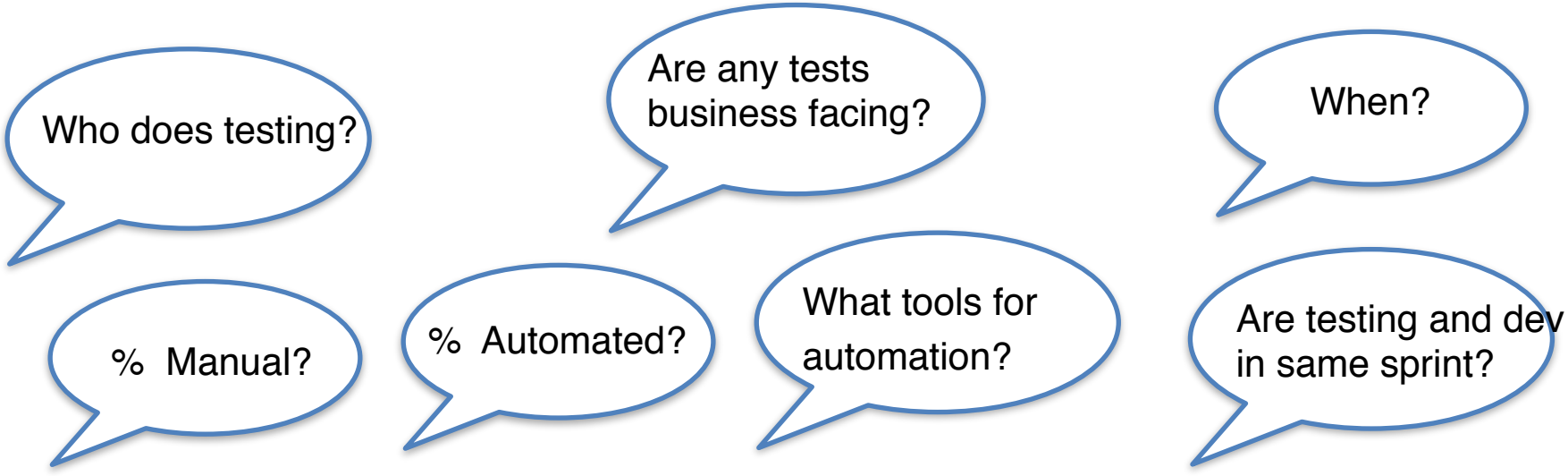
Organizational Dysfunctions on Testing - Agile Remedies

Introductions:

Bill Allen - agile Innovation Labs
Bill@agileInnov.com

Kieran Murphy - ThoughtWorks
KiMurphy@ThoughtWorks.com

What does testing look like in your organization?



What does testing look like in your organization?

Typical testing challenges we see

Developers:

1. "These is no time to write test cases"
2. "There's no project time allocated to perform unit testing"
3. "It is difficult keeping unit tests up to date"
4. "Deadlines (The DATE) prohibit good testing"
5. "Challenging to know that code is thoroughly tested"
6. "Testers are unfamiliar with coding. They only do manual. They have no visibility into the code operation."
7. "Challenges with early prototyping"
8. "Legacy applications are fragile and difficult to change"
9. "Getting viable test data is difficult"
10. "Business test cases vs code test cases (translation: Testing Business Facing vs Code Facing)"
11. "We'll write unit tests, but only after we write the code and only if we have time"

Testers:

"We're told that Automating Acceptance Testing could be beneficial, but we don't code and we don't work closely with the developers."

Senior Management:

"Yes, we see the need for change but planning for it takes too much time."

Correctness-Driven Development

Not Testing practices, but instead *whole team* analysis

QA
Test Engineer
BA
PM
PO
Developer

Test-Driven Development (TDD)

*from Test-Driven Development by Example - Kent Beck

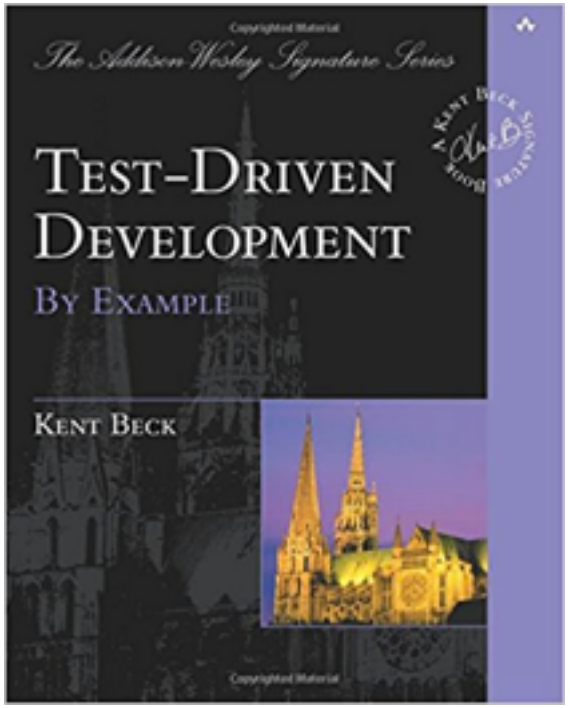
1. Red - Write a little test that doesn't work, and perhaps doesn't even compile at first.
2. Green - Make the test work quickly, committing whatever sins necessary in process.
3. Refactor - Eliminate all of the duplication created in merely getting the test to work.

