

Technology-Driven Development: Using Automation and Development Techniques to Grow an Agile Culture

Jul/29/2014

Hiroyuki Ito

Development Process Optimization Department, Rakuten, Inc.

<http://www.rakuten.co.jp/>

About me



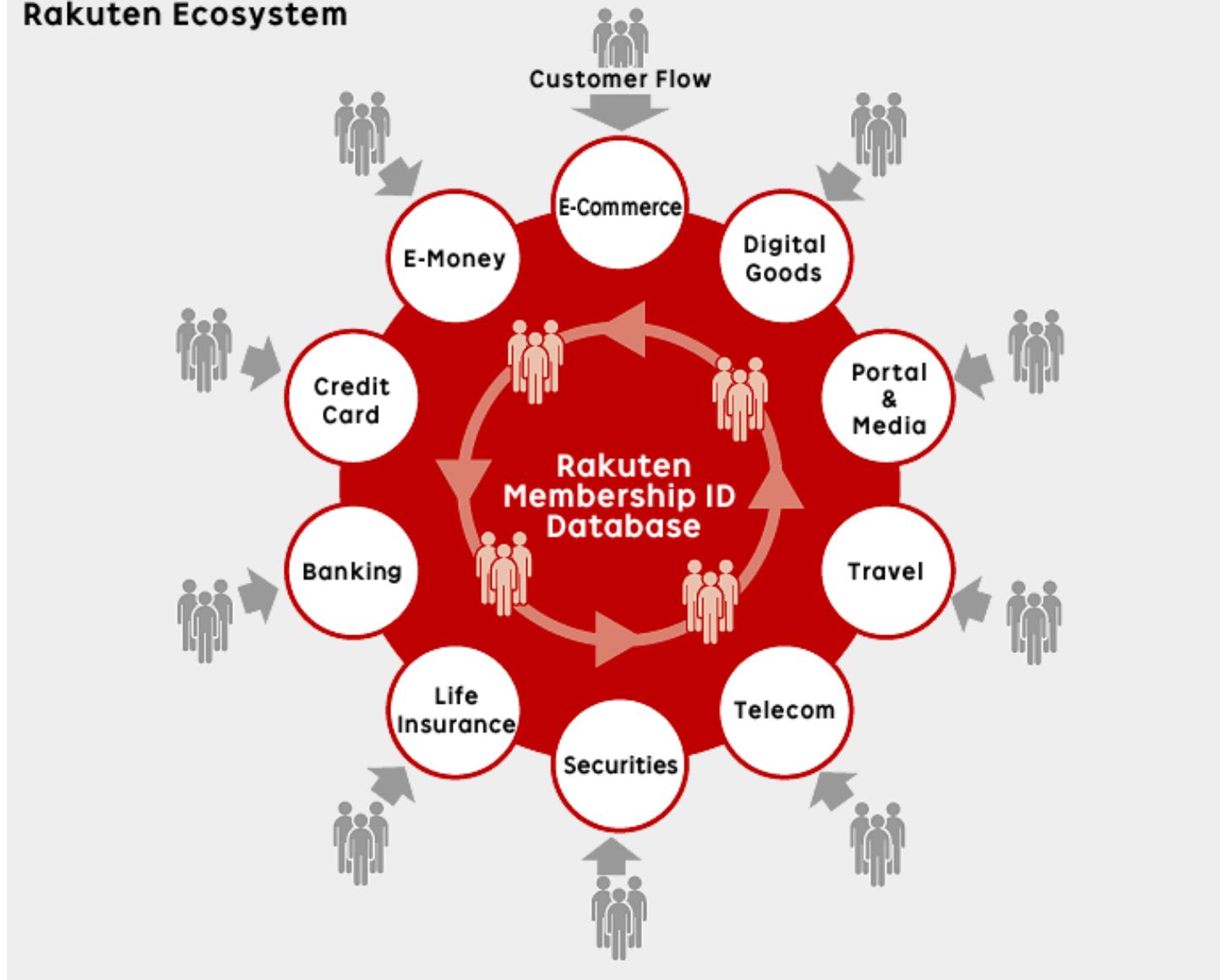
Hiroyuki Ito (The Hiro)

[@hageyahhoo](https://twitter.com/hageyahhoo)

**Test-Driven
Development Group**



Rakuten Ecosystem



[http://global.rakuten.com/corp/about/strength/
business model.html](http://global.rakuten.com/corp/about/strength/business_model.html)

It's my 3rd time to be here!

