ETSN15: Exercise 3

Prototyping & Functional requirements

ELIZABETH BJARNASON

Prototyping Aspects Model [PROTO]

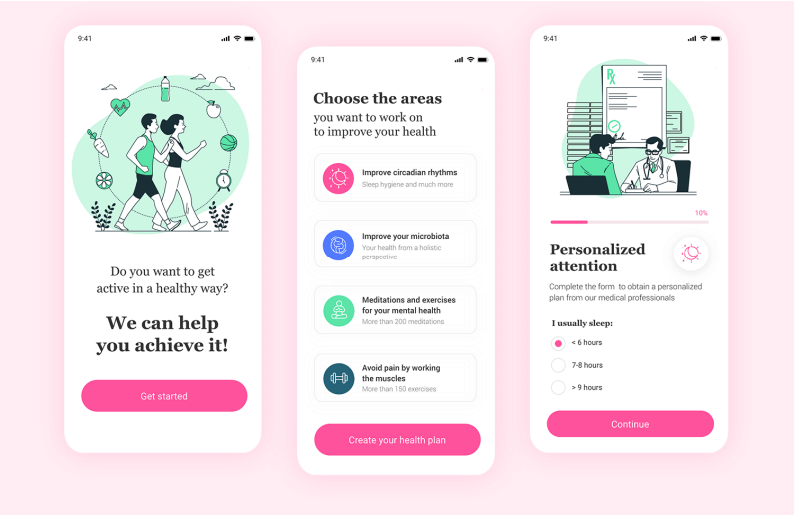
PURPOSE of Prototyping	USE of prototype
Exploration & learning	Reviewers: internal, FFF, external
Communication: sales, alignment	Prototype interaction: yes, no (demo)
Incremental development	Review approach: scenario-based free
Quality improvement	Usage environment
Validation & Testing	Exploration STRATEGY
problem-solution / product-market fit, technical feasibility, usability testing	Single vs parallel exploration
SCOPE of Prototype	Iteration focus: Business, product, feature, optimisation
Breadth of functionality	Iteration size
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	

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.

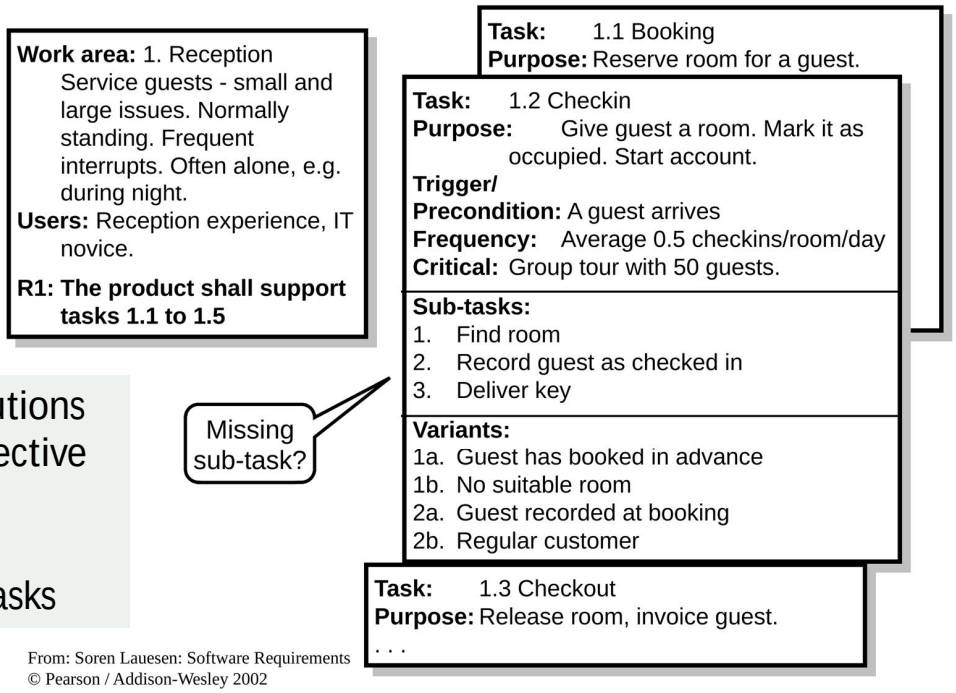
Used to present medium to high fidelity versions of a design, e.g. to determine which aspects of a product that work well, and which do not.

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



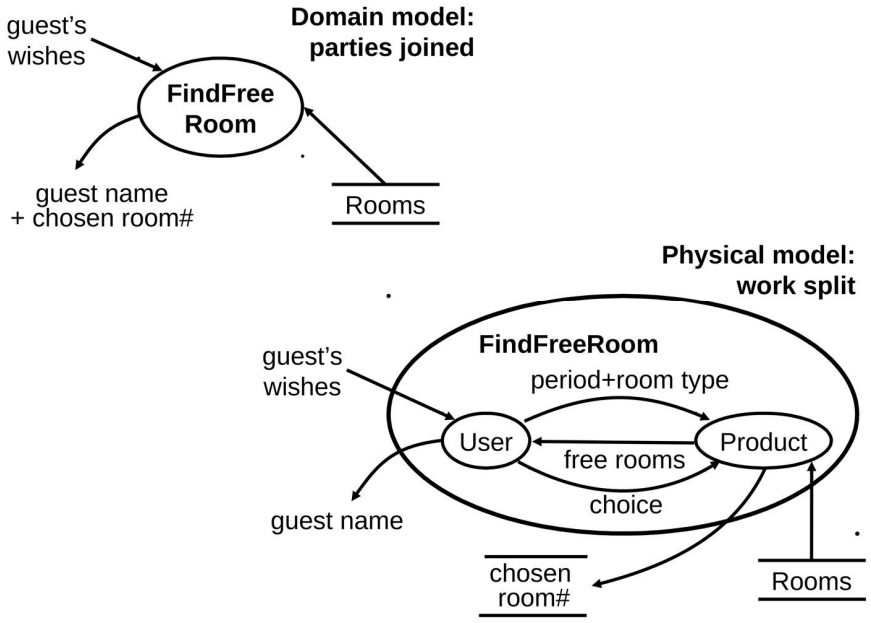
+ Tasks & support – w example solutions
+ High-level tasks – business perspective
+ Tasks with data

