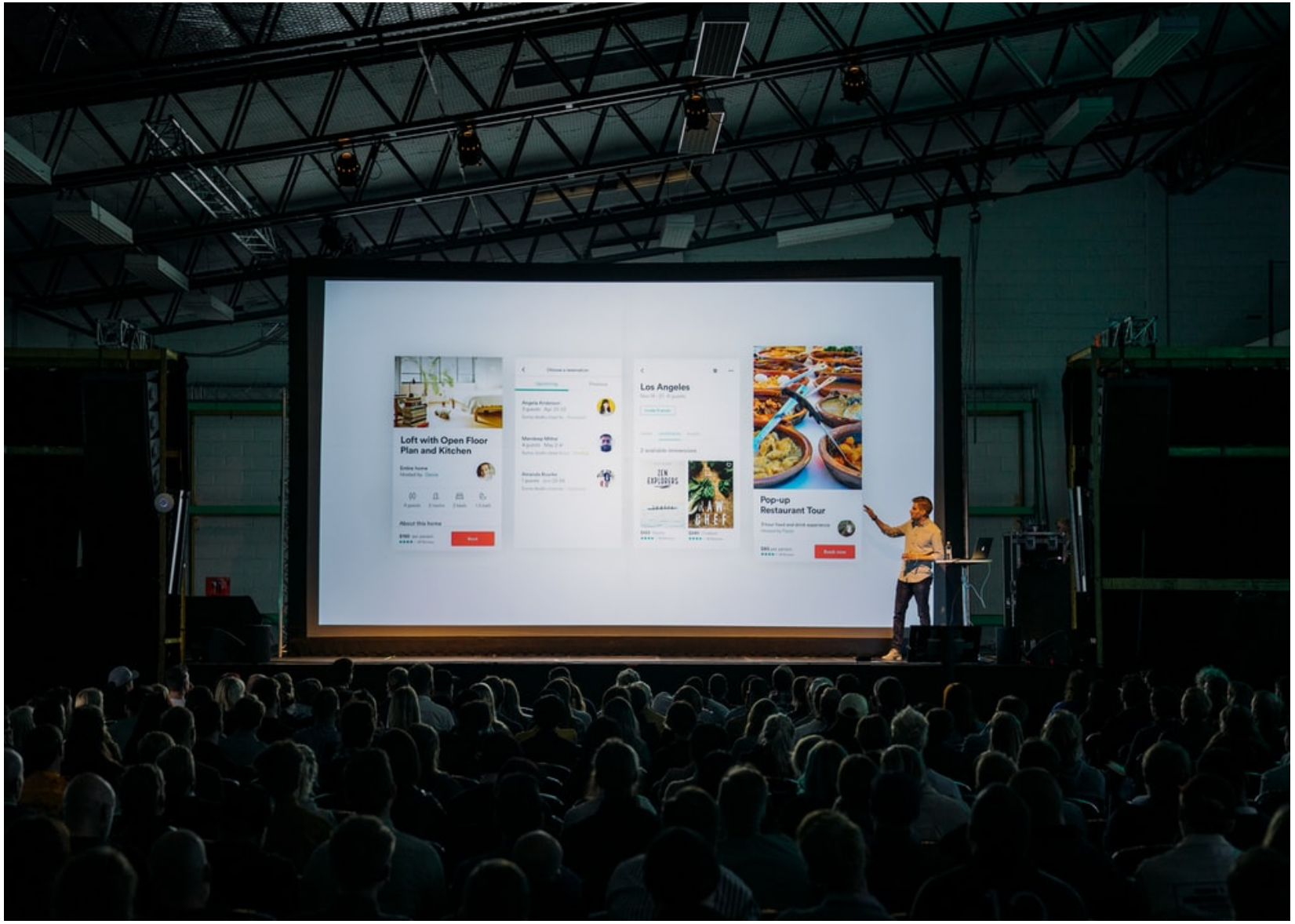
- Explore different user interaction techniques and design alternatives
- Envision the final product
- Optimize the usability of the system
- Demonstrate requirements feasibility through working design
- Confirm developers' understanding of the requirements

A planning and management tool

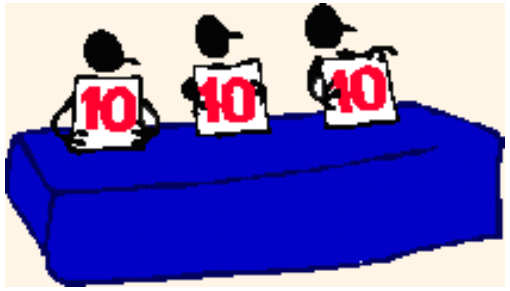
- Enable incremental development
- Decompose the system into manageable subsets that
  - Can be explored separately and
  - Can grow into the ultimate product
- Implementation of a subset of the product through a sequence of small-scale development cycles
- Reduce failure risk by starting early
- A good heuristic to define milestones

