Can traditional web design be better?

A study on a variety of theories, methods and frameworks for the improvement of web design

Master thesis by Elitsa Dimitrova,

A student in Persuasive Design 2020, from Aalborg University

52 pages (124.897 characters incl. spaces)

Personal information

Name: Elitsa Dimitrova

Address: Sebbersundvej 39, 2 TH, 9220 Aalborg Øst

Email (university): edimit18@student.aau.dk

Email (personal): elitsa_d@hotmail.com

Project title: *Can traditional web design be better? A study on a variety of theories, methods and frameworks for the improvement of web design*

Knowledge resource group affiliation: E-LEARNING LAB - CENTER FOR USER DRIVEN INNOVATION, LEARNING AND DESIGN (ELL)

Table of Contents

Introduction

Websites have come a long way since they first became a thing. Nowadays, they are all around us, and serve all kinds of purposes - from encyclopedias like Wikipedia, social media platforms like Facebook and Instagram, professional platforms like LinkedIn, portfolio websites like Behance, and freelancing services like Fiverr and Freelancer.com, to video sharing platforms like Youtube and forums like Reddit. Not to mention all the company websites used for advertising specific products and brands. In a sense, websites are everywhere around us - and people do not really realize the extent to which we have come to rely on them for obtaining information.

Despite websites being an integral part of modern life, very few people are even aware of one of the key components of a website - its information architecture - and what is its function. To put it in simple terms, information architecture is the way we structure information based on similarities in the types of information so it is easier to find, understand, process, and use. This inspires a question - if information architecture is as important, why are people not aware of it? Unfortunately for us information architects (well, soon-to-be information architects, in my case) good information architecture is often invisible - and people should be busy with perceiving the information itself, not its structure.

Speaking of good information architecture, it is sadly not sufficient by itself if we want to retain users on a website for longer periods of time. Generally, if users are to stay and browse a website beyond sticking to the home page for a couple of seconds, a balanced, aesthetically pleasing website design is also necessary. Both the information architecture and the design are integral parts of good, positive user experience.

However, perhaps due to the presence of websites in all kinds of areas, I have found that the process which leads to their creation is often grossly underestimated by people outside the industry. I have been told in person and read opinions that designing a website is easy, takes a little time and effort, and that - my favorite claim - anyone could do it.

In theory, people who make such claims are not wrong - especially since there are content management systems (CMS) like WordPress or website creators like Wix that allow anyone to design and launch their own website. The internet might be full of all kinds of websites because of that to an extent. In practice, however, these websites tend to be of inferior quality compared to those created by specialists - which is understandable, and hopefully the case in other industries as well (or I would feel very concerned about visiting my doctor, for example).

That being said, a website design process does not always go as planned, despite what web designers would like. That brings another question to the table - where do things go wrong? And should the website design process be, in fact, easier and less time-consuming?

Problem Description

To answer the last question from the introduction, ideally yes. As for where things go wrong, a survey conducted by HubSpot with more than 100 marketers as cited by **Summerfield (2017)** claims that emotions that people feel while redesigning a website are mostly negative, such as “frustration”, “stress”, and “overwhelmed”. Some even listed “vacation” as something they were thinking about while working on a website. That might be an indicator that something is, indeed, not going well.

According to the same source, traditional website design is risky for businesses for several reasons. For example, website design for a small to medium-sized business takes three to six months on average and can cost between $15.000 to $80.000 **(Summerfield, 2017)**. Despite the large upfront cost and the amount of dedicated resources, energy and time, web design projects can still go over their budget and delivery date.

Another point of view presented by **Summerfield (2017)** is that while all marketers agreed that a brand’s website is crucial for its business growth, only 42% of them made any impactful improvements (or, in other words, optimization and adding key parts of the website in order to drive business value - not offers or blog posts!) only once a year or less.

From my personal experience, traditional website design can be slow and cumbersome. Granted, I have not been a part of an experienced web development team yet, but from what I have seen, maintaining documentation and tracking milestones or team performance are not often a priority. Once there is a mishap and the team goes behind schedule, the said schedule and planning get completely disregarded and consequent challenges are tackled as they come with little to no prior planning. Communication between team members is not always optimal (for example, two people end up parallelly working on the same task without collaborating with each other), and user testing is insufficient if conducted at all.

This brought me to my motivation for the current project. While I have been studying and implementing several methodologies and theories during website design processes, neither have really “clicked” or produced satisfactory results as fast as I would hope. As a result, I began looking into other methods, theories, and frameworks, as well as deepening my knowledge on those I was already aware of during my studies and professional development. Some of my discoveries included the Lean philosophies, such as Lean Startup and Lean UX, Growth-Driven Design, information architecture, as well as a better understanding of user experience (UX) design, research methodologies, and agile software development.

Since I expanded my toolkit (still an ongoing process, mind you - it is never late to learn something new), I have been eager for an opportunity to test what I could do if I combined some of the methods, theories and frameworks into a website design process of my own. While I have no pretenses to be a web design guru, ninja, or superstar (see the principles of Lean UX), I believe it will make for an interesting experiment.

Problem formulation

Based on what I have said so far, my initial problem formulation will be along these lines:

What methods, theories, and frameworks can be used in order to further streamline the traditional website development process without sacrificing the quality and the functionality of the final product?

Project limitations

The purpose of the project is purely exploratory and therefore it will not include an actual website prototype created by following the proposed project design.

My goal is to explore different web design processes, how they can be altered in order to adhere to the Lean UX philosophy and how and where would information architecture fit into the process. Some other methods, such as Growth-Driven Design, will be discussed as well, but not necessarily used in the design plan. Research methodology is also described purely as a point of reference, since it is a core component of UX processes, but is not actually shown to be utilized.

Lastly, my idea to present mockups for a website based on one of the methods was put on hold due to time constraints and due to the recent COVID-19 outbreak. Good mockups and prototypes require several iterations of user testing, which have been made time-consuming and cumbersome thanks to the epidemic.

Reading Guide

The APA referencing system is used throughout the report.

In the Methodology chapter, I have covered the methods and theories that I used as a point of reference, such as information architecture, user experience, Lean UX, and research methodologies. I have also discussed users and their relationship with design, as well as methods and frameworks such as Growth-Driven Design and Agile and Lean methods. It also includes a subchapter on my considerations behind these methods, and a literature overview.

In the Analysis chapter, I present the reasons for why traditional web development is no longer efficient, and afterwards suggest two web development methods based on Lean UX - one combined with Scrum, and one combined with Kanban. I also discuss their advantages and disadvantages. I conclude the chapter with a sample project timeline based on the Kanban variant for the creation of a portfolio website.

Afterwards I draw my conclusions from this thesis, and provide a perspective on how I wish it had gone differently and how I feel about the whole process.

Methodology

In this chapter, I will describe the methods and theories I will use in my plan's design, as well as discuss the reasons behind the selections I have made. For convenience, it is split up into several subchapters - "Theory", "Methods and Frameworks", and "Considerations and selection process".

Theory

In this subchapter, I will explain the theory behind information architecture, user experience (UX), and Lean UX - the three core pillars behind my project. Besides them, I will also explain about qualitative and quantitative research methods - they will not be used in this paper per se, but they are a necessary step in a website development process and will be discussed as such.

Information Architecture (IA)

Information can be processed differently by different readers or receivers because of the challenges of language and interpretation. That is where information architecture comes in - it makes definitions more findable and understandable. However, due to the different ways people understand information, information architecture can be defined in several different ways **(Rosenfeld, Morville, and Arango, 2015, p. 24)**:

1. "*The structural design of shared information environments.*

2. *The synthesis of organization, labeling, search, and navigation systems within digital, physical, and cross-channel ecosystems.*

3. *The art and science of shaping information products and experiences to support usability, findability, and understanding.*

4. *An emerging discipline and community of practice focused on bringing principles of design and architecture to the digital landscape.*"

To describe it in a little more detail, information architecture can be summarized as a set of guidelines for structuring information based on its similarities, and managing it in a way that makes processing and finding it as user-friendly as possible **(Lange-Sadzińska, 2012, p. 2)**. Although I said earlier that I perceived my internship project's information architecture strategy as not good enough due to relying on "gut feeling" too much, intuition can actually be considered a good thing as well. Information architecture should not rely only on metrics and "raw" data, but also on creativity, experience, and the ability and desire to take risks **(Rosenfeld et al, 2015, p. 26)**.

Sadly for information architects who put their time into meticulously designing it, good information architecture is practically "invisible" and not consciously felt by the users **(Rosenfeld et al, 2015, p. 26)**. The reason behind this phenomenon is that users should ideally just process information, and not think about why the information they perceive is structured in the way they see.

Information ecology

The core of information architecture design is information ecology, or the correlation between context, content, and users **(Rosenfeld et al, 2015, p. 32)**. It helps information architects obtain a better understanding of the resources available, the business goals behind the project, the volume and nature of existing content and how it might change, as well as the information-seeking behaviors and needs of the users. See figure 01 below for a Venn diagram of the concept.

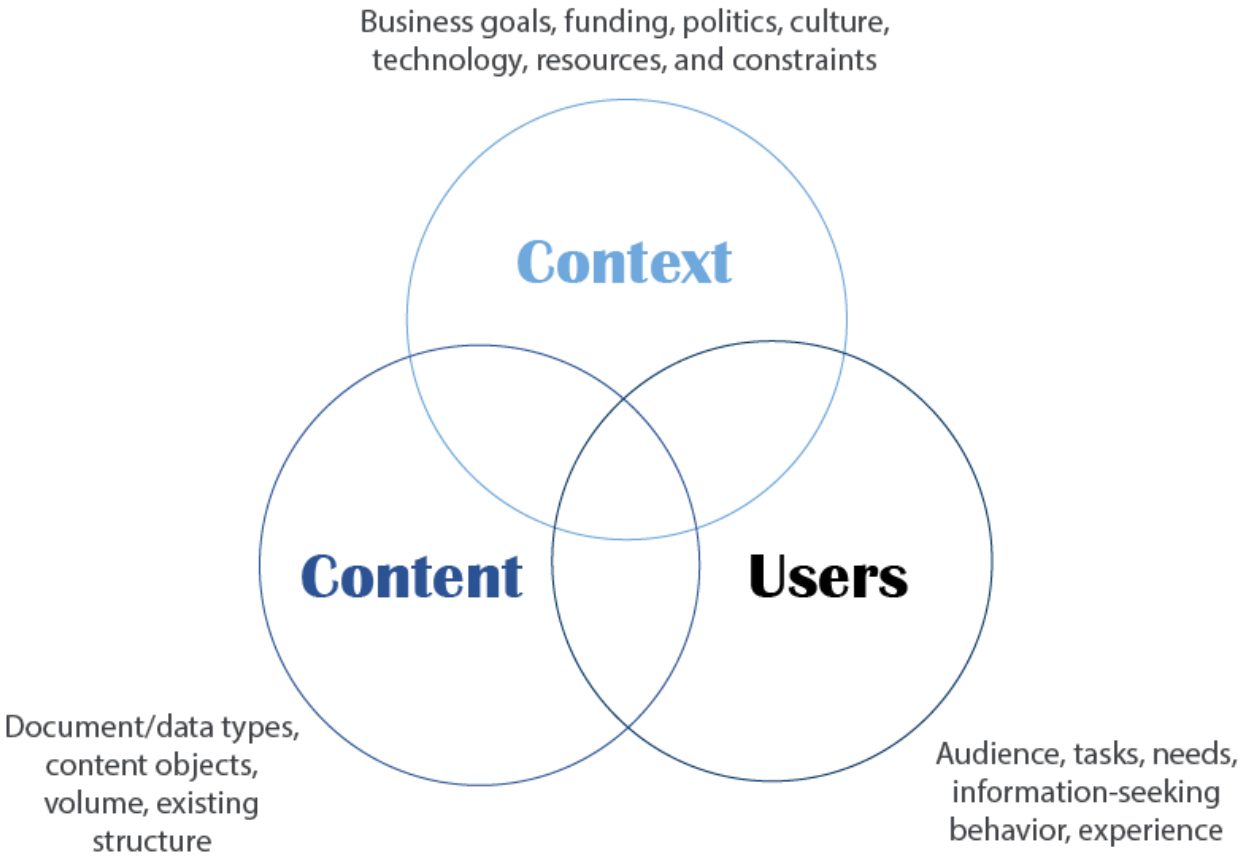

Figure 01 - a Venn diagram of the components of information ecology. Copied from Rosenfeld et al, 2015, p. 32.

Context

As Rosenfeld, Morville, and Arango have stated, "*All digital design projects exist within a particular business or organizational context. … This collective mix of capabilities, and resources is unique to each organization. Because of this, information architectures must be uniquely matched to their contexts.*" **(Rosenfeld et al, 2015, p. 32)**. Information architecture for a specific project is highly dependent, for example, on the business context - what makes the client unique, who are they and where do they want to take their business in the future.

Another context to be considered is the channels that the user will be using to interact with the product (e.g. a desktop-based browser on a PC, or a handheld device like a smartphone or a tablet) since they also influence aspects of the information architecture by limiting the label length, requiring shorter navigation menus, etc.

Content

If we describe it broadly, content can include anything from application and services to documents, metadata, and schemas **(Rosenfeld et al, 2015, p. 35)**. However, there are several facets of content that need to be taken into consideration if we want to see the bigger picture.

For starters, ownership - meaning, who owns and creates the content, how much of it is produced by users, how much of it is licensed from external sources, etc. The level of ownership directly corresponds with the level of control over all the other facets.

Format, on the other hand, can cover anything from documents, databases, technical reports, video clips, audio files, product catalogs, etc. They can all be accessed via websites and intranets, which have become the go-to means of access to digital formats for many organizations.

Structure, the third facet, deals with the granularity of content (or, in other words, the smallest discrete units which content can be broken into).

The fourth one, metadata, concerns the extent to which metadata which describes a system's objects and content has already been created. Other areas of concern are the way documents were tagged (manually or automatically), the level of quality and consistency, whether there are controlled vocabularies in place, etc.

Volume deals with the size of a system (e.g. how many pages, applications, documents it consists of), while dynamism, the last facet, describes the content turnover and growth rate.

Users

The third aspect of information ecology is the people who use the system, or, its users **(Rosenfeld et al, 2015, p. 37)**. An information architect should always remember that users are before all else people, each with their own preferences, needs, and ways of perceiving and

using information. Depending on the target group, the way information is accessed and what types of information are searched may vary greatly. The best way to determine that is to study users “in the wild” and not during brainstorming sessions.

Basic principles of information architecture

There are four systems in information architecture - organization, labeling, navigation, and search. Another key principle is metadata, thesauri, and controlled vocabularies.

Organization systems

As their name suggests, organization systems help us organize information by common attributes. There are several aspects of organizing information that can make the endeavor easier or more difficult, namely ambiguity, heterogeneity, the difference in perspectives, and internal politics.

The first aspect I mentioned, ambiguity, is one of those that makes things harder for information architects. One word can have multiple meanings - and the meaning a user thinks of by default might not be the same one as what the information architect thinks of **(Rosenfeld et al, 2015, p. 100)**. Besides definitions, another facet of information architecture that can be influenced by ambiguity is the categories we put information into - this becomes especially evident when we are organizing abstract concepts such as functions, topics, or subjects.

The second aspect is heterogeneity, or a “collection of objects composed of unrelated or unlike parts” **(Rosenfeld et al, 2015, p. 100)**. Its opposite, homogeneity, on the other hand, refers to things composed of identical or at least similar elements. Most information environments, websites included, are heterogenous - they display different types of content side by side, such as text, images, videos, databases, documents, and so on. That being said, elements of a different type and granularity should not be treated and organized the same way. For example, a magazine and a single article should be handled in different ways, as well as images of different formats and resolutions.

On another hand, the difference in perspective, or how different people perceive, understand, and categorize information, also comes into play when considering how to organize information. An organization system that might make perfect sense to me and have a logic behind it that is obvious in my eyes can be a chaotic mess according to someone else, and vice versa **(Rosenfeld et al, 2015, p. 102)**. In order to avoid the fallacy where an information architect designs an organization system according to their own understanding and preferences, instead of those that the users actually prefer, it is recommended to conduct extensive user research, analysis, and testing.

Finally, internal politics. The choices for labeling, organization and presentation of information on a company’s website or intranet can have a big impact on how users perceive said company, be it customers or even employees **(Rosenfeld et al, 2015, p. 103)**. While politics in a company or organization can make the creation of information architecture more complex and difficult, it

is also an opportunity to manage their impact on the architecture if the information architect is aware of them.

