WatirMelon.Blog
A Software Testing Blog by Alister Scott & Friends

Specification by Example: a love story

I attended a three day Advanced Agile Acceptance Testing/Specification by Example workshop in Melbourne last week run by Gojko Adzic, which I enjoyed immensely.

I took copious amounts of notes so I could regurgitate them here, but instead I decided to do something a bit different and write a story. It's a story about how a team can apply what he taught us to improve their software development practices. It's entirely fictitious (although I'd love to live in the Byron Bay hinterland!) and any resemblance to any real person or project is purely coincidental.

So here it is: Specification by Example: a Love Story (PDF). Enjoy.

Agile software, continuous delivery and passionate users

Rowly Emmett, a test consultant who lives here in Queensland, Australia, recently dismissed agile software development in a blog post titled "Agile Recipe". "...there is no difference between Agile and Waterfall" "The thing is, if people followed the (rigourous software methodology or waterfall) process correctly, then they wouldn't need to...

In "Agile Software Dev"

Australian Software Testers (who blog)

I was thinking it'd be a good idea to share my list of software testers I know in Australia. Of no coincidence, most of the great people in software testing in Australia write a blog about the craft. I wish everyone who worked in software testing wrote a blog because,...

In "community"

The 10 Do's, and 500* Don'ts of Automated Acceptance Testing

In "Automated Acceptance Testing"

Author: Alister Scott

Alister is an Excellence Wrangler for Automattic. View all posts by Alister Scott

Alister Scott / May 18, 2011 / Agile Software Dev, ATDD, Cucumber, Specification by Example / gojko adzic, living documentation, melbourne, Thoughtworks

25 thoughts on "Specification by Example: a love story"

Drew Preston

May 19, 2011 at 6:08 am

Your story clearly shows the difference between basic automation scripts, Cucumber style specs and Specification by Example. I'll be sharing this with my teams. Thanks for sharing with the community.

★ Like

Gio

May 19, 2011 at 3:32 pm

I like it :-)

★ Like

Hossam El-Deen

May 19, 2011 at 4:38 pm

Very Good Story (Y), the most interesting part is showing the evolution of from script to living documentation

★ Like

Arti

May 19, 2011 at 7:15 pm

Great illustration. But where was the love?! :)

★ Like

Alister Scott

May 20, 2011 at 9:50 am

Good point. I've added a final paragraph:

"The team had fallen in love in Specification by Example"

★ Like

Brad

May 24, 2012 at 12:31 am

> I've added a final paragraph:
Yes you did. And it wraps up very nicely.
But shouldn't it be:
The team had fallen in love *with* Specification by Example.
Or is this a US/Aus difference?
Thanks,
Brad.

★ Like

Andrew Premdas

May 20, 2011 at 2:37 am

Nice story, I like the way you develop the specifications, but I think you can go further. Your final solution requires you to read comments and then also translate them into a table. I think the following is much simpler:

Feature: Beautiful Tea Shipping Costs

Scenario: Australian customers pay GST
Given the customer is from Australia
When the customer buys something
Then the customer pays GST

Scenario: Overseas customers don't GST
Given the customer is from overseas
When the customer buys something
Then the customer does not pay GST

Scenario: Australian customers can get free shipping
Given the customer is from Australia
When the customer buys lots
Then the customer gets free shipping

Scenario: Australian customers can get free shipping
Given the customer is from Australia
When the customer buys bugger all
Then the customer pays for shipping

Scenario: Overseas customers can't get free shipping
Given the customer is from overseas
When the customer buys bugger all
Then the customer pays for shipping

Scenario: Overseas customers can't get free shipping, even if they buy lots
Given the customer is from overseas
When the customer buys lots
Then the customer pays for shipping

all best

Andrew

★ Like

Alister Scott

May 20, 2011 at 9:57 am

Thanks for your comment Andrew.
I guess we'll have to disagree. I dislike having multiple scenarios as you have to read through each one. I like the table as it shows you 5-6 key examples all together which you can easily compare and contrast.
The comments aren't exactly required, and aren't executed, but they clearly explain the examples in the table below.
I also noticed your scenarios don't include the $100 value. I think that's key, because whilst it is specific, it is the rule for free shipping. Without including this in the specification, you have to dig into the system code to work out this value, which is the thing I am trying to avoid.
Sure, if this $100 is updated you need to update your specs, but this have the opposite effect, if the $100 was inadvertently changed, then these specs would fail immediately, and you could easily see why.

★ Like

Andrew Premdas

May 20, 2011 at 11:29 am

Alister

Of course there is more than one way to skin a cat, and the table example

does at first seem concise. However in general I think tables are error prone and harder to understand and generally overused.

I have to disagree with you about the comments above the table. They are essential without them the feature makes no sense.

Whether the $100 value should be in the spec is very context dependent. I left it out to highlight a different way of doing things. The context is very much about whether the business really wants/needs the exact value in a feature. If you do put the value in a feature, you still have to specify the value in the code somewhere so now you have two places where the value is defined (not very DRY). In addition when the the real value is changed deliberately and the application still works, the feature failing will be a false result. You have to balance the likelihood and consequences of each type of change.