A communication tool

“A prototype is worth a thousand meetings”

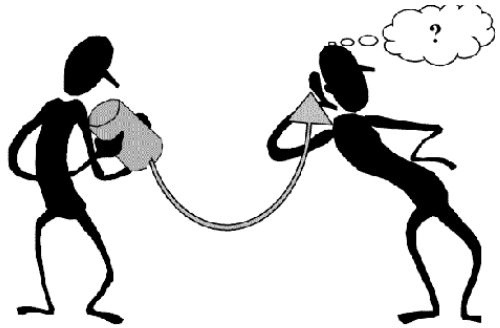


Criteria to choose the right model



Stakeholder

Speed



Medium

Fidelity

Stage of project

Style

Cost

Longevity

The Characteristics are interdependent!

Questions guiding the model selection process

Who is the main **Stakeholder?**

How fast can the prototype be delivered?

Which medium should be used?

Prototyping tools

- Help to create fast prototypes
- Create navigation links between custom mockups
- Create gestures and transitions
- Collaboration support

Overview

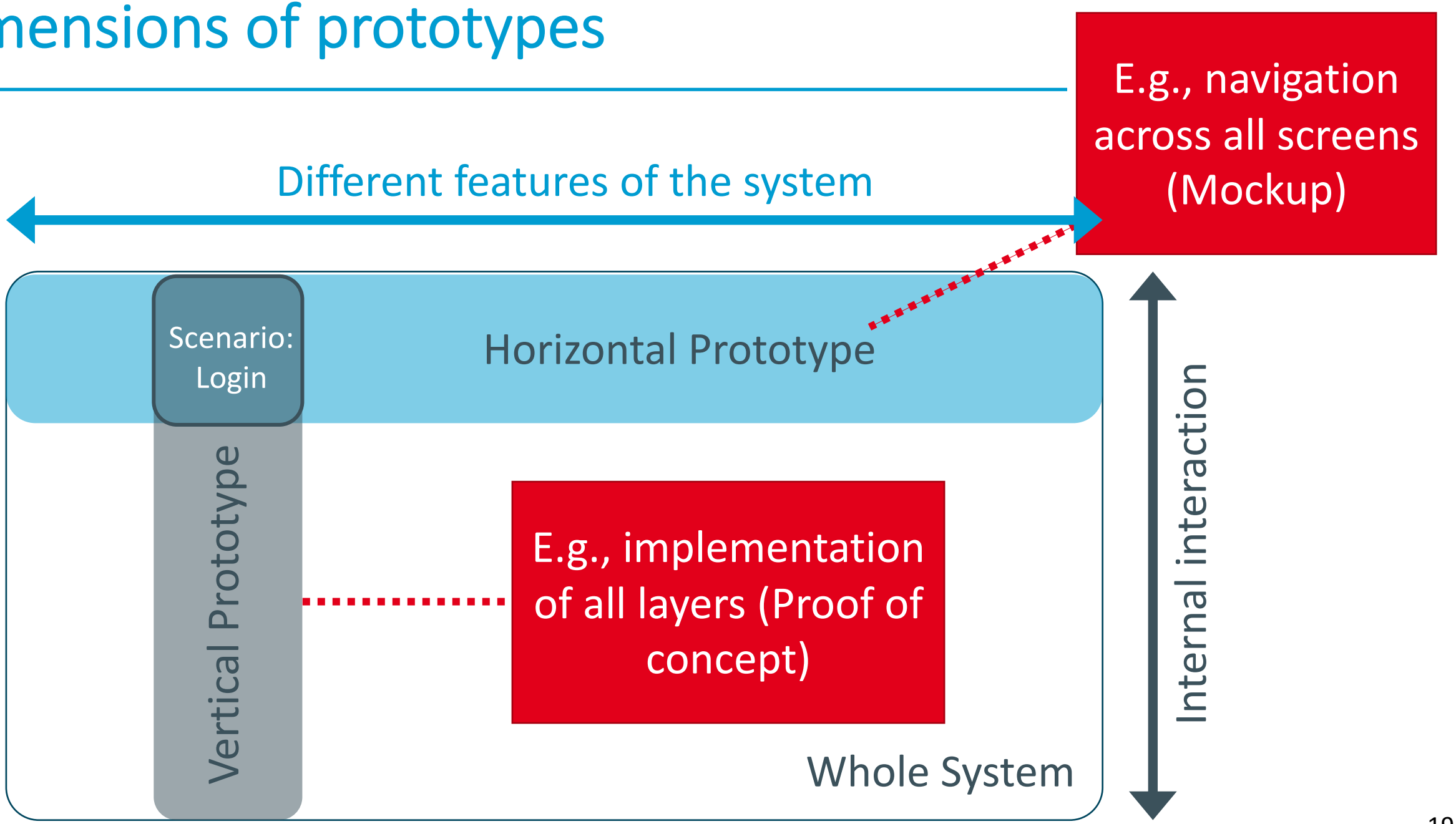
1 Motivation

2 **Dimensions of prototypes**

3 Types of prototypes

4 Prototype evaluation

Dimensions of prototypes



Horizontal prototypes

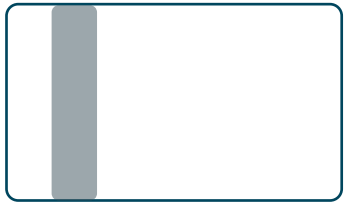
- Represent a **broad view** of the entire system
- Focus on the **user interface** or a portion of it
- Ignore architectural and implementation details
- Mockups are typical horizontal prototypes

When to use horizontal prototypes?

- Help stakeholders **externalize** and **explore** specific functionality and behavior of the intended system