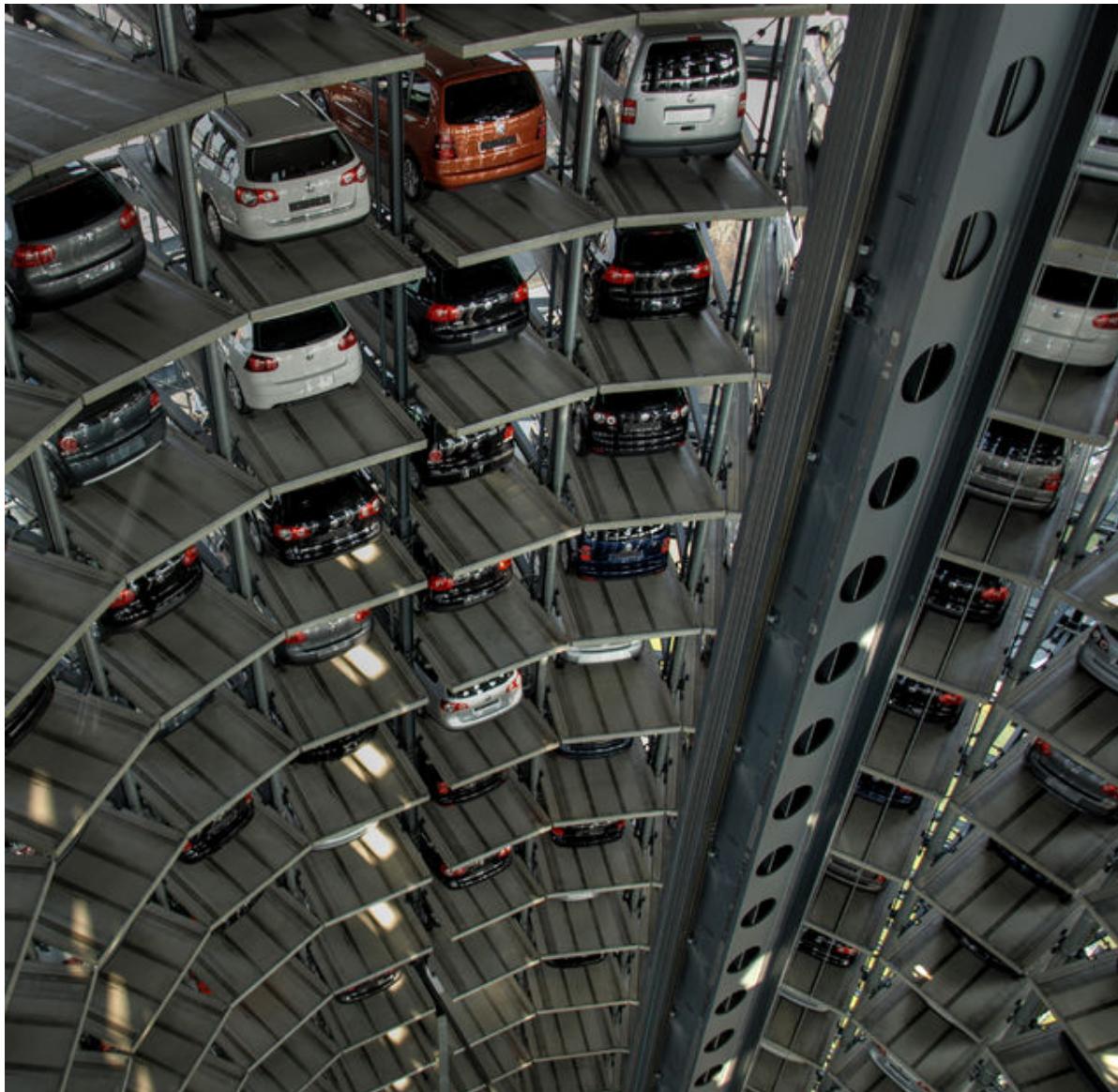


Agile2014 : as a Speaker

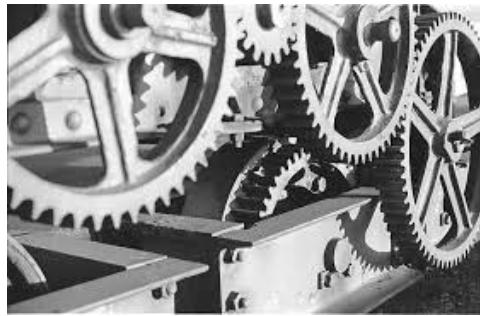
This session's theme

Technology- Driven Development

Additional possibilities of automation



“TDD” stands for three purposes



Efficiency



Learning



Collaboration

By three approaches

CI/CD



TDD



BDD



Three approaches by

CI/CD



TestFlight
iOS Beta Testing On The Fly

TDD



BDD

Cucumber

Agenda

1. Conditions and Challenges

2. CI/CD

3. TDD

4. BDD

5. Results, Problems, Possibility and Future

6. Conclusions

1. Conditions and Challenges

2. CI/CD

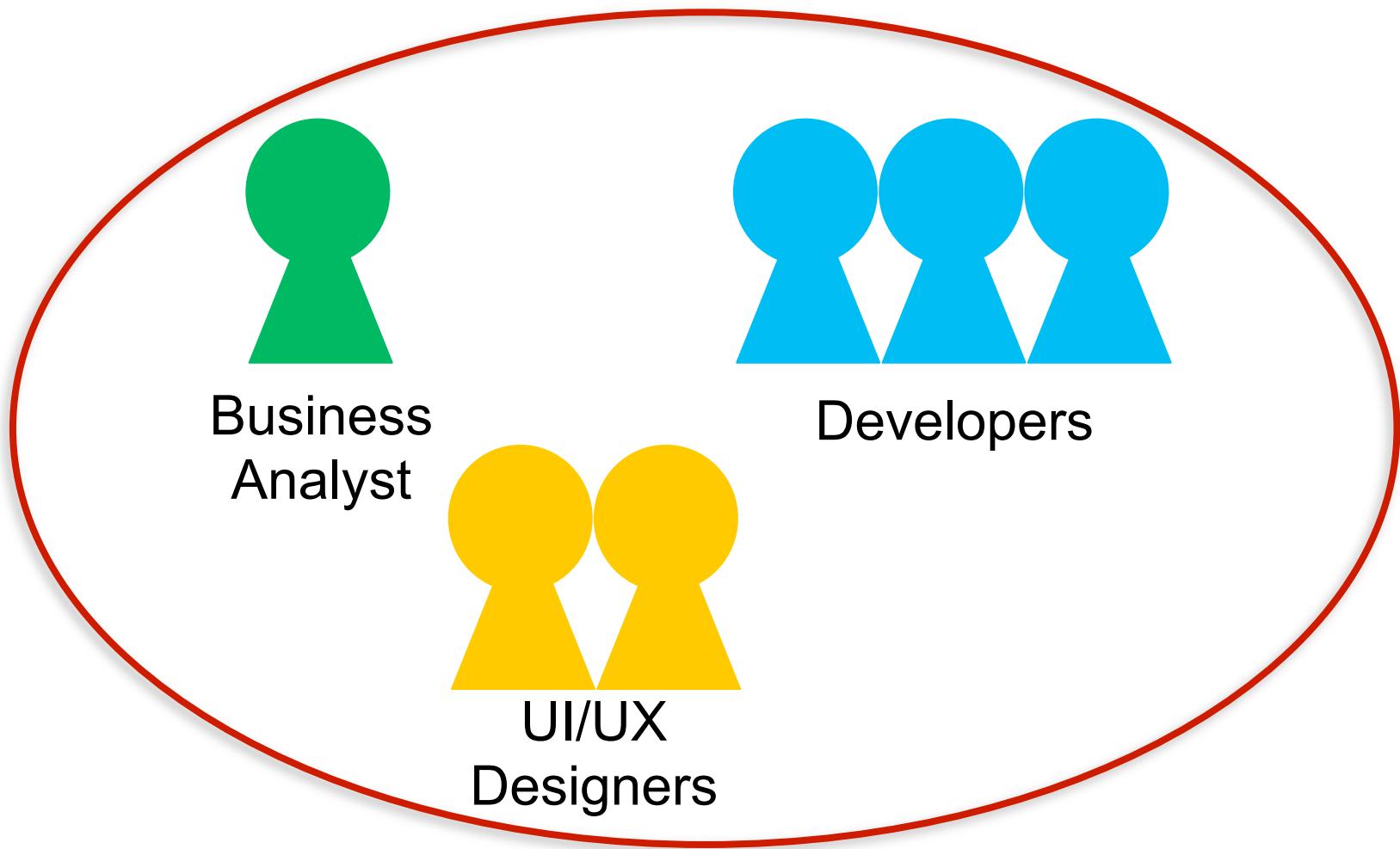
3. TDD

4. BDD

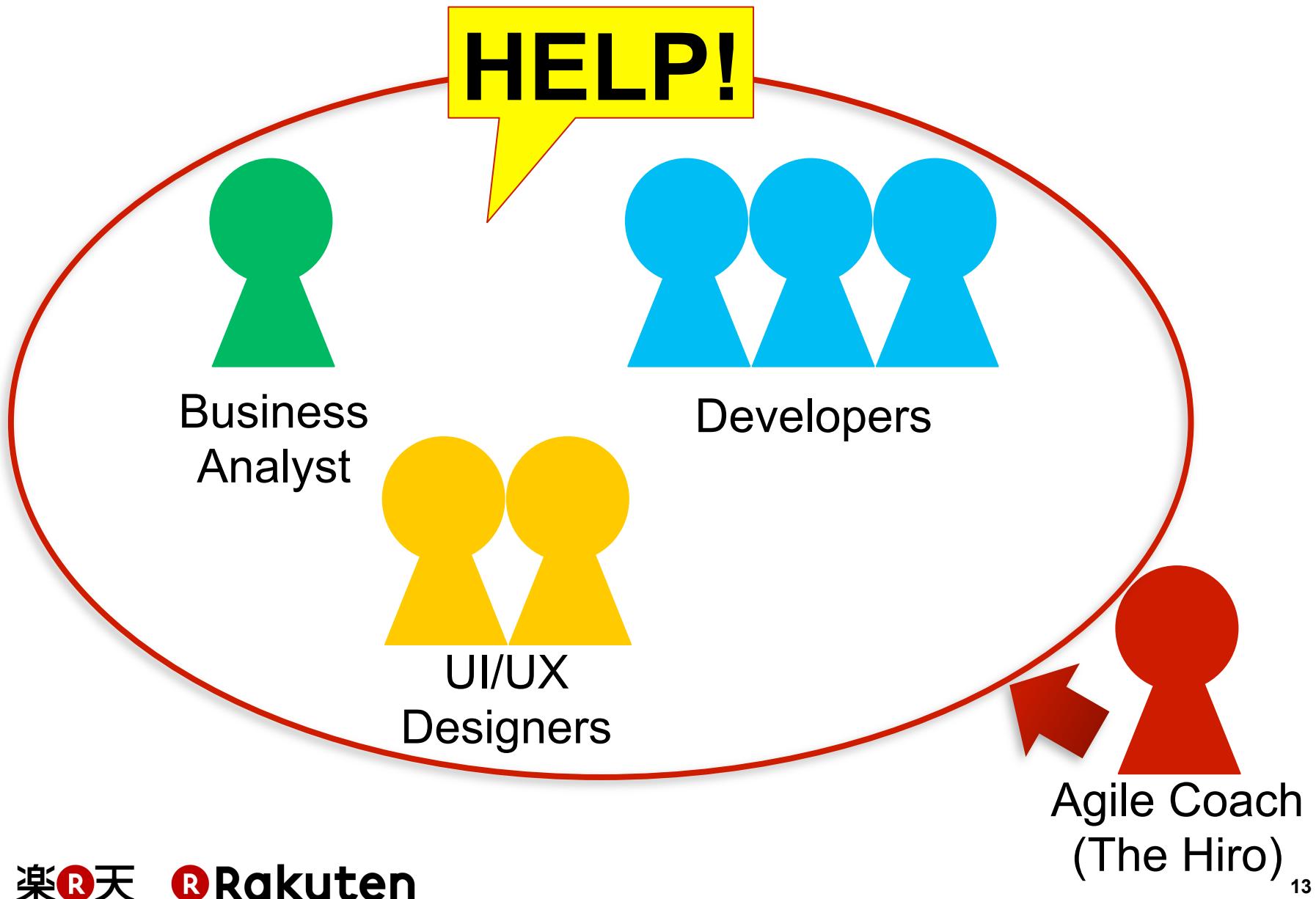
5. Results, Problems, Possibility and Future

6. Conclusions

At the end of April 2013



At the end of April 2013



Our target application is



Conditions and Challenges



Conditions and Challenges

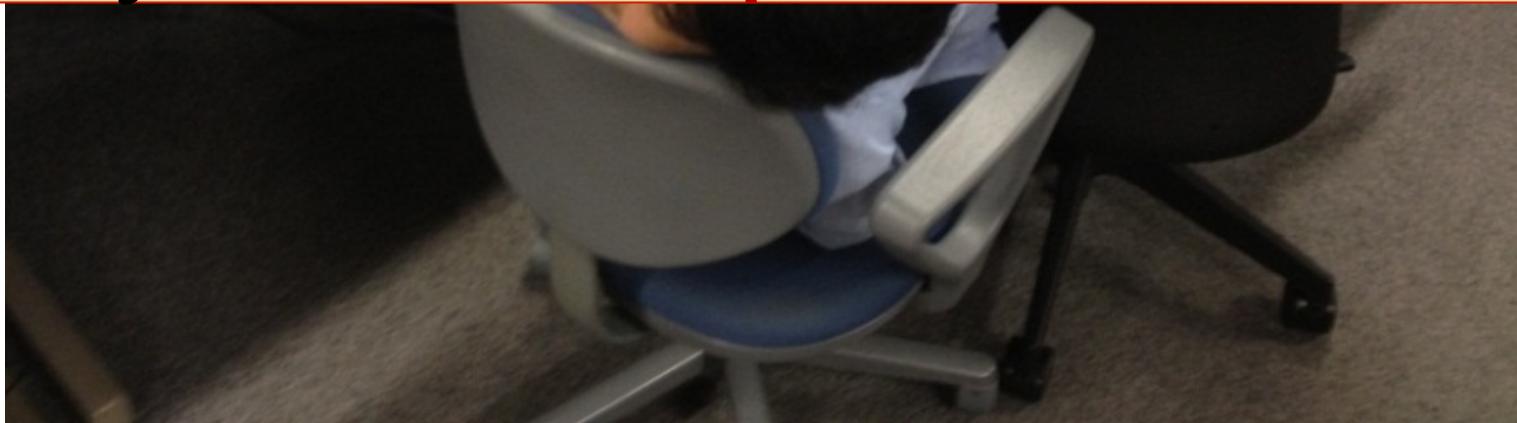
None of the team members had any experience with agile



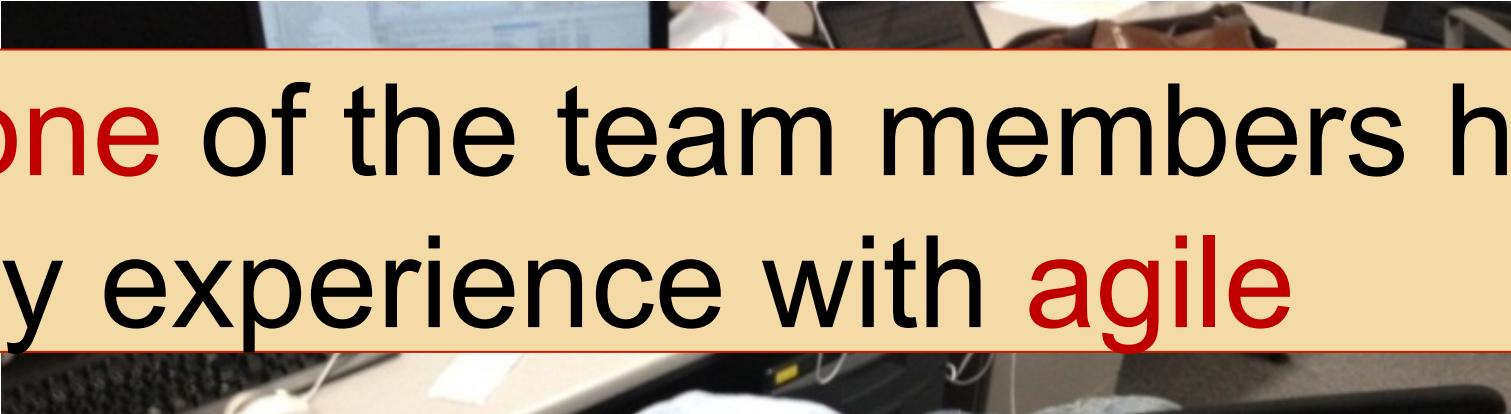
Conditions and Challenges

None of the team members had any experience with agile

There had been many manual operations



Conditions and Challenges

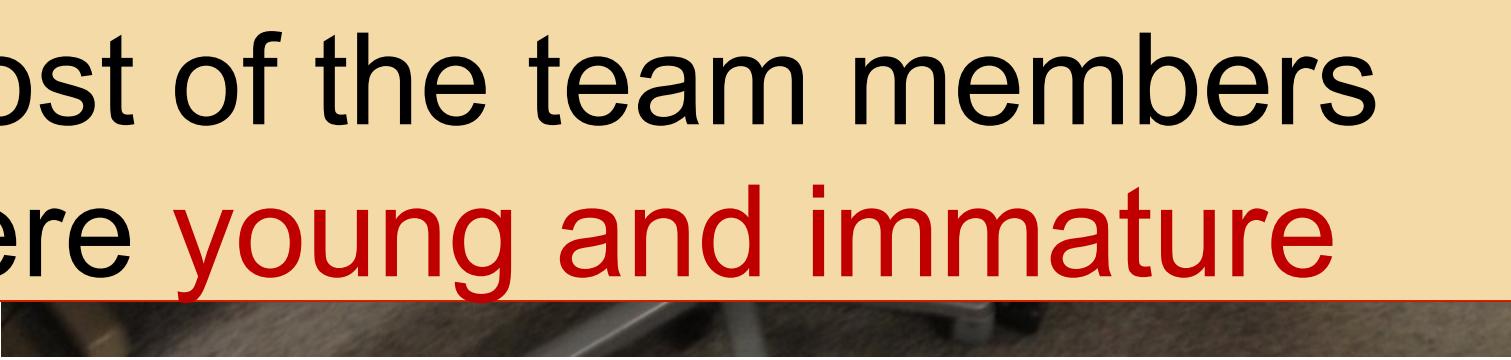


None of the team members had any experience with agile

There had been many manual operations



Most of the team members were young and immature



**What do you
think?**

I was so much excited!



