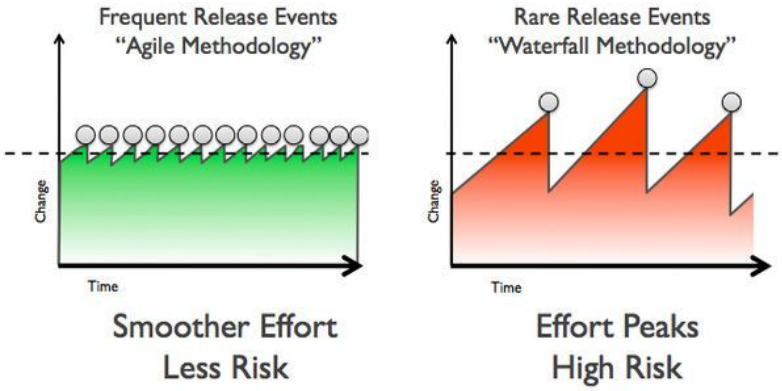
<패스트캠퍼스 애자일 특강>

작성자: 김영환

-저자 소개
박용표 odd-e 애자일 코치. 애자일에 대해 2006년 부터 강의하고 있음. 기업강연, 컨설팅을 주로 하며, 액티비티를 통한 애자일의 필요성을 전달하는데 초점을 맞추고 있음. odd-e 는 싱가폴에 본사가 있으며, 애자일 개발 방법론 전파에 집중하고 있는 회사임.

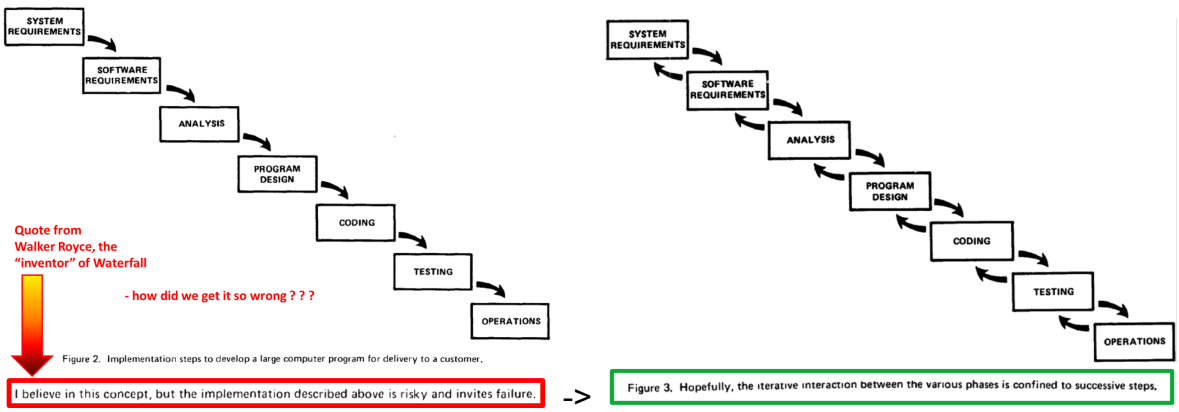
1. Intro.

애자일(Agile) 이라는 단어는 **민첩한**, **빠른**의 의미를 가지고 있다. 현재 개발업계에서 이슈가 되고 있는 개발기법을 의미하기도 한다. **애자일은 왜 등장**했으며, 구체적으로 **어떻게 하는 것인가?** 오늘 수업에서는 수많은 애자일 기법들 중 가장 유명하고 가장 많이 사용되는 **스크럼**(scrum)이라는 기법을 전달할 예정이다.



