UE&UX Lecture 6: Handbook Chapters 10, 11, and 12

Egon L. van den Broek
Chapter 10

ESTABLISHING REQUIREMENTS
How to read the book?

Important:
- main text
- objectives and summaries
- figures and tables are important
- boxes

Interesting and illustrative
- dilemma’s
- activities (inspiration for exam)
- Assignments (inspiration for exam)
• Requirements have many shapes
• And are treated in several courses

• This course has its own perspective on requirements: UX requirements.
• This is different from software requirements and other requirements.

• So, requirements as considered in other courses are often *not* valid for UE&UX!!!
Overview

• The importance of requirements
• Different types of requirements
• Data gathering for requirements
• Data analysis and presentation

• Task description: Scenarios
  Use Cases
  Essential use cases

• Task analysis: HTA
What, how and why?

What needs to be achieved?

1. Understand as much as possible about users, task, context
2. Produce a stable set of requirements

How can this be done? (all of this is iterative)

• Data gathering activities
• Data analysis activities
• Presentation
• Express findings as requirements
What, how and why?

- Why bother?
  Requirements definition is the stage where failure occurs most commonly.

Getting requirements right is crucial.
Establishing requirements

• What do users want? What do users ‘need’?

Requirements need clarification, refinement, completion, re-scoping

Input: Requirements document (maybe)
Output: stable requirements

• Why ‘establish’?

Requirements arise from understanding users’ needs
Requirements can be justified & related to data
Volere shell

Requirement #: 75        Requirement Type: 9        Event/use case #: 6

Description: The product shall issue an alert if a weather station fails to transmit readings.

Rationale: Failure to transmit readings might indicate that the weather station is faulty and needs maintenance, and that the data used to predict freezing roads may be incomplete.

Source: Road Engineers
Fit Criterion: For each weather station the product shall communicate to the user when the recorded number of each type of reading per hour is not within the manufacturer’s specified range of the expected number of readings per hour.

Customer Satisfaction: 3        Customer Dissatisfaction: 5
Dependencies: None        Conflicts: None
Supporting Materials: Specification of Rosa Weather Station
History: Raised by GBS, 28 July 99

7 June 2017
Volere requirements template

<table>
<thead>
<tr>
<th>PROJECT DRIVERS</th>
<th>FUNCTIONAL REQUIREMENTS</th>
<th>NON-FUNCTIONAL REQUIREMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The Purpose of the Product</td>
<td>6. The Scope of the Work</td>
<td>10. Look and Feel Requirements</td>
</tr>
<tr>
<td></td>
<td>8. The Scope of the Product</td>
<td>12. Performance Requirements</td>
</tr>
<tr>
<td></td>
<td>9. Functional and Data Requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13. Operational and Environmental Requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14. Maintainability and Support Requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15. Security Requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>16. Cultural and Political Requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>17. Legal Requirements</td>
<td></td>
</tr>
<tr>
<td></td>
<td>18. Open Issues</td>
<td></td>
</tr>
<tr>
<td></td>
<td>19. Off-the-Shelf Solutions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20. New Problems</td>
<td></td>
</tr>
<tr>
<td></td>
<td>21. Tasks</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22. Migration to the New Product</td>
<td></td>
</tr>
<tr>
<td></td>
<td>23. Risks</td>
<td></td>
</tr>
<tr>
<td></td>
<td>24. Costs</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25. User Documentation and Training</td>
<td></td>
</tr>
<tr>
<td></td>
<td>26. Waiting Room</td>
<td></td>
</tr>
<tr>
<td></td>
<td>27. Ideas for Solutions</td>
<td></td>
</tr>
</tbody>
</table>
Different kinds of requirements

• Functional:
  — What the system should do

• (Non-functional: security, response time...)

• Data:
  — What kinds of data need to be stored?
  — How will they be stored (e.g. database)?
Different kinds of requirements

Environment or context of use:

• physical: dusty? noisy? vibration? light? heat? humidity? …. (e.g. ATM)

• social: sharing of files, of displays, in paper, across great distances, synchronous, privacy for clients

• organisational: hierarchy, IT department’s attitude and remit, user support, communications structure and infrastructure, availability of training
Underwater computing

Figure 10.2 (a) The components of WetPC’s underwater computer.

