Pilot Program Access List

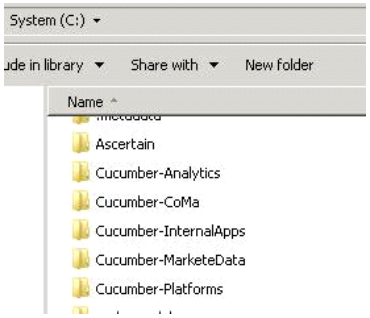
Friday, May 22, 2015 4:40 AM

- Kate Kuzin
- Rebecca Karger
- Rajeeta Krishnan
- Swetha Baswapuram
- Deekshith Lingoju

Getting started

Thursday, July 16, 2015 8:51 AM

The Cucumber Pilot Machine (tellusdevb04) will be available on Monday 25 May for RDP connections (max 10). It will have installed Cucumber-Ruby, Cucumber-Protractor-Jasmine, Notepad++, Aptana (IDE), trial version of Webstorm and Fiddler(for those who need overrides).
Also Eyme has installed a couple of sample projects for Ruby and Protractor in the respective group folders in C:\. A shortcut to the IDEs have been placed in C:\ so that you can launch them easily. When you launch the IDEs ensure that they open the correct project associated with your group to avoid editing other's projects. Refer Cucumber for instructions



Getting started:

1) Launch Remote Desktop Connection
2) Enter tellusdevb04
3) Click on Connect
4) Log on with windows credentials
   a. Greenwich*yourname*
   b. Windows password
5) Click on OK button

Once VM is up and running

6) Review Sample via Notepad++
7) File > Open C:\Cucumber - InternalApps\Cucumber-Protractor\Specs\filter-view.feature

Cucumber server machine Access list-

Friday, May 22, 2015 4:41 AM

Winnie Oyulu
Sandhya Nadig
Shiva Kumar Macharla
Asha Veerapaneni

A couple of findings regarding the ongoing discussion on running feature files:

I was able to replicate the following scenarios;

1. In a Normal instance a feature file should match a Step definition or ruby file, this is the ideal scenario where the result will show a 'pending'Status after running- which means its pending code implementation

2. In a case where we run "duplicate feature" files against a step definition file,it will return an 'Undefined' error meaning cucumber cannot tell which feature file to run.

-The same error is thrown when there is no step definition file created.

3. Running a feature file whose steps partially match the step definition file will return 'Undefined, passed and pending steps

4.Each feature file can be tagged eg @featurefile1 and the same Tag added to a matching Step definition file, however that does not make the two files Unique

Which Concludes that each feature file has to be unique as cucumber only matches word to word with the step definition file.
-How to determine the level of complexity as far as code reuse within a growing project remains to be researched

-Winnie

Feature List Test Run

Monday, June 15, 2015 1:35 PM

Step 1: Cucumber Automation Tool Pilot with a Price History Test Case: http://is.factset.com/rpd/Summary.aspx?messageId=17710826 (Last update June 15, 2015)
Step 2: Demo for stakeholders.

Cucumber Pilot Program Access List - MktData

Tuesday, May 26, 2015 4:00 PM

Robert Konstantinis
Joe Valiante
Nisar Ahmed Hothur
Kranthi Shishupth Ghanta
Michael Reid
Dan Enticknap
Jim Butler

Observations:

1. While writing feature files, analyst has to write each scenario independent of other, initially I have written 3 scenarios and 1 background. When I have started writing script, I felt like those are dependent so I made it to 2 scenarios and 1 background to be independent.
   *Constant discussion or interaction should be there between tester and scripter to exchange the ideas.
2. Numbering should not be given to scenarios, like (scenarios1, scenario 2) .
3. We can consider each feature file to be a test case.
4. Background is a good option for repetitive test steps.
5. Even colon (:) has to be given properly.
6. Passing arguments for feature file to the script should be done carefully. (inputs provided in the feature file)
7. Avoid using (*) in writing feature file steps, it will reduce the readability for other testers or automation people.
8. All the key words should start with the capital letter.

Automation observations:

1. Every test step definition should match with the feature file, even a word change in the ruby script will be seen as step not defined.
2. Variables declared in the definitions should be instance variable they cannot be local variable to use them in any other part of the script.
3. Nothing can be written other than in the definitions part, as it will through an error. So requiring some external files may be a difficult task to use it in the script.
4. Easy part to implement in practice, simple ruby commands will do the work.

Helpful suggestions:

1. Improving the way to catch the errors in the feature file.
2. Limiting not more than 5 minutes for feature file to run the step definition.
3. Limitations of the ruby script or cucumber should be well known to analyst who is writing the feature file.
   For e.g.: If an analyst writes a feature file for the given specs from development team which has a feature wherein we cannot automate the stuff using cucumber , that would be waste of analyst time in writing the feature files which is related to the not automatable stuff.
4. Any change of feature file after implementation has to come through automation scripter.

