

A decorative background element consisting of a series of horizontal, wavy teal lines of varying thicknesses, creating a rhythmic, wave-like pattern. The lines are set against a white background and are partially obscured by the text.

By Chuqing Dong

# Lean UX

Mindset &

Delivery.

# Lean UX - Mindset and Delivery

Politecnico di Milano

School of Design & Management Engineering

Product Service System Design

Management Engineering

Supervisor

**Raffaele Boiano**

Author

**Chuqing Dong**

Student Number

**873561**

Academic Year

**A.A 2019/2020**

# Abstract

With the Internet market growing more and more competitive, various IT enterprises are struggling to revamp or even re-build their product development processes. In the search of the best methodology, most seem to get puzzled by blur the different methodology buzzwords among different departments. This is reasonable, as even within the same single organization, UX design has its own work pattern varying from software development one. We could apply the same thinking, such as lean, on these departments, but the outputs differ from each other.

For example, when companies tend to adopt the Lean Software Development method, the development teams usually could find enormous successful case studies who have already been applying that. The methodology of lean regarding software development is relatively mature.

Turning to UX design, who plays an important role in software development, it is not suitable to adopt the lean software methodology entirely. But its development adventure and practices are of great value for us to explore the lean ux way. In fact, there has been several related publications and papers offering their own answers towards this from different perspectives. This is resulting from that UX design covers a vast array of areas, including interface design and usability. Or in another word, the work of UX department goes beyond their solo part, but even also involves the customers, the stakeholders outside the organization.

This paper will explore a possible lean ux development methodology, based on the agile development mechanism and agile UX. I wish to resolve the problem in an innovative way. More specifically, I attempt to contain the work skill, team management, organization collaboration and timeline perspectives into consideration.

Finally, enjoy your reading.

Keyword:

*Lean UX, UX Design, Agile Development, KOR Management, Lean Software Development Management*

# Abstract

A causa della crescita e dell'aumento di competitività del mercato internet, varie società IT stanno faticando a rinnovare o addirittura a ricostruire i loro processi di sviluppo del prodotto. Nella ricerca della metodologia migliore, la maggior parte delle compagnie sembra essere confusa dalla sfocatura delle diverse concetti della metodologia tra i diversi dipartimenti. Questo è ragionevole, poiché anche all'interno anche della stessa singola organizzazione, il design UX ha il proprio schema di lavoro che varia da quello di sviluppo software. Potremmo applicare lo stesso modo di pensare, come lean, a questi reparti, ma gli outputs differiscono l'uno dall'altro.

Ad esempio, quando le aziende tendono ad adottare il metodo di sviluppo Lean del software, i team di sviluppo solitamente possono trovare giganteschi casi studio di successo che l'hanno già applicato. La metodologia di lean riguardante lo sviluppo del software è relativamente matura.

Passando al design UX, che gioca un ruolo importante nello sviluppo del software, non è adatto ad adottare completamente la metodologia del software lean. Ma la sua avventura di sviluppo e le sue pratiche sono di grande valore per noi per esplorare il modo UX lean. Infatti, ci sono state diverse pubblicazioni e articoli correlati che offrono le proprie risposte a questo caso da diverse prospettive. Ciò è dovuto al fatto che il design UX copre

una vasta gamma di aree, tra cui progettazione dell'interfaccia e usabilità. In altre parole, il lavoro del dipartimento UX va oltre la loro parte solista, ma coinvolge anche i clienti, le parti interessate all'esterno dell'organizzazione.

Questo documento esplorerà una possibile metodologia di sviluppo lean UX, basata sull'agile meccanismo di sviluppo e agile UX. Desidero risolvere il problema in modo innovativo. Più in particolare, cerco di tenere in considerazione le capacità lavorative, la gestione del team, la collaborazione dell'organizzazione e le prospettive del timeline.

Buona Lettura

*Parola chiave:*

*Lean UX, UX Design, Agile Development, KOR Management, Lean Software Development Management*

# Table of Contents

## 1 Introduction

1.1 Subject Background	10
1.1.1 Gap between UX design and programming	10
1.1.2 Gap between narrow-concept UX design & research	13
1.2 Subject Purpose and Significance	14
1.3 Subject Content and Framework	16

## 2 Literature

2.1 Lean Thinking	18
2.1.1 Origin of Lean Thinking	18
2.1.2 Lean Principles	20
2.1.3 Lean Software Development	22
2.2 Agile development	25
2.2.1 Waterfall Method (plan-driven)	26
2.2.2 Agile Method	30
2.2.3 Scrum Process	34
2.2.4 Waterfall versus Agile	37
2.2.5 Lean and Agile	40
2.3 UX Design	43
2.3.1 Design Thinking	44
2.3.2 UX Design	48
2.3.3 UX Design Practice	51
2.3.4 UX Research	54

## 3 Research Method

3.1 Translate with Lean Operation technique	59
3.2 Learn from the book Lean UX	64
3.3 Case study - Autodesk Usability Investigation iteration	66
3.4 Case study – Spotify Matrix Management Structure	76
3.5 Case study – Google OKR framework	83

<b>4 Learning Interature</b>	
4.1 Context & Collaboration of Lean UX	94
4.2 Persona & Original Customer Journey Map	97
4.2.1 Personas	97
4.2.2 Original Customer Journey Map	100
4.3 Implementation of Lean UX	103
4.3.1 Pre-Phase (Preparation & Collaboration)	105
4.3.2 During-Phase (Deliverable & Test)	107
4.3.3 Post-Phase (Iteration & Management)	117
4.4 Conclusion	120

<b>5 Interview</b>	
5.1 With Han Li From Tencent Technology, China	124
5.2 With Haonan Li From Baidu Technology, China	128

<b>6 Conclusion Envolvement</b>	
6.1 Strength & Limitation	133
6.2 Future evolvement	135

## **7 Reference**



# 01

# Introduction



# 1.1 Subject Background

## 1.1.1 Gap between UX design and programming

The project development team usually consist of analysts, designers, programmers. There are two kinds of team structures, one consists of staff within one functional organization, another one is made up of members from many different functional organizations. There is increasing number of IT companies choose the later kind of structure, because the matrix structure is more flexible and has an ideal collaboration.

Let's see what happened when a company chose to adopt a Lean development process for new products.

On one hand, development team, including the programmers, product managers and quality analysts could adopt the "Lean Software Development" which has been relatively mature. This paper will introduce the method in detail in the following section. One of the most popular processes resulting from it is the Scrum Methodology. Roles involving in Scrum are product owner, scrum coach and development team, without ux designers. In the Organization the UX team is separated from development in order to have the corporate design under control.

On the other hand, UX design, as an important participator in the development flow, is quite different from other participators: not only ask for efficiency, but also put a high requirement to users' feedback, sensible design details, and enormous iterations. These features define that the UX design must be treated separately among the whole agile development, if we want to maintain the same fate within this single development flow.

Practically speaking, programmers would begin coding work once the project is set, even before product owner specifying features totally. And UX designers need to be able to catch design failures, iterate the design proposals as many times as needed, while dev team complete only one list of backlogs for one time in the same single period. So, the project approach for dev team is not feasible for ux team. They are in two separate but parallel tracks. [Adapting] As a consequent, it's of significance to develop a Lean UX process following the same thinking as Lean Software Development but different in practice.

In general, most IT dev teams are perfect in development project, while weak in the management of the UX design and dev team.

For example, at the very start, it's the product owner and UX team that defines user stories, product features, and the backlog. So, the dev team might feel frustrated and limited in terms of creativity. Because in this way, the only touchpoint between the dev team and the product is the coding. This results in that these programmers are not empowered to devote into the product.

Another phenomenon is, sometimes the UX team and dev team have different opinions on the final surface and experience design. The product owner is in awkward when both sides are waiting for his last word about the UX. The dev team debate out of passion while UX team out of expert identity.

From above, we could say that the current scrum method is limitedly appropriate in the separate development process and management, while the UX design is not involved appropriately. On the other side, this proves there is opportunity for improvement.

## 1.1.2 Gap between narrow-concept UX design & research

Enterprises usually hold expectations on UX design to dig the real-time customer needs, so as to bring the correct product features to market as early as possible. UX design experts have developed an authoritative and ideal framework for UX design from other design fields (explicit framework will be introduced in Sec.2.3). However, in practice, the UX design work is forced to eliminate a few work steps, or to express the research steps for time-saving.

In this paper, we will pay more attention on the analysis of some Chinese IT companies. Let's take Tencent as an example. WeChat is one of the most important and popular products made by Tencent. Its product manager leader-Allen Zhang has ever said in public that demands come from your knowledge of customers, rather than customer research, nor analysis, nor discussion. As a consequence, Tencent company pays more attention on product & design work rather than the pure customer research works. Even though the workflow does not have a negative effect on the Webchat's success, in fact it satisfies the requirement on discovering demands by other human resource, such product managers, and disrupts the planned efficient UX design workflow.

# 1.2 Subject Purpose and Significance

## 1.2 Subject Purpose and Significance

While there is such a high demand for UX design to catch up with development process, it's meant that in the UX field, we have had to examine and adapt our practices to stay in tune. Even though there are great challenges in incorporating UX design into dev process under Lean Thinking, there are also tremendous benefits following.

But why it's the Lean Thinking that we choose to follow? First of all, Lean is proved to be the most efficient management thinking till now, so it has become most popular adopted by most IT companies. Besides, there are many existing skills and resources on the lean software development. Although it's not suitable to apply the same lean software development process on UX, but now that they are applied in the same scenario, there may be useful skills from the dev methods that we could employ to answer the Lean UX question.

Then how to apply Lean thinking, a philosophy from manufacturing? It's true that Lean thinking previously is used in Operation Management for manufacturing, and the classical tools are tailored for the manufacturing process, such as the typical value stream map. However, "If lean is thought of as a set of principles rather than practices, then applying lean concepts to product development and software engineering makes more sense and can lead to process and quality improvements. "[lean software development tutorial]. The purpose of this subject is, in one word, trying to provide a feasible Lean UX Methodology to meet the IT companies' need for rapid development.

# 1.3 Subject Content and Framework

This article is structured as follows:

Sect. 2 introduces the literal work, especially the basic concepts, including Lean production, Agile development process and customer-centered interaction design.

Sect. 3 lists the research methods and specific research process used for the thesis, including the re-interpretation of Lean theory, re-interpretation from the book Lean UX, and several case studies which closely related to the methodology.

Sect. 4 presents the learnings and experiences in the format of top-down lean-based ux design process, followed by an assumption for the future evolvement.

Sect. 5 wraps up the practices from 2 interviews with senior UX designers and PMs who are on the job. Due to lack of resource on real project experiment, it's nearly no possible to experiment the Lean UX methodology theory by author. Instead, 2 senior stakeholders are invited to estimate the theory via interviews.

Sect. 6 is the conclusion of strength and limitation, as well as the expectation for future evolvement.



# 02

# Literature



# 2.1 Lean Thinking

## 2.1.1 Origin of Lean Thinking

Lean techniques have been dramatically applied to manufacturing industry and increased its success for over 50 years. It's derived from the Japanese manufacturing industry. More specifically, it's from Toyota Production System (TPS). TPS was widely referred to, in the 1980s as just-in-time manufacturing, but now contains many other sources. The core, just-in-time is a methodology aimed primarily at reducing times within production system, as well as response times from suppliers and to customers. The system's creator, Mr. Taiichi Ohno introduced his inspiration was from Henry Ford's management.

More specifically, Toyota's techniques result in 3 main advantages: reduce in-process and final inventories, expand worker responsibility to operate more machines, and increase production quality. The key of practice is that Toyota create Kanban as the information transmission tool. This reverses the flow of information signals controlling production operations, from the previous plan-determined push strategy to current feedback-determined pull strategy.

Lean is regarded by many as a generalization of TPS. Simply speaking, Lean's focus is upon improving the "flow" or smoothness of work, thereby steadily eliminating mura ("unevenness") through the system and not upon 'waste reduction' per se, according to Hoseus et al.(2008). Techniques to improve flow include production leveling, "pull" production (by means of kanban) and the Heijunka box.

But in the early period it was quite common that entrepreneurs copy the basic application of the lean tools, without building the completed Lean management mechanism, in return blamed Lean for not living up to its reputation. Lean is a fundamentally guidance rather than a collection of instant improvement techniques. Besides, Lean has a toolkit of recommended practices that to choose from. More importantly, Lean requires more persistence and mindset shifting.

Comparing to Lean manufacturing, Lean software development is much younger. Only in the last few years, it has been applied to this field, but has been developing in a surprisingly high speed.

## 2.1.2 Lean Principles

Lean thinking changes the focus of management from optimizing separated technologies, assets, and vertical departments, to optimizing the flow of products and services through entire value streams that flow horizontally across technologies, assets, and departments to customers. [WHAT IS LEAN?] As we introduced before, Lean Thinking does not consist of specific tools but flexible principles which could be employed in detailed, customized scenarios. So what are the principles?

It's the 7 Lean Principles that support this transformation. Besides, there is another version of 5 Lean Principles, which was described by James P. Womack and Daniel T. Jones in 1997. The 5 principles are Value, Value Stream, Flow, Pull and Perfection. According to J&D's theory, they build on each other, and form a continuous but improving loop in logic. When we employ some lean tools, it's much simpler if we follow the 5 principles. For example, to redesign the value stream map in manufacturing, usually managers would explore the new proposal from these 5 perspectives in sequence.

The difference between 5 & 7 principles is, the former one is more process-oriented, while the later one is people-oriented. But these 2 kinds of principles are similar in essence. In this paper, for the account of either software development or UX design is people-oriented activity, this paper will give emphasis to the 7 principles.

## Eliminate Waste

Anything that does not add value to the customer is defined as waste. The typical wastes are including overproduction, unnecessary transportation, inventory, unnecessary movement, defects, over processing, waiting.

## Deliver Fast

Think about what slows down the delivery, then eliminate it.

## Defer Commitment

To defer commitment means to defer making decision without necessary data to support. This Lean principle encourage to keep options open and continue collecting enough information.

## Respect People

Encourage to communicate more proactively and effectively, and empower each other, with respect.

## Optimize the Whole

Optimize single touchpoint considering the dual influence with the upstream and downstream, thus optimize them if necessary.

## Build Quality In

Avoid creating defects rather than focus on fixing defects.

## Amplify Learning

Create Knowledge.  
Retain valuable learning, encourage and embrace involvement.

## 2.1.3 Lean Software Development

Lean Software Development concept was put forward in a book written by Mary Poppendieck and Tom Poppendieck for the first time. In this book, the Poppendiecks interpreted the 7 traditional lean principles into software development projects in a new way, as well as put forward a set of lean tools. With the Poppendiecks' efforts on promotion, Lean Software Development has been more widely accepted and applied by the development world.

The inspiration for the Poppendiecks to apply lean to software development is that, Michael and Richard Selby noted a similarity in the philosophy behind Microsoft's daily builds and Toyota's JIT production.

## Toyota-style Lean Production

manual demand-pull with Kanban cards

1. JIT 'small lot' production
2. Minimal in-process inventories
3. Geographic concentration-production
4. Production leveling
5. Rapid setup
6. Machine and line rationalization
7. Work standardation
8. Foolproof automation devices
9. Multiskilled workers
10. Selected use of automation
11. Continuous improvement

## 1990s Microsoft-style Agile development

daily builds with evolving features

1. Development by small-scale features
2. Short cycles and milestone intervals
3. Geographic concentration-development
4. Scheduling by features and milestones
5. Automated build tools and quick tests
6. Focus on small multifunctional teams
7. Design, coding and testing standards
8. Builds and continuous integration testing
9. Overlapping responsibilities
10. Computer-aided tools, but no code generators
11. Postmortems, process evolution

The pre-form of Lean Software Development explored the systematic reuse of product platforms and major components, as well as short, overlapping phases to reduce project time and engineering hours. Then, developed based on that, the final theory also emphasized eliminating waste and bureaucracy in product development, encouraged learning through short cycles and frequent builds, and promoted late changes and fast iterations.







## **2.2 Agile development**

Agile software development is an approach to software development, refers to a group of methodologies based on iterative development. It concerns more the specific practice of developing software and the project management. This paper will interpret regarding methodologies and its development as following.

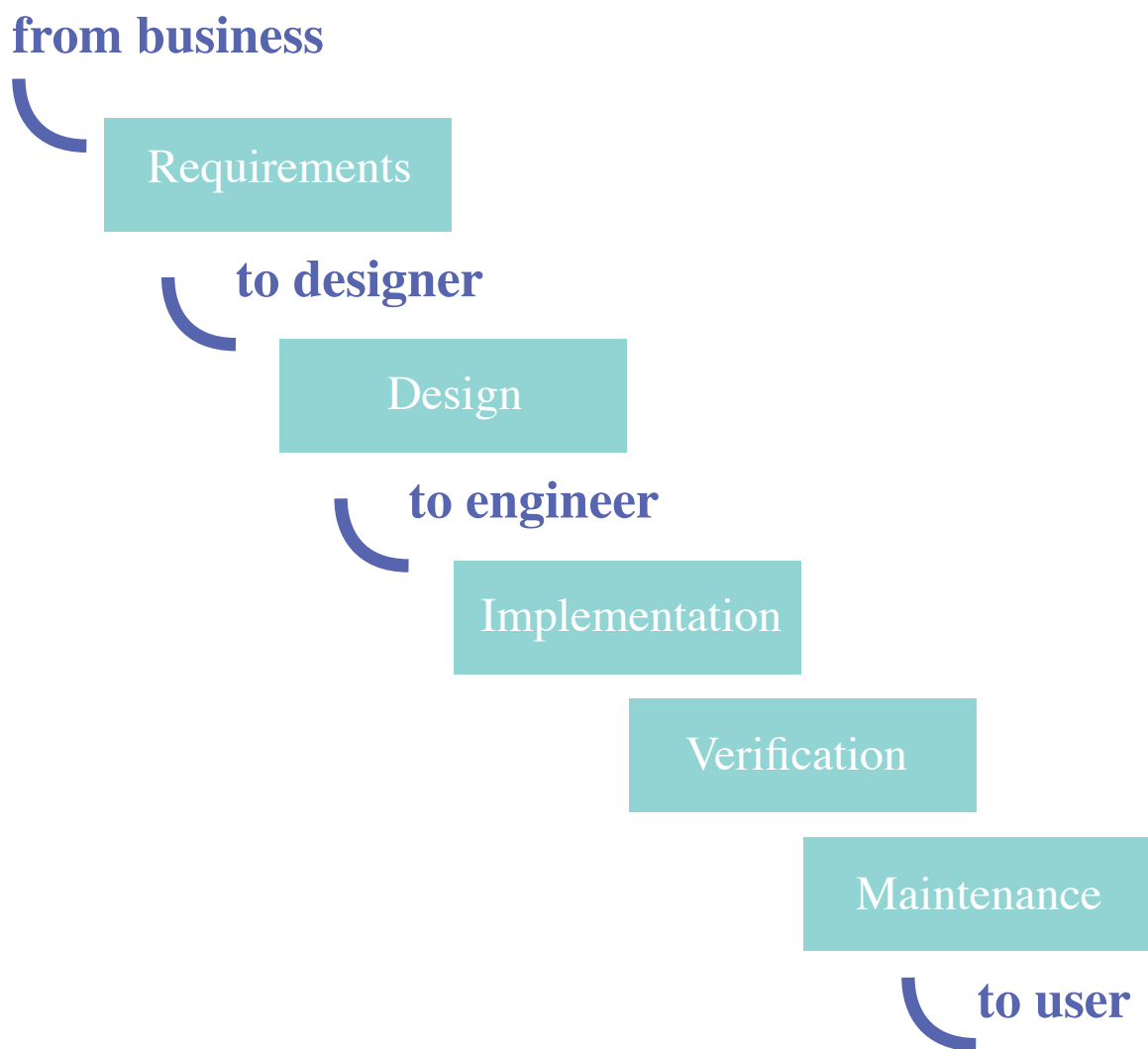
## 2.2.1 Waterfall Method (plan-driven)

There are 2 software development methods of the mainstream, Waterfall and Agile Methods.