Red/green/refactor – the TDD mantra. (preface, p.x)

In other chapter the same order is described as "general TDD cycle":

1. Write a test…
2. Make it run…
3. Make it right…

The goal is clean code that works (thanks to Ron Jeffries for this concise summary)… First we'll solve the "that works" part of the problem. Then we'll solve the "clean code" part. (p.11)



Behavior Driven Development

Dan North and Chris Matts, 2003 - 2004, published 2006

> "*While using and teaching agile practices like test-driven development...Programmers wanted to know where to start, what to test and what not to test, how much to test in one go, what to call their tests, and how to understand why a test fails."**

Tooling for test driven development at the time was primitive, JUnit was a leaky framework and imposed inelegant language on developers practicing TDD

Dan North began JBehave in late 2003 intending it to be a replacement for JUnit, removing all reference to "test" and using "behavior" instead

> *"I found the shift from thinking in tests to thinking in behaviour so profound that I started to refer to TDD as BDD, or behaviour- driven development."**

In 2004, after implementing JBehave, Dan North and Chris Matts (BA who collaborated on JBehave), realized that their behavior driven language was an analysis language:

> *"If we could develop a consistent vocabulary for analysts, testers, developers, and the business, then we would be well on the way to eliminating some of the ambiguity and miscommunication that occur when technical people talk to business people."**

Drives development from "outside in", in "pull-based" fashion, from business-facing tests

> *"BDD is a second-generation, outside-in, pull-based, multiple-stakeholder, multiple-scale, high-automation, agile methodology"***

* Dan North, "Introducing BDD": https://dannorth.net/introducing-bdd/

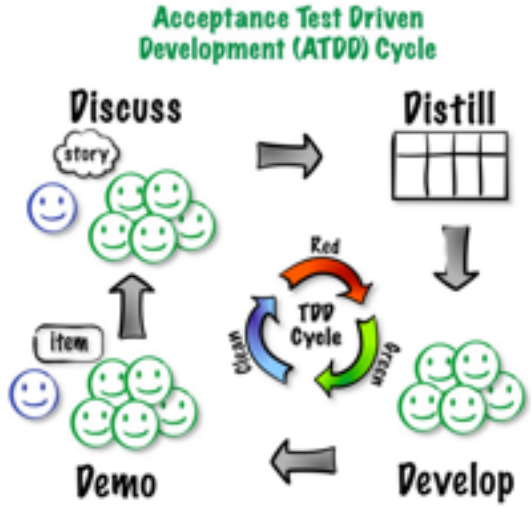
** Dan North, "Agile Testing, Specifications and BDD Exchange", 2009

Acceptance Test Driven Development (ATDD) involves team members with different perspectives (customer, development, testing) collaborating to write acceptance tests in advance of implementing the corresponding functionality.

customer perspectives …what problem are we trying to solve?

development perspectives …how might we solve this problem?

testing perspectives …what about…?



(ATDD cycle model developed by James Shore with changes suggested by Grigori Melnick, Brian Marick, and Elisabeth Hendrickson.)


David Hussman
To: William Allen
RE: question about BDD/ATDD

It sounds like I miscommunicated. Instead of them being totally different, I was trying to say that BDD was TDD people who finally discovered a lamer version of ATDD. In the end they are all just more noisy label to spin on and around. I tend to think of these spaces as:

- Developer testing (by the geeks for the design) – they tell the story of the code
- Story testing (by the community for the product) – they tell the story of the product

From: Bill Allen
Sent: Tuesday, May 08, 2012 12:16 AM
To: david hussman
Subject: Re: question about BDD/ATDD

