# UX design in the wild

## Lateral Nord's tips to successful product developer life

Lateral Nord.

# Jerry Jäppinen

## Product design consultant

DC MN · elisa · DIGITA · OLX GROUP · ProductBeat

jerryjappinen@lateralnord.com

+358 40 7188776

@jerryjappinen

d.

# About me

- Jerry Jäppinen, Senior Designer
- @jerryjappinen on Twitter
- Worked on the new Sovelia web client 2014-2015
- Background in web design, games, web + mobile applications
- And philosophy

# Today we talk about

- What product development is
- UX as a tool to evaluate of your own work as a product developer
- **Design thinking** as a problem solving tool
- Practical tools to introducing UX considerations into your (team) workflow
- Sovelia web client introduction (case study)
- Evolution and future of web
- Web vs. desktop interfaces

# Product development

Development of things that work and that people care about.
Those are the two things that matter.

# What we call design

- Individual disciplines/vocations
  - Graphic design
  - Motion design
  - Typography
  - Information architecture
  - Concept design
  - etc.

- Overall product development process

# What are UX and UI?

- UX short for **user experience**
- UI short for **user interface**
- That’s exactly what they mean! What a silly question!
- Do we ever ask what the difference between **car dashboard** and **driver experience** is?
- One is a catch-all term for human thought and emotions and experiences. The other is a software component.

# User experience is…

- The combined expectations and experiences of the user
- Emergent and always there
  - You can’t **design** a user experience
  - But you can facilitate certain kinds of experiences
- Since it’s always there, **we can use it as a measurement**
- What is a **good** UX then?
  - We can use multiple metrics (Revenue, error rate, surveyed user satisfaction…)
  - But it's always determined by end-users

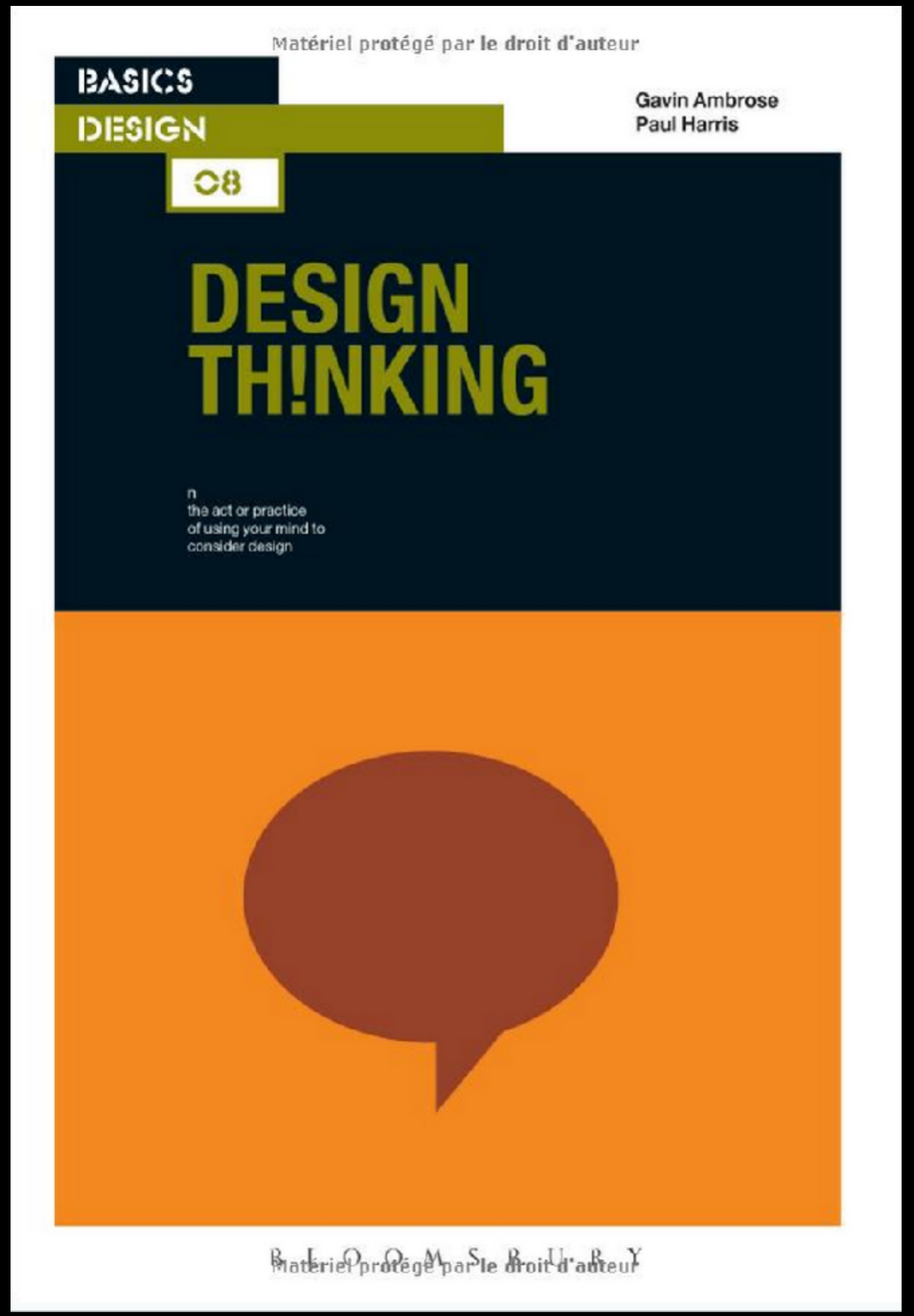
**Design is a problem solving process**
**UX is measure of end result**

# Product design is everyone's business

# Design process

# Design thinking

1. Define
2. Research
3. Ideate
4. Prototype
5. Choose
6. Implement
7. Learn



# Define & research

**Stage 1 – Define**

**Establishing what the problem is.**

This is the first stage in any design process and almost always involves generating or receiving a design brief.