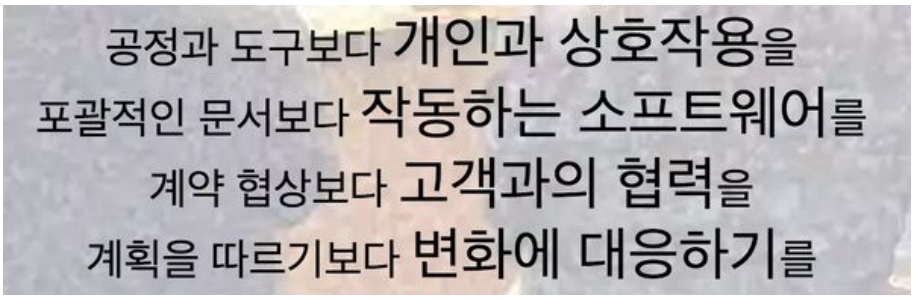
2. 폭포수 방법론의 단점: 변화에 취약함

기존의 수 많은 프로젝트, 특히 소프트웨어와 관련한 프로젝트들은 **50년간 폭포수**(Water fall) 방식을 고수해왔다. 폭포수 방식은 처음에 모든 계획을 세우고 분석해서, 이를 바탕으로 정해진 기간동안 프로젝트를 완성하는 방식이다. 관련한 논문이 있는데, 저자는 **이 방식의 위험성**에 대해 다음과 같이 언급하고 있다.



“워터폴 방식은 **외부의 변화가 없다는 가정** 하에 가장 잘 들어맞으며, 이를 그대로 실무에 적용하기에는 무리가 있다. (...) 따라서 **끊임없이 앞 단계로 되돌아가** 변화에 대응한 설계 디자인을 바탕으로 다시 제작이 진행되어야 한다.”

저자의 경고에도 불구하고 이 폭포수 방식은 미 국방성에 채택되면서 세상에 널리 알려지게 된다. 이 방식으로 개발하게 되면, 설계를 아무리 잘 해도 **시장의 변화에 유연하게 반응할 수 없다.** 또한 **개발자에게**도 기획이 변할 때마다 전체 시스템을 갈아엎고 다시 제작해야 하는 문제점이 발생한다. **변화는 곧 재앙**이 된다. 기획자와 디자이너도 많이 싸우게 된다. **문서만으로 생각을 주고받기 때문**이다. 완성된 제품을 보면 기획자들과 디자이너는 서로(의 반응에) 실망하게 된다. 또 한 가지의 문제점은 **피드백이 느리기 때문**에 **시장에 대해**, 그리고 **팀(이 얼마나 능력을 낼 수 있는지)**에 대해 **‘학습’할 수 없다.**



3. 애자일이란?

새로운 개발 방법론의 필요성이 제기되고, 젊은 개발자들을 필두로 **애자일 개발선언**이 대두되었다. 대표적으로 **켄트백(Kent Beck)**이 제안한 **애자일 방법론**이라는 **철학**이 있다.

- 지메일의 사례

애자일의 철학을 살펴보기 전에 지메일(Gmail)의 사례를 살펴보면 좋을 것 같다. **지메일**이 처음 등장했을 때 **사람들은 열광**했다. 그 이유는 **메일 용량이 1기가** 였기 때문이다. 지메일이라는 이름도 여기서 왔다. **당시 메일 용량이 20메가** 하던 시절이었다. 그래서 개인은 크게 부족함 없는 용량이었으나, **비즈니스 하는 사람들에게**는 **턱없이 부족한 양**이었다. 그래서 서버가 꽉 차서 메일을 못 받는 경우도 비일비재했다. (이때 아웃룩같은 크롤링 서비스, 메일 서버에서 메일을 내 컴퓨터로 긁어오고 서버를 비우는 서비스, 가 시중에 나와 있었다) 구글은 **사람들의 고통**을 **포착**하고 여기에 집중했다. 나머지 기능은 다른 메일 시스템들과 똑같았음에도 불구하고 구글의 지메일은 엄청난 인기를 끌었다.

제품을 개발하다보면 의견이 갈릴 때가 있다. 이때 가장 좋은 방법은 시장의 검증을 받는 것이다. 켄트백은 제품에 대한 스펙 리스트를 만들지 말고, 소비자에게 도움이 될 만한 기능들로 리스트를 만들라고 언급한다. **애자일**은 **빠르게 만들어서 빠르게 전달**하고, **변화에 유연하게 대처**하는 **개발 방법론**이다.

모든 **제품**은 **시장에 대한 가정**을 가지고 있다. 그 가정이 완전히 틀렸다면 제품은 존재가치가 없다. 열심히 만든 제품들이 시장에서 퇴출되는 이유다. 그래서 우리는 끊임없이 되물어야 한다.