Bill

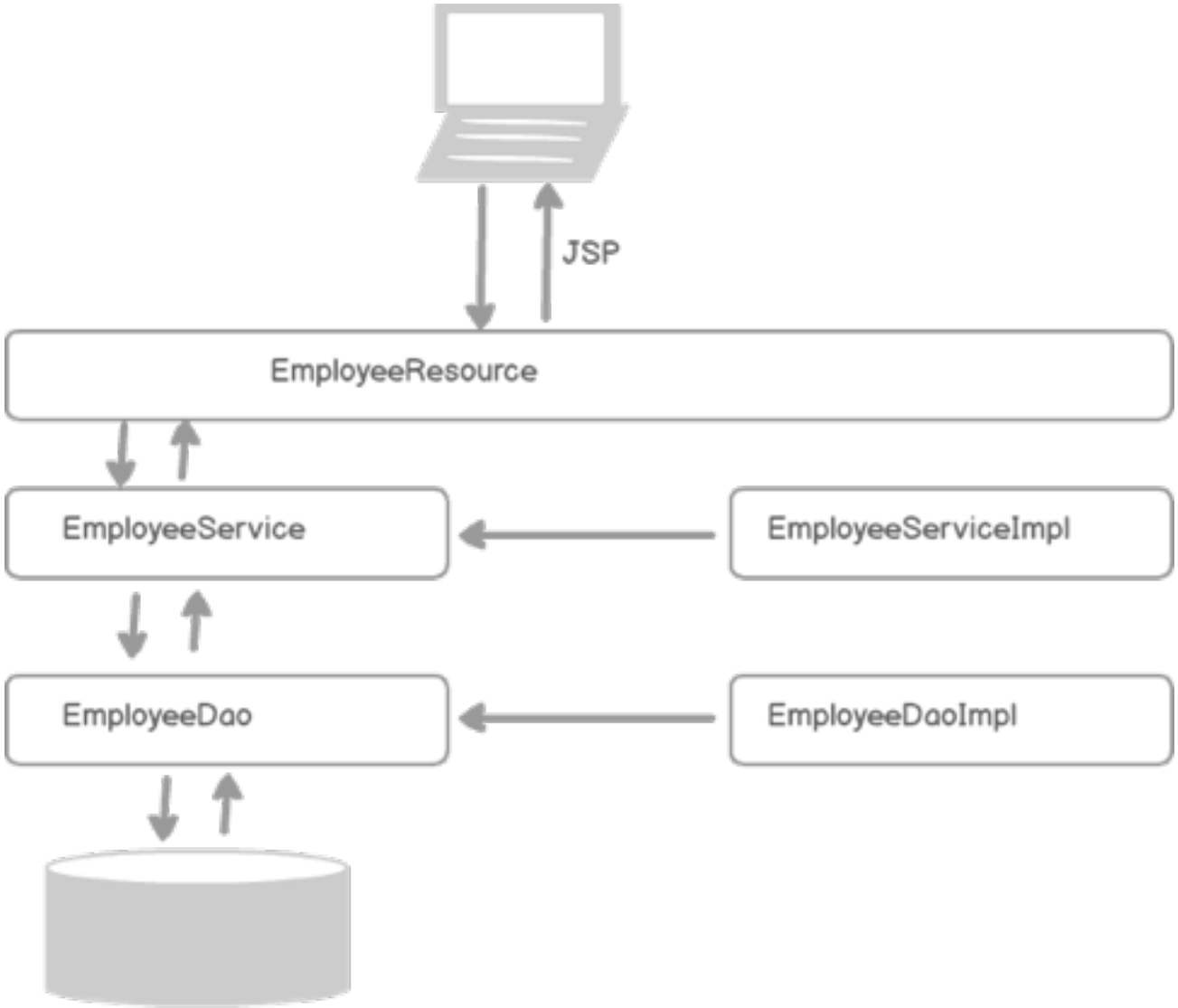
"Imagination is more important than knowledge" -- Albert Einstein

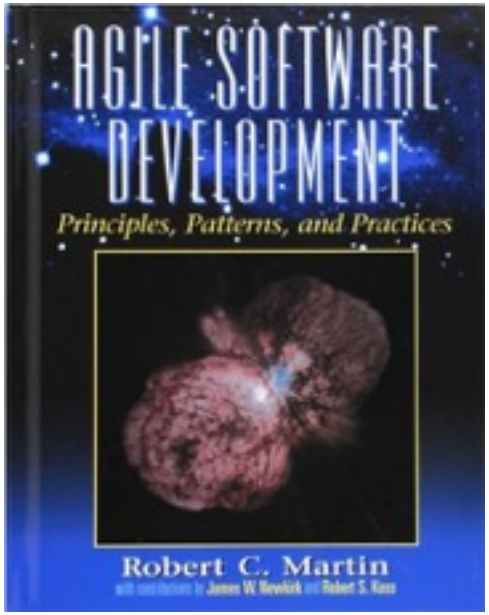
"BDD is TDD people who finally discovered a lamer version of ATDD. In the end they are all just more noisy label to spin on and around. I tend to think of these spaces as:

- *Developer testing (by the geeks for the design) – they tell the story of the code*
- *Story testing (by the community for the product) – they tell the story of the product"*

