1.8: Lean UX vs. Agile UX

#### **Introduction**

Good to have you back! In the previous Exercise, we discussed competitive analyses and how you can use them to better understand the market for your own product. We also talked about how a UX analysis can help you uncover UX issues within a competitor’s product, which can ultimately be used to improve your own product. You even had a chance to perform competitive and UX analyses of your own!

Today, we’re going to shift gears just a tad and start diving deeper into the nitty-gritty aspects of UX design. If you’ve done any other research into UX design, you may have seen the terms Agile UX and Lean UX pop up from time to time. Although these are two different methodologies, both draw at least some of their everyday practices from Agile and both provide frameworks for building better, more user-friendly products.

UX designers love to argue about the differences between Agile UX and Lean UX. Some believe there’s no difference between the two, while others loathe the terms altogether! At the end of the day, however, what’s most important is that you understand the goal behind these methodologies and are able to apply them to your own products. Let’s get started!

#### **Getting Lean**

##### **What Is Lean UX?**

Lean UX is a methodology that combines the principles of user experience design with a concept known as "Lean Startup." **Lean Startup** was first proposed by entrepreneur Eric Ries back in 2008. It was originally formulated to help startups succeed but is now also used by established companies when designing and releasing a new product. We’ll take a closer look at how the principles of Lean Startup shape the practices of Lean UX in a moment.

Lean UX is underpinned by three foundations from which it draws a set of principles to inform everyday practice of the methodology. The foundations of Lean UX are:

**Design Thinking**: Thinking back to Exercise 1.5, you’ll remember that Design Thinking brings together business requirements, technological capabilities, and user needs to identify innovative solutions to problems. Lean UX uses Design Thinking to assert that design methods can be applied to all aspects of a business, in turn bringing designers and non-designers together to facilitate collaboration through cross-functional teams.

**Agile Software Development**: Agile software development methods are well-established among developers who use them to shorten development cycles and maintain a focus on delivering value to customers. Lean UX adopts the four key values that underlie Agile software development and incorporates them into everyday practice of Lean UX.

**Lean Startup**: Lean Startup aims to reduce the risk associated with launching a new product by releasing products early and using validated customer feedback to inform further iterations of a product. To do this, a feedback loop called **build-measure-learn** is used.

Before "building" the product, an idea for a product solution first needs to be formulated into a **hypothesis** so that the project team can set goals for what they want to learn during product testing. Once a hypothesis is established, you can then start the build-measure-learn cycle. Now, let’s take a look at each phase of the cycle in more detail:

* **Build**: In this phase, the hypothesis is tested with real or potential customers in the form of a **minimum viable product**, or MVP. An MVP is the most-pared-down version of a product that still provides enough value to potential users that validated learnings can be obtained from customers’ early experiences. These learnings are then used to inform ongoing improvements to the product.
* **Measure**: During this phase, data is collected from users to determine whether the project team’s initial hypothesis was correct.
* **Learn**: In this phase, project teams use the information from the "measure" phase to understand what features of the product are working well and which need to be changed or improved. At the end of this phase, the whole cycle starts again with a new hypothesis.

Let’s apply the build-measure-learn feedback loop to the example of adding a chat feature to a web app. You would first build an MVP of the chat feature, then release it to a sample of your existing customers for testing, having first defined a hypothesis for your test. This MVP would be nothing but a bare-bones version of the feature and might include the chat function itself, for instance, but not the upload and share functions (these being still to come). Learning from past iterations is what marks progress in Lean UX.

You’ll notice right off the bat that the build-measure-learn cycle is extremely simple. This is by design. Rather than restricting your design process with granular direction, the loop focuses, instead, on high-level guidance. The power of the cycle lies in its rapid-fire iterations that lead to prompt solutions, making designing a user-friendly solution easier, quicker, and more affordable.

##### **The Principles of Lean UX**

From these three foundations–Design Thinking, Agile Software Development, and Lean Startup–Lean UX draws a set of principles that are used to inform everyday practice of this methodology. As there are quite a few principles, we’ve selected some of the most important ones for you to consider as you work on your project. If you’d like to take a look at the principles in more detail, check out [this article](https://www.oreilly.com/ideas/what-is-lean-ux).

