Agile fundamentals

Agile one day course

Who am I?

Name : 김기훈

Company : Samsung SDS. ACT Lab

Role : Developer, Software Architect, Agile Trainer

Email : koreakihoon@gmail.com

Agenda

08:00 Introduction, 방법론

09:00 Agile Methodologies & Culture

10:00 Agile Project Approach

11:30 Lunch

13:00 Define the Project

14:00 Estimation

15:00 Engineering Practices

16:00 Communication

Agile이 무엇인지 느끼기

Team 으로 어떻게 일하는지 느끼기

Who are you?

...

Programming 경험?

Team으로 일한 경험?

Agile이 뭔지 들어봤나요?

...

Drawing Game

ANALYSTS

DEVELOPERS

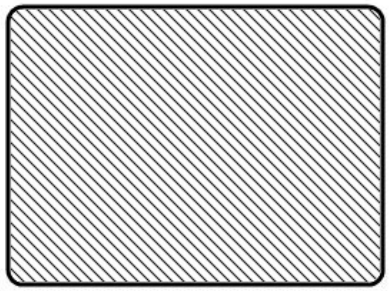
분석한 내용은 종이에만 작성하세요.

어떤 형식, 어떤 언어로 작성해도 괜찮아요.

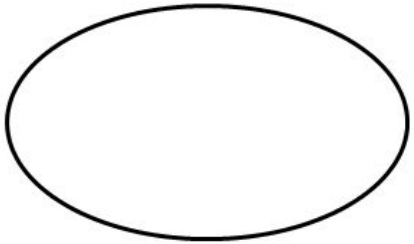
단, 그림을 그리면 안되요.

문장이나 단어로만 작성하세요.

analysts 와 **developers** 는 서로 대화하면 안되요.



yu Can'T sTop IT



Review

잘 그려졌나요?

뭐가 문제였나요?

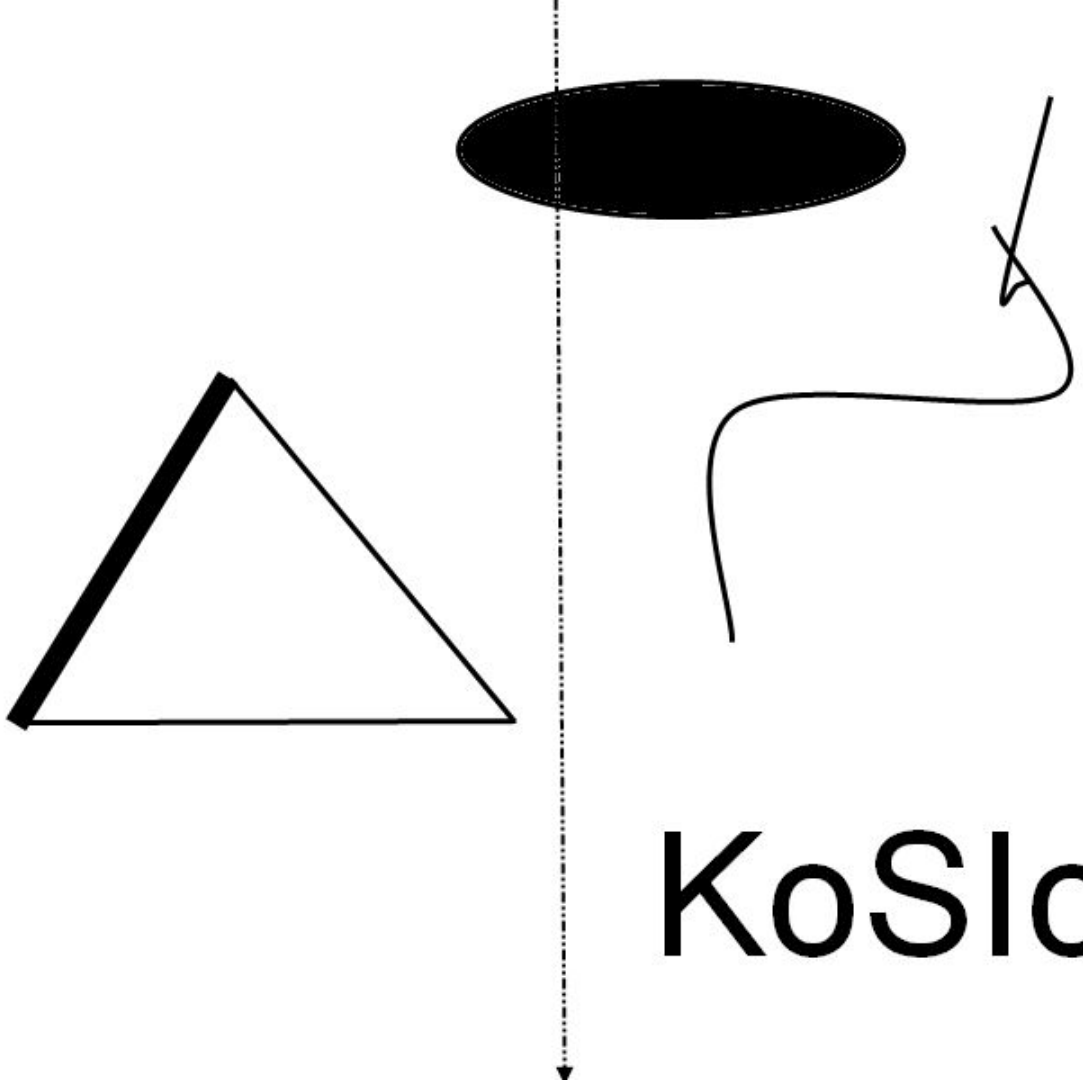
분석한 내용은 종이에만 작성하세요.

어떤 형식, 어떤 언어로 작성해도 괜찮아요.

단, 그림을 그리면 안되요.

문장이나 단어로만 작성하세요.

analysts 와 **developers** 는 대화를 많이 하세요.


KoSIce

Learning point

Incremental development

Iterative development

Documented analysis results vs. communication with a customer/analyst

Collaboration

Self-organization of the team

Leadership vs. management

Early feedback

Changes are welcome

Time box

Result driven work

Retrospective/Process improvements

방법론이 뭐에요??

1) 김치와돼지고기를 3대 1 비율로 준비한다.
2) 쌀뜨물에 고기를 넣고 끓이다 김치를 넣는다.
3) 간마늘한숟갈을넣고, 파와고추를 썰어 넣는다.
4) 고운 고춧가루, 굵은 고춧가루를 섞어 넣어 색감을 더한다.
5) 국간장을넣어 향을내고, 새우젓으로 간을한다.
6) 된장반 스푼을넣으면 깊은맛을더할수있다.

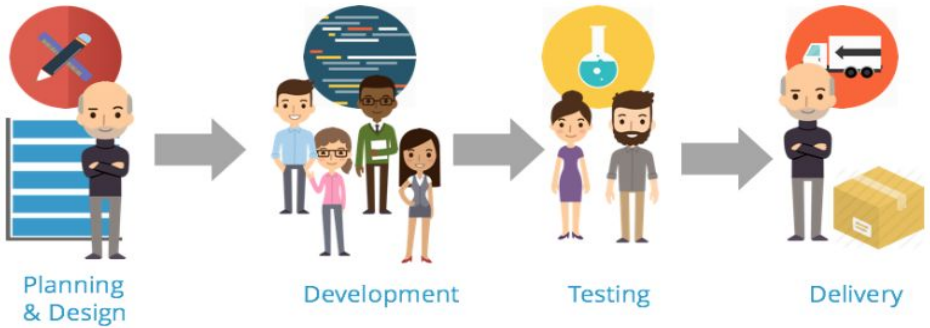
Methodology

소프트웨어 개발 방법론은

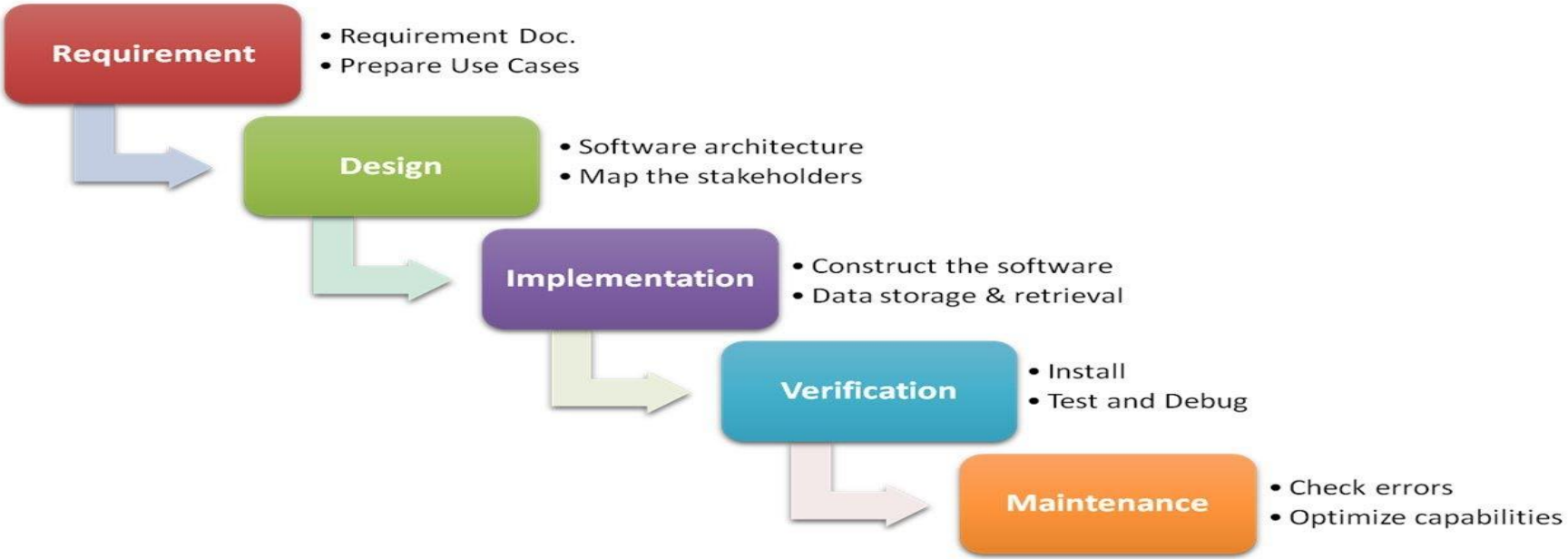
정보시스템을 개발하기 위한 작업활동, 절차, 산출물, 기법 등을 체계적으로 정리한 것

Why???

- 개발경험 축적 및 재활용을 통한 개발 생산성 향상
- 작업의 표준화/모듈화
- 효과적인 프로젝트 관리
- 정형화된 절차와 표준 용어의 제공
- ...



WATERFALL


Requirement
• Requirement Doc.
• Prepare Use Cases
Design
• Software architecture
• Map the stakeholders
Implementation
• Construct the software
• Data storage & retrieval
Verification
• Install
• Test and Debug
Maintenance
• Check errors
• Optimize capabilities

프로젝트에서 관한 세가지 진실

프로젝트 초기에 요구사항을 모두 수집하기는 불가능하다.

수집한 요구사항들이 무엇이든 반드시 변하기 마련이다.

시간이나 비용이 허락하는 것보다 해야 할 일들이 항상 더 많다.

세가지 진실을 받아 들인다는 것은..

프로젝트 초기에 요구사항을 모두 수집하기는 불가능하다.
-> 수집한 요구사항을 모두다 알지 못하는 상태로 여행을 시작하는 것을 두려워하지 않는다는 것

수집한 요구사항들이 무엇이든 반드시 변하기 마련이다.
-> 변화를 두려워하거나 피하려 하지 않는다는 것.

시간이나 비용이 허락하는 것보다 해야 할 일들이 항상 더 많다.

->주어진 시간과 자원보다 해야 할 일이 많더라도 스트레스에 시달리지 않는 다는 것.