WHY?

I can
achieve **anything**
through such a
challenging project!

Three approaches

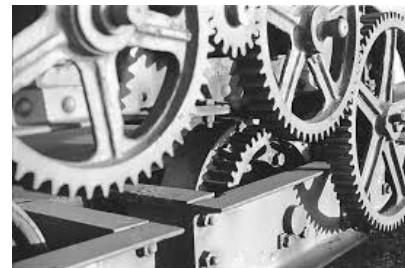
CI/CD



TDD



BDD



1. Conditions and Challenges

2. CI/CD

3. TDD

4. BDD

5. Results, Problems, Possibility and Future

6. Conclusions

Challenges



Low performance

- So many manual tasks



Going in circles

- No clear vision and no requirements
- No timely progress information

Before CI/CD

13.5 hours/week

- Change requests : 3 times/week
- Regression testing : **4 hours**/change
 - Need to retry if we find bugs...
- Install applications : **0.5 hour**/change
 - 5-minite work for 6 persons

The Implementation of CI/CD in our project

The Implementation of CI/CD in our project

My PC



The Implementation of CI/CD in our project

Atlassian



Check-in build (hourly)

My PC



The Implementation of CI/CD in our project

Atlassian



Check-in build (hourly)

My PC



Build applications
and run regression tests
automatically

The Implementation of CI/CD in our project

Atlassian



Check-in build (hourly)



Deliver to
all team members
automatically

My PC



Build applications
and run regression tests
automatically



The Implementation of CI/CD in our project

Atlassian



Check-in build (hourly)



Deliver to
all team members
automatically

My PC



Build applications
and run regression tests
automatically

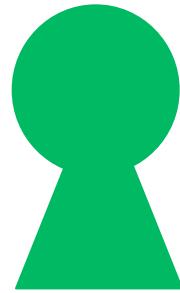


楽天

Rakuten

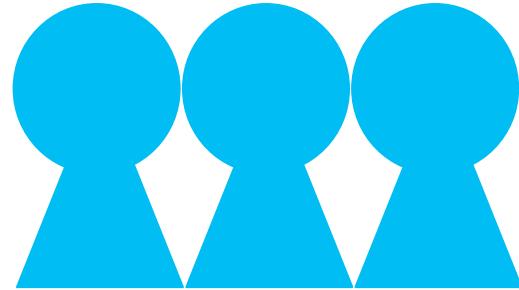
We demonstrate latest application
to the business analyst and managers
in every daily scrum

Shared understanding by the working software



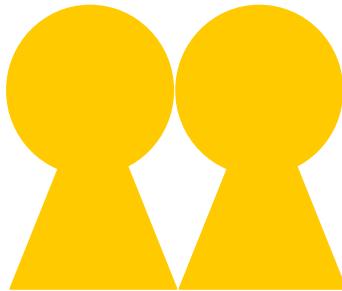
Business
Analyst

Know about
the progress



Developers

Get fast feedback



UI/UX
Designers

15 minutes/week

- Change requests : 3 times/week
- Regression testing : **3 minutes**/change
- Install applications : **2 minutes**/change

After CI/CD

15 minutes/week

- Change applications : 13 hours!
- Change environments : 2 minutes/change
- Change infrastructure : 2 minutes/change

1. Conditions and Challenges

2. CI/CD

3. TDD

4. BDD

5. Results, Problems, Possibility and Future

6. Conclusions

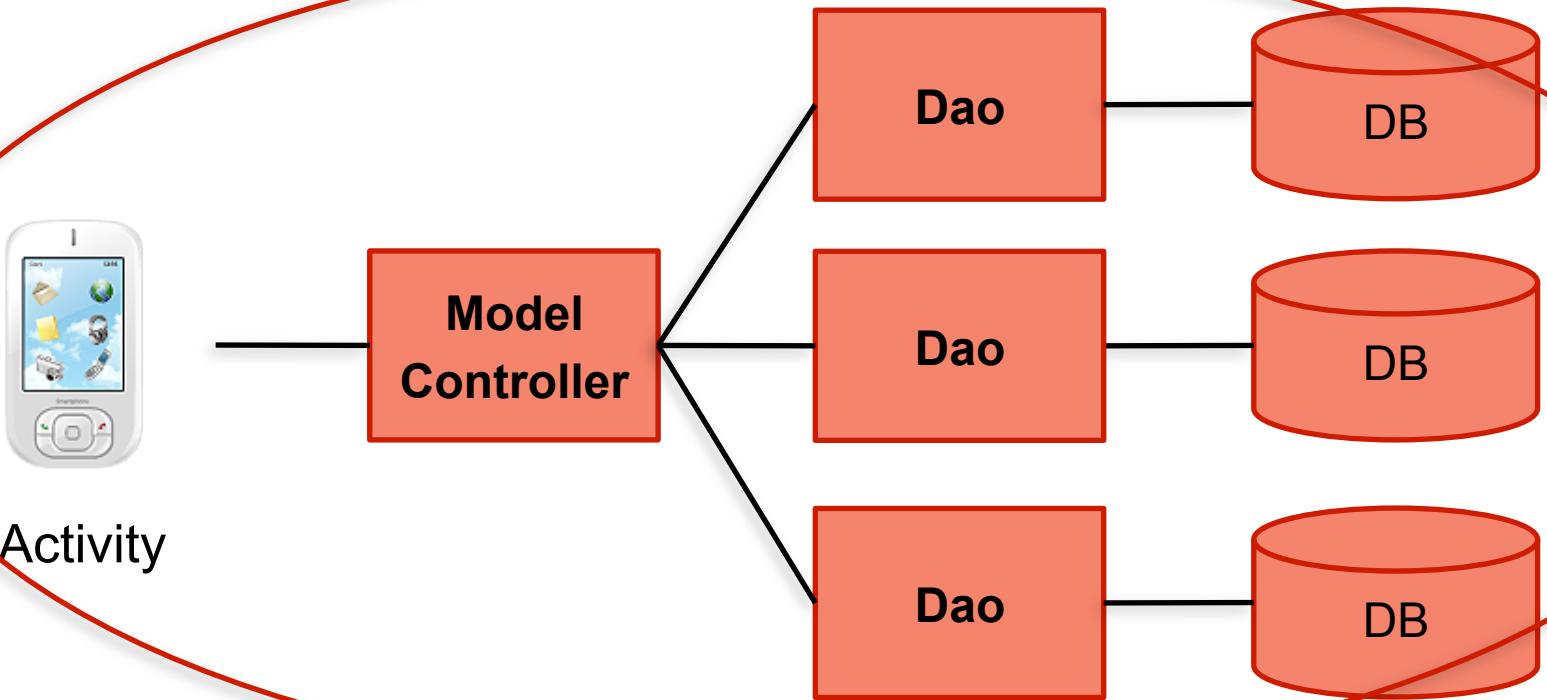
Challenge



Lack of skill and knowledge of Android

- the architecture of Android
- how to develop the Android application
- how to access the database on the device
- how to implement the UI

Before TDD



- Could not test after we implemented **all components**
(Debug Later Programming)
- It took **five days** to implement one activity set

Too difficult to use Android JUnit 😞



Too difficult to use Android JUnit ☺

java.lang.RuntimeException: Stub! (°Д°)



Too difficult to use Android JUnit ☹

java.lang.RuntimeException: Stub! (°Д°)

Why we need an emulator or a device? :-o

Too difficult to use Android JUnit ☹

java.lang.RuntimeException: Stub! (°Д°)

Why we need an emulator or a device? :-o

Please don't start a heavy lifecycle of Android
for each test case :-(<

Solution to do TDD on Android

Solution to do TDD on Android



- **Robolectric** : Do all unit testing only on JVM
 - <http://robolectric.org/>
 - Without any emulator or device!