Cucumber Pilot Program Access List

Friday, May 22, 2015 4:41 AM

Dave Caruso
Terrence Tseng
Priya Suryadevara
Monica Taylor
Dan Hoffkins
Nkem Omokpo
Yasaswi Taliyakula

Il4 Feedback

Thursday, May 28, 2015 8:55 AM

- Limited documentation to be found--examples provided on Server helpful once Eyme was able to explain in detail
- I was able to create a simple IL4 basic search test case with cucumber, Giri has to create the accompanying steps file
- Used the following these resources to broaden my understanding : https://www.youtube.com/watch?v=57134cHJlAs; https://angular.github.io/protractor/#/tutorial; http://christian.fei.ninja/Write-your-Protractor-tests-in-Cucumber/
- I think this would be more useful if I had the knowledge to script the feature step file myself
- For those that don't script how is this saving time over creating regular test cases in QAI?

SDM Feedback

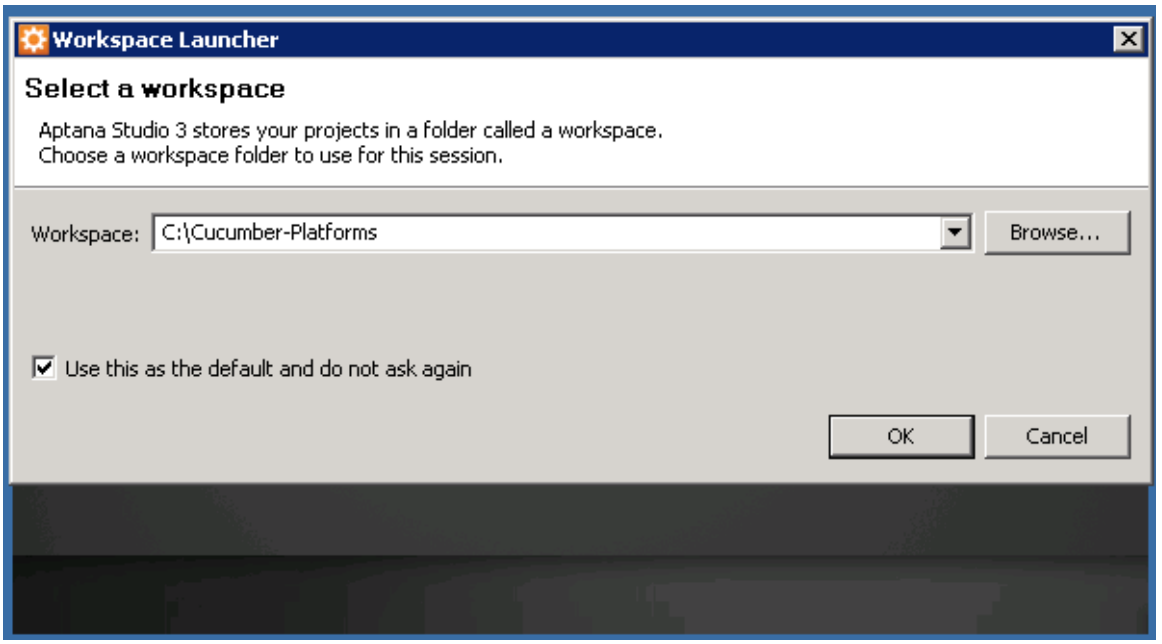
Thursday, May 28, 2015 8:55 AM

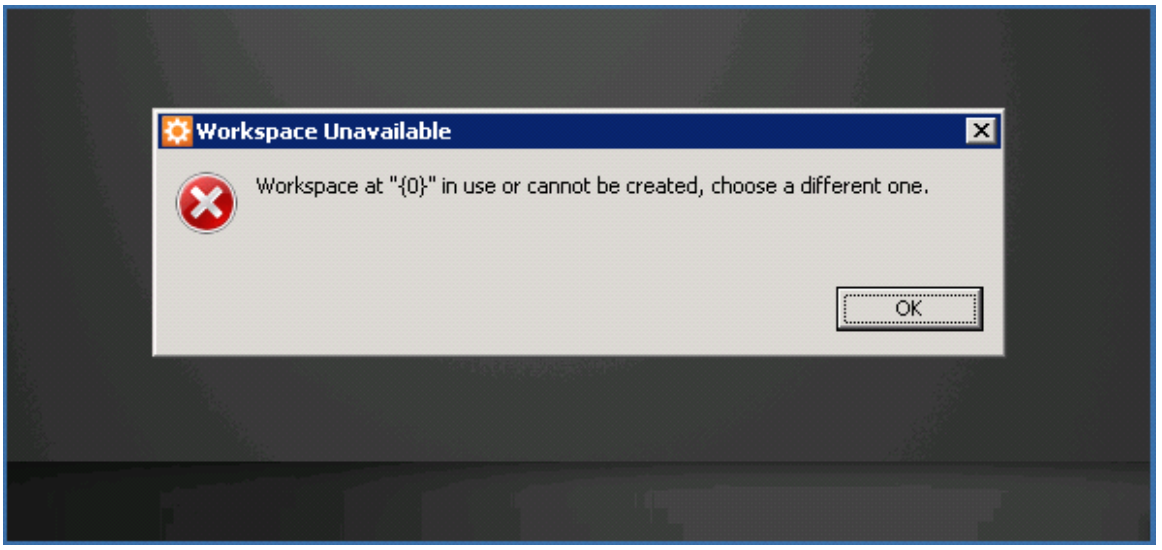
6/4 I RDPed into tellusdevb04, information on getting started is a little confusing