Agile methodologies & Culture

Agile manifesto

Individuals and interactions

over processes and tools

Working software

over comprehensive documentation

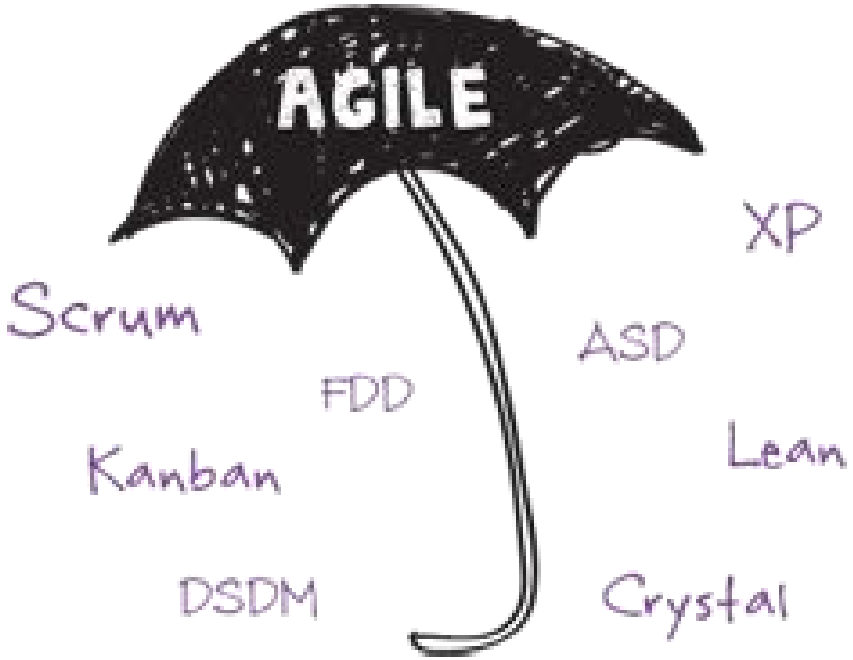
Customer collaboration

over contract negotiation

Responding to change

over following a plan

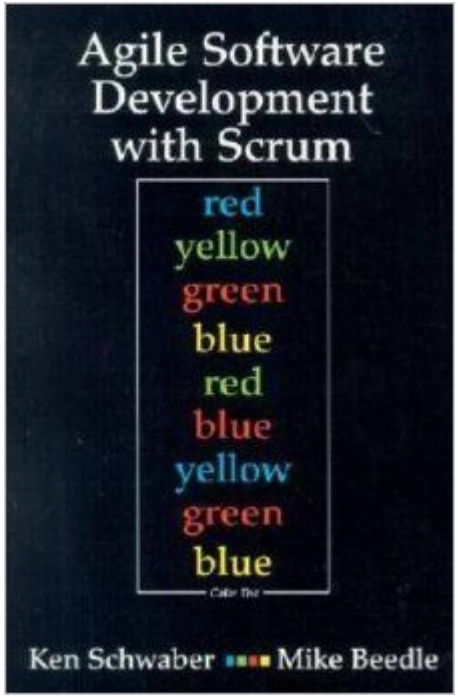
http://agilemanifesto.org/iso/kor/

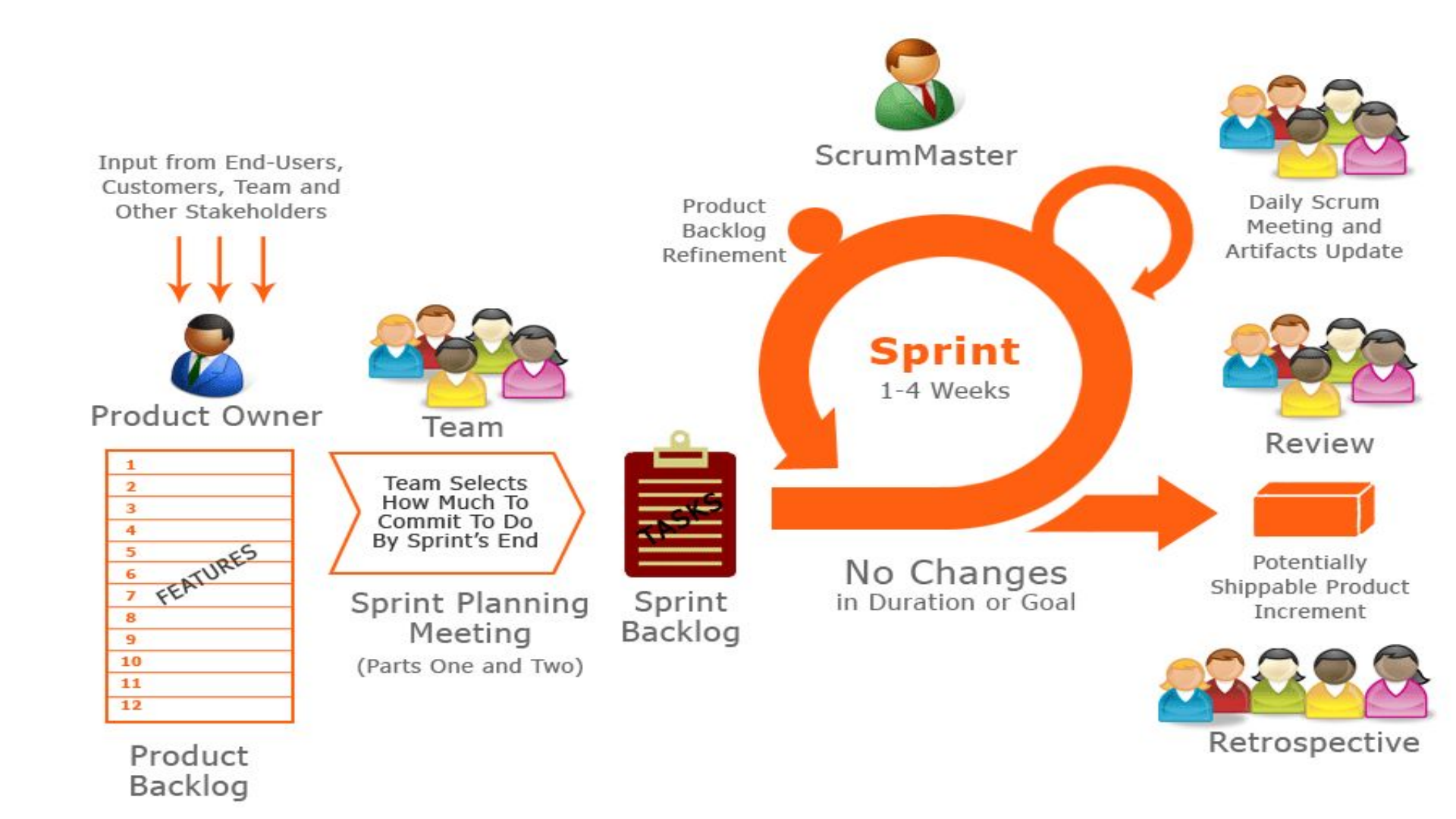

AGILE
XP
Scrum
ASD
FDD
Lean
Kanban
DSDM
Crystal

Scrum

Primarily a project management method

- Project planning
- Release planning
- User Stories
- Daily stand-up
- Story points
- Velocity

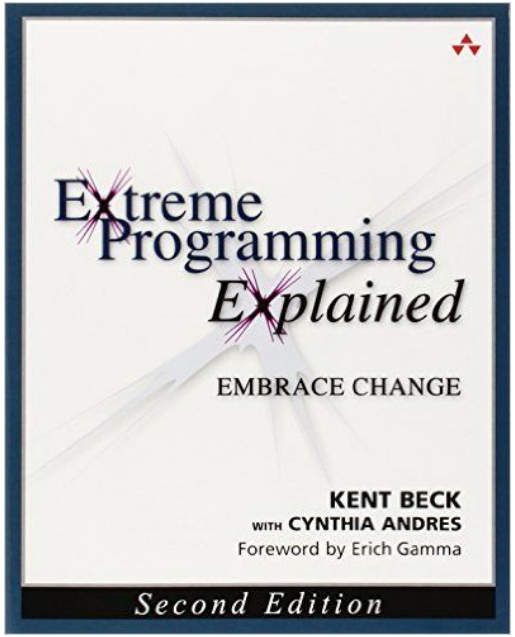



Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Product Backlog
Team Selects How Much To Commit To Do By Sprint's End
Sprint Planning Meeting
(Parts One and Two)
TASKS
Sprint Backlog
ScrumMaster
Product Backlog Refinement
Sprint
1-4 Weeks
No Changes
in Duration or Goal
Daily Scrum Meeting and Artifacts Update
Review
Potentially Shippable Product Increment
Retrospective

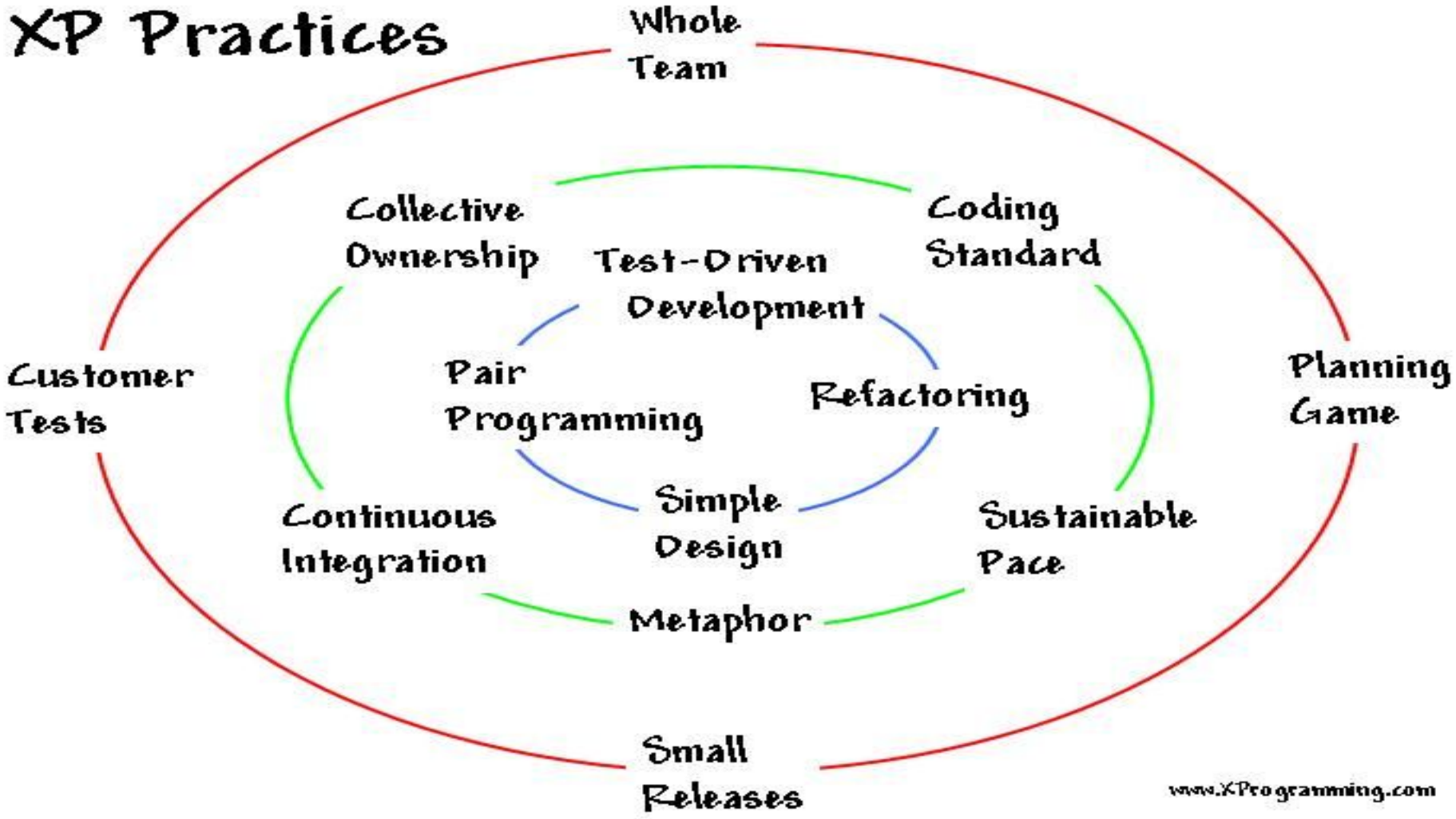
Extreme programming (xp)

Primarily a developer-centric approach

- Test Driven Development (TDD)
- Unit tests
- Pairing
- Continuous Integration (CI)
- Refactoring

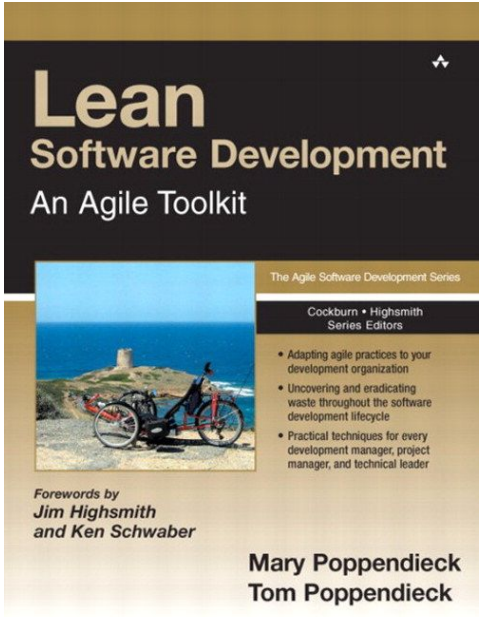


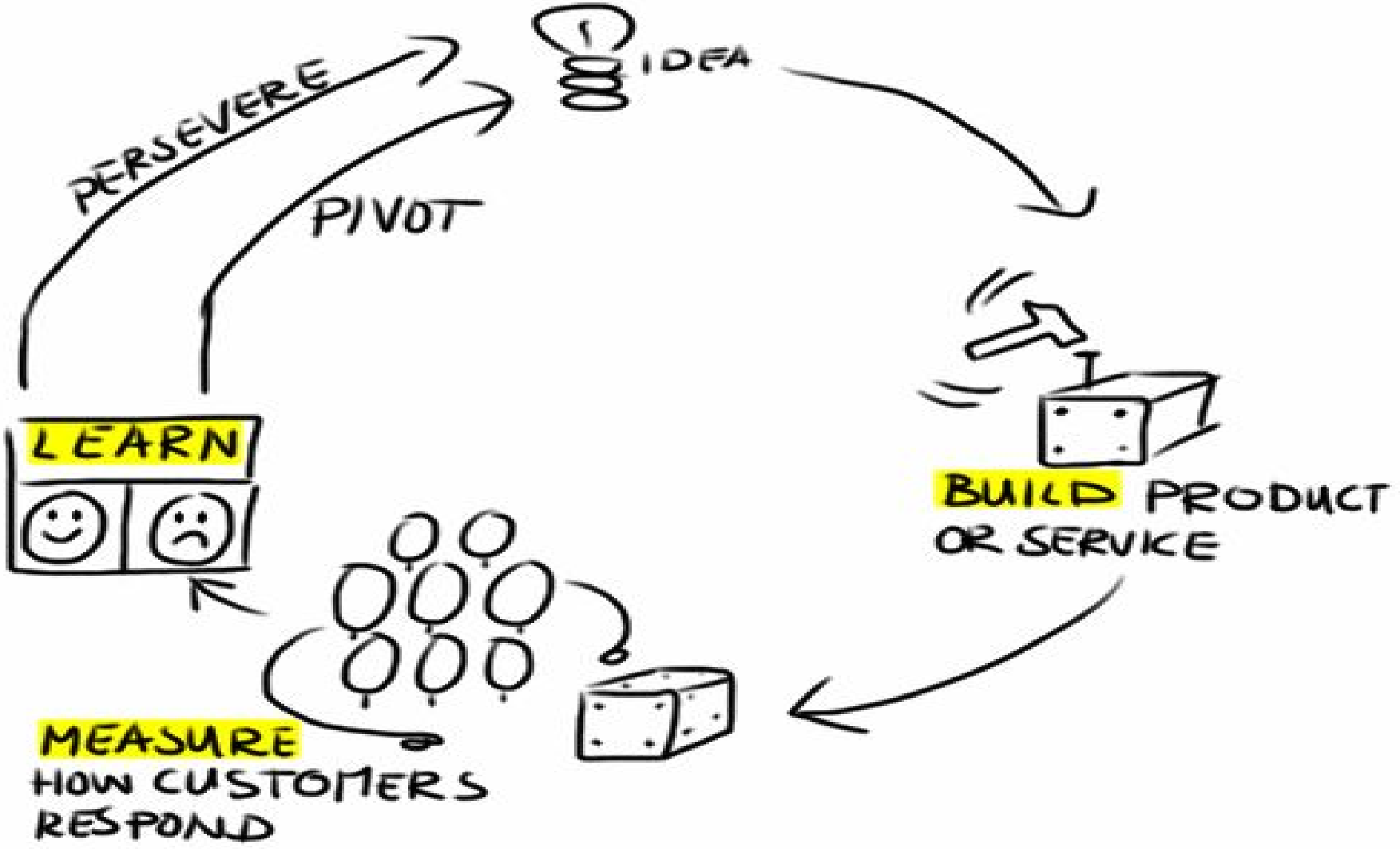
XP Practices



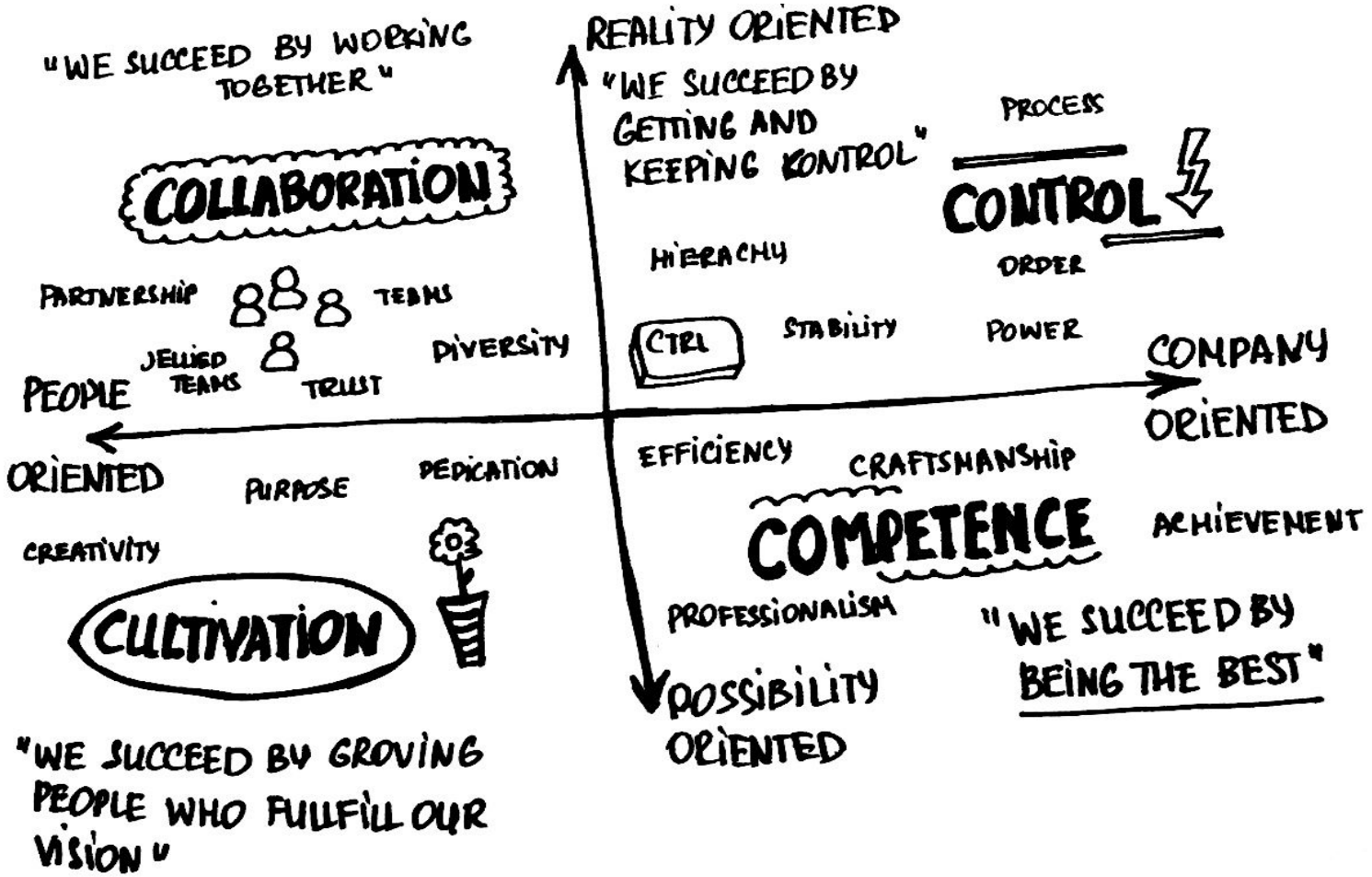
Lean Software Development

- Optimize the whole
- Eliminate waste
- Build quality in
- Learn constantly
- Deliver fast
- Engage everybody
- Keep getting better