Underwater computing

Figure 10.3 (a) The KordGrip interface and (b) the KordGrip in use underwater

Source: (a) Reproduced by permission of WetPC Pty Ltd (b) Reproduced by permission of the Australian Institute of Marine Science.
Different kinds of requirements

Users: Who are they?

— Characteristics: nationality, educational background, attitude to computers

— System use: novice, expert, casual, frequent
  — Novice: prompted, constrained, clear
  — Expert: flexibility, access/power
  — Frequent: short cuts
  — Casual/infrequent: clear menu paths
What are the users’ capabilities?

Humans vary in many dimensions:

— size of hands may affect the size and positioning of input buttons

— motor abilities may affect the suitability of certain input and output devices

— height if designing a physical kiosk

— strength - a child’s toy requires little strength to operate, but greater strength to change batteries

— disabilities (e.g. sight, hearing, dexterity)
Personas

• Capture a set of user characteristics (user profile)

• Not real people, but synthesised from real users

• Should not be idealised

• Bring them to life with a name, characteristics, goals, personal background

• Develop a small set of personas with one primary
Example Persona

**BACKGROUND**
- 15, Female
- Ongoing Private Education
- Ambitious
- Comfortable using technology to communicate

**MOTIVATIONS**
- Keeping in touch with her network
- Fashion/street cred
- Keeping up with peers.

**FRUSTRATIONS**
- Sad people trying to be ‘friends’ on Facebook
- Having to be in bed @ 11pm
- Being swamped in friends updates
- Missing important status updates

**Ginnie**

Receives private tutoring in Maths and English as these are not her strong subjects. Enjoys playing for the school’s 2nd teams for netball and Lacrosse and is good at art.

She loves recording her favourite shows: ER and Sun Valley High on Sky+ and spends some of her time on her Laptop that Daddy bought her watching videos on YouTube, downloading music, keeping up to date with her friends on Facebook and chatting via MS IM to her cousin who is at University in Leeds.

She loves Ugg boots and Abercrombie & Fitch and uses the Internet to shop and find the cheapest prices.

“I want to easily hook up with my friends whilst watching TV”
Data gathering for requirements

• Interviews:
  — Props, e.g. sample scenarios of use, prototypes, can be used in interviews
  — Good for exploring issues
  — Development team members can connect with stakeholders

• Focus groups:
  — Group interviews
  — Good at gaining a consensus view and/or highlighting areas of conflict
  — But can be dominated by individuals
Data gathering for requirements

• **Questionnaires:**
  — Often used in conjunction with other techniques
  — Can give quantitative or qualitative data
  — Good for answering specific questions from a large, dispersed group of people

• **Researching similar products:**
  — Good for prompting requirements
Data gathering for requirements

• **Direct observation:**
  — Gain insights into stakeholders’ tasks
  — Good for understanding the nature and context of the tasks
  — But, it requires time and commitment from a member of the design team, and it can result in a huge amount of data

• **Indirect observation:**
  — Not often used in requirements activity
  — Good for logging current tasks
Data gathering for requirements

Studying documentation:

— Procedures and rules are often written down in manuals

— Good source of data about the steps involved in an activity, and any regulations governing a task

— Not to be used in isolation

— Good for understanding legislation, and getting background information

— No stakeholder time, which is a limiting factor on the other techniques
Some examples

Ethnographic study, interviews, usability tests, and user participation

Figure 10.6 (a) Exploring mouse gene expression using G-nome Surfer 2.0 (b) G-nome Surfer Pro displaying the chromosome visualizations, an aligned sequence, and publications

Contextual Inquiry

• An approach to ethnographic study where user is expert, designer is apprentice

• A form of interview, but
  — at users’ workplace (workstation)
  — 2 to 3 hours long

• Four main principles:
  — Context: see workplace & what happens
  — Partnership: user and developer collaborate
  — Interpretation: observations interpreted by user and developer together
  — Focus: project focus to understand what to look for
Considerations for data gathering (1)

• Identifying and involving stakeholders: users, managers, developers, customer reps?, union reps?, shareholders?

