Behavior driven design and user stories

Dr. Kosta Damevski

CMSC 355 - Software Engineering: Spec and Design

Fall 2015

VCU

School of Engineering | Computer Science

Announcements

- Start with group formation
  - Once done, use the web form to submit the composition of the group
- Assignment 2, handed out later this week
  - Group assignment
  - First batch of requirements for your app

Last Time

- Software phases
  - Requirements, Design, Implementation, Testing, Maintenance
- Software development models
  - Plan and Document: Waterfall and Spiral
  - Agile: Extreme Programming (Scrum, DSDM, others.)
- Today: start talking about requirements
  - Our goal for now is to cover a minimum set of items in order to start the project

Agile model review

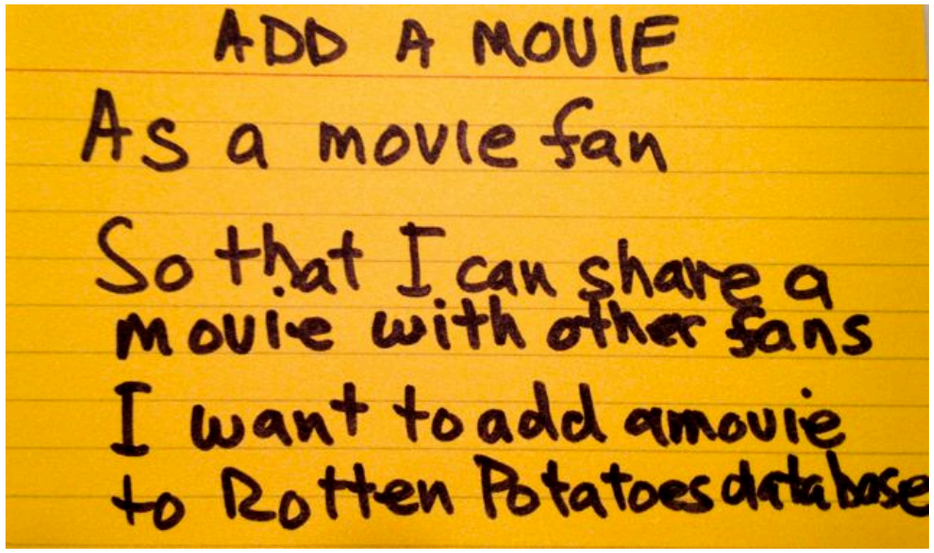
- Work closely, continuously with stakeholders to develop requirements, tests
  - Users, customers, developers, maintenance programmers, operators, project managers, …
- Maintain working prototype while deploying new features every iteration
  - Typically every 1 or 2 weeks
  - Instead of 5 major phases, each months long
- Check with stakeholders on what's next, to validate building right thing (vs. verify)

Behavior-Driven Design

- BDD asks questions about behavior of app before and during development to reduce miscommunication
    - Validation vs. Verification
- Requirements written down as user stories
    - Lightweight descriptions of how app used
- BDD concentrates on behavior of app vs. implementation of app
    - Test Driven Development or TDD (future segments) tests implementation

User Stories

- 1-3 sentences in everyday language
  - Fits on 3” x 5” index card
  - Written by/with customer
- “Connextra” format:
  - Title = feature name
  - ***As a [kind of stakeholder],***
    So that [I can achieve some goal],
    I want to [do some task]
  - 3 phrases must be there, can be in any order
- Idea: user story can be formulated as acceptance test before code is written



Why 3x5 cards?

- (from User Interface community)
- Nonthreatening => all stakeholders participate in brainstorming
- Easy to rearrange => all stakeholders participate in prioritization
- Since stories must be short, easy to change during development
    - Dev's often get new insights during development

Project backlog

- Real systems have 100s of user stories
- Backlog: User Stories not yet completed
- Prioritize so most valuable items highest
- Organize to form SW releases over time