**Stage 2 – Research**

**Collecting background information.**

- If you don’t do this, you should
- Write down any loose requirements you have been given or can formalize yourself
- A brief can be refined later in agile software projects
- Brief can be one sentence or a pages long document, but not overly detailed

# Ideate

**Stage 3 – Ideate**

**Creating potential solutions.**

During the ideate stage, the design team draws on the research gathered and the constraints established during the define stage. This information is used to create ideas with which to tackle the design brief.

- Ideating is about identifying *potential* solutions
- Standards for a potential solution should be very low
- Bad solutions are weeded out later
- This is the *creative* part

# Prototype

**Stage 4 – Prototype**

**Resolving solutions.**

The ideate stage generates a variety of potential solutions to the design brief. Prior to selection, it may be necessary to further work up the most promising of these solutions. This will allow particular aspects to be tested and will provide a better basis for comparison at the selection stage. In such cases a prototype can be created.

- Prototyping is cheap in software development
- Work to make it even cheaper for your team
- Prototypes can (sometimes) be extended into final solutions

# Choose

- You need tools for coherent decision making
  - reread the brief
  - get data on your product and users
- Try to use reason and not emotion
- But stay emphatic to your *users'* emotions!

**Checklist:**

| |
|---|
| Does the design meet the defined needs of the brief? |
| Does the design resonate with the target audience? |
| Can the design be produced on time and on budget? |
| Are there other factors to take into account? |
| Has the client signed off the design? |

## Stage 6 – Implement

**Delivering the solution to the design brief.**

- This is what many developers think is their territory
- And some designers too: graphic designers might execute technical graphic design based on requirements gathered by a concept designer or a manager.
- The implementation part might of course take years
- Might seem stupid to have it in one step. But, then we have

## Stage 7 – Learn

## Obtaining feedback.

The final stage in the process involves learning from what has happened throughout the design process. This is a feedback stage during which the client and design agency might seek to identify what worked well and where there is room for improvement.

Following the implementation, the client may begin to look for or receive feedback on how the product has been received by the target audience and how beneficial its effects on the target audience have been. Thus, a design firm can find out how the audience responded to the design.

- Software-based service or product development is iterative and cyclical
- The process can and should be repeated
- The model can describe overall process as well as individual feature development

# Iterate

Apply the same method for creating the next version.
Write a better brief.
Make more informed choices.
Craft a better implementation.

# Practical tools

# UX maxims

- Easy, understandable pattern to assess and improve UX
- Classical examples
  - System state must be visible
  - Location in application should be clearly marked
- More down-to-earth and useful
  - Menus should always have one and only one item selected
  - All clickable items should have hover feedback
  - Hit areas should be at least 9 mm x 9 mm
  - Design and develop layouts mobile-first

# UX myths

- Myth #13: Icons enhance usability
- Myth #15: Users make optimal choices
- Myth #21: People can tell you what they want
- Myth #29: People are rational
- Lesson of UX myths: a theory won’t get you a good UX
- uxmyths.com et al

# Use cases

- More modern way to define and measure UX
- Used both as a form of documenting requirements as well as assessing successful UX
- Can be used as a basis of user stories and development tasks
- Danger close! User interfaces are systems
  - Use cases offer **particular sample data**
  - You’re responsible of creating coherent systems and architectures, both in design and tech.

# Lessons from the web

And lessons *for* the web

# History of web

- Web is a platform
- Long, interesting, gradual evolution
- Loosely-controlled interoperable technologies
- Insane backwards-compatibility
- Standards seen as only having instrumental reference value
  - Implementations (rendering engines) carry more meaning now

# Promise of web

- There’s a clear promise to users!
- Always functional (backwards-compatibility)
- Doesn’t break between versions
- Always up-to-date (hosted on server)
- Distributed computing. Services and content, not apps and tools.

# Rules of the web

- User’s attention span is ridiculously low
  - This will affect all apps (loading times etc…)
- Only one-click interaction available
  - Right-click hacks are available but users won’t discover them
  - The same goes for keyboard shortcuts

# Rules of the web

- No loading times
- Everything is synced and stored in the cloud
- Page refreshes are somehow acceptable, but not for long
- All implementation requires a fair bit of hacking
- Tools will change all the time
- Soon users will expect nothing less. Tough crowd.

# Native applications

- Users expect interoperability with other services, cloud storage, syncing etc. no matter the platform
- Native implementations are married to server-side services more and more
- Evolution of Windows OSs offer clues of trends

# Is desktop different?

- Traditional environments lack the emphasis on design, poor design guidelines etc.
- Web patterns are rooted in marketing communications. Modern native environments do have strong design-driven guidelines.
- Are "mouse cursors bad?"
  - They're not
  - But touch is great - infinitely more intuitive
  - Pointers only exist because we couldn't touch screens before
  - They're great for environments that require precision

# That's it!

# Jerry Jäppinen

## Product design consultant

DC MN · elisa · DIGITA · OLX GROUP · ProductBeat

jerryjappinen@lateralnord.com
+358 40 7188776
@jerryjappinen

Lateral Nord.