Since then I have reviewed :
https://cucumber.io/

Yasaswi's feedback:

6/10: I tried opening Cucumber-Platform workspace in Aptana using the instructions here: Cucumber
It throws the below error and I am not able to open the workspace.





Does the error mean, someone else has already opened and not closed the workspace?

6/12: SDM doesn't seem to be right piece for cucumber. I was able to open the Workspace in another way, but the issue I mentioned still the same.

Note there is a new version of the Platform workspace located on tellusdevb04, **Cucumber_Platforms**

Note instructions on launching the Aptana could be clearer.

Had an overview with Vasu which was very helpful in getting me on the right track.
SDM Homepage is not "truly" a web base product and we were able to identify only limited functionality to Automated via Cucumber using Protractor.

To help write good Feature files
https://github.com/cucumber/cucumber/wiki/Gherkin

6/17: From a one-one meeting with Nkem:
We cannot add Favorite websites in web version of SDM home page - Websites added in PC version are never carried forward.

So below are the limited things we may consider automating with cucumber:

- Section header
- Tooltips
- Text boxes
- Tooltips of few elements

Nkem and I would be drafting a feature file for the above said scenarios.

****Note, while the created project folders can be accessed.. The same folder can't be used as workspace across different users. Any user who logs into the VM can open a new workspace(may be created by the current user) and can import to the project folders which were created on product name(Platform, MDQA, etc).

Vasu recommended Terrence Tseng would be able to assist scripting our Cucumber Feature File since he has knowledge of Protractor
Yasaswi has created and sent over the SDM Feature file to Terrence

sdm-home
page

After speaking to David Caruso and Dan Hoffkins, Terrence has other priorities on his plate. I have reached out to Vasu to provide another QA Automation team resource would be able to assist us

- Vasu has assigned Giri to assist, Yasaswi gave Giri a brief demo of SDM Homepage 6/24
- Yasaswi has been able to generate the Protractor scripts from the SDM Home page Feature file into Cucumber

- Giri no longer has do it. Vasu confirmed Giri hasn't began coding yet.
- We will reach out to Giri if we need assistance down the line.

Yasaswi finished integrating protractor code to cucumber framework(7/1)

Prerequisites:

- Install node.js
- Install Protractor
- Install Webstorm(Trail Version)
- You can set up the Protractor following the instructions here: http://infonet.factset.com/view/QualityAssurance/ProtractorTraining

Cumumber - Protractor - Project structure:

The config.js is the main file that has all the scripts to be run. This file has to be created in the Project folder.
Create the Project folder and create the below two directories:

- Specs
- PageObjects

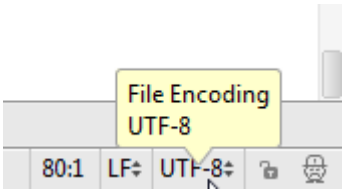
Under specs folder, create the step_definitions directory. The feature file is also to be created under this directory.

Feature file is to be created with .feature extension.

Spec file is .js file which actually has the jasmine code - the actual executable code.

Both feature file(s) and spec file(s) are to be provided in the conf.js file which runs the cucumber project.

Important things - writing feature file:

- Avoid special characters in the .when, .then, .given description - Jasmine do not accept the special characters - though it doesn't throw any error while writing and saving.
- Make sure your txt file format is UTF-8. Which can be set at the bottom right corner of the Webstorm IDE.
  - 

- Use double quotes in the gherkhin's description for the strings that you pass to the functions as parameters. Scripters can make use of exact parameters as expected values for assertions so as to minimize the code maintenance. For example, if you want to verify a tooltip to have "click below" enclose it in the .then description so the exact string can be used as expected tooltip. So in future, if there is any change in the tooltip, only feature file updating would fix the change. Spec file need not to be modified.

