프로덕트 디자이너의 입장에서
요구사항 구체화와 최소 구현에 관해

최종 수정일: 2024-05-17

이지훈 / ZUZU product designer / jihoon@kodebox.io

ZUZU

요구사항 구체화와 최소 구현을 위한 마인드셋

- 요구사항은 어느정도 추상적이어야 한다.
  (너무 구체적이라면 확증 편향에 빠지기 쉽다.)

- 요구사항을 준 사람은 대답의 의무, 요구사항을 받은 사람은 질문의 의무가 있다.
  (질문이 없다면 의심할 것)

- 요구사항 전달 과정에서 마찰은 당연히 발생한다. 배경지식 차이가 있기 때문이다.
  (하나의 마찰도 없다면 의심할 것)

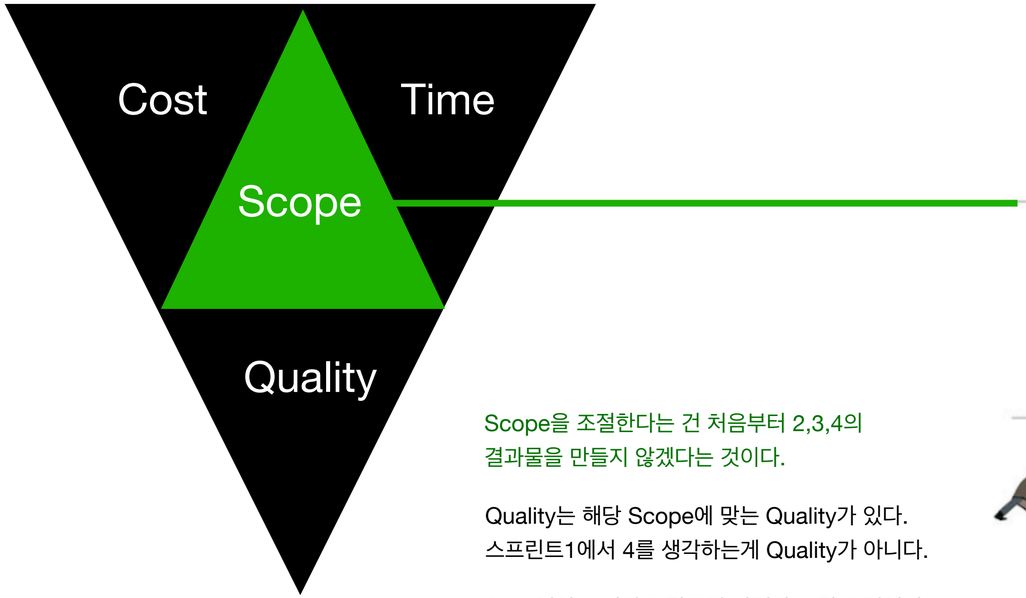
- 요구사항은 작든 크든 대부분 변경된다. (변경되지 않는다면 의심할 것)

- 우리는 대개 다른 이야기를 같게 하거나, 같은 이야기를 다르게 한다 = 한번에 100% 이해할 수 있는 상황은 거의 없다.
  (너무 쉽게 이해가 다 된다면 의심할 것)

- 위의 문제는 적어도 문서화만으로는 해결되진 않는다.

요구사항 구체화와 최소 구현을 위한 마인드셋

우리가 조정 가능한 것은 어디까지 구현할지(Scope)이다.
다만 Scope의 기준은 바퀴를 만드는 게 아니라 **굴러가는 무언가**를 만드는 것이다.
최소 굴러가는 무언가는 되어야 고객에게 의미를 줄 수 있고, 계속 만들지 아닐지 판단할 수 있다.