**Cross-Functional Teams**: In order for Lean UX to work successfully, cross-team collaboration is essential. Depending on the nature of your product, everyone from developers, content strategists, UI designers, marketers, and customer service agents needs to be involved from day one of creating the product.

**Removing Waste**: One of the core elements of Lean UX is the removal of anything that isn't an improvement to the product experience. Design deliverables aren't created merely for the sake of it–rather, they must be valuable in helping to shape and improve the product experience for users.

**Small Batch Size**: Only create features necessary for testing the product and gathering validated learnings. There's no reason to waste time creating a big list of features that will never get implemented or tested.

**Continuous Discovery**: You should be constantly engaging with your customers throughout the design process. Teams test whatever they have available on a regular basis, be it some sketches on sticky notes or a mid-fidelity digital prototype. At each stage, ideas are tested in order to inform the next stages of a product’s design and development.

Before this new type of thinking came about, everything was about deliverables—wireframes, mockups, flow charts, anything that could be considered a tangible product. Lean UX, on the other hand, aimed to move the focus away from deliverables and, instead, towards the success of the experiences being designed.

What happens, however, when you’re not sure what it is you’re building? Or you’re building on top of a brand-new technology? This is where Lean UX really shines because it allows you to make assumptions and test them through an iterative, short-cycled process. This process aims to get rid of (or greatly strip down) requirement documents, providing only the information necessary to get started with the implementation. Instead of learning through trial and error, one can learn through “trial and iterations” and effectively minimize the negative impact of any unforeseen errors.

#### **Agile UX**

##### **What Is Agile?**

Before we can understand Agile UX, we first need to take a step back and explore the values and principles of the Agile methodology itself.

Agile is an all-encompassing term that covers a range of different methodologies for software development, all of which use working processes that emphasize highly collaborative and cross-functional teams. Agile Software Development came about after a crisis in the 90s when the proliferation of the personal computer led to a high demand for computer software. Developers wanted to get new software out and fast, but the lack of a development cycle meant that moving from defining a business need to developing a piece of software could take as long as three years!

In a world of rapidly changing technology, you can see how this would be a problem. By the time a company could figure out what users needed, that need was long gone, and the software was already out of date.

The result of this was the creation of the “Agile Manifesto," a set of four key values that aimed to shorten product cycles and keep the focus on the customer. This is achieved through prioritizing the creation of working software over documentation, providing value to customers instead of following strict processes, and being open to change rather than following a pre-set plan.

In contrast to earlier software development methodologies that tended to follow inflexible, sequential working processes with considerable documentation, Agile offered development teams the ability to deliver valuable experiences to customers within short time frames and to continue iterating on the product after its launch without the need for detailed, up-front project documentation.

##### **Agile Software Development in Practice–the Scrum Framework**

Let’s take a brief look at a popular Agile framework called Scrum. The Scrum framework uses the following elements to manage complex projects by breaking them down into smaller chunks, allowing teams to focus on one aspect or feature of a product before deciding what needs to be built next.

**User Stories**  
In Scrum, **user stories** are created to describe, from the user’s perspective, exactly what tasks they need to be able to complete in relation to interacting with the features of a digital product. UX designers play a key role in crafting user stories that reflect the needs, goals, and behaviors of users and ensuring that the development team is clear on the exact requirements of each user story.

**Sprints**  
Once user stories have been created, they get put into a list known as a **backlog**. A set of these user stories are then taken from the backlog and built via a short development cycle known as a **sprint**. A sprint is a fixed time period (most commonly two weeks) during which a working increment of the product is built–ready to be reviewed and, potentially, released.

**Sprint Goals**  
Sprint goals are used in Scrum to describe what the team wants to achieve during an upcoming sprint. For example, a sprint goal might be to build one small feature of a web app such as a sign-up form for a newsletter.

**Daily Standups**  
Each day, the Scrum team takes part in a **daily standup**. This is a short, informal meeting during which each member of the team gives a brief update on the work they’re doing to meet the sprint goal. The purpose of daily standups is to maintain an open and collaborative working process and keep everyone focused on the sprint goal. You may also hear these sessions referred to as the "daily scrum."