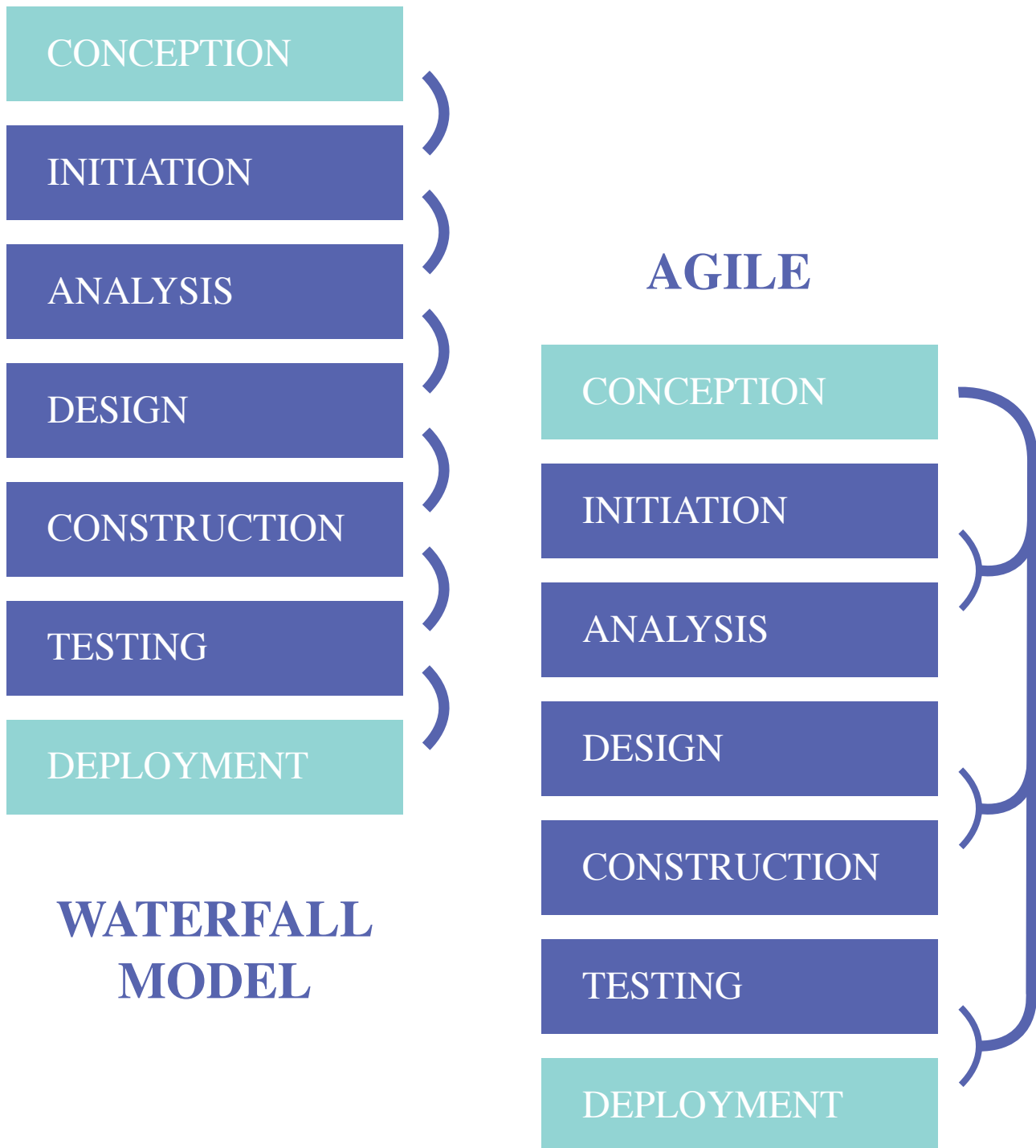
Earlier than Agile method, in 1970 Waterfall method was introduced by Winston W. Royce. Waterfall model is another software development approach, of which several development phases is seen as a waterfall, flowing downwards. The graph below shows the typical sequential phases.

In a waterfall project, each stage will not begin unless the former one ends. There is a stage gate between two stages, in a traditional linear way. Waterfall is much suitable for those big projects where change is uncommon. Within it, the upfront requirements are usually defined clearly. Moreover, a strict waterfall approach discourages revisiting and revising any prior phase once it is completed.

In general, waterfall model has been gradually but broadly superseded by more flexible software development methodologies, such as Agile. But both waterfall and agile method are usable, mature methodologies, having been employed in software development projects for a long time. The reference depends on the project situations where they are applied. We would say either of these is great rather than saying one is much better than another.



*Figure 2.2.1-1 The overview process of waterfall methodology*



*Figure 2.2.1-2 Comparison between Waterfall and Agile*

Let's compare the pros and cons of Waterfall method:

Pros:

1. Planning and designing more straightforward;
2. It's easier to assess the progress;
3. It's not mandatory that all functions and customers present during the whole process;
4. The software can be designed completely and more carefully.

Cons:

1. It costs plenty of time gathering the customer requirement to pre-determine;
2. There is a risk that customers are not satisfied with the output.

In general, the root cause of Lean advantage comes from the advanced well-defined planning of the projects, concrete benefits are resulted from the butterfly effect. First of all, a well-defined project would be cheaper, due to upfront planning allows predicting flaws and adjust as soon as possible, which save the cost of changing in late phases; and it is easier to arrange specific skill sets according to existing project plan and schedule.

From a entire practical point of view, a waterfall approach is acceptable and appealing. Most people like the idea of planning, and feel nervous about jumping in without proper planning, which is what waterfall offers.

## 2.2.2 Agile Method

The official origin of Agile development could be back to 2001, when 17 software developers together published the Manifesto for Agile Software Development. Agile refers to a toolkit of practices and principles imbedded in the Agile Manifesto. At that time, the Manifesto was a creative disruption against popular heavyweight methodologies.

The Agile Manifesto's values are:

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

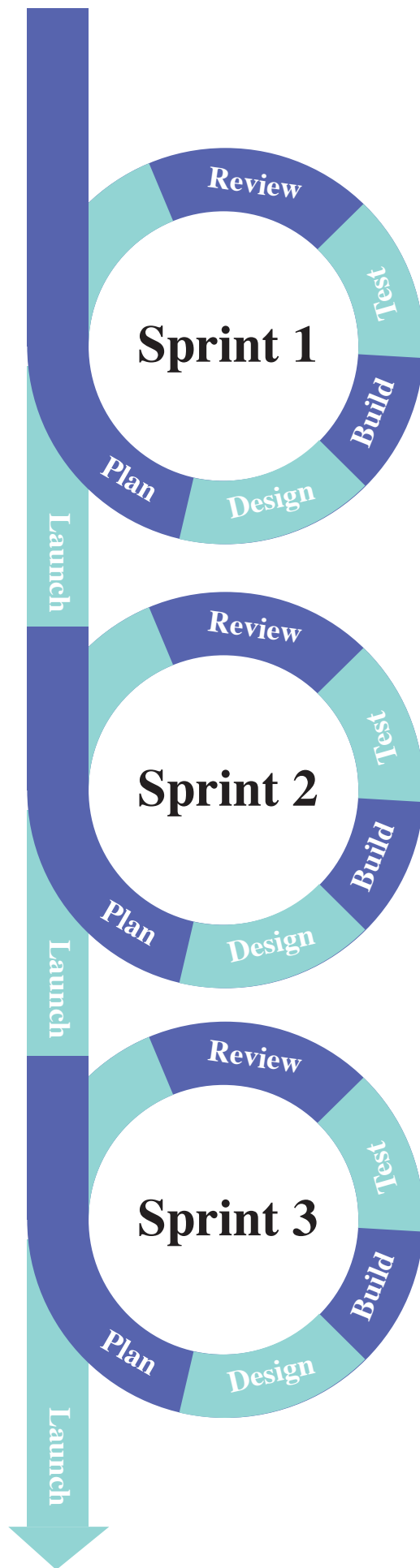
Its principles are:

1. Highest priority is customer satisfaction
2. Welcome changing requirements
3. Frequent delivery of software
4. Business people & developers cooperating daily
5. Build projects around motivated people
6. Face-to-face conversation is best
7. Progress measured by working software
8. Sustainable development pace
9. Continuous attention to technical excellence
10. Simplicity
11. Self-organizing teams
12. Regular reflection & adaptation

The minimum work unit of Agile development is cross-functional team. Unlike the waterfall model, the cross-functional team is not required to meet the final ideal purpose with one-time effort. In contrary, the giant development process is broken down into several periods. This ought-to-be development purpose is unreliable under Agile model. Development team is encouraged to develop step by step, guided by themselves, and adapt period purposes in real-time.

Based on that, the project would deploy a set of sprints (the professional expression of the periods), which are the duration-defined phases instead of tasks or schedules. Within each separate sprint, the agile team is ought to complete the planned work according to some priority. Sometimes the work is prioritized by business value that could be perceived by customers. Sometimes the work is mixed with the accumulated uncompleted work from last sprint. The united work for the team would be translated into a running list of more detailed deliverables.

Agile is a framework filled with a broad range of development methods, covering the various phases of the software development cycle, from practices, workflow management to supporting activities.



*Figure 2.2.2 The process of Agile methodology*



The main popular agile development methods include:

Adaptive software development (ASD)	Feature-driven development (FDD)
Agile modeling	Lean software development
Agile unified process (AUP)	Kanban
Disciplined agile delivery	Rapid application development (RAD)
Dynamic systems development method (DSDM)	Scrum
Extreme programming (XP)	Scrum ban

The paper will introduce in detail the most popular method among those - Scrum in the following section.

Let's compare the pros and cons of Agile method:

Pros:

1. The customers are involved in the project earlier and have some voice;
2. Frequently release small-improved version to modify the deadline pressure;
3. Allow changes;
4. High quality.

Cons:

1. High requirement on the customer involvement;
2. High requirement on team member dedication and co-location;
3. May lead to frequent refactoring if not fully consider.

## 2.2.3 Scrum Process

Scrum is a specific software development process, a variant of the Agile method. It emphasizes the self-organizing principle by encouraging physical co-location and inner-team communication.

There are 3 roles in the Scrum Process: product owner, scrum coach and the important development team. The product owner is responsible for the whole team, representing for customers, thus defining the product features. The scrum coach is the bridge between product owner and dev team, coordinate their task conflict. This role should not engage in the specific dev operation or the product definition, just coordination. The dev team usually is responsible for carrying out the product delivering, like analyzing, coding and testing.

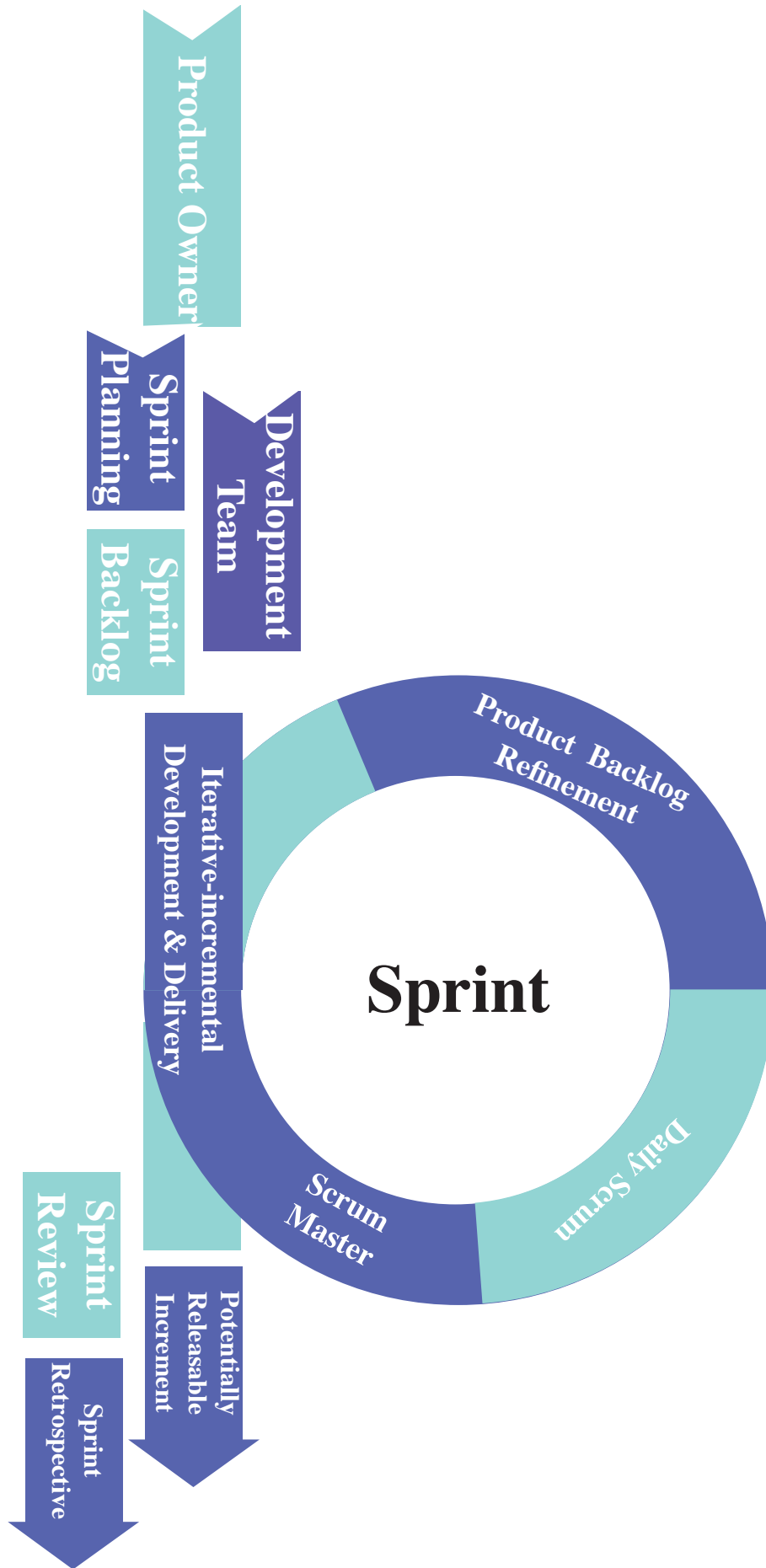


Figure 2.2.3 The overview process of Scrum development

The typical workflow of Scrum includes:

### Backlog Definition:

Product Owner gathers the user stories from other channels, especially customer research team. The whole development project is divided into several sprints, each of which usually last for 2-3 weeks. In the beginning of each sprint, product owner priors the user stories and translate into sprint backlog.

### Daily Scrum:

The key is the scrum standup meeting, where every team member is present theoretically. This meeting encourages team members to communicate with each other, on their current progress and what issues they meet to reach the final goal. This meeting is useful for the members to keep up with the whole team's pace, coordinate the resource among them towards the goal.

### Scrum Review:

After completing a sprint, the scrum team is encourage to review the project. It is very significant for a team form the habit of continuous learning. The sprint project becomes more than just a development project, but also an opportunity for team members to improve themselves. They could conclude experience from both success moments and difficult tasks.

## 2.2.4 Waterfall versus Agile

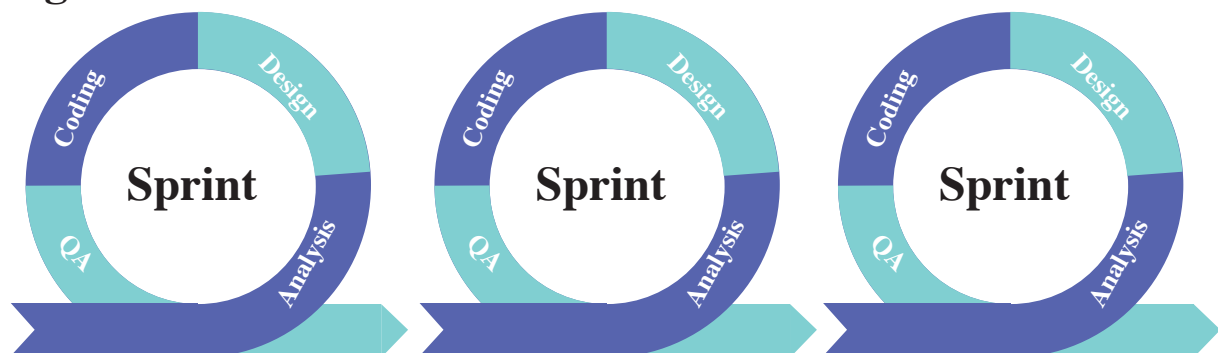
So what is the correct methodology for the company to adapt? Is it surely the newest one?

In fact, according to one 2015 survey, only 2% of companies still operate using purely traditional Waterfall practices, which means most companies do choose the newest one. In another word, Agile development has almost completely taken over software development. In short, Agile is everywhere.

### Waterfall



### Agile



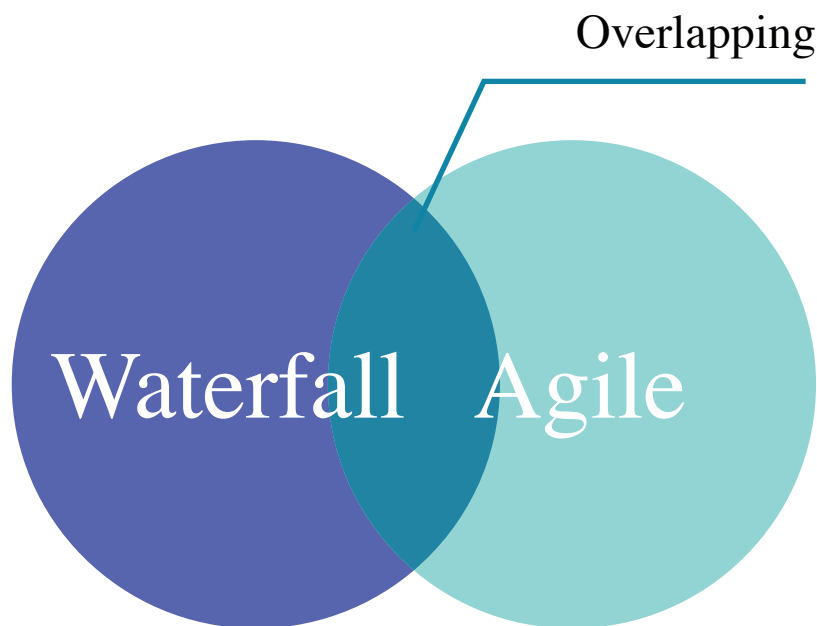
*Figure 2.2.4-1 Comparison between Waterfall and Agile*

The graph above states the difference between Waterfall and Agile:

In a Waterfall development cycle, analysis, design, coding and quality assurance testing (QA testing) are separate stages of a software release in sequence.

In Agile development, “each of a set of incremental mini-releases has these stages.”  
Adapted from Cutter Consortium.

One article by Smart Data Collective states that “Agile is not right for every project team and is absolutely not a silver bullet that will solve your organization’s delivery problems. In fact, if you are already struggling, trying to change to a new methodology might make things worse.”



*Figure 2.2.4-2 Connection of Waterfall and Agile*

So, when should the organization use Waterfall methodology?

1. If the company has a clear plan or forecast of final product.
2. If it is difficult to switch different sizes of the project team during the development period.
3. If release speed does not matter too much.

When should the organization use Agile methodology?

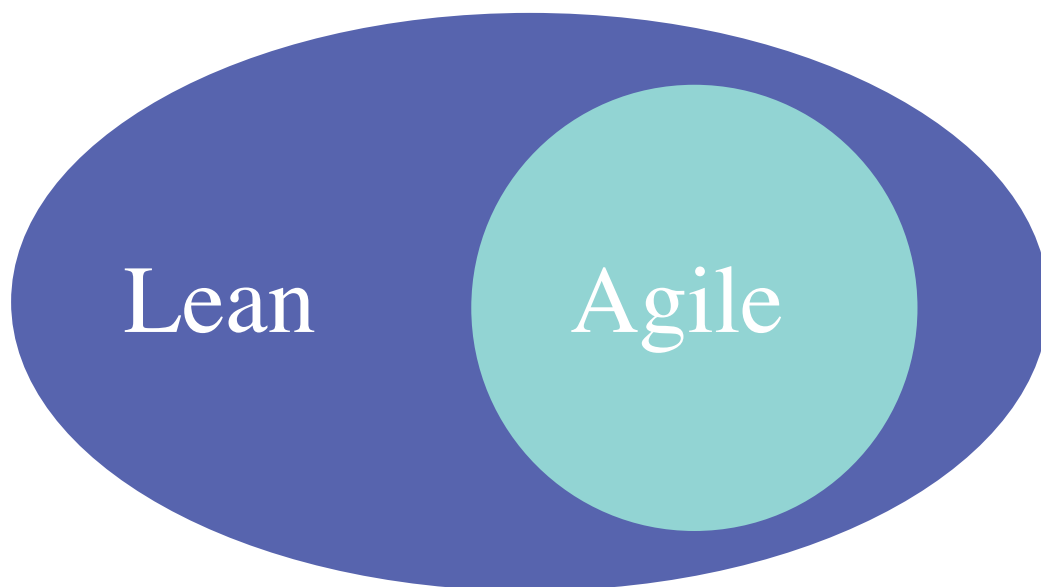
1. If the company is keen to release product.
2. If the project team is able to switch its size after project began.
3. If the developers are able to self-organize.
4. If the product has the potential to adjust rapidly

Agile method raises requirements on the whole team, not just those directly responsible for development. There also should be some flexibility for the requirements to change potentially and incrementally.

## 2.2.5 Lean and Agile

“In any case, it can be safely said that Lean views all Agile methodologies as valid supporting practices.” adapted from *The Art of Lean Software Development*. There is another article states that Agile’s values & principles work because of the science behind Lean, and so there are several similar themes repeated in lean and agile.

Let’s see the underlying connection between Agile and Lean



*Figure 2.2.5 Subordination of Lean and Agile*



1) Lean is usually employed in any scope, from the management strategy by high-level, to the specific practice by basic-level; from the workflow adjustment inside company, to the collaboration improvement outside company. Agile is a collection of the filtered practices, compared to a simplified version of Lean. Its scope is limited in the developing software and the project management, all surrounding the software development.

2) The expansion of Lean within one company is similar to the wildfire: starting out small then expanding, and benefiting more and more in the process. The Agile is one torch comparing to the whole fire, the starting point of the wildfire. It empowers the following change, preparing for the potential benefits.

3) The focus of Lean is, originating from the lean manufacturing, to eliminate waste and add value for the customers. While the focus of Agile is rapid delivery of software and customer engagement for iteration. The focus of Agile could be understood as the mapping result of Lean on smaller scope.

In essence, Agile is about being flexible and responding rapidly to changes in feedback, requirements, and market needs. In order to do that, Lean thinking is necessary, as its principles help promote the mentality needed.

The Agile Software Development has been developing for long before it is recognized the similarity and relationship with Lean. And during previous evolvement, Agile form its representative toolkit which consist of set of methods, like Scrum. Gradually some Lean tools are absorbed into Agile, such as Kanban. As a consequence, the similarity of Agile and Lean, in terms of tools, is more and more apparent. In fact, in order to apply Lean software management, project teams tend to start out from Agile. Based on the Agile methods, they continue to apply other Lean tools so as to pursue the expected development goal.



## 2.3 UX Design

Before we interpret the new Lean UX process, let's explain why we spend so much efforts involving UX design into software development, and introduce the current UX design methods in detail. This chapter is important, because only company understands the significance of UX design, would the whole company be empowered to adopt Lean mindset, to accept and support the Lean UX design.