- 우리의 제품의 **가정**은 **무엇**인가**(What)**
- 그 **가정**은 **언제 확인**되는가?**(When)**

폭포수 방식은 속도가 느리고 제품이 다 완성되어야 확인이 가능하다. 반면 애자일 방식은 매 스프린트(sprint, 전력질주)마다 나온 결과를 바탕으로 전략을 수정할 수 있다.

- 소프트웨어 개발 = 학습
우리가 짜는 소스코드는 무엇이라고 생각하는가? **소스코드**란 **해당 산업에 대한 지식**이다. **소프트웨어 개발**이란 바로(시장과 소비자들의 반응에 대한) **학습**이다. 애자일과 스크럼의 전제가 여기에 있다.

- 삼성의 사례, 죽어있는 조직과 살아있는 조직의 차이: 학습에 대한 자세
자주 삼성에 컨설팅을 가는데, 거긴 부서마다 하나의 회사다. 그러다보니 조직의 분위기가 많이 대비되는 경우를 본다. 어떤 조직은 죽어있는 모습인 반면에 어떤 조직은 생기 있는 모습을 하고 있다. **유효학습**를 할 수 있는 분위기인가, 아닌가의 차이라고 본다. (농구에서) 자유투를 하는 경우를 생각해보자. 내가 공을 대충 던지고 안 들어가면, '다음번에 잘하면 되지'로 끝난다. 반면에 고심끝에, 자세를 이리저리 바꿔보다가 던져서 안 들어가면 그 과정에서 학습이 남는다, 다음번에 던질 때를 위해 지식이 남는 것이다. 앞의 두 조직은 그러한 부분에서 근본적인 차이가 있었다.
유효학습은 **가정, 가설을 검증하는 과정**을 말하며, **분석할 수 없는 행동**에 대해서는 **학습할 수 없다**.

-애자일은 사람의 자율성을 인정한다
프로젝트 관리에서는 **X이론과 Y이론**이 있다. **X이론**은 기본적으로 **사람을 믿지 않는 것**이다. 사람들은 무능하고 탐욕스럽고 가급적 일을 하려 하지 않는다고 본다. 때문에 관리자가 필요하고 **끊임없이 그들의 일에 참여**해야 한다. 반면, **Y이론**은 **사람의 자율성을 인정**한다. **애자일 기법**은 **Y이론을 바탕으로 한 조직**에서 **가장 잘 동작**한다.

- 도요타의 린 개발: 끊임없는 개선
도요타의 린(lean) 개발은 이미 유명한 개념이 되었다. 도요타의 (과거) 장점은 **평범한 직원들의 지혜를 집대성**하는 능력에 있었다. 또한 카이젠(개선)이라는 개념을 바탕으로 **끊임없이 개선**하는 것이다. 그들은 자주 이렇게 묻는다. "우리 이번에는 이렇게 일해 볼까요?" 폭포수 방식으로는 나올 수 없는 질문이다.

30분이라는 시간동안 할 작업을 2분 단위로 쪼개서 자주 그 방법을 개선해보는 것으로 예를 들 수 있겠다. 첫 번째 방법은 자기가 잘 하고 있는지 잘못 하고 있는지를

알기 어려운 반면에 두 번째 방법은 매번 전략을 바꿔볼 수 있기 때문에 끊임없는 개선이 이뤄진다. 30분 후면 확연히 다른 결과가 나타난다.

애자일 철학은 **소프트웨어 개발이란 학습**이라는 가정을 인정하는 데 있다. 이 개념을 확장한 오픈 애자일은 자신의 삶에 애자일 철학을 도입한 것이다. 기회가 있다면 찾아보면 좋겠다. 끊임없이 개선하는 삶에 대한 이야기이다.

4. 애자일 개발선언

애자일이라는 단어는 크게 두 가지 의미를 갖는다. 하나는 앞서 언급했던 애자일 선언을 의미하며, 다른 하나는 앞으로 우리가 배우게 될 애자일 방법론을 의미한다. 먼저 애자일 선언은 다음과 같다.