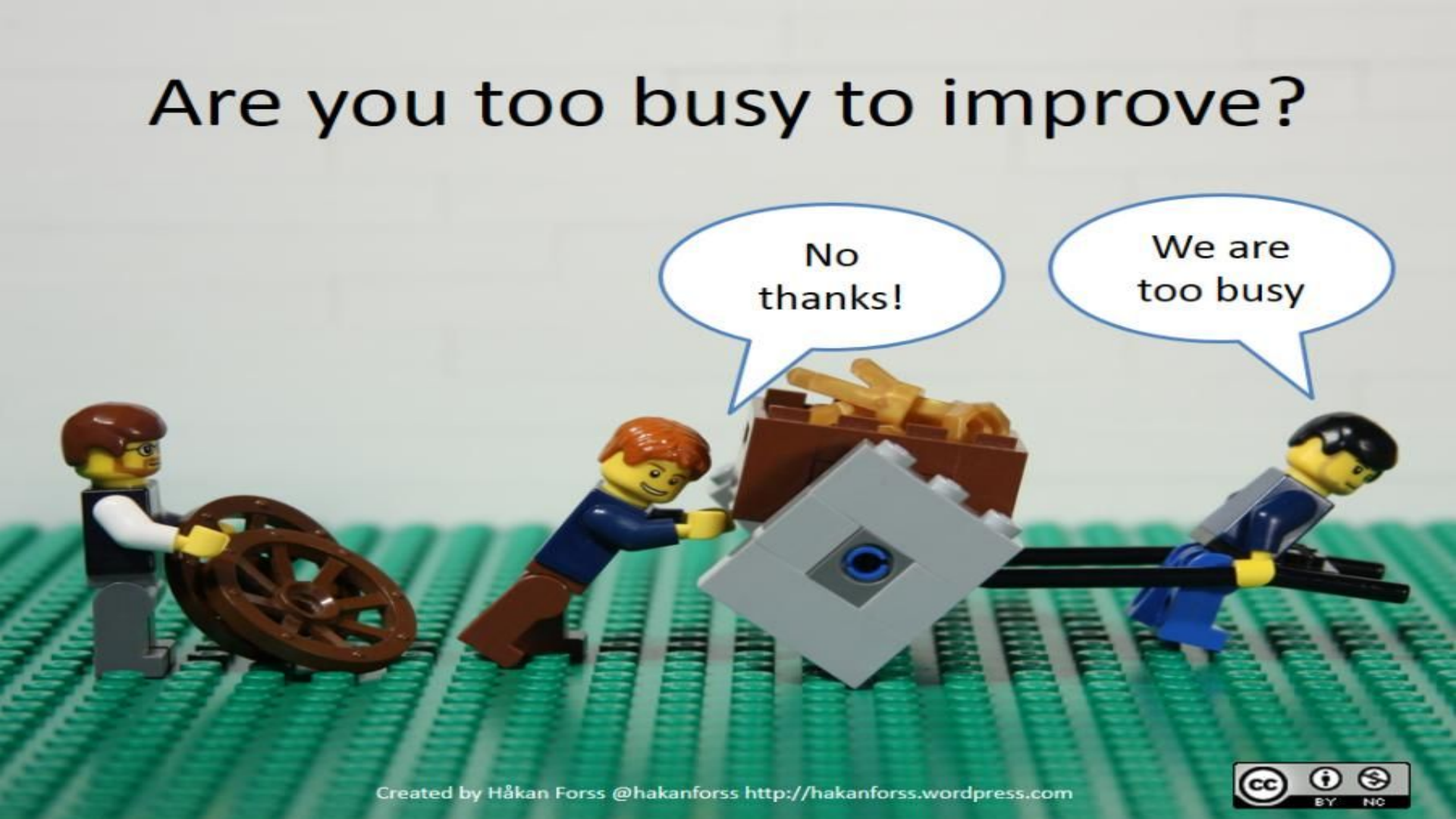



IDEA
PERSEVERE
PIVOT
LEARN
BUILD PRODUCT
OR SERVICE
MEASURE
HOW CUSTOMERS
RESPOND

CULTURE


"WE SUCCEED BY WORKING TOGETHER"
REALITY ORIENTED
"WE SUCCEED BY GETTING AND KEEPING KONTROL"
PROCESS
COLLABORATION
CONTROL
HIERACHY
ORDER
PARTNERSHIP
TEAMS
JELLIED TEAMS
TRUST
DIVERSITY
CTRL
STABILITY
POWER
PEOPLE ORIENTED
COMPANY ORIENTED
PURPOSE
DEDICATION
EFFICIENCY
CRAFTSMANSHIP
CREATIVITY
COMPETENCE
ACHIEVEMENT
CULTIVATION
PROFESSIONALISM
"WE SUCCEED BY BEING THE BEST"
POSSIBILITY ORIENTED
"WE SUCCEED BY GROVING PEOPLE WHO FULLFILL OUR VISION"

Are you too busy to improve?




CONTINUOUS
IMPROVEMENT

#practice> discuss

Manifesto for Agile Software Development

http://agilemanifesto.org/iso/ko/

Twelve Principles of Agile Software

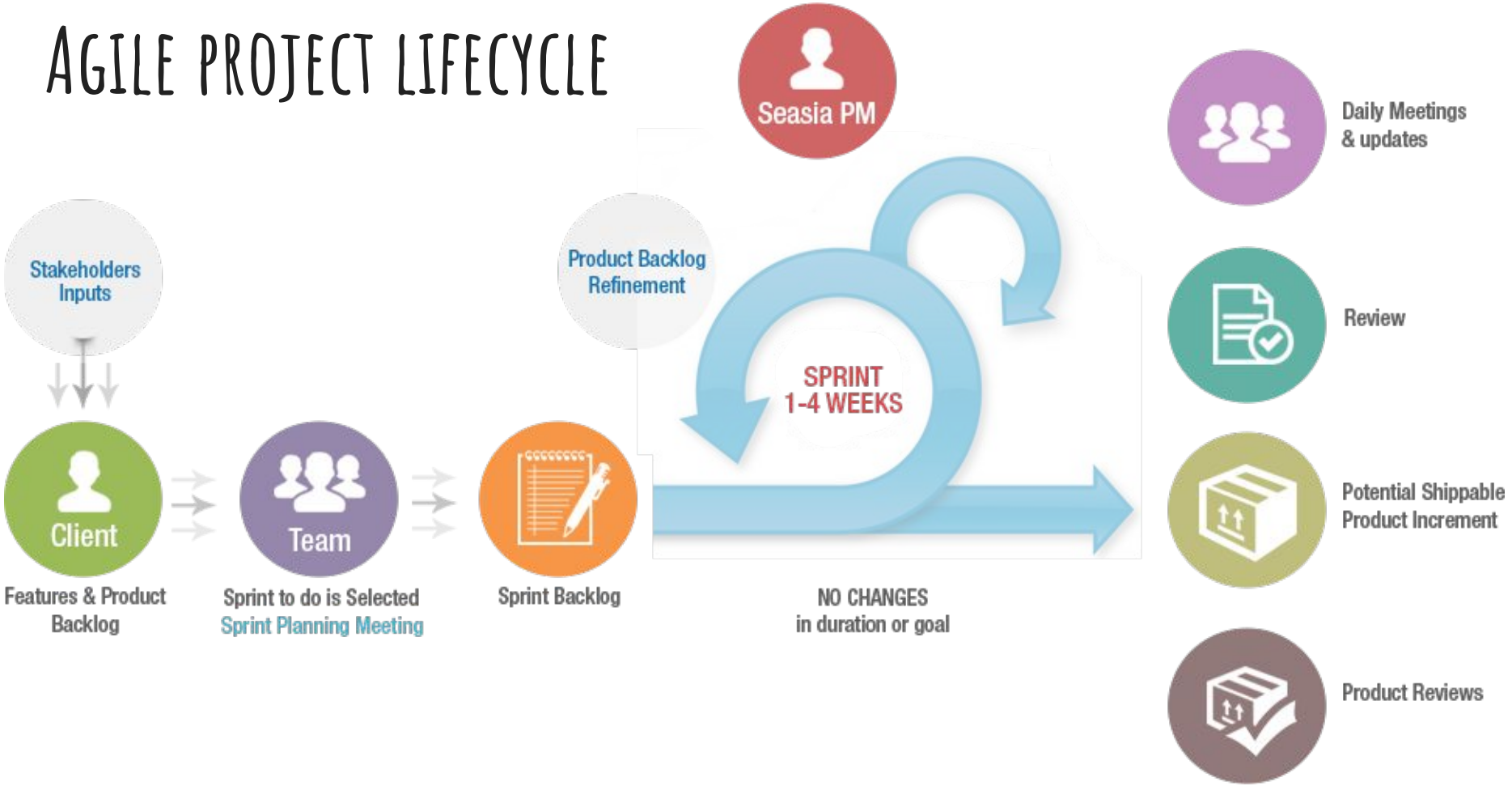
http://agilemanifesto.org/iso/ko/principles.html

Agile project approach

Agile project is...

- Adaptive
- Evolutionary
- Collaborative
- Just-in-time
- Start with a plan
- …

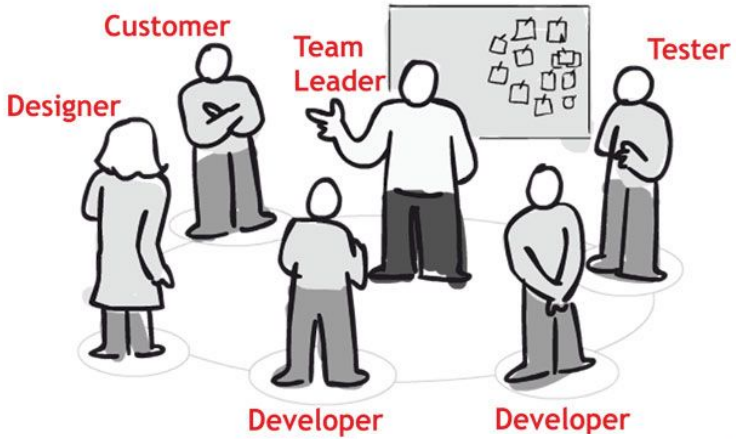
Agile project lifecycle

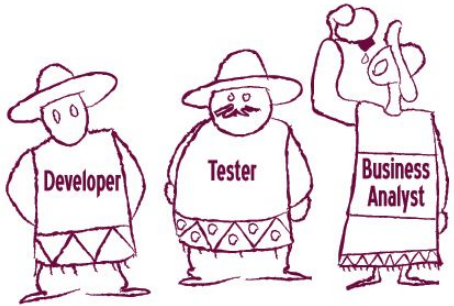


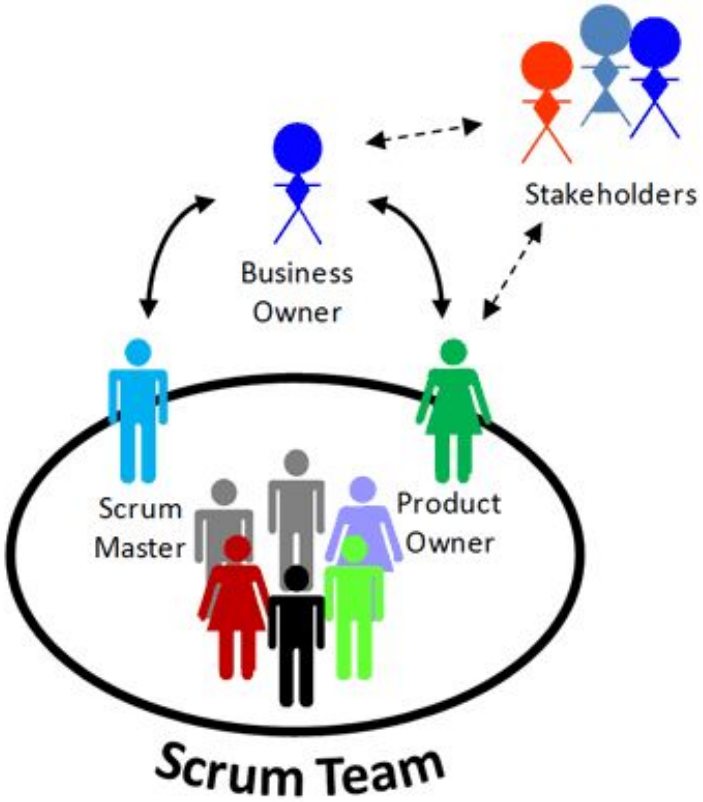
Iteration & improvement

AGILE METHODOLOGIES

Agile project team







Product owner (customer)