## 2.3.1 Design Thinking

“Design thinking can be described as a discipline that uses the designer’s sensibility and methods to match people’s needs with what is technologically feasible and what a viable business strategy can convert into customer value and market opportunity.”

– Tim Brown CEO, IDEO

When companies seek for the techniques helpful to acquire competitive advantages, the C-level executives are always blurred by the Design-thinking norm. They know that Design-thinking is adopted by some of the world’s leading brands, like Google, Apple and Facebook. Design thinking is also taught at some leading universities, like Harvard, Stanford. But executives are still confused about what the concrete benefit design-thinking could bring in and how to employ in their specific operation.

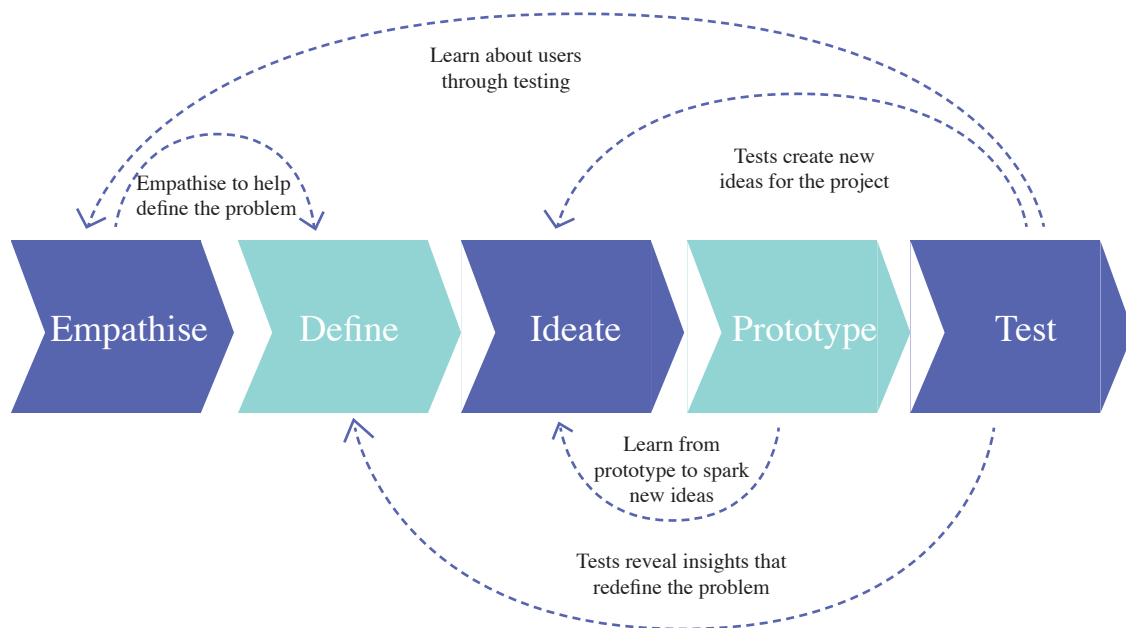
Earlier than IT era, the approach to get the competitive advantage is related to physical product, such as competitive price by production in scale, unique market position by different product definition or promotion, or focus on segment markets, adapted from Porter’s generic competitive strategies. But situation has changed when customers could gain stronger bargaining power with IT development: they have more access to market through Internet. Enterprises therefore turns their focus on how to grasp the customers’ attention. Only in this way, brands could have continuous lifespan in customers’ minds. Design thinking is able to playing an important role towards this goal.

Design thinking stems from design process, used to explore innovative design proposals, including 4 basic principles:

- resolve ill-defined or ‘wicked’ problems
- adopt solution-focused strategies
- use abductive/productive reasoning
- employ non-verbal, graphic/spatial modelling media

Gradually, Design thinking is applied in other process which in need of innovative capabilities, like business process, computer science and even education. The growing design thinking is based on design field practice, besides there are several adjustments and inheritances. Design thinking is a solution-based approach to solving problems in both design and other fields. It still revolves around a deep understanding or empathy of customers. Standing in customers’ shoes, enterprises could uncover the hidden customer need before their competitors do. In the long term, these enterprises would accumulate the competitive advantage gap over others.

The design thinking process could be extracted as simple 5 steps, as shown in figure 2.3.1



*Figure 2.3.1 The non-linear process of design thinking*

Adopted from: <https://www.interaction-design.org/literature/article/5-stages-in-the-design-thinking-process>

However, it should be noted that these 5 steps is not merely happens in sequence, they are actually in a loop, or a system of overlapping steps. In real practice, it is for sure not enough to only apply these 7 steps for one time in order to discover the best solution. These steps sometimes occur simultaneously or in a linear, and the whole loop repeats with more ideas created.

“The most secure source of new ideas that have true competitive advantage, and hence, higher margins, is customers’ unarticulated needs,” says Jeanne Liedtka. Design thinking engage the customers in the innovation process, so that extract the the true need and insights from the real user experience, which is more valuable than market data or innovators’ imagination.

However, there are some drawbacks of design thinking: it over-simplifies the design process and underestimates the job of technical knowledge and skills.

## 2.3.2 UX Design

“No product is an island. A product is more than the product. It is a cohesive, integrated set of experiences. Think through all of the stages of a product or service – from initial intentions through final reflections, from first usage to help, service, and maintenance. Make them all work together seamlessly.”

— Don Norman, inventor of the term “User Experience”

UX (User Experience) design involves a broad range of expertise knowledges, like visual design, psychology, coding and interaction design. People often misunderstand UX design with UI design and Usability Design. In fact UX design goes beyond those two, both two are merely constituent elements of UX design. Except for those, UX design covers even branding and function definition. In general, UX design is responsible for the entire process of acquiring and integrating a product (including software or application product).

In conclusion, UX design in theory is a scientific process, encompassing any kind of interaction between an end-user and a company. However, the term UX design is more and more used in IT fields, despite is in nature a professional user-centered term. One possible explanation is that the digital industry started blowing up around the time of the term’s invention.



So what on earth does the UX design do? The jobs of UX design include customer and market research, wireframe and prototype, execution and collaboration. In another word, the UX role is multi-faceted as part marketer, part designer and part project manager. Among these jobs, the most important part is analysis or test, because this process would run at least twice. Through the process of testing and refinement, UX design team ultimately gain the balance point between user's needs and business goal.

### Customer and Market Research

- Competitor Analysis
- Customer Analysis
- Product Structure/Strategy
- Content Development

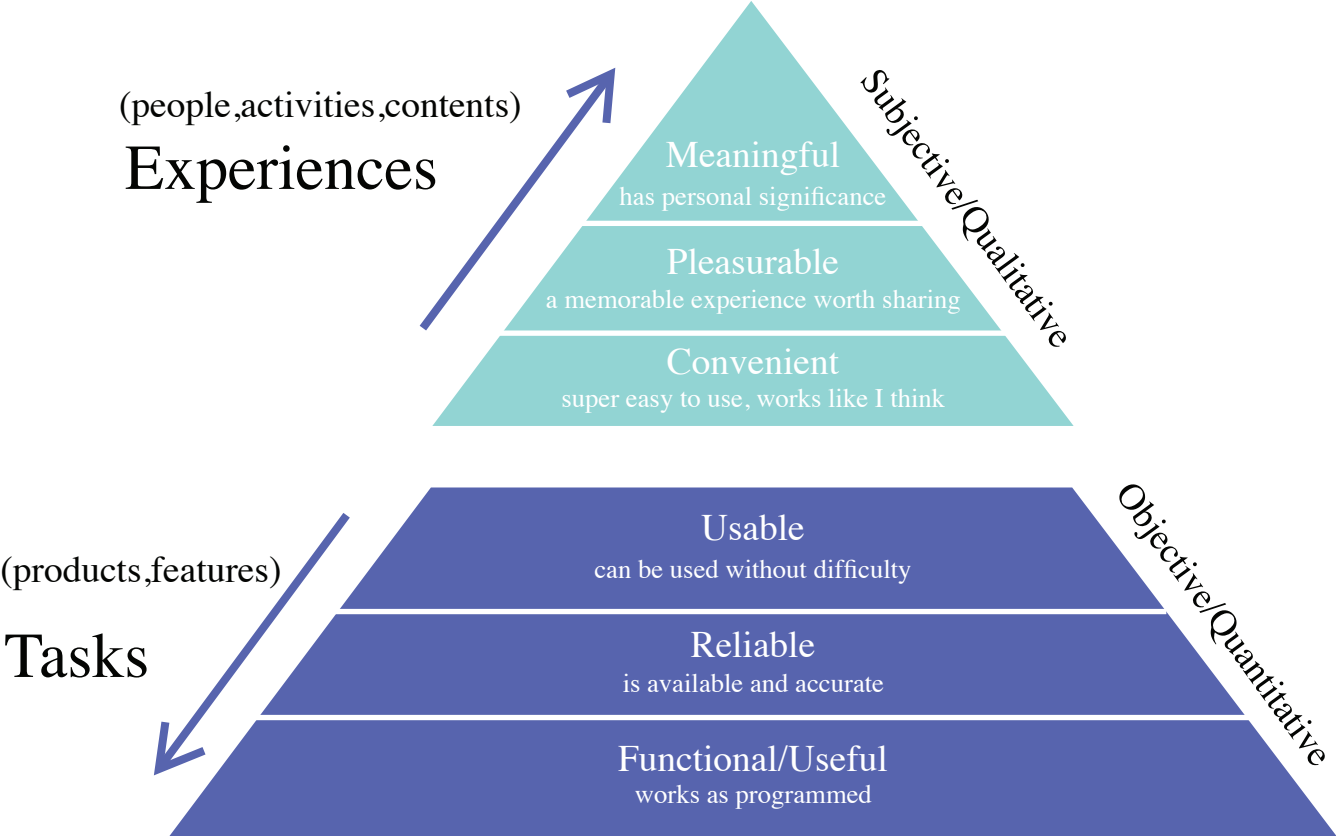
### Wireframe and Prototype

- Wireframing
- Prototyping
- Testing/Iteration
- Development Planning

### Execution and Collaboration

- Coordination with UI Designer(s)
- Coordination with Developer(s)
- Tracking Goals and Integration
- Analysis and Iteration

In addition, the following graph will give introduction on the assessment of UX work, based on Maslow’s hierarchy of needs. Because the UX involves various arrays of work, it is difficult to find ways to assess UX effort and track progress, while the UX pyramid would be a useful tool in this case.



*Figure 2.3.2 The UX pyramid*

Adopted from <https://blog.marvelapp.com/introduction-user-experience-design/>

### 2.3.3 UX Design Practice

In practice, the UX design doesn't have a fixed workflow but it provides a colorful toolkit including various tools. These tools actually are the tasks that designers should complete in order to acquire corresponding goals. UX designers have the full authority to pick and arrange these tools depending on the specific project context.

Discovery & User Testing	Strategy & Ideation	Content & IA	Interaction Design
Competitor analysis Stakeholder interviews Contextual Enquiry Surveys Daily study User interviews Heliristic review User testing Personas A/B or split testing Accessibility testing	Affinity diagramming Use cases Scenarios Mental models Experience map Workflow diagram	Content audit Sitemap Card sorting	Storyboards Wireframe & Diagram Paper prototype Design prototype

*Figure 2.3.3-1 Tool list used in UX design practice*

For example, to design a complex bank app, the UX designers might focus on the information architecture organization, so conduct the mix of all the 3 IA tools and some from the other 3 categories. The logic of arrangement is familiar no matter what project is: from ideation to execution with testing before submitting the final output, with some iteration in the middle.

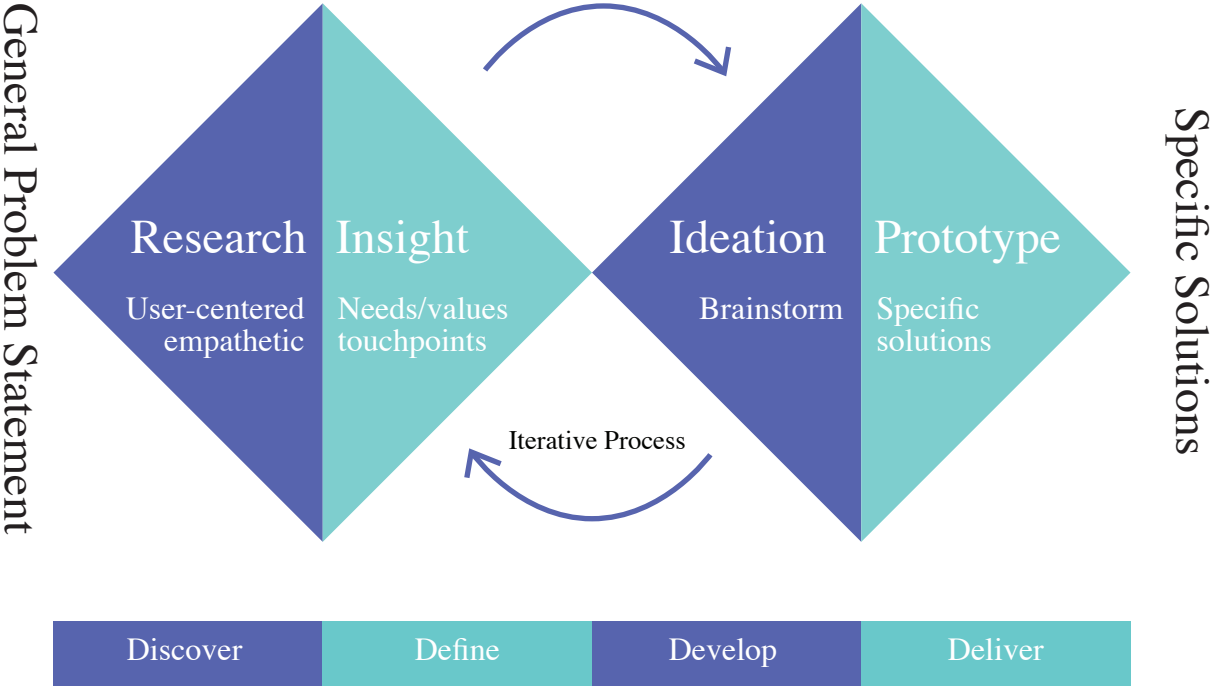
Let's take a birdview of workflow of UX design within the whole project. First of all, let's understand the context. As the last chapter described, there are two main project methodologies, Waterfall and Agile (Agile is regarded as the practice of Lean), where UX design would be employed in

both. For UX designers, there is no need to figure out exactly all the details of these different project methodologies, but it is important to know whether you are following a traditional linear workflow(waterfall) or flexible iterative one. So that UX designers would plan their work to collaborate with other project roles.

In the waterfall workflow projects, each person must complete their individual task before passing down to the next person. In this case, the UX designers more or less have sufficient schedule to complete their job, from user research, ideation, testing till final output, all of which should be done as a bunch.

Under Agile workflow, let's just concentrate on the Scrum Development as an example. Before the Scrum Sprints begin, product owner always discusses with UX designer or the User researcher to define the user stories, and they together to break the whole project into several sprints. Then within each sprint, the product owner performs the traditional Scrum Operation, including backlog definition, the standing up meeting, etc. Usually the UX designers complete the design sprint simultaneously at the same pace of the coding work, but UX job content is at least one sprint earlier than coding one. A important reason is that the speed of coding is much faster than UX design in fact, so UX design prepares one extra sprint time to test and make responses to change.

In detail, the design sprint looks the same as the mini design process applying design-thinking:



*Figure 2.3.3-2 Double Diamond Design Process*

Please review the last section(UX Design) if would like to understand better.

## 2.3.4 UX Research

“User research focuses on understanding user behaviors, needs, and motivations through observation techniques, task analysis, and other feedback methodologies.”

– Explained by Wikipedia

Comparing to user research, UX research is slightly different, for UX research does not necessarily assume an iterative process. User research concentrates on a more general level, discovering or validating customers’ real needs, or improving it. This is very significant cause it determines and adjust the product’s insight continuously. While UX research refers to one specific process of the whole product development. The purpose of UX research is to add context and insight to the design process. But generally speaking, UX research and user research are often used interchangeably. One possible explanation is that most of the research tools and designs of UX research are inherited from user research.

Either UX research or user research is almost translated from other forms of research directly. This may explain why there are so many same research techniques as other fields’, such as questionnaires and interviews, in academics, scientists, market researchers, and others.

But why UX design put an emphasis on research? Looking at those successful softwares or applications, what do they have in common? The answer is “they cater to their users”. In product design, there is one normal but dangerous trap that, the development team, including the product owner realize that they are developing the product for only one user-themselves after a long time of development. A potential terrible result is the whole dev team is trapped in the faked customer need, provide some features that customers do not want. In order to prevent this risk, it is well accepted to inform the design process from the perspective of the end user through UX research and usability test. It’s research that leads us to define the correct user and need, as well as in what context that they’ll use this product.

The specific process of UX research is:

- 1) Define the research problem clearly.
- 2) Then select the right techniques from the research toolkit, divided into qualitative ones (e.g., ethnographic studies, scenarios, personas, focus groups, prototyping), and quantitative ones (e.g., surveys, eye tracking, controlled laboratory or field testing).

Card Sorting

Expert Review

Eye Movement Trackment

Field Studies

Usability Testing

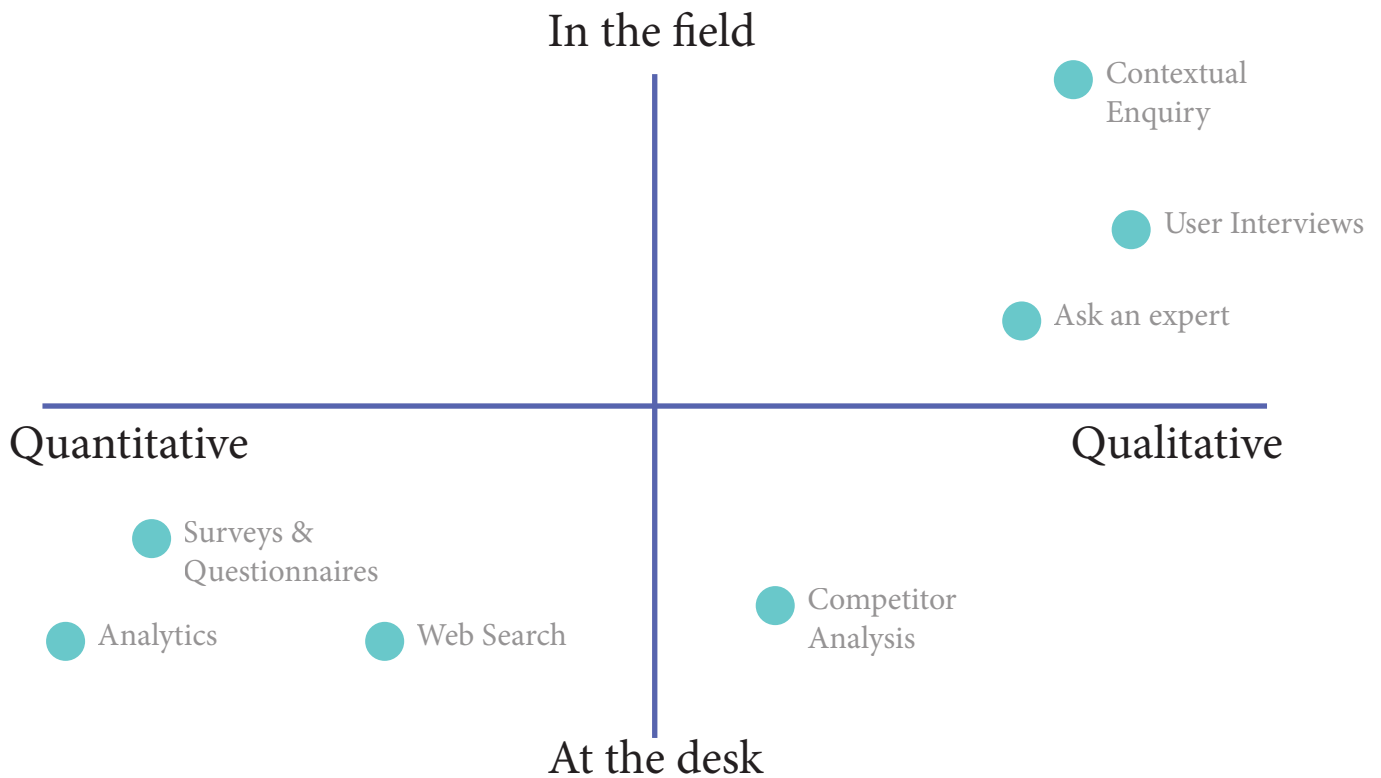
Contextual enquiry

User interviews

Surveys & questionnaires

Web Search

Competitor Analysis



*Figure 2.3.4 Coordinate of research techniques*

3) Finally document and share your findings by means of:

Personas

Customer journey map

Service blueprint

Offering map

System map



# 03

## Research Method



There are already developed theories about Lean and UX, and even Lean UX. The thesis respects the existing theories and learns from some of them.

Simultaneously, in order to get more inspirations, it is necessary to review the root Lean theory and techniques, deeply understand the impact of Lean.

