Brief recap (HCI perspective)

- Why so many software development projects fail, go wrong or often produce less usable results:
  - Construction relies on requirements
  - Requirements are difficult to formulate
    - The complexity of natural systems (such as businesses)
      - » Current modeling tools tends to exaggerate regularity
    - Human cognitive and social functioning

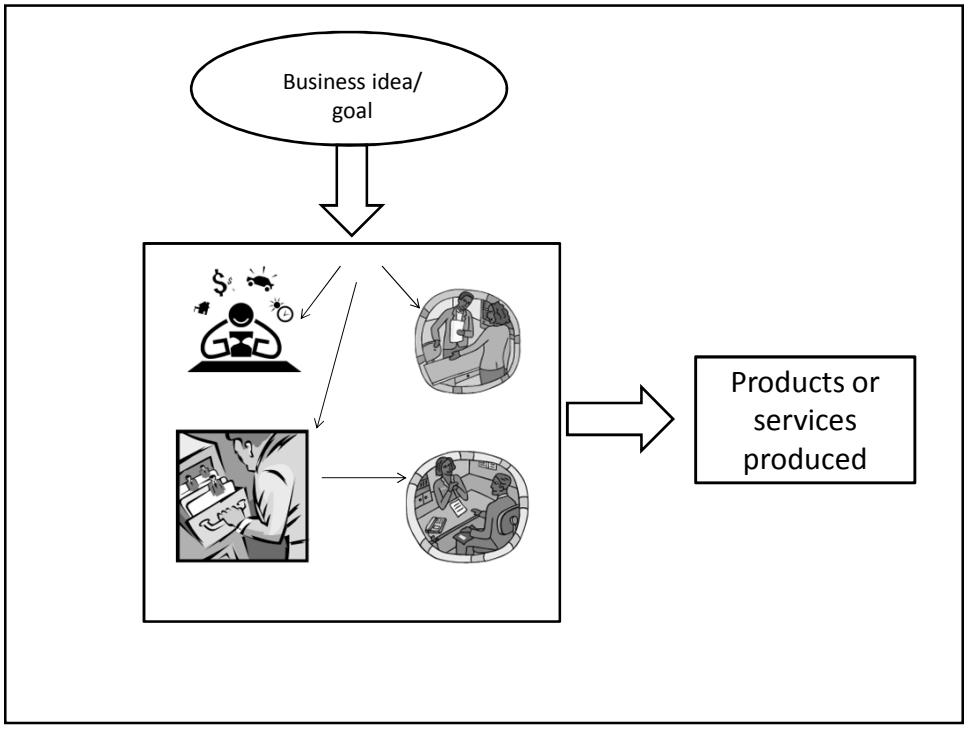
These difficulties are not generally recognized as such outside of HCI

Brief recap (HCI perspective)

- Why so many software development projects fail, go wrong and quite often produce less usable results:
  - Construction relies on requirements
  - Requirements are difficult to formulate
    - The complexity of natural systems (such as businesses)
      - » Current modeling tools tends to exaggerate regularity
    - Human cognitive and social functioning

Furthermore:

Requirements often/always lack important information



Organization of work

1. Who does what?
   - Different user profiles may have different demands on an IT-system
     - Sales representatives usually do not need access to technical documentation. If engineers can double as sales representatives at times, they could be well served by such access.

Organization of work

2. Where is the work performed?
   - The surrounding environment and the hardware necessary in different environments heavily influence system design
     - Security issues when working in public areas
     - Simultaneous use of phones
     - Noise levels
     - Room for keyboard?
     - etc

Organization of work

3 Flexibility of work organization?
   - Is there a need:
     - to help one another amongst users?
     - for spontaneous re-organizing to meet customer demands?
     - etc

Organization of work

4 Does a new IT-system enable other ways of organizing work?

Requirements often/always lack important information:

1, Organization of work

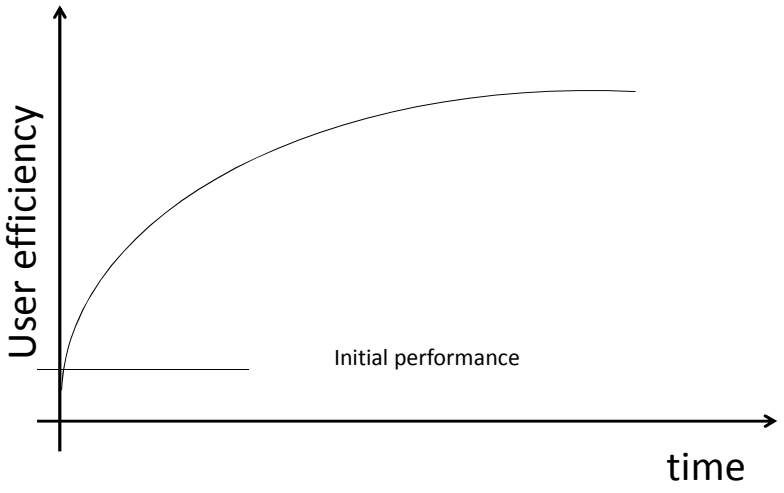
2, Details of the work situation from the workers' perspective

ISO 9241 part 11

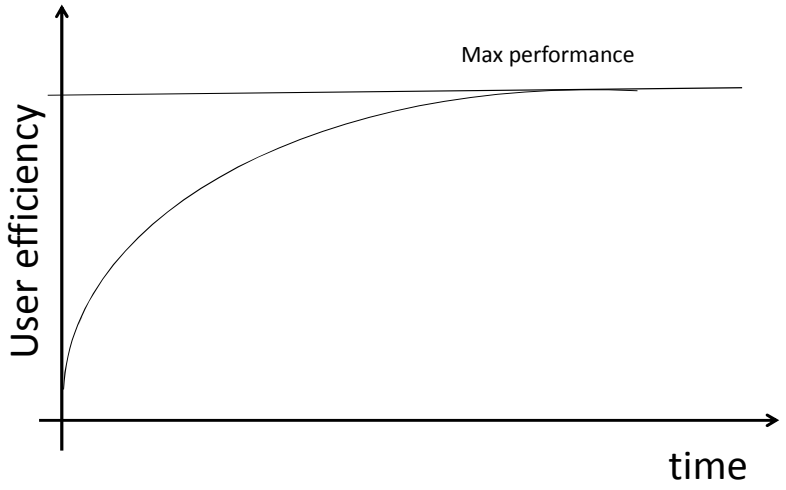
- (*Usability is*) the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