• Involving stakeholders: workshops, interviews, workplace studies, co-opt stakeholders onto the development team

• ‘Real’ users, not managers

• Political problems within the organisation

• Dominance of certain stakeholders

• Economic and business environment changes

• Balancing functional and usability demands
Considerations for data gathering (2)

- Requirements management: version control, ownership
- Communication between parties:
  - within development team
  - with customer/user
  - between users... different parts of an organisation use different terminology
- Domain knowledge distributed and implicit:
  - difficult to dig up and understand
  - knowledge articulation: how do you walk?
- Availability of key people
Data gathering guidelines

• Focus on identifying the stakeholders’ needs
• Involve all the stakeholder groups
• Involve more than one representative from each stakeholder group
• Use a combination of data gathering techniques
• Support the process with props such as prototypes and task descriptions
Data interpretation and analysis

• Start soon after data gathering session

• Initial interpretation before deeper analysis

• Different approaches emphasize different elements e.g. class diagrams for object-oriented systems, entity-relationship diagrams for data intensive systems
Task descriptions

• Scenarios
  — an informal narrative story, simple, ‘natural’, personal, not generalisable

• Use cases
  — assume interaction with a system
  — assume detailed understanding of the interaction

• Essential use cases
  — abstract away from the details
  — does not have the same assumptions as use cases
Task analysis

• Task descriptions are often used to envision new systems or devices

• Task analysis is used mainly to investigate an existing situation

• It is important not to focus on superficial activities
  – What are people trying to achieve?
  – Why are they trying to achieve it?
  – How are they going about it?

• Many techniques, the most popular is Hierarchical Task Analysis (HTA)
Hierarchical Task Analysis

- Involves breaking a task down into subtasks, then sub-sub-tasks and so on. These are grouped as plans which specify how the tasks might be performed in practice.

- HTA focuses on physical and observable actions, and includes looking at actions not related to software or an interaction device.

- Start with a user goal which is examined and the main tasks for achieving it are identified.

- Tasks are sub-divided into sub-tasks.
Summary

• Getting requirements right is crucial

• There are different kinds of requirement, each is significant for interaction design

• The most commonly-used techniques for data gathering are: questionnaires, interviews, focus groups, direct observation, studying documentation and researching similar products

• Scenarios, use cases and essential use cases can be used to articulate existing and envisioned work practices.

• Task analysis techniques such as HTA help to investigate existing systems and practices
Chapter 11

DESIGN, PROTOTYPING and CONSTRUCTION
Overview

- Prototyping
- Conceptual design
- Concrete design
- Using scenarios
- Generating prototypes
- Construction

- Uitdagingen uit de praktijk & tentamenttraining.
Prototyping

• What is a prototype?
• Why prototype?
• Different kinds of prototyping
  - Low fidelity
  - High fidelity
• Compromises in prototyping
  - Vertical
  - Horizontal
• Final product needs to be engineered
What is a prototype?

In other design fields a prototype is a small-scale model:

• a miniature car
• a miniature building or town
• the examples here come from a 3D printer

*Figure 11.1 (a) Color output from a 3D printer: all the gears and rods in this model were ‘printed’ in one pass from bottom to top, and when one gear is turned, the others turn too. Source: (a) The Computer Language Company, Inc., courtesy of Alan Freedman*
What is a prototype?

In interaction design it can be (among other things):

- a series of screen sketches
- a storyboard, i.e. a cartoon-like series of scenes
- a Powerpoint slide show
- a video simulating the use of a system
- a lump of wood (e.g. PalmPilot)
- a cardboard mock-up
- a piece of software with limited functionality written in the target language or in another language
Why prototype?

• Evaluation and feedback are central to interaction design

• Stakeholders can see, hold, interact with a prototype more easily than a document or a drawing

• Team members can communicate effectively

• You can test out ideas for yourself

• It encourages reflection: very important aspect of design

• Prototypes answer questions, and support designers in choosing between alternatives
(Filtering) dimensions of prototyping

<table>
<thead>
<tr>
<th>Filtering dimension</th>
<th>Example variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appearance</td>
<td>size; color; shape; margin; form; weight; texture; proportion; hardness; transparency; gradation; haptic; sound</td>
</tr>
<tr>
<td>Data</td>
<td>data size; data type (e.g., number; string; media); data use; privacy type; hierarchy; organization</td>
</tr>
<tr>
<td>Functionality</td>
<td>system function; users’ functionality need</td>
</tr>
<tr>
<td>Interactivity</td>
<td>input behavior; output behavior; feedback behavior; information behavior</td>
</tr>
<tr>
<td>Spatial structure</td>
<td>arrangement of interface or information elements; relationship among interface or information elements – which can be either two-or three-dimensional, intangible or tangible, or mixed</td>
</tr>
</tbody>
</table>
# Manifestation dimensions of prototyping