Solution to do TDD on Android



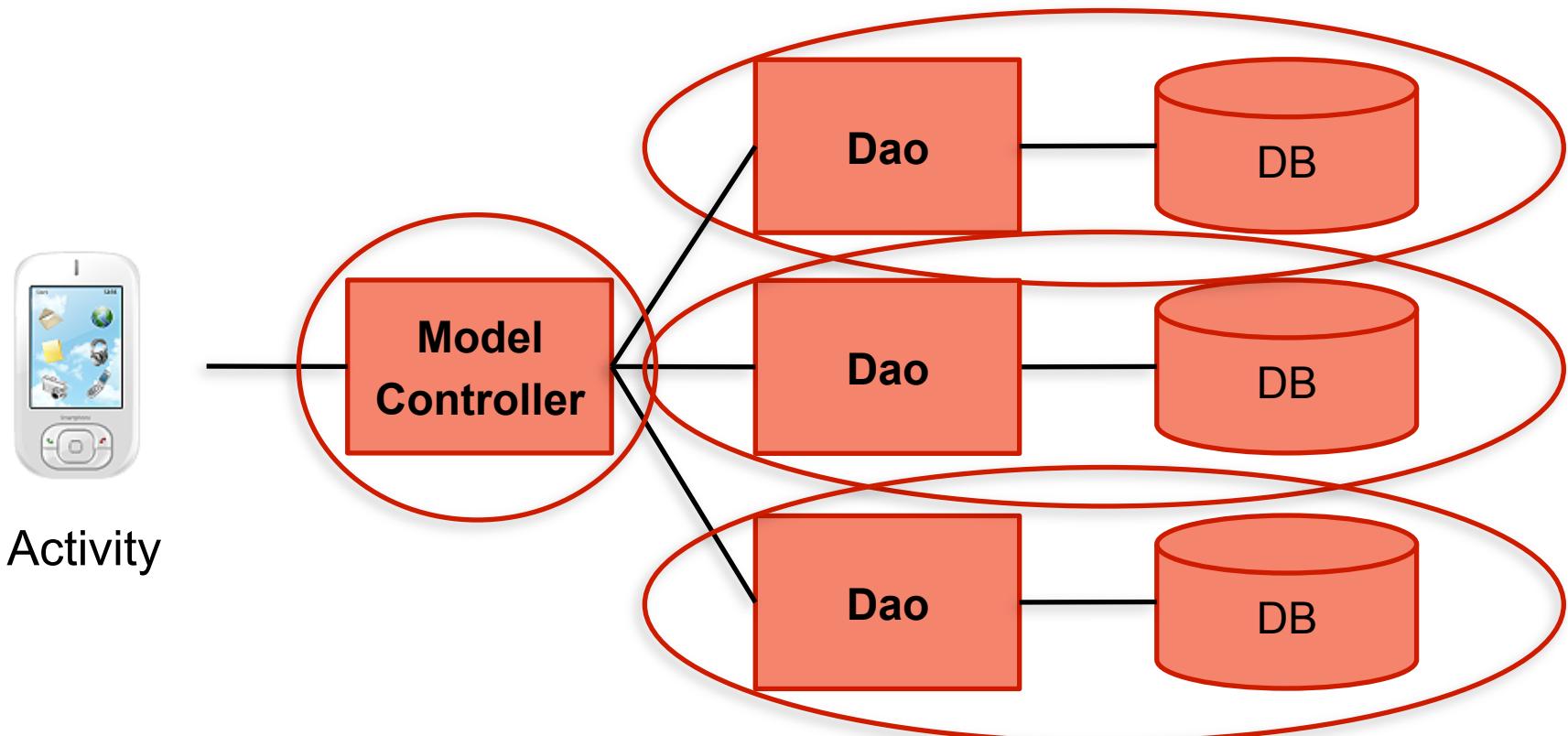
- **Robolectric** : Do all unit testing only on JVM
 - <http://robolectric.org/>
 - Without any emulator or device!
- **Mockito** : Can use the “Test Double”
 - <http://code.google.com/p/mockito/>

Image of Unit testing for Dao by using Robolectric

```
@Before  
public void setUp() {  
    Create database for Test;  
    Insert test data;  
}  
  
@Test  
public void findXxx() {  
    Assertions;  
}  
  
@After  
public void tearDown() {  
    Drop Database for Test;  
}
```

5 minutes -> 0.5 seconds
to run each test case.

After TDD



- Can test each component **independently and separately**
- It takes **one day** to implement one activity set
(five times faster than at the start of the project)