Reading through the many smaller scenarios highlights, that there are things missing from the specification (for example exploiting the potential of free postage to encourage further purchases for an order). Tables make this harder to see. The smaller scenarios show that you actually have two or three shipping features here (overseas shipping, domestic shipping ...). So you could consider a refactor to separate these different areas; having small simple scenarios makes refactoring easier. There are strong parallels here with the composed method pattern in coding. Additionally step definitions are far easier to write and re-use in these simple scenarios.

★ Like

David Peterson

June 2, 2011 at 1:39 am

> It seems ironic that you're talking about DRY principles when your version of the examples continually repeat the words "Given the customer, When the customer, Then the customer".
>
> Reading repetitive scenarios like this is extremely tedious, which means people don't read them properly, which means bugs slip in. That's my experience anyway.

★ Like

Chuck van der Linden

October 13, 2011 at 6:49 am

Having the value in there is useful for a number of reasons. It's 'food' for the dev as they get to the writing of unit tests, it gives them an exact value, instead of having to hunt down a PO and say 'wth is the dividing line between bugger-all and lots?' It It allows the tester reviewing the story to add a few more rows and make sure the boundaries are covered, and it expresses the business rule of what exactly defines when someone gets treatment x or y.

It also allows for some discussion when the scenario is initially presented, such as "so, if someone has $99.99 then they don't get free shipping? or are we going to round up anything above $99.00? Is the Tax included or is that before tax?

In short, at least as far as I'm concerned, when it comes to things like this, specific examples good, vague examples bad.

★ Like

Andrew Premdas

June 2, 2011 at 10:06 pm

David,

Thankyou for reading my scenario's. The repetition in my version is tedious, but that just indicates there is more work to be done.

As I pointed out there are actually several different scenarios being covered by the table example. When you expand that into the format I've used this becomes more obvious. Now we can improve readability further by dividing things up and refactoring. The following divides on shipping destination as an example but you could divide on GST/nonGST or buying lots/little

Feature: Overseas Shipping

Background:
Given the customer is from overseas

Scenario: Buying lots
When the customer buys lots
Then the customer pays for shipping

Scenario: Buying bugger all
When the customer buys bugger all
Then the customer pays for shipping

And now you have easy to read features and a place to further explore overseas shipping.

★ Like

Miguel Almeida

June 3, 2011 at 3:38 am

Alister, forgive me if my question seems obvious (I've never used cucumber), but how are things actually tested in the last phase? More specifically, is it:
a) using a black box, ie integration test on a test website deployed somewhere?
b) testing the internals (in this example, it would mean testing, for instance, the service that calculates shipping rate).

The question arises because while b) seems logical, we're actually losing what was being tested in the selenium script. For instance, if you forget to put the "order" button, the service might be perfect but the system won't work. On the other hand, if you're in a) and still testing the system as a whole, then how do you refrain from having to do the brittle stuff like having to use xpath or css to select buttons or inputs?

★ Like

Tony Godwin

March 31, 2012 at 3:22 am

I have enjoyed this love story and I have shared it with my team at work. In your conclusion you use the acronym "BA/SME." I am guessing that SME is subject matter expert, can you please confirm this and give me a definition of BA?

★ Like

Alister Scott

March 31, 2012 at 12:52 pm

Yes, SME = Subject Matter Expert and BA = Business Analyst. Thanks

★ Like

OM

March 31, 2012 at 7:18 pm

Good stuff mate. Explained very well :)

★ Like

EasonHan

May 22, 2012 at 12:25 pm

Can not download the pdf file.

★ Like

Alister Scott

May 23, 2012 at 8:43 am

Works fine for me. Emailed it to you.

★ Like

Jonah

July 1, 2012 at 7:34 am

Alister,

Would you mind pasting the actual code file for the final table version

specification in your article? Thanks!

★ Like

Alister Scott

July 4, 2012 at 1:54 pm

The article was theoretical so there wasn't any actual code sorry.

★ Like

Artem

July 12, 2012 at 9:30 am

If I am getting it right, the Cucumber is for people without knowledge of If..else.. syntax? I just can't get who need it....

★ Like

Sébastien Bouffard

August 10, 2012 at 12:32 pm

I am in the process of reading "Bridging the communication gap" and "Specification by example" both from Gojko Adzic and I am still a little confused with all this stuff.

Consider this:

1- In "Bridging the communication gap", Gojko makes it clear that specification by example is not a replacement for testing.

2- Gojko advocates against testing business rules through the UI.
3- Specification by Example is something that should happen before implementation when you probably have no UI to test from.

So considering this, specification by example should not be seen as a replacement for your Selenium scripts.
Strict application of Specification By Example as defined by Gojko would result in preventing the developer from coding the system so that it charges GST to customers outside Australia. But it would miss out on the integrator screwing up the "Add to cart" button.

Seems to me that even using Specification By Example, you would still need at least one Selenium (or any other such tool) test to assert that you can actually do a purchase through the UI. But it might save you from the burden of running it several times to cover all the business cases through the UI, which is a very good thing.

Anyway, thanks for this great post!

★ Like

Mike

February 27, 2013 at 2:49 pm

Natural language systems like this are always a bad idea. They appear simpler at first but natural language is NOT ambiguous and too loose to be used well in testing. Some people get excited by natural language tools like this. Apple made the same mistake and created the horrendous AppleScript, which is hated by most people. It probably excites managers and people who don't develop software!

★ Like

Pingback: Most interesting links of May « The Holy Java

Pingback: Behaviour-Driven Development at the FT | Engine Room, a blog by the Financial Times Technology Department

Comments are closed.