<table>
<thead>
<tr>
<th>Manifestation dimension</th>
<th>Definition</th>
<th>Example variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material</td>
<td>Medium (either visible or invisible) used to form a prototype</td>
<td>Physical media, e.g. paper, wood, and plastic; tools for manipulating physical matters, e.g. knife, scissors, pen, and sandpaper; computational prototyping tools, e.g. Macromedia Flash and Visual Basic; physical computing tools, e.g. Phidgets and Basic Stamps; available existing artifacts, e.g. a beeper to simulate a heart attack</td>
</tr>
<tr>
<td>Resolution</td>
<td>Level of detail or sophistication of what is manifested (corresponding to fidelity)</td>
<td>Accuracy of performance, e.g. feedback time responding to an input by a user (giving user feedback in a paper prototype is slower than in a computer-based one); appearance details; interactivity details; realistic versus faked data</td>
</tr>
<tr>
<td>Scope</td>
<td>Range of what is covered to be manifested</td>
<td>Level of contextualization, e.g. website color scheme testing with only color scheme charts or color schemes placed in a website layout structure; book search navigation usability testing with only the book search related interface or the whole navigation interface</td>
</tr>
</tbody>
</table>

**Table 11.2** The definition and variables of each manifestation dimension
What to prototype?

• Technical issues

• Work flow, task design

• Screen layouts and information display

• Difficult, controversial, critical areas
Low-fidelity Prototyping

• Uses a medium which is unlike the final medium, e.g. paper, cardboard

• Is quick, cheap and easily changed

• Examples:
  – sketches of screens, task sequences, etc
  – ‘post-it’ notes
  – storyboards
  – ‘Wizard-of-Oz’
<table>
<thead>
<tr>
<th>Type</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-fidelity prototype</td>
<td>Lower development cost</td>
<td>Limited error checking</td>
</tr>
<tr>
<td></td>
<td>Evaluates multiple design concepts</td>
<td>Poor detailed specification to code to</td>
</tr>
<tr>
<td></td>
<td>Useful communication device</td>
<td>Facilitator-driven</td>
</tr>
<tr>
<td></td>
<td>Addresses screen layout issues</td>
<td>Limited utility after requirements established</td>
</tr>
<tr>
<td></td>
<td>Useful for identifying market requirements</td>
<td>Limited usefulness for usability tests</td>
</tr>
<tr>
<td></td>
<td>Proof of concept</td>
<td>Navigational and flow limitations</td>
</tr>
<tr>
<td>High-fidelity prototype</td>
<td>Complete functionality</td>
<td>More resource-intensive to develop</td>
</tr>
<tr>
<td></td>
<td>Fully interactive</td>
<td>Time-consuming to create</td>
</tr>
<tr>
<td></td>
<td>User-driven</td>
<td>Inefficient for proof-of-concept designs</td>
</tr>
<tr>
<td></td>
<td>Clearly defines navigational scheme</td>
<td>Not effective for requirements gathering</td>
</tr>
<tr>
<td></td>
<td>Use for exploration and test</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Look and feel of final product</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Serves as a living specification</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Marketing and sales tool</td>
<td></td>
</tr>
</tbody>
</table>

Table 11.3 Advantages and disadvantages of low- and high-fidelity prototypes
Storyboards

• Often used with scenarios, bringing more detail, and a chance to role play

• It is a series of sketches showing how a user might progress through a task using the device

• Used early in design
Sketching

• Sketching is important to low-fidelity prototyping

• Don’t be inhibited about drawing ability. Practice simple symbols

Figure 11.5 A storyboard depicting how to fill a car with gas
Card-based prototypes

- Index cards (3 X 5 inches)

- Each card represents one screen or part of screen

- Often used in app and website development
‘Wizard-of-Oz’ as lofi prototyping

• The user thinks they are interacting with a computer, but a developer is responding to output rather than the system.

• Usually done early in design to understand users’ expectations
High-fidelity prototyping

- Uses materials that you would expect to be in the final product
- Prototype looks more like the final system than a low-fidelity version
- High-fidelity prototypes can be developed by integrating existing hardware and software components
- Danger that users think they have a complete system.......see compromises
‘Wizard-of-Oz’ as hifi prototyping

• The user thinks they are interacting with a computer, but a developer is responding to output rather than the system.

• Usually done early in design to understand users’ expectations
Compromises in prototyping

• All prototypes involve compromises

• For software-based prototyping maybe there is a slow response? sketchy icons? limited functionality?