1. Conditions and Challenges

2. CI/CD

3. TDD

4. BDD

5. Results, Problems, Possibility and Future

6. Conclusions

Challenges



Avoid feature creep

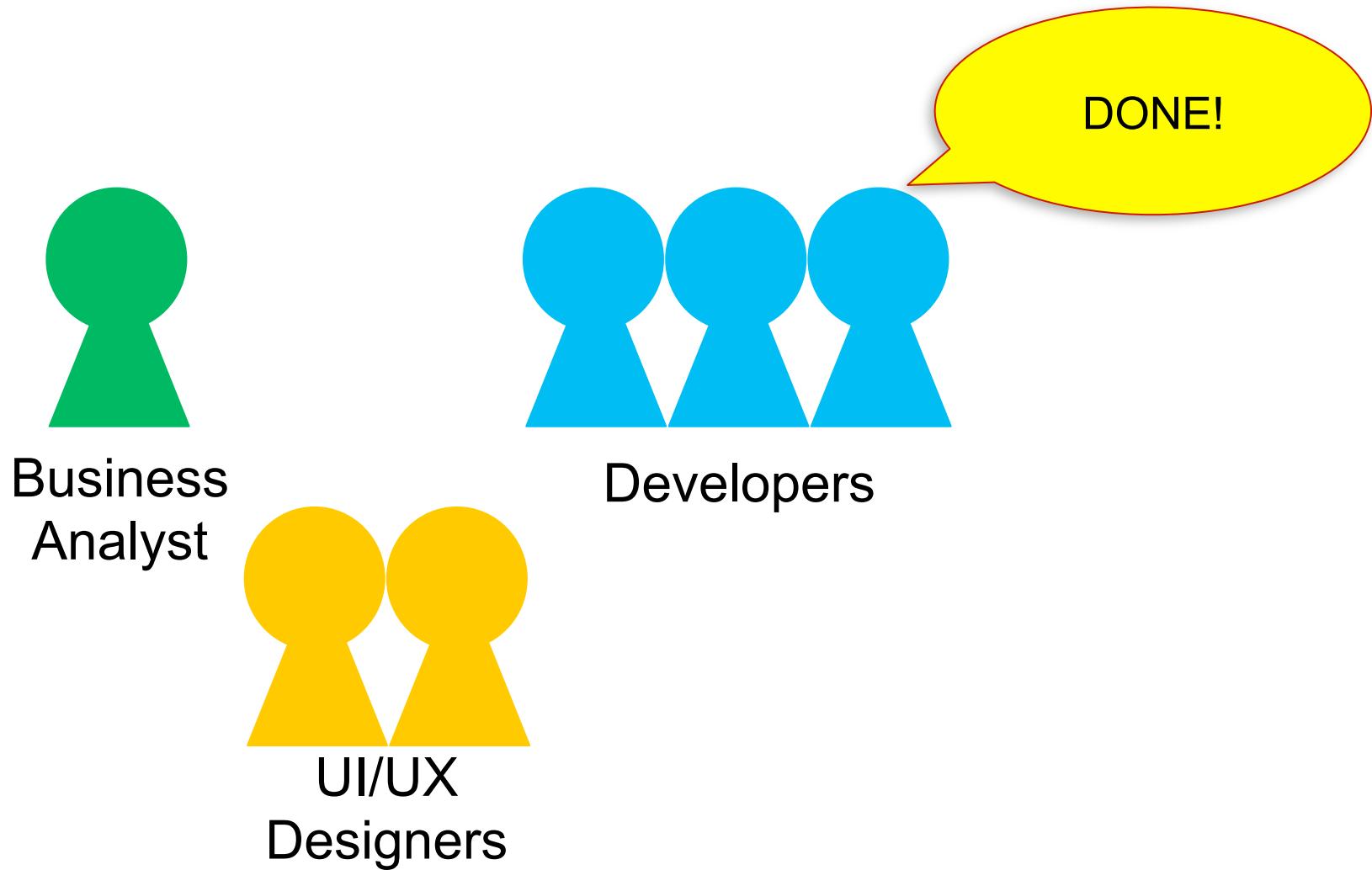


Detect bugs and regressions
on use-cases

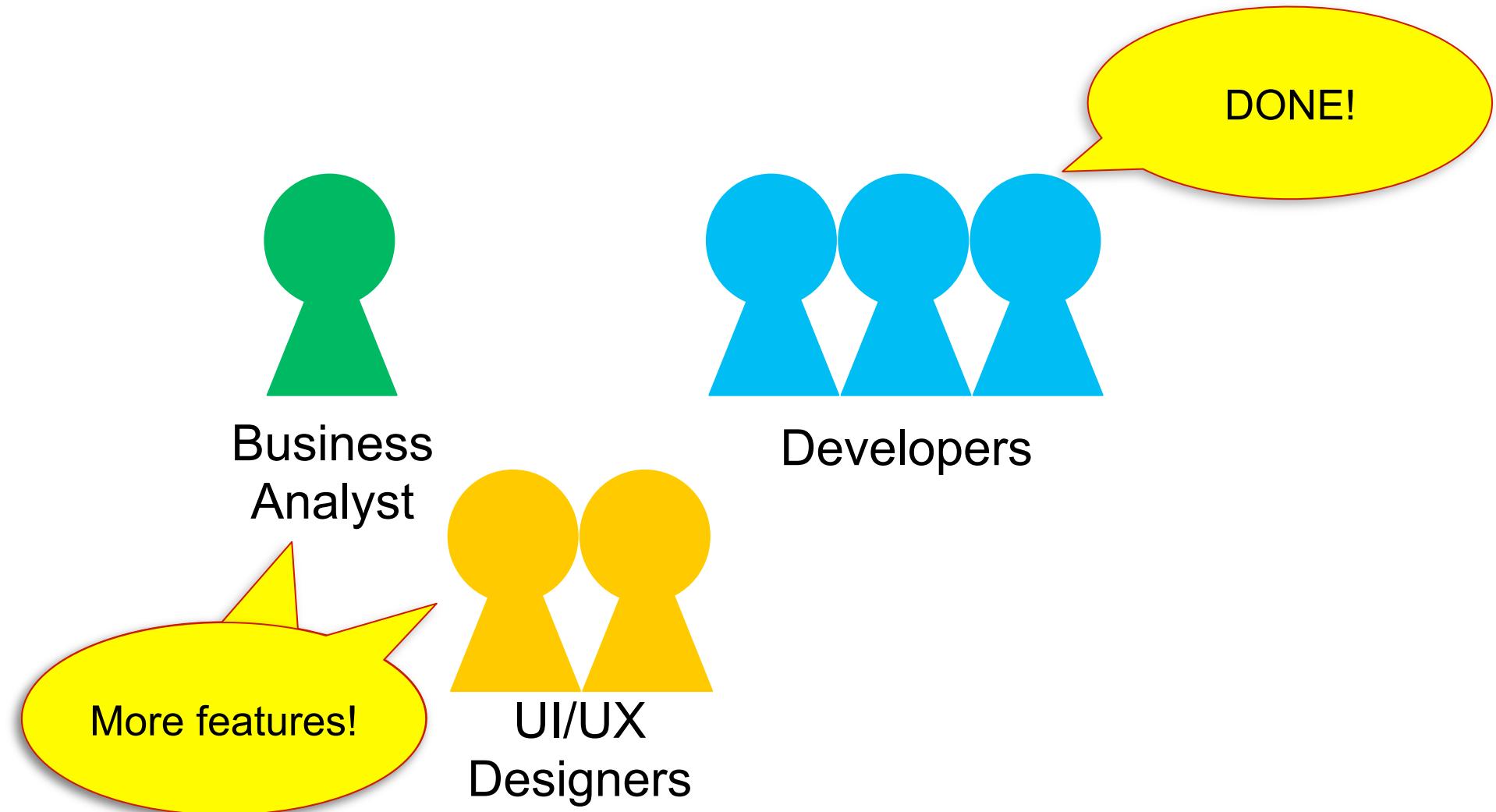


Learn domain knowledge effectively

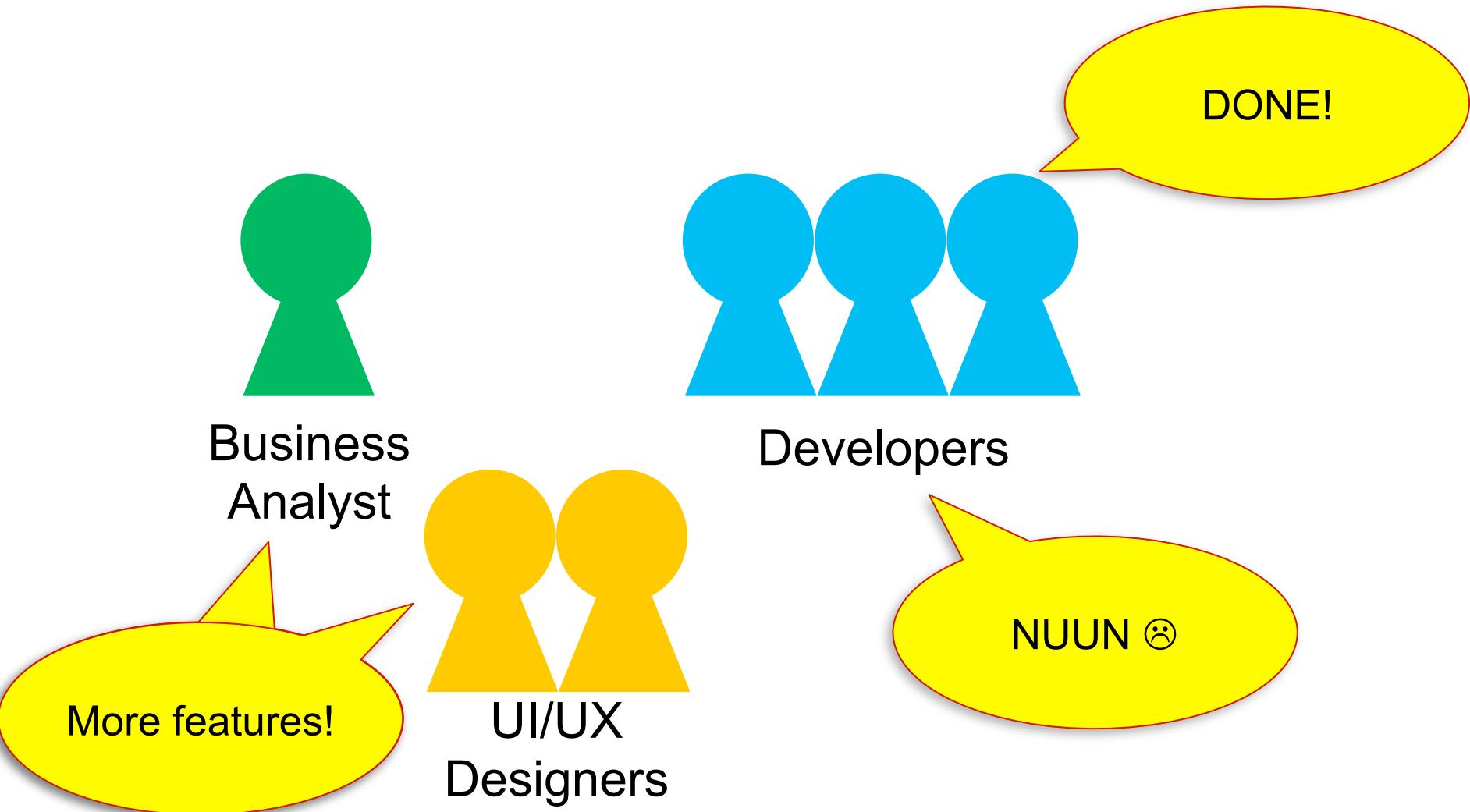
Example of feature creep



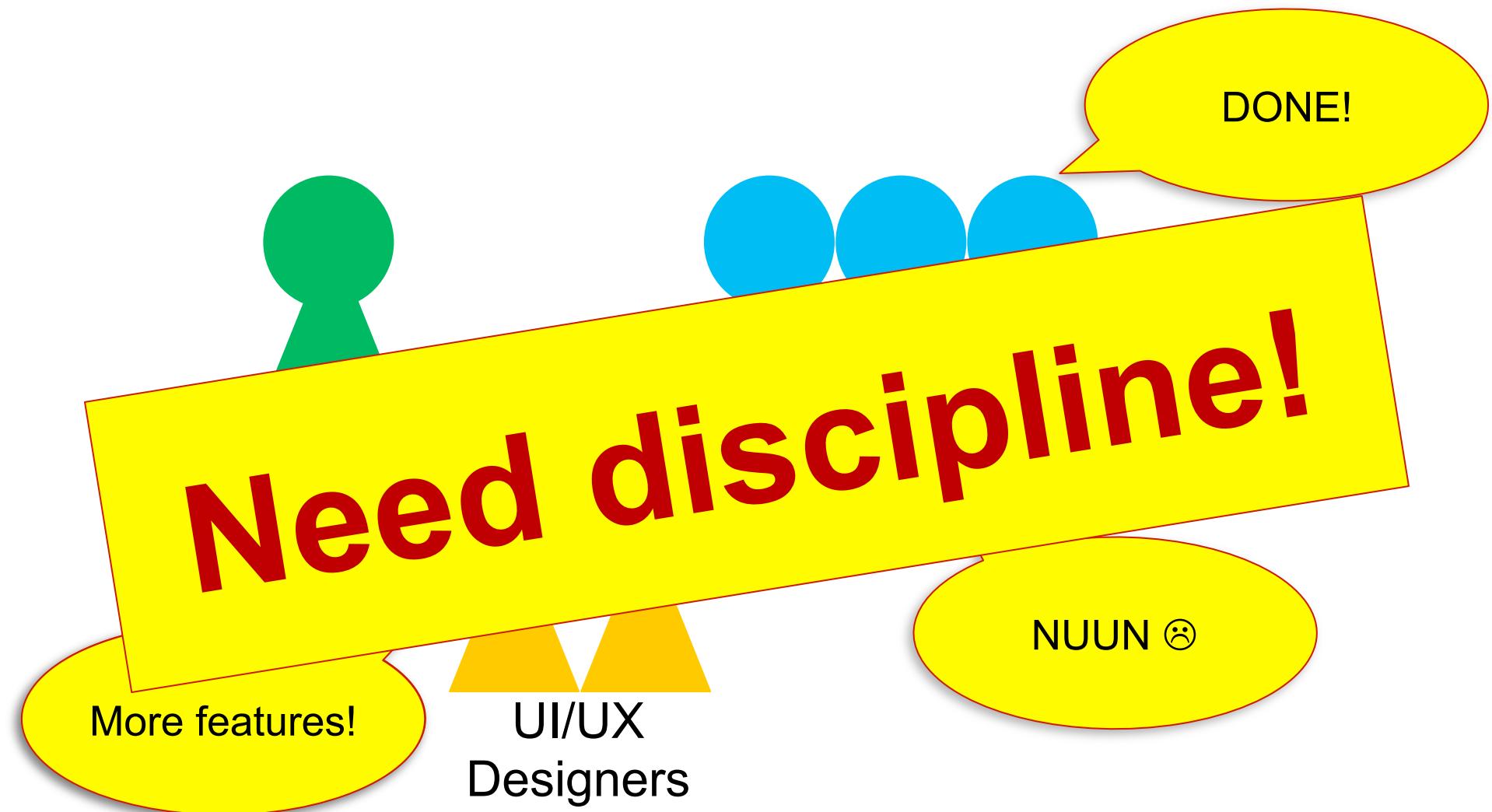
Example of feature creep



Example of feature creep



Example of feature creep



Calabash-android: improve the discipline



Cucumber

- The wrapper of Cucumber for Android
- As an **executable specification**
- As a **communication tool**
 - Specifying collaboratively with business analyst, designers and developers
- By **specification with examples**

Example of BDD test scenario with Calabash-Android

Feature: Input

- Feature : name of all cases
- Scenario : name of each case

Scenario: Input today's data

Given I kick drumroll

These statements are
RUNNABLE!

And drumroll show today

When press next

Then I should see "xxx" screen

When I press keys and calculator should show like this:

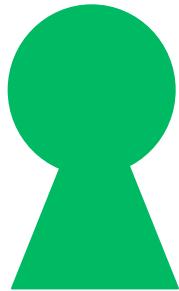
	2	2
	0	20
	0	200
	*	200
	3	3
	=	600

We can write data
with table style like this

Then take photo

...

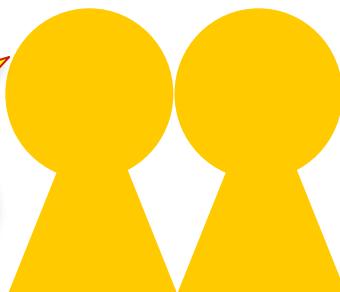
Process of BDD



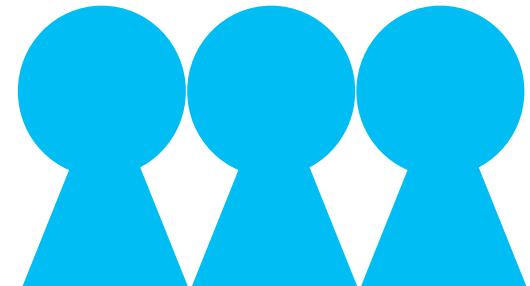
Business
Analyst



We want to...

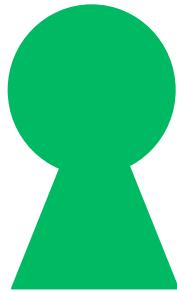


UI/UX
Designers



Developers

Process of BDD



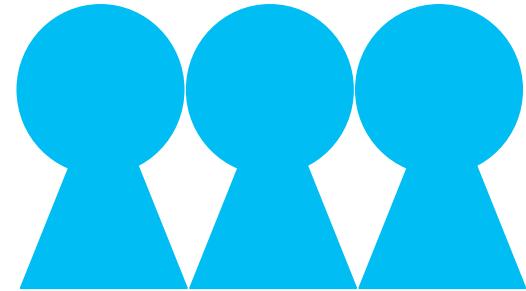
Business
Analyst

GIVEN ...
WHEN ...
THEN ...



UI/UX
Designers

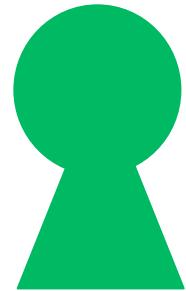
We want to...



Developers

Is that right?

Process of BDD



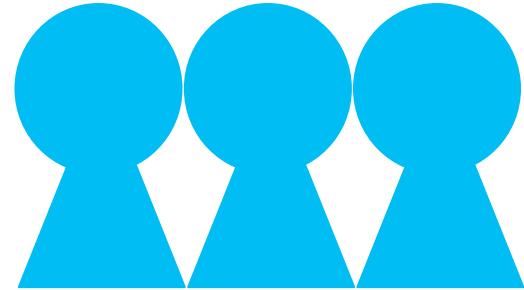
Business
Analyst

GIVEN ...
WHEN ...
THEN ...

OK, go ahead!



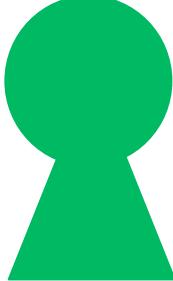
UI/UX
Designers



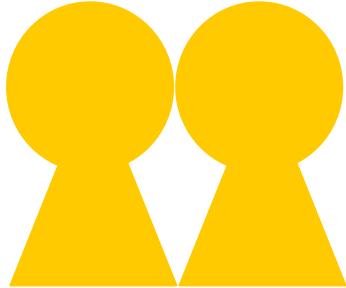
Developers

Is that right?