"우리는 소프트웨어를 개발하고, 또 다른 사람의 개발을 도와주면서 소프트웨어 개발의 더 나은 방법들을 찾아가고있다. 이 작은을 통해 우리는 다음을 가치 있게 여기게 되었다:

> 공정과 도구보다 개인과 **상호작용**을
> 포괄적인 문서보다 **작동하는 소프트웨어**를
> 계약 협상보다 **고객과의 협력**을
> 계획을 따르기보다 **변화에 대응하기**를

가치있게 여긴다. 이 말은, 왼쪽에 있는 것들도 가치가 있지만, 우리는 오른쪽에 있는 것들에 더 높은 가치를 둔다는 것이다."

우리는 다음 원칙을 따른다:

1. 우리의 최우선 순위는, 가치 있는 소프트웨어를 일찍 그리고 지속적으로 전달해서 고객을 만족시키는 것이다.
2. 비록 개발의 후반부일지라도 요구사항 변경을 환영하라. 애자일 프로세스들은 변화를 활용해 고객의 경쟁력에 도움이 되게 한다.
3. 작동하는 소프트웨어를 자주 전달하라. 2주 ~ 2개월 정도의 간격으로 하되 더 짧은 기간을 선호하라.
4. 비즈니스 쪽의 사람들과 개발자들은 프로젝트 전체에 걸쳐 날마다 함께 일해야 한다.
5. 동기가 부여된 사람들을 중심으로 프로젝트를 구성하라. 그들이 필요로 하는 환경과 지원을 주고 그들이 일을 끝내리라고 신뢰하라.
6. 개발팀으로, 또 개발팀 내부에서 정보를 전하는 가장 효율적이고 효과적인 방법은 면대면 대화이다.
7. 작동하는 소프트웨어가 진척의 주된 척도이다.
8. 애자일 프로세스들은 지속 가능한 개발을 장려한다. 스폰서, 개발자, 사용자는 일정한 속도를 계속 유지할 수 있어야 한다.

9. 기술적 탁월성과 좋은 설계에 대한 지속적 관심이 기민함을 높인다.
10. 단순성이 -- 안 하는 일의 양을 최대화하는 기술이 -- 필수적이다.
11. 최고의 아키텍처, 요구사항, 설계는 자기 조직적인 팀에서 창발한다.
12. 팀은 정기적으로 어떻게 더 효과적이 될지 숙고하고, 이에 따라 팀의 행동을 조율하고 조정한다.

