ThoughtWorks studios

Essential Agile Engineering Practices

Agenda for Day 1: Lecture and Discussion Content

Introduction

- Introduction of trainers and participants
- Hopes & Fears exercise
- Learning objectives
- Intended audience
- Agile best practices review

Essential Agile Engineering Practices

- Pairing
- Refactoring
- Test Driven Design / Development
- Functional Testing - Tools and Practices
- Continuous Integration
- Regression Testing
- Test Automation
- Build and Release

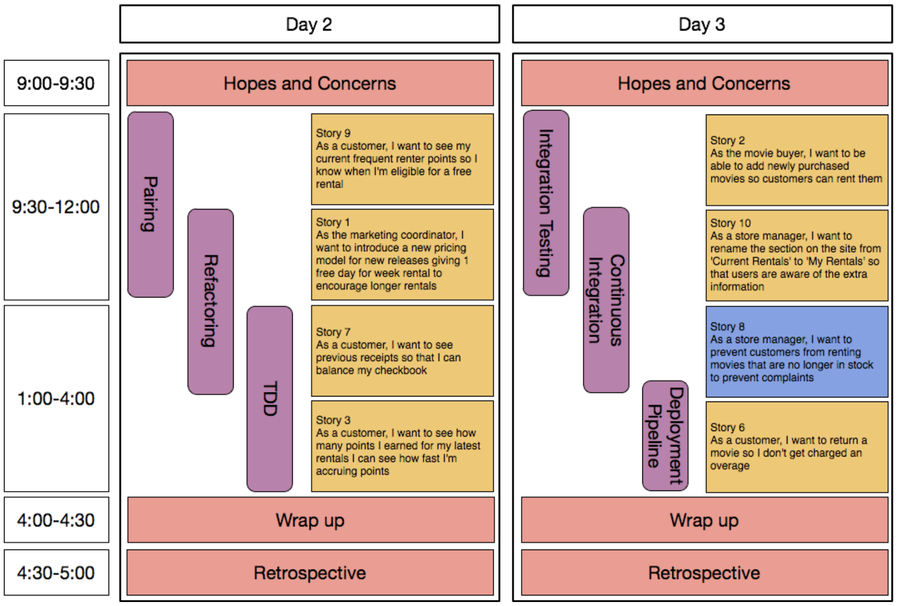
Days 2 and 3: Agile Engineering Lab

The Agile Engineering Lab is focused around an existing application for an online Video Rental business. Over the course of two days, participants are presented with a series of User Stories that detail new features to be added to the application. Agile engineering practices are introduced one by one as new challenges arise in the development effort.

Students will be working in development pairs and will be provided with a virtual training lab that contains all of the necessary development tools for the exercises. Typically, concepts will be introduced in a discussion context, followed by a demonstration by the instructor, and ultimately worked through by the students, both individually and as a collective group.

Many of the exercises are structured so that the students encounter a number of opportunities to exhibit anti-patterns. Sidebars will be conducted as these events occur on such topics as evolutionary design, self-documenting code, and the ability for developers to better determine story sizing based upon experience dealing with new requirements.

Agenda for Days 2 and 3: Agile Engineering Lab



Story 9
As a customer, I want to see my current frequent renter points so I know when I'm eligible for a free rental

Story 1
As the marketing coordinator, I want to introduce a new pricing model for new releases giving 1 free day for week rental to encourage longer rentals

Story 7
As a customer, I want to see previous receipts so that I can balance my checkbook

Story 3
As a customer, I want to see how many points I earned for my latest rentals I can see how fast I'm accruing points

Story 2
As the movie buyer, I want to be able to add newly purchased movies so customers can rent them

Story 10
As a store manager, I want to rename the section on the site from 'Current Rentals' to 'My Rentals' so that users are aware of the extra information

Story 8
As a store manager, I want to prevent customers from renting movies that are no longer in stock to prevent complaints

Story 6
As a customer, I want to return a movie so I don't get charged an overage

Day 1 - Start

Story #9 - *As a customer, I want to see my current frequent renter points so I know when I am eligible to receive a free rental.*

This should be a very basic story to get up and running as well as understanding the code base. There should be minimal required refactoring/testing.

Lessons: Pairing

Story #1 - *As the marketing coordinator, I want to introduce a new pricing model for new release movies that gives 1 free day for a full week of rental so that I can encourage longer rentals of new movies*

This is the start of learning about refactoring. Before refactoring in order to complete this story, we need to make sure the students enhance the existing tests to ensure we do not break functionality.

Lessons: More pairing ideas & Some Refactoring

Day 1 - Afternoon

Story #7 - *As a customer, I want to see previous receipts so that I can balance my checkbook*

This brings refactoring & TDD together. For the refactoring, it involves passing in a statement object (classic one from the book).

Lessons: More Refactoring & intro of TDD

Story #3 - *As a customer, I want to see how many points I earned for my latest rentals I can see how fast I'm accruing points*

This will focus mostly on TDD (although there will clearly be some refactoring around this)

Lessons: TDD

Day 2 - Start

Story #2 - *As the movie buyer, I want to be able to add newly purchased movies so customers can rent them*

The purpose of this story is to introduce automated acceptance testing. In order to minimize complexity of new technologies, we will introduce Twist 2.0 using Sahi and its recorder capability. The story complexity will be minimized so that the Developers can focus on writing the integration test.

Lessons: Integration testing

Story #10 - *As a store manager, I want to rename the section on the site from 'Current Rentals' to 'My Rentals' so that users are aware of the extra information*

The major purpose of this story is around keeping domain language consistent within integration tests and code. We will also comment on the different approaches of writing integration tests (i.e. test first, finish story before updating tests). Pairs can decide which they do. We'll also show them an already running Cruise instance and hook up CC tray. Each pair will have a pipeline & we will encourage them to check in when they finish the story.

Lessons: Integration testing & CI

Day 2 - Afternoon

Story #8- *As a store manager, I want to prevent customers from renting movies that are no longer in stock to prevent complaints*

This story will already be checked in and will represent work completed by an external dev team. It will have broken the build, but in a tricky way. Basically, we want the test to have to tear down, which will cause an individual test to succeed, but running the suite will fail. The pairs will have to fix the build.

Lessons: CI

Story #6 - *As a customer, I want to return a movie so I don't get charged an overage*

This is mostly recap of other things and also an introduction into build pipelining. After checking in, we will show them a continuous deployment setup that installs to a staging environment.

Lessons: Deployment Pipeline