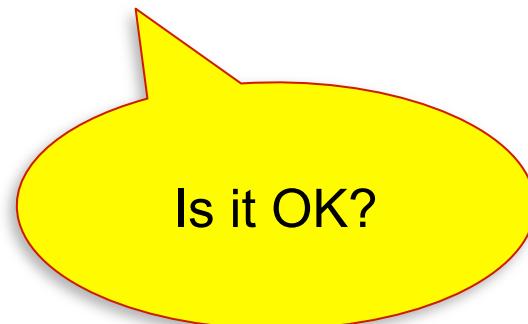
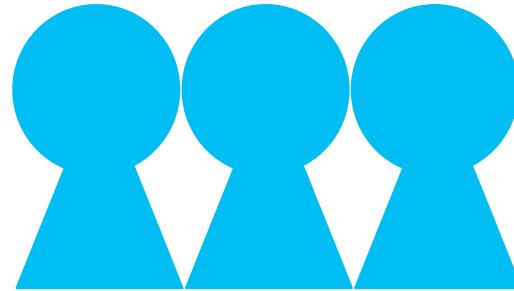
Process of BDD



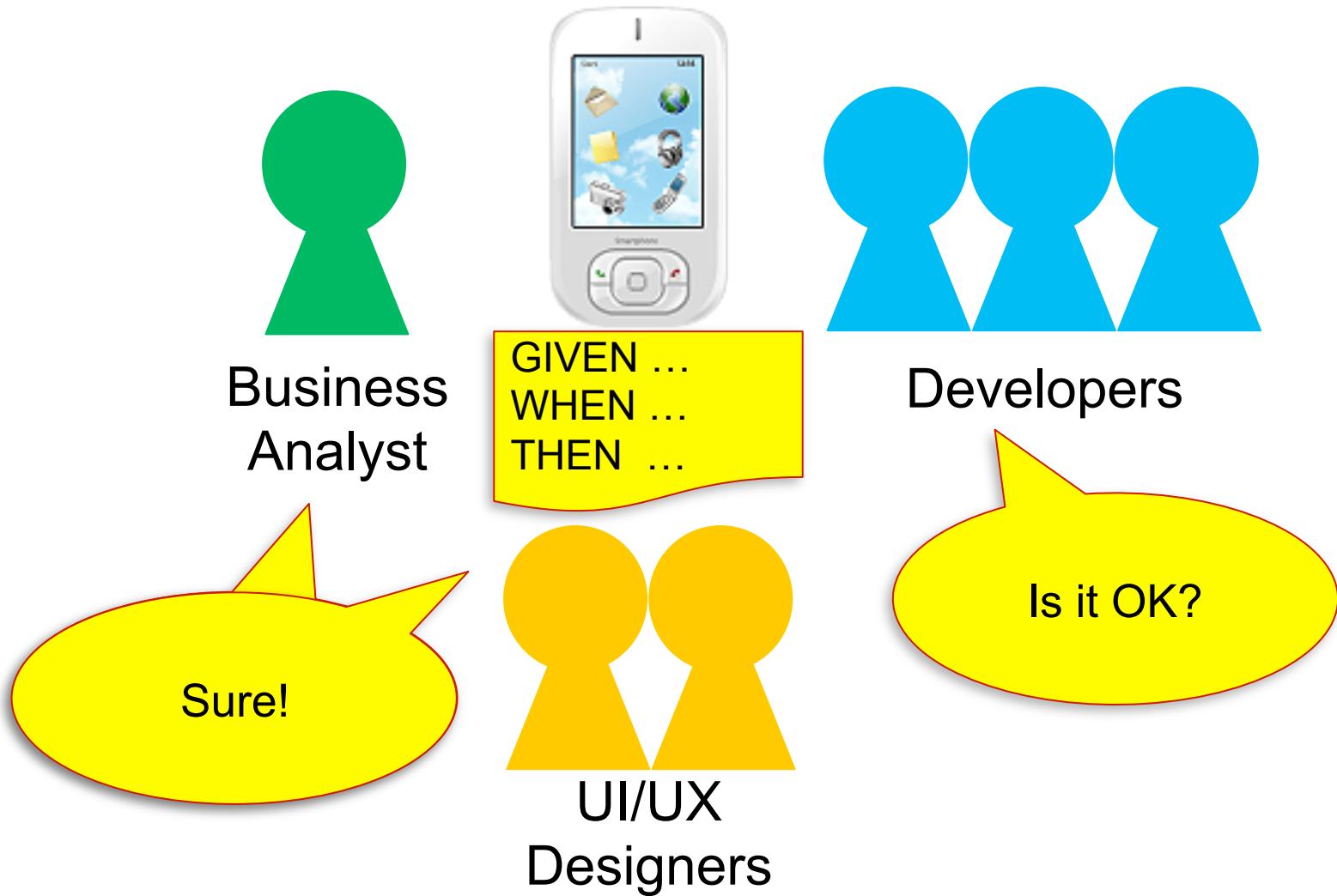
Business
Analyst



UI/UX
Designers



Process of BDD



After BDD

- Bugs : **-67%**
- Change requests : **-70%**
- Regressions : **-60%**

After BDD

- Bugs : -67%

- Changes : +30%

Improved!

1. Conditions and Challenges

2. CI/CD

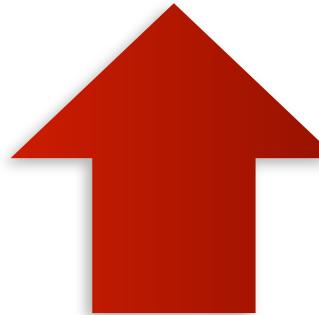
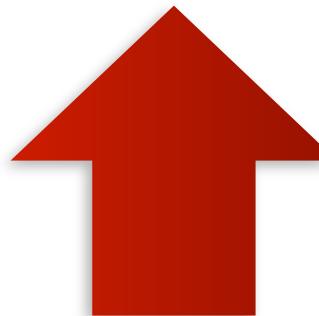
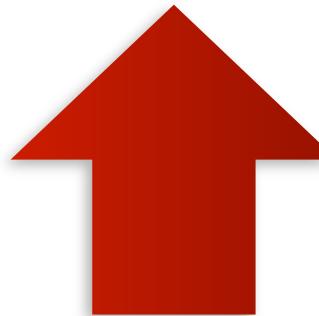
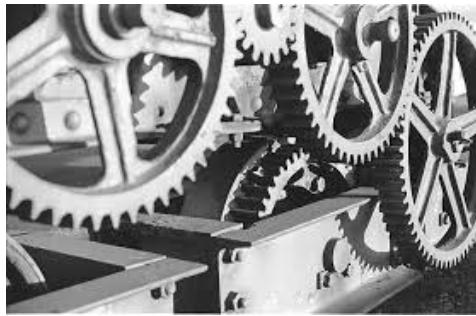
3. TDD