* David Hussman, DevJam, Minneapolis,MN

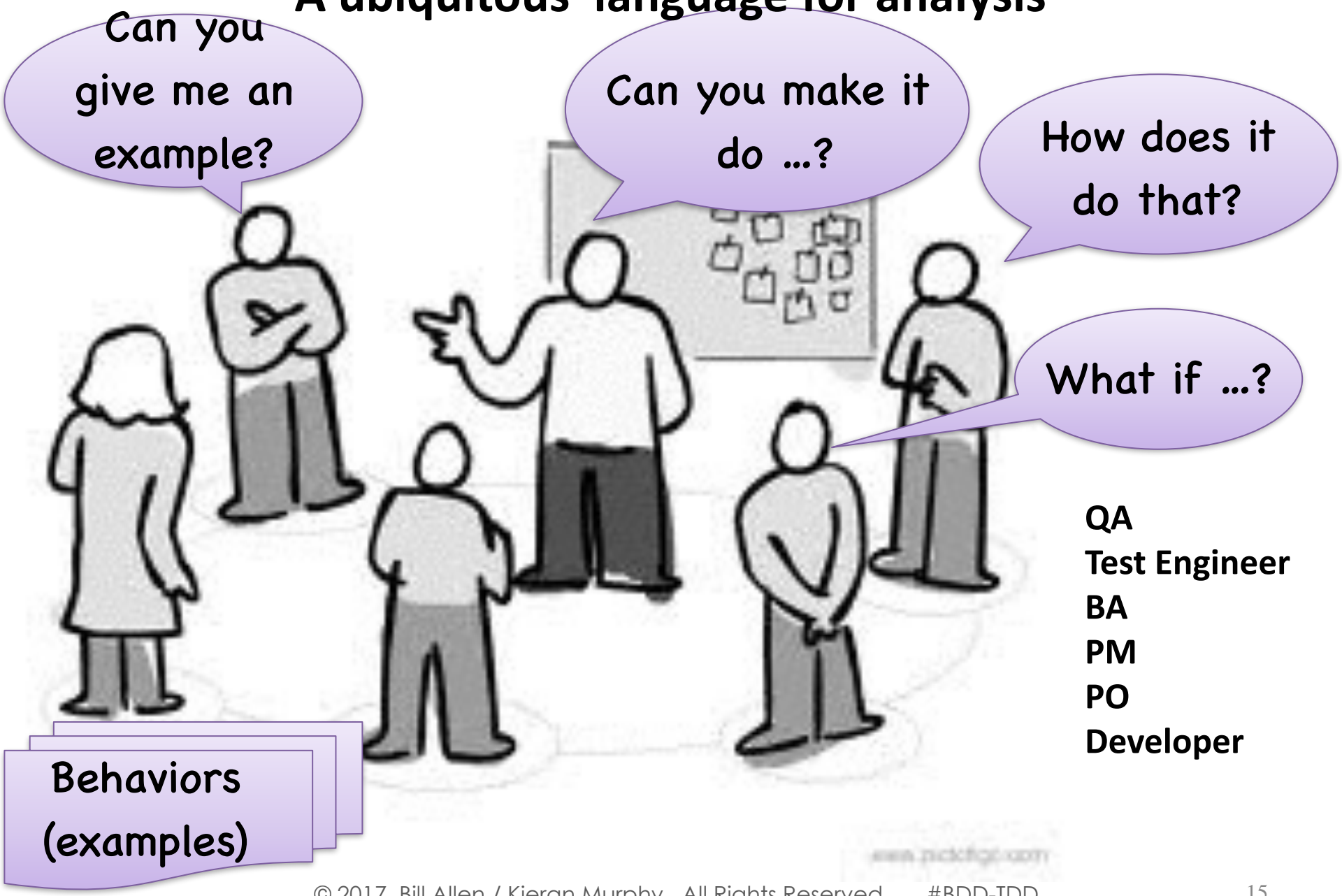
Project Example: Payroll System





BDD / TDD in Action

A ubiquitous language for analysis



QA
Test Engineer
BA
PM
PO
Developer

Gherkin - a formal language for specifying behaviors

Feature: Some terse yet descriptive text of what is desired

Textual description of the business value of this feature
Business rules that govern the scope of the feature
Any additional information that will make the feature easier to understand