Besides the aspects I mentioned until now, there is more to organization systems - namely, how we organize information environments. Organization systems consist of organization schemes and organization structures. Organization schemes deal with the shared characteristics of content elements and influence the logic by which we group them, while organization structures are about the types of relationships that content groups and items have between each other **(Rosenfeld et al, 2015, p. 103**) Both schemes and structures are composed of several types each.

Organization systems can be two types - exact (that divide information into clear and mutually exclusive categories; they are generally easier to work with because of being more objective) or ambiguous (that defy exact definition; they are generally harder to work with due to their subjective nature). Examples of exact organization systems are, for instance, alphabetical (sorted by name), chronological (sorted by date), or geographical (sorted by location). Ambiguous organization systems, on the other hand, can be topical (by topic), task-oriented (organizing content into groups of tasks, processes, or functions), audience-specific (e.g. a website having a section for customers and for employees), and metaphor-driven (that help users understand what is unfamiliar to them by correlating it to something they know, like the Recycle Bin on a desktop). There is also the possibility of mixing several types of exact and ambiguous schemes, thus creating a hybrid scheme. However, they are difficult to design in a way that makes them understandable and user-friendly and, as such, are best avoided if possible **(Rosenfeld et al, 2015, p. 114)**.

As for organization structures, they are split into several types. The first one is hierarchies, or the top-down approach **(Rosenfeld et al, 2015, p. 117).** Family trees, biological taxonomies, organization charts, sitemaps, and books usually use a hierarchical organization scheme (e.g. books are split into chapters, which consist of paragraphs that are made up of sentences, which are written with words that are put together with letters). Hierarchies can be narrow and deep (a small number of categories which branch out into several subcategories each that may also branch out further) or broad and shallow (many categories that branch out into only a few subcategories that rarely branch out further than that). The second type of organization structure is the database model, or a bottom-up approach **(Rosenfeld et al, 2015, p. 122)**. Their main importance in relation to information architecture is metadata, or "the primary key that links information architecture to the design of database schemas" **(Rosenfeld et al, 2015, p. 124)**. This model is especially useful when utilized in homogenous websites or pages, such as staff directories, product catalogs, and such.

Apart from organization schemes and structures, there are other ways to organize information - namely, hypertext and social classification. Hypertext is a nonlinear way to structure information that consists of two types of components - the chunks of information or items that will be linked, as well as the links between them **(Rosenfeld et al, 2015, p. 126)**. While hypertext allows for great flexibility, it relies heavily on a person's associations for different

types of content and the links they see between them. As such, it is not an optimal choice for a primary organization system, since it makes it easy for users to become confused and frustrated due to the lack of context in their eyes. Instead, it can be used as a supplementary system to a hierarchical or database model. The second type, social classification, is user-driven content tagging, primarily seen on social media **(Rosenfeld et al, 2015, p. 127)**. Examples include hashtags (#) and skill endorsements on LinkedIn. When this system is utilized via a large number of users, it can create opportunities for shifting user behavior and tagging patterns into new navigation and organization systems.

Labeling systems

The second information architecture system is the labeling system. As its name suggests, it is primarily concerned with the way we label items. On another hand, a label's main goal is to "explain" the meaning of an item without taking much of a page's space, or a user's cognitive power **(Rosenfeld et al, 2015, p. 134)**. While designing information architecture for a system, it is of great importance to be aware of the users' understanding of the labels we apply, and whether they are the right ones for our target audience - poorly chosen ones can confuse and drive users away instead.

There are several types of labels - contextual links, headings, labels in navigation systems, index terms, and iconic labels. Contextual links occur naturally within text, and are thus easy to create. However, that does not make them easy to utilize **(Rosenfeld et al, 2015, p. 141)**. They are highly dependent on the text author's subconscious associations with information, and as such, they are heterogeneous and unsystematic.

Another way in which labeling can be utilized is through headers. Unlike contextual links, they usually have a clear hierarchy of headings and subheadings that can be visually displayed via different styles and colors **(Rosenfeld et al, 2015, p. 144)**.

The third category, labels in navigation systems, demands a very consistent application of labels within a system - more so than any other type of label **(Rosenfeld et al, 2015, p. 147)**. Users typically rely on a navigation system to be predictable, and well-applied labels that do not change from one page to another are crucial for creating a sense of familiarity. For example, if we want to hint at users where a company's contact information is, we could use labels like "Contact" or "Contact Us". However, using both at different pages would confuse users, since they will try to find out what is the difference between the two.

Index terms are also referred to as tags, keywords, taxonomies, descriptive metadata, thesauri, and controlled vocabularies **(Rosenfeld et al, 2015, p. 149)**. They can be used to describe virtually any type of content, be it pages, sites, content chunks, etc. Index terms are also optimized for more precise searching since they describe the content's meaning - searching the index terms should be more successful in producing satisfactory results than searching full text. They are also used in indexes, and as such can make browsing easier by providing an alternative to a primary organization system.

The last type, iconic labels, have a name that hints at their nature - they utilize image icons instead of a written label **(Rosenfeld et al, 2015, p. 152)**. While they add to the visual appeal of the information environment, they pose the same problem as some other types of labels, since they depend on the users' association with any given icon. For example, an icon of a running man could mean "Run" to some, and "Exercise" to others. Despite that, they are still an important component of many information environments due to the widespread use of touchscreen handheld devices which have a limited screen size.

Navigation systems

At first glance, navigation systems in information architecture deal with finding our way in an information environment and they are standalone systems on the same level as the rest. However, in reality, all four systems, as well as the environment's structure contribute to an effective navigation **(Rosenfeld et al, 2015, p. 176)**. As with the other systems discussed so far, navigation systems also have several subsystems. There are two types - primary, which is split into global, local and contextual embedded navigation systems, and supplemental - such as sitemaps, indexes, and guides. Embedded systems are most common in desktop-oriented websites, but are also present in mobile websites, albeit to a smaller extent **(Rosenfeld et al, 2015, p. 183)**.

The first subtype, global navigation systems, are the ones that are made to be visible on every page throughout an entire website, e.g. a navigation bar at the top of the page **(Rosenfeld et al, 2015, p. 183)**. They allow for easy access to key functions, no matter where the user is on the website at the moment.

Local navigation systems, on the other hand, can be seen in tandem with global navigation systems, especially in large websites, where the sections are so diverse that they can be referred to as sub-sites, or sites within sites. Sub-sites often have different navigation systems since they often feature different types of content, are aimed at different target audiences, or have different people that are responsible for content areas.

Contextual navigation relies on the contextual links between objects and on the users' desire to explore the predefined relationships between said objects or items **(Rosenfeld et al, 2015, p. 188)**. Therefore, contextual navigation systems are less structured than global and local systems. As other less structured subsystems in information architecture, contextual navigation systems can cause problems when they are crucial to the content, since users tend to scan content, jump across the page and thus skip over or ignore less noticeable links.

Supplemental navigation systems, on the other hand, provide additional ways to navigate a website, complete tasks, and find content **(Rosenfeld et al, 2015, p. 193)**. The first subtype, sitemaps, usually visualize the several top layers of an information hierarchy - their equivalent in a book would be the table of contents, and in the beginning, the two terms were used interchangeably. Sitemaps are usually used in large systems with a hierarchical organization. In case the opposite applies, an index might be a better option.

In their digital variant, indexes are a visual representation of phrases or keywords that are sorted alphabetically. However, unlike table of contents, they usually have only one or two levels of depth **(Rosenfeld et al, 2015, p. 195)**. One of the biggest challenges with indexing websites is the level of granularity involved, e.g. if we index whole pages, paragraphs, terms, etc.

The last subtype is guides, such as walkthroughs and tutorials. Their primary function is to introduce new users to the content of a website, as well as its functionality **(Rosenfeld et al, 2015, p. 198)**. They usually feature linear navigation, since new users prefer to be guided along the information environment and not get thrown in the "deep end of the pool" from the start. Wizards and configurators that help users with navigating complex decision trees or configure products can be considered a special class of guides **(Rosenfeld et al, 2015, p. 200)**, such as an option to choose the color of a product on Amazon.

While a searching system is an integral part of a good supplemental navigation, they will be discussed in more detail in the next part of the report.

Search systems

An information environment might need a search system depending on several factors. The first is the amount of content in the environment. While it is hard to say how big of an environment warrants a search system **(Rosenfeld et al, 2015, p. 212)**, the volume and types of content, as well as how much time the system will require for maintenance compared to how much users utilize it should be considered as deciding factors.

While search systems are extremely useful without a doubt, in many cases they are often implemented as a panacea for a poorly designed navigation system **(Rosenfeld et al, 2015, p. 212)**. In that case, focusing on fixing the navigation system first instead of implementing a search system is advised.

Another possible issue that may impact the decision of using a search engine is the time needed to optimize it, as well as whether an information architect knows how to do that. Poor search results are often the consequence of an unoptimized search engine that was left at its default settings and then forgotten about during development **(Rosenfeld et al, 2015, p. 213)**.

As I mentioned, depending on the information architect's expertise, another solution such as an index might be a better choice than a search engine (for example, the skills to configure it properly, or the lack of funding for one). Yet another option is to provide access to a third party search engine, like Google or Bing. That option also has its downsides, however - the inability to generate the same data from search analytics, for one **(Rosenfeld et al, 2015, p. 213)**.

Another consideration for the choice of implementing a search engine is the way users prefer to interact with the system. Searching as a means of navigation may be of a lower priority to them, and as such, get delegated a smaller part of the information architecture budget.

Despite all the warnings discussed so far, there are several instances and scenarios when a search engine might be warranted **(Rosenfeld et al, 2015, p. 214)**. One such case is when an information environment has too much information to be browsed, or when there are fragmented websites like large public websites or intranets. Besides that, search-log analysis provides valuable data on what users want or need from an information environment. Another option to consider is that users simply expect search engines to be present - they have become a default convention **(Rosenfeld et al, 2015, p. 215)**. Last, but not least, another scenario where a search engine will prove valuable is in cases where there are large amounts of dynamic content that cannot be cataloged fast enough manually, such as story files in an online newspaper.

Metadata, controlled vocabularies, and thesauri

According to Wikipedia as cited by **Rosenfeld et al (2015, p. 270)**, metadata or metacontent is "defined as the data providing information about one or more aspects of the data", such as the data's means of creation, purpose, time and date of creation, who the author or creator of the data is, the location on a computer network where it was created, and the standards that were used. Metadata tags are used to describe different types of content for the purpose of better retrieval and navigation of data. For instance, the metadata of a text document could contain information about its length, creation date, author, as well as a short summary.

Controlled vocabularies are, in their simplest, synonym rings (lists of equivalent terms) or authority files (lists of preferred terms). There are several types of controlled vocabularies, as seen in Figure 02 below.

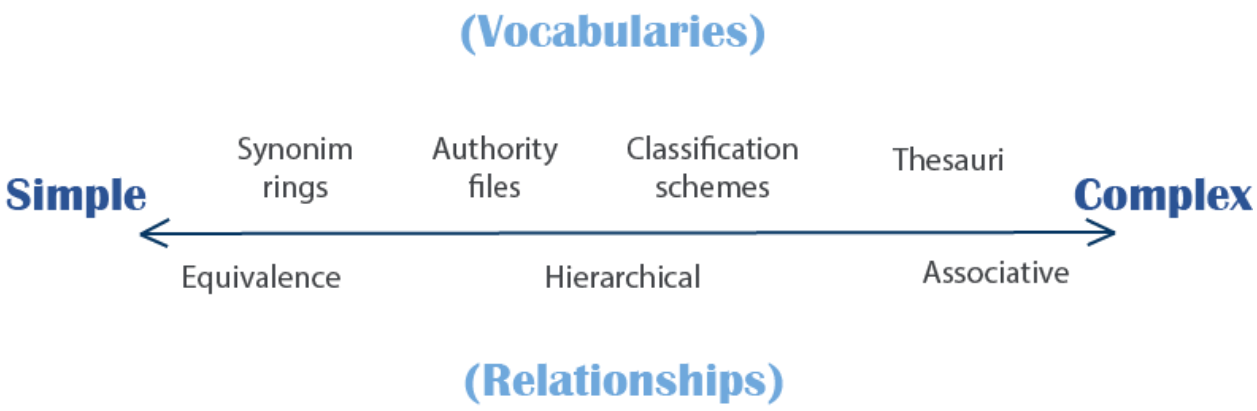

Figure 02 - An overview of controlled vocabularies. Copied from Rosenfeld et al, 2015, p 271.

While synonym rings are called as such, they often feature words that are not true synonyms (e.g. *itouch* and *ipod touch*, or *blender, kitchen aid*, and *food processor*). While their goal is to widen the scope of searchable words that produce a result similar to the desired one, they also come with their limitations **(Rosenfeld et al, 2015, p. 273)**. For example, using synonym rings

can result in lowered precision in returning relevant results. This, however, is compensated by increasing recall, or retrieving all relevant documents in a system.

The second type of controlled vocabulary are authority files, or lists of accepted values and preferred terms **(Rosenfeld et al, 2015, p. 275)**. While they do not use synonyms or variants of a word in theory, in reality they are often a synonym ring where one of the terms has been defined as the acceptable value or preferred term.

Classification schemes, or taxonomies, refer to an arrangement of preferred terms **(Rosenfeld et al, 2015, p. 279)**. They can have several purposes, such as a visible browsable hierarchy that is part of the user interface (UI), or an organizing and tagging tool for documents that is used by indexers and authors in the backend.

The last type of controlled vocabulary, thesauri, has a different function and form than the one typically used in situations like the one I am in right now while writing this thesis. When relating to information environments, a thesaurus is "a semantic network of concepts, connecting words to their synonyms, homonyms, antonyms, broader and narrower terms, and related terms" **(Rosenfeld et al, 2015, p. 282)**. In other terms, thesauri are controlled vocabularies in which hierarchy, association, and equivalence are identified in order to improve information retrieval.

User experience (UX)

User experience, or UX as I will refer to it throughout the thesis, can be broadly summarized as the effects that a user feels after interacting with a device, system, or product **(Hartson and Pyla, 2012, p. 5)**. These effects include the influence of usefulness, usability, and emotional impact. As for interaction itself, it is a broad term that describes the user seeing, touching, or thinking of the product or system - even before there was physical interaction with the product or system of any kind. UX does not deal with what is inside of a product or how it works on a technical level (e.g. how it is programmed), but with the outside, which is what users see **(Garrett, 2011, p. 6)**.

In the previous paragraph, I mentioned usefulness, usability and emotional impact. They, along with functionality, are important facets of UX. According to **Hartson and Pyla (2012, p. 6)**, usefulness is the UX component that is given the ability to use the product or system by the said system's functionality in order to fulfill its intended purpose. Usability, on the other hand, is the pragmatic side of UX **(Hartson and Pyla, 2012, p. 6)**, which encompasses efficiency, effectiveness, ease-of-use, productivity, retainability and learnability, as well as the practical facets of user satisfaction. The third facet, functionality, is the ability to fulfill a product or a system's intended purpose via its computational features and capabilities that are not part of the user interface **(Hartson and Pyla, 2012, p. 6)**. The final one, emotional impact, is the side of UX that influences the users' feelings and emotions. It consists of effects like fun, pleasure, joy of use, desirability, aesthetics, originality, novelty, coolness, sensations, engagement and appeal, although it can also reach deeper emotional levels like self-expression, identity, pride of ownership and feeling of contribution to the world **(Hartson and Pyla, 2012, p. 6)**.

Five planes of user experience

User experience has several layers, which are also called the five planes of UX **(Garrett, 2011, p. 19)** - Strategy, Scope, Structure, Skeleton, and Surface. The planes provide a conceptual framework that can be used to discuss UX problems and possible solutions for them.

The framework is built from the bottom to the top **(Garrett, 2011, p. 19)** which means that the Strategy plane is always at the bottom, or the groundwork for the UX design process, while the Surface plane (which is concerned solely with the appearance of the product) is always at the top, as shown on Figure 03.