Important things - spec writing(coding in jasmine)

- Make sure, the main class of the angular application is given as the rootElement in the conf.js file. Found this workaround on google.. This is very important when dealing with non-angular page interaction in the cucumber - protractor project.

- ```
  var cucumber = require('protractor-jasmine-cucumber');

  exports.config = {
    baseUrl: 'https://spar.staging-cauth.factset.com',
    rootElement: '.ng-scope', // location of ng-app directive
    seleniumAddress: 'http://localhost:4444/wd/hub',
  ```
- Choose to set the flag true: `browser.ignoreSynchronization` while dealing with non-angular applications. Set it back to false when working on angular apps.
- A sample project is saved at H:\Department\QualityAssurance\AtlasQA\Office Integration\Yasaswi \Automation**Cucumber_SDMHomepage**
  - Copy the entire folder into your C:Drive
  - Install Protractor, Nodejs, Jasmine, Cucumber and Webstorm as documented in the Twiki
  - Open the folder as project in Webstorm IDE.
  - Run the conf.js

Please reach out to Yasaswi if you have any questions.

Sources:

To learn Jasmine: http://jasmine.github.io/2.3/introduction.html
Feature files, Cucumber integration: https://github.com/DealerDotCom/protractor-jasmine-cucumber

Feedback as Analyst:

- We need a well documented workflow
- Analyst needs to understand what can and can't be Automated via cucumber
- Easy to maintain and keep track of active changes in application which may affect automation scripts.
- Makes communication better between analyst and automation engineer.
- The concept of QAIDs has not yet been adopted by OI Web/PC development
  - Is this a limitation for us to use Cucumber?
    - Note we had a meeting 7/15 with OI engineering leads about adopting QA Automation code abstraction methodology, they are onboard

Feedback as automation engineer:

- Protractor is more complex language compared to Ruby.
- Jasmine provides readymade logging methods.
- Variables need not be declared in the scripts and can directly be taken from feature files which saves lot of maintenance time.
- The concept of QAIDs has not yet been adopted by OI Web/PC development
  - Is this a limitation for us to use Cucumber?
    - Note we had a meeting 7/15 with OI engineering leads about adopting QA Automation code abstraction methodology, they are onboard
  ```

Chart Formatter Feedback

Thursday, May 28, 2015 8:55 AM

06/08/2015
Browsed the following links:
https://cucumber.io/
Watched 8 min video on BDD
https://github.com/strongqa/howitzer/wiki/Cucumber-Best-Practices
https://github.com/angular/protractor/blob/master/docs/frameworks.md
"*Note: Limited support for Cucumber is available as of January 2015. Support for Cucumber in Protractor is maintained by the community, so bug fixes may be slow."*
http://infonet/view/Training/ProgrammingTopicals

RDP'd to tellusdevb04 - successfully logged in and viewed project in WebStorm

06/09/2015
Initial questions / observations:

File - "Filter-viewSteps.js"
1)It looks like we are using both Jasmine and Cucumber as a Protractor supported BDD test framework.
-Is this what we in QA prefer at this time?
2)It would be helpful if we have examples of a typical cucumber BDD like workflow…
How each step in the process looks.

i.e.	a)Create a scenario detailed in a cucumber like format: Given, And, When, Then…
	b)Write a definition
	c)Write code
	d)What about infrastructure (who creates the rest of the code?)

3)Layers of abstraction end goal: Who is expected to create what?
i.e. 1)analyst creates scenario details, 2)Scripter creates test code, 3)Engineering creates infrastructure?

Feature file
It basically described what the test case will do using cucumber specific syntax.
Using what we already have for formatter flyout test case, I simply create the feature file based on 1 of the scenario.

Page object/Helpers
These were created for formatter automation and was easily reusable here since.
IF this were actual TDD, we would have feature files first written out, then create these page objects.
But since we started cucumber pilot program after formatter automation, a lot of the infrastructure can be reused here.

E2espec

Basically wrote this simple scenario using FDSChartAPI and following the cucumber-jasmine format.

Summary

Links to Cucumber Documentation?

Friday, May 22, 2015 4:42 AM

- Cucumber Pilot FM Program
  - Project Plan listed here RPD:17619013
  - Links
    - Cucumber: https://cukes.info/
    - FactSet Ruby Cucumber Installation: Cucumber Ruby Installation
    - FactSet Jasmine-Protractor Installation: Getting Started with Cucumber-jasmine for NodeJS
      - Note this is different from Cucumber.js
    - Internet - Cucumber Protractor: https://github.com/AndrewKeig/protractor-cucumber
    - Cucumber best practices: https://github.com/strongqa/howitzer/wiki/Cucumber-Best-Practices

What's a sample workflow?

Friday, May 22, 2015 4:45 AM

From Alisa

Example:

1. What do stakeholders need to do to get cucumber installed?
    a. Scripters required to install on Laptop – only (analysts use the VM see item b)
        i. Project Plan listed here RPD:17619013
        ii. Links
            1. Cucumber: https://cukes.info/
            2. FactSet Ruby Cucumber Installation: Cucumber Ruby Installation
            3. FactSet Jasmine-Protractor Installation: Getting Started with Cucumber-jasmine for NodeJS
                a. Note this is different from Cucumber.js
            4. Internet - Cucumber Protractor: https://github.com/AndrewKeig/protractor-cucumber
        iii. Cucumber best practices: https://github.com/strongqa/howitzer/wiki/Cucumber-Best-Practices
    b. The VM can be used for analysts & FM
        i. Be ready Monday 5/24
        ii. Allow FM to write feature files
            1. Need to write in proper file structure
            2. Licenses for 10 people
    c. Who writes Gherkin in cucumber?
        i. Pooja (will need training)
    d. Who writes the code behind the Gherkin?
        i. Gireesh
    e. QAID creation
        i. Josh and Gireesh
2. Do we use Morph for Issue Tracker or FDSQAR_C account and serial #?
    a. Staging site is Lima Auth (focus on staging which is where most testing occurs)
        i. Error without fiddle
        ii. Staging should bring up factset.net
        iii. Need to pull out a serial # out of the automation pool for the Issue Tracker tests
        iv. Need to create issues on production environment for the Issue Tracker serial numbers
            1. Allows us to copy production DB to Staging and maintain the issues
    b. Dev need to use the Fiddler hack to inject headers
        i. we will not do automated testing on Dev
        ii. Fiddler issues that would need to be resolved
3. Where is all the documentation for the tasks?
    a. Use a OneNote with a tab for each team
        i. Automation will merge the data into a generic help file….
    b. Can we update as we go along? Yes
4. Initiate the use of QAIDs
    a. Review the documents: http://tellus.pc.factset.com/QA%20Data%20Attributes.htm#_Toc417352206
    b. Josh and Gireesh working on this
        i. Need to be strict on the naming at 1st.
        ii. Focus on the elements that we know that will change more often.
        iii. Focus on the elements used for automated tests
        iv. Ex: use a OneNote to document the QAIDs

Name of the Cucumber Pilot Machine?

Friday, May 22, 2015 4:48 AM

tellusdevb04

Guidelines???

Monday, June 08, 2015 1:58 PM

The idea for the Pilot program is they just need to try and figure out how they would like to implement it to coincide with our existing options in QAI and Tellus.

If they don't, work with the scripter to try out all these and figure out the best way. However if you are looking for some rules of thumb: A feature file should contain 1 feature of an app. A scenario is a test case, so yes 1 scenario per test case. But a feature can have many scenarios, no coupling between scenarios, they should be independent of each other

Handling Cucumber files on the VM:

All scripters will copy the files from the VM C drive to their C drive on their PC and write the code and then copy it back up to the VM C drive.

Cucumber Processes

Wednesday, August 12, 2015 9:22 AM

Cucumber production process/design meeting continues.

Teams are moving forward off of the pilot program and on to production for Cucumber.