Provides vision for the solution

Single point of escalation for prioritization

Iteration manager / scrum master / agile coach

Facilitates collaborative activity

Encourages and mentors

Carries water

Removes boulders

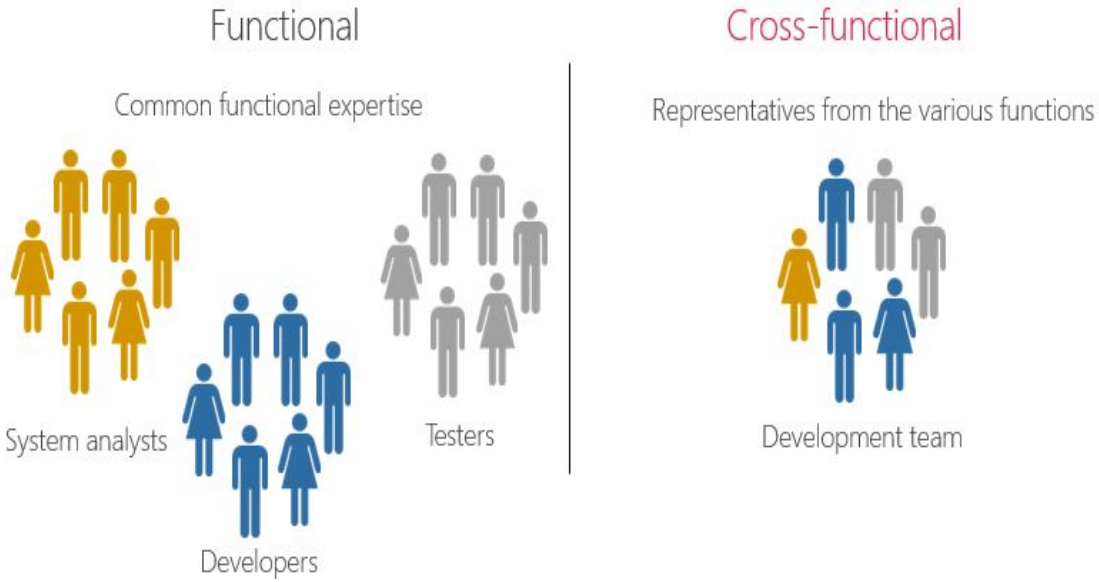
Agile team member

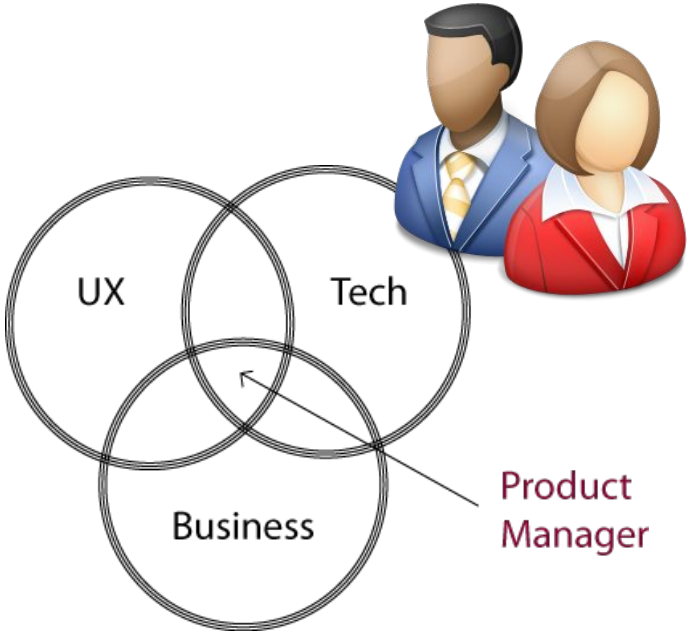
Business Analyst

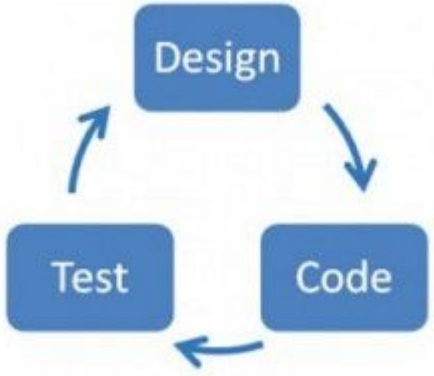
UX Analyst

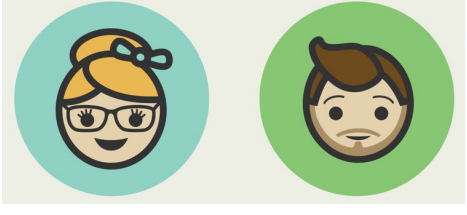
Developer

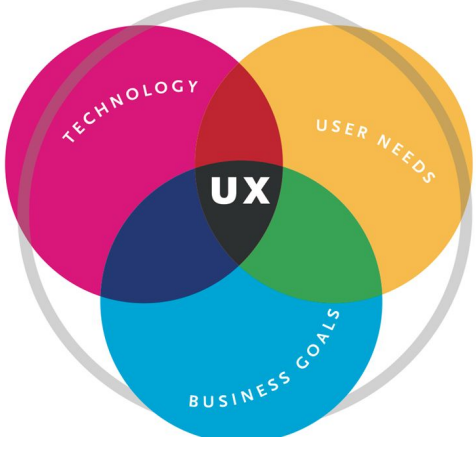
Tester(QA)

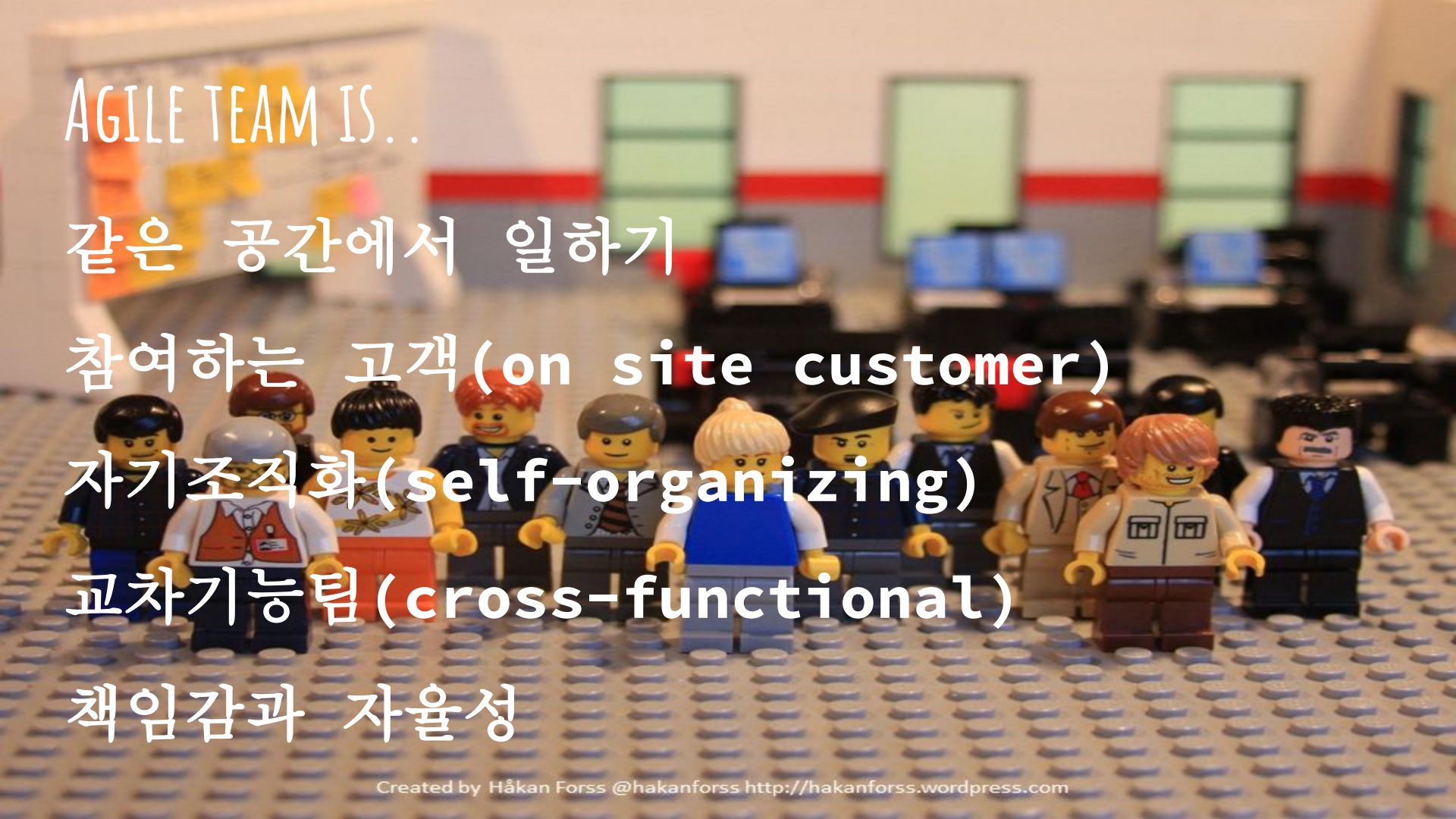



UX
Tech
Business
Product
Manager

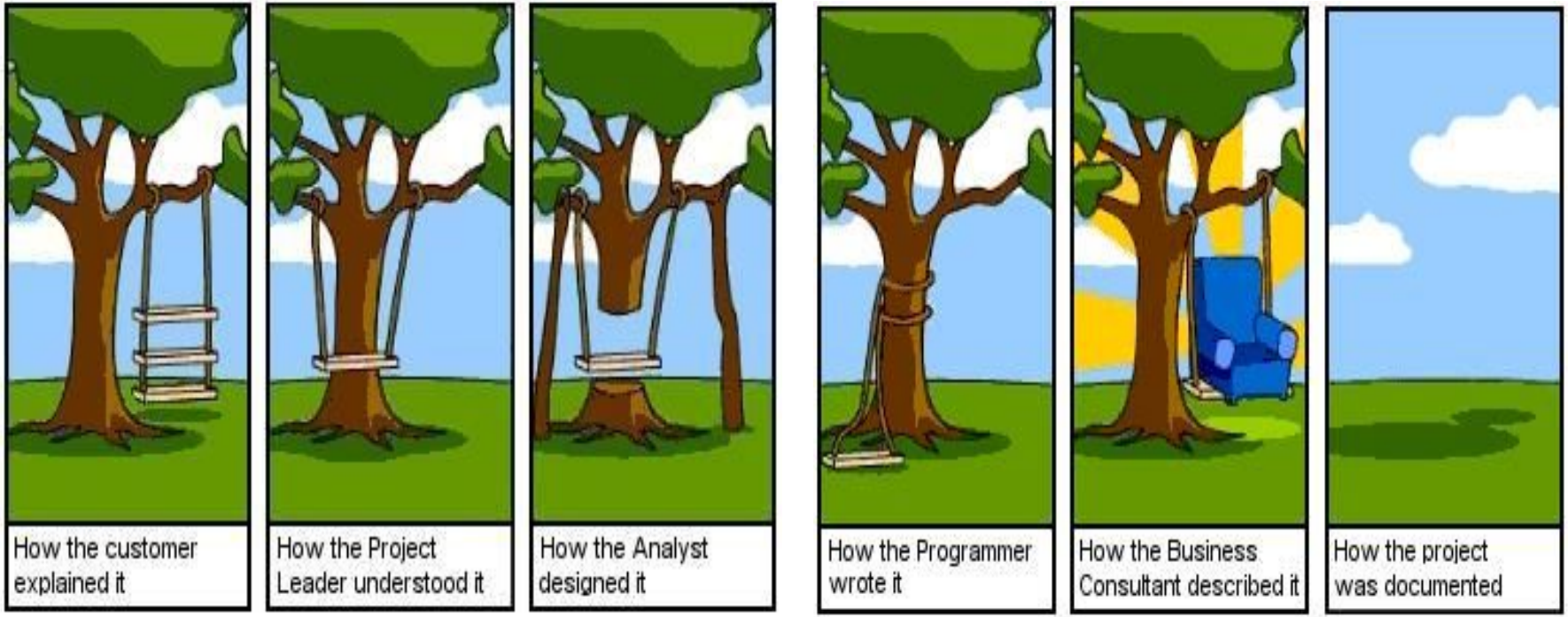

Design
Code
Test

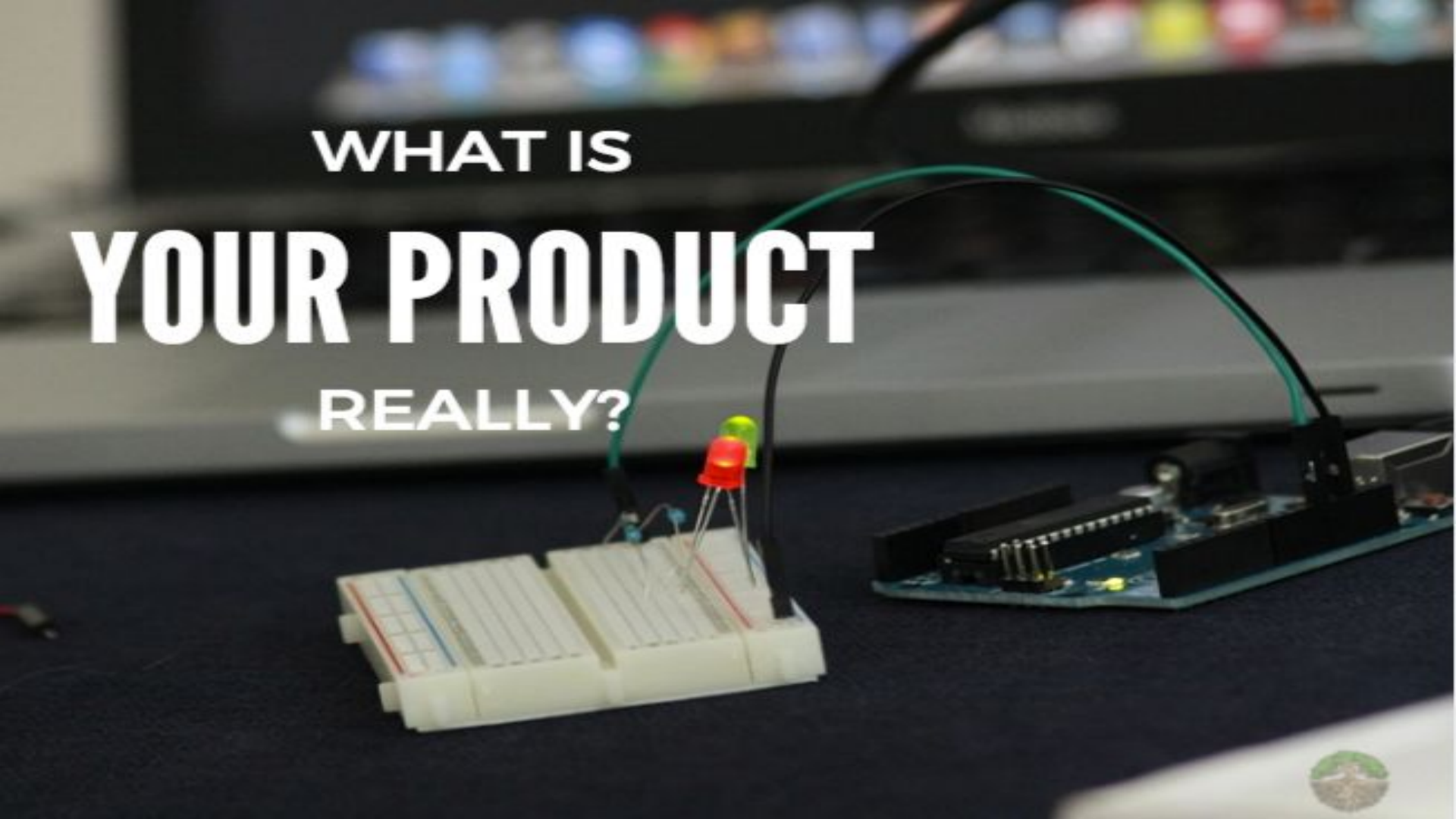



TECHNOLOGY
USER NEEDS
UX
BUSINESS GOALS


AGILE TEAM IS..
같은 공간에서 일하기
참여하는 고객(on site customer)
자기조직화(self-organizing)
교차기능팀(cross-functional)
책임감과 자율성
Created by Håkan Forss @hakanforss http://hakanforss.wordpress.com

Define the project


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented


WHAT IS
YOUR PRODUCT
REALLY?