5. 관리주도 Vs. 자기주도

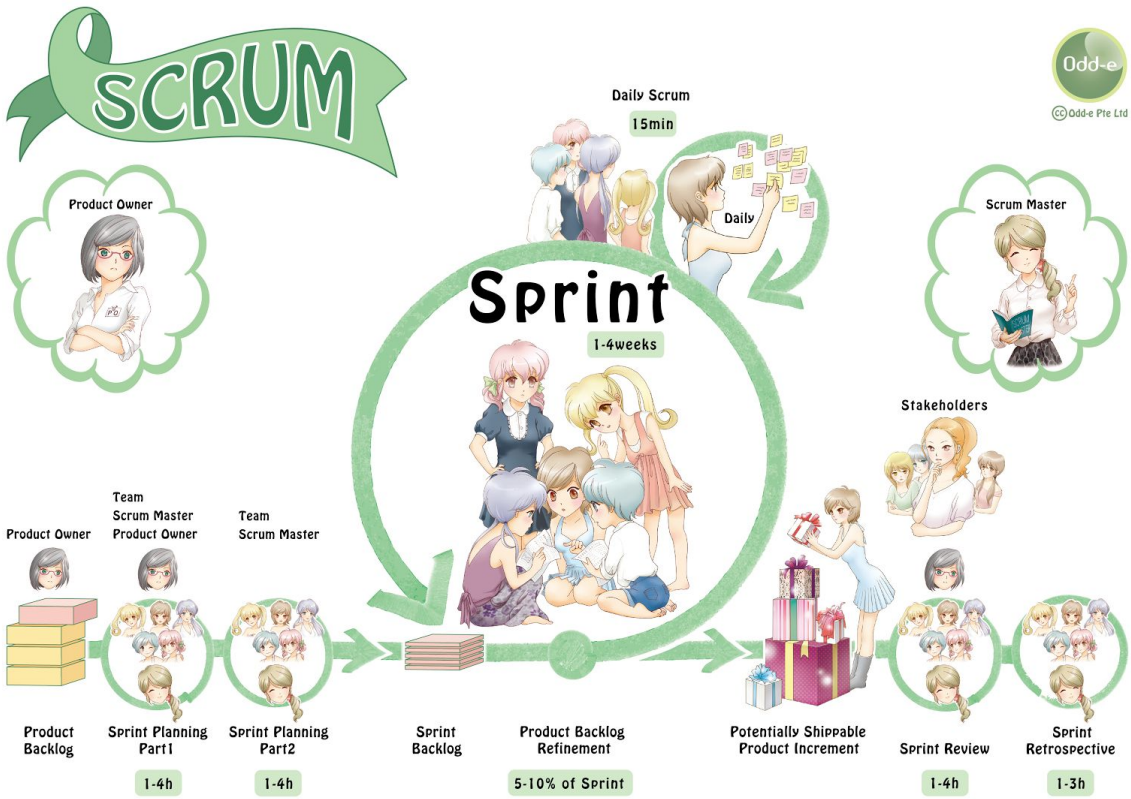
수평적인 조직이 갖는 장점이 있다. 바로 **관리자 없이도 돌아갈 수 있다**는 점이다. 이것이 바로 self-organizing 이다. 각자가 서로의 할일에 집중할 수 있다. 또한 조직은 스스로 할 일을 찾아 해결할 수 있다. 물론 새로운 방법이 조직에 처음 적용되는 순간(학습기간)에는 불가피하게 **비효율**이 생긴다. 이 **비효율의 벽**을 넘는순간 조직은 **새로운 DNA**로 **새로 태어나**게 된다.
대부분의 **기존 회사들**은 이 비효율의 기간을 기다리지 못하고 기존의 방법으로 되돌아간다. 때문에 **더 나은 개발 방법론** 대신 폭포수 방법론에 머물고 있는 것이다. 여담이지만 여러분들이 그런 회사에 들어가지 않으려면 **회사 규모**보다 **회사의 방향성**에 주목해야 한다.

프로젝트 기간동안 팀원들이 **집중력을 유지하는 방법**은 어떤 것이 있을까?
여러분들이 앞으로 프로젝트를 하면서 고민하게 될 부분이다. 다음과 같은 요소가 있다.

1) **명확한 공통의 목표(Shared Goal)**
    a) 패캠 프로젝트를 보면 여러 목적을 가진 사람들이 있다. 스타트업을 하기 위해 프로토타입을 이걸로 만드는 사람들도 있고, 재미로 해보려는 사람도 있고, 그냥 한 번 해보고 싶은 사람들도 있다. 이런 여러 목적을 지닌 사람들이 모여서 한 팀을 이루게 된다. 명확하고 구체적인 공통의 목표가 없다면 속도를 내기 힘들다.
2) **재량권(Resource)**
    a) 각 개인이 팀 내에서 할 수 있는 것의 범위를 의미하며, 넓게는 스프린트 길이(시간제한)까지 포함한다. 일반적으로는 하지 말아야 할 것을 정의하고, 그 외에는 재량권을 주는 것이 바람직하다. 창의력은 거기서 나온다.
3) **고정된 팀 멤버(Fixed Team)**
    a) 이 부분은 회사의 지원이 필요한 부분이지만, 아마 여러분들이 결정할 수 있는 부분은 아닐 것이다. (스타트업을 제외한) 기존의 회사들은 대개 잠깐의 비효율에 대해 인내심을 발휘하지 않는다.
    b) 소스코드는 지식이라는 이야기를 앞서 했었다. 지식이 조직 내에 퍼지고 자리를 잡기까지는 시간이 걸린다. 멤버들이 바뀌지 않는다는 가정 하에. 문화 역시도 그렇다. 멤버들이 자주 바뀌면 문화가 자라지 못한다.
4) **투명한 정보교류(Information)**