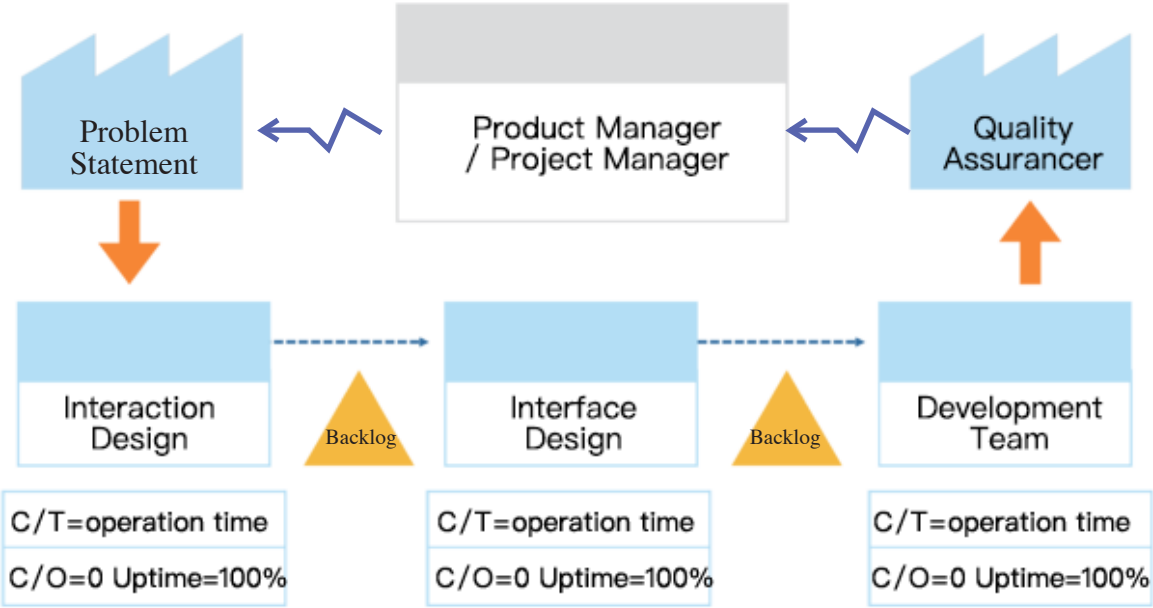
Meanwhile, in order to make up for the lacking of practical experience, a case study about Autodesk is discussed in the thesis as well.

# 3.1 Interpret with Lean Operation technique

Operation has been developing lean thinking for years, there are various lean techniques of value. Value Stream Mapping is among the most classic ones. “Value stream map” is a vital tool for analyzing process in order to reduce delay and waste, dig out the root cause of the waste in development process (agile developer’s guide-page 16 + VSM of design process) It uses a system of standard symbols to depict various work streams and information flows that represent the process required to bring a product to real. Each stage is defined as adding value to customers or not, so as to re-arrange the remove the stages that don’t add value.

The purpose of lean operation improvement is to remove waste, strengthen collaboration among various stages to improve efficiency. Based on this purpose, usually the very first step of lean improvement is to calculate the takt time, which is the average production time that each piece of product deserves. Then compare the real execute time of each stage with this takt time. In this way, the stages that causes direct waste are defined. The next step is to re-organize the workflow of existing stages, to remove, to add up, to change order, or to parallel them. This step is ought to consider the transfer stages within the whole workflow, should it be one-piece-flow or FIFO? By doing so, the overhead waste is likely to be removed.

Why the lean operation improvement is explained so detailed? The execution steps are defined and have been in practice for decades, which proves that the whole execution is valuable enough to be learnt, to be transplanted for the lean UX improvement. Let's start to compare these two lean improvements. Basically, both of them have the same final purpose, although the explicit objectives might be different. If we translate the concepts of UX into Operation, by means of metaphor, it would be much easier to understand how to transplant the lean operation improvement.



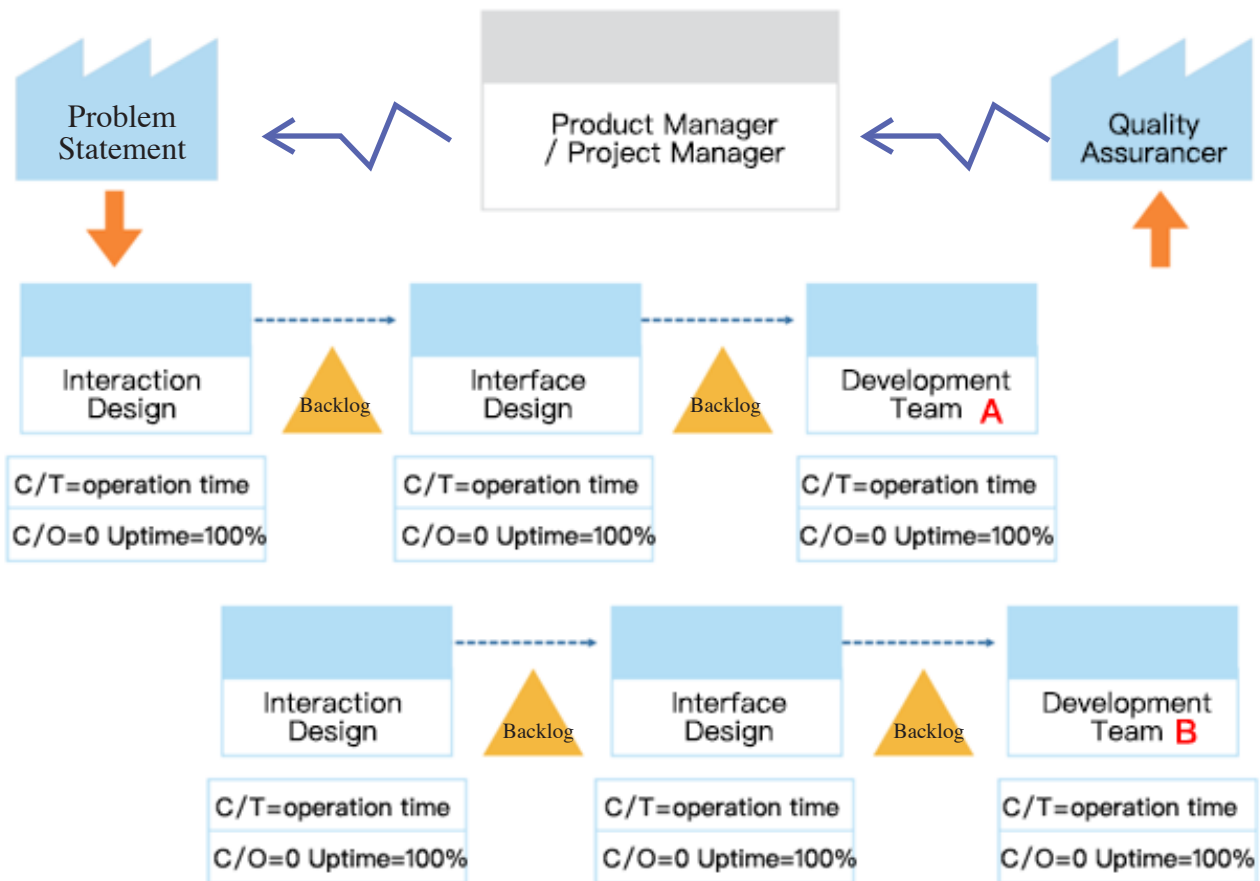
*Figure 3.1-1 The value stream map of basic UX workflow*

For the sake of clear explanation, Value Stream Mapping is taken to visualize the improvement process.

First of all, the actors are explained like this:  
Customers remains; Production Planner and Supplier ->  
Product Manager / Project Manager; each stage -> each  
role in the team, UX/UI/developers in order; inventory  
-> time and workload gaps between document delivery;  
final transportation -> Quality Assurancer; information  
flows and timeline remain.

Secondly, it's about flow. In the Toyota lean operation,  
it's PULL flow that is preferred so that to avoid the  
waste of both over-production and over-transportation.  
Comparing to production operation, the individual UX  
department seldom faces the overload problem, rather  
than the late-load problem. Consequently, this thesis  
holds a hypothesis that the lean UX is ought to take  
PUSH flow, starting from PM. In the classic operation  
execution, the smallest unit of flow is kanban, referring  
to dozens of products, cause production system focuses  
on repeating producing standardized product parts. This  
concept has to be transformed in UX, which is operating  
on different tasks each time, no matter on unique version  
iterations or product modules. This thesis defines the  
flow unit as the grouped tasks based on single module. It  
ensures that the individual unit could be passed to next  
stage and managed in time.

Thirdly, the calculation concepts are re-defined. For example, the cycle time is expressed as the operation time for each task, which means the C/T would be various depending on the task complexity. Both change over and uptime are estimated as close to the ideal time as possible, so C/O equals to zero while Uptime 100%. Then how should we estimate for the feasibility for one flow? Similar to production operation, takt time is needed as the first step which equals to overall scrum schedule time divided by the number of iterated modules (scrum time/#iterated modules). Overall C/T equals to maximum C/T of the flow. If TT is more than the overall C/T, the development efficiency is acceptable. In order to improve the development efficiency, the problem that development team should focus on is how to decrease the overall C/T.



*Figure 3.1-2 The value stream map of scrum flow*

The VSM above is an expression for one development scrum for 2 modules in the real context. In this case, there are 2 individual development teams working for separate 2 modules. Even though these teams could work parallel, the total C/T for the whole flow is more than maximum C/T of Dev. + UX. Because there is usually only one UX team for both modules, the overall C/T has to include the C/T of UX twice. This leads to the waiting time of development teams, which is one kind of waste that Lean should avoid.

## 3.2 Learn from the book Lean UX

In the beginning of the preparation for the thesis, the thesis focus was more on the innovation of UX deliverables skills than overhead UX work due to author's professional background. Along with the research on existing Lean UX literatures, the final direction turns to explore a balance between practical deliverables and overhead. The book Lean UX is one of the literatures that has great influence.

Lean UX is among the earliest literature that aimed at exploring how to implement Lean in UX. It emphasizes the theoretical and cultural shift, rather than the innovative attempts about practical deliverables. Basically speaking, Lean UX introduces a lean methodology consisting of 4 main steps: form assumptions, create hypothesis, make MVP and finally test and iterate. The link of these four steps is a simplified design-thinking solution, to create suitable product design which centered with customer value. As the basement of this methodology, these steps put a unique requirement on the team composition as well as the cultural shift simultaneously.

Even though the book name indicates an invisible connection or influence from Lean startup thinking, Design-thinking does play a significant role in Lean UX. The simplified model of design-thinking is to discover problems, find the corresponding solution to the problem, and then measure the feedbacks, so that to enhance the previous solutions according to these feedbacks. The content of Lean UX could be recognized as the reflect of design-thinking on UX work.



For example, the first step - form assumption, starts from a problem statement. The problem statement is a given problem that the team encounters while the team is ought to dig the “Why” rather than “How”, whose very nature is to discover problem just like design-thinking. After that, the whole team are encouraged to give their individual ideas to the specific problem which are the basic to form assumptions. And this step is compared to the create solution to problem as design-thinking. Next step in Lean UX is to create a hypothesis, which is designed to test these assumptions that formed. This step, together with the very next step-make Minimum Viable Product, could be understood as the test process of design-thinking.

As mentioned above, Lean UX is influenced by Lean Startup greatly as well, which could be told through its stress on the ability of mistake-adjust and quick-iterate. Distinct from traditional UX process, the product idea does not mainly come from traditional user research which in usual costs long time, but from the inner-assumption based on problem encountered. In this way, the team is able to save time from hesitating among different proposals but devote into testing and shifting directions in time.

Both Lean Startup and UX are implemented by a small but cross-functional team. This team composition, on one side, ensures that there are colorful background knowledges which may generate fruitful inspirations for the solution. On the other side, the team scale is not too heavy to manage and is capable enough to implement flexibly. However meanwhile, both lean startup and ux face the same challenge: how to persuade and promote the methodology shift successfully within company. Either lean startup or lean ux is different from the traditional execution methodology, which requires different managements in parallel. Correspondingly, Lean UX introduces the cultural shift within company in advance.

## **3.3 Case study - Autodesk Usability Investigation iteration**

This case is summarized from a paper - Adapting Usability Investigation for Agile User-centered design, written by Desirée.

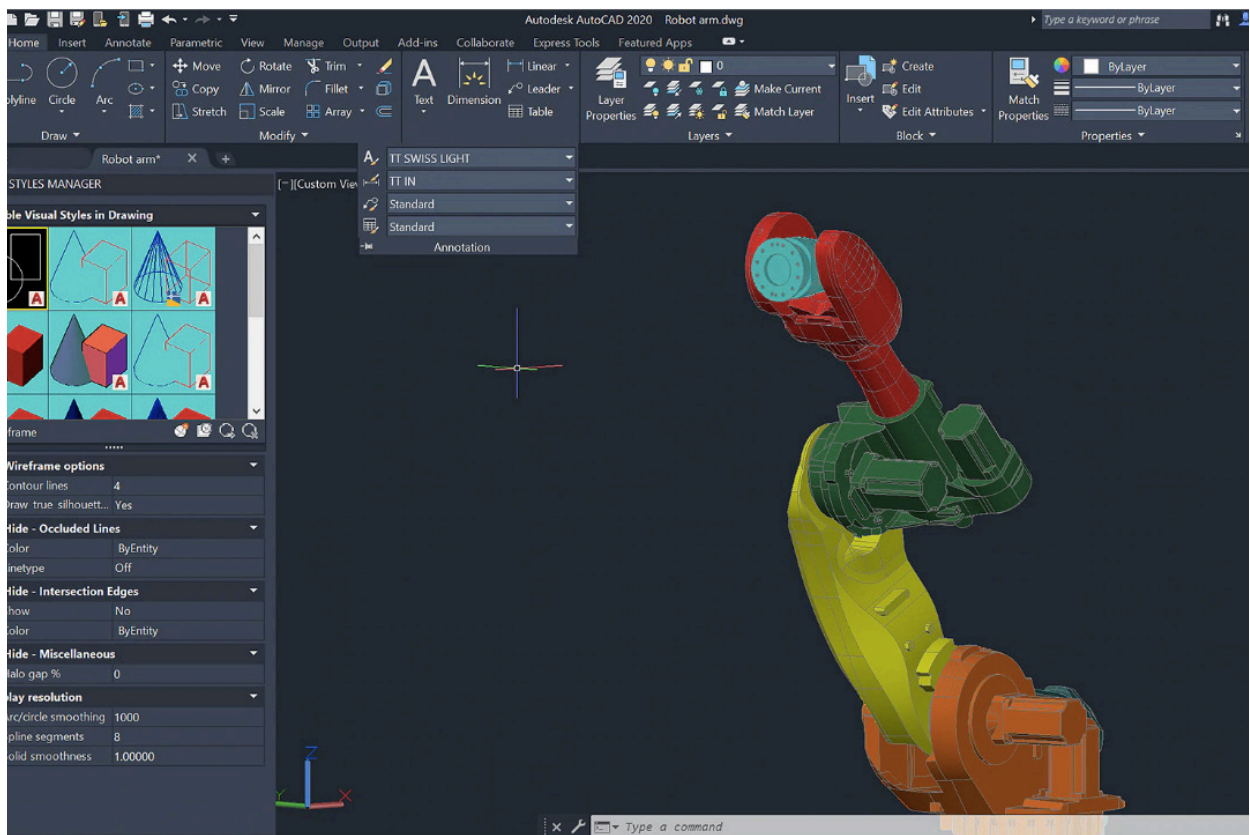
It is a real experience of User Experience team, concentrating on their adjustment appealing to the agile development methodologies. This case describes the specific adaptations in terms of timing, granularity, and reporting used for Agile interactive usability investigations, with an intended audience of usability practitioners.

Sy, an User Experience Designer. In the paper, Sy introduced the adapted usability investigation conduction. Even though the methodology that he introduced is not directly dedicated to the general Lean UX design, the process involves the UX design and coding implementation, and there are a lot of details that are of value for the Lean UX.

# 1. Context

Autodesk Company is a giant software corporation, dedicated in the professional architecture, engineering, construction, manufacturing, media and entertainment software field. It became best known for AutoCAD, but has developed a broad range of other software, especially for design.

From the customer fields, we could see that Autodesk Company has a large number of various product lines. And each product line requires a professional level of product quality.



*Figure 3.3-1 Screenshot of AutoCAD, representative Autodesk product*

In 2002, the Autodesk Company started to adopt the Agile development entirely, including the User Experience department. The specific Agile method is Scrum, which put a great challenge for the Usability Investigation work. The Usability Investigation activities in general refers to the prototype design, gathering customer data by means of interviews and other methods, conducting contextual inquiry and usability test, and analyzing or elaborating these data.

Even though the Agile training lesson introduced several agile usability approaches, these operations are not practical enough in real Autodesk working context. For example, the trained method for gathering information is to conduct a focus group, after a feature was implemented. But this is insufficient for the Usability Investigation work because this approach relies heavily on observing detailed behaviors.

On the other side, previous to this change, the User Experience team adopted the waterfall method, which is different from the agile one, in terms of the development flow and schedule. For example, in the waterfall method, the release is completed in a fixed single-direction flow: User Experience team completes gathering information from customer feedbacks, to better define the product features. Then these documents would pass to interaction designers, who will complete the design proposals, and finally pass to the implementation team. Each cell has to complete their own task before passing it down to the next cell. But in the new Agile method, the release is divided into few mini-releases which will be operated within continuous sprints. All the stakeholders are working in a continuous pace. The Designers could pass the first release output down to the coding team, then continue to work on the next release while the coding team devote to the first release.

## 2. Blocking Issues

During the conduction of new Agile methodologies, the User Experience team encounters several problems. Some problems are the remaining problems from the waterfall methodologies, while some come from the unsuitable Agile operations. Let's see the specific problems in detail.



Autodesk Company had been developing several features simultaneously. The ideal workflow arrangement should be one interaction designer works on one feature, then one programmer implements that, so that released features are all completed. But in fact, there were more developers than interaction designers, which leads to some features in implementation are not designed yet. The consequence is that not all the released features are completed in terms of design, and the gap between design completion pace and coding completion pace becomes larger and larger.



The second issue is that, actually the programmers started coding very early, even at the onset of the project, before the User Experience team published the feature specifications. The gap between the designed and the implemented appears again, and this is exactly a waste of time. In order to combat this tendency, the User Experience team has to begin their work much earlier, sometimes even one total release ahead. But this solution results in another waste, that the User Experience team prepared a lot of unused or out-of-date feature specifications.



Under the Agile (Scrum) method, all the features would be released in several sprints, each of which lasts for 2-4 weeks. That means the User Experience team must complete all the Usability Investigation within the sprint time, no matter how complex the corresponding design is. This requires a high-efficient working capability. Meanwhile, during each sprint, except for a few numbers of usability tests, the User Experience team are also responsible for the putting forwards the design solutions. So, the conflict between the timeframe and workload is a tough pressure for the User Experience team.



The last big issue is about documentations. Originally the User Experience team prepared a large number of documents for different stakeholders, such as the marketing, the development team, and the design team. However, this does not respect the Agile value that working software over comprehensive documentation. Because these documents are not corresponding to the way how other teams communicate. In turn, the work on writing these documents becomes a waste unfortunately of both time and team work, a hurdle to deliver on the Agile quality of rapid response.

### 3. Innovative UCD Process

Along with the practical conduction of the Agile methodologies in the User Experience teamwork, the team adapts in various fields, and innovates several approaches, especially corresponding to the blocking issues that we introduced in the last section. The focus of the new innovative UCD process is the 3 changes-timing, test size(granularity), and reporting.

A. For the issues related to the unbalance of complete and incomplete features, Sy introduced the Two Parallel Tracks: iterating the design and implementation separately but simultaneously. The graph below explains the key principle visually.

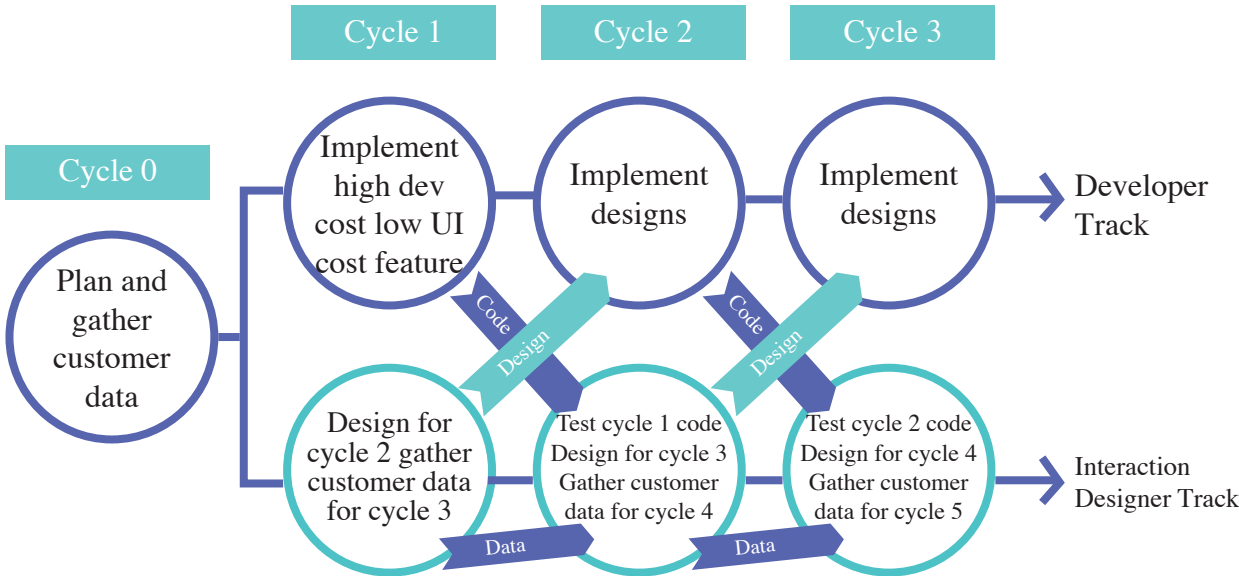


Figure 3.3-2 The overview workflow of UCD process

This track model is built on the base of Scrum method. So first of all, following the Scrum principles, the releases are planning to be developed in order. The timeframe of each release is called Sprint, or Cycle here. Autodesk Company hold a 4-or 6-week phase, called Cycle Zero, to do the release-level planning.