모두 한 버스에 타는 법

목표, 비전, 프로젝트의 현재 상태에 대해 모든 팀원들과 소통하기

User Story Workshop

Inception deck

고객(사용자) 가치 기반의 상품을 만들기 위한 **Ideation & Consensus**

만들 제품 공감대 형성 : Vision Sharing, Elevator Pitch, Product Box

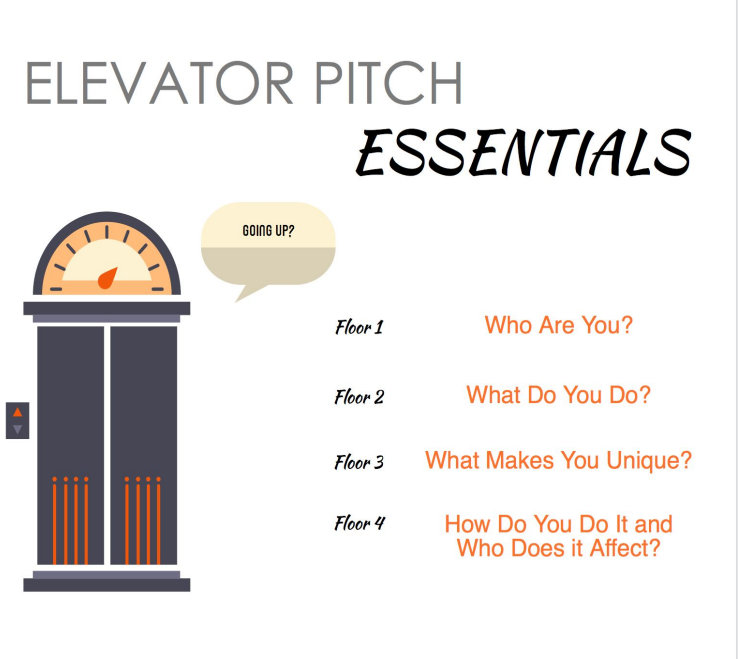
사용자에 대한 이해 : Persona

사용자 가치 기준 주요 Feature 정의 : Customer Journey Map, Stakeholder Map

Inception deck

Vision Sharing

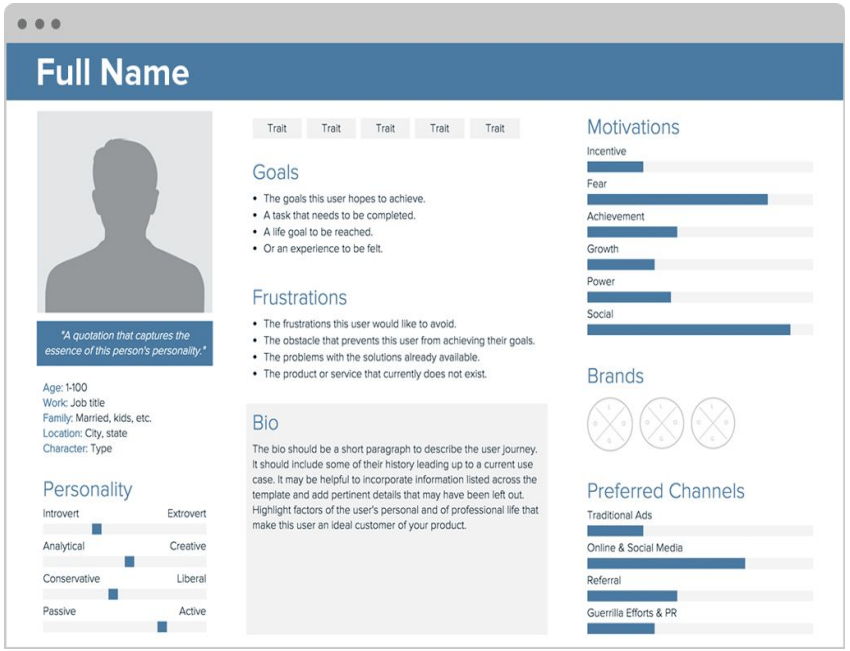
Elevator Pitch



Inception deck

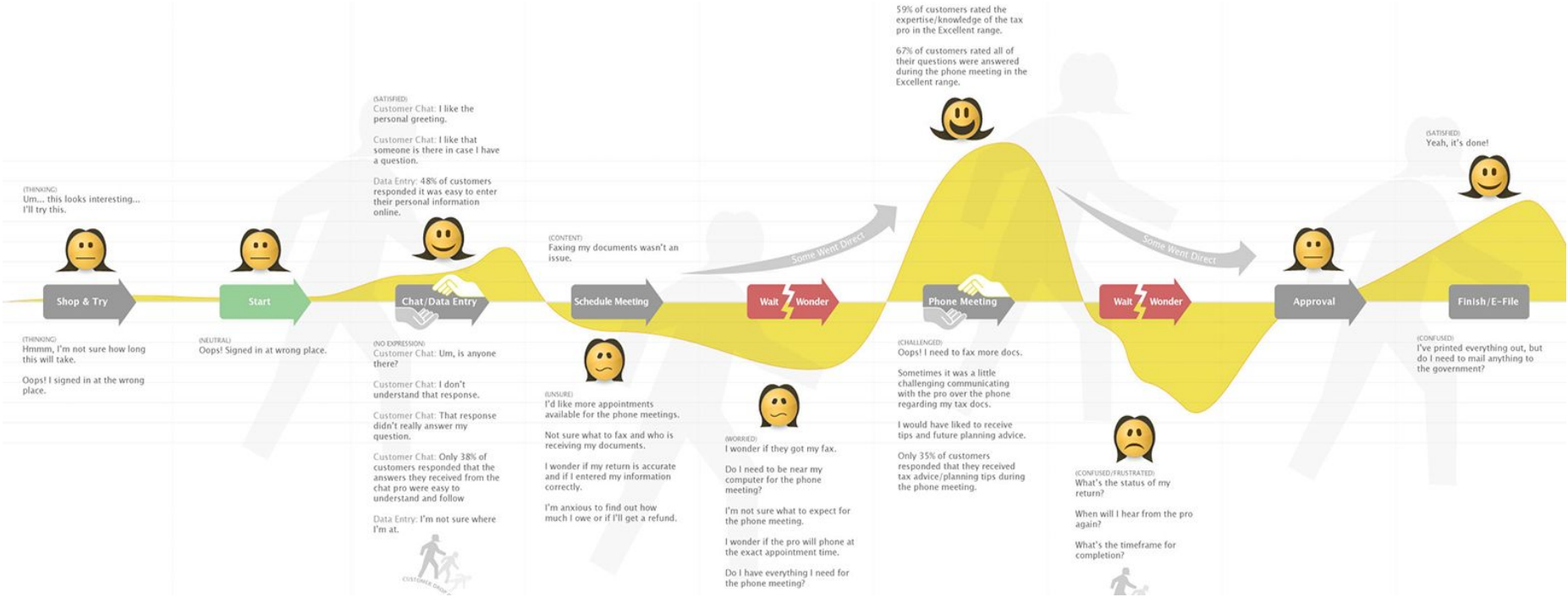
Product Box

Persona



Inception deck

Customer Journey Map



User story deck

고객(사용자) 가치 중심의 **요구사항** 정의

사용자 가치 기준 주요 Feature 정의 : Roles & Goals

Feature 도출 및 우선순위 선정 : Feature Identification, Feature Pyramid

비지니스 흐름 도출 : Business Scenario

화면기반의 업무 흐름 이해 : Low-Fi Prototype

사용자 가치기준 요구사항 식별: User Story(Acceptance Criteria)

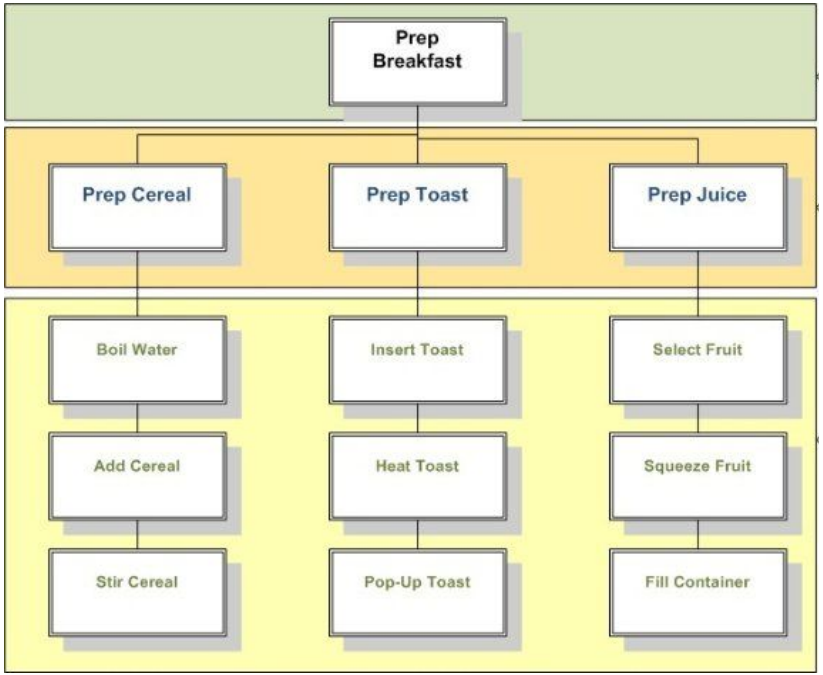
User story deck

Roles and Goals

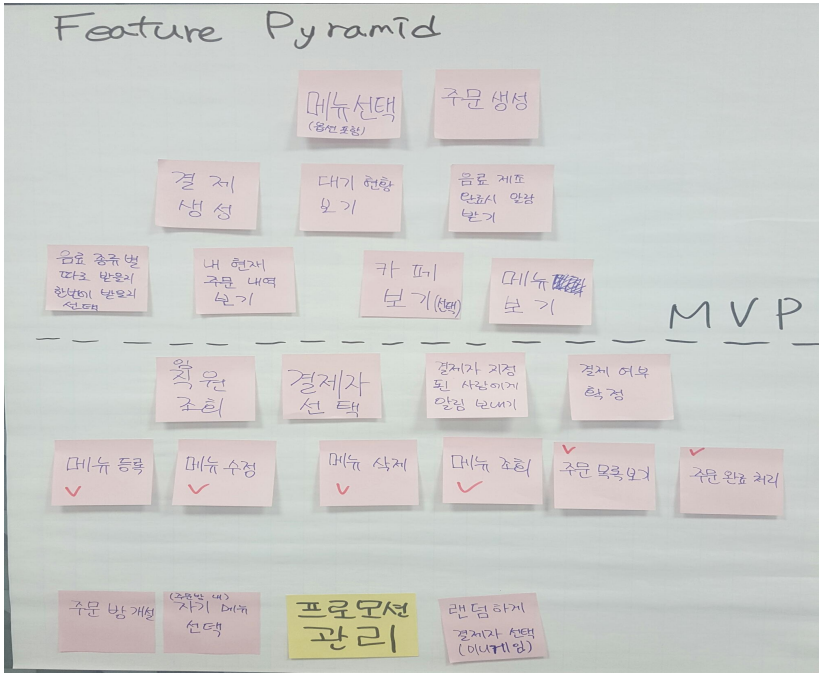
	Who am I?	What do I want to do?	How will you support my goals?	How important am I?
	Karen | 30 | secretary |fashion conscious | lives in Surbiton| forgetful	"I'm going on holiday to California tomorrow. I want to buy travel insurance"	Quickly help me buy travel insurance	
	Chris | 24 | Student | part time DJ | lives with girlfriend | drives performance car | web savy	"Car insurance is so expensive, and getting someone to quote me is hard. I want to know quickly if I'm going to get a good quote"	Enter minimal details | provide quick quote | save quote | comprehensive quote if interested	
	Sara | 22 | recruitment consultant | seeks advice | internet 'lingerer'	"I crashed my car on the way to work. We've swapped addresses, now to claim on the insurance..."	Let me download claim forms | Let me monitor my claim on-line	

User story deck

Feature Identification

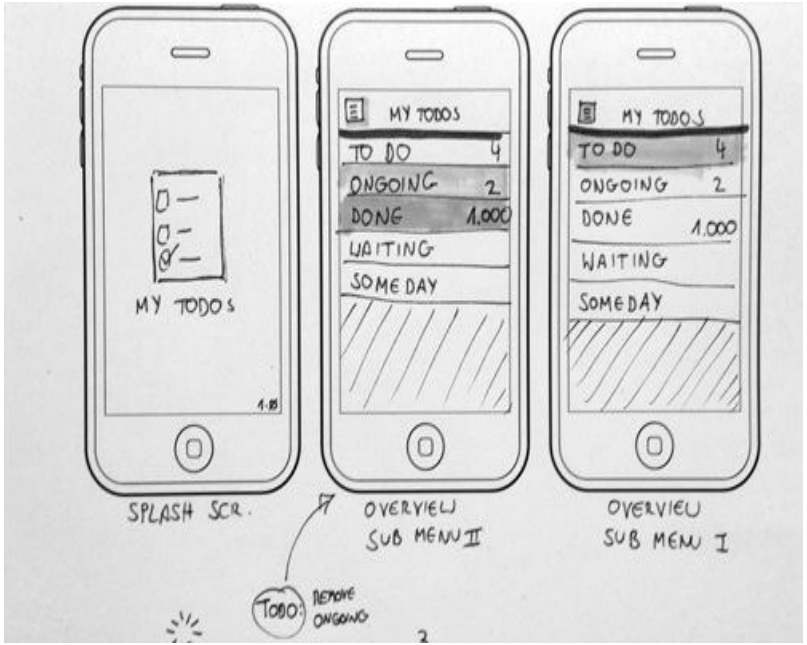


Feature Pyramid



User story deck

Low-Fi Prototype



User Story

As a/an	I want to...	so that...
moderator	create a new game by entering a name and an optional description	I can start inviting estimators
moderator	invite estimators by giving them a url where they can access the game	we can start the game
estimator	join a game by entering my name on the page I received the url for	I can participate
moderator	start a round by entering an item in a single multi-line text field	we can estimate it
estimator	see the item we're estimating	I know what I'm giving an estimate for
estimator	see all items we will try to estimate this session	I have a feel for the sizes of the various items
moderator	see all items we try to estimate this session	I can answer questions about the current story such as "does this include ___"
moderator	select an item to be estimated or re-estimated	the team sees that item and can estimate it

User story

고객에게 가치를 전달할 수 있는 서비스의 기능을 기술한 내용

사용자 스토리는 기술적인 용어가 아닌 **비즈니스 언어로 작성**

INVEST 원칙에 따라 작성

Independent

Negotiable

Valuable

Estimable

Small

Testable

User story template

Role: As a…
Goal: I want…
Value: So that...

```
# Story
As      : 의사로서
I want  : 내 환자 리스트를 보고 싶다
So that : 내 환자를 찾는 시간을 줄이기 위해서
```

vs

```
# Story
환자 목록을 database에서 조회하여 jqgrid에 출력한다.
```

Acceptance criteria

스토리의 완료 조건

The **Context:** Given…
The **Event:** When…
The **Outcomes:** Then…

#A/C
Given 사용자가 로그인 상태가 아닌 경우
When 쇼핑카트에 담기 버튼을 눌렀을 때
Then 로그인 화면을 띄워 준다

Definition of Done

개발자가 유저스토리를 완료(Done)했다고 말했는데,
Unit Test 코드가 없거나, VCS에 커밋하지 않았다면?
QA가 해당 스토리는 테스트가 더 필요하다고 말한다면?
나중에 개발자가 소스코드를 다시 수정했다면??
성능테스트를 해야되는 상황이라면??

완료(Done)의 의미를 정의할때,
개발자 뿐만아니라 PM, Architect, DA, UX, 보안담당자 등의 의견이 반영되어야 한다.

“거의다했어요!?” Almost Done은 Done 이 아니다!!

애자일 프로젝트에서 스토리는 작다. 그러므로 단순하게 Done / Not Done 으로만 관리 해라.

스토리가 고객에게 보여줄 준비가 되지 않았다면, 그건 Not Done 이다.

https://trello.com/ https://ko.atlassian.com/software/jira

ESTIMATION

PLANNING...

How much can we do?

How many iterations?

여기서 집까지 가까운가요?

한 시간쯤 걸려요.

이 User story 개발하는데 얼마나 걸려요?

```
# Story
As : 의사로서
I want : 내 환자 리스트를 보고 싶다
So that : 내 환자를 찾는 시간을 줄이기 위해서
```

Agile estimation

BA/PM이 스토리에 대해 설명

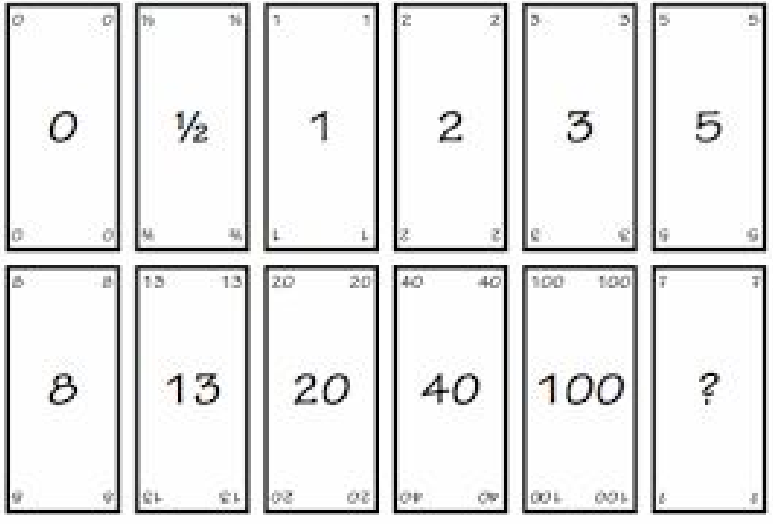
Dev는 해당 스토리에 대해 **스토리포인트** 산정

동시에 자신의 스토리포인트를 던짐

토론 및 설득

다시 추정.. 모두가 동의 할 때까지

Planning poker Game



스토리의 크기는 다른 스토리에 상대적이다.