- Help stakeholders **assess** whether a system based on the prototype will let them do their job in a reasonable way

Vertical prototyping



- Also known as a **proof of concept**
- Implement a slice of system **functionality through all layers**

  (from the user interface through all services)
- Work like the real system is supposed to work

When to use vertical prototypes?

- Clarify architectural **feasibility**
- **Optimize** algorithms
- **Evaluate** database schema
- Confirm the soundness of a cloud solution
- Test critical timing requirements
- Estimate the effort involved in implementing

Overview

1. Motivation
2. Dimensions of prototypes
3. **Types of prototypes**
4. Prototype evaluation

Throwaway vs. evolutionary prototype

Throwaway

- Low fidelity
- Exploratory

Evolutionary

- High fidelity
- Prospective part of the system

- Vertical and horizontal prototypes can either be a throwaway or evolutionary
- Decide **before** constructing!

Why should we build a throwaway?

…to answer questions

…to improve requirements quality

…to resolve uncertainties

[Davis, 1993]

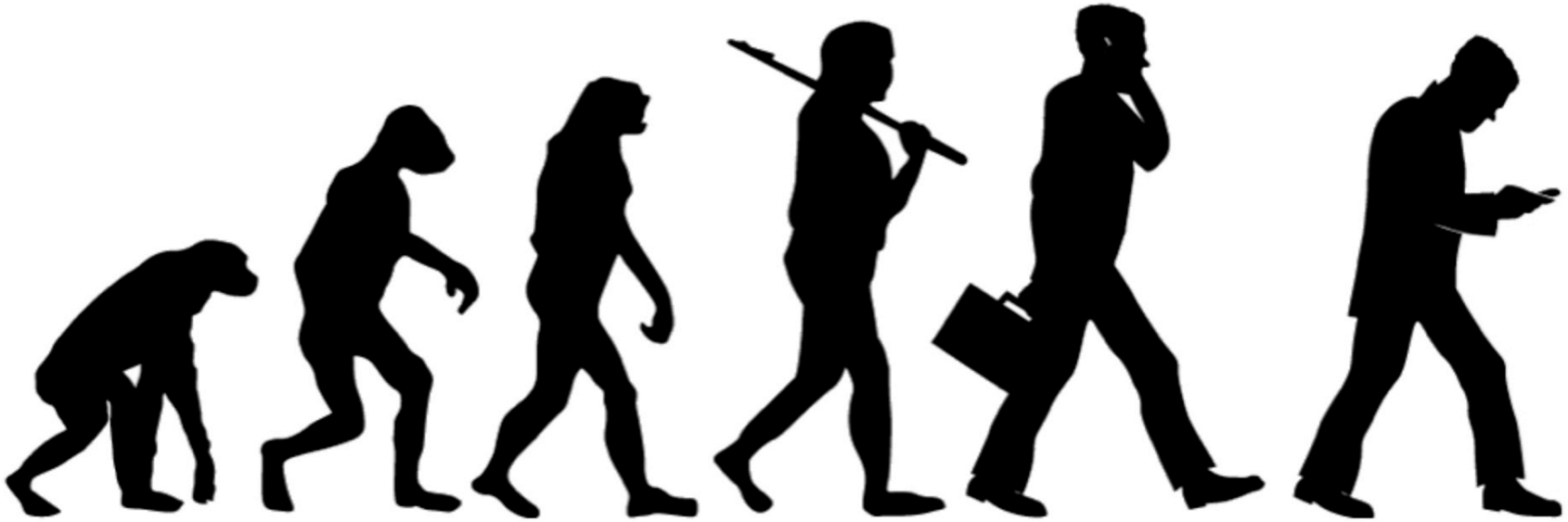
How to build a throwaway?