Before the sprint begins Scrum, team will hold a meeting to arrange the features together. The decision occasion is usually pushed as late as possible, to make sure that before setting the plan, the teams could get the out-of-date customer feedback. During each sprint only few features are picked to be developed.

The Sprint or Cycle here of different tracks is corresponding, happening simultaneously. In Cycle Zero, the iteration team makes the feature schedule loosely. Then the Cycle One is closely following the Cycle Zero. In Cycle One, developers are working on coding product architecture, or important features, both of which need few design works. At the same time, designers are hushing for usability investigation activities, including the next cycle graphic design and gathering customer data for the following cycle. This pattern is continuing until the product is released. But no matter which cycle is, the designing is always one cycle ahead of developers, and the gathering customer data two cycles ahead. The whole collaboration process is like a relay race.

Within each sprint, the developers are concentrated on only a few well-designed features at one time, and interaction designers have already worked for the same features. This model actually builds a strong bridge between the development process and design process, to ensure that all the released are the implementations of the designs.



B. For the issue related to the coordination of workload and timeframe, Sy introduced the approach of Design Chunking: breaking design apart into cycle-sized pieces. The issue is that the design duration that the work required is longer than the sprint time that assigned. The root reason is that the real complexity of the usability investigation is much more than the assumption when Scrum team plan for it. Or even the Scrum team did not consider the complexity of the work. The inspiration to solve this problem is to take the usability investigation complexity into consideration when set the cycle planning.

Specifically, the Design Chunking requires the interaction designers to deconstruct the big design into small, cycle-sized pieces, based on the design intent that they understand from the observation of customers. These design pieces have the inner-link to each other, as the elements to the overall design. In this case, the User Experience team investigate, prototype and test these small design pieces progressively in the design track. When the User Experience team completes the design to one progress, they pass the completed design to the developer track, following the workflow that we introduced before.

Comparing to initial approach, the design chunking process actually decomposes the whole design release in a logical way, into progressively add-value elements. This decomposing logic has a bias on the design validation level in the product lifecycle, rather than the merely task planning. Besides, as Agile principle said, components build on one another, so the design chunks are developing, from low-level and fundamental to high-level. That means at least the early chunks will not change because the later ones will be added on top of them.

C. For the issue related to the documentation, in Sy's paper there are a few interpreter tools introduced which taking place of the traditional documentations, such as the design card, feature card, issue card, the cycle planning board, and the user experience board.

Other Agile team members have adjusted their communications under the Agile principles. For example, developers and project managers adopt the daily scrums and cycle planning sessions, so that they abandon documents such as marketing and product requirements documents which cost long time to write or read. Influenced by this change, the User Experience team turns to a more timely, free and specific manner as well.

For the features, there are two kinds of representation tools, feature cards and design cards. Feature cards are used to represent the features that are in development in the current cycle, while design cards represent the upcoming features to be developed in the coming cycles. Issue cards are from the gathering customer feedbacks after the scrum. The User Experience team would track the completed features by inviting the Design Partners to review the new features. Any request for new feature based on that or dissatisfaction regarding uses of the product would be reported as the issue cards. In the next step, these issue cards would be present to the Agile team members, technical writers and QA people. They would discuss about how to deal with these "issues". Usually some issues are moved into the development phase, the cards become the feature cards or design cards and are moved to the cycle planning board. The other issues are remaining to be fixed by the User Experience team, and the cards are moved into the User Experience Board. Besides, it should be noted that the issues cards do not include the bugs, which should be recorded in the bug-tracking database.

Except for the cards and boards tools, there are still traditional document deliveries for some Agile team members, such as the design team. One of the mandatory documents is the design history documents, recording the version changes. The main function of this document is to notify the related Agile team members of the updated design specifications. But this type of document is just light specification, including the simple UI description and Interaction description. During the daily work, the communication between designers and developers are still face-to-face oral conversation.

## 3.4 Case study – Spotify Matrix Management Structure

The second one is the introduction of Scaling Agile Conduction at Spotify. This case brings some inspirations in particular on the organization structure. The organization is not only a matter of how to assign people. A good organization could avoid some waste, and result in much more efficient work.

“Google, Amazon, and Apple could crush Spotify in a nanosecond if the company wasn’t perpetually striving to be faster, better, and cheaper. To survive, Spotify has to be Agile. They have to keep on running out ahead.”

— Dr. Jeff Sutherland, co-creator of Scrum

This case is from a work review regarding the Scaling Agile, written by Henrik Kniberg & Anders Ivarsson who work through it. In this article, the authors shared the specific experience about how Spotify organize the huge development departments brilliantly with an agile mindset. Even though this article does not concentrate so much on the concrete agile workflow, it introduces another significant element of agile development-team context for the agile.

## 1. Context & Waste

Spotify Company is a music streaming platform, based in Sweden. The main service offered is the music player. Spotify adopts a freemium business model, the basic features are free for the customers but there are some advertisements or limitations in the basic version. Customers could enjoy a premium service with additional features and reduction of limitation by paying for the subscription.

Spotify Company has only conquered the music player market for 6 years, but has a large customer base for approximately 15 million active users. In order to maintain the competitive advantage, Spotify team is always put high requirements on the product updating. Meanwhile, Spotify Company has a great-scale development team, for almost 30 mini teams in 3 cities, which sets a big challenge for the team coordination organization management.

From Henrik Kniberg & Anders Ivarsson's perspective, there are at least 3 wastes in the formal agile process:

- Production Waste, development is slowed down or blocked caused by dependency on some organizational support
- Waiting Waste, release is suspended because the work process between design and development team is not synchronous
- Lacking of economy of scale, the developers with same skills repeat seeking for solution on the same problem because of the communication gap

About the first waste, usually the single development team who is responsible for one feature relies on the organizational supporting, such as with product owner & agile coach, and synchronous collaboration with other feature teams. The specific dependency on organization results in production delay.

Similar to the first waste, the dependency happening between different development teams results in the waiting queue problem. For example, one source of dependency issues is development vs operations. The developers are the people coding or programming for the creation of new programs, websites or databases. While the operators are the system administrator, responsible for the deployment of the coding on the existing servers, websites or databases. Usually, operators' job is to make releases for the coding after the developers completing. But there are usually some kind of handoff from developers to operators, accompanying friction and delays. In another word, this waste is the waiting in distribution.

The third waste is related to the production efficiency. The matrix structure is broadly adopted by companies, in which the roles with similar working or skills are divided and assigned into separate mini teams. These roles are supposed to build the advantage of share experience and create solutions for the benefit of all the same roles. But in this case, the advantage is blocked by the communication gap from team border.

Spotify Company has evolved the Matrix Structure fascinatedly successfully towards a better Agile development in scale. In the following section, the summary of Spotify Working mechanism is presented.

## 2. Innovative Organization Management

Spotify Company, in general follows a Matrix Organization structure. It is consist of several Chapters or Guilds from the horizontal dimension. In vertical dimension, Spotify is composed by a few Tribes, each of which constitutes with squads. These component units have various collaborate mechanism, different collaborate level.



Squad: this vocabulary's meaning is the same as team. Squad is the minimal basic development unit, in nature is the normal scrum team. So inside one squad, there are the product owner, development team, and an Agile coach shared with other squads. Squad has all the roles necessary for the development, from UX designers, Programmers, to Interaction designers, QA testers. Spotify Company is service-leading, assign and arrange the development team based on the services, not specific modules. Each squad is responsible for single feature (such as the payment), towards its own mission (like convenient payment process).

In addition to having a long-term mission, squads are also encouraged to work in the Lean Startup model like startups, following the principles like the MVP and validated learning. This means each squad is ought to autonomously iterate and test frequently. Different from the real startups, squads have more or less dependencies with other squads, and organizational supports. These dependencies could be of value to complete multi-squad project. But squads tend to avoid or reduce those bad blocking dependencies, as one kind of waste.

For the blocking or slowing squads down dependencies, Spotify Company usually holds a survey for the squads regularly, to figure out what kind of dependencies they encounter, and to what extent the bad influence is. In the next step, the result of survey would guide the correction solutions, such as reprioritization, reorganization, architectural changes or technical solutions.



**Tribe:** is the collection of various squads which work in related areas. An approximate definition of area is the certain service of a product. For example, the music player is a separate service involving all the features related to music-playing, like the album list, player window, radio, and what's new section. Comparing to music player, the Payment, not directly related, is another service. So squads working for payment system belong to another Tribe.

Strictly speaking, Tribe is not a management unit, but a equal community, with high degree of freedom and autonomy. Squads within the same tribe are able to get acknowledge of other squads' projects, tools, techniques and experience. By means of learning from each other, these squads grow faster.



**Chapter & Guild:** Chapter is the collection of people with similar skills, and within the same Tribe. Some examples are like the testing chapter, the web developer chapter or the backend chapter. The most significant advantage from Chapter pattern is the scale of economy in development production. For example, tester A met the same problem that tester B just solved last week, so fortunately tester A could learn this experience from B rather than spending additional time on this problem. This kind of benefit



improves the efficiency of development very well, eliminate the waste of repeated “production”.

Guild is similar to Chapter, where members meet to share knowledge, tools, code, and practices. The difference is Guild’s range is beyond the same Tribe, coming across the overall organization. And not as formal as Chapter, Guild is more organic, like a community of interest. For example, the testing Guild include all the testing chapters, but a backend developer could also take part in as long as he is interested in as well. The inclusiveness for members of Guild is much better than Chapter.



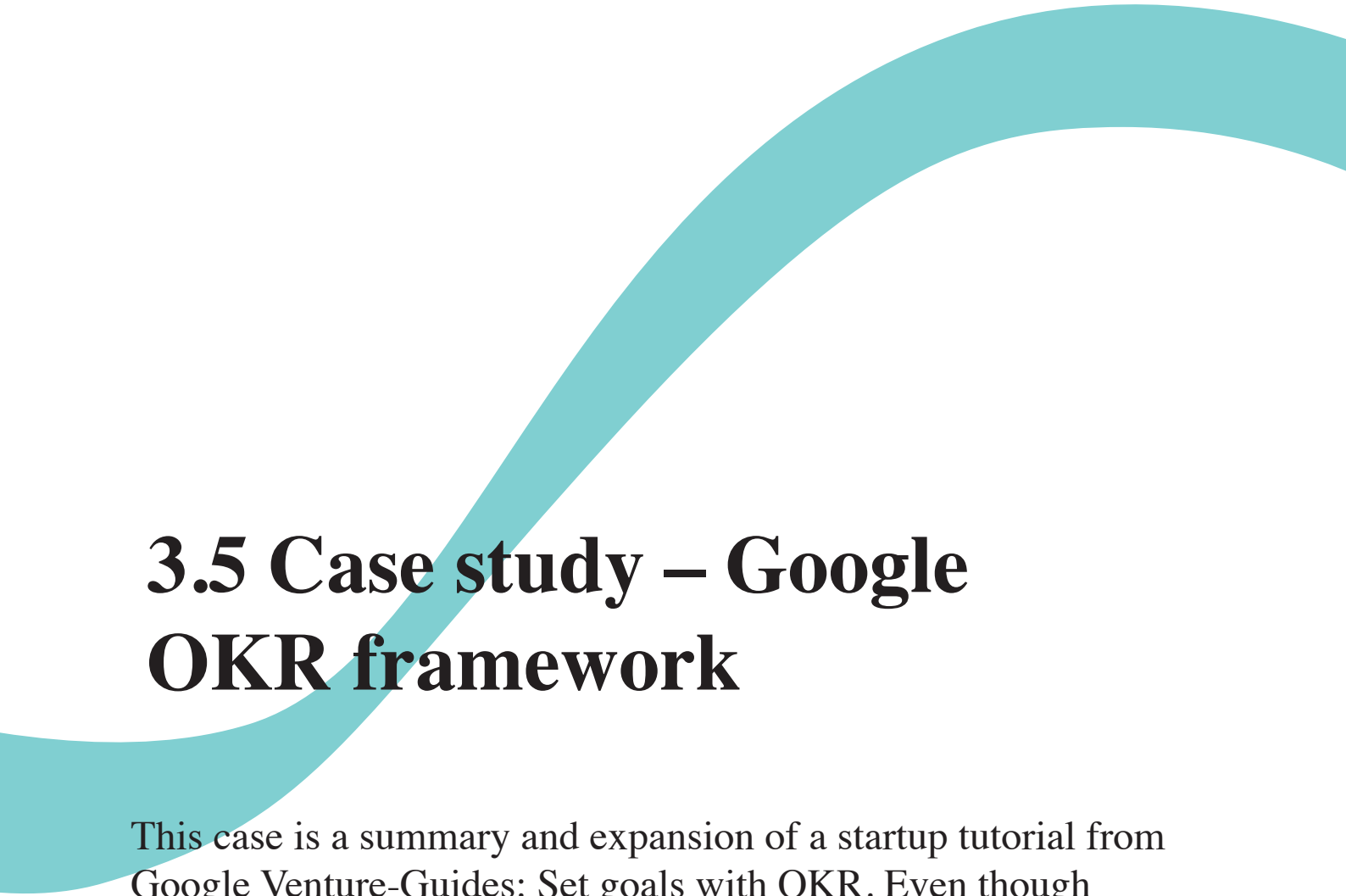
Structure: The information architecture is the habitat, the basement where all the coding would be deployed on. But because each scrum team has the autonomy to deploy its own coding, without asking for permission ahead of time, there is the risk that the architecture of this coding system gets more and more messy. To tackle this problem, there are structure leaders, called system owner, working in pair to coordinate and guide people related to the architecture. The pair-working means that behind one system owner role, there are usually one developer and one operator working on the architecture, to make sure they consider the structure issue from both developer and operator perspectives.



Comparing with General Matrix Organization: According to the authors’ opinion, Spotify Company’s organization structure is a different type of matrix organization, more than the common matrix organization.

In the vertical dimension, each member belongs to a squad, or called as the scrum team where they are grouped towards delivery. This development unit is self-organized, and as the development base, squad members spend the most of time in this vertical level. The most Agile factor is in the horizontal dimension, there is the chapter or guild organization, allowing members to share knowledge, tools, and practices, as well as setting salaries. Instead of generally pooled together as the common matrix organization, these horizontal groups make full use of mental production capabilities.





## **3.5 Case study – Google OKR framework**

This case is a summary and expansion of a startup tutorial from Google Venture-Guides: Set goals with OKR. Even though the target of this tutorial is mainly the CEOs of startups, the method expanded through it is adopted quite broadly. From giant companies, like Intel, Google, to enormous startups, this method brings continuous creation and perfectly facilitates Agile development, so it deserves to be studied.

## 1. Context

In early 2000, Google was just founded within 1 year. As one of Google's early investors, John Doerr introduced OKRs to Google's leadership to help Google employees focus on a set of priorities in order to receive the success as soon as possible. As one important source of continuous growth, Google has its own venture for acquisition or partnership with other companies. In order to synchronic with the the parent company's efficient working pattern, Google educates these new member companies with OKR methods.

Guided by clear and sync goals, Google has been developing fascinatingly in the past decade, triggered a chain of products beyond Google's initial core products and services. But the quality of Google products maintains or even improves, because the product iterations are agile. And the challenge of huge organization management has been balanced by the OKR, doesn't have a terrible influence on product developments.



### Common Issues: 1) Waterfall Goal-setting

When it comes to development delivery process, the Agile mindset and processes have been normal. But for the strategy and goal setting, most development teams are still using the waterfall mindset, which refers to an annual, top-down process to create a set of goals. This is in conflict with Agile.

In the waterfall goal model, this organization system is just like a feature factory, each team works like a machine. The developers are “just sitting in the factory, cranking out features, and sending them down the line,” as John Cutler described. “The teams have little understanding of the bigger context, and even less belief that these are in fact the right solutions.” Teams are indifferent with the outcome, focus mainly on the output.

In the long term, waterfall goals make it harder for organizations to adapt, and increases risk and waste. The team with waterfall goals are project-based, focusing on using Agile to deliver waterfall plans. Their plan is relatively static comparing to OKR, dynamic plan. The static model carries several assumptions:

1. We can define all the steps of the plan in detail in advance;
2. The vast majority of the plan will be correct;
3. Market conditions will remain mostly the same;
4. Changes will be small. We will deal with them with in a review in the middle of the year. We will then create an updated detailed static plan.

This kind of static planning is created based on the predict of the future, carries the risk of high mistake-adjusting cost. Besides, teams are used to stay within their comfort zones, gradually against autonomous innovation.



## Common Issues: 2) Difficulty in Re-priority

In the Agile development for delivery, more specifically in the scrum process, teams used to set a set of development backlogs according to the user stories ahead of each sprint, then work on implementing these backlogs. If the product owner adds some urgent or temporary user requirements into the current backlogs, the whole team has to adjust their development proceeding temporarily. The same for the deleting some backlogs. In a word, changing backlog temporarily is an careful incident, because it might have a great influence on the whole development.

However, re-priority is a common issue in development, because there are always brilliant ideas appearing continuously, facing future generations (Scaled Agile, Inc, 2018). Sometimes, the hurdle in the development could also lead to the work adjustment. So how should we deal with those situation? The ideas seem to be valuable, but no one could give a guarantee. Those work hurdles interrupt the workflow, but the development team does not have enough time to re-arrange the whole development, and not all the development works should be replaced. In this case, the order of different development backlogs are ought to be re-prioritized, where the OKR fits in very well.



### Common Issues: 3) Limitation of KPI

KPI, Key Performance Indicator, has become a useful metrics for performance evaluation. Almost all the organizations use KPI as the navigation instrument to help employees track their progress, have a picture of their current levels of performance.

But one of the most common problems is organizations misunderstand KPI as the target, rather than the indicators. This is a subtle but very important difference between these two roles. There is one extreme example of KPI target. The metric of KPI is set measurable, so if the team treats this KPI as a target, it will concentrate efforts merely on the metric events, over and over, no matter if the metric events have been completed enough. For example, a product team sets KPIs to measure the amount of efficient features that could increase the page view. The product managers started to put forwards different feature ideas. Finally they add more than three new features, but actually they have already increased the page view to a brilliant level by the first three. The extra efforts become a waste to some extent, and those extra features may affect the product value proposition.

In order to fix the issue of pursuing KPI overly, organizations might tend to set several KPIs which could balance each other. But meanwhile this kind of correction means stricter rules in fact. As a result, if we use KPIs as targets, we get what we measure, and that's all. Accordingly we could say, organizations are in need of new goal-setting methods.

## 2. Using Agile with Agile Goals

OKR, short for Objectives, and Key Results, is a value-driven method, helpful to create the value-driven team. OKR methods have been adopted to set goals not only by Google, but also by most of Google's portfolio companies. Research reveals that when people are committed to their goals, their performance are higher than normal, The more challenging and specific is the goal, the further can enhance the employee engagement in attaining those goals. So OKR is an important part of the development work, like a gas station to a travel.



The components of OKR. OKR is originated from Intel, Andy Grove's theory. Grove explained the OKR as "The key result has to be measurable. But at the end you can look, and without any arguments: Did I do that or did I not do it? Yes? No? Simple. No judgments in it." OKR is consist of two basic parts, the Objective, and Key Result. The objective is compared as the general picture of the final goal, while the key results as the specific directions, in a more "metric" way.

One of the components, the Objective refers to the ambitious goals that employees believe is worth their time and efforts.

There are few principles of setting these objectives:

- Not too many objectives, 3-5 is suitable;
- Require improvements, instead of maintaining;
- Clearly charitify the objectives as far as possible;
- Avoid using ambiguous expression.



Another component, the Key Results, follows after one specific objective. Key results could be understood as the direct results as the corresponding objective. Its principles are:

- Approximate 3 key results are committed to each objective;
- Set useful tools that could advance the objective as key results;
- Describe the outcome/impact of activities, rather than activity merely;
- Charitify clear, obvious, and discoverable evidence of completion.



The mindset of OKR. OKRs are the ambitious value-based goals, different from the normal waterfall activity-based goals. If you still use the Activity-based targets with Agile development, it creates friction. In Agile, especially Scrum process, the development team defines the product backlogs in the beginning of the sprint which are actually the roadmaps already. As a result, this team has to struggle to solve the overlapping or conflict with OKRs, to connect OKR and Agile.

OKRs are value-based, which means the implementation team has much more autonomy and freedom when conduction. The only principles or rules are the releasement of the value, not any static project. It is a shared list of target, not shared to-do-list. This setting would facilitate the implementation team to realize value in better ways, not limited to the promised project. Accordingly, the team not only delivers tasks planned by executives in Scrum, but more importantly they might put forward and complete other tasks towards the same value.

The most important point is, those objectives and key results are defined by the implementation team itself, which empowers the team to be responsible for the target. What's more, those objectives or results are somewhat challenging, stimulating the team's motivation harder. Employees understand what impact they could make, so that they would have a clear purpose and great passion for their work.

But note that: OKR is value-based, which means OKR focus on value, does not means the OKR could not be defined on activity. For example, the Objective is to raise the user experience, Key Results could be to increase lead conversion from 5% to 8%, or to develop 3 new landing pages. The later one is just an activity goal but focus on a value.



The execution of OKR. There is one flawed assumption behind the normal Agile development process: some stakeholders tell the teams are asked what needs to be done, define their targets. So actually the targets are waterfall setting-flow. While successful OKRs should contain both top-down and bottom-up suggestions, come from the voice all over the organization.

When setting objectives, it's more useful to start with the organizational OKRs, and set 3 key results for each objective. Then the team OKRs would be set to serve one Key Result of the organization level. It is accepted that "not every organizational OKR needs to be reflected in every team OKR. It's possible that a team's OKRs will focus on just one of the organizational OKRs." But the team OKRs must be related with organization OKRs, not totally innovated by team. Otherwise, the team OKRs may have no contribution towards the general value, become less important and lower priority.