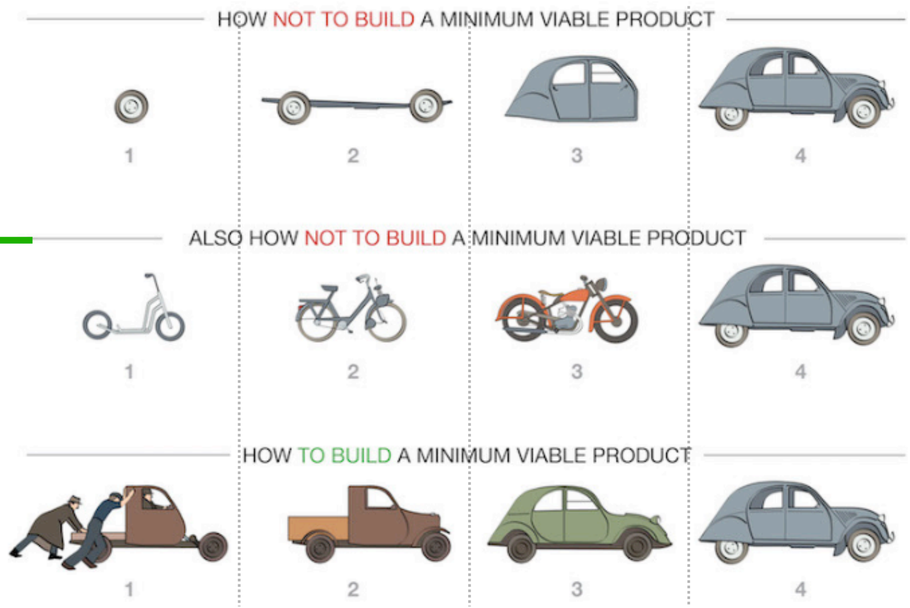


Scope을 조절한다는 건 처음부터 2,3,4의
결과물을 만들지 않겠다는 것이다.

Quality는 해당 Scope에 맞는 Quality가 있다.
스프린트1에서 4를 생각하는게 Quality가 아니다.

Agile에서 쓰레기를 만들지 않겠다는 건 고객에게
의미없는 걸 만들지 않겠다는 뜻에 가깝다고 생각함.

[https://www.linkedin.com/pulse/entrepreneurs-il-ne-sagit-pas-de-construire-un-produit-david-pinneau/]

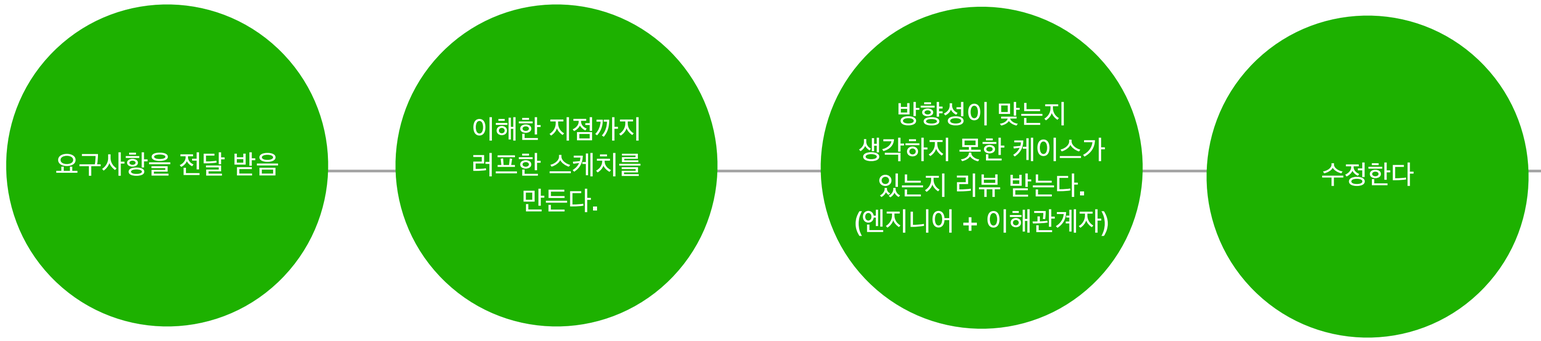


Sprint 1	Sprint 2	Sprint 3	Sprint 4
Scope	Scope	Scope	Scope
Quality	Quality	Quality	Quality

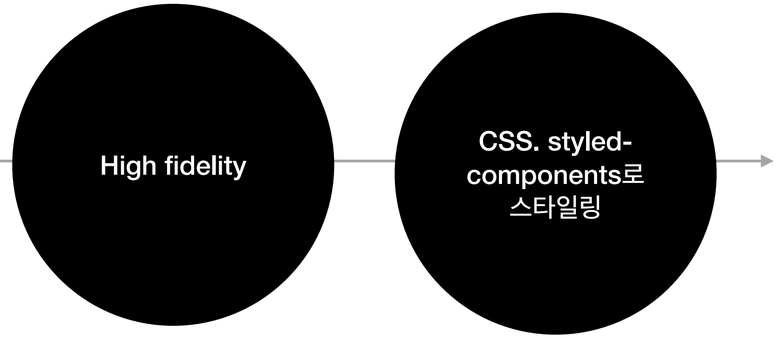
적절한 요구사항 구체화 & 최소 구현 파악 절차

High fidelity 디자인과 엔지니어링 전에는 얼마나 많은
Baseline을 쌓았는지가 중요하다. (뒤에서 나옴)

몇 번 반복하면 Agile에서 말하는 *acceptance criteria가 자연스럽게 만들어진다.
당연하게도 요구사항 구체화와 최소 구현은 동시에 되는 경우가 많다.



