CS1632, Lecture 24: Behavior-Driven Development

Bill Laboon

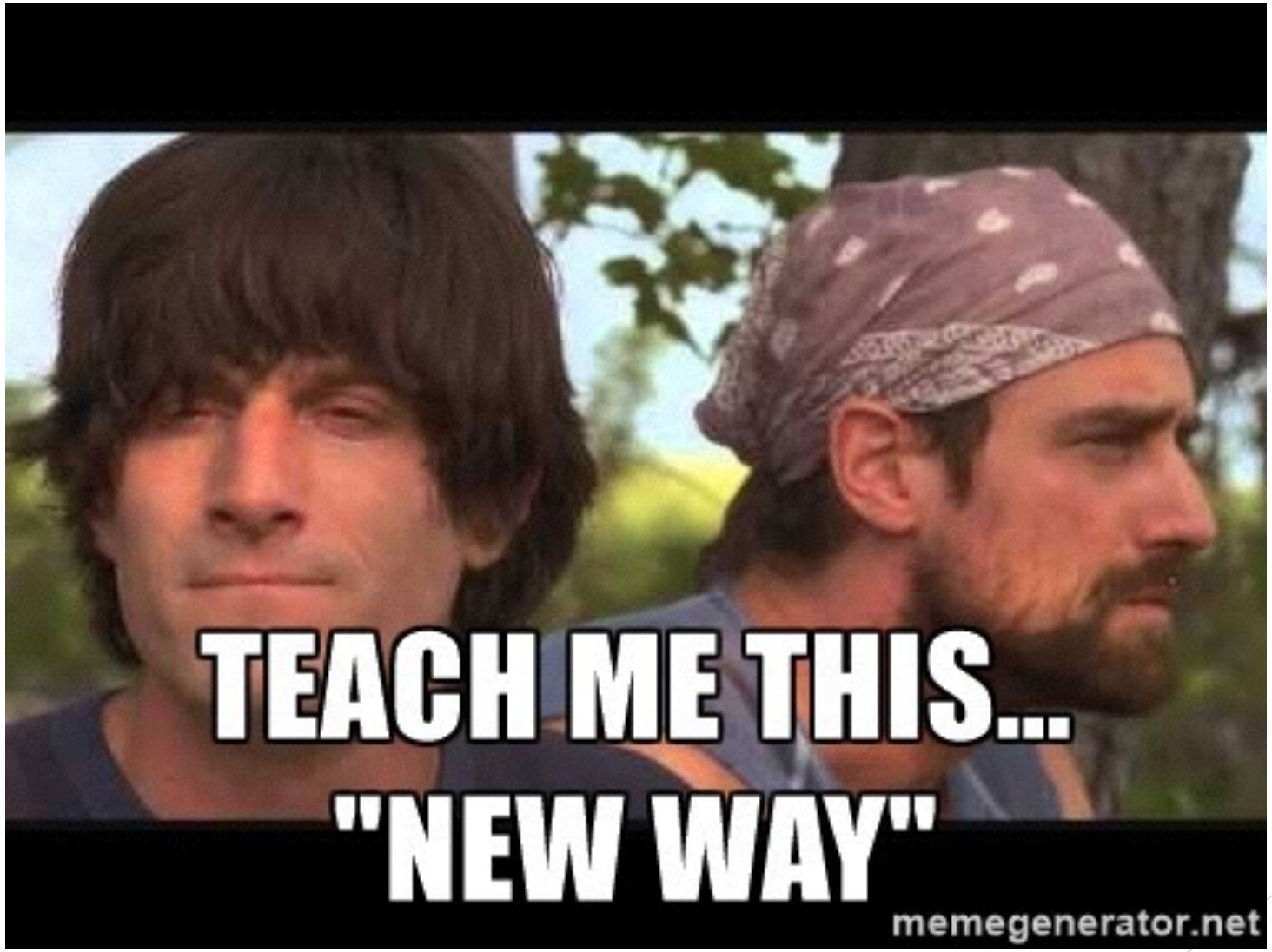
TDD is great and all, but…

The focus is still on code *qua* code. Remember, as developers and testers of software, we want to meet the needs of the end user, not the needs of the developer.

TDD is great and all, but..