Tip

가장 작아 보이는 스토리를 찾아라

그 스토리를 기준으로 크기를 측정하라

Planning game

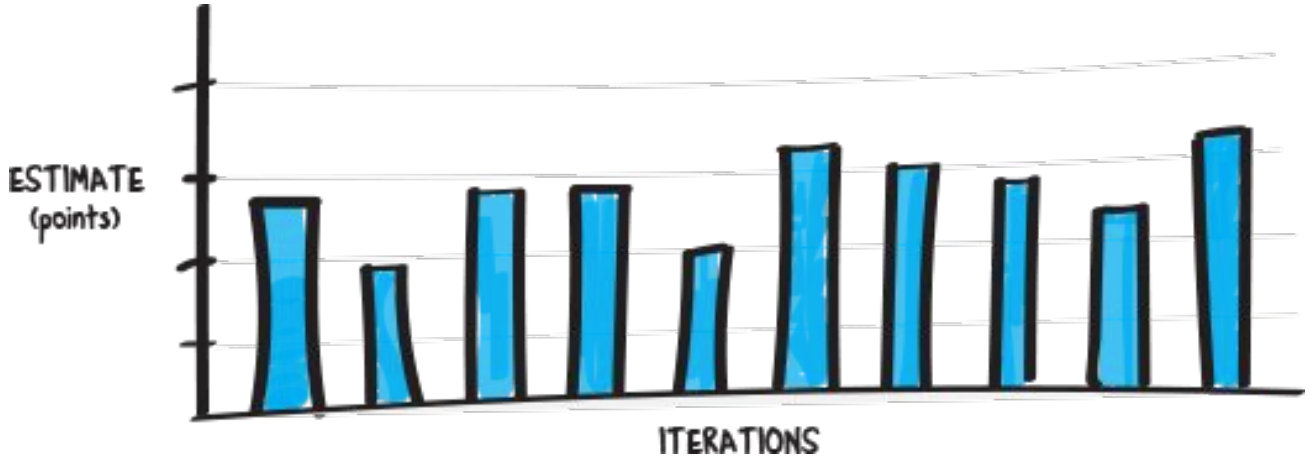
토론을 통해 여러 전문가의 의견을 모으는데 효과적

자신의 추정치에 대해 설득하며 정확도를 높여 나감

Team velocity

the amount of work that we can complete in an iteration

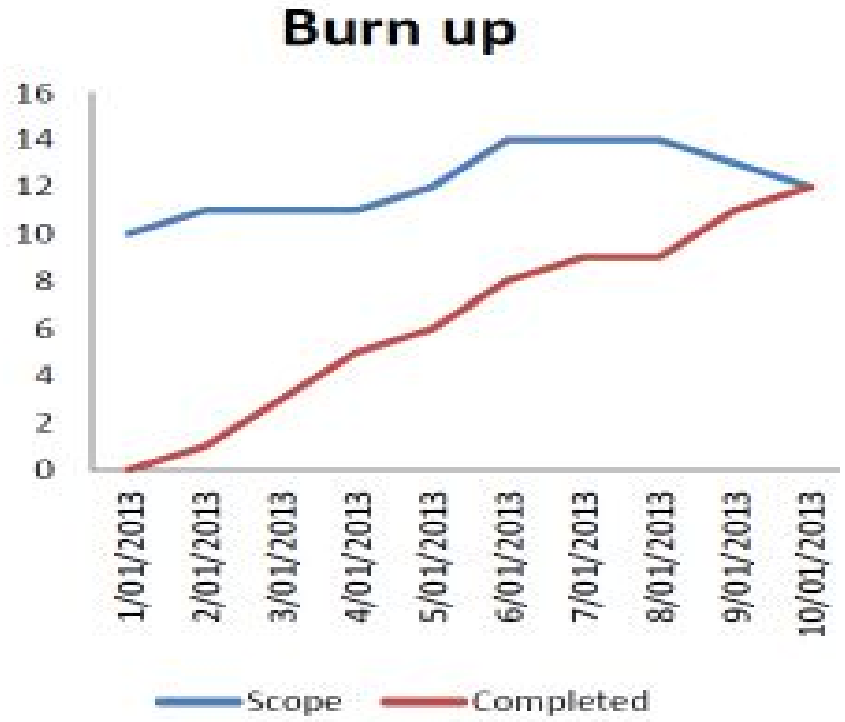
Team velocity = completed user-stories / iteration



How many iterations?

Total Story Points / Team Velocity = Number of iterations

Burn down vs. Burn up



Planning poker game

과일을 먹어봅시다

주어진 모든 과일을 팀원들과 나눠서 먹으면 됩니다.

10분씩 나눠서 3번을 먹는 다고 가정해 봅시다.

(즉, 3번의 Iteration을 돌아요. 한 Iteration 당 10분)

과일 한 개/알/통(한 송이X)을 얼마만에 먹을 수있는지 추정해 봅시다.

피보나치 수로 포인트를 산정해 주세요.

1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144……

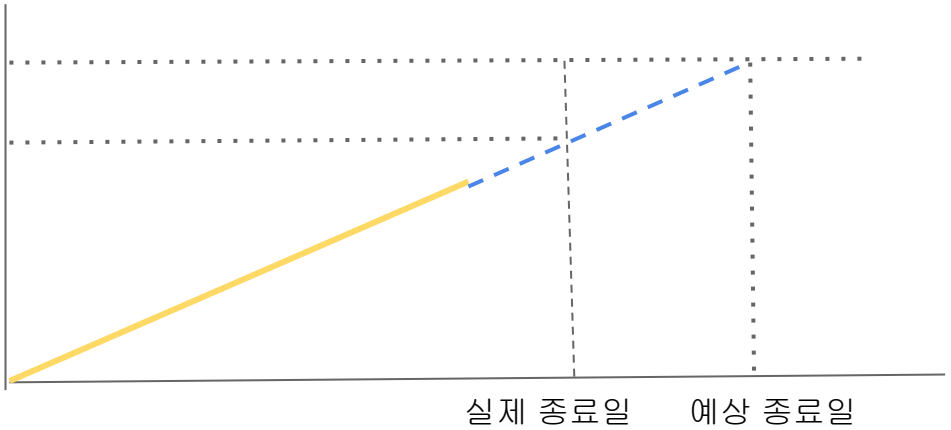
추정한 포인트를 팀간 비교하는게 맞는가?

이터레이션마다 velocity가 큰 차이가난다면 제대로 추정한것이 맞는가?

파보나치수가 의미하는 것이 무엇인가? 큰스토리는 작게 나누자

우리가 할 수 있는 일은 뭐가 있을까?

- 사람을 더 투입?
- 종료일 연기?
- 품질 낮추기?
- 일의 양(scope) 줄이기?

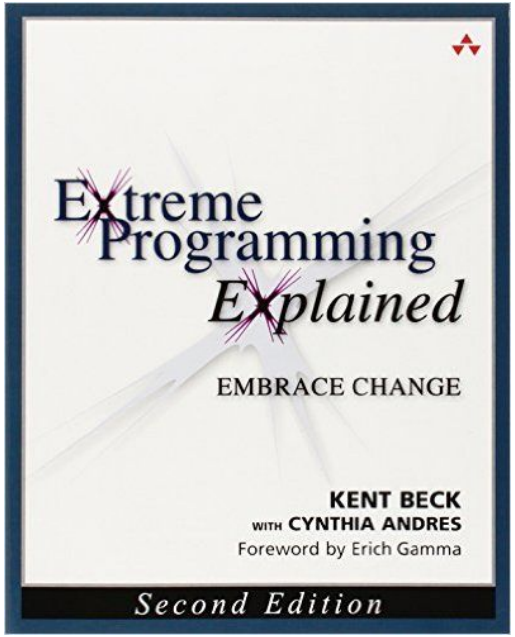


Engineering practices

Extreme programming (xp)

Primarily a developer-centric approach

- Test Driven Development (TDD)
- Unit tests
- Pairing
- Continuous Integration (CI)
- Refactoring




YES
NO
YES
NO

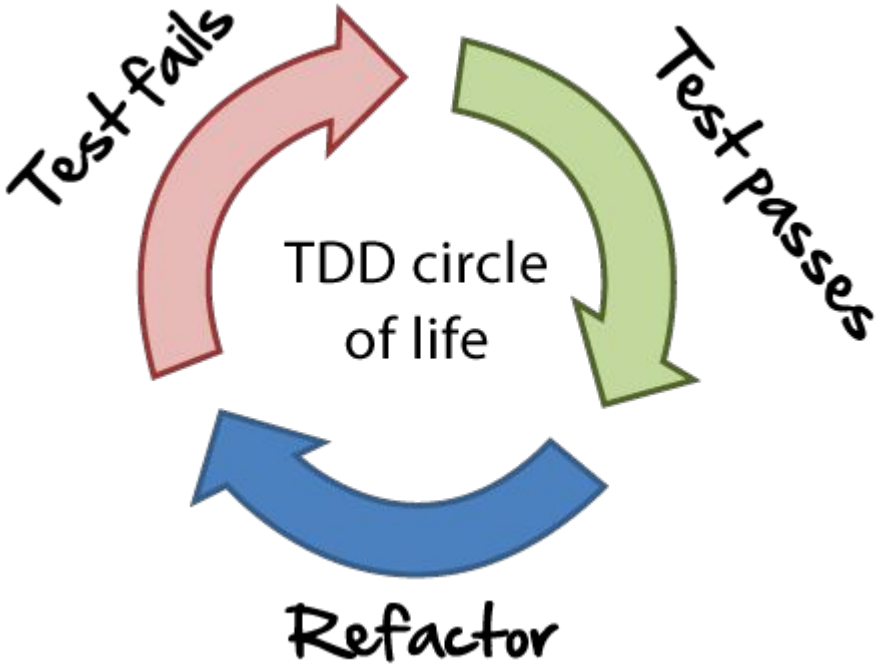
잘못 된 것을 언제 알게 될까??

잘못된 설계로 개발했다면?

오류가 있는 코드를 커밋했다면?

내가 고친 코드가 다른 팀에 영향을 준다면?

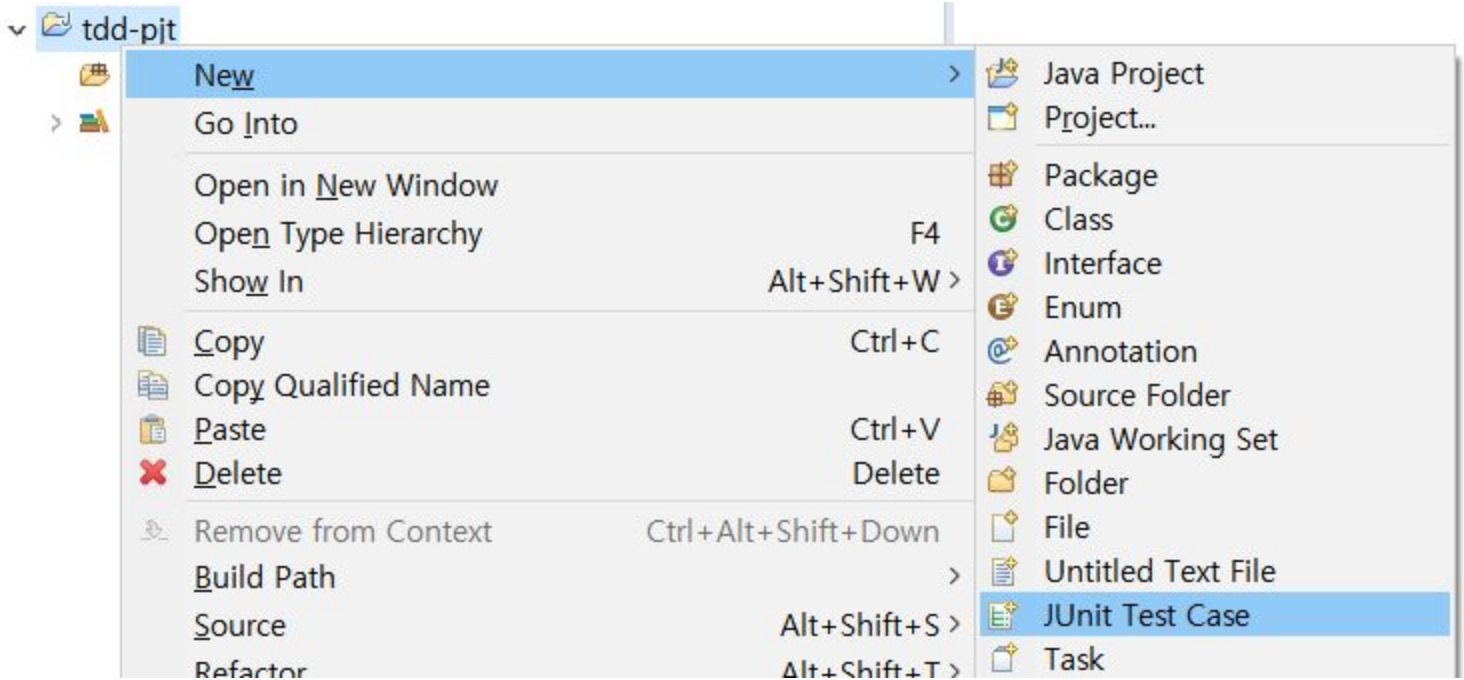
TDD - Test Driven development

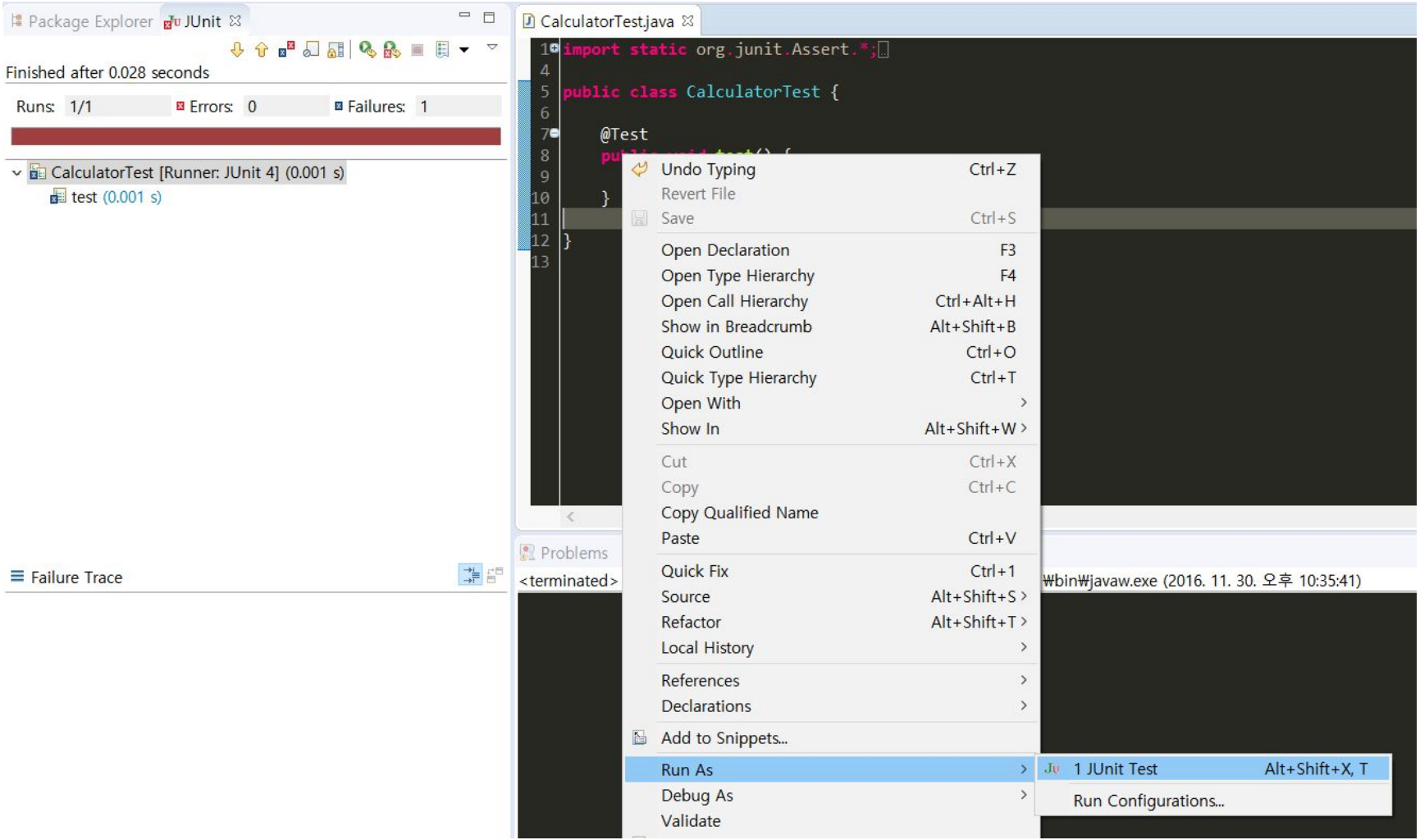


계산기 모듈 좀 짜주세요!!

```java
public class Calculator {
    public int add(int a, int b) {
        return a - b;
    }
}
```