*하드스킬이 좋으면 이 시간을 줄이고 Baseline을 쌓는데
시간을 좀 더 쓸 수 있다.



이 과정을 통해서 최소 구현이 실마리가 잡힌다.
최소 구현 범위는 혼자 정할 수 없다. 디자인 x 엔지니어링 x 요구사항이 결합이기 때문.
모두가 구체화하고 최소구현을 같이 만들 수 있도록 시각화 하는게 우리의 역할.

엔지니어와는 구현 복잡도를 논의하기 위해서, 이해관계자와는 요구사항을 잘 이해했는지 파악하기 위해서 이야기가 필요.

추천 방법: 최적의 플로우를 생각해두고 빼는 형태로 리뷰 받는다. 처음부터 최소 구현을 만들 순 없다. 다 넣고 빼는게 더 쉽다.

*A set of predefined requirements that must be met to mark a user story complete

엔지니어링 과정에서 숨은 케이스들이 나온다. 해당 케이스가 실제로 있는지 없는지 이해관계자와 소통이 필요하다.

디자이너의 자유도가 보장됨.
최소 UI 퀄리티 맞추는 지점.

소요시간

소요시간

실패 사례

- **고객이 쓸 지 안 쓸지 모두가 확신이 없는 상태에서, 무의식적으로 많이 쓴다고 생각하고 만드는 경우**
  - 10을 요구했지만 20을 만들려고 하는 상황. 만드는 과정에서 힘은 많이 들지만 쓰는 고객이 없다.

- **목표를 이루기 위한 디자인이 아닌 디자인을 위한 디자인을 하는 경우**

- **러프 스케치 공유가 너무 늦은 경우.**
  - 정확히 어느 부분을 고치거나 수정해야 하는지 알지 못하거나 혹은 작게 뭘 추가해야 하는지 알지 못하는 경우가 많음 : 구체화가 덜 됨.

- **이해 관계자에게 고객의 유즈케이스가 아닌 디자인 결정을 묻는 경우.**
  - 이해관계자가 대답해줄 수 있는 건 (그리고 대답을 받아야 하는 건) 구체적인 디자인 결정이 아니다.
  - 특정 케이스를 대응하는 디자인이 추가로 필요한 경우, 그 케이스를 대응해야 할지, 대응하지 않아도 될지를 물어야 한다.
    대응해야 한다면 이번 스프린트에서 대응해야 할지 다음으로 넘겨야 할지도 물어야 함

실패 사례

- **요구사항 안에 답이 있을 거라고 생각하는 경우**
  - 요구사항은 숨은 그림 찾기가 아니기 때문에 그 안에 우리가 찾는 답은 없다. 우리의 역할은 요구사항을 고민하는게 아님

- **요구사항이 항상 옳다고 생각하는 경우**
  - 요구사항은 성역이 아니다. 목적을 이루기 위한 다른 수 많은 방법이 있을 수 있다.

- **요구사항 그대로 만들어야 한다고 생각하는 경우**
  - 요구사항이 구체적으로 왔는데, 효과적일 것 같지 않다면 더 물어봐야 한다.
    (이게 어떤 의미에서 효과가 있다고 생각하는지, 어떻게 나온 결론인지 등)
  - 어떤 의미에서 효과가 있다고 생각하는 지점이 뭔지 정확히 알면, 다른 요구사항이라도 같은 목표를 이루게 할 수 있다.
  - 요구사항과 목적을 분리하지 못하면 요구사항 그대로 만들고 결과물이 안 좋으면 누구의 잘못인지 찾기 시작한다.

실패 사례

- **High fidelity 전까지 엔지니어와 아무런 대화가 안 되는 경우**
  - 요구사항을 잘 구체화 했으면 엔지니어도 high fidelity 디자인을 기다릴 필요 없이 시작할 수 있는 지점이 있는 경우가 있다. (이게 시스템&패턴&컴포넌트 중심의 디자인을 하는 이유)
  - 만약 디자인이 없으면 구현 시작할 수 없는 상태라면 적어도 둘 중 한 명은 요구사항을 제대로 이해하지 못하고 있을 가능성이 높다. (혹은 요구사항 구체화가 충분히 되지 않았을 가능성이 높다)
- **엔지니어가 아무 코멘트 없이 디자인 그대로 만들 수 있다고 하는 경우**
- **요구사항을 주는 사람도 잘 모르는데 너무 빨리 만들어 버린 경우**
  - 보통 개발팀을 너무 믿으면 발생한다.