a) 투명한 피드백이 오고가야 팀은 '학습'할 수 있다. 나의 문제상황에 대해 이야기하는 것은 용기가 필요하다. 이를 위해서는 팀원들간의 신뢰가 있어야 한다. 누군가의 문제에 대해 이야기하고 그것을 다 같이 해결할 수 있는 분위기를 조성해야 한다. "다른 사람이 도와줄거야" 라는 신뢰가 있어야 한다.
b) 기업사례: 스타트업에 간 적이 있었다. 22살 CTO가 있었던 곳인데, (19살 때부터 창업을 했었다고 한다) 추석 이벤트를 준비하고 있었다. 그때 디자이너들이 일하는 것을 보니까, pdf 문서에서 상표 이미지를 일일이 따서 jpg 파일로 정리하고 있었다. CTO를 조용히 불러다가 디자이너들이 일하는 것을 보라고 했었다. 그는 화를 내며 파이썬을 이용해서 pdf 문서 내에서 그림파일만 긁어내는 프로그램으로 2시간 만에 문제를 해결했다. 디자이너 세 명이서 3일 동안 한 일을 2시간 만에 해결한 것이다. 그 이후로 그 팀은 직책 간에 협업을 할 수 있는 분위기를 조성했다. 결과는 컨설팅 이전보다 훨씬 좋았다.

애자일 기법은 크게 XP(애자일 개발에 적용할 수 있는 실제적인 방법들), Scrum(반복개발 방법론), Kanban(각자 어떤 일을 하고 있는지 서로 확인할 수 있는 보드)의 요소로 나뉜다. 오늘 우리는 **스크럼 기법**에 대해 주목할 것이다.



6. 스크럼이란?

스크럼은 다음과 같은 그림으로 나타낼 수 있다.

1. 먼저 **프로젝트 오너**(project owner)가 있다. 그는 **프로젝트의 요구사항 리스트**를 가지고 있다. 이를 **프로젝트 백로그 인덱스(PBI, Project Backlog Index)** 라고 한다.
2. 요구사항들을 바탕으로 **스프린트 플래닝**(sprint planning)을 한다. 회의는 part1, part2 로 구성된다. 프로젝트 오너와 개발팀이 같이 들어가는 **part1** 에서는 **what**, 제품에 들어갈 **기능들**을 **구체적**으로 정하고 **우선순위**를 **선정**한다. part2 에서는 개발팀 내에서 **How**, 기능들을 **어떻게 잘라서 개발**할지에 대해 논의한다. 제한시간은 각각 한 시간으로 잡는 것이 바람직하다. (1주 스프린트 기준)
3. 앞서 스프린트 플래닝을 통해 개발팀은 **약속**(commitment)을 한다. 이는 프로젝트 오너와의 약속이 아니다. 바로 개발팀 스스로와의 약속이 되어야 한다. 이후 개발팀은 스프린트를 진행한다. 스프린트 기간은 1-4주로 결정되며 요즘은 1주 간격으로 많이 진행한다. 스프린트 기간동안 개발팀은 스스로에게 할당한 만큼의 일을 해내고 데일리 스크럼 미팅(DSM, Daily Scrum Meeting)을 통해 **하루 15분** 동안 자기가 **한 일**, **할 일**, 그리고 **문제점**(장애물)에 대해 서로 이야기한다.
4. 스프린트의 결과로 잠정적 출시가능 증분(PSPI, Potentially Shippable Product Increment)이 나온다. 이 결과물을 가지고 회의를 진행하는데 **제품에 대한 회의**(Review)와 **팀 자체에 대한 회의**(Retrospective)를 진행한다. 각각 한 시간 정도가 적당하다. (1주 스프린트 기준)

참고: 개발팀 인원은 7 +- 1 명 정도가 적당하며 더 많으면 두 팀으로 쪼갠다. 디자이너와 개발자가 함께 포함된 팀(cross-functional team)이 스프린트를 뛰어야 맞다. 인원이 너무 많다면 DSM 을 할 때 정보가 너무 많이 공유되어서 정보가 그 의미를 잃게 된다.

대답해볼 질문들
Q. 제품개발=학습 으로 전환하면, 프로젝트 하는데 무엇이 달라집니까?
Q. 좋은 학습을 위해 팀의 리더/멤버로서 내가 할 일은 무엇입니까?