When reviewing OKRs, OKRs are graded on a scale from 0 to 1, in decimal. 1 usually represents satisfaction while 0 means the target is not reached, while the sweet spot for OKRs is 0.6-0.7. Firstly the key results are graded, then the average result is calculated as the grade for objective. At last, all the OKRs should be shared and graded publicly within the organization, to ensure that everyone have the access of the whole OKR, knowing their contribution.

But do not focus so much on the grades itself, because the grade is just a number, and there is some weighting difference. More important part is what elaborated by grades. Be conscious that OKRs are not synonymous with employee evaluations. Performance evaluations should be conducted by KPI nevertheless, independent from the OKRs.

**04**

**Learning  
Integration**

A decorative graphic element consisting of multiple horizontal, wavy bands in a light teal color, positioned on the right side of the page and partially overlapping the text.

This chapter would introduce the blueprint of ideal Lean UX Design workflow, including the collaboration situation from the whole company perspective, but also the specific implementation workflow of Lean UX. A whole workflow is divided into pre, during and loopback phases. The team composition and collaboration process are included in the end of the introduction.

Due to the risk of Lean UX experiments in real business, the author decided to make interviews with some experienced UX designers to estimate the potentiality of the proposals.

# 4.1 Context & Collaboration of Lean UX

Cooperated with the scrum tactics, the Lean UX value stream map is revised as followed:

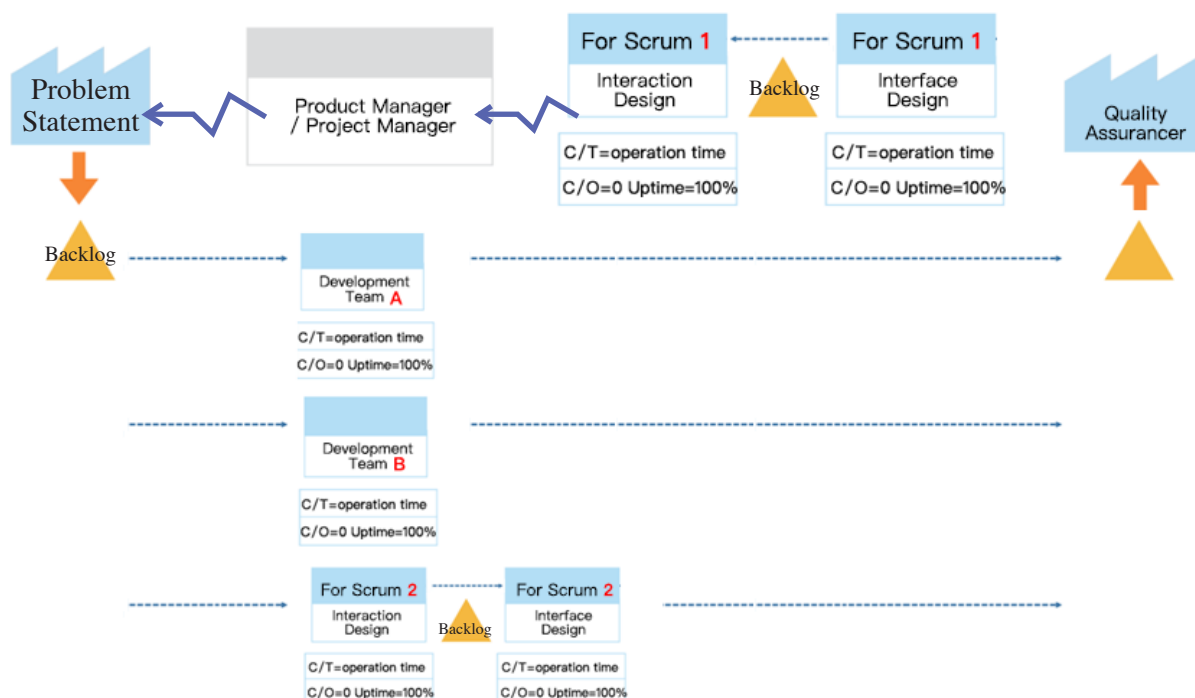


Figure 4.1 The value stream map of new Lean UX

This VSM focuses on the introduction of the overall workflow of the development, create a bird-view rather than detailed description of the UX work activity for deliverables. Comparing this VSM with the original VSM introduced in the section 3.1, the biggest difference is the opportunity when UX join the scrum. Originally, the UX department work on the same scrum. This results in the time gap between the UX and development team,

because different development teams work on various modules but UXers are not able to complete all the modules before development begins. In reality, thanks to the popularization of

Agile thinking, more and more UXers changed their working mode, complete their work at least one scrum earlier than the development.

The revised development line is consisting of product manager, UXers, developers and quality assurers. In Cycle Zero, PM activates the scrum, and send requirements as Kanban to UXers. In the next cycles, PM and UXers work together to prioritize and assign Kanban - the UX outputs which UXers worked in advance to developers. Meanwhile, PM would send new Kanban to UXers which require them to complete in the same cycle. The output of developers, the iteration demos have two main channels: as First-In-First-Out order, are 'transported' to quality assurers in time; or waiting for QA till the overall iteration system is done.

The iteration workloads are divided into several scrums following the development arrangement. UX department works at least one scrum earlier than development team, and the UX works have been assigned based on modules so that the development team are able to work on separate modules in parallel, without any cross-work as possible. At the same time, the UX department are busy in preparing for the next scrum, working on the necessary deliverables. The greatest benefit of this work flow is to reduce the total Cycle Time of one flow, from the max (C/T Dev. modules)+UX work gap time to solely max(C/T Dev. modules). The explanation is built on the common

knowledge that the very module iteration always costs much longer development time than UX design time, so the UX work gap time is ignored under the max (C/T Dev. modules) when calculation for the total C/T.

There was one stubborn thinking that the UX design is an area of specialization. UXers are those people who have been skilled in design, user interaction, take charge of the design process. They seem to target at the 'pixel perfect' early designs, measured by how well the implemented user interface complied with the initial UX design. UXers in fact were separated from the centered roles, working as a useful screw, rather than a key. However, the situation changes dramatically from this revised VSM. What Lean UX does is it totally amend the UX time into decision-making process in terms of time, apart from it adds a decision-making assistant into the development process.



# 4.2 Persona & Original Customer Journey Map

## 4.2.1 Personas



### **PSSD Tea Retail Company**

PSSD TRC is a startup company, founded by 5 90s Polimi students. Its business focused on tea retail, both online and offline. PSSD TRC is running a tea bar in the Polimi Leonard Campus. Their main product is actually an app called iTea where customers could order and pay for the tea delivery.

### *Scenario*

PSSD TRC says iTea allows customers to enjoy the happiness of bubble tea. All they need to do is shop through iTea and wait. Then iTea could reach through delivery within promised time, create great convenience and comfort customers.

### *Goals & Expectation*

PSSD TRC is a group of ambitious young student. They are making efforts to build tea consumption as a brand-new popularity through the fashionable retail model. PSSD TRC is always expecting iTea could become as inclusive as iPhone, everyone is able to and look forward to buying a cup of bubble tea and they are working towards it.

## *Pain point*

PSSD TRC is a small company currently. They spend the most of time promoting brands, building the delivery system and maintaining suppliers. The first customers are quite meaningful and valuable to them. But they are weak in updating app which usually takes at least 3 weeks to update for one time. In case they insist on treating nice service as a competitive advantage, they must find a solution to fasten the update process.



## **TheFork Online Payment Project**

TheFork has been the No.1 of restaurant booking industry for years. There are over 100 employees, 4 main departments like coding, product, ux design and quality assurance. The online payment project is a newly built project, and TheFork builds a sub-team which is responsible for building the online payment system and maintaining.

## *Scenario*

OP team is working to provide a safe and convenient payment channel within TheFork app. This would allow customers paying for the meal digitally which is to the social tendency.

## *Goals & Expectation*

Online payment is growing strongly in Europe, more and more customers are used to pay online. OP team expects to satisfy the growing and various customer demands around TheFork consumption, such as paying for tips or taking loans.

## *Pain point*

OP team is sub-team of TheFork company. Even not all the employees are totally devoted to the project, their uptime is around 70%. Meanwhile, the management is not clearly defined. It seems very hard for these team members to deal with new project demands in the traditional way within limited time.

# 4.2.2 Original Customer Journey Map

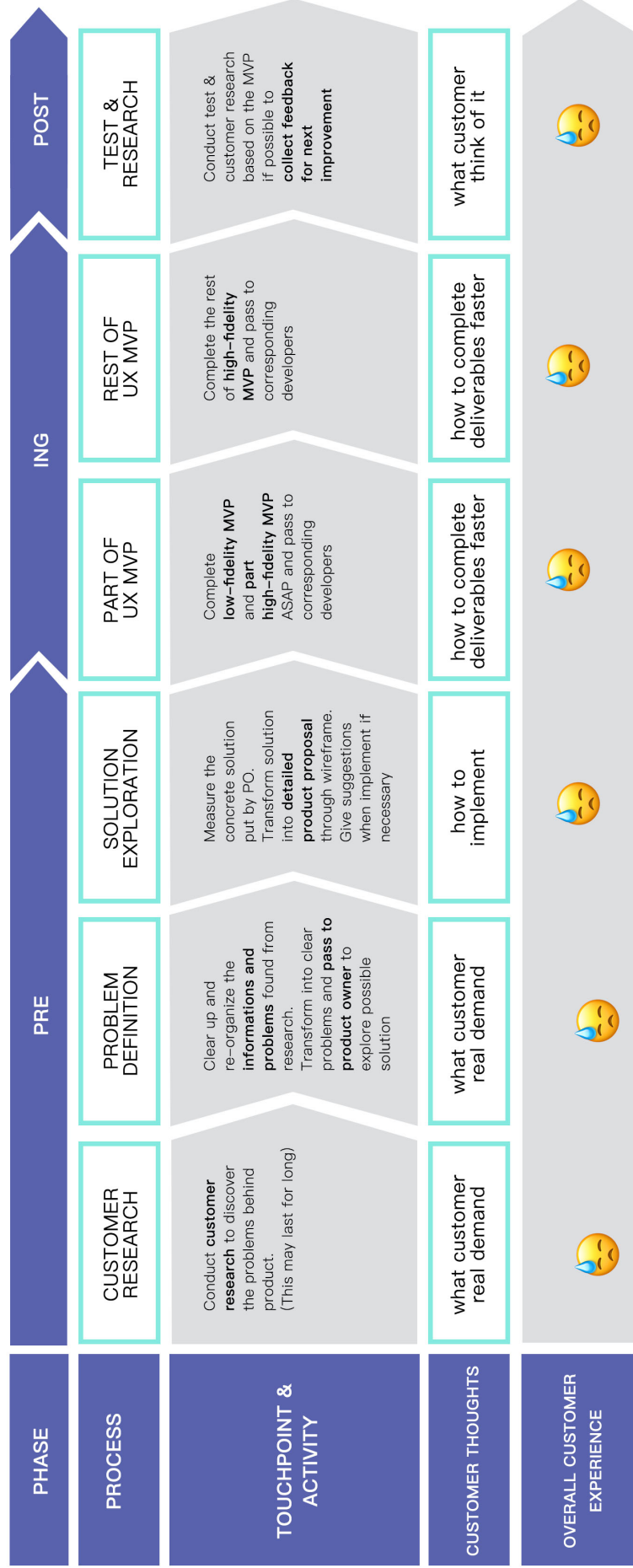


Figure 4.2.2 Customer Journey Map of the original UX process

This map is the original customer journey map which concludes how current UX designer work. In this CJM, the UX designers are treated as the customers, the process of UX design for a project are abstracted as the journey. In order to express more clearly, the CJM has hidden less important project details and stakeholders.

The first persona PSSD TRC is a startup. For them, it is the very first and significant step to find the correct and valuable customer demand. So, UXers and other team members of PSSD TRC spent the majority of their time in conducting customer researches in the beginning of the project, in order to dig the useful demand out. As soon as they found the potential demand, they tried to solve this problem with limited cost but best effect, then they got the business model in this way. Next, UXers were devoted into making the UX MVP in very-low fidelity. After the best proposal is selected out, they moved into making the high fidelity while the developers are ready to realize these designs into real programs. In the end, they test the programs by themselves and went to release. In the market, not all the customers like the product immediately, they in nature have different experience and opinions towards the early products. If lucky enough, these feedbacks could be collected by team members directly. And these feedbacks would work as the evidence, or strictly speaking guidance for the next UX design iteration. In another word, the UXer journey would repeat with guidance from their own customers.

The same UXer journey map work on the second persona-TheFork online payment project as well, even though they are not the real startup. But from some other perspectives, they are rewarded as the startup within the existing big company. The difference with PSSD TRC is that payment project is supported by TheFork, so that they would have enough resources on

development, QA and some UX design materials. These benefits allow UX journey proceed much faster, and even skip some phases such as the split UX MVP. Apart from that, in general both personas are following the similar journey map as the picture above.

In the next section, we would discuss how to improve and implement lean ux based on these two personas.

# 4.3 Implementation of Lean UX

The revised CJM represents the detailed Lean UX workflow, as following:

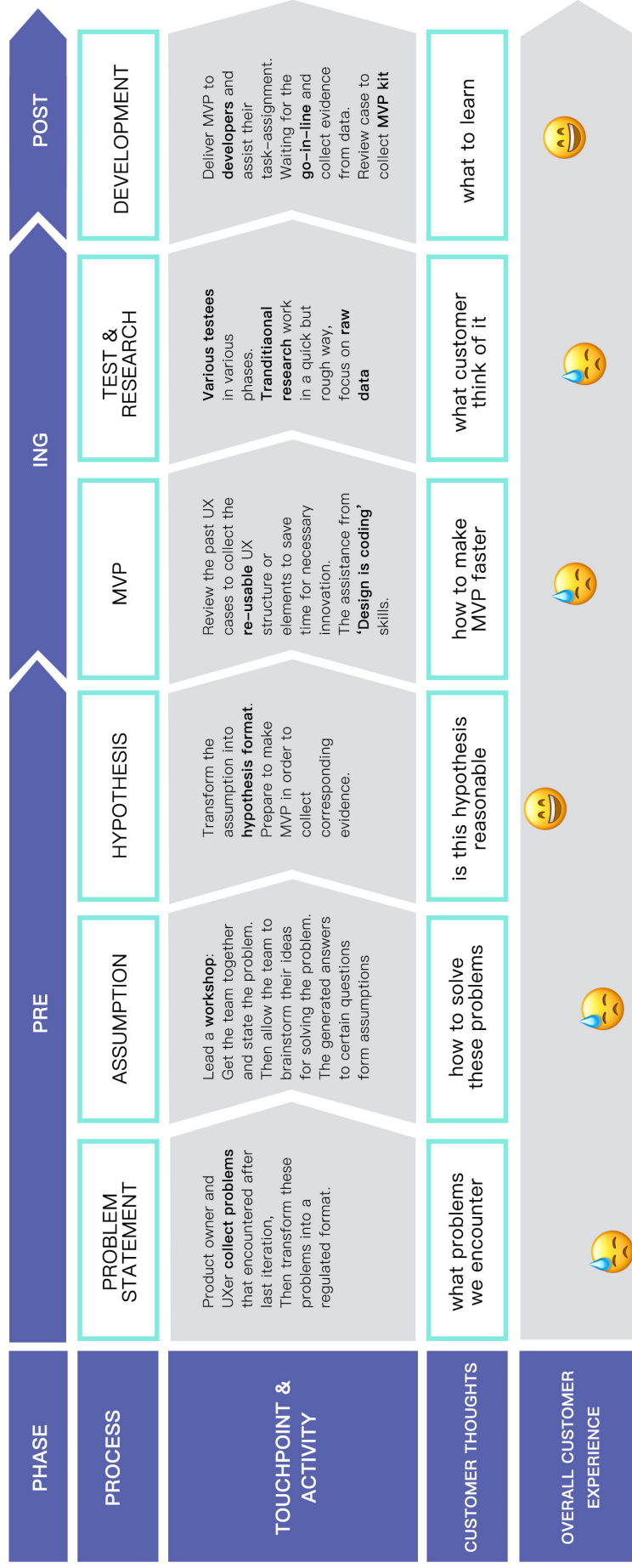


Figure 4.3 Customer Journey Map of the developed UX process

The new journey map is able to release the pressure of UXers effectively, because it extends the activities of UX with more development roles, the responsibility of UXers are distributed into several roles. The focus of UXers is still on the customer experience, but the new work arrangement allows UXers to approach to business requirement and customer demands simultaneously, and catch up with the pace of development iteration.

Based on this graphical journey map, the following sections would explain the specific journey activity by phase.



### 4.3.1 Pre-Phase (Preparation & Collaboration)

The unique value that UX design is ought to bring to the Development is to discover and define customer demands clearly through User Research. During the development process of early prototype, the brilliant effort of digging customer demands is necessary. However, except for the that early process, UXers' goal is harder and harder to achieve. UXers become more and more embarrassed, closed to graphic designers. What resulting in this situation is that user research on average takes a long time, requiring Product Owner to digest the research output in addition.

The new UXer journey changes the method of getting user demands from honest, traditional user research to problem statement originating from real tasks and business problems. No matter it is within a startup or big enterprise, all departments are tasked with doing something, and responsible for some business problem to solve. What's more important is that, those first-mover customers usually have diverse feedbacks or encounter diverse problems towards these earlier versions. To take full use of these feedbacks, all the roles in the same team, designers and non-designers are encouraged to extract useful information from that. Typical information is used to answer key questions as following: (according to Interaction Design Foundation, 2017)

- Who are our users?
- What is the product used for?
- When is it used?
- What situations is it used in?
- What will be the most important functionality?
- What's the biggest risk to product delivery?

Typically, this information is generated on a workshop basis. Answering those questions is of value to narrow the product directions effectively, and polish product features positively. This information is not the first-hand material from the direct customer research, but development team's extraction and conclusion. Even though they are assumptions in nature, this information remains worthwhile to some extent.

For the next step, assumptions are transformed into hypotheses, in other words, testable statements that inspire a solution to a business problem. As stated by the Interaction Design Foundation, a hypothesis states a belief, its importance, and the personas it is important to. Moreover, a hypothesis offers criteria to assess its rationality and effect, so it involves some expectation and a final result that will prove the belief. Another benefit of the standard configuration of a hypothesis is that it gets rid of communication overhead when collaborating and in the following teamwork.

The rest of the team beyond UXers now has the chance to give some additional input to the design team by taking part in extracting assumptions and forming hypotheses, rather than merely working as a coding machine.

According to the hypothesis, here comes the outcome for the next process: outcome + persona + feature.

### 4.3.2 During-Phase (Deliverable & Test)

This new journey is set in the modern software development background. In the classic agile software development, teamwork begins from Product Owner (PO) writing story. When the PO writes a new story, he first works with the UX team to get some initial ideas from hypothesis. In return, PO is ought to assist UXers to make the MVP, providing feature-related ideas for MVP, or even interaction suggestions. In fact, except for some strictly normal companies, the role of UXer and PO, in this phase, is overlapped in most companies. They only get diverged in the professional UI phase. As the very next step, PO gets to prepare stories for the development sprint.

Took a bird-view, the During-phase is involved not only with UX, but also with the delivering with development team, hence it is reasonable to break this process into 3 sub-phases: building MVP by UX, deliverables and release by Dev., and Test & Research by UX.



#### **Building MVP: useful kit**

After the Pre-Phase, UXers and PO have access to diverse customer demands. Using all the ideas brainstormed and the hypotheses created, PO then is able to build the MVP ideas, ranging from detail improvements to feature innovation. All of these MVP ideas would be exhibited in the final MVP made by UXers later. There are multiple specific methods on how to extract and develop MVP ideas, such as brainstorming, researching on competitors, working backward from branding expectation, etc. Afterwards, UXers start to build specific MVP.

When building MVP, Lean UX encourages UXers to follow Lean principles, like reducing waste and increasing efficiency. Just-in-time and Standardization are the two most out-standing principles employed. The activities UX at these phases are designed following these principles.

First of all, UX team defines what features to build for this MVP. On Agile projects, the development team is devoted to few new features at one time, rather than re-build all the features each time. This is a reasonable solution originated from Just-in-time principle. Out of the same principle, the UX team does not have to work on all the designs in a release at one time, but the most important designs.

UX team then devote themselves into the regular material-making work. This process is quite various, from person to person, and from idea to idea. There is hardly universal formula or routine about it, but some useful kits could be concluded and implemented. A basic UX tool-kit is a set of UX elements libraries, usually consist of 4-5 pages of basic buttons and other fundamental elements, 5-10 sample pages of universal page, visual design and content guidelines like color palettes and fonts.

When UXers build material, they create draft in mind or physically, then build the low-fidelity MVP correspondingly by drawing these drafts digitally via just putting materials from the tool-kit. This innovation is able to largely reduce the repetitive workload caused by repeated designs. Moreover, this UX tool-kit is widely accessible and under source control which allows the UX team to modify.

Here is the existing example, the prototype software-Axure, explaining the primary level UX tool-kit. The lower left quarter is the library of the essential elements, such as check boxes and radio button. Just put some sample to the page, the static prototype could be completed graphically. After this prototype is published, these graphic elements are converted to limited coding and get a dynamic prototype, the embedded feature of specific sample could be realized in dynamic prototype.