해결책

방법론을 몰라서 가이드라인이 없어서, 못하고 있는 것은 아니다.
방법론과 가이드라인은 필요하다. 하지만 없다고 요구사항 구체화나 최소 구현을 못 하는게 아니다.
방법론 만능주의에서 벗어나서 생각해 볼 필요가 있다.

해결책

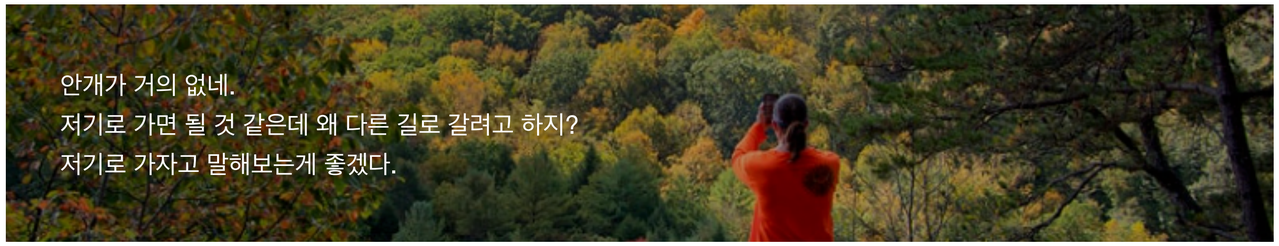
You need a brief PAUSE after hearing an issue. (A walk or something similar would help)

시카고대학의 미하이 칙센트미하이(Mihaly Csikszentmihalyi) 교수는 그의 책 '창의성의 즐거움'에서 창의성이 발휘되는 과정을 호기심, 아이디어 잠복기, 깨달음, 여과과정, 완성의 5단계로 설명한다. 과학저술가로 유명한 스티븐 존슨(Steven Johnson) 역시 이 유레카 순간을 만들기 전에 '인큐베이터 순간(Incubator period)'은 필수적이라 말한다.[2] 칙센트미하이 교수가 말한 잠복기와 깨달음 단계 사이에는 의식에 의해 정돈되지 않은 아이디어가(잠복기) 스스로 움직이게 되어 뜻하지 않은 결합(깨달음)이 만들어지는 것 처럼 보인다. 인식론자들은 이것을 "아이디어들이 의식적인 지시에서 벗어남(예: 샤워, 쇼파, 잠, 헬스장, 산책, 대화 등)에 따라 임의적으로 결합하면서 아무 상관이 없는 것처럼 보였던 아이디어간의 연결이 잇따라 일어난다."고 설명한다.[3] 샤워장이 무의식적인 몰입을 유도하고 이 몰입이 인큐베이터에 있는 아이디어를 깨어나게 하는 것이다.

해결책

2. Maintain a bird's-eye-view while sketching

Let's think of the issue as scenery: **Sketching is a process of clearing a fog.**



전달받은 요구사항은 어떤 목적을 이루기 위해서인가?

이 문제를 해결해야하는 시간이 1시간 밖에 없다면 어떻게 할 것인가?

이 문제를 해결하기 위해 1년을 준다면 어떻게 할 것인가?

세계 최고의 엔지니어와 함께한다면 어떻게 해결하고 싶은가?

세계 최악의 엔지니어와 함께한다면 어떻게 해야하는가?

목적을 이룰 수 있는 다른 방법은 없는가?

나는 이걸 왜 하고 있는가?

내가 세계 최고의 게으름뱅이라면 어떻게 해결할 것인가?

말이 잘 안 통하는 다른 나라 사람에게 이 요구사항을 설명한다면 어떻게 할 것인가?

이 문제를 해결하기 위해 100억을 준다면 어떻게 해결할 수 있는가?

나는 무엇을 가정하고 있는가? 꼭 코딩이나 디자인이 필요한가?

전달받은 요구사항을 제대로 다른 실무자
(디자이너, 엔지니어)에게 설명할 수 있는가?

해결책

3. Establish a BASELINE indicating the danger threshold

이번 스프린트에서 넘어가면 돌이킬 수 없는 지점을 만든다.
이 지점을 주기적으로 쌓아 올리며 이해관계자+엔지니어와 커밋한다.
바꾸면 큰일나는 지점을 여러겹 쌓아 올리면 결국 스프린트의 결과가 된다.

- 이 지점을 넘어서 (이전으로 돌아가도록) 변경되면 요구사항이 바뀐 건지, 파악이 덜 된 건지, 파악이 덜 되었다면 어떤 점에서 덜 된 건지 확인해야 한다.