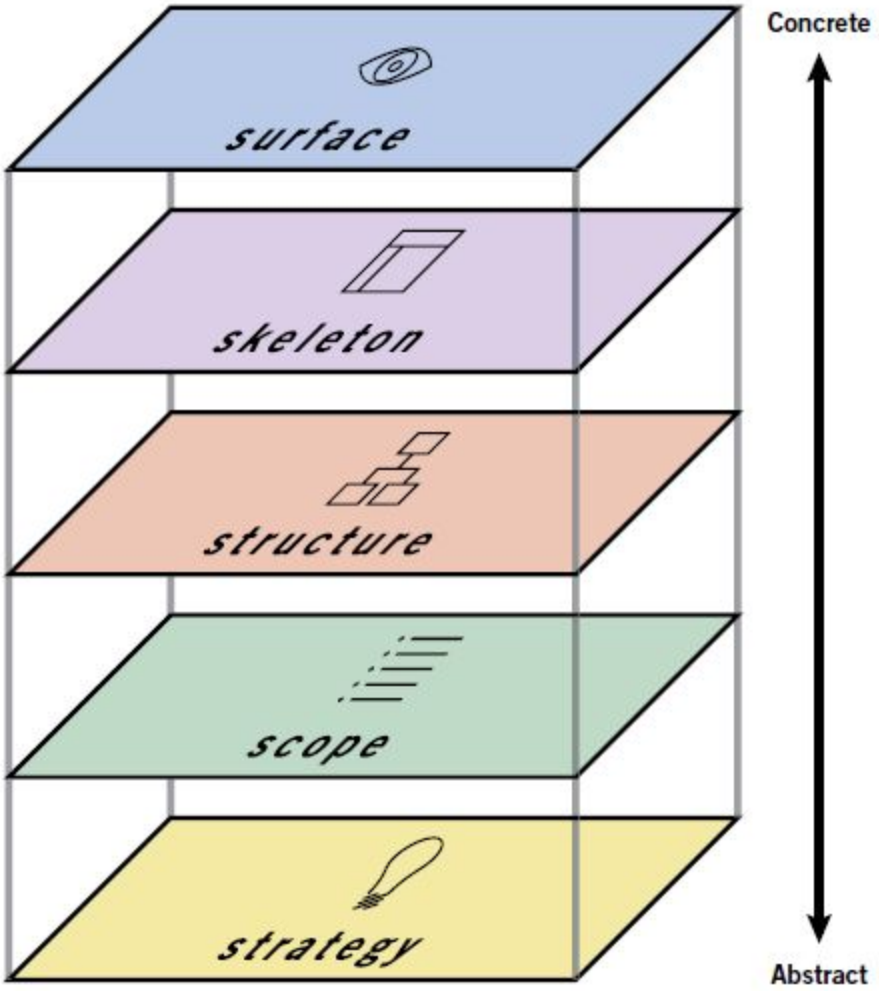

Figure 03 - An overview of the five planes of user experience. Taken from Garret, 2011, p. 22.

Each of the five planes depends on the ones below it. When the development team's choices in a specific plane do not align with those below (or above) it, the project begins to go astray, deadlines get missed, and costs start to rise while the team attempts to bring together elements that are not a good fit and make them work together. To make things worse, if a product does get launched in such cases, the users often do not like it, as it does not give them a good user experience. See figures 04 and 05 for an illustration of this phenomenon.

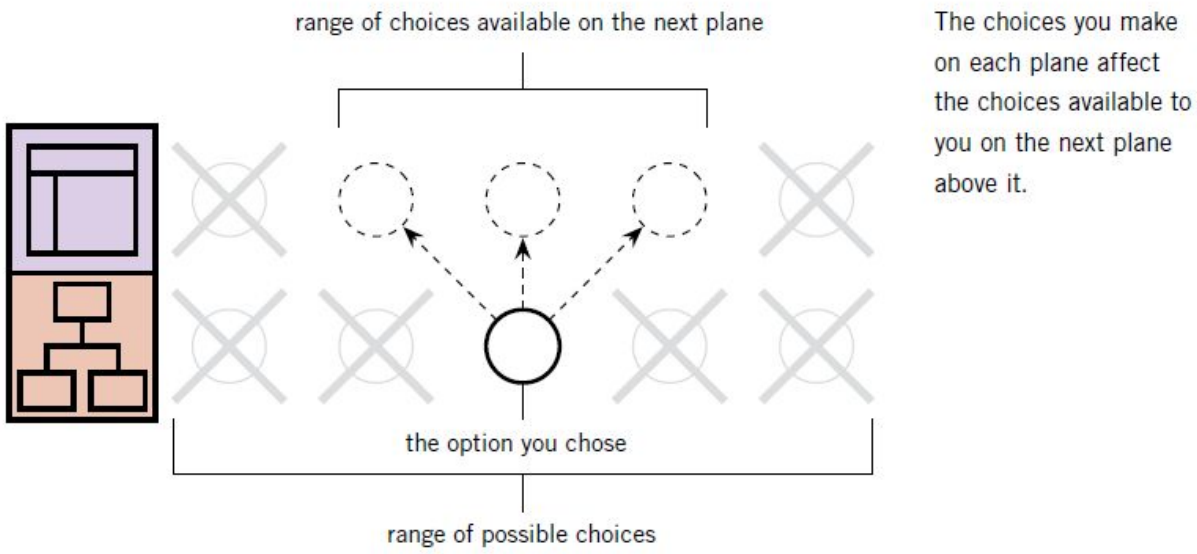

Figure 04 - The range of choices available when going through the five planes. Taken from Garret, 2011, p. 22.

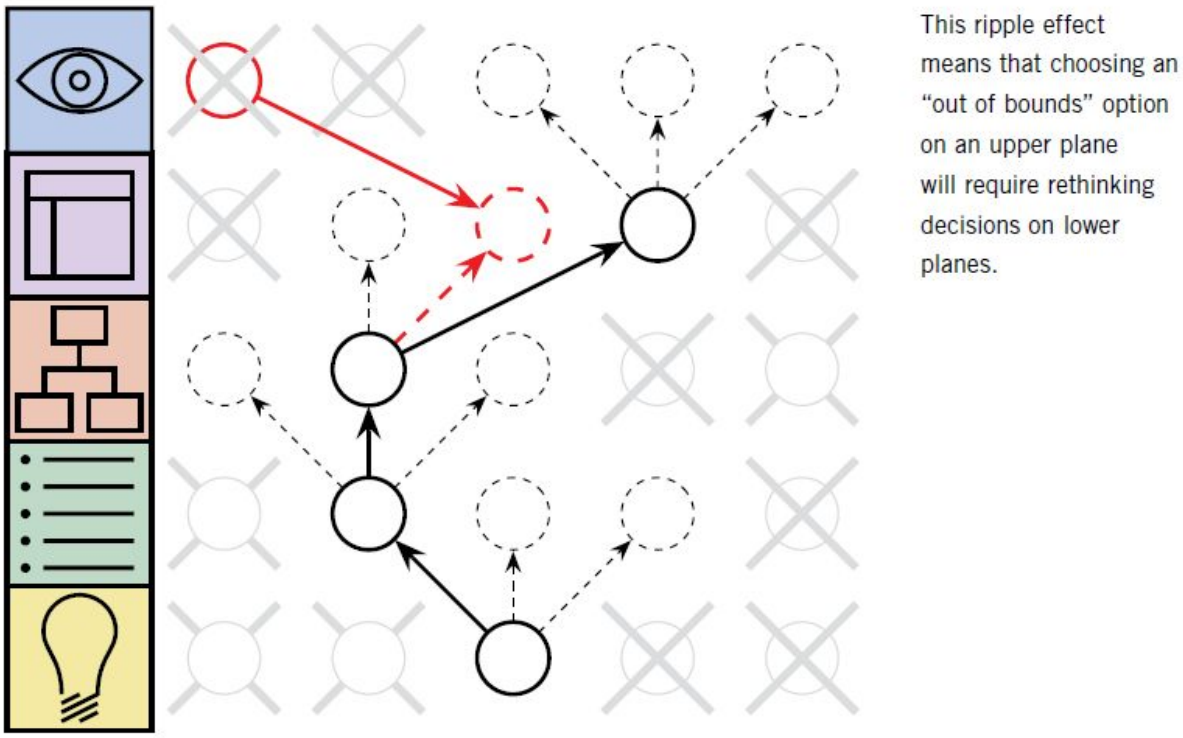

Figure 05 - Explaining how some choices may be invalid and warrant a reexamination of the choices from prior planes. Taken from Garret, 2011, p. 22.

As seen in Figure 04, it is not necessarily the case that all decisions for a plane that sits lower on the framework must be made before the one above it can be addressed. Sometimes, decisions made on the upper planes will force the development team to reevaluate their earlier decisions - dependencies go both ways **(Garrett, 2011, p. 24)**. Decisions made at each plane depend on the industry's best practices, current knowledge about the users' needs and what they are doing, and competitor analysis. Optimally, a project should be planned in such a way that work on a higher plane cannot be finished unless work on the planes below it is complete - and not cementing decisions on lower planes before proceeding to the ones on top.

Definitions of the five planes

The strategy of a website is determined by the Strategy Plane **(Garrett, 2011, p. 21)**. The primary focus is what the users need from a website, but also what the product team wants to get out of it. Some objectives might be clear depending on the purpose of a website (e.g. users would want to buy products and the product team would want to sell them on a website like Amazon), but others might not be so easy to express (e.g. what role does user-generated content play in the business model).

The second-lowest plane, Scope, deals with the kinds of functions and features of a website **(Garrett, 2011, p. 21)** and how they work together. The scope of a website itself is comprised of what its features and functions are.

The Structure Plane is the middle one of the five **(Garrett, 2011, p. 20)**. It defines how users get to certain pages of a website and what they can do once they have completed their task on those pages. It also sets the stage for information architecture on a website, since that is where the categorization of information begins to take shape.

The second-highest plane, Skeleton, defines the placement of different elements of the interface on each page **(Garrett, 2011, p. 20)**. Such elements could be blocks of text, buttons, images, controls, and so on - the placement of all of them optimized to be as efficient and effective as possible.

The Surface Plane is the highest of the five **(Garrett, 2011, p. 20)**. It consists of all the visual elements of a webpage that the users can see - images, text, clickable elements, etc.

The basic duality - product as functionality vs product as information

While the definitions of the five planes are straightforward enough, depending on the person, what is done on each plane might be different. In other cases, two people might know the same element under different terms - for example, “information architecture” and “information design”, or “interaction design” and “interface design” **(Garrett, 2011, p. 25)**.

In order to address these discrepancies, the five planes can be split down the middle, so they each have two sides - functionality and information (See Figure 06).

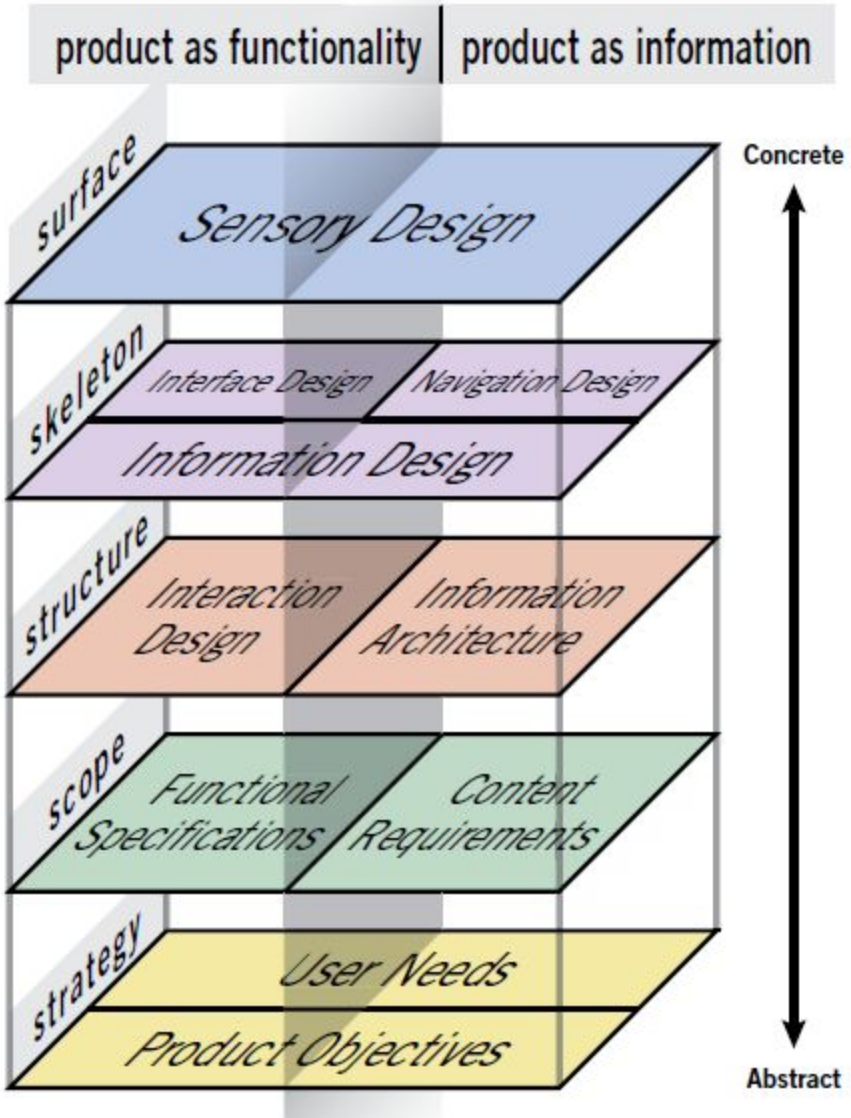

Figure 06 - An expanded view of the five planes of user experience, where the functionality and information aspects are visible. Taken from Garret, 2011, p. 29.

The functionality side of each plane is concerned with what tasks people need to complete and their train of thought while doing so. The product is seen as a tool (or set of tools) for the user to complete specific tasks. On the other hand, the information side of the planes deals with

what the product can offer to the users, and what that would mean to them, thus creating a user experience that is as rich as possible. That allows users to find, process, and have a better understanding of the information provided.

In the paragraphs below I will discuss how each plane can be broken down to its component elements, which will allow for a closer inspection of how all pieces fit together during the user experience design process.

First, the Strategy Plane has two “building blocks” - user needs and product objectives **(Garrett, 2011, p. 28)**. They can both be seen from a functional and informative perspective, as seen in Figure 05 above. The user needs are the goals that have an external influence on a product, while the product objectives are of internal influence and can be business goals or other kinds.

The Scope Plane is split vertically depending on whether we look at it from the perspective of functionality or information **(Garrett, 2011, p. 29)**. On the functional side of things, we would be concerned with the functional specifications of the product, or its set of features. In terms of the product as information, a description of the different forms of content (or the content requirements) will be necessary.

The Structure Plane has two parts as well **(Garrett, 2011, p. 30)**. On the functionality side, it is concerned with how the system responds to the users (interaction design), while on the other side, its focus is on information architecture.

The Skeleton Plane, unlike the others, is broken down into three components **(Garrett, 2011, p. 30)**. The presentation of information in a way that makes it easier to understand, or the product’s information design, is in the focus of both sides of the plane. At the same time, the interface design is part of the functionality side, while the information side is expressed through the navigation design.

Finally, the Surface Plane **(Garrett, 2011, p. 30)** is thoroughly focused on the sensory experience facilitated by the finished product.

A UX lifecycle process template

Definition and terminology

A design process typically consists of four elemental UX activities - Analyze, Design, Implement, and Evaluate **(Hartson and Pyla, 2012, p. 53)**. They are also referred to as analysis, design, implementation, and evaluation, and are applicable regardless of the type of product that is being designed (see Figure 07).

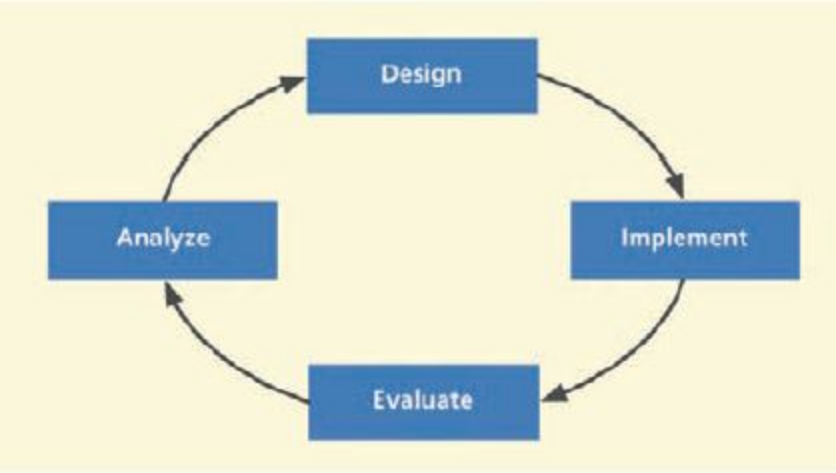

Figure 07 - The UX lifecycle process. Taken from Hartson and Pyla, 2012, p. 53.