9

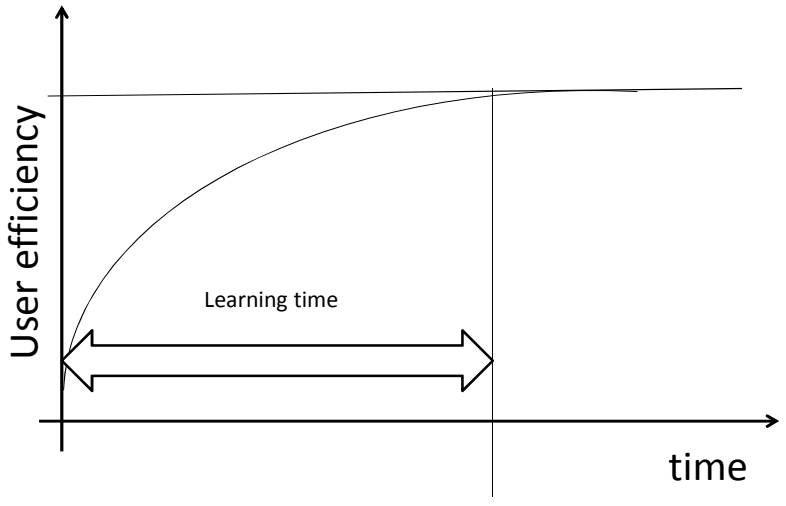
A simplified graph



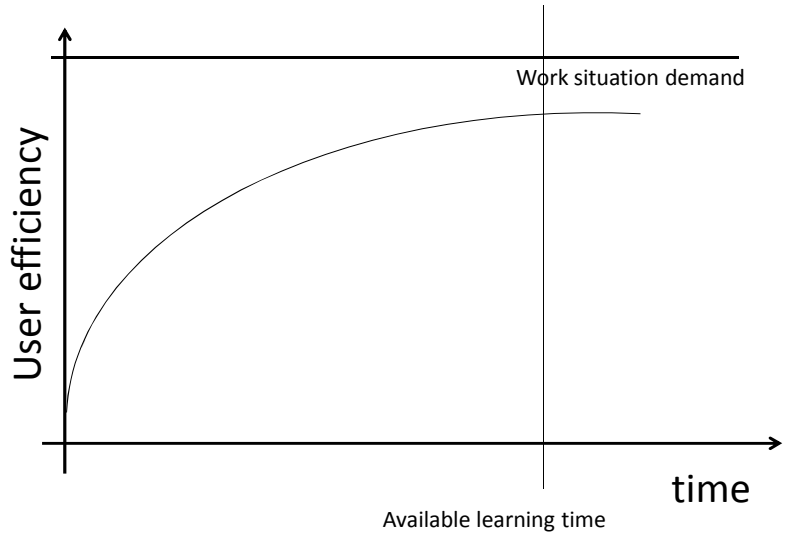
A simplified graph



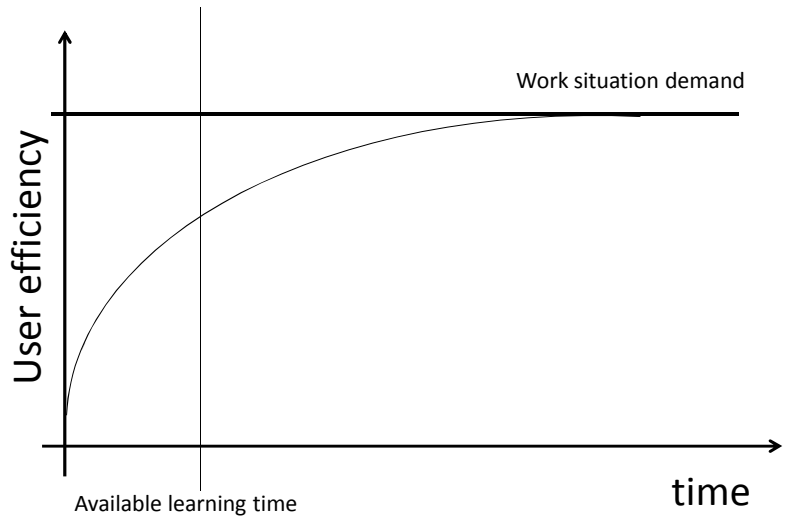
A simplified graph



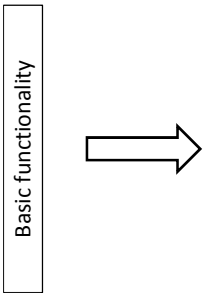
A simplified graph



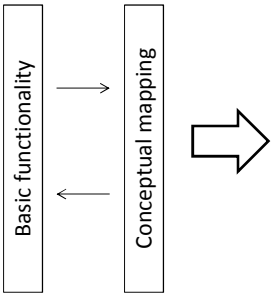
A simplified graph



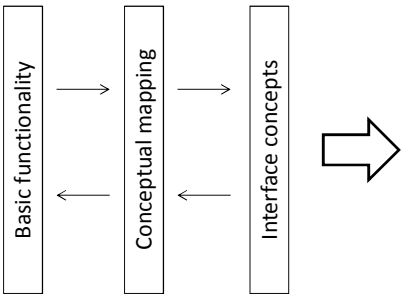
Anatomy of a software system



Anatomy of a software system



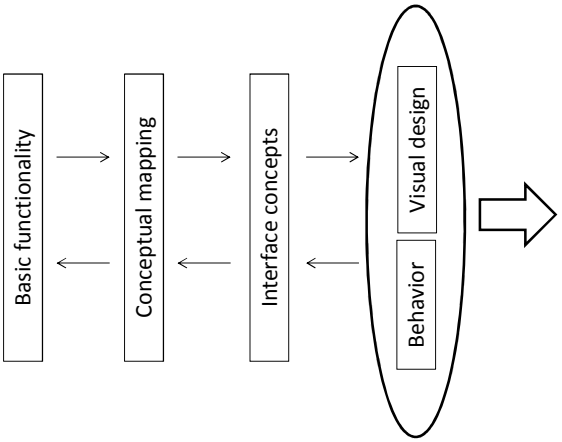
Anatomy of a software system



User interface terms, objects and functions:

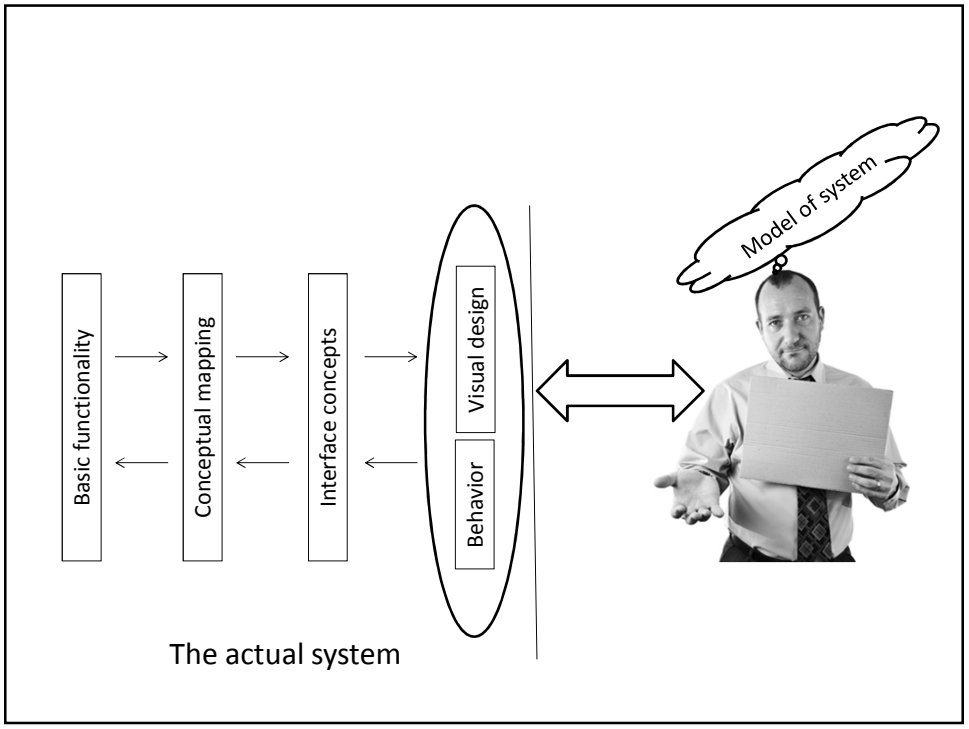
- Customer form
- Sales form
- Client form
- Cut
- Paste
- Save
- Copy
- etc

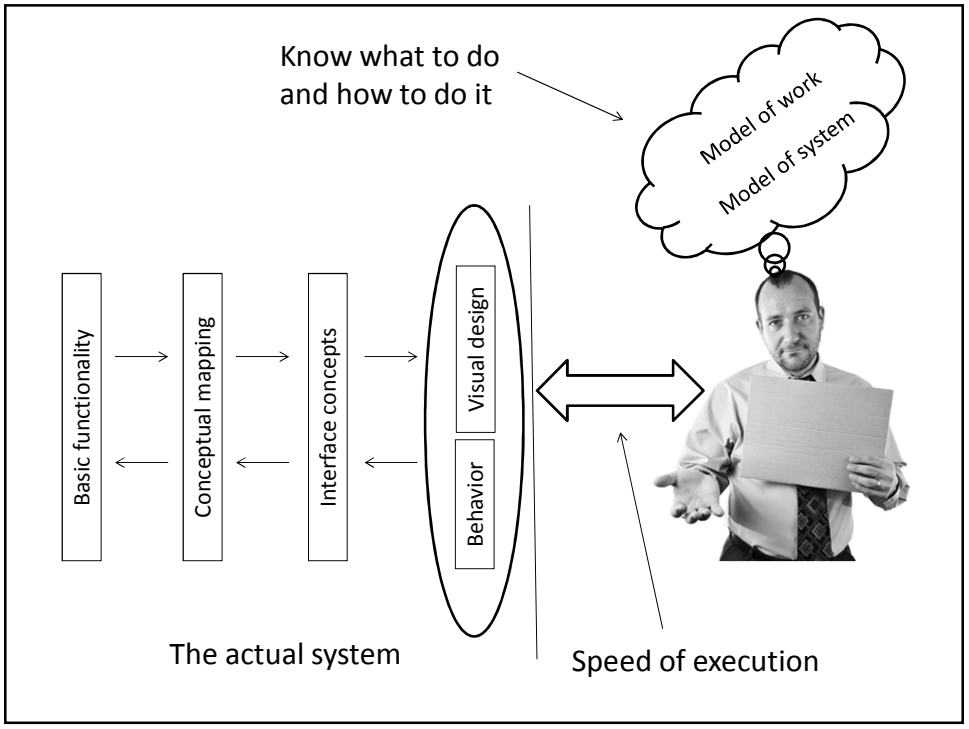
Anatomy of a software system



User interface:

- Customer tab
- Sales pane
- Cut icon
- Paste icon
- Visual cues for states
- Feedback rules
- Event handling
- etc

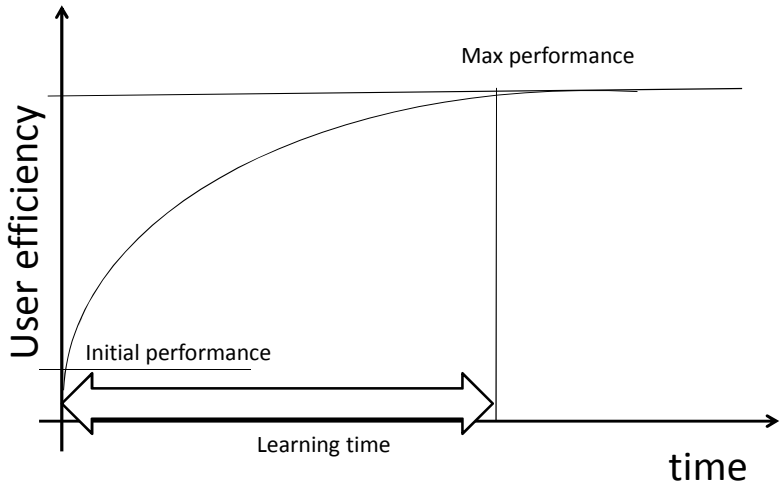

Basic functionality
Conceptual mapping
Interface concepts
Visual design
Behavior
Model of system
The actual system


Know what to do
and how to do it
Model of work
Model of system
Basic functionality
Conceptual mapping
Interface concepts
Visual design
Behavior
The actual system
Speed of execution

Difference between sporadic use and regular use

- Many IT-design educations ('interaction design') focus heavily on sporadic use and its problems.
  - The big commercial potential (for IT-professionals) is often today with apps and websites
- In working life there is usually a mix between sporadic use of some parts of a systems and regular use of other parts

A simplified graph

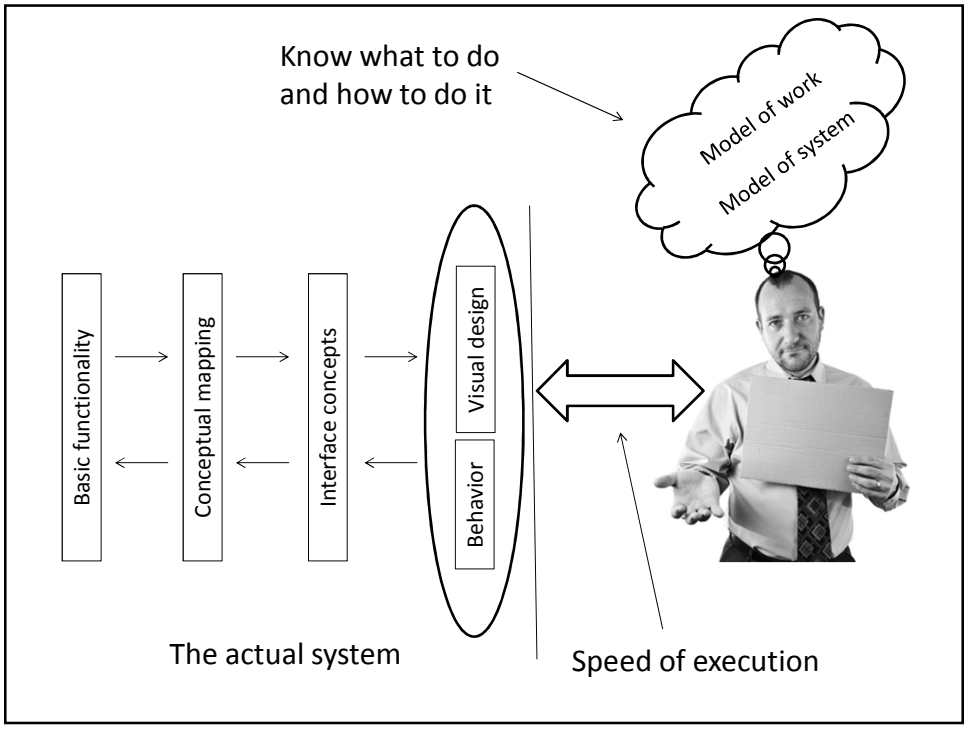


For regular use

- Initial performance and speed of learning are important if substitute workers are common and need to be up to speed asap
- Max performance levels are the most important factor
- Caveat: Systems needs to be sold/accepted
  - Who makes that decision?