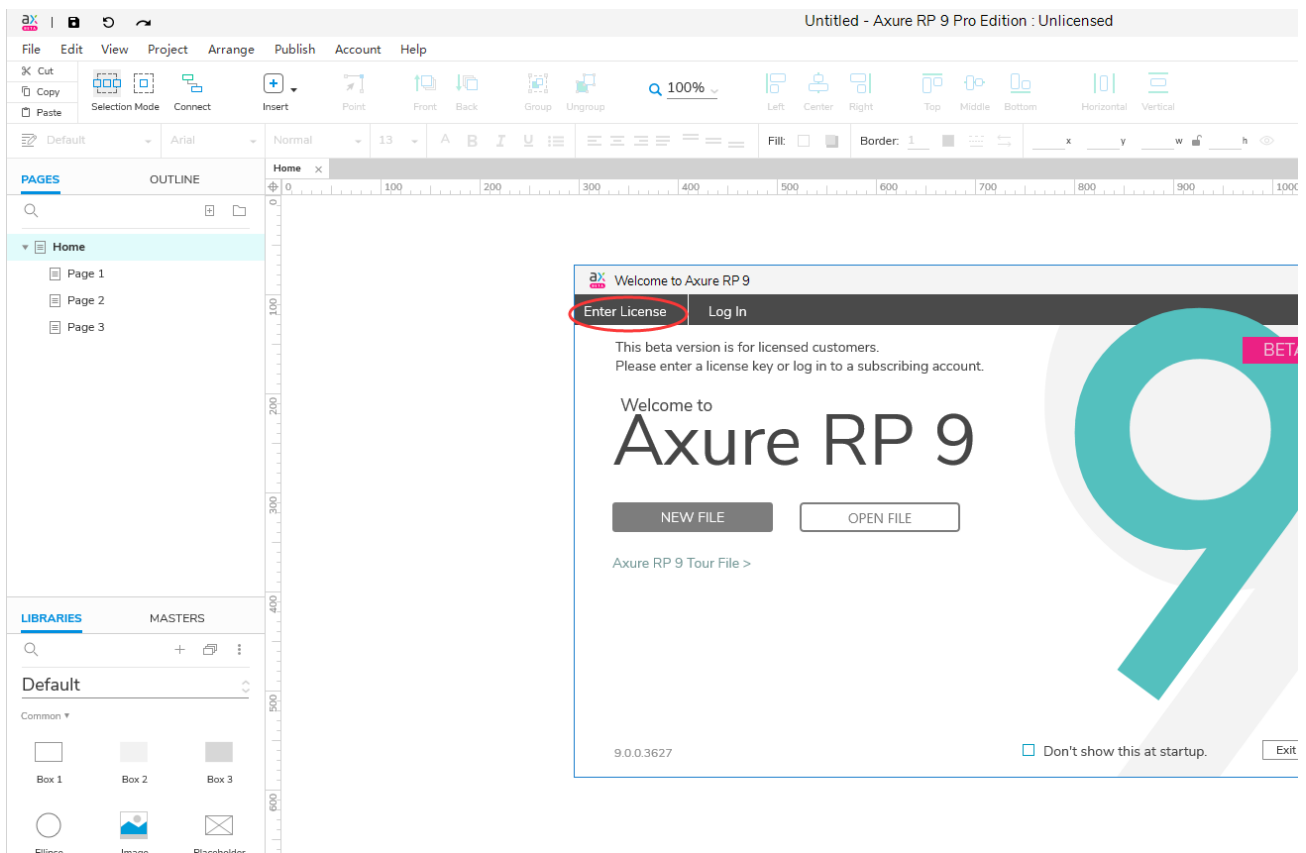
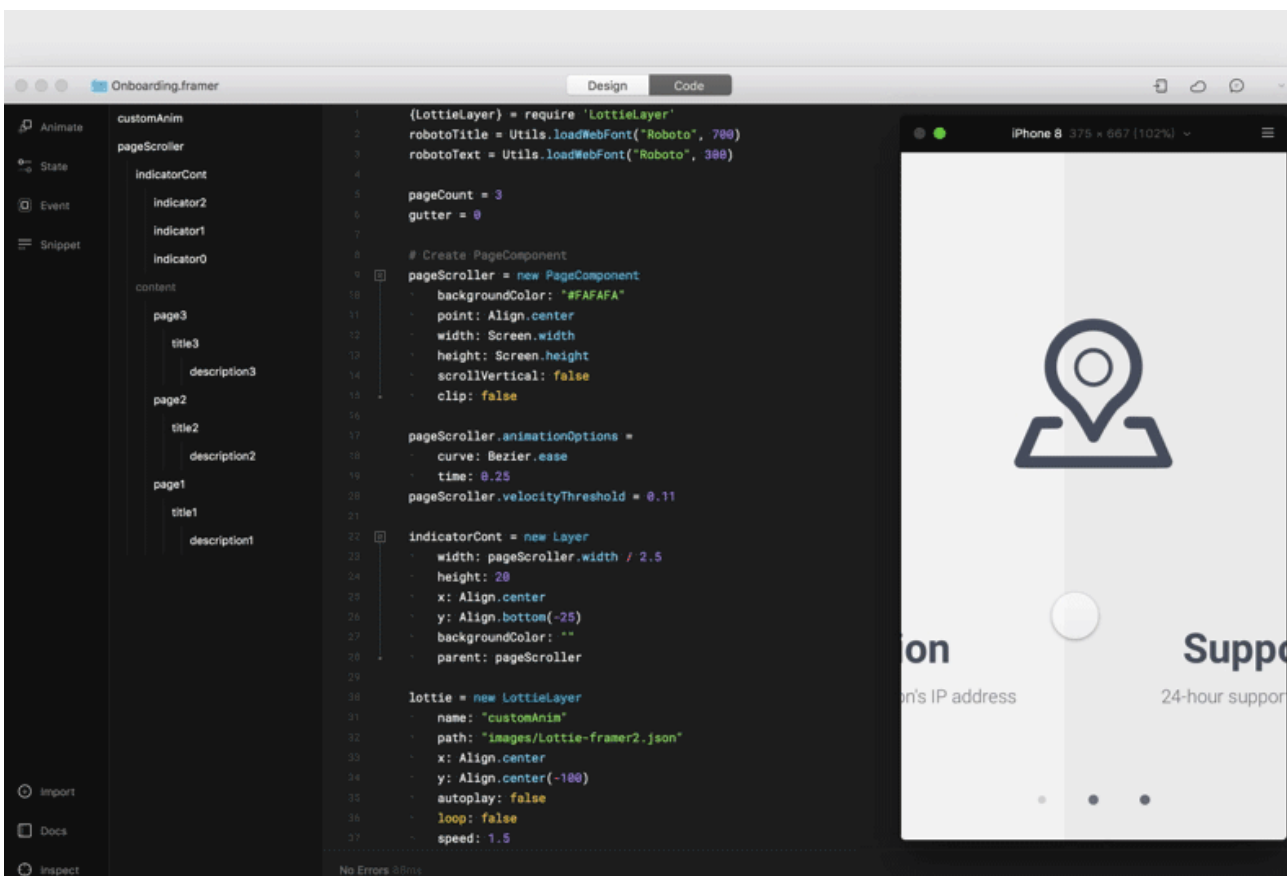


Figure 4.3.2-1 Screenshot of Axure RP 9

In the future, the tool-kit is expected to get much stronger and stronger. On the one hand, the conversion between design and coding would be smoother and faster. Take Lottie as an example to explain. Lottie is a conversion software for making animation used by designers broadly. It solves the complicated problem of animation delivering in a very smart way. It allows the designers to complete one animation in After Effect, then translate it into json documents using the plug-in software Bodymovin. What Lottie does is to render this json again and finally we get a new json document, which could be directly inverted into the coding structure by developers. Lottie is a well-developed software. It builds the bridge between graphic design and coding. The technique has been developing, which provides confidence for the conversion expectation.



*Figure 4.3.2-2 Screenshot when export animation in json data format for Lottie to use*

On the other hand, the AI (artificial intelligence) techniques, as the most important technical development of this era, is of brilliant value in many ways. One example is the Luban banner robot developed by Alibaba. In 2018 the 1111 Shopping Festival, Luban completed 4 billion banners individually. If one banner costs 20 minutes, 100 designers have to work for 152 years without sleeping or eating. Luban is trained to pick suitable layout for the banner and export that, while what human do is just limit what products to promote and wait for a while.

This is the Official website of Ali

Luban: [https://www.aliyun.com/product/](https://www.aliyun.com/product/luban)

luban?spm=a2c4e.11155472.1280361.309.62e635bfjEDRIc



Figure 4.3.2-3 Banners made by Luban

Moreover, there is an open source program replacing part of programmers' work: this program is able to automatically create corresponding interface coding to some interface picture exported. The link of this open source program is: <https://github.com/tonybeltramelli/pix2code>

These two examples of AI could in some extent prove that the AI technique is developed in an extremely high speed and have the potential to realize design as coding. In the recent future, there must be more AI products working as tools to assist UXers to complete the MVP building in a more efficient way.

With the help of all kinds of UX tool-kit, the MVP would be built in time and delivered to the development team.





## **Deliverables and release**

In this phase, the MVP built is transformed by development team into real product and is about to be released into test or even into market directly. The role of UXers in this process is really simple: deliver the MVP to developers. The bottleneck is not related to UX, but development team. It seems that from this moment on, it's none of UX's business. In fact, there are still several works involving UXers, and Lean principles would play important roles in directing these.

In the classic agile software development, teamwork begins from Product Owner (PO) writing story, instead of problem statement or hypothesis. When the PO writes a new story, he first in advance with the UX team to get an initial idea about the iteration. Taking the new UX journey into consideration, PO writing stories is the very next step after getting MVP according to hypothesis. When the story is chosen for development in Sprint Planning, the design is already finished, which is the MVP. Except for that, it deserves great attention on task disassembling & assignment, and general version management.

Usually the project team follows a classic matrix management structure: each department is managed by the department leader individually. In addition, members from various functions form a sub-team for the project. They are guided by the project leader. The department leader takes control on department members' KPI and profession mentorship. Comparing to that, the project leader is much less powerful, usually he merely manages the task assignment and assist inner cooperation within the project team. This classic structure is developing gradually. Till now

the modern management structure turns more reasonable: it decentralizes the power of department leaders to project leaders while the human resource structure remains. Simultaneously the project team members are encouraged to work close to each other in space. And they get evaluated by the project leader directly, rather than the department leader.

This new arrangement would result in a more frequent communication among project team. One benefit is the team members are able to explain the deliverables more clearly and even save much more time. In most cases, members had to spend lots of time writing the deliverable documents, to explain the context and content of the deliverables. Took the new arrangement, it is not necessary for both UXers and other members to waste time explaining all the tiny details of the deliverables, and avoid reworking due to some misunderstands.

Meanwhile, from a higher view of point, the whole company is divided into several cells according to the project lines. Each cell runs in their own pace individually but simultaneously, and does not have an influence on each other. Consequently, the overall deliverable cycle time is defined by the longest cell cycle time. In order to reach the ideal structure and shorten the overall cycle time, project is ought to be defined as concrete as possible, like a fixed feature, like a fixed tab-page, depending on the existing software structure.



## **Test and research: diverse test candidate**

The test and research conduction is various, according to the project character and project cycle. Not all the companies tend to conduct test and research with customer, but there are still several cases where the test and research is essential, such as the to-Business projects, such as the consultancy projects, etc. These projects usually have more freedom in time when iterate comparing to those projects running in line at present.

When there is not too much pressure about time, the capable UXers are ought to maintain several fidelity levels of prototypes: low-fidelity for quick expression when collaborate and validate the current proposals, high one as the output which based on the former one. After conduct test and research using the low-fidelity prototype with various test candidate, the UXers iterate and get the updated high-fidelity prototype based on experience from the low one. In a word, “Two parallel tracks”: iterating the design and implementation separately, but simultaneously, facing future generations (Sy, 2007.). As soon as UXers reach a relative satisfying prototype, they would deliver it to development team and transform it into real product.

Practically speaking, the test and research varies from each period of the whole project. For example, the design interns could support the earlier phase, while in the late phase close to final release, there must be some expert real users involved in the testing. The customers for testing does not merely refer to the real users. According Desirée Sy, 2007., the suitable testers include the Interaction Designers who have adopted Agile very well. The test and research methods vary from the periods either, from the basic customer survey, to co-creation workshop,

depending on the practical situation and demands.

Many companies are still using the “voice of the customer” model. In this model someone represents the end customer. It made sense in the past, when collecting data was hard. But nowadays it is just another waterfall residue. In fact, in the most cases, the projects face the heavy pressure of releasing soon. So UXers conduct the research only once in the very beginning of the project, and after that few UXer team would test again. The project team would release the new version directly, as soon as the development ends. From another point of view, this kind of projects often is equipped with the data analyst who is responsible to monitor the customer usage data, to analyze their behaviors, to conclude their habits, and to discover the key “funnel” of usage about this app. Even though data analysts do not work for UX in particular, their work could serve as the test and research process. In another word, data analyst takes the real customer usage as the test and research, then analyze the hidden pattern as the research conclusion.

The biggest difference between this and usual test and research is, in usual test and research customers tell their feedbacks initiatively, while in data analysis customers “tell” passively. It’s more efficient to relying on the data analysis, explore the signals of segments rather than concentrating on individual, specific users. These feedbacks discovered from the data analysis are of the same value to the product iteration as the real feedbacks.

Meanwhile some techniques used in test and research could be used in this case as well, such as the A/B test especially.

### 4.3.3 Post-Phase (Iteration & Management)

In the last phase, the development team is ought to be ready to release the product into market, no matter how satisfying the test result shows. With the help of Market and Promotion department, the product goes to market. At this moment, regardless of whether the product has been tested or not, it would be tested by real customers. From this moment, it's data analyst play an important role in monitoring and guiding route to the right development.

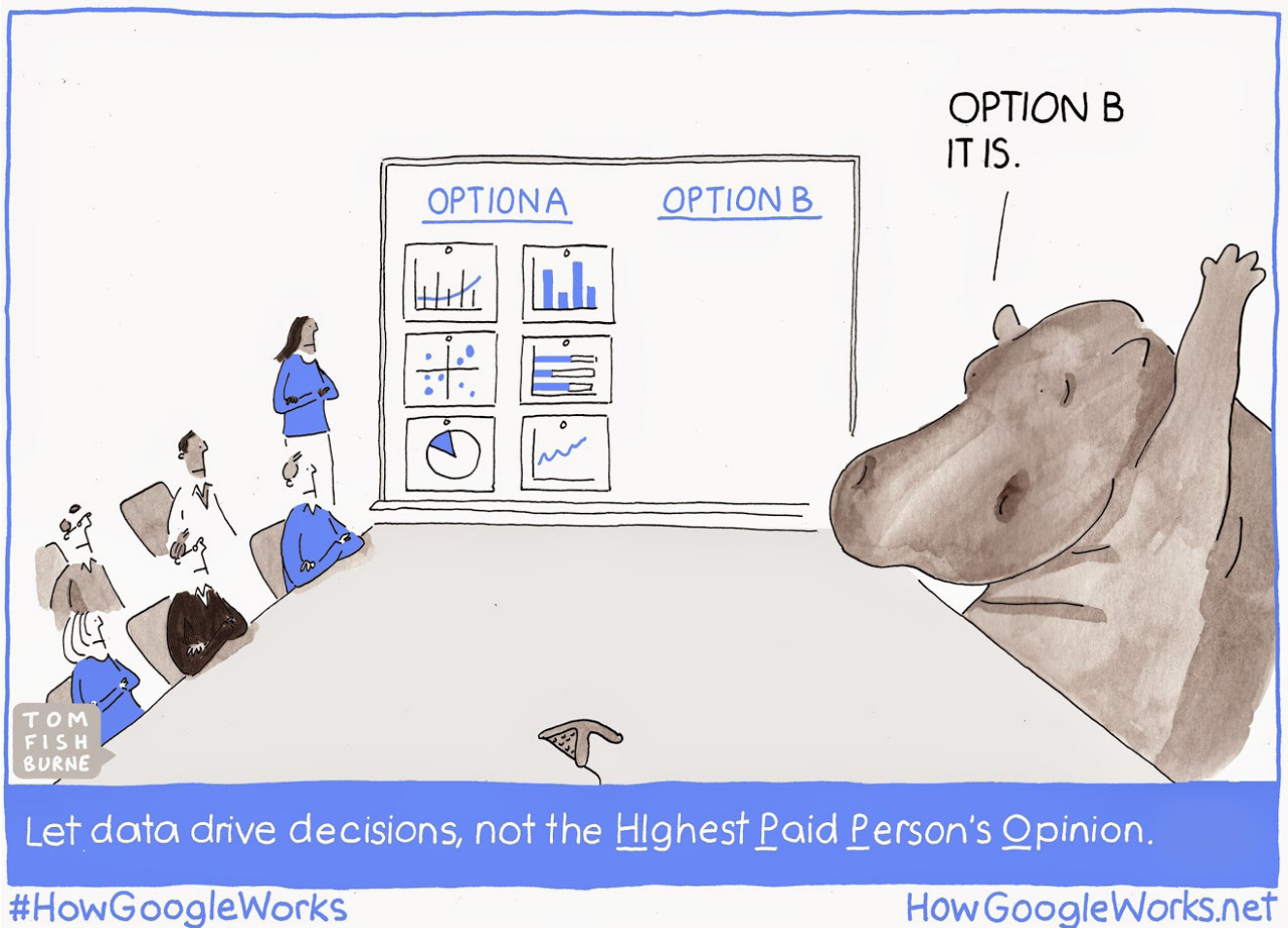
The data analysts acquire enough conclusions by measuring customers behaviors. Perhaps they could discover some flaws in the interaction design, perhaps they distinguish that some demands are out of team's imagination, instead of customers' need, perhaps they find out hidden demands not met before. These conclusions have great effect on planning the next iteration. UXers and Pos are likely to get inspiration from that, take them into consideration when modify prototypes, plan for the next iteration.

The key in this phase is to arrange the whole sprint workflow among different functions. The new Lean UX encourages iteration, each workflow is recycling, not a one-time solution. In this way, the whole development team could work like a production line, running automatically and repeating the proficient workflow without disruptive change. The concrete workflow in this phase is: when version No.1 has been released into the market, the data analyst is analyzing data from No.1. Meanwhile the developers are coding on version No.2 which is just delivered from UXers. This coding process usually lasts for a long time. During this period, the data analyst would discover

some problems and reply to UXer and PO in time. Consequently UXer and PO are going to make hypothesis and prototype for version No.3 in order to solve these fresh problems and add value for customers. It seems wasteful to leave these fresh problems to version No.3 rather than No.2. But what matters here is the No.2 is used to solve problems that discovered in No.0 which is released earlier than No.1. Even though there is one-version-gap, the iteration cycle is shortened by quick-response and remote-planning.

To ensure the iteration is strongly supported, it is necessary to implement the corresponding mindset and measurable criteria. The outdated Lean UX is adopted in the “waterfall” agile context where the decision is made top-down. Resulted from the eager to efficiency, the whole development team works like a feature factory, not focusing on value-creation too much. Marty Cagan highlights the huge missed opportunity of feature factories: “teams are just there to flesh out the details, code and test, with little understanding of the bigger context, and even less belief that these are in fact the right solutions.” Correspondingly the new Lean UX promote the OKR (Objective, Key Result), the goal setting framework, according to Marty C., 2016.

OKR can replace the HIPPO (Highest Paid Person’s Opinion) with experiments that allow the team to learn and iterate. This is hopeful to transcend the team’s mindset, from the traditional improving software development through lots of practices, to achieving business agility and continuously delivering real value to customers, according to Felipe C. and Alexandre F. K., 2017.

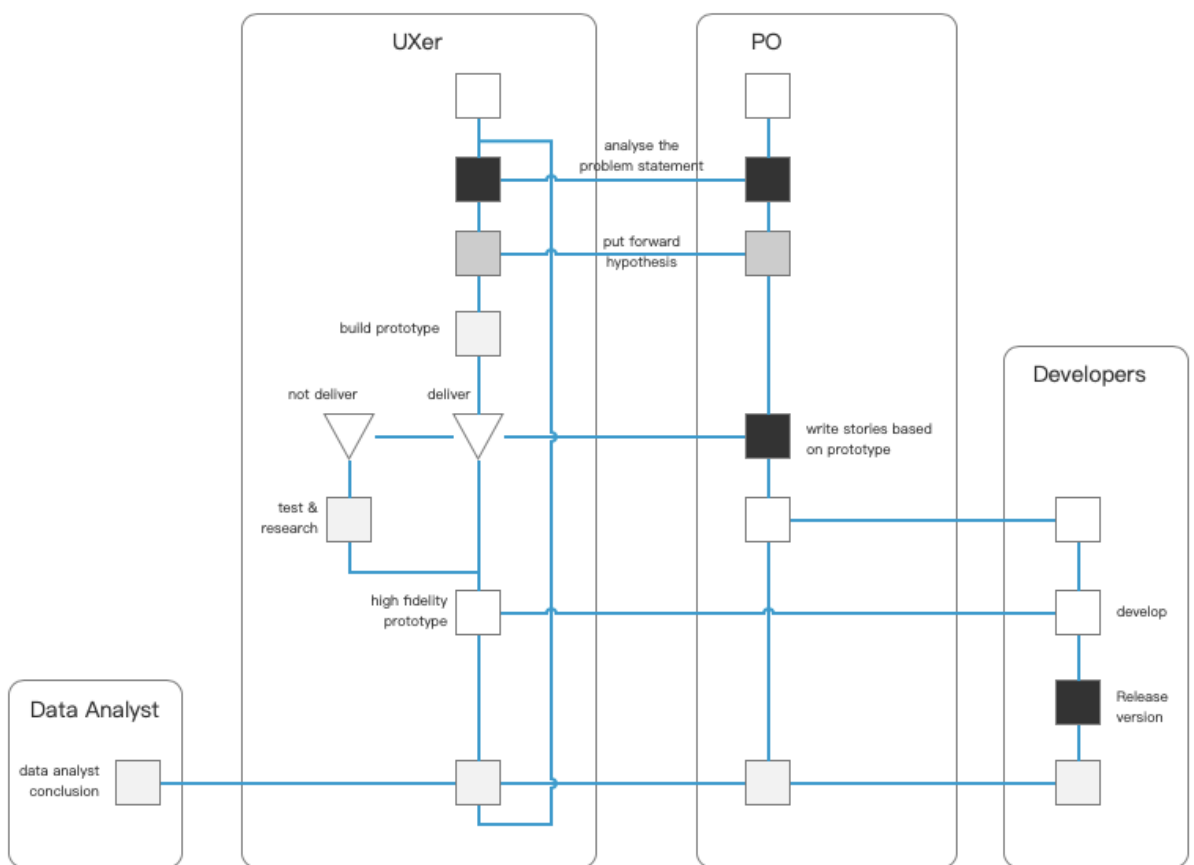


*Figure 4.3.3 How Google Works  
Adapted from the book of the same name*

Simply speaking, OKR sets the business agility to each team, sub-organization for one period. This is quite useful for the team cause team members would have much more motivation and decision power if they have the responsibility for the statistic result. They have the voice in interpreting the problem statement which would turn into hypothesis with other factors.