Scenario: Some determinable business situation

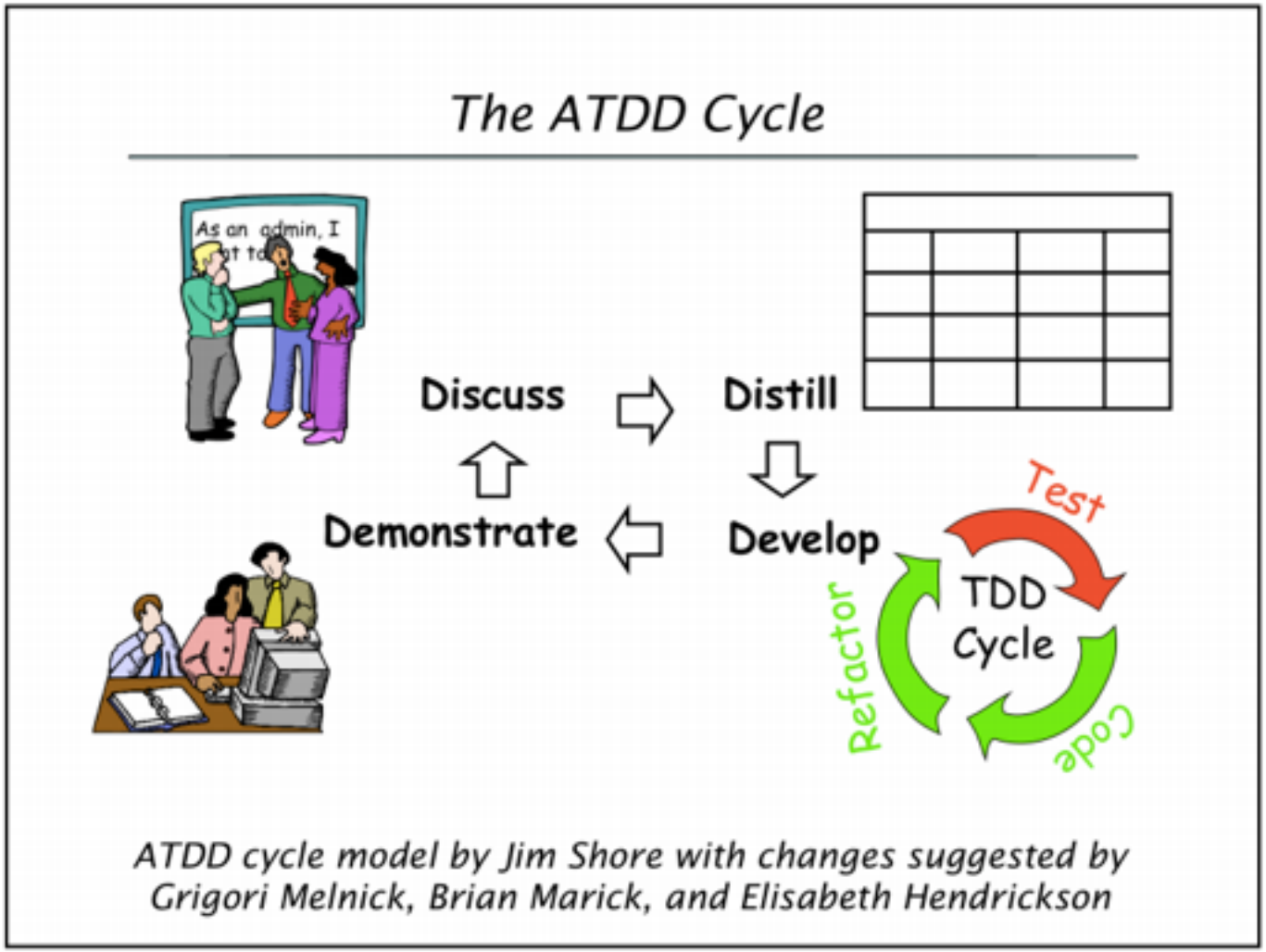
Given some precondition
When some action by the actor
Then some testable outcome is achieved

Scenario: Some other determinable business situation

Given some precondition
When some action by the actor
Then some testable outcome is achieved

...

What does whole team analysis look like?



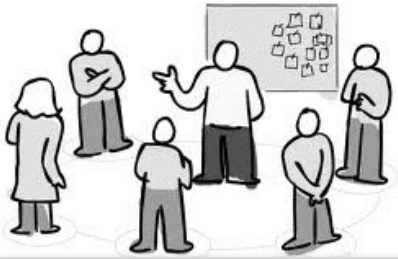
As a **team**, we will develop a single feature - Add a hourly employee to the payroll system.

Discuss

Form a team: Product Owner, Developer, Tester; discuss mPayroll system.

Tell a story about add a new employee to mPayroll:

1. ***Why does the feature provide benefit?***
2. ***How will you know the feature works?***
3. ***Ask PO for examples of how this feature should behave.***



```
Feature: Add a new employee
  •Discuss behaviors
```

"How can we expose the existing payroll system to the network, for access by apps, devices, web browsers?"