- As quickly and cheaply as you can
- Ignore solid software construction techniques
- Emphasizes **quick implementation** and **modification** over robustness, reliability, performance, and long-term maintainability

Remember!

- **Do not allow** low-quality code from a throwaway prototype to migrate into a production system
- Otherwise, users and maintainers will suffer from the consequences for the product life

Evolutionary prototypes



- Provides a **solid architectural foundation**
- Build the product **incrementally** as the requirements become clear over time
- Used e.g., in agile development

[McConnell, 1996]

How to build evolutionary prototypes

- Build with **robust, production-quality code**
- They **need time** and take longer to create than throwaway prototypes
- Design for **easy growth**, **frequent changes**, and **enhancement**

Summary

	Throwaway	Evolutionary
Horizontal/ Mockup	• Clarify and refine user and functional requirements • Identify missing functionality • Explore user interface approaches	• Implement core requirements • Implement additional user requirements based on priority • Often used for apps and websites • Adapt system to rapidly changing business needs
Vertical/ Proof of concept	• Demonstrate technical feasibility • Evaluate performance • Acquire knowledge to improve estimations	• Implement core functionality and communication layers • Implement and optimize core algorithms • Test and tune performance

Pressure to release prototype

Wow, looks like you are almost done!

The **biggest risk** is that a stakeholder will see a running throwaway prototype and conclude that the product is nearly completed.

Overview

1. Motivation
2. Dimensions of prototypes
3. Types of prototypes
4. **Prototype evaluation**

How to evaluate prototypes?

Possible evaluation questions

Is any functionality unnecessary?

Is the functionality implemented as expected?

Where you ever unsure what to do next?

Can we simplify any task?

What functionality is missing?

Any possible error conditions not covered by the prototype?

Is the navigation logical and complete?

Evaluation guidelines

- Prototype evaluation is related to usability testing
- Watch users work with the prototype – e.g., observation
- Ask users to tell you what they think of it – e.g., interview/survey
- Include various users like experienced and inexperienced – sampling
- Stress that it addresses only a portion of the functionality – expectation
- Evaluate with stakeholders! – e.g., (controlled) experiments

Example of a controlled experiment

Hypothesis

Caffeine intake increases the number of pages an author can write in an hour

Controlled variables:

- 2pm (in time zone)
- Same pen
- Same paper

Independent variable:

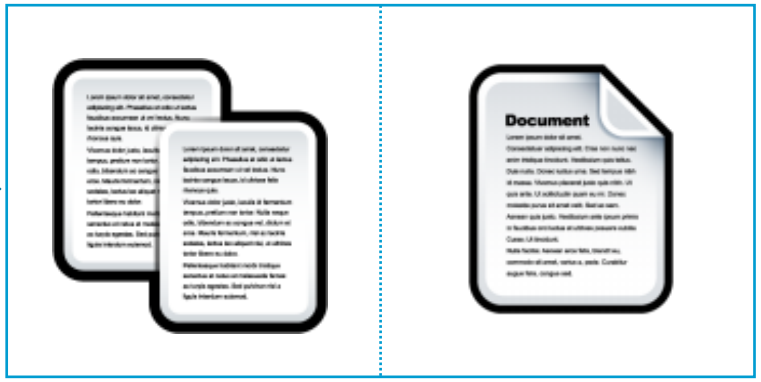
Caffeine intake

Experiment Group

Control Group

Dependent variable:

Number of written pages



Experiment Group

Control Group

Controlled experiments

Advantages

- Quantitative analysis of the benefits of a tool/technique
- Establish cause-effect relationship in a controlled setting

Limitations

- Hard to apply if you cannot simulate the right conditions in the lab
- Limited confidence that the lab setup reflects the real situation
- Ignores contextual factors
- Time-consuming!