However, there are some specifics to the activities when it comes to UX design processes. The Design activity refers to designing concepts, as well as interaction behavior, feel, and look. Prototypes are created during the Implement activity (limited to the interaction design, without actual programming), and testing for the users' needs is carried out during the Evaluate activity. Finally, the results from the testing are analyzed and interpreted during the Analyze activity. Depending on the discoveries and progress during each activity, iterations of the same activity or moving back to the previous one may take place (see Figure 08).

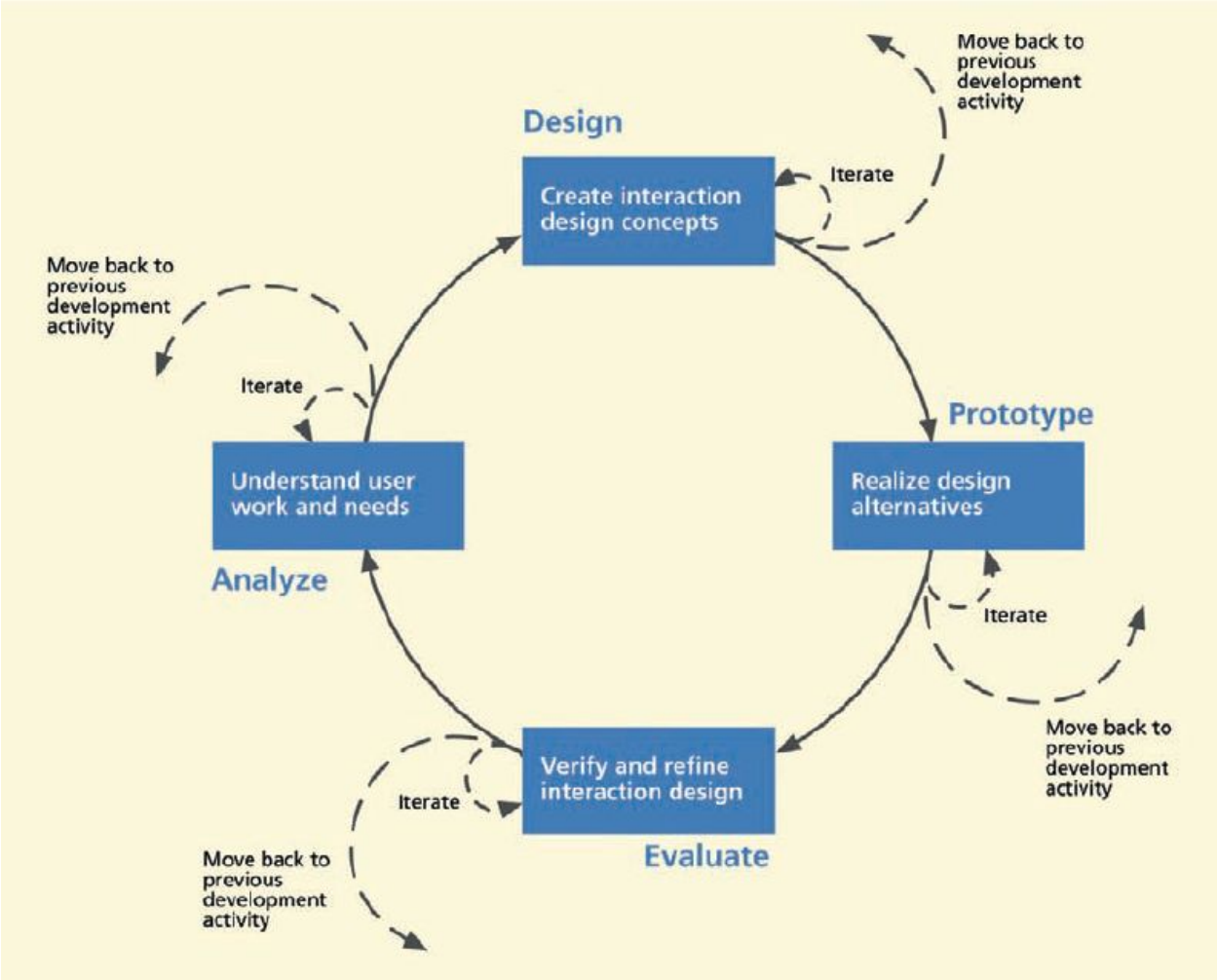

Figure 08 - The expanded UX lifecycle template, where the possible iterations and summaries of the four activities. Taken from Hartson and Pyla, 2012, p. 54.

There are some terms that need to be defined in order to make sense of the model and the UX process activities fully **(Hartson and Pyla, 2012, p. 55)**. Each activity can have *sub-activities* (such as contextual analysis during the Analysis activity). On another hand, a *method* is the approach to completing an activity or sub-activity (e.g. lab-based evaluation for the Evaluate activity). Finally, a *technique* is a specific practice applied within a method - for instance, "think-aloud" is a data collection technique that can be applied within the lab-based evaluation method.

UX process activities

Each activity can consist of several sub-activities, depending on the situation and findings.

During the analysis process, possible sub-activities could be, for example, contextual inquiry and contextual analysis **(Hartson and Pyla, 2012, p. 55)**. By utilizing these two sub-activities,

the development team can deduce what the user needs are, in order to create a new system design. Another possible sub-activity is the extraction of requirements from contextual data, which are technically interaction design requirements - they will help determine the features, look, and feel of the product. One more example of an analysis sub-activity is synthesizing design-informing models (different extensions of the design space and work activity). They include models that describe how different work roles interact and how work is done, different artifacts that are created, and so on. The goal for the sub-activities and the analysis activity itself is to understand the business domain, as well as the user needs and how they work.

The design process is not only limited to designing a product from the ground up, but also includes redesigns and changes for future versions **(Hartson and Pyla, 2012, p. 56)**. Sub-activities that can take place during it are, for example, design ideation and sketching, which are expressed through brainstorming, design thinking, and sketching new ideas. Design ideation itself can lead to conceptual design, the creation of storyboards, and exploring different product design ideas through mockups of varying fidelity. Design production, another design process sub-activity, consists of the application of requirements and design-informing models in order to create and shape interaction design. It entails in the prototyping and iterating of the conceptual design.

The implementation, or prototype activity, is often conducted together with the design activity. Depending on their different purposes, there can be several types of prototypes **(Hartson and Pyla, 2012, p. 56)**. Horizontal prototypes incorporate many features but have little depth, while vertical prototypes are the opposite - a narrow array of features, but great depth in functionality. There is also a type that is a mix of the two - the “T” prototype, which is mostly shallow but has a few parts that are developed in depth. When a specific interaction needs to be examined for possible alternatives, a local prototype can also be utilized. Those represent a small area of inspection where the horizontal and vertical types come together. Besides in different types, prototypes also come in different fidelities - from low fidelity (e.g. paper prototypes), through medium to high fidelity (functional prototypes).

Finally, during the evaluation process, rapid or rigorous UX evaluation methods can be utilized in order to see whether the usability and business goals of the product have been reached **(Hartson and Pyla, 2012, p. 56)**. This is ensured via UX targets and metrics.

Lean UX

Definition

Besides “classic” UX, I am also going to talk about one of its variations - namely, Lean UX. It is a streamlined version of classic UX that is deeply rooted in collaboration and cross-functionality **(Gothelf, 2013, p. 4)** and puts emphasis on cooperation between UX designers and the remainder of the product team. The main goals of Lean UX are optimizing the UX design process by stripping away heavy deliverables and talks about documents and features, and focusing on the production of minimum viable products (MVPs) instead.

Foundations

Lean UX is based on three foundations - design thinking, agile software development, and the Lean Startup method.

Design Thinking

Design thinking is a mindset where the primary foci are the concept of the product and the design for the user experience and emotional impact **(Hartson and Pyla, 2012, p. 259)**. It is an approach in creating products that elicit an emotionally impactful, aesthetically pleasing user experience that is based on value- and social-oriented interaction.

In its essence, design thinking is focused on the design process, relies on many different sources of inspiration and ideas in order to solve design problems, and requires a deep understanding of the users' needs due to its human-centered nature. Besides that, it is also market-oriented, since a good grasp on competitors and their products, as well as trends in technology and its usage are crucial as well - it provides attention to many details, such as the product's presentation, packaging, and customer support.

As for its relation to Lean UX, design thinking is important, since it strongly emphasizes on the fact that all aspects of a business can be tackled with design methods **(Gothelf and Seiden, 2013, p. 6)**. It encourages designers and non-designers alike to think outside the box and use design methods to solve their tasks, and collaborate across their roles. It also provides a holistic perspective.

Agile Software Development

Since I will discuss agile software development in more detail in the "Methods and Frameworks" subchapter further in the report, here I will focus on the core principles in Agile methodology that are applied to product design in Lean UX instead.

The first principle, "Individuals and interactions over processes and tools" **(Gothelf and Seiden, 2013, p. 6)** claims that the entire team must be engaged in the development process, and that ideas must be often discussed and exchanged freely among everyone regardless of their role. Collaboration between teammates should take precedence over the constraints of the methods used in the process, or over the use of specific tools at any cost.

"Working software over comprehensive documentation", the second principle, makes us focus on finding which is the most viable solution to a problem, regardless of who suggested it and what their role on the project team is **(Gothelf and Seiden, 2013, p. 6)**. When working software is built faster, it can be released on the market and assessed "in the wild" so the team can see how viable the solution is and to what extent it fits the market.

The third principle is "Customer collaboration over contract negotiation" **(Gothelf and Seiden, 2013, p. 6)**. Faster iterations, a larger extent of involvement, and a reduced dependency on

heavy, detailed documentation come as a result of the collaboration between users and all team members throughout the development process. Detailed and specification-heavy documentation especially becomes less necessary, as the whole team participates in making all decisions that normally demand defending choices and written communication.

The final principle, "Responding to change over following a plan" **(Gothelf and Seiden, 2013, p. 6)** relies on the assumption that initial designs will be wrong by default. Therefore, it is of great importance to find out where they fall short, and afterward tweak and test them once more. That allows for a constant betterment of a product, which gets "more right" with each iteration.

Lean Startup

The third foundation of Lean UX is the Lean Startup philosophy and method, which was created by Eric Ries **(Ries, 2011, p. 77)**. According to Ries, as cited by **Gothelf and Seiden (2013, p. 7)**, it:

> *"... advocates the creation of rapid prototypes designed to test market assumptions and uses customer feedback to evolve them much faster than more traditional software engineering practices... Lean Startup processes reduce waste by increasing the frequency of contact with real customers, therefore testing and avoiding incorrect market assumptions as early as possible."*

That philosophy of Lean Startup is also utilized in Lean UX, but unlike the Startup version, Lean UX applies it to product design, not startup formation.

Like Agile methods and Lean UX itself, Lean Startup is also based on several core principles. While they all play a role in Lean UX to an extent, two of them are of the greatest importance - validated learning and the Build-Measure-Learn feedback loop **(Ries, 2011, p. 7)**.

Validated learning, the first of the two principles, focuses on the building of a sustainable business (or, in the case of Lean UX, product) that can be scientifically validated through experimenting (testing) often, thus allowing the entrepreneurs (product team) to evaluate all elements of their vision or product. As for the Build-Measure-Learn feedback loop, its fundamental idea is that startups grow ideas into products, then measure and analyze how customers (users) respond to the product, and based on the information obtained, decide whether to continue with their current approach or adjust and try something new.

Lean UX Principles

Speaking of principles, Lean UX has a list of them of its own. I will discuss them in this section of the thesis, as well as why it is a good idea to observe them throughout the development process.

The first principle is "Cross-functional teams" **(Gothelf and Seiden, 2013, p. 7)**. Teams like these include members with various fields of knowledge and competence, and they are expected to collaborate continuously throughout all stages of the development process. If

followed, this principle brings greater team efficiency, since conversation between team members is expected and encouraged at any given time during the project, and collaboration is not time-gated by stage, as in some development methods like Waterfall **(Gothelf and Seiden, 2013, p. 8)**.

The second principle, "Small, dedicated, collocated" means that a team should ideally be kept small, with no more than ten core members **(Gothelf and Seiden, 2013, p. 8)**. They should all be at the same location and be dedicated to one project at a time. The reasoning behind this principle is that smaller teams are easier to establish teamwork and a sense of camaraderie between members, as well as that they have better communication and focus. Keeping all team members on the same project at the same location allows them to grow relationships with their team, which has the same priorities at all times.

"Progress = outcomes, not output" is the third principle of Lean UX. According to **Gothelf and Seiden (2013, p. 8)**, *"Features and services are outputs. The business goals they are meant to achieve are outcomes. Lean UX measures progress in terms of explicitly defined business outcomes."* The logic behind this principle is that there is no way to have meaningful insight into whether a service or feature will prove to be effective unless it has been released on the market. Instead, by managing the progress towards outcomes and the outcomes themselves, the team gains a better understanding of the features they are working on. In case a feature is not performing as well as expected, the team can then decide whether to keep it, alter it, discard it, or replace it.

The fourth principle, "Problem-focused teams", says that teams should be assigned problems to solve, and not sets of features or services to implement **(Gothelf and Seiden, 2013, p. 8)**. This way, the product owner shows trust in the team, by allowing them to offer their own solutions. This, in turn, helps instill in them a sense of ownership of the solutions they implement, as well as a sense of pride in their accomplishments.

The fifth principle is one that is perhaps the core of Lean UX, namely "Removing waste" **(Gothelf and Seiden, 2013, p. 9)**. Any artifact or action during the development process that does not contribute to the end goal (improved outcomes) must be removed from the process. Project teams have limited resources and the more meaningful additions they can make to the products without unnecessary detours and distractions, the faster they can work. Regular and disciplined waste removal can help them with keeping their focus where it should be, thus increasing their efficiency.

"Small batch size" is the sixth principle, and another one of great importance to Lean UX **(Gothelf and Seiden, 2013, p. 9)**. The goal of this principle is to keep the inventory or product backlog low, while the quality of deliverables remains high. In Lean UX, this means focusing only on the creation of deliverables that will help the team progress and avoiding big lists of "to-do" design ideas. A good reason to abide by this principle is that large batch design makes the product team less efficient by forcing them to wait for big deliverables; it also keeps them from seeing whether the ideas implemented in the deliverable are valid. Some team members

may go idle, which in turn leads to design assets that do not get used. Delivering in large batches is a wasteful approach that does not allow product teams to learn as much as they could.

The seventh principle, “Continuous discovery” focuses on constantly engaging the customer throughout the design and development of the product **(Gothelf and Seiden, 2013, p. 9)**. This is expressed via the use of qualitative and quantitative methods during regular meetings and activities. The entire team should be involved in regular research, which tells them what the users are doing with the product and why. Applying “Continuous discovery” allows the team to test and validate their ideas with the users frequently, as well as to build a sense of empathy for the users and their problems and needs. Also, as the whole team learns together on the go, the need for documentation and debriefing is reduced.

“GOOB”, or “Get out of the building”, is an acronym coined by Stanford professor Steve Blank, and is the eighth principle of Lean UX **(Gothelf and Seiden, 2013, p. 9)**. The core of this idea is that debates and speculation about the users’ needs without involving the actual users are inefficient and that users should be involved in the product design as soon as possible. It is a smarter approach to test extensively before the team commits a lot of resources and time to an idea that the users are not interested in. Ultimately, the success of a product is determined by the customers and users, not the product team. Giving them an opportunity to voice their preferences and needs as early as possible can be a good indicator of whether a product is ready to be released or not.

The ninth principle is “Shared understanding” **(Gothelf and Seiden, 2013, p. 10)**. This refers to the collective knowledge that the team possesses of its process, product, and customers. The principle is core to Lean UX - the more shared knowledge there is among the team, the lesser the need to depend on documentation and secondhand reports is.

The tenth principle of Lean UX is one of my favorites about the philosophy, and the one I referred to in the Problem Description earlier. Namely, “Anti-pattern: Rockstars, Gurus, and Ninjas” **(Gothelf and Seiden, 2013, p. 10)**. The problem with loud, brash-sounding roles like Rockstar, Guru, and Ninja is that people who are labeled as such usually do not like to share their ideas or the spotlight, because they are THE expert in their field, period. As a result, they break the cohesion of the team and inhibit effective collaboration. This, in turn, damages the shared understanding that allows the team to avoid repeating the same mistakes and slows their progress. On a more personal note, if I see a job announcement looking for a Code Ninja or Design Guru, that is an immediate red flag to me about the way the team works in that company and how unnecessarily cumbersome their work process is likely to be.