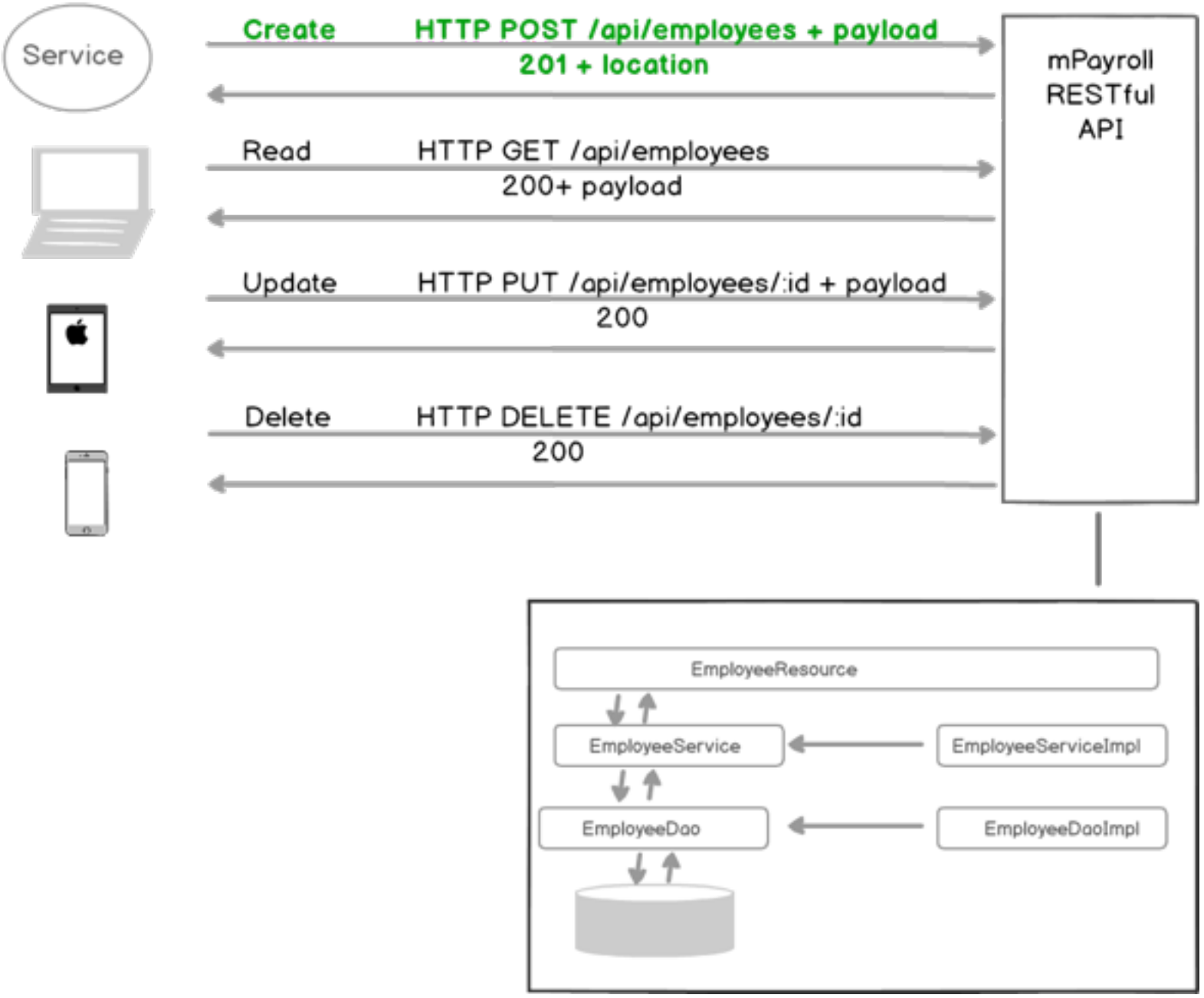
"How do I perform C R U D actions?"

"What does an employee records contain?"

"What happens when a new employee is created?"

"What happens if that employee's record already exists?"

"How do I verify that a new employee was created?"

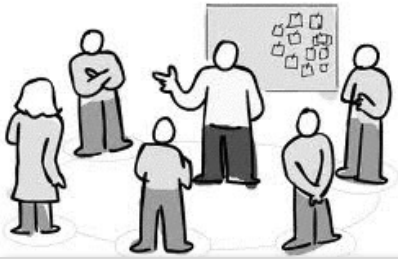


Discuss → Distill

Form a team: Product Owner, Developer, Tester; discuss mPayroll system.

Tell a story about add a new employee to mPayroll:

1. *Why does the feature provide benefit?*
2. *How will you know the feature works?*
3. *Ask PO for examples of how this feature should behave.*



```
Feature: Admin can add a new employee

  Notes: Use HTTP POST to create an
  employee record?
```

```
Feature: Add a new employee

    Notes: Use HTTP POST to create an
    employee record?

  Scenario: Add a new hourly employee

  Given The employee resource is available

  And A new hourly employee record

  When I submit the employee record

  Then A new hourly employee is created
```

(Behavior Test)

BDD Example

You can implement missing steps with the snippets below:

```
@Given("^A new hourly employee record$")
public void a_new_hourly_employee_record() throws Throwable {
   // Write code here that turns the phrase above into concrete actions
   throw new PendingException();
}

@When("^I submit the employee record$")
public void i_submit_the_employee_record() throws Throwable {
   // Write code here that turns the phrase above into concrete actions
   throw new PendingException();
}

@Then("^A new hourly employee is created$")
public void a_new_hourly_employee_is_created() throws Throwable {
   // Write code here that turns the phrase above into concrete actions
   throw new PendingException();
}

@feature
Feature: Add new employees
```