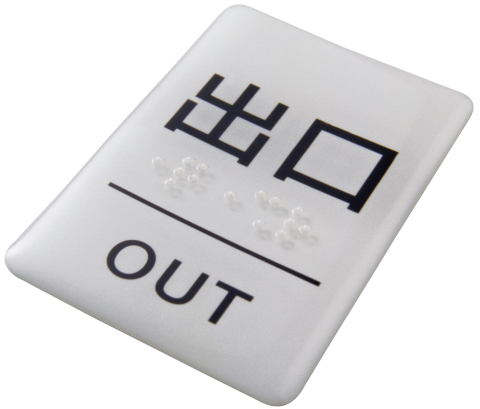
Experiment design terminology (1/2)

Independent variables

- Variables (factors) that are manipulated to measure their effect (“input variables”)
- E.g., a placement/name of a navigation element
- Typically specific levels of each variable to test

Dependent variables

- Tested to see how the independent variables affect them
- “Output variables”



Experiment design terminology (2/2)

Treatments

- Each combination of values of the independent variables is a treatment
- Simplest design:
  1 independent variable x 2 levels = 2 treatments, e.g., tool A vs. tool B

Subjects

- Human participants who perform some task to which the treatments are applied
- Note: subjects must be assigned to treatments **randomly**

Hypothesis testing in experiments

A clear hypothesis guides all steps of the experiment design

- Which variables to study?
- Which variables to ignore?
- How to measure the variables?
- Who are the subjects?
- What are the tasks?

Set up the experiment to refute the theory

H0

"the theory does not apply"

- Independent variables will not cause a difference between the treatments
- We assume H0 to be true unless the data say otherwise

H1

"the theory applies"

- Rejecting H0 gives evidence that the alternative is correct

Assigning treatments to subjects

Within-subjects Design

- Each subject tries all treatments
- Reduces chance that inter-subject differences impact the results

Challenge

Increases risk of learning effects (subjects get better from one treatment to the next)

Mitigated by

Balancing: vary order of the treatments

Note: if learning effects are symmetric

Between-subjects Design

- Different subjects get different treatments
- Reduces load on each subject

Challenge

Increases risk that confounding factors affect results (differences due to varying skill levels, experiences…)

Mitigated by

Blocking: group subjects equally

Note: if confounding factors can be measured

Experiments are positivist

- Assume we can reduce complex phenomena to a few relevant variables
- If critical variables are ignored, results may not generalize
- Other variables may dominate the cause-effect relationship shown in the experiment

Interaction effects

- Variables might together have an effect that none has on its own

- Reductionist experiments may miss this

  (e.g., a series of experiments, each testing one independent variable at a time)

- Using more than one independent variable is hard:
  - Larger number of treatments – bigger sample size!
  - More complex statistical tests

Examples

Joe has a brand-new idea!

Context

Joe is working on a startup with his team that is brainstorming about their product. He has a very cool idea and concept in mind that he wants to explain the others.

Prototyping technique

Sketches

or

Wireframes

or

Mockups

or

Prototype

Prototyping dimension

Horizontal

or

Vertical

Prototyping type

Throwaway

or

Evolutionary

Joe has a brand-new idea!

Context

Joe is working on a startup with his team that is brainstorming about their product. He has a very cool idea and concept in mind that he wants to explain the others.

Prototyping technique	Prototyping dimension	Prototyping type
Sketches	Horizontal	Throwaway
or	or	or
Wireframes	Vertical	Evolutionary
or		
Mockups		
or		
Prototype		

Lea needs feedback from customers!

Context

Lea is working in a commission-based company that has an advanced evolutionary prototype. She and the customer discussed a new feature idea, which feasibility should be tested.

Prototyping technique

Sketches

or

Wireframes

or

Mockups

or

Prototype

Prototyping dimension

Horizontal

or

Vertical

Prototyping type

Throwaway

or

Evolutionary

Lea needs feedback from customers!

Context

Lea is working in a commission-based company that has an advanced evolutionary prototype. She and the customer discussed a new feature idea, which feasibility should be tested.

Prototyping technique	Prototyping dimension	Prototyping type
Sketches	Horizontal	Throwaway
or	or	or
Wireframes	Vertical	Evolutionary
or		
Mockups		
or		
Prototype		

Summary

01 Prototypes are requirements, design, and planning tool. A prototype is worth a thousand meetings!

02 Horizontal prototypes explore functionality while vertical prototypes explore feasibility

03 A throwaway (e.g. mockup) is quick and cheap prototype. An evolutionary prototype should be a product increment

04 We can evaluate prototypes with interviews, survey, or controlled experiment if quantification is desired