• Two common types of compromise
  • horizontal: provide a wide range of functions, but with little detail
  • vertical: provide a lot of detail for only a few functions

• Compromises in prototypes mustn't be ignored. Product needs engineering
Concrete design

• Many aspects to concrete design
  – Color, icons, buttons, interaction devices etc.

• User characteristics and context
  – Accessibility, cross-cultural design

• Cultural website guidelines

successful products “are … bundles of social solutions. Inventors succeed in a particular culture because they understand the values, institutional arrangements, and economic notions of that culture.”
Using scenarios

• Express proposed or imagined situations

• Used throughout design in various ways
  – as a basis for overall design
  – scripts for user evaluation of prototypes
  – concrete examples of tasks
  – as a means of co-operation across professional boundaries

• Plus and minus scenarios to explore extreme cases
Generate storyboard from scenario

Figure 11.4 Some simple sketches for low-fidelity prototyping
An experience map drawn as a wheel

Figure 11.19 (a) An experience map using a wheel representation. (b) An example timeline design map illustrating how to capture different issues.

An experience map drawn as a timeline

Design Map: Megan delivers the presentation

last updated by TA, 2/16/06

Map Title

Step

Question

Comment

Megan logs on to the presentation system

Megan sees the presenter page

Megan sees that her slides are ready and she does a last-minute flip through

Megan fixes an error on one of her slides.

Megan sees that audience members are starting to arrive

Should we let Megan log on if Ivan hasn't set everything up yet?

The presenter page should reassure her that the streams are started and the preso is good to go.

Let's create a way for her to flip through her slides (and change them?) without any audience members seeing this process.

Are we going to be able to support this kind of last-minute change?

See Design Map: Sam arrives for the Presentation.

Megan has already uploaded all of her slides.

The presenter page should let her see that Ivan is there already.

What if an audience member tries to connect before Megan or even before Ivan?

(b)

Figure 11.19 Continued
Construction: physical computing

• Build and code prototypes using electronics

• Toolkits available include
  – Arduino
  – LilyPad (for fabrics)
  – Senseboard
  – MaKey MaKey

• Designed for use by wide range of people
What is BITalino? Is it for you?

DIY toolkit turns people's bodies into data-tracking machines

Board Kit
what's included
The one and only standard toolkit with our awesome all-in-one board.

Freestyle Kit
what's included
We break out all the blocks for you to make custom designs fast & easy.

Plugged Kit
what's included
Cabled sensor connection for interchangeable use of our sensors and custom designs as well.

OpenSignals
show me more
Our signature software for real-time data visualization and recording.
Construction: SDKs

• Software Development Kits
  – programming tools and components to develop for a specific platform, e.g. iOS

• Includes: IDE (Integrated Development Environment), documentation, drivers, sample code, application programming interfaces (APIs)

• Makes development much easier

• Microsoft’s Kinect SDK has been used in research
Summary

• Different kinds of prototyping are used for different purposes and at different stages
• Prototypes answer questions
• The final product must be engineered appropriately
• Two aspects of design: conceptual and concrete
• To generate conceptual design, consider interface metaphors, interaction types and interface types
• Storyboards can be generated from scenarios
• Card-based prototypes can be generated from use cases
• Physical computing kits and SDKs facilitate transition from design to construction
Chapter 12

INTERACTION DESIGN IN PRACTICE
Overview

• Recap / the old stuff
• Agile UX
• Lean UX
• Design Patterns
• Customer journey mapping
recap
Waterfall model

1. Requirements Analysis
2. Design
3. Development
4. Testing
5. Maintenance

Software Development Life Cycle
Agile development

• Short (one to three week) time boxes of iterative development (sprint, iteration, cycle)

• Early and repeated customer/user feedback

• Re-prioritisation of work based on customer/user so that emergent requirements can be handled

• Many approaches, e.g. eXtreme Programming (XP), Scrum, DSDM
AgileUX

- Integrates techniques from interaction design and Agile software development
- AgileUX requires a change of mindset
- In Agile, as implementation proceeds:
  - requirements are elaborated
  - requirements are re-prioritised
- All techniques in UX are still relevant but when and how much needs re-thinking
  - focus on product, not design, as deliverable
  - cross-functional teams
- Three practical areas: user research, aligning work practices, documentation
User research

- Aims to characterise users through data collection and analysis
- Agile’s time boxing approach does not support long periods of user research
- User evaluations and some detailed work can be fitted within a time box
- Some user research can be performed in iteration 0 (zero), before implementation starts
- Ongoing program of user research
Agile usability testing & User eXperience analytics
Old versus new style (1)

Waterfall UX (Bad!)