Running compiler code test cases is different from executing test cases outside of the build process and creating new ones. The best thing to do is divide these into separate processes.

Phases- Follow existing processes

Manual Process

1. Analyst - create a new test case in QAI following the feature file format
2. Analyst - request Scripter to automate in cucumber the test case
3. Scripter - copy or writes text of the test case to the feature file on their laptop or another machine
4. Goes to Tellus Test Manager (TTM) and set the status to InProgress
5. Scripter - validates feature file
6. Scripter - Generates the core for the step definition file
7. Scripter - Completes the step definition file
8. Scripter - Tests the codes if is good then the code is copied to the Source Code Repository (SCM)
9. Scripter set the status in TTM to Staging
10. Scripter - when the Staging version is good the code is move to Production in the SCM TTM status is set to Production.

Install cucumber on QA Auto VMs and SG server Ruby and Protractor versions
Analyst and Scripter put files in right place
Flag TC with QA defined Gherkin (QAI needs to provide this), tellus needs to read
PHP for cucumber CRON - dispatcher
Vasu - Cucumber dispatcher
Condition - Results log. Node .Feature instead of Test Case name. Tellus results parsing. Cucumber - Protractor
Project Structure for Ruby, investigate structure for Ruby

Automated Process

1. Analyst - create a new test case in QAI using the editor provided and the processes outlined in Getting Started
   a. Possible editors that have to be vetted (list features of editor)
      i. https://gherkineditor.codeplex.com/
      ii. https://github.com/cucumber/gherkin-editor
   b. http://is.factset.com/rpd/summary.aspx?messageId=19439053
2. Analyst - Validates the test case (feature file) using the editor listed above
   a. Syntax checker
   b. Source code link up
3. Analyst - set the status to can be automated
4. Scripter - slurps the code down to the laptop or other machine for testing
5. Scripter - Generates the core for the step definition file
6. Scripter - Completes the step definition file
   a. Looks at tags to link code
7. Scripter - Tests the codes if is good then the code is copied to the Source Code Repository (SCM)
8. Scripter set the status in TTM to Staging
9. Scripter - when the Staging version is good the code is move to Production in the SCM TTM status is set to Production.

7/21/2015

Tuesday, July 21, 2015 1:48 PM

Here are the design questions

1. Where will the Feature Files reside (they need source code management (SCM) and versioning)
2. How and where will we edit them.
3. Where will the Step Definitions file reside also SCM
4. ~~Cucumber will be a tool type~~ (Selenium or Protractor)
5. Cucumber test cases in QAI.

Phase 1.
Edits
Sync to a SCM for.
Edit on QAI send to TTM, Automation create
Factset Product
Validate the Feature file

Edit in TTM and validate
Once validated pushed to QAI as a test case.

Validate and Share buttons in QAI

Running compiler code test cases is different from executing test cases outside of the build process and creating new ones. The best thing to do is divide these into separate processes.

1. Creating a new feature file
2. Validate the feature file
3. Updating the step definition

Getting Started from Lessons learned

Wednesday, July 22, 2015 6:45 PM

INTRODUCTION

Cucumber runs automated acceptance tests written in a behavior-driven development (BDD) style. It allows automation of functional validation in easily readable and understandable format (like plain English) called Gherkin to Business Analysts, Developers, Testers, etc.

(Suggested intro addition from Jay)

Cucumber is all about separating the "what & why" from the "how" in our QA tests. We are used to maintaining detailed Test Cases with very specific instructions on how to manually navigate application functionality with just a vague reference to the "what" (the business workflow) and "why" (the value or goal / outcome of executing that workflow).

Now Analysts (even PD) can use Cucumber to describe workflows and values that then refer to automated test code for the "how" (scripted steps for detailed, specific execution of the navigation). By so doing, Analysts are empowered to create Test Cases before development occurs, and to maintain them with fewer test code updates required.

Here are the components:

- **Feature File** - *Feature files are used to write the requirements or acceptance tests in plain English (Gherkin). The steps are the application specification. All the feature files ends with .feature extension.*
  - **Feature** - *This gives information about the high level business functionality and the purpose of the test. Everybody should be able to understand the intent of feature file by reading this description and should be brief*
  - **Scenario** - *The scenario should represent what the test is all about. Each scenario should follow given, when and then format. This language is called as "gherkin".*
    ***Given**: Given specifies the pre-conditions.*
    ***When**: Used when some action is to be performed (Instructions)*
    ***Then**: The expected outcome or result should be placed here (Expected Result)*
    And**: And is used to combine two or more same type of action. If you have multiple actions to be performed and want to use Whens or multiple verifications and want to use Thens, the second one can be replaced by an **And
  - **Scenario Outlines -** *Scenario outlines are used when same test(scenario) has to be performed with different data set.*
  - **Tables -** *Tables are arguments to steps usually as input to a Given or as expected output from a Then.*
    Don't confuse tables with scenario outlines - syntactically they are identical, but they have a different purpose. Outlines declare multiple different values for the same scenario, while tables are used to declare multiple values for the same step like a Given/When/Then
  - **Background -** *Whenever any step is required to be performed in each scenario then those steps needs to be placed in Background in other words, common steps for multiple scenarios in a feature file*
- **Step Definition File -***Step Definition File contains the code behind Gherkin. i.e., the actual script associated with the steps in the Feature file. The extension could be .rb or .js file based on if Selenium or Protractor is being used.*

Examples for each section above TBD

We have a habit of writing our BDD at a really low implementation detail level.
Writing BDD in this "imperative steps" style goes against the whole purpose of creating scenarios in the Gherkin style