LET'S TDD


tdd-pjt
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Java Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
JUnit Test Case
Task

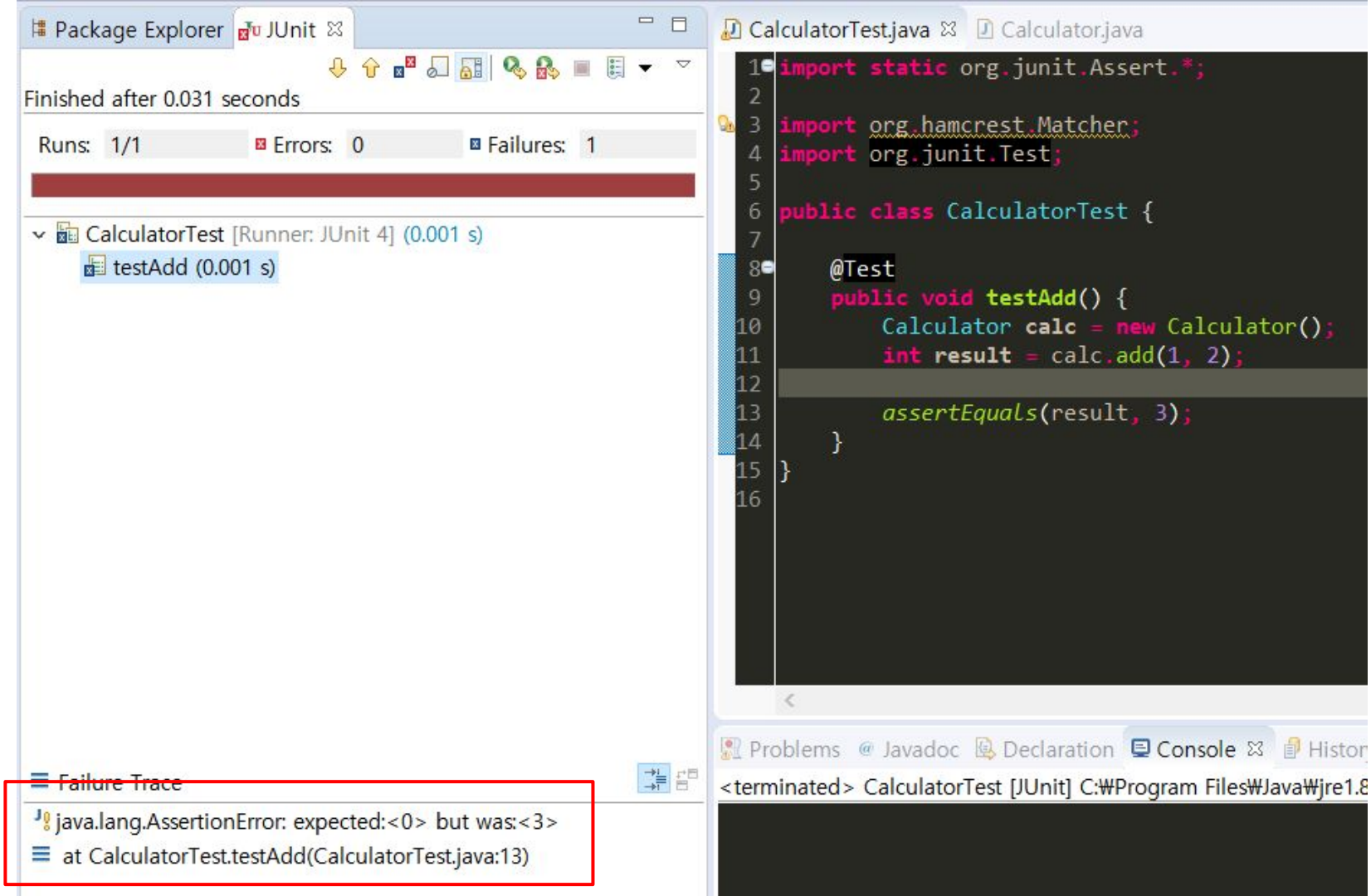

Package Explorer
JUnit
Finished after 0.028 seconds
Runs: 1/1
Errors: 0
Failures: 1
CalculatorTest [Runner: JUnit 4] (0.001 s)
test (0.001 s)
Failure Trace
CalculatorTest.java
import static org.junit.Assert.*;
public class CalculatorTest {
@Test
}
Undo Typing Ctrl+Z
Revert File
Save Ctrl+S
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Show in Breadcrumb Alt+Shift+B
Quick Outline Ctrl+O
Quick Type Hierarchy Ctrl+T
Open With
Show In Alt+Shift+W
Cut Ctrl+X
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Quick Fix Ctrl+1
Source Alt+Shift+S
Refactor Alt+Shift+T
Local History
References
Declarations
Add to Snippets...
Run As
Debug As
Validate
1 JUnit Test Alt+Shift+X, T
Run Configurations...
Problems
<terminated>
₩bin₩javaw.exe (2016. 11. 30. 오후 10:35:41)

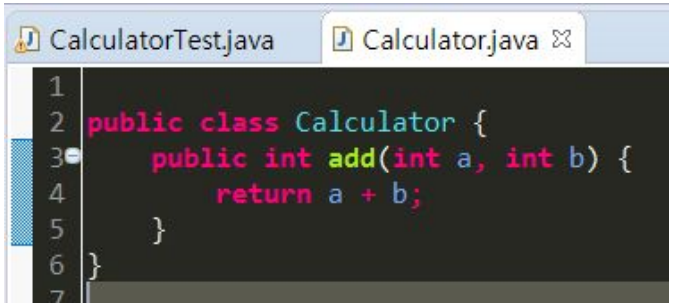
*CalculatorTest.java

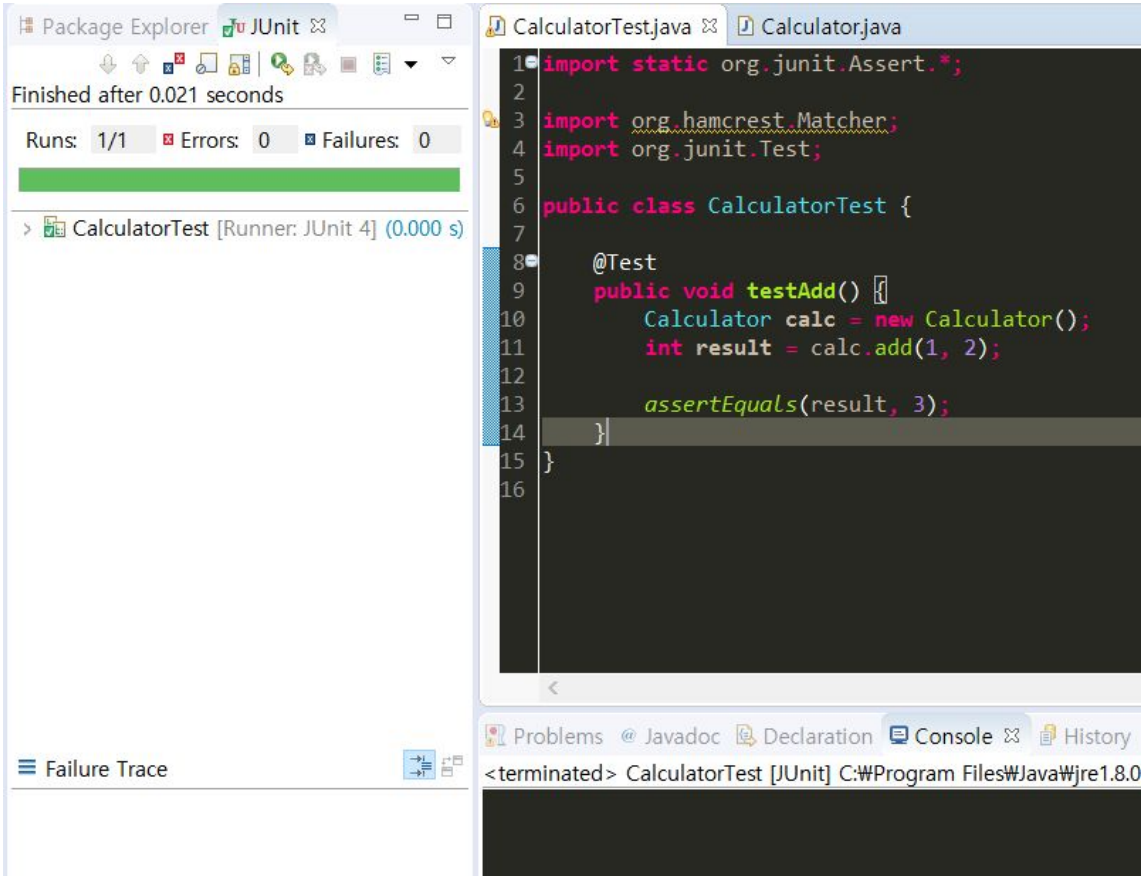
```java
import static org.junit.Assert.*;

import org.hamcrest.Matcher;
import org.junit.Test;

public class CalculatorTest {

    @Test
    public void testAdd() {
        Calculator calc = new Calculator();
        int result = calc.add(1, 2);

        assertEquals(result, 3);
    }
}
```

CalculatorTest.java | Calculator.java

```java

public class Calculator {
    public int add(int a, int b) {
        return 0;
    }
}

```

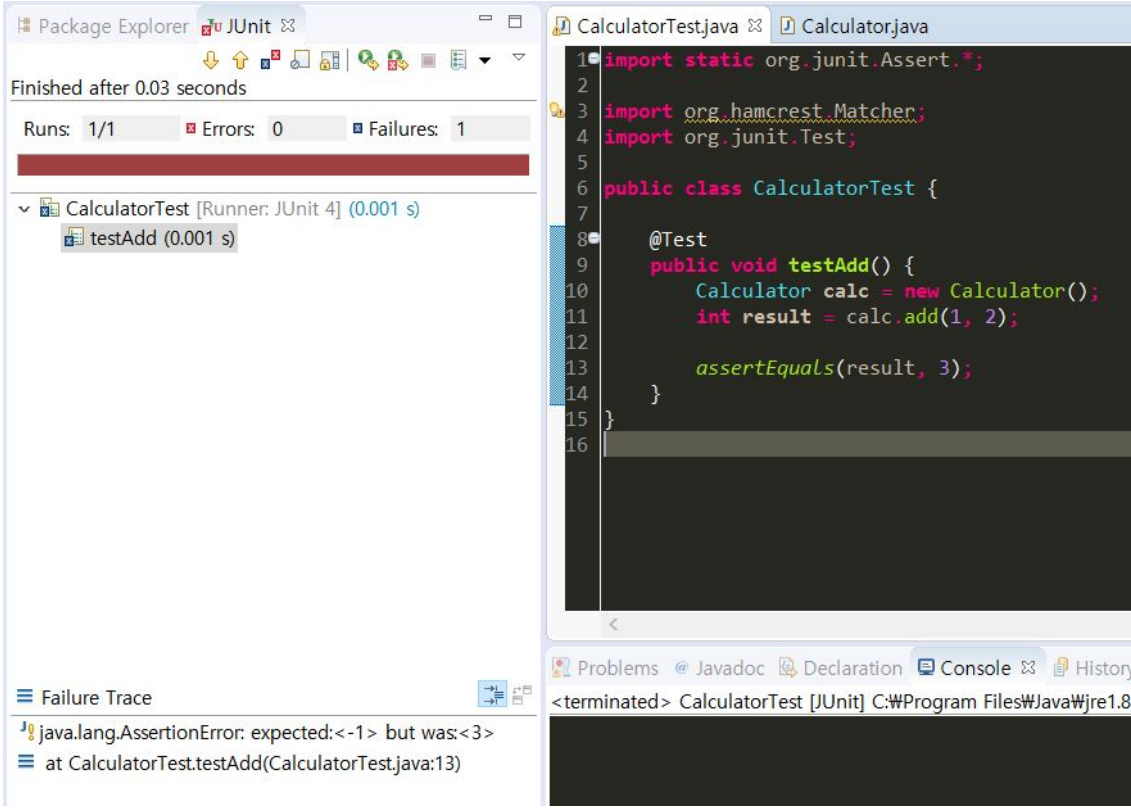

Package Explorer
JUnit
Finished after 0.031 seconds
Runs: 1/1
Errors: 0
Failures: 1
CalculatorTest [Runner: JUnit 4] (0.001 s)
testAdd (0.001 s)
Failure Trace
java.lang.AssertionError: expected:<0> but was:<3>
at CalculatorTest.testAdd(CalculatorTest.java:13)
CalculatorTest.java
Calculator.java
1 import static org.junit.Assert.*;
2
3 import org.hamcrest.Matcher;
4 import org.junit.Test;
5
6 public class CalculatorTest {
7
8 @Test
9 public void testAdd() {
10 Calculator calc = new Calculator();
11 int result = calc.add(1, 2);
12
13 assertEquals(result, 3);
14 }
15 }
16
Problems
Javadoc
Declaration
Console
<terminated> CalculatorTest [JUnit] C:₩Program Files₩Java₩jre1.8


CalculatorTest.java
Calculator.java
1
2 public class Calculator {
3 public int add(int a, int b) {
4 return a + b;
5 }
6 }
7


Package Explorer
JUnit
Finished after 0.021 seconds
Runs: 1/1
Errors: 0
Failures: 0
CalculatorTest [Runner: JUnit 4] (0.000 s)
Failure Trace
CalculatorTest.java
Calculator.java
1 import static org.junit.Assert.*;
2
3 import org.hamcrest.Matcher;
4 import org.junit.Test;
5
6 public class CalculatorTest {
7
8 @Test
9 public void testAdd() {
10 Calculator calc = new Calculator();
11 int result = calc.add(1, 2);
12
13 assertEquals(result, 3);
14 }
15 }
16
Problems
Javadoc
Declaration
Console
History
<terminated> CalculatorTest [JUnit] C:₩Program Files₩Java₩jre1.8.0

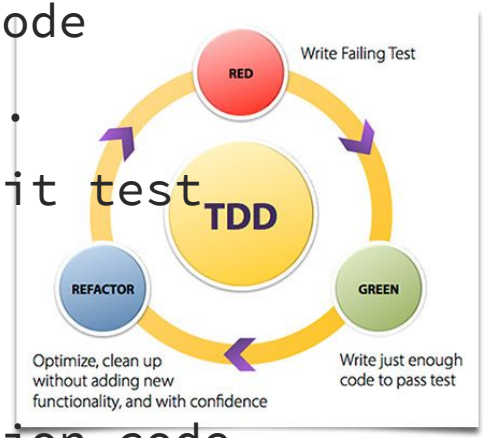
실수로 잘 못 고쳤더라도..

```java
public class Calculator {
    public int add(int a, int b) {
        return a - b;
    }
}
```



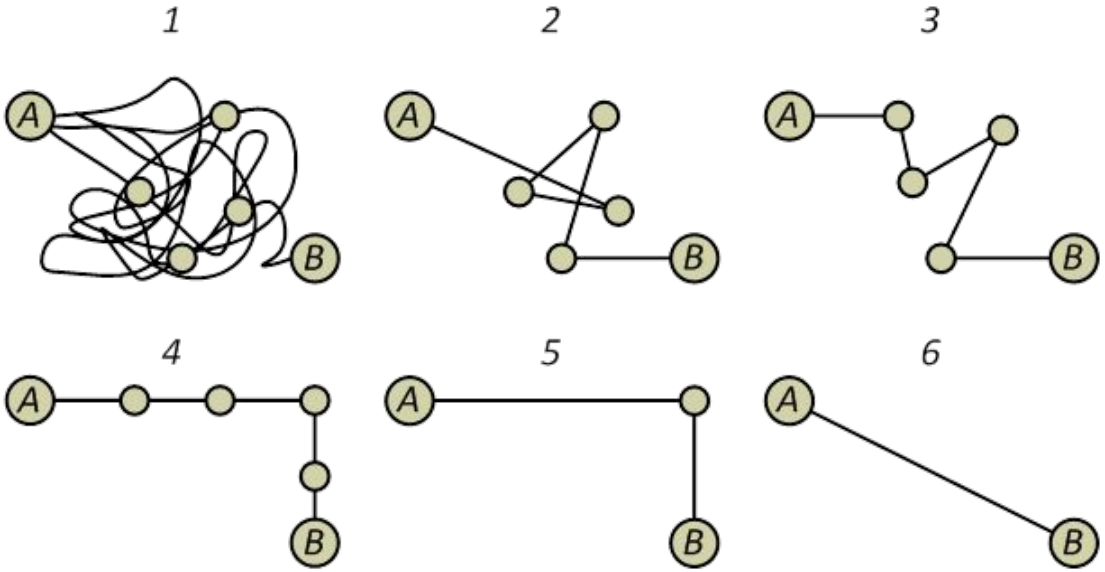
The Three Rules Of Tdd - uncle bob

1. You are not allowed to write any production code
   unless it is to make a failing unit test pass.
2. You are not allowed to write any more of a unit test
   than is sufficient to fail;
   and compilation failures are failures.
3. You are not allowed to write any more production code
   than is sufficient to pass the one failing unit test.