Develop

Run mPayroll

```
[

     {
          firstName: "Bill",
          lastName: "Allen",
          type: "Commissioned",
          rate: 0.10000000149011612
     },

     {
          firstName: "Stanley",
          lastName: "Heron",
          type: "Hourly",
          rate: 10
     },

     {
          firstName: "Kieran",
          lastName: "Murphy",
          type: "Hourly",
          rate: 10
     }
]
```

Demo

Pause … Applause (optional)

Top 5

BILL'S - TOP FIVE REASONS TO USE BDD

1. Drives collaboration and communication between business and developers.
2. Provides a common language for the whole-team to understand what is being developed.
3. Allows Business to have quantitative insight on the progress of development.
4. Provides business the answer to the question, "What can the software do?"
5. Allows behaviors to be written in a natural Business orientated language -- the Given/When/Then format, meaning some can be written by the business.

KIERAN'S - TOP FIVE REASONS TO USE TDD

1. Improved technical architecture - SOLID Principles come for free
2. Results in the least amount of code possible to get the job done
3. Software built from a process of constant incremental change is easy to change later
4. Fosters collaboration and transparency among developers, QA, immediate business stakeholders and management
5. Very satisfying practice

What would BDD / TDD look like for you?

BDD / TDD in your organization

Will it solve your challenges? What can you control:

Who performs testing?
When?
Are testing and dev in same sprint?
What % are manual?
What % are automated?
What tools are used for automation?
What % are business facing?

How would BDD/TDD resolve these testing challenges

Developers:

1. "These is no time to write test cases"
2. "There's no project time allocated to perform unit testing"
3. "It is difficult keeping unit tests up to date"
4. "Deadlines (The DATE) prohibit good testing"
5. "Challenging to know that code is thoroughly tested"
6. "Testers are unfamiliar with coding. They only do manual. They have no visibility into the code operation."
7. "Challenges with early prototyping"
8. "Legacy applications are fragile and difficult to change"
9. "Getting viable test data is difficult"
10. "Business test cases vs code test cases (translation: Testing Business Facing vs Code Facing)"
11. "We'll write unit tests, but only after we write the code and only if we have time"

Testers:

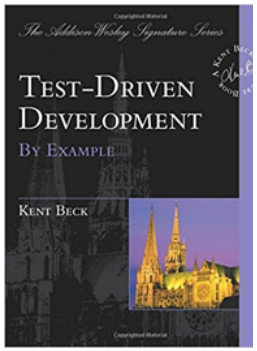
"We're told that Automating Acceptance Testing could be beneficial, but we don't code and we don't work closely with the developers."

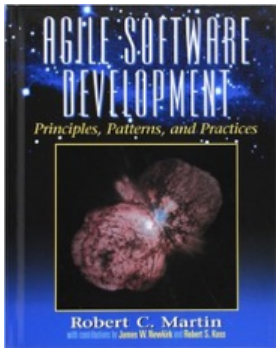
Senior Management:

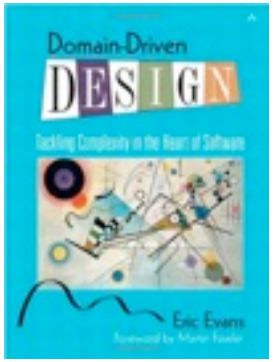
"Yes, we see the need for change but planning for it takes too much time."

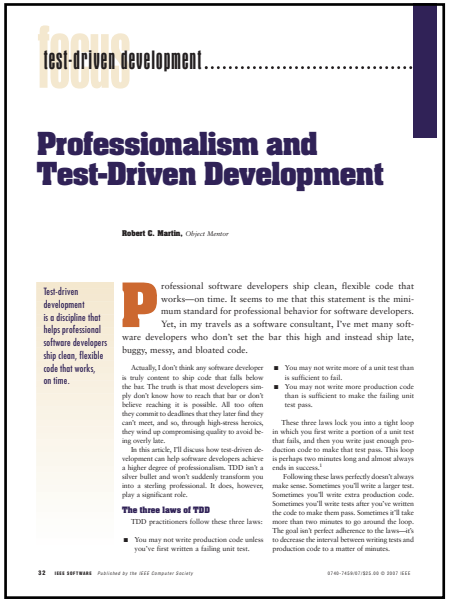
Questions?