“Externalizing your work” is the eleventh principle of Lean UX **(Gothelf and Seiden, 2013, p. 10)**. In Lean UX, work is done in teams, and everyone has access to all ideas and participates in all discussions. Therefore, ideas are put out in the open in the shape of sticky notes, whiteboards, printouts, and such. This allows the team to see where everyone stands at the moment and how far along the process is; it also creates an ambient, passive flow of

information that constantly inspires new ideas based on the ones already on display. It also allows everyone on the team to participate in information sharing, even if they are more quiet or shy.

The twelfth principle is “Making over analysis” **(Gothelf and Seiden, 2013, p. 11)**. The focus of this principle is that it is better to create the initial version of an idea, instead of spending time on debating whether it is any good or not. As the GOOB principle stated, the validity of an idea can be evaluated the best by customers and users out “in the wild” and not by assuming and discussing in a meeting room. Therefore, instead of going over potential scenarios, it is better to create something and, well, get out of the building.

It is difficult to grow a business around an idea and see whether that idea is the right one at the same time. The thirteenth principle of Lean UX, “Learning over growth” deals with that exact problem **(Gothelf and Seiden, 2013, p. 11)**. Lean UX favors learning first and scaling (growing) an idea second. That is due to the fact that scaling an idea that is unproven and untested is risky - it is just as likely to work as it is to not work. In the case where it turns out that the idea is not viable, the team expends time and resources that could be used for something else that would prove to be a better fit. Therefore, it is advisable to test whether an idea is the right one before scaling it out to all users.

According to the fourteenth principle, “Permission to fail” **(Gothelf and Seiden, 2013, p. 11)**, product teams should be allowed to make mistakes in order to learn and progress. This requires a safe environment in which to experiment and fail - both in a technical sense which promotes idea sharing and in a cultural sense, where team members will not be penalized for pushing unsuccessful ideas. The reasoning behind this principle is that by allowing a project team to fail sometimes, they are given an opportunity to learn from their mistakes and grow. Bigger tasks become less intimidating when there is no fear of failure, which, in turn, makes team members more prone to taking risks that can generate even bigger, better ideas. Another positive side effect is that failures allow for a better mastery of one’s skills.

Finally, the last and fifteenth principle, “Getting out of the deliverables business” is very similar in its essence to the third one, “Progress = outcomes, not input” **(Gothelf and Seiden, 2013, p. 12)**. Lean UX changes the focus of the design process from documentation and what deliverables are being created to what is the desired outcome and how it can be achieved. The team’s priority should be understanding which features and services have the biggest impact on the users. How exactly they achieve that is irrelevant to the final result, as what matters, in the end, is the quality of the released product - which is measurable by the users’ reaction to it.

The Lean UX lifecycle

The core tool for outcome-focused methodologies such as Lean UX is a hypothesis statement **(Gothelf and Seiden, 2013, p. 17)**. It is a way to describe assumptions and record them in a testable form, and is composed of several elements.

Assumptions, the first element, are declarations at a high level of what is believed to be true **(Gothelf and Seiden, 2013, p. 18)**. *Hypotheses* are their more granular version that are aimed at specific areas of the product or its workflow. *Outcomes* are the signal from the market that aids the team in approving or disapproving their hypotheses, and can be of qualitative or quantitative nature. *Personas* are fictional models of the users, who are believed to be solving a problem. Finally, *features* are the improvements or changes in the product that will help the team achieve the desired outcome.

Bearing that in mind, we can now look at the Lean UX process, as seen on Figure 09.

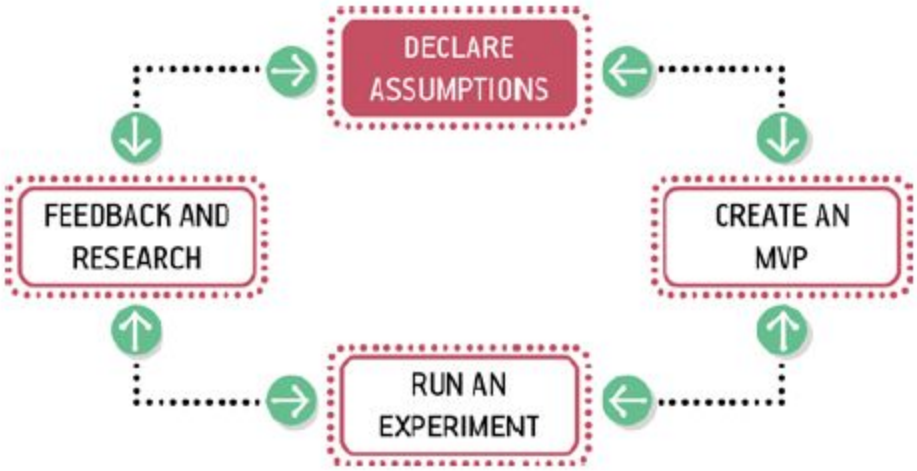

Figure 09 - The Lean UX process. The first step is highlighted. Taken from Gothelf and Seiden, 2013, p. 18.

The first step is to announce the assumptions that the team will base the start of the project on. This allows them to have a common starting point that can be discussed by everyone regardless of their skills or role. After defining some assumptions, they are prioritized based on the level of risk and to what extent the team has an understanding of the issues expressed by them. Afterwards, the assumptions are tested by converting them into hypotheses, and further breaking them down into sub-hypotheses **(Gothelf and Seiden, 2013, p. 22)**. Other artifacts for this step are creating a list of outcomes, personas representing the target audience, and a list of features that the team believes will help the users. They are all used to finalize the sub-hypotheses used in the first designs, which end up looking like this:

We will [create this feature] for [this persona] in order to achieve [this outcome].

The team then focuses on designing mockups that reflect the hypotheses they created, which concludes the first step of the process.

The second step is to build a minimum viable product, or MVP. MVP-s are the smallest possible creations that help to test a hypothesis **(Gothelf and Seiden, 2013, p. 56)**. They can be used in two ways - teams use them to either learn something about the market, or to create a small

version of a feature or product in order to create value for the market as fast as possible. MVP-s can be created via prototypes of varying fidelity, with each having different advantages and disadvantages. After an MVP is created, it is time to test it in the third step.

Running tests or experiments is the third step of the Lean UX process. They do not necessarily need to cover the entire product; it is sufficient to prototype only the most impactful parts of the product for the customer. The key is to focus on the primary workflows of the MVP, so the project team can focus on specific sections of it and evaluate how effective it is, and to what extent is it valid **(Gothelf and Seiden, 2013, p. 66)**.

The last step of the process is collecting feedback through research **(Gothelf and Seiden, 2013, p. 74)**. This is done via collaborative and continuous discovery - research activities are distributed throughout the entire team, and take place every sprint, instead of being concentrated only at one point during the project.

Research methodology

In this subchapter, I will discuss the three approaches to research, as well as their components and the criteria for selecting a research approach.

Research approaches

There are three approaches to research that I will discuss in this thesis - qualitative, quantitative, and mixed **(Creswell and Creswell, 2018, p. 3)**. While qualitative and quantitative research might seem like set categories that are mutually exclusive, in reality they are more of two opposite ends of a spectrum, with mixed research in the middle.

Qualitative research is an approach for investigating and having a better understanding of the way groups or individuals perceive a social or human problem. Researchers who use this method typically focus on the meaning that individuals ascribe to different problems, as well as reporting how complex a situation is **(Creswell and Creswell, 2018, p. 3)**. Qualitative researchers look at their process and findings from an inductive perspective (they infer general laws from particular instances).

Quantitative research, on the other hand, tests theories through the examination of relationships between variables, and typically utilizes numerical data **(Creswell and Creswell, 2018, p. 3)**. The numerical data is then measured and analyzed by using statistical procedures. Quantitative and qualitative researchers alike are cautious of testing ideas deductively (having a theory or hypothesis and then testing whether it is true through observations).

Mixed methods research, as its name implies, is an approach that combines both qualitative and quantitative characteristics **(Creswell and Creswell, 2018, p. 3)**. The base of this method is that when researchers combine qualitative and quantitative data, they may acquire additional insight beyond what the two types of data can provide on their own.

Components in a research approach

There are three components to a research approach - a philosophical worldview, and research designs and methods specific to each approach.

Philosophical worldviews

Philosophical worldviews in research can also be referred to as *paradigms*, *epistemologies and ontologies*, or *broadly conceived research methodologies* depending on the source. In Creswell and Creswell's case, they have chosen the term *worldview*, or "*a basic set of beliefs that guide action*" **(Creswell and Creswell, 2018, p. 5)**. A researcher develops their worldview based on varied factors, such as the research communities they belong to, who is their mentor or advisor, past research experiences, or their discipline orientations. I will discuss three such views: postpositivist, constructivist, and pragmatic.

The postpositivist worldview is sometimes referred to as the scientific method, or conducting science research. Other names are postpositivist research, empirical science, or postpositivism. The post- in its name indicates that it came after the positivism worldview, and that unlike it, it recognizes that there is no way researchers can be absolutely positive of the absolute truth of knowledge and their claims when it comes to researching human actions and behavior **(Creswell and Creswell, 2018, p. 6)**. Postpositivist researchers subscribe to a philosophy according to which causes determine outcomes or effects. Their primary focus is to reduce ideas to small samples and measuring the objective reality - therefore, numeric measurement of observations and researching individual behaviors is crucial to postpositivists. They begin their research with a theory, then collect data that supports or disproves it, and finally make adjustments or collect additional data.

Constructivism is another approach to research that is usually associated with qualitative methods. Its foundation is that people want to understand as much as they can about the world in which they live and work, and thus develop subjective meanings of what they experience, usually tied to specific things or objects **(Creswell and Creswell, 2018, p. 7)**. These meanings can vary a lot, , which leads the researchers to investigate the complexity of views instead of narrowing their findings down to just a few ideas or categories. Constructivism, also called social constructivism, relies heavily on the way participants perceive the situation that is being studied. This is accomplished through broad and general questions, often utilized in discussions or interactions with others. Constructivists acknowledge that their own views and backgrounds can influence the way they interpret their findings. Unlike postpositivists, constructivists do not start out with a theory to test, but rather develop one through inductive reasoning.

The pragmatic worldview is not committed to a single philosophical system **(Creswell and Creswell, 2018, p. 10)**. Instead, pragmatists put emphasis on the research problem and research questions without paying heed to specific methods - any approach that can lead to understanding the problem is valid and can be used. To summarize, pragmatic researchers

focus on situations, actions and consequences and not preexisting conditions, unlike postpositivists.

Research designs

While researchers choose an approach to subscribe to for their studies, they must also select a research design that will give specific directions for the procedures in their study. Research designs are also called *strategies of inquiry*.

Qualitative research designs include, but are not limited to: grounded theory (deriving an abstract or general theory of a process, interaction or action from the participants' views), narrative research (recording a chronology of a participant's life), phenomenological research (recording a summary of the experiences of participants who have lived through the same phenomenon), ethnography (studying similar patterns and behaviors actions and language in homogenous cultural groups), and case studies (in-depth analyses of a case that is bound by time and activity) **(Creswell and Creswell, 2018, p. 13)**. Other designs from this group are participatory action research, discourse analysis, and so on.

As for quantitative designs, two of the most common types are surveys and experimental research **(Creswell and Creswell, 2018, p. 11)**. Surveys allow the researchers to observe a numerical depiction of attitudes, trends, and opinions of a group by studying a target sample of that group. It utilizes tools such as structured interviews or questionnaires with the end goal of generalizing the results from the target sample and applying them to the whole group. Experimental research aims to find out whether a specific treatment influences specific outcomes. In such cases, the researcher will give the specific treatment to one group of test participants and not to a second one.

Mixed method research designs converge both qualitative (usually without a predetermined response and open-ended) and quantitative (usually closed-ended, like in questionnaires) data and research **(Creswell and Creswell, 2018, p. 14)**. The reasoning behind using mixed method research is that each of the two base types has its own weaknesses and biases, and when used together, they neutralize each other's issues. Three major designs in this group are convergent mixed methods (qualitative and quantitative data are merged in order to comprehensively analyze a problem), explanatory sequential mixed methods (quantitative research is conducted first, then the results are analyzed and further developed with qualitative research), and exploratory sequential mixed methods (qualitative research is conducted first, then the data is used for a second phase, which is quantitative).

Research methods

The final element of a research approach is the methods that researchers use to collect and analyze data, as well as interpret their findings **(Creswell and Creswell, 2018, p. 16)**.

Researchers who use quantitative methods tend to utilize predetermined responses and instrument-based questions in order to gather data (can be on attitude, performance, census,

observational, etc.) and then use statistical analysis in order to finalize a statistical interpretation of their findings.

Qualitative researchers, on the other hand, use open-ended questions without predetermined responses in order to generate observation-, interview-, audiovisual-, or document data. Then they resort to image and text analysis which result in the interpretation of patterns and themes.

Mixed method researchers use a combination of questions and responses in order to extract multiple forms of data, utilize both text and statistical analysis, and interpret their findings across all databases.

Research Approaches as Worldviews, Designs, and Methods

In this subsection, I will summarize all three approaches to research and their characteristics as worldviews, designs, and methods **(Creswell and Creswell, 2018, p. 17)**.

First, the qualitative approach tends to subscribe to the constructivist philosophy, and utilizes research designs such as grounded theory, case study, phenomenology, and narrative research. The methods typically associated with a qualitative approach are open-ended questions and text or image data. Qualitative researchers usually focus on a single phenomenon or concept, study the participants' setting or context, and bring their personal values to the area of study. They validate the accuracy of their findings, make interpretations of the data, collaborate with the research participants, and utilize text analysis.

A quantitative researcher, on the other hand, is usually a postpositivist that uses experiments and surveys to gather data. The methods they use are close-ended questions, and numeric data. Quantitative researchers identify variables that they study, test and verify explanations and theories, measure and observe information numerically, are unbiased in their approach, and use statistical procedures.

FInally, the mixed method approach tends to be pragmatic in nature, and uses convergent, transformative, and sequential strategies of inquiry. They employ any method available, as long as it brings results that point to resolving the research problem. Mixed method researchers collect both types of data and integrate it into different research designs. They employ practices from both quantitative and qualitative research approaches, and develop their own rationale for mixing them.

Criteria for Selecting a Research Approach

There are several factors that might affect the choice of a research approach, namely the research problem and questions, the researchers' personal experience, as well as the intended audience of their study.

First, some types of approaches are more suitable for specific problems than others (**Creswell and Creswell, 2018, p. 19)**. If the research problem requires a generalization of results or a statistical representation of a study, a quantitative approach may be the best choice. Less

understood or poorly studied situations or phenomena, on the other hand, merit a qualitative approach. This approach is especially useful if a researcher is not yet aware of the key variables that they need to examine, e.g. if a topic is new, or in cases where existing theories cannot be applied with the current sample group. In cases where neither approach is sufficient to understand the research problem or generate enough data for a substantial conclusion, mixed methods have to be employed.

On another hand, it is possible for the researcher's personal experiences and training to influence the choice of approach as well. Researchers are more likely to select whichever approach they are trained in, and have more experience with **(Creswell and Creswell, 2018, p. 20)**. Researchers more comfortable with a quantitative approach tend to prefer a more structured approach with systematic rules. Others that prefer the freedom of a more creative approach might be more inclined to use qualitative methods. Mixed methods fit researchers who enjoy working with both types of data and possess skills in qualitative and quantitative research alike.

Finally, the experience of the intended audience - colleagues, conference attendees, journal readers, etc. - can influence the selection of a research approach **(Creswell and Creswell, 2018, p. 20)**, depending on the experience they have with different approaches.

Users

In this subcategory I will describe what users are, as well as what is their relationship with design.

Definition and misconceptions

Users are the people that interact with a technological product at varying levels **(Massanari, 2010, p. 3)**. Despite commonly being referred to just as "users", the individuals that use products are rarely a homogenous group - in fact, more often than not, they are fragmented and diverse.

Despite the fact that users are a core stakeholder in digital product design, their needs are often misunderstood or overlooked, as those working within user-centered design observe **(Massanari, 2010, p. 1)**. Developers and designers alike have the unfortunate inclination of designing a website based on what they perceive the users' needs to be, or they mistakenly see themselves as part of the target audience. The typical user is much less skilled with technologies, and is unlikely to see a poorly designed website or product as an interesting puzzle to solve. Instead, they just give up and find the information they need elsewhere.