REFACTORING

소프트웨어를 보다 쉽게 이해할 수 있고, 적은 비용으로 수정할 수 있도록 겉으로 보이는 동작의 변화 없이 내부 구조를 변경하는 것



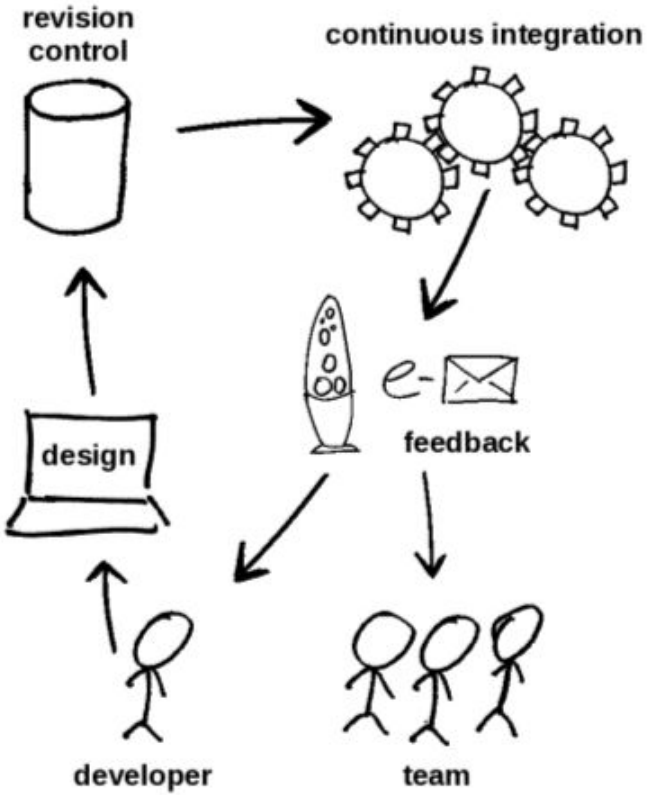
REFACTORING

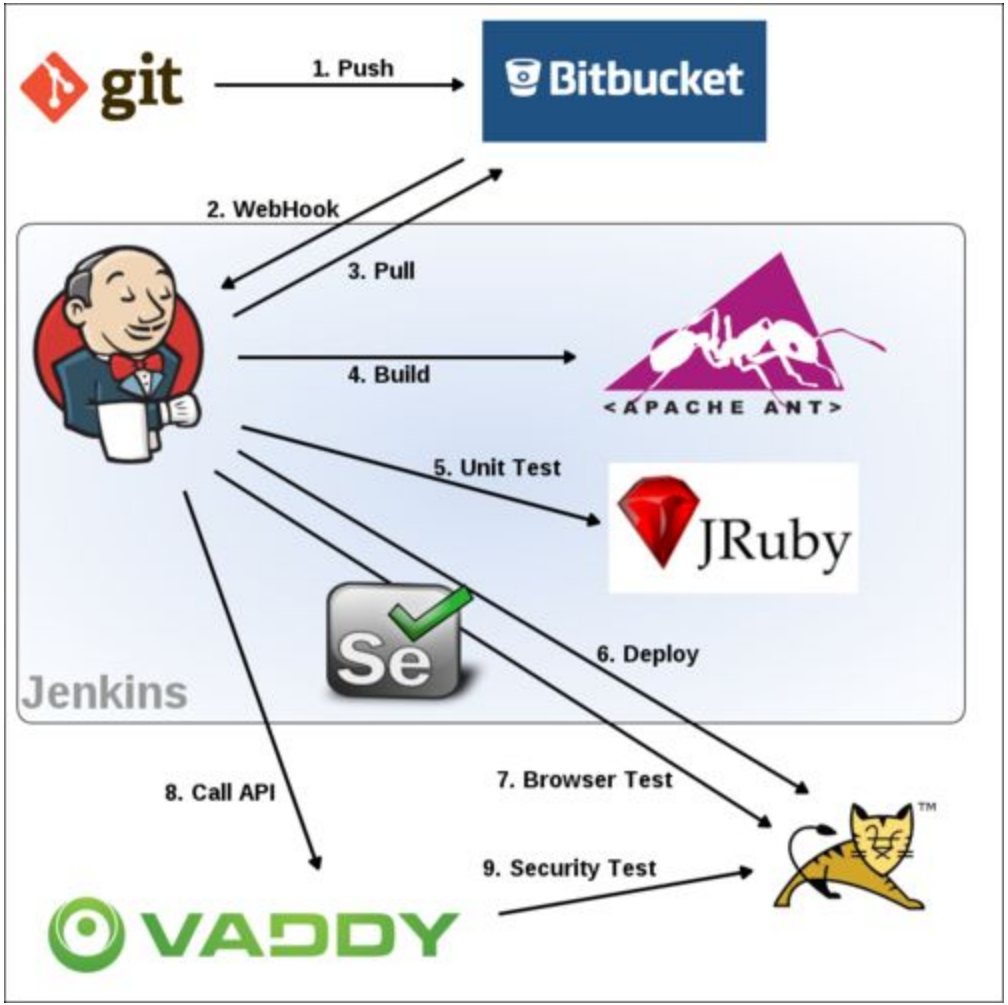
리팩토링을 장려하는 문화가 필요

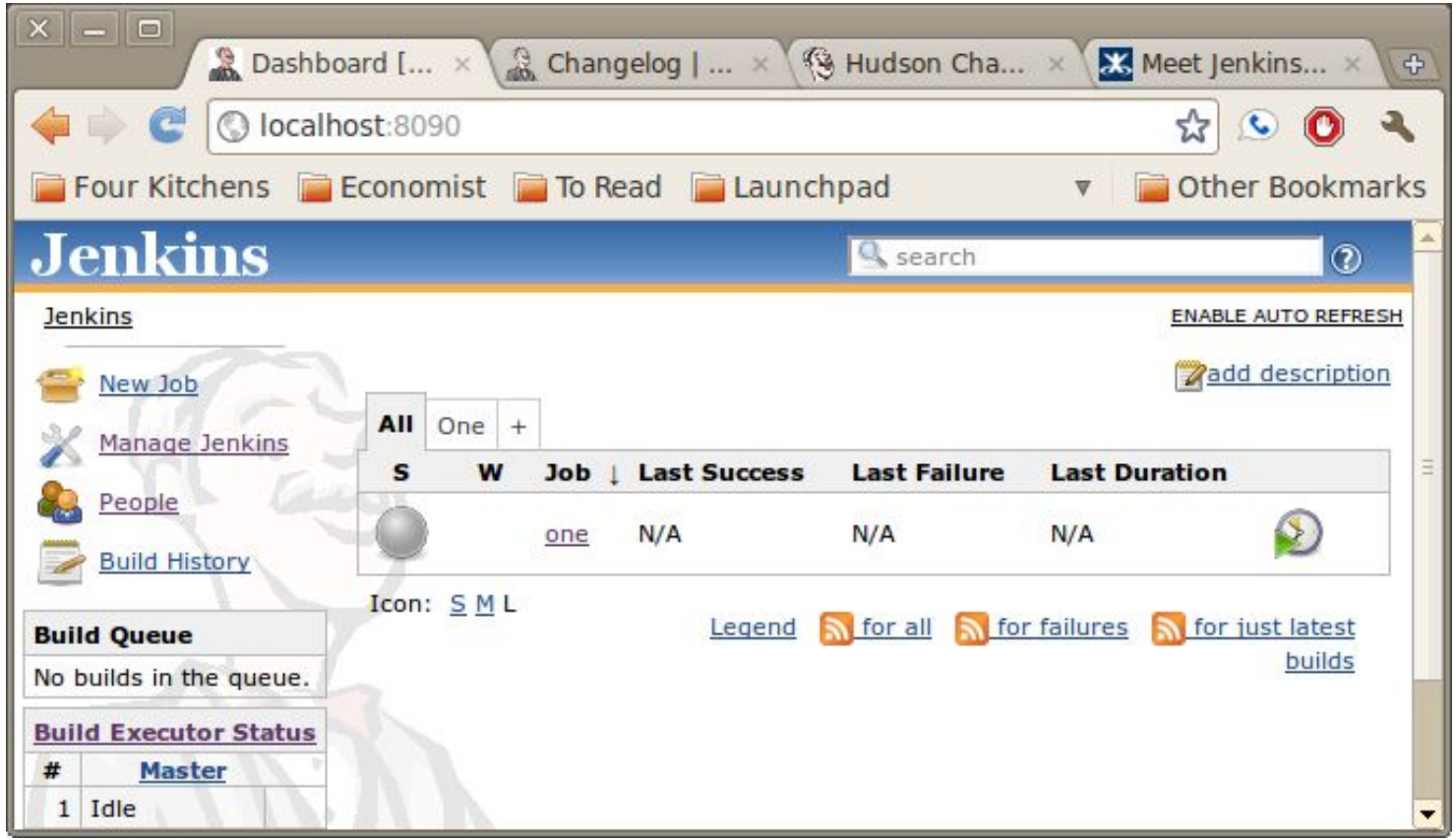
리팩토링을 위해서 테스트가 기반되어야 됨

Collective Code Ownership

Continuous integration




git
1. Push
Bitbucket
2. WebHook
3. Pull
4. Build
<APACHE ANT>
5. Unit Test
JRuby
Se
6. Deploy
Jenkins
8. Call API
7. Browser Test
9. Security Test
VADDY


Dashboard [...
Changelog | ...
Hudson Cha...
Meet Jenkins...
localhost:8090
Four Kitchens
Economist
To Read
Launchpad
Other Bookmarks
Jenkins
search
Jenkins
ENABLE AUTO REFRESH
New Job
Manage Jenkins
People
Build History
add description
All
One
+
S
W
Job
Last Success
Last Failure
Last Duration
one
N/A
N/A
N/A
Icon: S M L
Legend
for all
for failures
for just latest builds
Build Queue
No builds in the queue.
Build Executor Status
#
Master
1
Idle


go
PIPELINES
ENVIRONMENTS
AGENTS
ADMIN
[redacted]_API
PAUSE
Build_and_test
Int_Deploy
UAT_Deploy
Prd_Deploy
205
revision: 70457a7fd...
4 days ago
Triggered by [redacted]
204
revision: b5d83bc5...
4 days ago
Triggered by [redacted]
203
revision: 25875e37...
5 days ago
Triggered by [redacted]

Pair programming

품질 향상, 지식 공유, 심리적 안정감

피곤함, 생산성

끊임 없이 대화 하기

서로 신뢰, 존중 하기

COMMUNICATION

Daily stand up

- What did you do **yesterday**?
- What are you going to do **today**?
- What **obstacles** or impediments are preventing you from making progress?

Daily stand up

오직‘돼지’만이 스크럼 스탠드 업 미팅에 참석하는 것이 허용된다.

닭 : 우리 식당 같이 해보지 않을래 ?
돼지 : 좋은 생각이야. 그런데 그 식당 이름은 뭐라고 하지 ?
닭 : “햄과 달걀” 이라고 부르는 것이 어떨까!
돼지 : 글쎄.. 좋은 생각이 아닌 것 같아. 난 희생해야 하는데 넌 단지 관여만 하려고 하잖아!

Showcase

각 Iteration이 끝날때마다 잘 작동하는 Software를 갖는 것이 목표

설계나 브레인스토밍을 하는 시간이 아니다.

지난 스토리에 대해 피드백을 얻을 수 있다.

피드백은 새로운 스토리로 작성한다.

Chickens을 위한 자리다.

Iteration planning meeting

다음 Iteration에서 스토리를 어떻게 작업할지 계획

개발자들에게 스토리를 설명해준다.

개발자들은 스토리별로 스토리 포인트를 산정한다.

경우에 따라 스토리의 우선순위를 결정한다.

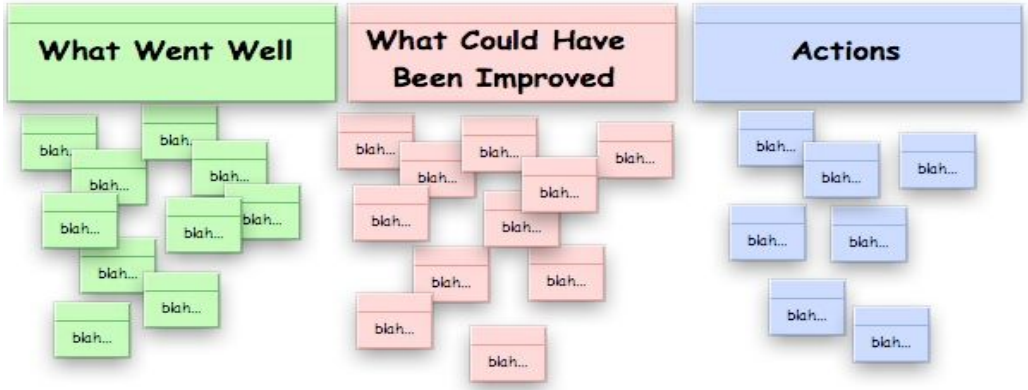
이슈나 장애물이 있는지 파악한다.

RETROSPECTIVE

각 Iteration이나 Release가 끝나는 시점에 진행

지난 일들을 되돌아보고 개선할 일들을 생각하는 것

Good / Bad
Keep / Problem / Try
Keep doing / Stop Doing / Start Doing



RETROSPECTIVE

Wrap up

Agile manifesto

Individuals and interactions

over processes and tools

Working software

over comprehensive documentation

Customer collaboration

over contract negotiation

Responding to change

over following a plan

http://agilemanifesto.org/iso/kor/

애자일 프로젝트로 안내해 줄 지도같은건 없다. 여러분 스스로 여러분의 팀과 프로젝트에 맞는 최고의 방법이 무엇인지 알아내야 한다.

누구를 설득하려고 하지 말자.
다른 사람들에게 그들이 무슨일을 해야
한다고도 말하지 말자.
그 대신, 행동을 보여주자.
다른 사람들이 항상 보고 있지는 않겠지만,
그저 여러분이 해야할 일을 해나가는 것이다.

애자일은 여행의 과정이지 목적지가 아니다.
애자일화 되는 것이 목적이 아니라
훌륭한 제품을 개발해서 고객에게 최상의
서비스를 제공하는 것이 목적이라는 것을
꼭 기억하도록 하자.

실천법에 너무 목매지 말자.
여러분만의 상황과 맥락에 적합하게 적용하기 바란다.
그러다가도 여러분이 과연 애자일을 잘 실천하고 있는지 궁금해진다면 다음의 두 가지 질문을 스스로에게 해보자.

우리는 매주 가치 있는 것을 고객에게 전달는가?

우리는 계속 발전하기 위해 노력하고 있는가?

만약 이 질문에 **네**라고 대답할 수 있다면,

여러분은 애자일을 하고 있는 것이다.

Thank you.