See Lauesen 3.10 on good vs bad tasks

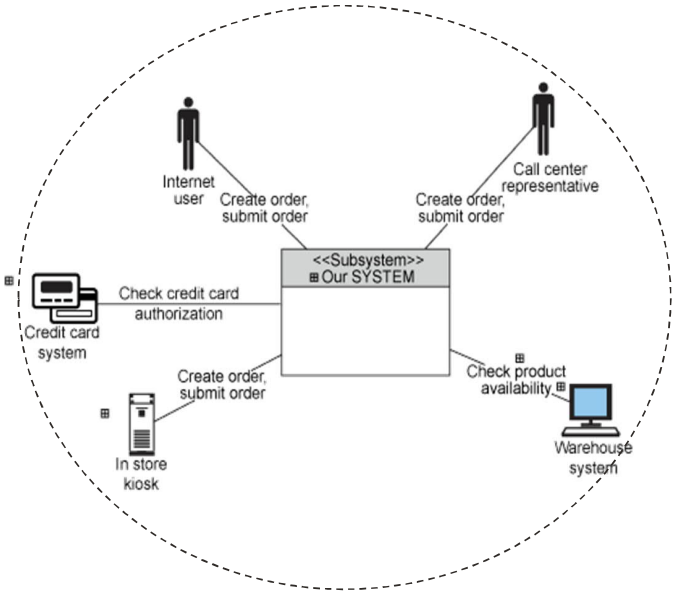
From: Soren Lauesen: Software Requirements

Dataflow diagram

Fig 3.1 Human-computer - who does what?



From: Soren Lauesen: Software Requirements



Styles in your SRS

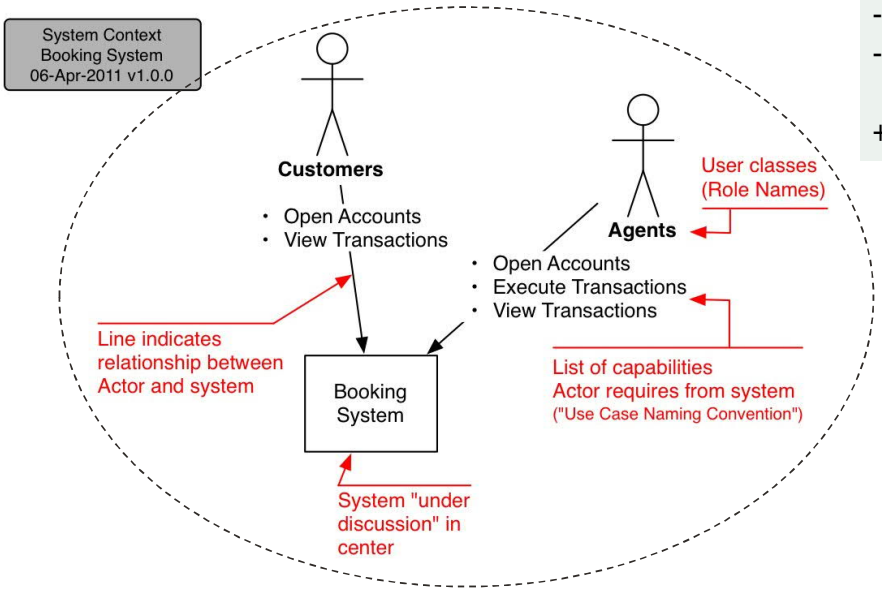
Should cover all of the inner domain (domain requirements)
- Product (product & design reqts)
- Actors (domain reqts)
- Interfaces to other systems (domain reqts)

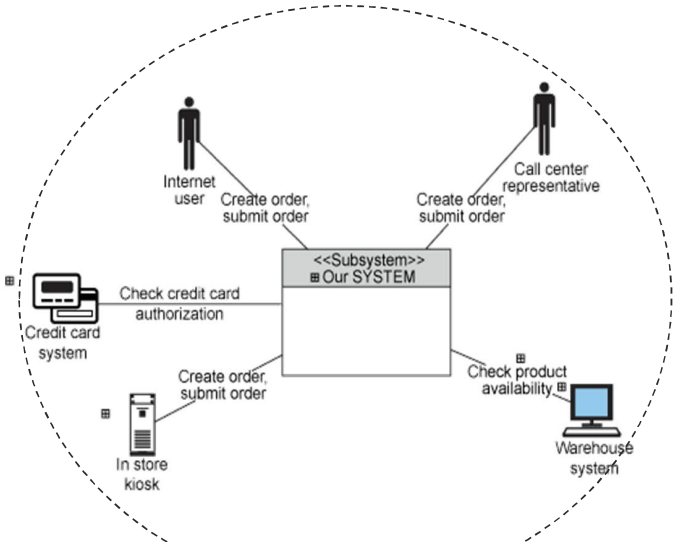
+ goal requirements
+ design-level requirements for a subset of functionality (first delivery of release plan)

Context diagrams

Provides overview of
- System (black box == no system design!)
- Users and interacting systems
- Functionality (or data) at domain level

+ business domain actors (not shown here)





Today's exercises

4) Revisit context diagram – peer-to-peer presentation and discussion (15 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)

LUND
UNIVERSITY