Depending on the extent the users are involved in the design process and how they are perceived by the designers, there are several common tropes in design - systems-centered design, user-centered design, and participatory design.

Users and design

The first trope, systems-centered design, is focused around the belief or expectation that users should conform to the design of a product. Despite that, it is rarely consciously chosen and is usually not stated officially **(Massanari, 2010, p. 4)**. There are two instances where this type of design occurs. In the first, the misconception from the designers and developers that they are part of the intended users leaves them to design a product for their own needs that they see as those of the users. Therefore, the product ends up exhibiting a user interface and functionality that a designer or developer would find useful. In the second case, the product team is pressured into systems-centered design due to deadlines and financial constraints. Another reason might be that systems-centered conventions see users' interactions with software as unpredictable, and the software itself as full of bugs and other issues. Due to that belief, it is assumed that even if a product is well-designed, more and more issues will come to light the more people use it. Therefore, developers might be reluctant to address those issues, especially if they will require a lot of time to fix and will result in a complete rework of the system - even if it does not fit the users' needs, from a developer's perspective, the system still works.

Unlike in systems-centered design, those that utilize user-centered design structure the development process around the users and their needs **(Massanari, 2010, p. 4)**. This trope puts heavy emphasis on user research and usability testing, and advises the use of iterative development cycles where prototypes are constantly improved on after each round of testing them with users. While this approach has the noble aim of designers working as the advocates for users, it does not wholly ensure that the users' needs are met and their pains - resolved. While in user-centered design users are more engaged in the development process, and the approach subscribes to the belief that user involvement improves design, this focus on the users can sometimes be driven from the market perspective rather than a humanitaristic one. There is also the issue that users typically have lower level design skills than the team; they largely do not know how to articulate their needs and talk about what they want to see as features instead. Thus, a designer is responsible for being a "translator" of sorts for the users.

Finally, in participatory design (also called "user-centered innovation"), users are considered to be co-creators throughout the entire design process, instead of having the product designed for them **(Massanari, 2010, p. 6)**. Users are seen as a crucial source of insight and their input is integrated throughout the entire development process instead of just the usability testing phases.

Methods and Frameworks

In the "Methods and Frameworks" subchapter I will cover ideologies and frameworks, mainly agile software development, Growth-Driven Design (GDD), as well as some others for the sake of a better overview and comparison.

Growth-driven Design (GDD)

Growth-driven design (GDD) is a design framework created by Luke Summerfield in conjunction with HubSpot **(Summerfield, 2017, lesson 1)**. Its aim is to streamline the website redesign process and focus on continuous improvement with meaningful artifacts, instead of leaving a website to its own devices after it is pushed live for six months or longer.

The framework is heavily geared towards digital marketers and allowing them maximum freedom and independence from developers when it comes to making changes to the website. This is achieved through the use of the HubSpot CRM, although it may be doable with other content management systems like WordPress depending on the complexity of the website.

GDD has three phases - Strategy, Launch Pad, and Continuous Improvement.

During the first phase, the primary goal is to develop an understanding of the target audience and how the website can help them solve tasks and problems along their user journey. Steps necessary to complete this phase include defining the goals for the website, conducting UX research, identifying the user needs and creating Jobs to be Done (tasks that the users need to complete in order to achieve their goal on the website). The designers then create their fundamental assumptions about their website, the market, and their business. This is also the part of the process when artifacts like user problem statements and value propositions are created. Afterwards, personas are created and used in order to map the user journeys on the website. Based on them and all the previous insights, the website-specific strategy that includes the website information architecture, SEO, technical requirements and so on is created. The last step in this phase is brainstorming an initial wishlist of ideas for the website, and it can include various items, such as sections, pages, types of content, and specific features.

The second phase, Launch Pad's goal, is to create a website that is better in looks and performance than what we currently have, but is not a final product. It should look and behave as a complete website, without any missing functionality or features. The goal is to be able to collect user data, upon which to make further improvements. This is achieved through short, concentrated design sprints on the pages and sections with the highest impact. FInally, the phase is rounded off by facilitating an efficient content creation process, as well as improving the internal efficiencies of the team - e.g. relying on libraries and plugins instead of designing and programming everything from the ground up.

Continuous Improvement, the last phase, focuses on identifying which actions to take based on their impact inferred by the user data. It has four steps - plan, build, learn, and transfer. During the first, the most impactful items for building or optimizing are defined via a website performance roadmap. The roadmap can be used as a "billboard" for setting clear expectations between the team and the stakeholders, as well as a tool to help with the decision of whether a feature should be created from the ground up, optimized, or expanded. The website's goals and the user needs are reassessed quarterly, and new UX research is conducted. Then, during the

build step, the team focuses on the tasks identified and resolves them. The new additions are then released, and based on user interactions with them, new conclusions are drawn during the learn step. The final step, transfer, is about disseminating the knowledge obtained during the previous steps throughout the entire company. This step can be used for feedback, knowledge sharing, and filling the gaps in user research. The cycle generally repeats every two weeks.

Agile and lean software development

In this subsection, I will discuss the Agile philosophy in software development, where it comes short in terms of UX design, how can Agile and UX methods be adjusted in order to work better together, as well as give an example of an Agile method - Scrum. Then I will briefly cover the Lean philosophy and a Lean software development method - Kanban. Finally, I'll draw a comparison between both philosophies and describe a hybrid development method - Scrumban.

Agile philosophy

Definition

Agile software development is a methodology and philosophy that arose as a consequence of the needs of developers who had to adapt to customer requirements that were more varied than usual. Besides that, it is also based on the twelve principles outlined in the *Manifesto for Agile Software Development* that was created at a meeting of 17 software development organizations in Utah, USA in 2001 **(Stoica, Ghilic-Micu, Mircea, and Uscatu, 2016, p. 1).**

The twelve principles are as follows:

According to the first principle (**Cunningham, 2001)**, client satisfaction is ensured through the quick delivery of usable software. The second claims that specifications should be met, even if that occurs late on in the development cycle. The third principle ensures that usable software is delivered often, preferably weekly, while according to the fourth, the main measure for progress is usable software. The fifth principle states that agile methods promote sustained development, and allow the product team and stakeholders to keep working steadily without fluctuations in speed. The sixth principle promotes cooperation between developers and their clients, and the seventh expands on it by clarifying that face-to-face cooperation is the best possible way to communicate. According to the eighth and ninth principles, projects should be as simple as possible and built by credible and motivated developers. The tenth and eleventh principles declare that the best designs and products are built by self-organized teams that embrace change and harness it for the client's benefit. Finally, the twelfth principle advises constant attention, so the development team excels both technically and in design, which in turn allows for greater flexibility.

Development lifecycle

Agile software development focuses on adaptability and flexibility while adopting an iterative approach. Also, unlike traditional software development methods, it puts less emphasis on creating documentation, which, depending on the project or situation, can be a major drawback for agile methods **(Stoica et al, 2016, p. 2)**.

An agile development cycle typically consists of several iterations that last between 2-4 months each, split into sprints of two weeks to a month long. Phases of each iterative cycle include planning, analyzing the requirements, designing a prototype, building it, and then testing, as seen on Figure 08 below.

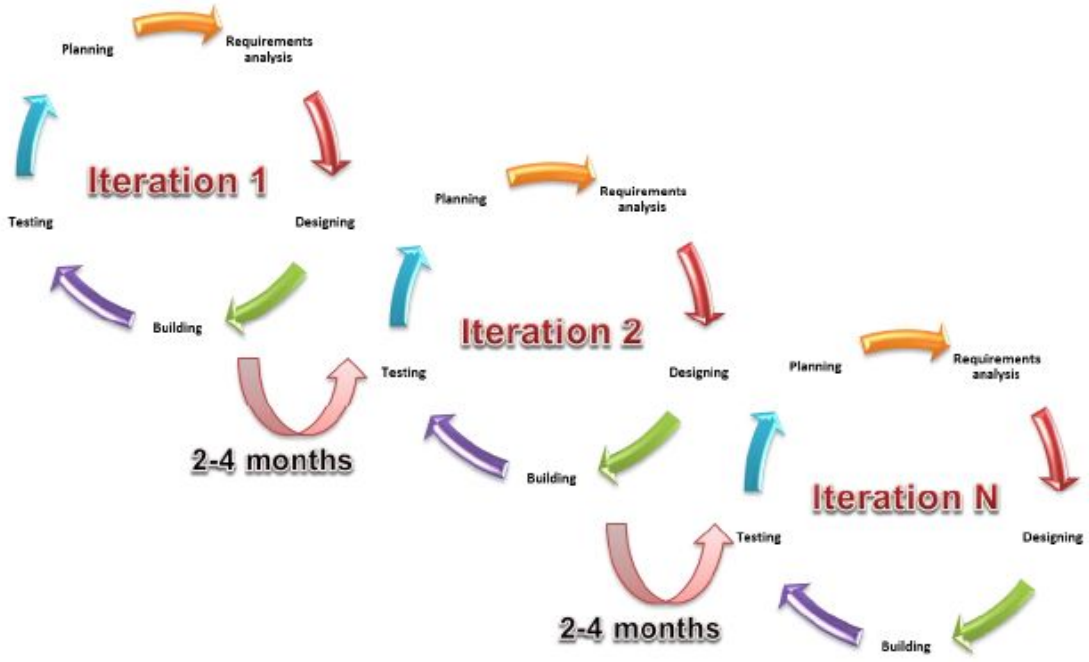

Figure 10 - The agile development lifecycle. Taken from Stoica et al, 2016, p. 3.

Scrum - an Agile method

Scrum is a software development method from the Agile methodology, and it has acclaimed a popularity for itself - according to a report by VersionOne as cited by **Stoica et al (2016, p. 10)**, Scrum holds a market share of 58% compared to other software development methods. It is based on three main pillars - transparency, inspection or verification, and adaptation.

Scrum relies on timeboxed iterations based on predetermined estimations and uses velocity as a metric for improvement of the processes. Iterations are further split into sprints.

Task tracking is done through a Product Backlog, from which items are drawn into Sprint Backlogs for each sprint. After the sprint starts, no new items may be added, and items must be

broken down, so each can be completed within one sprint. Then, after a sprint is concluded, the Sprint Backlog gets reset and new items are chosen for the next. One sprint backlog belongs to one team only.

The Scrum process may involve more than one self-organizing cross-functional team. Teams are involved in specific iterations depending on the members' skillset, and at least three roles need to be present - the Product Owner (who determines the course of development and owns the product), the Scrum Master (who oversees the Scrum process) and the Scrum Team (who develop the product based on the Scrum process).

The drawbacks of Agile methodology from a UX perspective

Despite its popularity, when it comes to a UX perspective, the Agile methodology has some weaknesses.

First, as a methodology created from developers for developers, it puts a lot of emphasis on concerns related to the quality of code, and little to no consideration or even definition for user experience or usability **(Hartson and Pyla, 2012, p. 626)**.

A customer representative on the product team is not required to be a member of the target audience, which can lead to a situation where any produced code is based on assumptions about the users' needs due to the lack of upfront user data. The user interface, the users themselves, or any user activities are not accounted for in the process; the UI then ends up "as-is" depending on what is produced based on assumptions and is then improved based on user feedback.

Other issues include that Agile methods are difficult to scale for large systems or development groups due to communication constraints and tasks getting too large. Similarly, there is no room for design ideation.

How can Agile and UX methods be used in conjunction

Despite the concerns I outlined before, Agile and UX methods and lifecycles can work well together, since they are both iterative **(Hartson and Pyla, 2012, p. 626).** The main difference between them, however, is that Agile is built around code-based deliverables, while UX relies on having as clear as possible understanding of the users and their needs before any working prototype is created. That being said, some adjustments need to be made to both if they are to be used optimally in conjunction.

An Agile method intended for use with UX needs to include early analysis dedicated to the users, as well as their work activities and context. More attention should be given to fleshing out the general concepts of the system, for example through design ideation.

As for UX methods that are to be used together with Agile, they must put an emphasis on collaboration between all team members and preferably house them at the same location. Another requirement is to include effective representatives of the users and the customer.

Finally, the UX evaluation should be adjusted so it is sprint-based and centered around bottom-up features design. A UX method meant for use with Agile methods should ideally be lightweight and have ways to control the scope of the project **(Anonymous, 2017, p. 1)**.

Lean philosophy

Definition

The lean philosophy's central belief, as I already covered when describing Lean UX and Lean Startup, is the removal of anything that is not useful for the design or development process. Or, in other words, eliminating waste - which is based on the Japanese industrial philosophy "Muda" **(Stoica et al, 2016, p. 6)**.

Kanban - a Lean method

Kanban, like Scrum, is an iterative method. However, they have some key differences **(Stoica et al, 2016, p. 6)**.

Kanban, unlike Scrum, has an event-driven workflow that is not timeboxed. Deadlines and lead times are used as a metric for improving processes. Estimations are optional, and tasks do not need to be prioritized. Kanban teams do not use roles.

Task tracking is done through a Kanban board, which can be shared across teams **(Powell-Morse, 2017)**. Each task is written on a Kanban card, which is attached to the board. The board itself can be split into several categories, depending on the purpose of the project. A common set is "To do", Ongoing" and "Complete" or "Done". New cards can be added at any given time, and may be moved freely between categories depending on how priorities change. Only a fixed number of cards can be work in progress (WIP); that number is decided on by the team beforehand. Some cards can be "blocked" if their completion is dependent on another task; in physical Kanban boards, that is usually noted by coloring one of the corners in the card in red.

Agile versus Lean - a comparison

To summarize the current subsection, I will now make a comparison between the Agile and the Lean philosophies, as well as introduce Scrumban, a hybrid Agile-Lean method.

Agile versus Lean

The purpose of Agile methods is to allow for flexibility and easy adaptation to changes, as well as give the teams freedom in the quick completion of tasks **(Stoica et al, 2016, p. 5)**. Lean methods, on the other hand, focus on streamlining the development process by eliminating elements that are useless to the clients. These considerations result in easier adaptability in the development process for Agile methods, and a sustainable development process for Lean methods.

Agile methods originated for software development and then grew to marketing and other fields, while Lean methods were designed for factory manufacturing processes in the beginning.

In Agile, artifacts are listed in a Product Backlog. Then, before each sprint, a selection of artifacts are moved to the respective Sprint Backlog. This is done indefinitely until delivering a potentially shippable prototype. Lean, on the other hand, relies on the Build-Measure-Learn feedback loop. An iteration of the product is created, then it is evaluated by testing it with users, and based on the learnings, it is decided whether new additions or changes are necessary and how to approach them.

Progress in Agile methods is demonstrated through the definition of “done”, while in Lean methods, it is through achieving validated learning.

Agile methods include Scrum, eXtreme Programming, and many others, while methods such as Kanban and Kaizen are part of the Lean group.

Agile methods utilize instruments and tools such as boards, sprints, mapping user stories, and acceptance tests. Lean methods, on the other hand, use hypotheses, funnel and cohort analysis, customer interviews and split tests.

Scrumban - a hybrid method

Scrumban is a hybrid method that combines aspects of both of its “parent” methods, Scrum and Kanban. For example, it retains the concept of removing waste and avoiding unnecessary processes from kanban, while largely maintaining the Agile iteration planning of Scrum. It was originally used as an adaptation method for cases when teams transition from using Scrum to using Kanban **(Stoica et al, 2016, p. 7)**, but has evolved since that and can now be considered a standalone method.

Scrumban offers a shorter lead time, continuous improvement and minimized waste. It is more flexible than Scrum, but more structured than Kanban, and it also requires less time to be spent in meetings compared to Scrum, where they are crucial.

Considerations and selection process

Finally, in the “Considerations and selection process” subchapter I will discuss which methods, theories, and frameworks I have chosen to use, and why I did so. This subchapter also includes a literature overview.

Considerations for methods’ selection

In this subsection, I will describe my reasons for selecting the methods I chose, as well as discuss why some of them did not make the cut in favor of others.

Initial idea for master thesis

My master thesis has changed quite a bit since I first began to work on it. My initial idea was to use GDD and Lean UX in conjunction in order to redesign the website of Dynaway, the company where I held my internship and where I was a part-time front-end web developer. Unfortunately, my work relationship with them ended after my internship, so I no longer had access to the company resources I would need in order to realize my idea. Therefore, this iteration came to a close.

The second iteration was close enough ideologically to the first - by using the same methodology, I would create a portfolio website for myself. That proved to be too ambitious of a project for one person given the time constraints, as well as the additional complications due to COVID-19 - user testing got more cumbersome and time-consuming, which I could no longer afford. GDD was also scrapped as a possible framework at some point during this iteration, for reasons which I will explain further down in the thesis.