4. BDD

5. Results, Problems, Possibility and Future

6. Conclusions

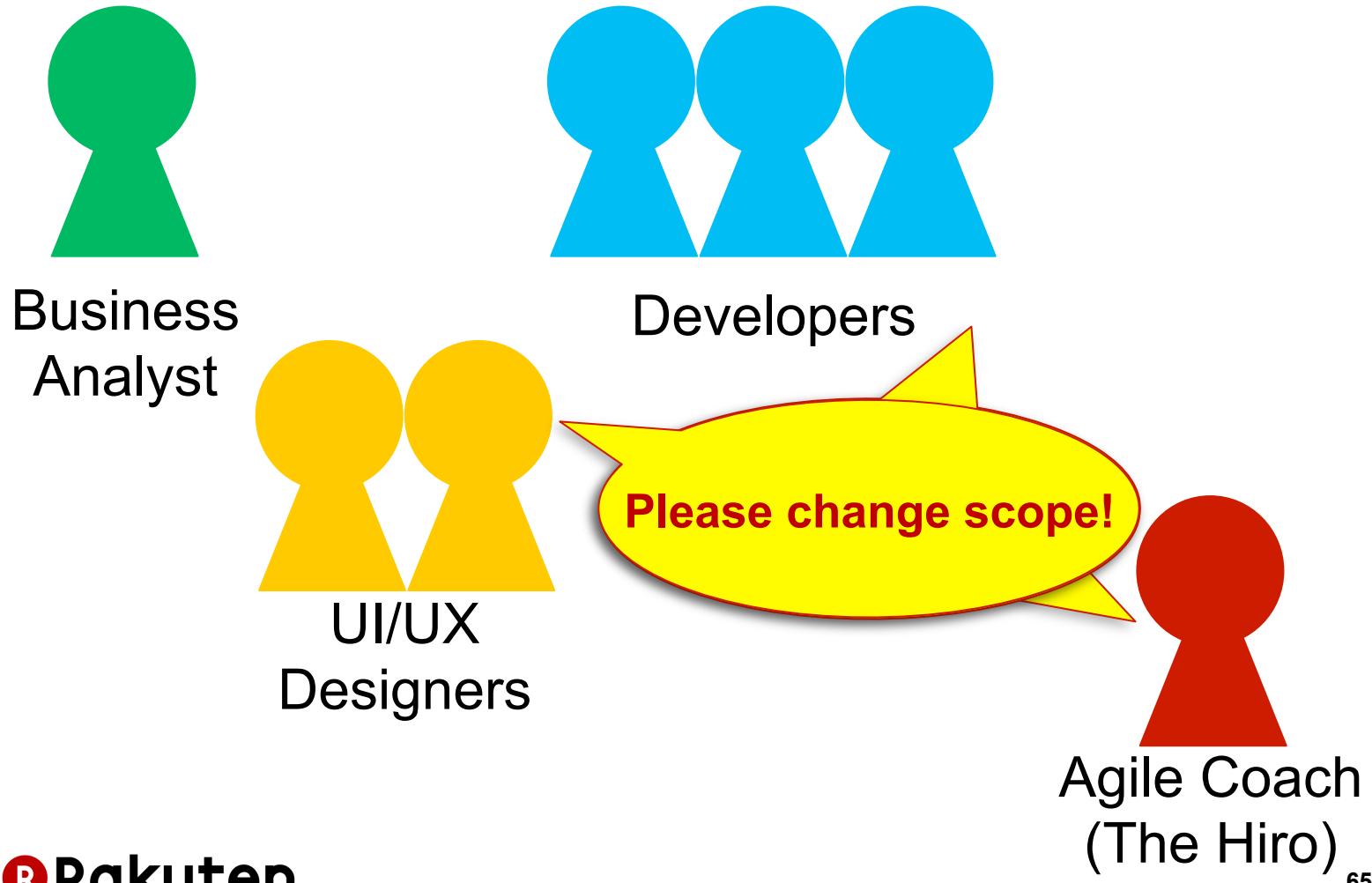
Results



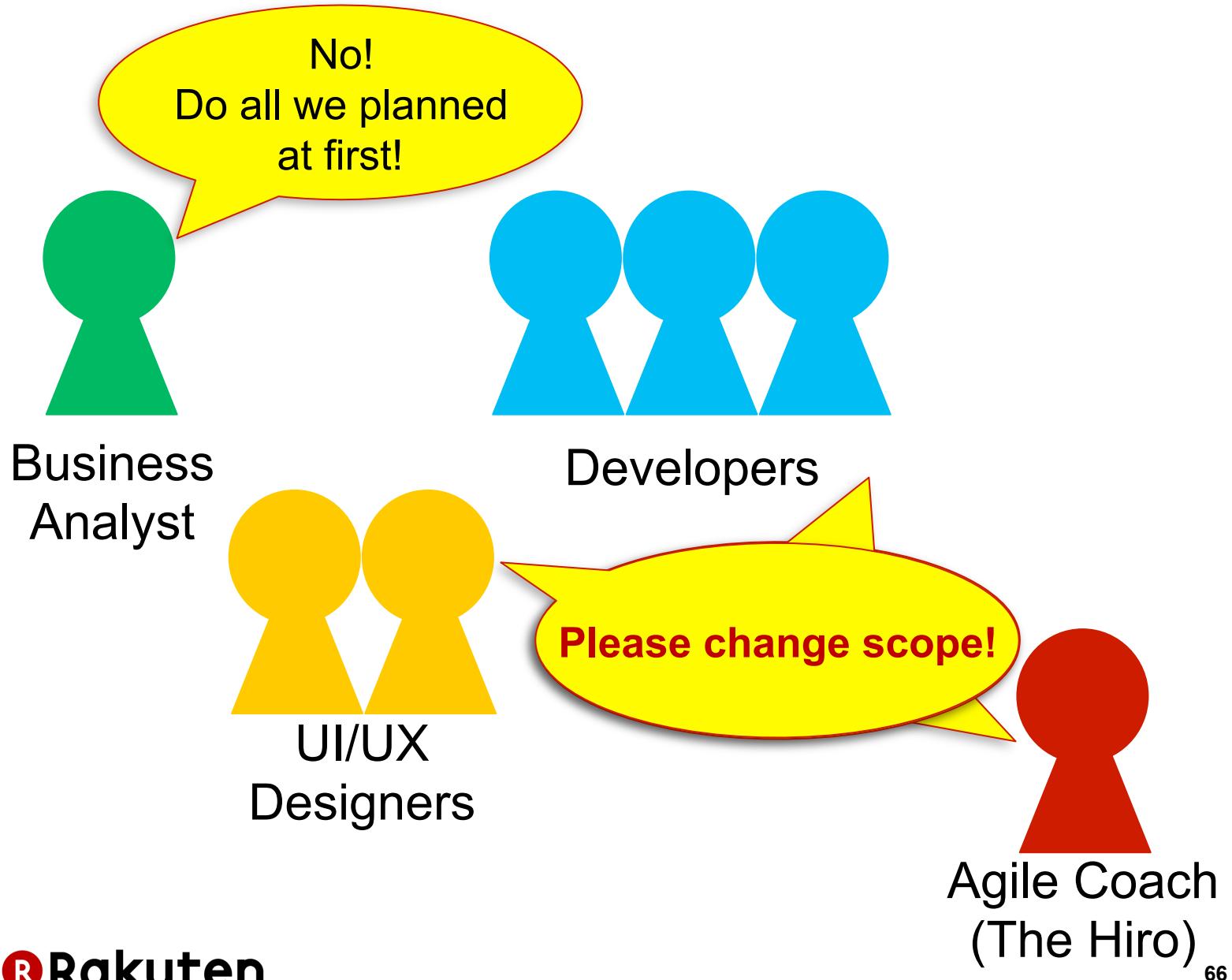
Challenges



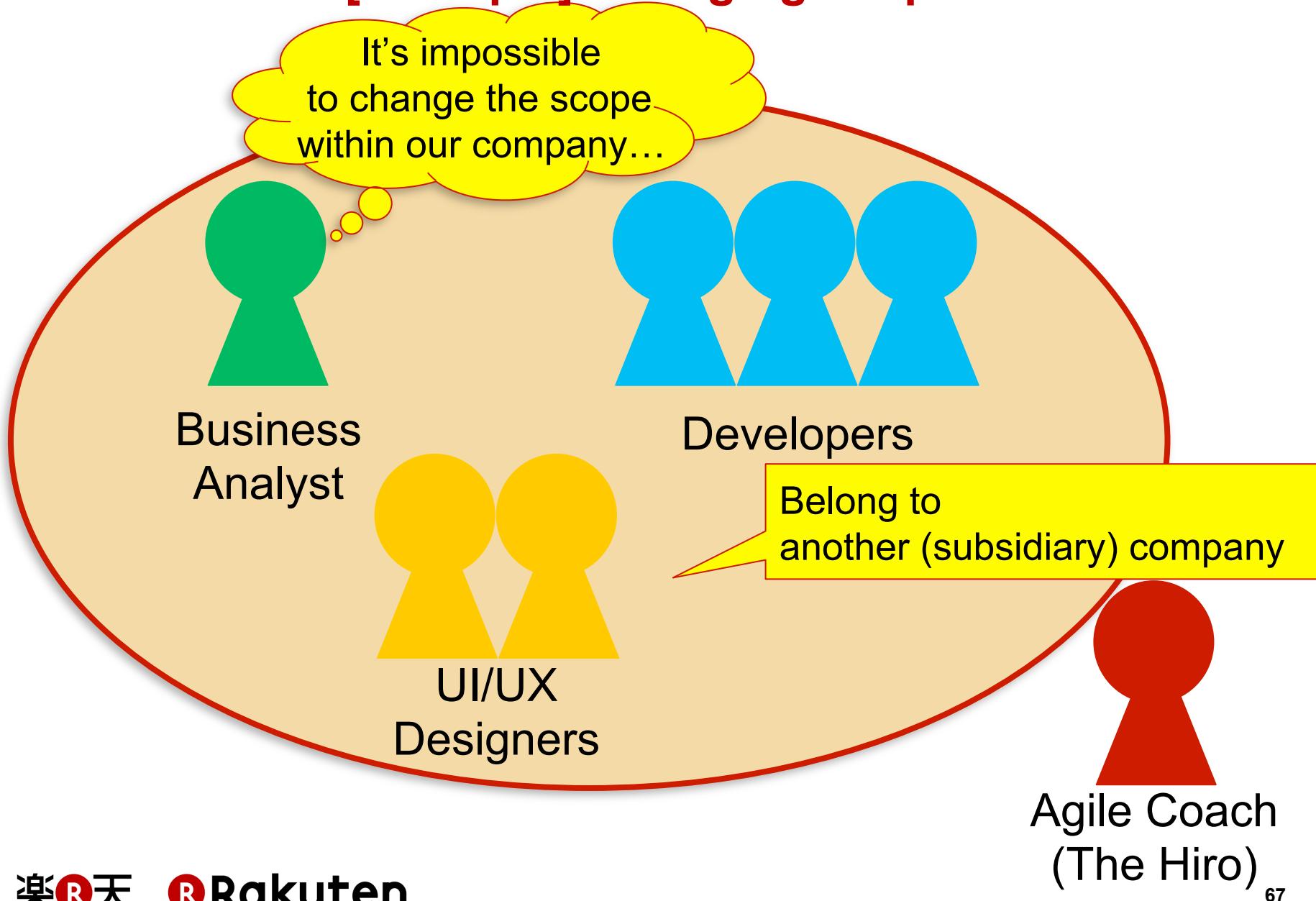
[Example] Changing scope



[Example] Changing scope



[Example] Changing scope



Asked for one executive

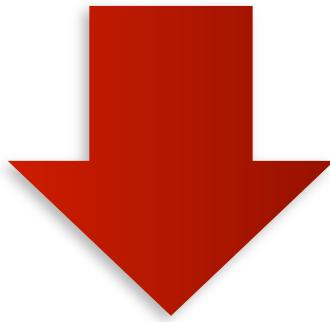


YES, YOU CAN!

We changed scope!

Point

Technical excellence and working software
are **not the only way** to improve projects.

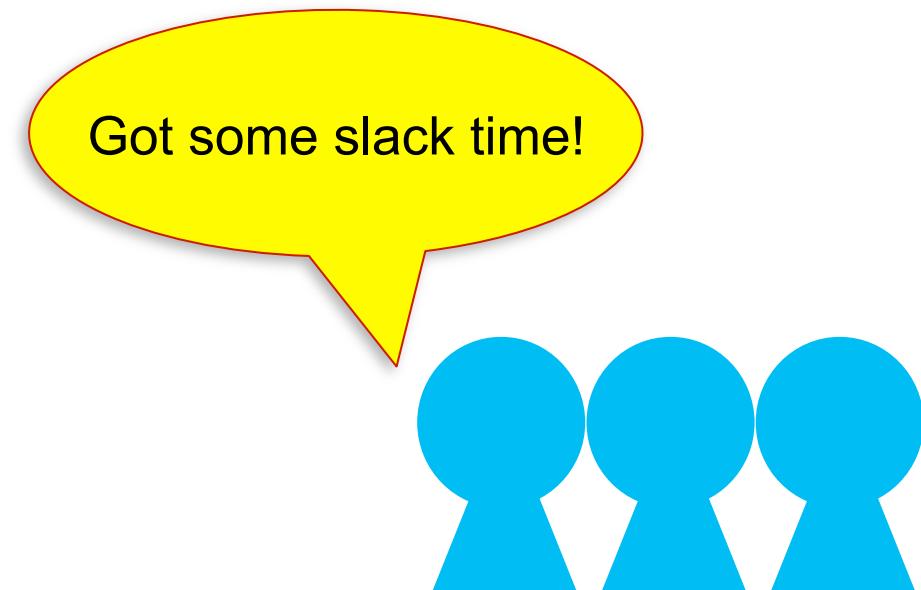


Anything is OK for improving your situation!
(Anything goes/Vale tudo)

Possibility and future



[Example] Growing a collaborative culture

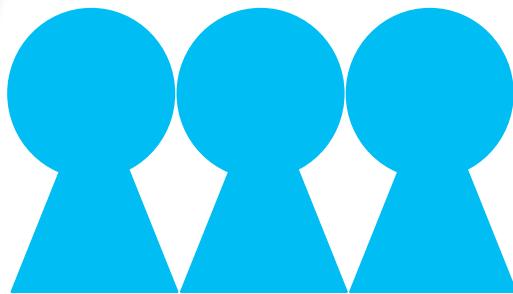


Got some slack time!

Developers

[Example] Growing a collaborative culture

Got some slack time!