Spike

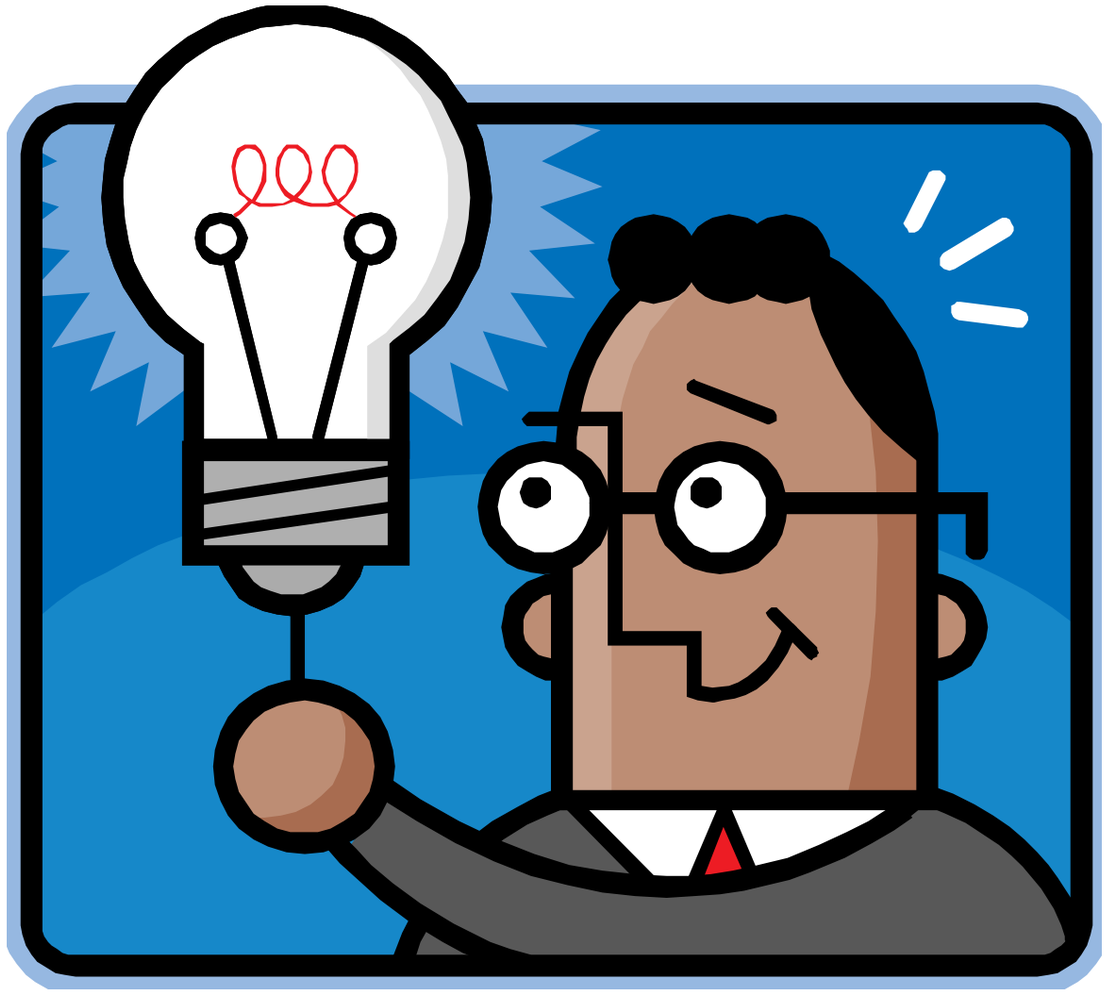
- Short investigation into technique or problem
  - e.g. spike on recommendation algorithms
- Experiment, hack, do whatever works
- Bound the time allotted
- When done, throw code away
- Now that know approach you want, write it right!

Which one is FALSE?

1. BDD is designed to help with validation (build the right thing) in addition to verification
2. User stories should include information about implementation choices
3. User stories in BDD play same role as design requirements in Plan-and-Document

User stories should be smart

- Specific
- Measurable
- Achievable
  - ideally, implement in 1 iteration
- Relevant
- Timeboxed
  - (know when to give up)



Specific & Measurable

- Each scenario testable
  - Implies known good input and expected results exist
- Anti-example: “UI should be user-friendly”
- Example: Given/When/Then.
  - **Given** some specific starting condition(s),
  - **When** I take specific action X,
  - **Then** one or more specific thing(s) should happen

Achievable

- Complete in 1 iteration
- If can’t deliver feature in 1 iteration, deliver subset of stories
    - Always aim for working code @ end of iteration
- If <1 story per iteration, need to improve point estimation per story

Relevant: “business value”

- Discover business value, or kill the story:
  - Protect revenue
  - Increase revenue
  - Manage cost
  - Increase brand value
  - Making the product remarkable
- Can you include stories that don’t have obvious business value?

Timeboxed

- Stop story when exceed time budget
- Give up or divide into smaller stories or reschedule what is left undone
- To avoid underestimating length of project
- Pivotal Tracker tracks velocity, which helps avoid underestimate

Which feature below is LEAST SMART?

1. User can search for a movie by title
2. Given I have a free movie pass, I want to redeem it for an eligible movie before it expires
3. When adding a movie, 99% of Add Movie pages should appear within 3 seconds
4. As a customer, I want to see the top 10 movies sold, listed by price, so that I can buy the cheapest ones first

Mapping user stories to acceptance tests

- Wouldn't it be great to automatically map 3x5 card user stories into tests for user to decide if accept the app?
- Tests from customer-friendly user stories
    - Acceptance: ensure satisfied customer
    - Integration: ensure interfaces between modules consistent assumptions, communicate correctly
- How could you run the tests without a human in the loop to perform the actions?

Use scenarios for user stories

- User story: typically maps to one feature
- Feature: ≥1 scenarios that show different ways a feature is used
  - Keywords Feature and Scenario identify respective components
  - both happy path & sad path scenarios
- Scenario: typically 3 - 8 steps

Example scenario

```
Feature: User can manually add movie
```

1 Feature

```
Scenario: Add a movie
```

≥1 Scenarios / Feature

```
  Given I am on the RottenPotatoes home page
  When I follow "Add new movie"
  Then I should be on the Create New Movie page
  When I fill in "Title" with "Men In Black"
  And I select "PG-13" from "Rating"
  And I press "Save Changes"
  Then I should be on the RottenPotatoes home page
  And I should see "Men In Black"
```

3 to 8 Steps / Scenario

Summary

- User stories are the means by which we express application requirements in agile processes
- Behavior Driven Design advocates a specific type of user story
- User stories should be SMART
- Map user stories into acceptance tests
    - via several scenarios

References

- *“Engineering Software as a Service” by Armando Fox and David Patterson (2nd Edition)*
- *“Software Engineering” by Ian Sommerville (10th Edition)*

Questions