Speed of use (efficiency)

- Initial performance is mostly determined by the similarity between the model needed to run the system and pre-existing mental models of the users.
- Speed of learning is mostly determined by the structure of the model needed to run the system and its implementation in terms of visual design and behavior
- Max performance is determined by:
  - the motor actions needed to transmit the users' intentions to the system.
  - the readability and visual structure of the information presented to the users



To summarize

- Main problem areas:
  - Design not adapted to the organizational structure and its degree of fluidity
  - Design too rigid for users to cope with the complexity of the natural system they work in
    - System to 'Taylorized'
  - Design does not support user efficiency in relevant contexts of use
    - Initial performance
    - Sped of learning
    - Max performance

To summarize

- Main problem areas:
  - Design not adapted to the organizational structure and its degree of fluidity
  - Design too rigid for users to cope with the complexity of the natural system they work in
    - System to 'Taylorized'
  - Design does not support user efficiency in relevant contexts of use
    - Initial performance
    - Sped of learning
    - Max performance

All of these factors are not generally recognized as such outside of HCI

SCRUM (1)

- Basically a software development process
  - Focused on the internal workings of the team
- Tries to solve the problem with requirements by having a 'product owner'
- The product owner
  - Has the responsibility of understanding end-user work and representing the end-user
  - One person and s/he has to be satisfied with the implementations!
  - Work to be done in the form of user stories

SCRUM (2)

- Scrum itself doesn't define how the product owner knows what user stories to select
- A user story is not a complete description of work
  - Only a memory aid, usually in the form
    - AS a [user role], I want [a feature] so that I can [achive a goal].
    - E.g. 'As a salesperson, I want to capture customer information quickly so I can keep a history of my interactions with this customer'
- The collection of user stories define the system to be built
  - Often found by more traditional requirements specification methods

SCRUM (3)

- A sprint:
  - is during which an implementation is created that corresponds to one or several items from the product backlog, usually user stories
  - Starts by selecting a items from the product backlog.
  - Ends with a review. Usually a short walkthrough with the stakeholders and the product owner
  - At the end, the team also evaluates its work and processes and discuss improvements to these processes

SCRUM (4)

- The backlog
  - Is the collective name for all pieces of work to be completed within a sprint
  - Mainly user stories
- The scrum master
  - Guides the team
  - Runs:
    - the daily stand-up meetings
    - the planning meetings
  - May not be played by the same person who plays the product owner

From the perspective of HCI: A user centered design process [ISO]

ISO 9241 – 210 (13407)

9241-210 key elements

- The design is based upon an explicit understanding of users, tasks and environments.
- Users are involved throughout design and development.
- The design is driven and refined by user-centered evaluation.
- The process is iterative.
- The design addresses the whole user experience.
- The design team includes multidisciplinary skills and perspectives.

Shortcomings of agile methods from a user centered point of view (1)

- The team is one
  - All members should be able to do all tasks
  - Having special skills in understanding work, talking to (real) users, designing a good user interface etc is thus problematic
- Doesn't get valid user input
  - Understanding end-users and their work
  - Evaluating implementations

Shortcomings of agile methods from a user centered point of view (2)

- No distinction between different levels of stability
  - Solutions and techniques can change fast
  - HOWEVER: The essence of the work supported changes slowly
- An integral user experience (over sprints) is difficult to accomplish
- Tends to underestimate the importance of ideation
  - Especially in terms of UX

Beyer's suggestions (1)

- 1: Redefine users
  - There's a difference between end-users, product owners and stakeholders
- 2: Embrace the 'no big design up front' idea BUT add a 'big picture up front' idea
  - Introduce a 'phase 0'
    - Already discussed in the agile community
  - Add user involvement and work-analyses to this phase
  - Add overall UX design to this phase

Beyer's suggestions (2)

- 3: Define a separate UX design function/team that works one sprint ahead and in parallel
- 4:Let a UX-team do validation (proper user testing) one sprint behind
  - Add the found problems (if any) in the form of design change suggestions to the backlog