Agile UX (Good!)

 UX

 Dev

 UX

 Dev

 Project time
Old versus new style (2)

**Figure 1.** In a waterfall development cycle, analysis, design, coding, and quality assurance testing are separate stages of a software release that spans months or years. In Agile development, each of a set of incremental mini-releases (each created in 2-4 weeks) has these stages. Adapted from *Cutter Consortium* [8].
Old versus new style (3): Theory versus practice

Figure 2. In the perfect theoretical version of waterfall development, usability investigations contributing to the analysis and design phases were supposed to precede coding, but in reality developers would begin coding immediately.
# Old versus new style (4)

Table 1: Comparison of heavy- and light-weight processes exemplified by a few chosen aspects of principles and methods.

<table>
<thead>
<tr>
<th>Heavy-Weight Processes</th>
<th>Light-Weight Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detailed, up-to-date documentations and models</td>
<td>Cards and hand-drawn abstract models. Travel light. Communicate rather than document.</td>
</tr>
<tr>
<td>High-fidelity prototypes</td>
<td>Abstract prototypes, use simplest tools</td>
</tr>
<tr>
<td>Develop and prove concepts with user feedback. Iterate.</td>
<td>Courage. Design for needs (user's tasks) rather than user expectations. Retrieve design from models rather than continuous user feedback.</td>
</tr>
<tr>
<td>Time-consuming usability evaluations, workshops with intense stakeholder integration</td>
<td>Fast usability inspections. No need to evaluate if models are right.</td>
</tr>
<tr>
<td>Problem</td>
<td>Symptoms</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
</tr>
</tbody>
</table>
| Not enough design time. | • Developers waiting on designs  
• Design quality drop  
• Designs not verified with customers | • Separate and parallel UX Design/Developer tracks [1,2,9,12,13]  
• Scope UX activities to be small, incremental [1,2]  
• RITE formative usability testing [3,4]  
• Rapid contextual design [5]  
• "Design studio" [6]  
• Design chunking [1,14]  
• Combine different UX activities into one session [7]  
• Bring user (and data) to you [7]  
• Lighten requirements gathering process [1,2,9,11,12] |
| Sprints are too short | • Designs can’t be finished in time  
• No time for usability testing  
• No time to set up customer contact | |
| Not enough user feedback | • Feedback not early enough  
• No data to act on – opinions rule  
• Product isn’t validated | • UX person can act as Agile customer role [8]  
• Each UX person works on one scrum team [8]  
• Choose which scrum teams to work with wisely [11]  
• Validated designs are passed to developers to implement [1,2]  
• UX participates in cycle planning [2,15], bringing appropriate user feedback [1]  
• No features go in unless something comes out [11] |
| Weak Agile "customer" [6] | • End-users and clients won’t participate  
• Can’t get buy-in from rest of team  
• Non informed decisions are made | |
| UX is not full-time on one Agile team | • UX time spent in many meetings instead of on designs and iterations  
• Demoralized by UX quality drop | • Persuade Agile team to adopt Cycle Zero [1,2,9,10,12]  
• Consider design goals from different levels of detail (product, release, capability, design chunk) [14] |
| No sprint/cycle planning | • Large backlog of features/bugs  
• Prioritization feedback ignored  
• No control over timing of designs | • Include developers in design process [1,2]  
• Usability included in acceptance criteria [1,2]  
• Daily contact to check progress [1,2]  
• Design cards for stand-up progress [1]  
• Issue cards for usability reporting [1]  
• Documents are for design team [1] |
| User feedback is ignored | • Feature set is cast in stone  
• No time to incorporate changes  
• No re-ordering of features is allowed | |
| Missing the "big picture" | • No shared vision or end goal  
• Too much focus on details  
• Hard to prioritize/make design decisions | |
| Poor communication. | • Misunderstood designs  
• Agile team doesn’t buy into designs  
• Important information is lost | • Telecommuting tools (phone and web-based replacements) [11,12]  
• Co-locate for cycle planning [11,12] |
| Team isn't co-located | • No sense of team - lack of trust  
• Language and/or time barriers  
• Not enough communication | • A scrum leader or facilitator with strong persuasion skills can move things along quickly. [11] |
| Dependence issues | • Requiring input from non-Agile teams (e.g., marketing sign-offs, lawyers) | |
Aligning work practices