Those circumstances forced me to reevaluate my approach. While I still wanted to have a thesis topic that was related to my initial idea, I started to think of a more theoretical project. The reasons why I could not realize the second iteration of my idea also got me thinking about the website development process in general, which in turn led to my current thesis.

Why Lean UX and not traditional UX methodology?

While I still intend to “borrow” some ideas, such as the Five Planes, and rely on classic UX for inspiration, my personal preference lies with Lean UX for several reasons.

First, it is more streamlined and gets rid of unnecessary artefacts and documentation, and the focus is on delivering MVPs. The traditional UX design process can be overly complicated, and unfortunately, I am not as well versed in it as I would like to be. From that standpoint, Lean UX makes a good methodology for a “first” project due to narrowing down possible choices for what I should be doing while working on UX design and research.

Another reason is that since it is based on the Lean Startup approach, it is suitable for smaller projects with little to no resources and less initial knowledge. I intend to use the method I create as a result of my thesis after I am done with my master degree, in order to create a portfolio website for myself. Having those considerations in mind, it is a suitable approach, since I will be working alone.

Lean UX is also based on Agile methodology principles, and will therefore be relatively easy to integrate with a development method from that group depending on my needs. That will make some of the decision-making process easier, since the iteration cycles in an agile method will give me a framework as a basis that I can build upon.

Finally, there is a heavy focus on user involvement and design thinking. As a digital concept developer bachelor, I intend to make full use of my design skills, and develop them further by

creating my portfolio website. In order for the portfolio to do as good a job as possible and attract clients for my work, I will need to conduct extensive user testing and keep improving on the website depending on tendencies and trends uncovered by the tests.

Why is GDD no longer a core part of the thesis?

While I initially intended for GDD to be a core framework that I would base my thesis on, it eventually got removed from my plans.

As of the second iteration, creating my portfolio website based on GDD and Lean UX, I found that there were some discrepancies between Lean UX and GDD, despite the fact that GDD is inspired by Lean UX.

First, GDD is targeted towards marketers and encourages them to cover as much of the development process as possible on their own. This, in my opinion, inhibits teamwork, as it practically turns marketers into “one-man armies”. This conflicts with the tenth principle of Lean UX, which claims that there should not be any rockstars, ninjas, or gurus.

There are further discrepancies with other principles, such as GOOB, or constant knowledge sharing throughout the team. GDD relies heavily on hypotheses and fundamental assumptions, and involves user testing very late on in the process, something that Lean UX advises against.

Furthermore, by encouraging marketers to work mostly on their own, do almost all improvements by themselves and disseminate knowledge only after each iteration instead of continually, the awareness of the team about their product, users and user behaviors can be impacted.

Another reason for me to decide against using GDD in the end, is that it is a method developed in conjunction with HubSpot, for use with HubSpot. While I have used HubSpot in the past and agree that it is a very powerful CRM with a wide array of capabilities, the fact that I have to make adjustments to the framework in order to use it with another content management system or website builder (or if I program the website myself, as unlikely as it is considering my skill set) creates more additional work for me that can be avoided. Depending on the type of website, HubSpot may not be the optimal choice for a platform - for instance, it is primarily geared towards small to medium businesses.

Despite my concerns about my own project, GDD can be a good choice for small teams that have a limited set of competencies and as such are forced to cover several roles per person.

Other considerations

Information architecture was part of my choice for theory, since it is a key component of UX, but is often overlooked, or not denoted properly in the process.

Research methodologies were discussed as a point of reference, despite not technically being utilized as a part of this thesis. The reason behind my decision is that research (especially user research) is crucial for good UX.

Literature overview

In this subsection, I will go over my literature list and explain why I chose the items on it as sources, as well as the topics they cover. I will split it into two groups - books, articles, and websites.

Books

The first book I incorporated in my thesis is the "polar bear book", as it is affectionately known among information architecture students. *Information Architecture for the web and beyond* by **Rosenfeld, Morville, and Arango (2015)** is my go-to source for information architecture theory and strategy.

Next on the list is *Lean UX: Designing Great Products with Agile Teams* by **Gothelf and Seiden (2016)**. I used it as a reference for the Lean UX process, as well as some further insight into agile methods, design thinking, and Lean Startup

The *Research Design* book by **Creswell and Creswell (2018)** is where I referenced any mentions about research methodology, approaches, worldviews, and designs from.

I used two books for UX-related references. The first one, *The Elements of User Experience* by **Garrett (2011)** was added to my list specifically for the Five Planes of UX. The second one, *The UX Book* by **Hartson and Pyla (2012)** was the one I used for all other UX references throughout my thesis, as well as the discussion on how agile methods and UX methods work together. I also found out more about design thinking thanks to it.

The Lean Startup by **Ries (2011)** is my source for the Lean Startup method, which I describe as a point of reference for Lean UX, since the latter is based on the former.

Finally, **Olsen and Pedersen (2015)**'s *Problem-oriented project work*, while not directly referenced in the report, was used for structuring my whole master thesis creation process. I also used it as a guideline for structuring my thesis writing.

Articles

The most crucial article for my thesis is definitely the *Analyzing Agile Development - from Waterfall to Scrumban* by **Stoica, Ghilic-Micu, Mircea, and Uscatu (2016)**. This article proved to be a great source on agile methodology, lean methodology, comparing the two, as well as Scrum, Kanban, and Scrumban. Thanks to it, I was also able to locate the *Manifesto for Agile Software Development* by **Cunningham (2001)** where I referenced the principles of Agile Development from.

There is one more article by the majority of the same authors, *Collaborative Environment and Agile Development* by **Ghilic-Micu, Stoica, and Mircea (2014)** that focuses on the benefits of software testing, validation and verification.

Another key article, this time for the discussion on users and their relationship with design, was *Designing for imaginary friends: information architecture, personas and the politics of user-centered design* by **Massanari (2010)**.

Collaborative UX Design Methods for Developing Social Solutions by **Yasu, Iwata, Yamazaki, and Kohno (2014)** is an article on two UX design methods that involve the entire team.

HOW TO IMPROVE YOUR UX DESIGN by an **Anonymous (2017)** author is a short list of things that one should be aware of when designing user experience.

Lean UX: Rethink Development is a secondary reference and shorter summary of Lean UX written by **Cyrillo (2011).**

Selected issues of information architecture by **Lange-Sadzińska (2012)** is an article that describes the difficulties with information organization, as well as labelling and choosing searching schemes in an information system.

UX Research vs. UX Design by **Karr (2015)** is an article on the difference between UX research and UX design, as well as what UX itself is.

Finally, *Voice and Tone as Information Architecture* by **Garklavs (2016)** is an overview on how to use voice and tone as part of the information architecture, as well as how to communicate through them.

Websites

I used several websites or web articles as sources. The first one, *Kanban Methodology: What is it and how do you use it?* by **Powell-Morse (2017)** is an article on the Kanban software development method, as its title suggests.

Two of the web sources I used are for the correlation between Lean [UX] and Kanban. The first one is *An introduction to Lean Kanban software development* by **Scott (2018)**, and the other - *The Kanban Method in Lean UX* by **Chung (2015)**.

Finally, I based my discussion on why traditional website design is insufficient on the article *7 REASONS THE TRADITIONAL WEB DESIGN PROCESS IS BROKEN* by **Nyquist (2018)**.

Last, but not least, the course on *Growth-driven Design* by **Summerfield (2017)** has been a source for statistics and information on why the traditional web design model is no longer working, as well as the Growth-driven Design method itself.

Analysis

In this chapter, I will explain why the traditional web design process is inefficient, and then present two possible solutions to the problem.

Why is the traditional web development process inefficient?

According to some sources, there is more to be desired from traditional web development processes **(Summerfield, 2017)**. While some only claim that it is risky and expensive for businesses, or that is simply "bad" with no further reasoning, others give a more detailed breakdown of reasons. **Nyquist (2018)**, for instance, has listed seven reasons for the inadequacy of traditional web design.

As I mentioned above, the number one reason is that traditional web development is risky due to the lack of user involvement throughout the process. Development is often based on assumptions about what the users might like to see, but that does not always correspond to their real needs or relieve their pains.

The second reason is that the process, especially when developing a new website, can be very bulky. It can take months for a website to be put live, and that includes organization, decision-making, meetings, and developing the website.

New websites can present a risky pitfall - developers are often tempted to make small tweaks here and there, which risks a domino effect. By making small fixes without exercising caution, the entire website is in danger of going down for repairs.

As I described before, traditional web design does not really focus on the users or customers. There are no user metrics to guide the process, which may result in unnecessary or unhelpful parts being added with no merit to the users, or to their detriment.

The fifth factor that plays into the inefficiency of traditional web design is that it tends to create siloes. Different areas and departments get isolated and each works on their own area with very little knowledge sharing involved. This creates ample opportunities for miscommunication and mistakes.

Yet another reason I mentioned before is that traditional web development methods are expensive. **Summerfield (2017)** quotes prices between $15.000 to $80.000 for a website. Website design can be time-consuming and use up a lot of energy, and outsourcing some of the work can get costly.

Last, but not least, a website design will likely not stay relevant for long. Traditionally created websites often collect dust for a long time after they get launched, and no real user metrics are collected for the purpose of the website's improvement. With the process as uncomfortable

and cumbersome as it is, it is likely that nobody would want to or be able to attempt redesigns or updates often anyway.

Possible solutions

In this section, I will present two possible solutions for optimizing the website development process. The first is based on Scrum with Lean UX incorporated in it, while the second is a Lean UX - Kanban mix.

Both feature the five planes of UX, since it gives a good outline and segmentation of different elements that are a part of a traditional UX design process, without being too specific or detailed, which might conflict with the Lean in Lean UX by creating unnecessary waste.

As information architecture is a part of the Structure plane of the five planes, its planning will also be incorporated in the designs.

Lean UX + Scrum

In this subsection, I will discuss the hybrid method between Lean UX and Scrum, its related terminology, advantages and disadvantages, and I will showcase its lifecycle.

Description and terms

In order to make sense of the method, and the specifics of Scrum in particular, I will outline some terms used.

First, *user stories* are the smallest possible work units that present any benefit to the user **(Gothelf and Seiden, 2016, p. 96)**. They are typically written as follows:

As a [user type] I want to accomplish _____, so that _____ happens.

The second important term is *backlog*, or prioritized lists of user stories. In Scrum, there are two types - a Product Backlog for all user stories of a project, and a Sprint Backlog in which stories selected based on their priority are added every sprint.

Third, a *sprint* is one team cycle, and the goal of each one is to deliver working software in the end. Scrum sprints usually last two weeks.

There are also three types of meetings associated with Scrum.

The first, the *Stand-up*, is a short, daily meeting where the challenges of the day are discussed by each member in a row. All team members must announce to the rest of the team what they are going to work on any given day, as well as what is hindering their work.

The second type of meeting is the *Retrospective*, which takes place at the end of each sprint. It facilitates an honest discussion about what went well during the sprint and what did not, and gives the project team a chance to optimize their workflow.

Finally, an *Iteration planning meeting* takes place at the beginning of each new sprint. This is the time when the project team plans the upcoming sprint, estimate deliverables and prioritize the tasks in the backlog.

Lifecycle

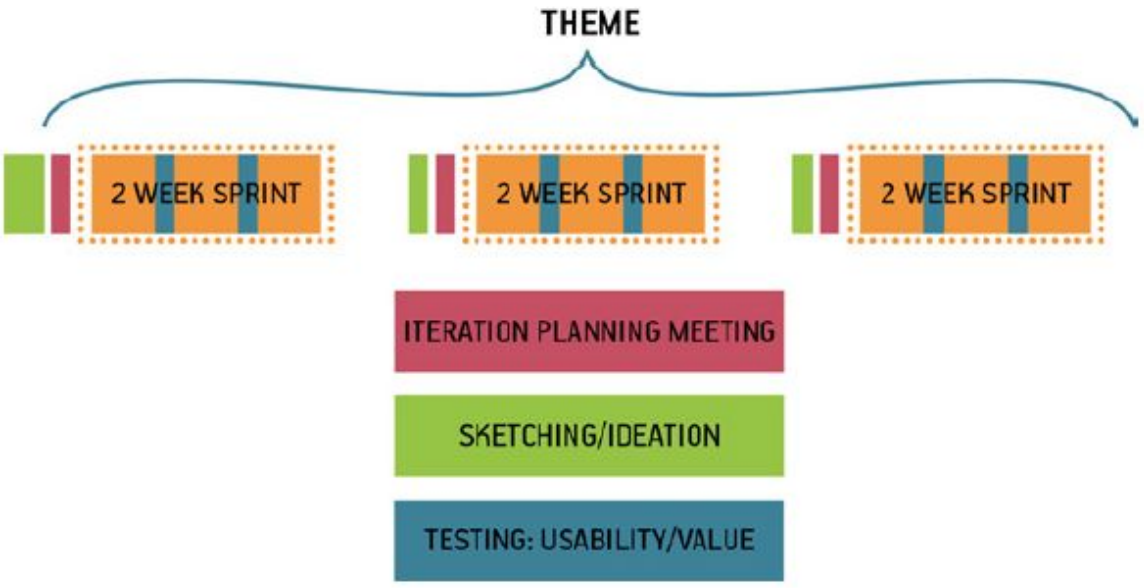

Figure 11 - The lifecycle of a Lean UX + Scrum hybrid method. Taken from Gothelf and Seiden, 2016, p. 100.

This is a model based on Scrum and Lean UX that incorporates a collaborative routine (Figure 11). It preserves Scrum's meeting structure and still adheres to the principles of Lean UX.

Meetings can take up a lot of time, and, in addition, waste a lot of it. This is circumvented by integrating them as milestones in the sprints. That way, the whole team is working on the same thing simultaneously **(Gothelf and Seiden, 2016, p. 98)**.

To begin, series of sprints with common goals can be grouped together under a theme. The lack of design iteration in classic Scrum can be compensated by adding a kickoff session for ideation and sketching at the beginning of each sprint. As there is more information to work with as the project progresses, the kickoff sessions get shorter and shorter with each consecutive sprint. As usual with Lean UX, everyone on the team should participate in all activities.

The Iteration Planning meeting should take place right after the kickoff session, and be conducted as usual, except for the fact that the whole team must be present regardless of their role and participate in the creation and prioritization of user stories.

Finally, in order to circumvent the lack of user involvement throughout the Scrum process, a User Validation Schedule must be created. For sprints that are two weeks long, they should ideally take place every week. That way, the project team will have enough time to react to user feedback until the end of each sprint, while never having to wait more than five business days until a feedback session. Initial tests should uncover whether the users are interested in an idea, while consecutive ones later on will reveal whether the solution created by the team is, in fact, usable.

To add to the method, the five planes of UX can be incorporated into the design ideation meeting by using them as the themes for several groups of sprints. For example, the first theme would be Strategy, the second - Scope, and so on, until the Surface plane is the last one. By doing this, every plane can be run through the sprint cycle and its results and findings - tested and validated **(Ghilic-Micu, Stoica, and Mircea, 2014, p. 5)**.

The inclusion of the five planes will allow information architecture to not be overlooked or designed based on “gut feeling” alone, since it is a part of the Structure plane of UX.

Advantages

Compared to using a traditional method for web design of “vanilla” Scrum, this hybrid puts more emphasis on user analysis and testing, as well as on design iteration and cross-team collaboration. It addresses several issues with Scrum that are visible from a UX standpoint, while not altering the process to the point where it becomes uncomfortable or alien for teams that are already used to Scrum.

Disadvantages

The main disadvantages that I can see with this model is that the Scrum part and its meetings need to be closely observed and attended by absolutely everyone on the team. If specific team members feel the need to miss out on a meeting because there is nothing there that involves them, that could break the communication flow further on in the process.

Similarly, special attention should be paid to ensure that all disciplines, ideas and points of view are taken into account during ideation sessions and iteration planning meetings. All team members should have equal input regardless of their expertise and role in the project.

Lean UX + Kanban

In this section, I will cover the Lean UX + Kanban hybrid method, its lifecycle, as well as advantages and disadvantages.

Description and terms

As with the previous hybrid method, I will now outline some terminology that makes it easier to understand Kanban.

First, one of the most important concepts in Kanban is *WIP - work in progress* or *work in process*. Both are used interchangeably for indicating how much work is actively ongoing at any given moment **(Scott, 2018)**.