Resources:









 #BDD-TDD

THANK YOU

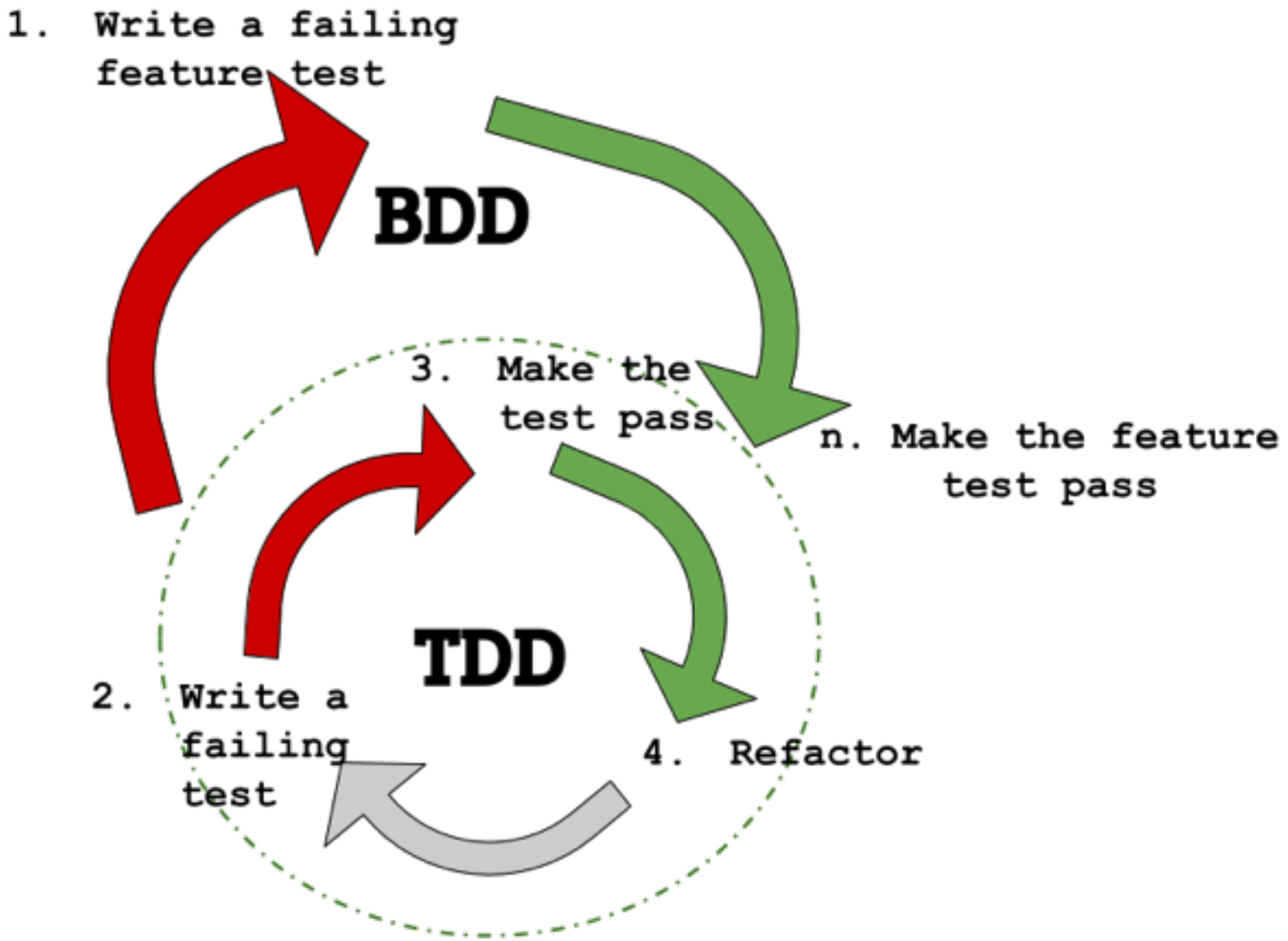
Bill Allen - agile Innovation Labs
Bill@agileInnov.com

Kieran Murphy - ThoughtWorks
KiMurphy@ThoughtWorks.com

The End

Slide Backlog

Process: The BDD - TDD Cycle



 #BDD-TDD

How NOT to write Behavior tests (these are functional)

Feature: : Payroll Admin - Add Employee

Scenario: Add a new hourly employee to the mPayroll system
Given I am a Payroll Admin
And I am on the 'home page'
When I click on the 'Add Employee'
Then I land on the Add Employee page
And I am asked 'Employee Type to Add'
When I click on the 'Hourly Employee'
Then I land on the Add Employee page with check box options for hourly

Scenario: Add a new salaried employee to the mPayroll system
Given I am a Payroll Admin
And I am on the 'home page'
When I click on the 'Add Employee'
Then I land on the Add Employee page
When I click on the 'Salaried Employee'
Then I land on the Add Employee page with check box options for salaried

Scenario: Add a new commissioned employee to the mPayroll system
Given I am a Payroll Admin
And I am on the 'home page'
When I click on the 'Add Employee'
Then I land on the Add Employee page
And I am asked 'Employee Type to Add'
When I click on the 'Commissioned Employee'
Then I land on the Add Employee page with check box options for commissioned