Now, let’s look at the chat feature example from an Agile UX point of view using the Scrum framework. This feature would be divided into many small but measurable tasks that could be accomplished in short periods of time. One task might be as small as choosing the color scheme of the feature or uploading an avatar. When sprint goals are set, a “definition of done” is also created, which sets out exactly what deliverables or outcomes need to be achieved in order to consider the work for that sprint complete. Meeting the “definition of done” for each broken-down and actionable task is what marks progress in Agile UX.

By working only on one small aspect of a product at a time, the Scrum framework and other Agile methodologies allow development teams to more easily make changes to a product’s features from one sprint to the next.

##### **Bringing Agile and UX Together**

As we now know, Agile software development was created by developers who needed a way to ship working software quickly and efficiently. As such, UX designers weren’t the primary focus of Agile, and, in order to incorporate designers into the Agile process, **Agile UX** came into being.

Agile UX seeks to unify developers and designers within the product development process by adopting UX processes into the Agile methodology. In many ways, this works well, as UX and Agile have many factors in common including collaborative working processes, a focus on providing value to users, and an iterative approach to making product improvements.

However, practicing Agile UX can be challenging for designers. Due to its focus on short, fast product development cycles, it can be difficult for UX designers to find sufficient time to follow the phases of the design process and give enough attention to understanding users’ needs and goals in order to create designs that solve genuine problems.

Additionally, because Agile is focused on building one small increment of a product at a time, this can lead to tunnel vision for the design team as they can easily lose sight of the overall user experience of the product as a whole.

Therefore, careful consideration is needed when implementing Agile UX working processes into project teams to ensure that designers have the time they need to focus on creating meaningful solutions for customers while still being able to match the pace of the development team.

#### **Lean UX vs. Agile UX**

Agile is an integral part of both Lean UX and Agile UX, with the difference being that in Lean UX, Agile values are incorporated to promote a more efficient and collaborative UX design process with a focus on continually testing product iterations in the form of MVPs. In Agile UX, however, UX processes are integrated into Agile methodologies such as Scrum with the aim of unifying designers and developers within short, incremental product design and development cycles. For a quick review of Lean and Agile UX, take a look at the table below.

| **LEAN UX** | **AGILE UX** |
| --- | --- |
| Focused on the design phase of the software development process | Integrates UX design processes into Agile methodologies |
| Enables design processes to be applied to all aspects of a business | Focuses on the needs of developers with UX practices added |
| Combines principles from Design Thinking, Lean Startup, and Agile | Integrates four key values that enable development teams to ship working software and valuable experiences to customers using short product cycles |
| Uses a build-measure-learn loop to continuously test and iterate on product design | Uses short product design and development cycles to release small pieces of working software |
| Products are shipped early in the form of an MVP that provides value for customers, while also providing validated learnings for future product iterations | Product features are designed and released incrementally |

As you can see, the two methods come with slightly different ways of looking at the process, but, in the end, the output is still a successful, polished product. At the end of the day, which method a team decides to use is up to the team itself, as well as the business requirements and market in which they’re operating.

Generally speaking, Lean UX is great for getting an initial product off the ground as it allows for quick ideas and iterations. While the product may not be complete, it still enables you to get something out there and into the hands of your customer. Agile UX, on the other hand, comes into play once you’ve gained more of a foothold with your product as it’s great for helping iron out kinks and solidifying the user experience.

#### **Summary**

In this Exercise, you’ve learned about the difference between Lean UX and Agile UX and how you can use each methodology at different stages in your process to continue improving your overall user experience.

#### **Resources**

* [Choosing a UX Strategy That Suits You and Your Users: Agile vs. Lean UX](https://www.justinmind.com/blog/choosing-a-ux-strategy-that-suits-you-and-your-users-agile-vs-lean-ux/)
* [Case Study - Lean UX at TheLadders](http://www.startuplessonslearned.com/2011/05/case-study-lean-ux-at-work.html)