# 4.4 Conclusion

In order to interpret the whole management in an organized way, this section would express the conclusion through “Role Activity Diagram”. RAD is a normal mapping method used in management. It exhibits the whole organization activity by roles. In the specific graphic view, each character represents a single activity or process. For example, the square represents an activity while the triangle represents the contrary tunnels. This map is ought to read from top to down, line indicating the flow direction and iterate direction.



**Single Project Team (Squad)**

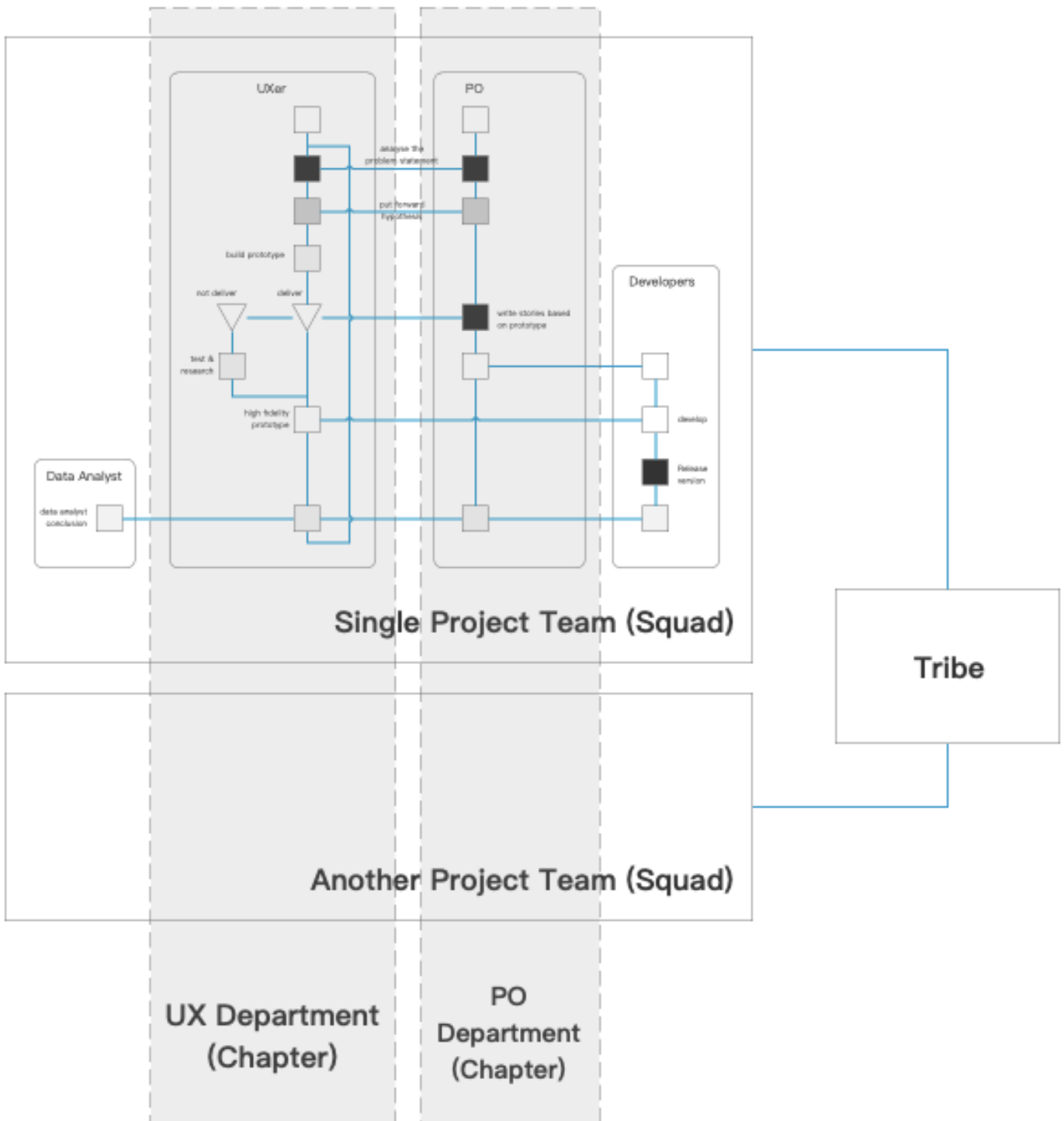
*Figure 4.4-1 The Role Activity Map of single project team*



In this map, there are mainly four roles: UXer, PO, developer and data analyst. Through this RAD mapping, it could be told that the UXers are in the center of this line of activity. The UXers and PO work together on the problem statement and hypothesis in the beginning of the line, then UXers complete the prototype individually. Under Lean principle, the deliverable-making is provided with several suggestions on how to prepare the universal tool-kit for that, so as to reduce the time and energy spent on the repeated UX work.

There is a fork of the road where UXers make a decision on whether to conduct test & research or not. At the same time, PO is ready to write the stories which would guide the developer team to do the development sprint. After developers release the software into market, it's data analyst working in the following step. Problems, potential demands and trends would be fed back to UXer and PO by data analyst, and they would be used in next cycle for problem statement and hypothesis.

Let's look at the whole organization from a higher level. It is essential to modify the mindset and adjust the organization structure. The first RAD map has ensured that each project team could work effectively like a skilled Toyota production cell. However, the purpose of Lean UX or Lean software is not to transcend the company into feature factory, but to make full use of development team's capability, either professional skill or working motivation. As a consequence, the new Lean UX encourage the OKR system to measure and motivate project members. Meanwhile the organization is re-arranged, focusing on the format of sub-team, emphasizing the team cooperation. The channel of communication among similar departments remains, but its role in KPI is weakened.



*Figure 4.4-2 The role activity map of the overall company*

In the next chapter, two interviews about the new Lean UX would be introduced. The interviewees are experienced Internet professionals, and they are working in Chinese top Internet companies. They have much experience achieved from practical development. These experience would be of value to adjust and improve the new Lean UX methodology.

# 05

## Interview



# 5.1 With Han Li From Tencent Technology, China

Han Li, is an Interaction Designer (UXer) of Tencent Technology Company, China. He is one of the team member of WeChat project, whose DAU is about 700 million. WeChat is consist of many sub-projects, such as Digital Payment, Public Account, Immediate Message, etc. Within each sub-project, there are several teams consist of various departments, flexibly developing around some feature. Han is working as the design department leader in one sub-project.

The extract of the interview is following, focusing on 8 questions:



Q1: What's User Experience, from Interaction Designer's point of view?

A1: In my opinion, User Experience is less than User Value. Meeting User Value is essential to any product. It's none of sense to discuss experience without user value. The 12306.com is a classic example (12306.com is the only official website for train ticket in China mainland). 12306.com is dissed for its terrible user experience in the very early period, but with pretty high user value. The fact is, users might curse it while visiting, and they have to buy tickets via it anyway. When some industry or product appears, it is pretty hard to define the value accurately. You should focus on the value before you are capable to satisfy users. Upon the market or the product become mature, and all the essential user values are met, it turns right to look at experience.

Q2: What is the role of User Research in the dev team? How does it cooperate with product manager?

A2: It depends. If it were a startup-style product team, there won't be the full-time user researcher. In this situation, user research is taken by the product manager, together with some other key members sometimes. When the product grows, becomes bigger, the user research team starts to join. My explanation for this is that, in the early develop period, the user research is more qualitative, most decision is made based on intuition. In the mature period, user research is quantitative, more rational decisions are needed.

Q3: If you find the direction is wrong, are you able to twist the direction fast in time?

A3: Well, this is a good question. What we do in WeChat is to assign few teams on the same direction at one time. This might lead to many colleagues become cannon fodder, but this is necessary. You know, in the Internet industry, everything is changing fast. We get no examples to learn from. Nobody knows what would be right. Then let's run forward together.

Q4: How do you measure the final effect of the product?

A4: The fundamental standard is, does it meet the promised purpose. For example, does it guide the users to the next page successfully. This kind of purpose would be quite objective. Another kind of standard is the response of customer satisfaction. And this measurement is ought to be compared with history result, rather than analyzed the individual value.

Q5: As an interaction designer, how do you communicate with product manager and programmer?

A5: Quote an old saying: Know the reason, move the emotion, induce the benefit and bring it to justice.

What does it mean in practice? Firstly “know the reason” is to agree on the same vision with others, making use of statistics, research, reasons or values. Secondly “move the emotion” is to build decent personal relationships. For example, if you treat the programmers as bro, invite them to hang out at weekends, they might be nice when work with you. Thirdly “induce the benefit” is simple, sometimes the extra wages from project budget is the strongest motivation. Lastly “bring to justice” means, if you have the power to make suggestion on performance grading, you would also have the power to motive others assist you.

Q6: Are there any mistakes that designers tend to make?

A6: Sometimes, designers would like to add the fancy elements when they learn some new techniques or trends, but in return they might become the bottle neck in project schedule. A real brilliant designer would recognize the period when user value is more important than user experience. So that they would provide more simple but effective proposal, rather than showing off design techniques.

Q7: Have you ever applied the Lean UX? Are there pain points during it?

A7: Yes, we are using lean ux in our own way, mixed with agile development. We do need to compress the work cycle of UX. The traditional workflow transfers UX into simple interface or graphic design due to the tough time pressure. I’ve ever stayed up very late for 3 days, to complete a radical interface change of one main feature. But the effect is just so-so, the usage of that feature doesn’t increase as we expected, and the executives are disappointed at this project. From that moment I realized that my value of existing in this

team is always to assist the team to discover user value. I was blind by the busy interface work and forgot that reason before.

It's not very easy to grasp the usage of Lean UX actually. In the very beginning we find it hard to adopt Lean UX in our work because it's too abstract for us, could not be taught by a coach. So we UX designers made the attempts gradually, starting from few roles (PM) and limited features. Then take the not-bad result to persuade the whole dev team.

Q8: Do you have suggestions on the Lean UX methodology introduced in this thesis? Is the methodology useful to solve any problem you encountered before?

A8: I read the chapters about this methodology roughly, and I notice some details mentioned in it too. I have to say the methodology introduced is very similar to the actual methodology we used now. But it involves some creative ideas in addition. For example, I appreciate the idea of UX kit, this is exactly what I have been wishing to advocate among our company. UX's work, to a great extent relies on the UI design. So I believe the design kit of the UI design worth to be transferred like a heritage among the design department. Expect for this, I also have a concern about the cycle schedule. In the map I saw the UX team is one cycle earlier than the other roles. In fact this might be dangerous, because the information about problem statement is not so accessible. There are always something happening out of our expectation. So the big lacking in this map is some necessary buffer work and time between UX and others' work cycle.

## 5.2 With Haonan Li From Baidu Technology, China

Haonan Li, is a Product Owner (PO) of Baidu Technology Company, China. Baidu is Chinese Top Internet Company, starting up with its search engine, like Google. DAU is about 1.5 billion in 2018. Baidu has a giant ecosystem, made up of various products. Within each product project, there are several sub-projects, focusing on concentrated features. Haonan is working as the project leader and product owner in one sub-project.

The extract of the interview is following, focusing on 7 questions:



Q1: What's User Experience, from Interaction Designer's point of view?

A1: PM is playing different roles in various companies.

At giant company, PM might be actually product marketing, responsible for market research, defining customer demands and offering general product schedule. The specific interaction schedule is provided by UX designers, while PM is ought to identify the priority of these interaction tasks. Then UX designers start the interaction work following that priority.



Different from giant company, at small and midsize companies PM usually focus more on the practical execution. They would spend less effort on demands discovery, but more on the feature planning and interaction, which means they are taking the role of interaction designer to some extent. Basically, there would not be special UX designers but UI designers if the company with fewer than 500 employees.

Q2: Have you ever applied Lean Software? What pain point that you met?

A2: In general, the pain point is how to build the communication channel among various departments. During lean software, it's necessary to have super-fast statistical analysis and strong-trust between. Only in this way can lean development be efficient. Otherwise the communication problem might result in greater time waste.

Q3: How do you change your development direction when realize that you may be wrong?

A3: Anytime you could "pivot" largely as long as your team could accept even reluctantly. This kind of decision relies on whether our product could meet the market demands.

Q4: How do you measure the final effect of your product?

A4: Depends on the prior purpose of the product. If the purpose is low customer acquisition cost, our KPI should be DAU, retention and user activation unit price. While if the purpose is product reputation, KPI should be Propagation rate and exposure. And so on.

Q5: What's User Experience, from PM's point of view?

A5: There are several routines within the products for customers to reach their purpose. But as PM, we are ought to promote that routine which with highest ROI. Here the Return represents how well the customer meet their demand, Investment is how effort it needs for that behavior.

A good User Experience represent that this ROI is very high. For example, the operation of Excel (Microsoft) is complex, so its Investment is high. Meanwhile its Return is high too, the effect works well. To conclude the user experience of excel is good. In contrast, the operation of Numbers (IOS) is quite simple, but its Return is not enough because its effects are poorly simple. In general, the user experience of Number is more terrible than Excel, for me.

Q6: How do you persuade the UXers and programmers when working together?

A6: Mainly two keys:

1. Deeply understand their work, their techniques. For example, remember clearly the font size of each interface, know more about Interface design principles, and learn of the principle of R&D coding work. Only in this way, you could win the respect of others, so that to improve the efficiency of communication largely. After all everyone hates layman guidance.
2. Feel for others, and try to find out their difficulties and pressures, and solve those for them without affecting the outcome of the output.

Q7: Do you have suggestions on the Lean UX methodology introduced in this thesis? Is the methodology useful to solve any problem you encountered before?

A7: I appreciate the Lean UX methodology very much, especially the revised workflow map. (Actually, it's the second RAD map he refers) During work, I deal with several work lanes for different small projects at the same time. It's not easy to make a balance between these, I have to say. The idea of rearranging them into different cycles deserves a try.

Moreover, it's very common that the execution levels have some sudden inspirations which might differ from the planning we made earlier. It's exactly the HIPPO case you introduced. It's not too easy to introduce OKR into companies, unless top-down. As a consequent, the regular workflow might be challenged by the HIPPO case from time to time. I suggest you think out other make-up plans for this case. I know it must be pretty hard.

# 06

## Conclusion Engagement

A decorative graphic element consisting of multiple horizontal, wavy bands in a light teal color, positioned on the right side of the page and partially overlapping the text.

# 6.1 Strength & Limitation

This research is interested in how User Experience team innovate themselves to integrate into development team and play greater value in software development. The corresponding solution is to introduce a new theory of Lean UX which combines various theory, extracting the essence and discarding the dross.

This thesis is developing from two directions. On one side, it tracks back to the origin of Lean theory. On the other side, it analyzes all the related theories about User Experience. Besides, the thesis takes theories about dev team management into consideration. With much struggling effort, a potential balance point of these theories is reached, which becomes the draft ideas of the methodology. Afterwards, a deeper research on the mechanism of existing Lean UX is conducted. Based on the experience and inspiration of this classic Lean UX, the draft methodology grows into a systematic methodology, including management of insides and outsides of UX team, the guidance of generic UX workflow and specific delivery.

Different from the Lean UX methodology that has been public, the new methodology in this thesis is more systematic. Its scope is broader, ranging from general program to specific delivery details. Due to the academic background of the author, this paper learn experience from management theories as much as UX. This brings a more professional and objective advantage to this paper comparing to public Lean UX.

It has to be admitted that there are still many limitations of this thesis. First, the selection of the literatures during the theory research. Second, the backstage design of the output delivery from UX to other stakeholders in the workflow. Third, the management support to push implementation within a company, such as persuading the management levels to adopt.

## 6.2 Future evolvement

The purpose of future evolvement is to propose ways to resolve its limitations, and perfect the specific methods. From this point of view, more researches on various literatures are needed, such as how the statistic affect UX Design Change, how the statistics have influence in Agile development. Besides, the process of the UX assumption delivering and daily delivery requires more careful deliration and more detailed interpretation. Moreover, more cases studies on Internet company management are worth doing. This time, the focus of case study would shift to how the management level and dev team negotiate and agree on the project strategy.

In the end, with the development of technology, especially AI techniques, the effort on transform technology into Lean UX toolkit would continue.

If it permitted, the methodology theory in this paper should be experimented in real projects. There would be more limitations or even mistakes showing up.

# 07

## Reference





# List of figures

Figure 2.2.1-1	The overview process of waterfall methodology	27
Figure 2.2.1-2	Comparison between Waterfall and Agile	28
Figure 2.2.2	The process of Agile methodology	32
Figure 2.2.3	The overview process of Scrum development	35
Figure 2.2.4-1	Comparison between Waterfall and Agile	37
Figure 2.2.4-2	Connection of Waterfall and Agile	38
Figure 2.2.5	Subordination of Lean and Agile	40
Figure 2.3.1	The non-linear process of design thinking	46
Figure 2.3.2	The UX pyramid	50
Figure 2.3.3-1	Tool list used in UX design practice	51
Figure 2.3.3-2	Double Diamond Design Process	53
Figure 2.3.4	Coordinate of research techniques	56
Figure 3.1-1	The value stream map of basic UX workflow	60
Figure 3.1-2	The value stream map of scrum flow	63
Figure 3.3-1	Screenshot of AutoCAD, representative Autodesk product	67
Figure 3.3-2	The overview workflow of UCD process	71
Figure 4.1	The value stream map of new Lean UX	94
Figure 4.2.2	Customer Journey Map of the original UX process	100
Figure 4.3	Customer Journey Map of the developed UX process	103
Figure 4.3.2-1	Screenshot of Axure RP 9	109
Figure 4.3.2-2	Screenshot when export animation in json data for Lottie	110
Figure 4.3.2-3	Banners made by Luban	111
Figure 4.3.3	How Google Works	119
Figure 4.4-1	The Role Activity Map of single project team	120
Figure 4.4-2	The role activity map of the overall company	122

## Book

Liker, Jeffrey K. and Michael Hoseus, 2008, *Toyota Culture: The Heart and Soul of The Toyota Way*

Henrik Kniberg, 2008, *Scrum and XP from the trenches*

James R. T., Guy B., and Alan S., 2009, *Lean-Agile Software Development: Achieving Enterprise Agility*, Chap.1. *An Agile Developer's Guide to Lean Software Development*.

Mary P. and Tom P., 2003, *Lean Software Development-An agile toolkit*

## Journal Article / Thesis

Desirée Sy, 2007. "Adapting Usability Investigations for Agile User-centered Design," *Journal of usability studies*, Vol. 2, Issue 3, May 2007, pp. 112-132.

Frank E., Konstantin H., Britta K., and Thomas I., 2017, *Rapid Lean UX Development Through User Feedback Revelation*, PROFES 2017, LNCS 10611, pp. 535–542.

Lassi A. L., Harri K., Lauri S., and Miko H., 2014, *Lean UX - The Next Generation of User-Centered Agile Development?*, Urho Kekkosen katu 7B, FI-00100 Helsinki, Finland.

Greg N., 2018, *Lean UX Communication Strategies for Success in Large Organizations*, Baker Hughes GE Digital.

Mary P., Tom P. and Michael A. C., 2-12, *Lean Software Development - A Tutorial*, Massachusetts Institute of Technology.

Henrik K. and Anders I., 2012, *Scaling Agile @ Spotify with Tribes, Squads, Chapters & Guilds*

## Web Page

Page Laubheimer, 2017, “Agile Is not Easy for UX: (How to) Deal with It”, (From <https://www.nngroup.com/articles/agile-not-easy-ux/>)

Amanda Stockwell, 2016, “How to adapt UX research for an Agile environment”, (From <https://uxmastery.com/how-to-adapt-ux-research-for-an-agile-environment/>)

Marty C., 2016, “When Performance Is Measured By Results” (From <https://svpg.com/when-performance-is-measured-by-results/>)

Felipe C. and Alexandre F. K., 2017, “Transcend the “Feature Factory” Mindset Using Modern Agile and OKR”, (From <https://www.infoq.com/articles/transcend-factory-modern-agile/>)

Scaled Agile, Inc, 2018, “Lean-ux”, (From <https://www.scaledagileframework.com/lean-ux/>)

Interaction Design Foundation, 2017, “A Simple Introduction to Lean UX”, (From <https://www.interaction-design.org/literature/article/a-simple-introduction-to-lean-ux>)

Juni Mukherjee, “Value Stream Mapping - Learn how this analysis technique can optimize your CD pipeline.”, (From <https://www.atlassian.com/continuous-delivery/principles/value-stream-mapping>)

# Acknowledge

My deepest gratitude goes first to my supervisor Prof. Raffaele Boiano, for his patient and valuable instruction. He is really a niceeeeeeeee guy!

## **Sincere Thank to:**

Dear Mom & Dad for supporting me to study in Polimi!

My internship supervisor Han Li for his interview and suggestions.  
My internship supervisor and buddy Haonan Li for his interview and suggestions.

My dude Nicola Morandini for his super nice support and assistance.  
My BFF Yujie Fan and Sen Wang for their encouragements and supports.  
My colleague Sibe Chen for her inspiring me to apply for Polimi.

Chuqing Dong, myself, for walking through this bitter but interesting, worthwhile experience, without losing myself along the way!

