

Prototyping Aspects Model [PROTO1]

PURPOSE of Prototyping

Exploration & learning

Communication: sales, alignment

Incremental development

Quality improvement

Validation & Testing

problem-solution / product-market fit, technical

feasibility, usability testing

SCOPE of Prototype

Breadth of functionality

Functional refinement

Visual appearance

Interactive & haptic behaviour

Data realism

Prototype MEDIA

Sketch: paper or computer-based

Wireframe: paper or computer-based

Mock-up: paper of computer-based

Source-code software

Other: video, interview

USE of prototype

Reviewers: internal, FFF, external

Prototype interaction: yes, no (demo)

Review approach: scenario-based free

Usage environment

Exploration STRATEGY

Single vs parallel exploration

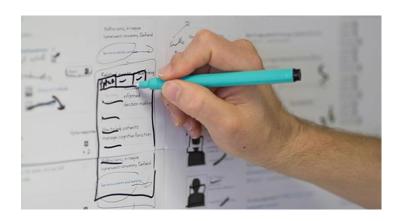
Iteration focus: Business, product, feature, optimisation

Iteration size



This Photo by Unknown Author is licensed under CC BY-ND

Wireframe

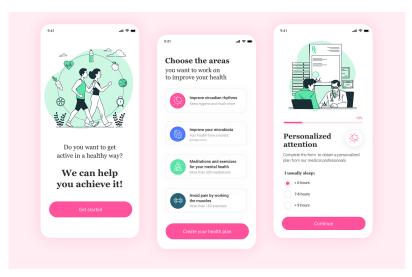


A way to design a website service at the structural level.

A wireframe is

- a lo-fidelity representation of product under development
- commonly used to layout content and functionality on a page
- used early in the development process to establish the basic structure of a page before visual design and content is added

Mockup



A representation of a product, showing users and stakeholders how it may look and be used.

Used to **present medium to high fidelity versions of a design**, e.g. to determine which aspects of a product that work well, and which do not.

Functional Requirements Styles

Context Diagram

- Diagram of product and its surrounding
- Defining product scope
- Very useful!

Event- and function lists

- Lists of events and functions
 - Domain or product level
- Good as checklists at verification
- Validation at product level?

Feature requirements

- Textual requirement: "the product shall ..."
- High expressive power
- Acceptable to most stakheolders
- Can lead to false sense of security
 - · How to ensure that goal-level covered?

Task descriptions

- Structured text describing user tasks
- Easy to understand and verify
- Good at domain level

(Vivid) Scenarios

- Rich descriptions of specific cases
- Improves developer intuition and imagination
- Products of elicitation but not "real" requirements

High-level tasks

- Client view of goal-related tasks
- Independent of existing domain-level tasks
- · Good for business process re-engineering

Use Cases

- Widely used in many styles and variants
- Some styles are good for design level (UI)
- Can be used at different levels
- Risk of pre-mature design

Standards as requirements

- Textual requirement: "the product shall follow standard xxx"
- Transfer the problem to the supplier
- Sometimes lead to false sense of security

Development process requirements

- A requirement to follow a certain procedure
 - Use prototypes
 - Use specific reviews at certain points
 - Test in a specific way
 - Max number of simultaneous change reports
 - ...etc
- Validation? Difficult to say how process quality relates to product quality

Task descriptions

Fig 3.6A Task descriptions

Work area: 1. Reception

Service guests - small and large issues. Normally standing. Frequent interrupts. Often alone, e.g. during night.

Users: Reception experience, IT novice.

R1: The product shall support tasks 1.1 to 1.5

Missing

sub-task?

Task: 1.1 Booking

Purpose: Reserve room for a guest.

Task: 1.2 Checkin

Purpose: Give guest a room. Mark it as occupied. Start account.

Trigger/

Precondition: A guest arrives

Frequency: Average 0.5 checkins/room/day

Critical: Group tour with 50 guests.

Sub-tasks:

- 1. Find room
- 2. Record guest as checked in
 - Deliver key

Variants:

- 1a. Guest has booked in advance
- 1b. No suitable room
- 2a. Guest recorded at booking
- 2b. Regular customer

Task: 1.3 Checkout

Purpose: Release room, invoice guest.

. . .

+ Tasks & support – w example solutions

+ High-level tasks – business perspective

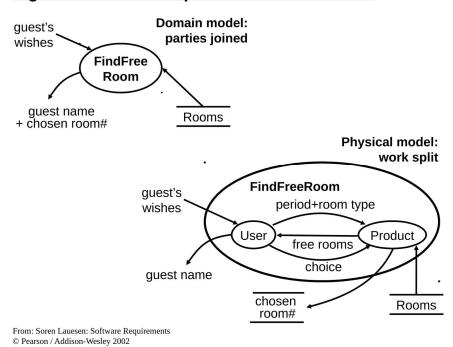
+ Tasks with data

See Lauesen 3.10 on good vs bad tasks

From: Soren Lauesen: Software Requirements © Pearson / Addison-Wesley 2002

Dataflow diagram

Fig 3.1 Human-computer - who does what?



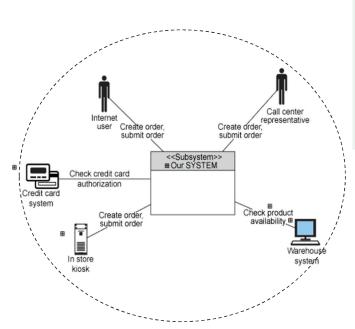
In store

Context diagrams Provides overview of **System** (black box == no system design!) Users and interacting systems System Context Booking System Functionality (or data) at domain level + business domain actors (not shown here) User classes Customers (Role Names) Open Accounts Agents View Transactions Open Accounts **Execute Transactions** View Transactions Call center representative Line indicates relationship between Create order submit order List of capabilities Actor and system submit order Booking Actor requires from system ("Use Case Naming Convention") System <<Subsystem>: ■ Our SYSTEM Check credit card authorization System "under discussion" in Create order, submit order center

Your requirements spec

Should cover all of the inner domain (domain requirements)

- Product (product & design reqts)
- Actors (domain reqts)
- Interfaces to other systems (domain reqts)
- + goal requirements
- + design-level requirements for a subset of functionality (first delivery of release plan)



Today's exercises

- 4) Revisit context diagram peer-to-peer presentation and discussion (20 min)
 - Actors and interacting systems
 - HL functionality
- 1) Plan prototyping in project groups (20 min)
- 2) Orientation of functional styles (10 min)
- 3b) Task descriptions (15 min)
- 3e) Dataflow diagram (15 min)
- 3f) Plan FR styles for your project per requirements level (20 min)