```
var feature = require('protractor-jasmine-cucumber').feature;

feature( 'LHP - Filter Tiles' )

  .scenario('Verification of  Filter View In SPAR Application')
      .given( 'You go to SPAR home page using the url "https://spar.staging-cauth.factset.com/index.html#/doc/CLIENT:DEVEL-AUTOMATION/report/1"' )
      .when( 'Clicked on Edit icon on any item on the LHP' )
      .then( 'You should be in the Edit mode of SPAR application' )
      .when( 'Return dropdown is clicked' )
      .then( 'The following options should be listed: All, Multi-Statistics, Returns, Multi-Horizon, Quick View, Overview, Returns' )

      .when( 'All is selected from the Reports dropdown and filter is rolling' )
      .then( 'List of Reports should show Rolling Horizon: Beta' )
      .when( 'Overview is selected from the Reports dropdown and filter is still rolling' )
      .then( 'List of Reports should be empty' )
      .when( 'Overview is selected from the Reports dropdown and filter is empty' )
      .then( 'List of Reports should show Cumulative Return' )
      .when( 'Quick View is selected from the Reports dropdown and filter is still empty' )
      .then( 'List of Reports should show Rolling Horizon: Beta, Multi-Statistic Universe Report, Cumulative Report' )
      .when( 'All is selected from the Reports dropdown and filter is Horizon' )
      .then( 'List of Reports should show Multi-Horizon Report: Annualized Return, Rolling Horizon: Beta, Multi-Horizon Report: Annualized Return' )
```

We can write the same scenario above at a more abstract-level, non implementation — like using a more declarative approach.

Example Tables can be a great way to summarize the business logic:

```
var feature = require('protractor-jasmine-cucumber').feature;

feature( 'LHP - Filter Tiles' )