TDD is supposed to move you away from the implementation details and have you focus on expected behavior, but the whole concept of very specific requirements and writing code to match them still keeps you in a “code-focused” mindset.

2006: Dan North shows a new way..



Behavior-Driven Development

In an article in Better Software magazine, Dan North explained an evolution of TDD that he called BDD, or "Behavior-Driven Development.“

-> An “evolution” of TDD which focuses on the general expected behavior of the application as opposed to focusing on blindly meeting the exact specifications of a system based on requirements

BDD != TDD

- TDD is focused on writing good code
- BDD is focused on building a good product
  - Good code should come out of that, though!

Let’s look at the “missing link” between TDD and BDD… ATDD

Fig. 1 Missing Link

TDD -> ATDD

- You can think of TDD as UTDD, “unit test-driven development”
- ATDD is “acceptance test-driven development”
- Write automated acceptance tests (“systems-level tests”), then fill in with TDD. Once full acceptance test works, move on to new acceptance test
- Still follow red-green-refactor, but two layers:
  - Acceptance tests
  - Unit tests

ATDD -> BDD

- Let’s work similarly to ATDD, but write everything in a way users, developers, and testers can all understand
- Develop an “Esperanto”
- Focus on describing the system – and tests! – in a way that users can undersand


ESPERANTO

Limitations of Jargon

- If you and your users/customers/project managers are all speaking different languages, communication becomes more difficult.
- If I told you a sorting algorithm was O(n!), you'd probably not want to use it.
- A non-technical person would probably not understand that sentence.

Requirements

- Do requirements really say what users want…
- … or do they say what developers should do?

No user will ever use this language to describe what they want

“The system shall enable the LOWTEMP warning light when two out of three internal thermometers agree that the ambient temperature is below -10 degrees Celsius (14 degrees F, 263 degrees Kelvin), as indicated by the INTHERM1, INTHERM2, and INTHERM3 registers.”

User Stories

What if... we could use our Esperanto to describe this in a language that both sides could understand?

We could easily go back to customers and see that we're on the right track, before or during development.

The Connextra Template (a/k/a the “role/function/reason” template)

As a <role>

I want <function>

So that <reason / benefit>

Examples

As a systems administrator
I want to create users
So that users in my domain can log in

As a user
I want to see my bank account balance
So that I know how much money I have

Taking Advantage of Common Humanity

- User stories allow flexibility and understanding

Role-Playing

You are part of the massive, global team tasked with creating the future of cat rental services,

Rent-A-Cat

Role-Playing

1. Users who want to rent cats for companionship
2. Users who want to rent cats for mousing
3. Users who want to rent cats for homework help
4. Cat trainers
5. Administrators
6. Marketing personnel

Testing User Stories

- Our “requirements” for a program are now a bit less specific than traditional requirements
- You will need to learn to deal with ambiguity!
  - Secret solution: communication, communication, communication

Scenarios are particular instantiations or situations of a user story

As a user
I want to log in
So that I can access my banking account

Scenario 1: I log in with correct username and password

Scenario 2: I log in with correct username, but incorrect password

Scenario 3: I log in with incorrect username

We can use the “Given / When / Then” template for scenarios

Example:

Given a correct username
And an incorrect password
When I try to log in with those credentials
Then I should receive an error page with "incorrect password entered" on it

Seem Familiar?

Given (preconditions)

When (execution steps)

Then (postconditions)

Re-writing in a way everyone can understand

Given one cat with name "Fluffernins“
When a user is logged in
And searches for "Fluffernins“
The cat named "Fluffernins" should appear on the search results

Preconditions: System is running. Cat named “Fluffernins” is in database. User is logged in.
Execution steps: Go to home page. In “Search” box, type “Fluffernins” and then click the “search” button.
Postconditions: New page appears. Of the cats listed, one should be “Fluffernins”.

Note Our Tests Are Less Specific!

- The blessing and curse of BDD
- Requires more of a tester, leaves room for ambiguity
- Pays off with increased user collaboration and understanding
- Depending on domain/field, may not be a good choice
  - Great for user-facing functionality
  - Worse for back-end or safety-critical development

Role-Playing

- Develop three scenarios for one of the user stories you developed earlier