• Designing a complete product upfront causes problems because of re-prioritisation

• Some upfront work is needed (technical and UX)

• Use a parallel tracks approach:
  – create product vision before development starts
  – do design work one iteration ahead of development
  – some teams work two iterations ahead
Minimum Viable Product (MVP)

1. Declare Assumptions / Initial Understanding
2. Create an MVP
3. Run an Experiment
4. Feedback and Research

---

How not to build a minimum viable product:

1. 🙁 2. 🙁 3. 🙁 4. 😊

---

How to build a minimum viable product:

1. 😊 2. 😊 3. 😊 4. 😊 5. 😊

Kirill Shikhanov “MVP”. https://dribbble.com/shots/1753131-MVP
A LEAN UX DESIGN PROCESS

ANNE WANG
I'M A UX TEAM OF ONE.

DESIGN CAN CHANGE BUSINESS
DESIGN THINKING

UX AS A FACILITATOR,
NOT JUST A DESIGNER

LEAN = SKINNY
REDUCE INVENTORY

MAKE

THINK
Generative Research
Ideation
Mental Models
Behavior Models
Test Results
Competitive Analysis

MAKE
Prototypes
Wiresframes
Value Prop
Landing Page
Hypothos
Corps
Deployed Code

CHECK
All Testing
Site Analytics
Usability Testing
Funnel
Sign-up

MY LEAN UX PROCESS
WHAT & HOW FOR STARTUPS I WORKED

Discovery
Discover and research to define user scenario and conceptual model

Wireframe
Use Cases
Work Flow
Site Map
Screen Wireframing
Major Interaction

Prototype
Visual Design
Icons + Style Guide
Screen Mockup
Flash Prototype
iOS Prototype

Validate Internally
Get quick and frequent feedback from executive and team members

Test Externally
Meet with customers, find niche group of target users

Summarize
Validate and learn from user behavior

Iterate
Modify Update Plan and move to next cycle

User Feedback is the Key
@JEFF GOTHETF

Ideal
OBJECTIVES

Realistic
OBJECTIVES

Solution
OBJECTIVES

My Design Philosophy
CLEAN & SIMPLE

Less is More

www.anniestudio.org
annie@anniestudio.org

My Lean UX Process

Discovery
Online Research
User Analysis
Storyboard
Sketch Work
Brainstorming

Wireframe
Sketchbook
Whiteboard
OmniGraffle / Balsamiq
Illustrator
Powerpoint

Prototype
Photoshop
Illustrator
HTML+CSS+JS
Flash+Action Script
Blue Print

Validate Internally
Whiteboard
Pinboard
A-B Testing
Google Analytics
Survey Monkey

Test Externally
Customer Meeting
Decision of Next Step
RoadMap

Summarize
Discussion
Documentation / Wiki
Meeting Minutes / Wiki

Iterate
Modify
Update
Plan and move to next cycle

Photo Stash
iOS app, download Drom app store
THE DIFFERENCE BETWEEN Lean UX and Agile UX

Lean UX
Application of User Experience Design methods into product development, tailored to fit Build-Measure-Learn cycles.

Design united with business and development.

Methods
Inspired by startup development and The Lean Startup Methodology.

Agile UX
An attempt to integrate User Experience Design and Agile Software Development Methodology

UX Design team & Dev team working together.

Methodology
Inspired by Agile Manifesto. Forced to blossom by growing popularity of Agile Software Development Methodology.

Try Lean UX & Agile UX in UXPin - The UX Design App (http://uxpin.com)
Documentation

• Most common communication approach for UX designers

• Agile discourages this kind of communication, in favour of discussion

• Only use documentation where needed. Ask:
  – Who will read it?
  – Who will use it?
  – What is the minimum needed?
  – Is there duplication anywhere?
  – How polished does it need to be?
From chaos to collaboration
How transformative technologies will herald a new era in travel

The future of travel is likely to be shaped by technological and social innovations to reduce stress, uncertainty and to encourage collaboration among travellers and with travel providers. Click below to share the frustrations and desires you agree with.