JIT or *just-in-time* is another key concept of Kanban and Lean methods in general **(Chung, 2015)**. It is based on the idea of delivering products or any kind of deliverable as it comes in demand, not sooner (stockpiling unnecessarily) or later (delaying the process).

Pushing and *pulling* are two central concepts to Agile and Lean, respectively.

Pushing a task means moving it to the next progressive category in a project board (e.g. from "In Progress" to "Tech review") **(Scott, 2018)**. This, however, means that whoever pushed the task card has now put it into someone else's hands - and that someone does not necessarily have the time to look at a card, thus creating possible bottlenecks.

Pulling, on the other hand, is a very similar concept with an additional detail - a buffer category between the two (e.g. from "Ongoing" to "Ready for tech review" to "In tech review"). The person responsible for the tech review can then "pull" the card to the respective column when they are ready to start work on it. This allows for a better overview of where a process might be stagnating, as well as whether a task is currently being worked on, or is in standby.

The *Kanban backlog* is where all tasks or user stories in the form of *Kanban Cards* are displayed. At some point, they are taken and put into the "To do" column (or its equivalent) of the *Kanban board*, and entered into the process.

Lifecycle

While it is hard to speak of a lifecycle with Kanban and its lack of timegated iterations, that is not necessarily a bad thing. Integration with Lean UX is easy, because both are Lean methodologies, and Kanban is already a key building block of Lean UX **(Chung, 2015)**.

Lean UX supports the pulling method of tasks from Kanban, and their conjunction optimizes the workflow, as well as the cycle time of production **(Chung, 2015)**. Utilizing them together also limits the scope of the work depending on the capacity of the project team, and puts emphasis on resources and data over assumptions. Thus, a quick development workflow is balanced with quality work.

As for the five planes of UX, a way to incorporate them into the process would be to split work into phases (similarly to the themes of the Lean UX + Scrum hybrid). This way, the five planes and any milestones and artifacts associated with them will become the "backbone" of this

method. The project would start with the Strategy plane, and go on until the Surface plane. To keep in mind the event-based sprints, completing work on a plane or one of its aspects (functional or information side) can be a possible event to signify the end of a sprint. User testing can be conducted as soon as a feature is completed and a testable MVP based on it is ready.

Advantages

One of the biggest advantages of this approach would be that by splitting tasks into bite-sized pieces, the project team will be better equipped to deal with bottlenecks in the project workflow. This will also help with the prevention of burnout syndrome.

Through the use of a Kanban board, the project workflow can be monitored by all parties related to the project, be it users, customers, or the project team. That approach helps by visualising and keeping a record of the workflow.

Since less time is spent on estimations, meeting times in Kanban-based methods are significantly lower than their Scrum counterparts - according to **Scott (2018)**, the average Scrum team spends 19.3% of their week on meetings, compared to 8% for Kanban teams.

Thanks to the JIT system, there is less of the product that needs to be reworked when changes arise. Tasks are only added as they become necessary, which removes the need for adjusting cards on the Kanban Board when new requirements become clear, thus reducing risk.

Finally, both customers and the team feel happier because they are part of the loop of the project. The clients see new work being tested with the users almost daily, and can see clear metrics that help them understand progress at any time. The team, on other hand, spends less time on meetings and more on working on the solution. The flexibility of the method allows them to easily adapt to change, instead of being encumbered by a detailed plan.

Disadvantages

Possible disadvantages that I believe are applicable for this method is that maintaining and constantly keeping an eye on a Kanban board can be chaotic or confusing for teams who have little experience with Kanban. The lack of prioritization of the tasks could further contribute to that.

Other issues include that the “To do” category (or its equivalents) can become too big and thus overwhelming.

Another concern is that too many or overlooked blocked cards can create bottlenecks if the tasks they depend on are dragged on for too long. However, that is also an issue with “vanilla” Kanban.

Conclusion

The traditional web design methods are considered to be cumbersome and inefficient more and more often. Some of the reasons are that such processes can be risky, bulky, too expensive, and unable to keep with user demands and needs.

By proposing two different combination methods for web development, I have presented two viable solutions that reduce risks by providing an approach based on participatory design that involves the users at every stage of the process. Both are based on Lean UX principles and best practices mixed with Scrum and Kanban, and rely heavily on the five planes of UX and information architecture in order to streamline the development process, reduce work time and costs, contribute to team cohesion and involve all team members in all aspects of the work.

Both methods are viable and usable depending on the situation and the needs of the project teams and clients.

In the future, I would like to develop my own portfolio website based on the Kanban version. While I have no specific preference myself and am equally comfortable with using both, my choice lies with it solely because of the reason that the Scrum variant has three mandatory roles that cannot be fulfilled by the same person, and I work alone. If I acquire some teammates, I would be more than happy to put both to the test and draw up a comparison on their performance with real data.

Perspective

This thesis was a wild ride, to say the least. It sounds cliche, but I never expected it to get to the point where it is now - my initial idea was very different. Nonetheless, I am happy with the results, and learned a lot in the process. I would like to think that the thesis went well.

Still, there are some things that I would have liked to do differently. I spent too much time getting caught up on ideas of my “ideal” thesis topic, that I got frozen up on it for months. Shaking that feeling off would have given me more time to test out more methods with Lean UX, and possibly find or come up with more hybrids to compare with.

If there is anything I regret, that is not finding out about Lean technologies and Lean UX earlier on while I was still in university. I think it provides an interesting perspective on optimizing the work process, and I would have used it in projects if I had known about it.

Bibliography

Literature list

Anonymous. (2017). *HOW TO IMPROVE YOUR UX DESIGN*. Computer Arts, (269), 81–81. Retrieved from http://search.proquest.com/docview/2002996724/

Chung, B. (2015). *The Kanban Method in Lean UX*. IXD@PRATT. Retrieved from http://ixd.prattsi.org/2015/04/the-kanban-method-in-lean-ux/

Creswell, J. W. and Creswell, J. D. (2018). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (5th ed). Thousand Oaks. SAGE Inc.

Cunninham, W. (2001). *Manifesto for Agile Software Development*. Retrieved from https://agilemanifesto.org/

Cyrillo, M. (2011). *Lean UX: Rethink Development*. InformationWeek, (1316), 40,42,44. Retrieved from http://search.proquest.com/docview/904423997/

Garklavs, K. (2016). *Voice and Tone as Information Architecture*. Bulletin of the Association for Information Science and Technology, 42(5), 19–22. https://doi.org/10.1002/bul2.2016.1720420506

Garrett, J.J. (2011). *The Elements of User Experience: User-Centered Design for the Web and Beyond* (2nd ed). Berkeley. New Riders.

Ghilic-Micu, B., Stoica, M., & Mircea, M. (2014). *Collaborative Environment and Agile Development*. Informatica Economica, 18(2), 32–41. https://doi.org/10.12948/issn14531305/18.2.2014.04

Gothelf, J. and Seiden, J. (2016). *Lean UX: Designing Great Products with Agile Teams* (2nd ed). Sebastopol. O'Reily Media, Inc.

Hartson, R. and Pyla, P. S. (2012). *The UX Book: Process and Guidelines for Ensuring a Quality User Experience* (1st ed). Waltham. Morgan Kaufmann, Elsevier.

Karr, A. (2015). *UX research vs. UX design*. Interactions, 22(6), 7–7. https://doi.org/10.1145/2834964

Lange-Sadzińska, K. (2012). *Selected issues of information architecture*. Informatyka Ekonomiczna, (24), 47–59.

Massanari, A. (2010). *Designing for imaginary friends: information architecture, personas and the politics of user-centered design*. New Media & Society, 12(3), 401–416. https://doi.org/10.1177/1461444809346722

Nyquist, M. (2018). *7 REASONS THE TRADITIONAL WEB DESIGN PROCESS IS BROKEN*. Franklin Energy. Retrieved from https://blog.franklinenergy.com/7-reasons-traditional-web-design-process-is-broken

Olsen, P. B. and Pedersen, K. (2019) *Problem-oriented project work: A workbook* (5th ed). Samfundslitteratur.

Powell-Morse, A. (2017). *Kanban Methodology: What is it and how do you use it?*. Airbrake. Retrieved from https://airbrake.io/blog/sdlc/kanban

Ries, E. (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses* (1st ed). New York. Crown Publishing Group, Random House, Inc.

Rosenfeld, L., Morville, P. & Arango, J. (2015). *Information Architecture for the web and beyond* (4th ed). Sebastopol. O'Reilly Media, Inc.

Scott, A. (2018). *An introduction to Lean Kanban software development*. Inviqa. Retrieved from https://inviqa.com/blog/introduction-lean-kanban-software-development

Stoica, M., Ghilic-Micu, B., Mircea, M., & Uscatu, C.. (2016). *Analyzing Agile Development – from Waterfall Style to Scrumban*. Informatică Economică, 20(4), 5–14. https://doi.org/10.12948/issn14531305/20.4.2016.01

Summerfield. L. (2017). *Growth-Driven Design*. HubSpot Academy. Retrieved from https://ap.hubspot.com/academy/5964155/tracks/21/152/779

Yasu, H., Iwata, N., Yamazaki, A., & Kohno, I. (2014). *Collaborative UX design methods for developing social solutions*. NEC Technical Journal, 8(3), 23–27.

Reference guides

Aalborg University. *APA Style*. Retrieved from https://www.cdm.aau.dk/student_handbook/project_report/apa/

American Psychological Association. *APA Style*. Retrieved from https://apastyle.apa.org/

Mendeley. *APA Citation Guide*. Retrieved from https://www.mendeley.com/guides/apa-citation-guide

Appendices

Appendix 1 - Literature list approval

Re: Literature list

David Jakobsen <davker@hum.aau.dk>
5/20/2020 22:00

To: Elitsa Aleksandrova Dimitrova

Hi Elitsa

Great! It is approved.
All the best
David

Sendt fra min iPhone

Den 20. maj 2020 kl. 20.22 skrev Elitsa Aleksandrova Dimitrova <edimit18@student.aau.dk>:

Hi David,

I went through the articles and several made the cut. Here is the updated list:

1. Anonymous. (2017). *HOW TO IMPROVE YOUR UX DESIGN*. Computer Arts, (269), 81–81. Retrieved from http://search.proquest.com/docview/2002996724/ **1 page**
2. Creswell, J. W. and Creswell, J. D. (2018). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (5th ed). Thousand Oaks. SAGE Inc. *pages 1-267* **total: 266 pages**
3. Cyrillo, M. (2011). *Lean UX: Rethink Development*. InformationWeek, (1316), 40,42,44. Retrieved from http://search.proquest.com/docview/904423997/ **3 pages**
4. Garklavs, K. (2016). *Voice and Tone as Information Architecture*. Bulletin of the Association for Information Science and Technology, 42(5), 19–22. https://doi.org/10.1002/bul2.2016.1720420506 **4 pages**
5. Garrett, J.J. (2011). *The Elements of User Experience: User-Centered Design for the Web and Beyond* (2nd ed). Berkeley. New Riders. *Pages 2-164* **162 pages**
6. Ghilic-Micu, B., Stoica, M., & Mircea, M. (2014). *Collaborative Environment and Agile Development*. Informatica Economica, 18(2), 32–41. https://doi.org/10.12948/issn14531305/18.2.2014.04 **10 pages**
7. Gothelf, J. and Seiden, J. (2016). *Lean UX: Designing Great Products with Agile Teams* (2nd ed). Sebastopol. O'Reily Media, Inc. *pages 3-173* **total:170**
8. Hartson, R. and Pyla, P. S. (2012). *The UX Book: Process and Guidelines for Ensuring a Quality User Experience* (1st ed). Waltham. Morgan Kaufmann, Elsevier. *Pages 1-75, 251-291, 333-352, 391-422, 427, 492, 503, 619-634* **total: 245 pages**
9. Karr, A. (2015). *UX research vs. UX design*. Interactions, 22(6), 7–7. https://doi.org/10.1145/2834964 **1 page**
10. Lange-Sadzińska, K. (2012). *Selected issues of information architecture*. Informatyka Ekonomiczna, (24), 47–59. **13 pages**
11. Massanari, A. (2010). *Designing for imaginary friends: information architecture, personas and the politics of user-centered design*. New Media & Society, 12(3), 401–416. https://doi.org/10.1177/1461444809346722 **16 pages**
12. Olsen, P. B. and Pedersen, K. (2019) *Problem-oriented project work: A workbook* (5th ed). Samfundslitteratur. *Pages 9-351* **total: 342 pages**
13. Ries, E. (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses* (1st ed). New York. Crown Publishing Group, Random House, Inc. *pages 1-309* **total: 308 pages**
14. Rosenfeld, L., Morville, P. & Arango, J. (2015). *Information Architecture for the web and beyond* (4th ed). Sebastopol. O'Reilly Media, Inc. *pages 3-451* **total: 448 pages**
15. Stoica, M., Ghilic-Micu, B., Mircea, M., & Uscatu, C.. (2016). *Analyzing Agile Development – from Waterfall Style to Scrumban*. Informatică Economică, 20(4), 5–14. https://doi.org/10.12948/issn14531305/20.4.2016.01 **10 pages**
16. Yasu, H., Iwata, N., Yamazaki, A., & Kohno, I. (2014). *Collaborative UX design methods for developing social solutions*. NEC Technical Journal, 8(3), 23–27. **5 pages**

All new additions are articles, except No.2 (research methods) an No. 4 (mainly for the theory on the five planes of UX, but also so I have an extra source to cite for definitions). I also removed a lot of pages from No.8, since I won't be using all 904 pages from that book 🙂

With the new additions, the page count comes up to **2,004 pages** (phew, close call!) Please let me know if anything needs changing again.

Best regards,
Elitsa Dimitrova

Please also find the *Appendix 1 - Literature list submission* PNG file attached in the exam submission.

Appendix 2 - GDD course transcript

Please find the *Appendix 2 – Growth-Driven Design course transcript* PDF file attached in the exam submission.

Appendix 3 - report structure template

Sample report structure based on "Problem-oriented project work", ch. 16.

1. Introduction
    a. Front page
        i. The function is to attract interest
        ii. Mostly visual
    b. Title page

        i. Personal info, uni program, year, supervisor, topic (check requirements)
    c. (Preface)
        i. Reading suggestions or thanking people for helping/supporting the project
        ii. Not mandatory
        iii. Be careful to not be too pompous/sentimental
    d. (Summary)
        i. What the report is about
        ii. Must contain
            1. Knowledge problem
            2. design/method
            3. Most important conclusions
        iii. ½ to 1 page long max
    e. Table of contents
    f. (Lists of figures, tables, and abbreviations)
2. Introduction
    a. Teaser
        i. It makes the reader interested and “hungry” to read on. It can be an anecdote, a statement from an interviewee, an incident, a paradox, etc. Something that sets the stage for the knowledge problem.
    b. Problem area/description
        i. This is where I describe the problem and set the stage for the problem formulation. Must contain:
            1. Knowledge problem - what is under investigation and why is it a problem?
            2. Documentation - show that the knowledge problem really is one. Explain the theory/knowledge that is the core of the problem and provide empirical evidence for findings.
            3. Interest - get the reader interested. Show why you are interested.
            4. Problem formulation - make it short with concepts that have been discussed/clarified beforehand.
            5. Define all significant concepts and set up limitations of the problem formulation.
    c. (Reading guide)
        i. How should the report be read? In what sequence? What is the function of each part/chapter/section?
3. Design or method report
    a. Splitting the problem formulation into tasks of inquiry
    b. Work and research techniques (e.g. choosing and working with the sources; interviews, tests, etc.)
    c. Analysis strategy - cause-effect chains, or reflections that justify my interpretation
    d. Possibly a graphic illustration of the design, if applicable

4. Analysis/body
    a. Analysis of tasks of inquiry
    b. Filling out design
    c. Empirical data and theory
    d. Method report - quality assessment
    e. Theory first, empirical data and analysis after
    f. Start every chapter with an introduction, finish with a summary/sub-conclusion
5. Conclusion
    a. Answer the problem formulation
    b. Should
        i. Summarize the problem area well and repeat the problem formulation if necessary
        ii. Be bold, but not presumptuous - say something that others might question or criticize, and conclude something that can actually be tested or discussed.
        iii. Be well documented - there must be empirical and analytical evidence for the conclusion reached
        iv. Show that I'm self-critical - how far-reaching are my conclusions and their certainty.
6. Perspective
    a. What are the limitations of the project report?
    b. New important knowledge problems that have emerged as a result of the project
    c. The significance of the conclusion to the problem area
7. Closing
    a. Bibliography
    b. (Appendix)
    c. (Reverse with short summary)