Developers

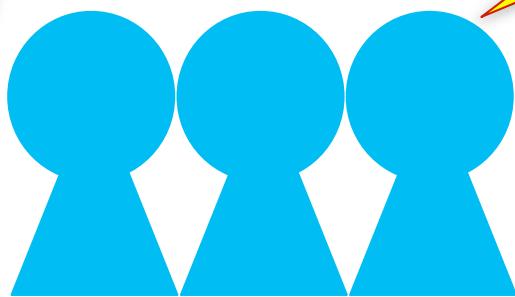
Too slow
emulator...



[Example] Growing a collaborative culture

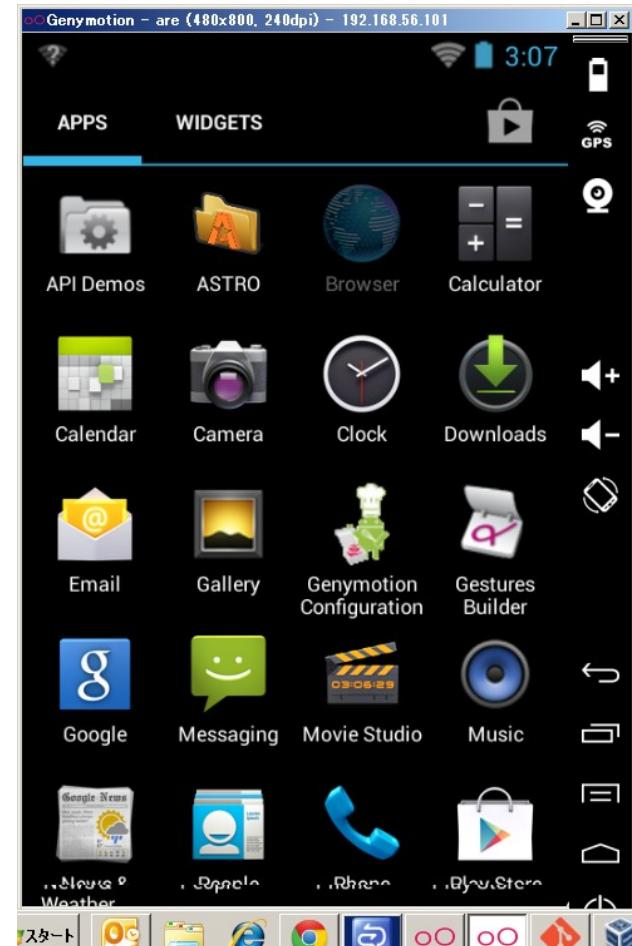
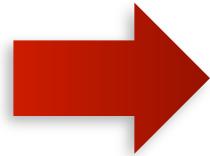
Got some slack time!

How about
Genymotion?



Developers

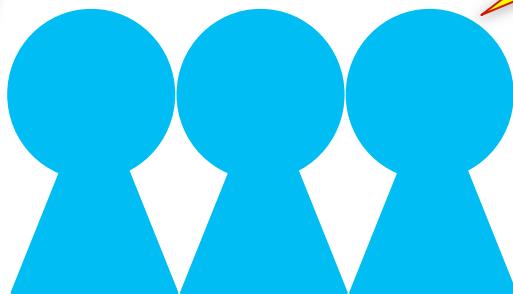
Too slow
emulator...



[Example] Growing a collaborative culture

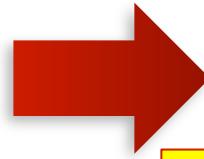
Got some slack time!

How about
Genymotion?



Developers

Too slow
emulator...



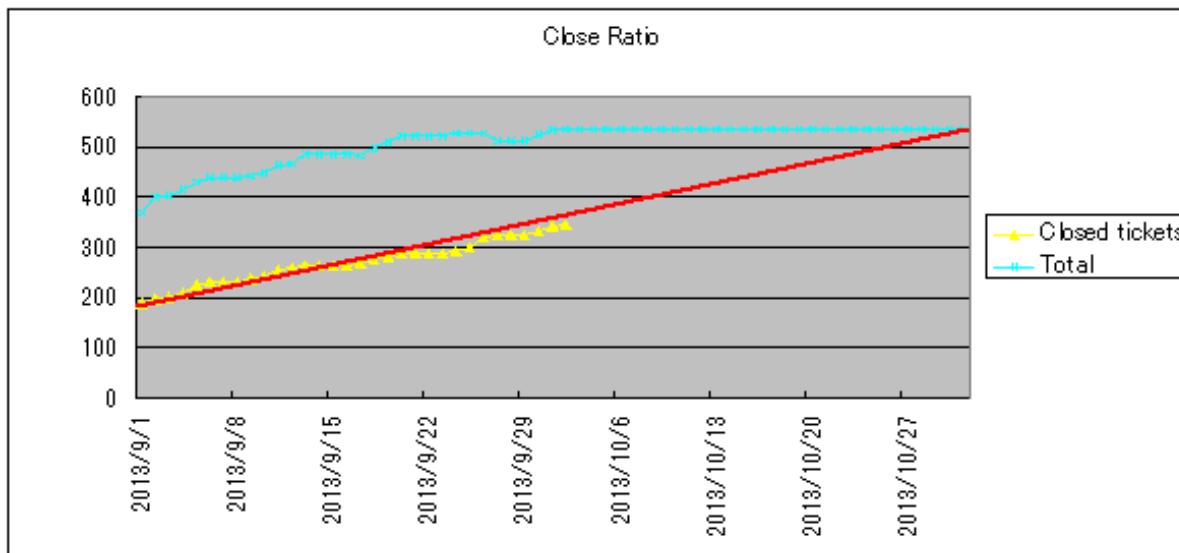
- Over 10 times faster
- Can run via Calabash-Android



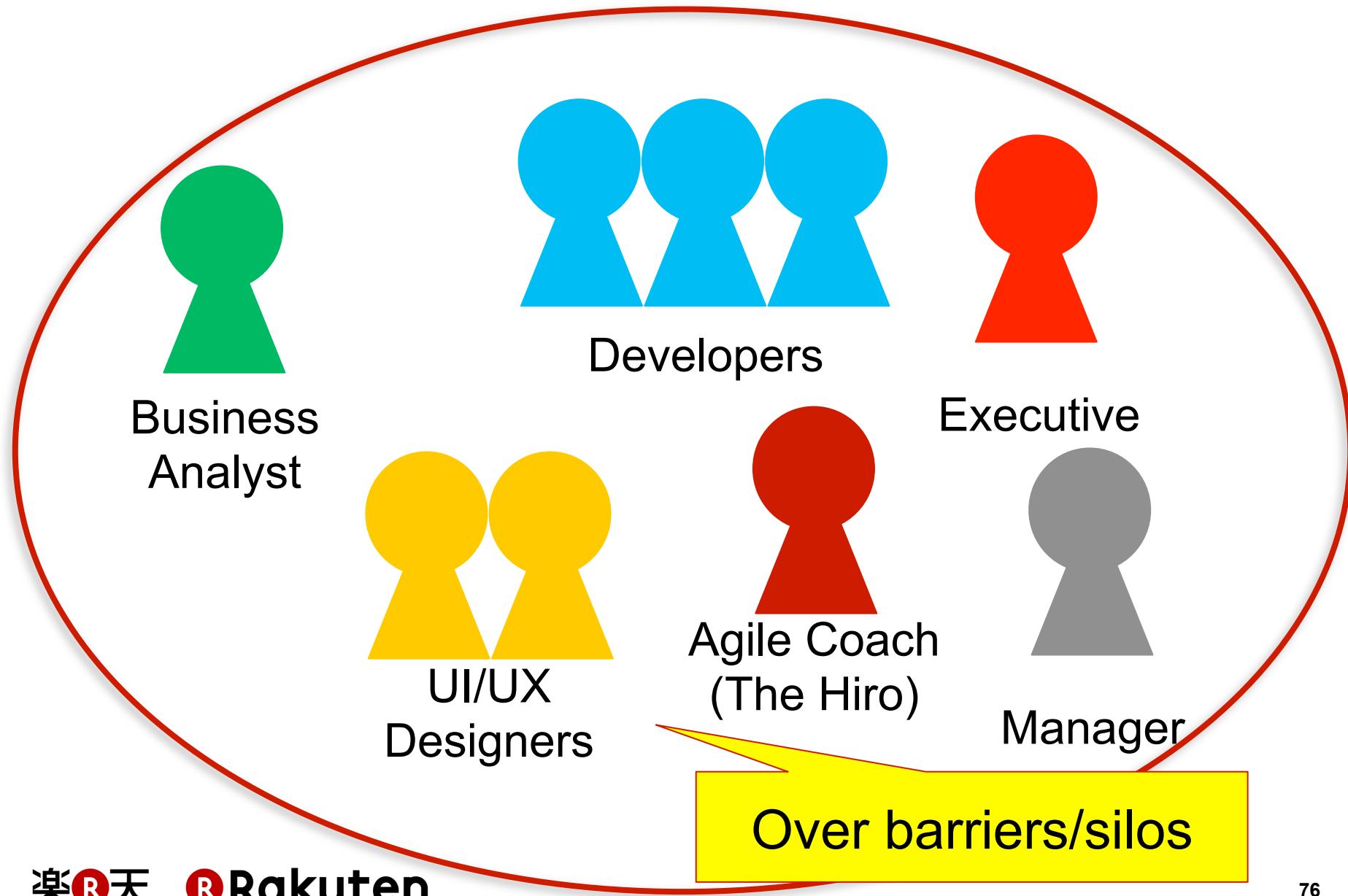
Can enhance “TDD” by numerical measurement

[e.g.]

- Change requests : 3 times/week
- Regression testing : **3 minutes**/change
- Install applications : **2 minutes**/change



Use “TDD” as a measure for total optimization



Don't lose the whole picture!



1. Conditions and Challenges

2. CI/CD

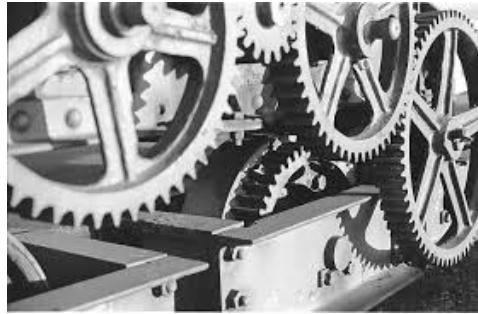
3. TDD

4. BDD

5. Results, Problems, Possibility and Future

6. Conclusions

Three purposes



Efficiency



Learning



Collaboration

Three approaches

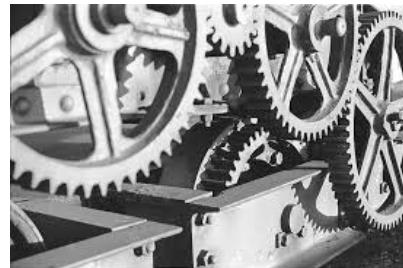
CI/CD



TDD



BDD



Three approaches by

CI/CD



TestFlight
iOS Beta Testing On The Fly

TDD



BDD

Cucumber

We found this practice

- through the project
- with passionate members
- with a lot of trial and error

Experience from Gemba 現場主義

Find your **answer**
by **yourself**
through your **experience**

Find your treasure!