**TRAVELLERS’ FRUSTRATIONS**

- **At the airport - automatic transit**: 47% going through security, passport control and customs
- **Taking the stress out of travel**: 51% over 50s - going through customs or passport control
- **Finding your way**: 47% time to research options
- **Next generation of experience**: 51% not knowing where the best local restaurants, bars and venues are
- **Travel services**: 47% lack of good quality advice
- **Work life balance for business tourists**: 51% not having access to home music, video and data

**TRAVELLERS’ EXPECTATIONS**

- **Automated identity**: management systems to track flows of people going through security and border control
- **Intelligent tickets**: can update the traveller on delays or changes
- **mHealth**: mobile technology and sensors to monitor heart rates and blood pressure
- **Intelligent recommendations**: information from expert blogs and online guides help create personalised travel guides
- **Augmented reality**: access historical and cultural information about a place and access peer reviews
- **Payment with memory**: memory of expenditure will enable intelligent passenger records
- **Cloud computing**: easy access to your own music, video and data from hotel rooms

- 53% happy: to provide more personal information for efficient travel
- 82% desire: a device to monitor and reduce levels of stress while travelling
- 86% desire: a personalised travel guide which aggregates recommendations
- 61% desire: a mobile application that overlays visual information about the physical world
- 47% happy: to provide personal data for a more personalised service
- 59% business travellers desire: complete access to everything they can get in their home
Overview of Customer-Centered Design

“You’ve gotta start with the customer experience and work back to the technology”
- Steve Jobs. WWDC 1997

Customer-Centered Design Process

1. Observe and empathize with customers' needs
2. Collect data
3. Brainstorm design ideas
4. Design a prototype based on observations
5. Test the prototype
6. Refine and make adjustments
7. Develop the final product

Customer-Centered Design Checklist

- ✓ Ethnographic observations
- ✓ Concept ideation
Customer-Centered Design Checklist

- Ethnographic observations
- Customer interviews
- Analysis of customer findings
- Concept ideation
- Rapid prototyping
- Concept validation by the customer

**CCD Benefits**

- Products that solve real customer problems
- Products that customers want to use
- Products that customers want to talk about!

**Technology driven**

- Component focused
- System driven use cases
- Success based on functional features
- System performance is a premium

**CCD driven**

- Customer needs focused
- Real-world use cases
- Success based on customer values
- Customer engagement is a premium

86% of consumers will pay more for a better experience

**How can Customer-Centered Design**

**LEAN UX**

Lean UX can use customer centered design throughout all phases of the project lifecycle to validate product value, market fit and the customer experience.
How can Customer-Centered Design fit into Agile, Waterfall and Lean UX?

**WATERFALL**
Waterfall’s linear approach makes the beginning of the product design lifecycle the best place for customer centered design to have an impact. Using it at the end of the process is possible but can be costly.

**LEAN UX**
Lean UX can use customer centered design throughout all phases of the project lifecycle to validate product value, market fit and the customer experience.

Customer-Centered Design in a Lean UX process

When to Consider
- Limited design time
- Solution based iterations
- Team and budget constraints

42% of online projects are underfunded and under staffed
Customer-Centered Design in a Waterfall process

When to Consider
- Limited design time
- Solution-based iterations
- Team collaboration

42% of online projects are underfunded and undermanned when it comes to UX

When to Consider
- Fixed-scope
- Fixed-price
- Clear vision

15% of fail points can be fixed with the addition of customer-centered design into a waterfall process

AGILE

Agile allows customer-centered design to have an impact within each sprint but more at a component or feature level. Concept pivots can be challenging.

The cost of not using Customer-Centered Design
The cost of not using Customer-Centered Design

$83 Billion in lost annually due to poor customer experiences online (US)

100x more expensive to correct a problem after release vs solving in the design phase

Learn more about Customer-Centered Design:
- Contextual Design: Defining Customer-Centered Systems -(by Hugh Beyer and Karen Holtzblatt)
- Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation (by Tim Brown)
- Design of Sites, The: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience (by By Douglas K. van Duyne, James A. Landay, Jason I. Hong)

References:
Summary

- AgileUX refers to approaches that integrate UX design and agile development
  - it requires a change in mindset by designers and developers
  - requirements are repeatedly re-prioritised, which aims to avoid wasted effort
  - UX design activities need rethinking: when, how much, and how to take forward
- Design patterns present a solution to a problem in a context
- Open source resources, e.g. on Github, make development of standard applications easier and quicker
- A range of automated tools to support interaction design in practice is available