  .scenario( 'Verification of Filter View In SPAR Application' )
    .given( 'You are in Edit mode of SPAR application' )
    .when( 'Return dropdown is clicked' )
    .then( 'The following options should be listed: All, Multi-Statistics, Returns, Multi-Horizon, Quick View, Overview, Returns' )

    .when( 'The following ReportItem from Reports dropdown and filterType is selected, the List of reports should show the following filterResults', [
      { 'ReportItem' : 'All'              , 'filterType' : 'rolling' , 'filterResult' : 'Rolling Horizon: Beta' },
      { 'ReportItem' : 'Overview'         , 'filterType' : 'rolling' , 'filterResult' : '' },
      { 'ReportItem' : 'Overview'         , 'filterType' : ''        , 'filterResult' : 'Cumulative Return' },
      { 'ReportItem' : 'Quick View'       , 'filterType' : '' , 'filterResult' : 'Rolling Horizon: Beta, Multi-Statistic Universe Report, Cumulative Report' },
      { 'ReportItem' : 'All'              , 'filterType' : 'Horizon' , 'filterResult' : 'Multi-Horizon Report: Annualized Return, Rolling Horizon: Beta, Multi-Horizon Report: Annualized Return' }
    ])
```

CUCUMBER FOR SELENIUM/RUBY:
TBD

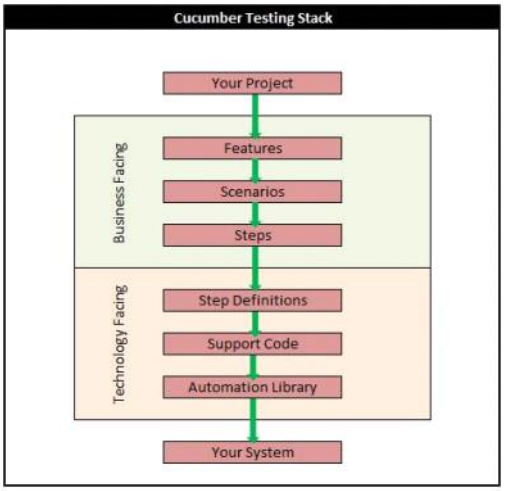
CUCUMBER FOR PROTRACTOR:
TBD

DO's and DON'Ts:

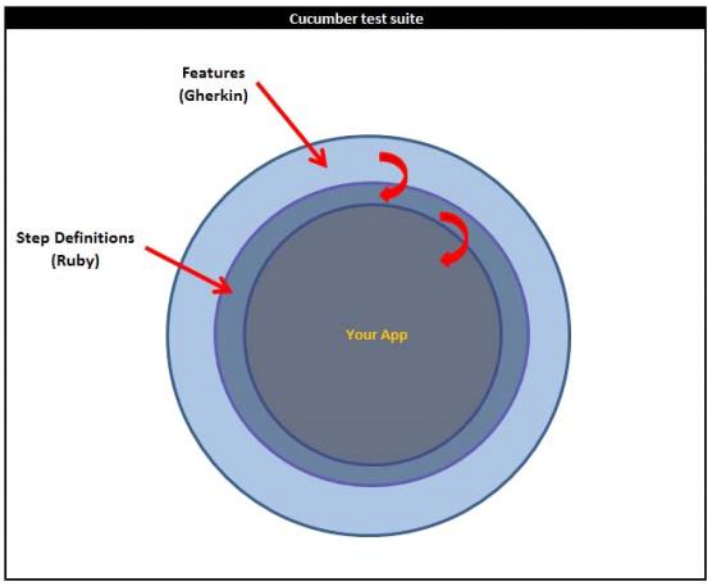
- Don't use special symbols in Feature descriptions.
- Avoid words like Click, Double-click which are low-level details, use English instead
  - ✗ When Clicked on the identifier widget
    And FDS is entered and Enter is hit
    Then……
  - ✓ When the ticker FDS is added
    Then…….
- Don't define the same step in multiple step definition files if they are meant to be run together(multiple test cases in a test plan) - Use of same step in multiple step definitions when the features are run together leads to ambiguous recognition by Cucumber

A. Why Cucumber?

- Behavior-Driven Development (BDD)
- How Cucumber Works?



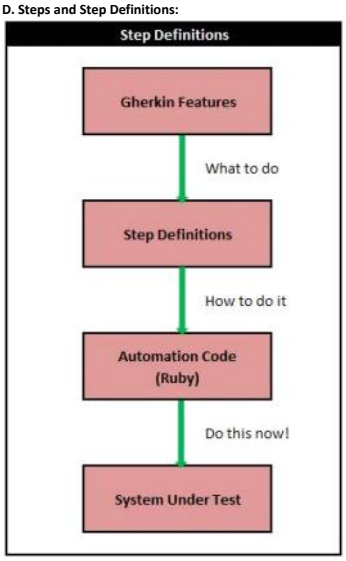
B. Cucumber test suite:



C. Gherkin Basics:

- Feature
- Scenario
- Given, When, Then
- And, But
- Replacing Given/When/Then with Bullets
- Stateless
- Scenario Outline

D. Steps and Step Definitions:



E. Expressive Scenarios:

- Background
- Data Tables
- Nesting Steps
- Doc Strings

OUTSTANDING ISSUES:

- Cucumber for Ruby - Tables -> Special characters get entered when we use tables in feature file. Cucumber seems to take the input from the tables and special characters are getting added with the input when used in the step definition
- Cucumber for Protractor - Scenario Outlines -> Not implemented yet by the Cucumber developer - We have an alternative for this but is slightly in the coding style. Working with the developers for an English-like Scenario Outlines

FAQ (Answers TBD)

1. Where will the hidden information of consolidated steps be stored?
2. Who writes the feature files and step definitions?
3. Where will the feature files be stored?
4. Will the Cucumber logs and the test script logs be integrated?
5. How many scenarios can a feature have?
6. Is there a limit on number of steps a scenario can have?
7. How do we know there are steps(Given/When/Then) that are already implemented?

TAKEAWAYS FROM REVIEWS:

1. Modular feature files – Split independent functionalities/scenarios into separate feature files(Test cases in QAI terms)
2. Input – Use the inputs syntax(double quotes) wherever you feel there cannot be a repetition but has a scope for change eg. FDS updated with IBM
3. Tables – Use tables if you find a pattern or possibility of repetition of a step. Think of ways to consolidate steps into a single When or Then if you need the data sets of the table to be repeated for that particular step. **Tables act only on the step right above it.**
4. Scenario Outlines – Use it **only** in cases where you think the entire scenario (Entire Test case in QAI terms if you consider only one Scenario in one feature file) can be repeated for various data sets
5. Comments – Try using them in-cases where the steps in the feature file is obvious enough but may require an additional tip for a person new to the application. Use # for comments in Feature file

Quick Guide

Outside vendors training doc: http://docs.behat.org/en/latest/guides/1.gherkin.html

CucumberWorkFlow

Monday, March 07, 2016 9:45 AM

CucumberWorkFlow

Monday, March 07, 2016 9:49 AM

Feature File Test Case development Factset (there will be a separate workflow for the Vendor)

1) What a person should know
    a. Need to be familiar with both the versions of feature files
    b. Need to know in advance if the app is angular/non-angular or automated in Ruby or protractor
    c. Will definitely need a way to run their feature file/ to see if the feature file written is in the expected format. Protractor could be challenging for Chris as there is not much documentation
    d. Need a decision on how high level the feature file should be as it could be challenging for new people executing or reviewing (regression/automation) the tests to follow if hidden details are not available
    e. For new test cases created in QAI, should there be a dummy step created that holds the feature file as there is confusion because people are able to create it at the test case level too
    f. Need some way to indicate step id in the feature file or plan for reporting bugs
2) Create/Edit a feature file in QAI
    a. From the widget or Dropdown (TBD in QAI) select Ruby or Protractor
    b. Create the feature file
3) Feature File

    a. Phase 2 - Feature file is stored in the GIT SCM based on the product
4) Scripter development
    a. Phase 1 - automation scripter assigned to that application will place the file in the proper SCM
        i. Scripter will do the test case development on their laptop
    b. Phase2- QAI will send the feature file to the SCM - GIT for that product
5) Scripter Testing non-Tellus JEP
    a. Scripter points their laptop at the Tellus Selenium Grid to do testing
    b. It the test case needs a VM then they will test on a QA Auto single use VM
6) Scripter Testing on Tellus Development Environment
    a. Scripter moves the test case to the development environment in the SCM for testing
7) Scripter Testing on Tellus Production Environment
    a. Scripter moves the test case to the production environment in the SCM for testing

CucumberWorkFlowVendor

Tuesday, March 08, 2016 4:41 AM

Feature File Test Case development Factset - Vendor

1) What a person should know
    a. Need to be familiar with both the versions of feature files
    b. Need to know in advance if the app is angular/non-angular or automated in Ruby or protractor
    c. Will definitely need a way to run their feature file/ to see if the feature file written is in the expected format. Protractor could be challenging for Chris as there is not much documentation
    d. Need a decision on how high level the feature file should be as it could be challenging for new people executing or reviewing(regression/automation) the tests to follow if hidden details are not available
    e. For new test cases created in QAI, should there be a dummy step created that holds the feature file as there is confusion because people are able to create it at the test case level too
    f. Need some way to indicate step id in the feature file or plan for reporting bugs
2) Feature File
    a. Phase 1 - Feature file is stored in a location determined by QAI on a shared drive
    b. Phase 2 - Feature file is stored in the GIT SCM based on the product
3) Vendor Scripter development
    a. Phase 1 - automation scripter assigned to that application will place the file in the proper Vendor SCM
        i. Scripter will do the test case development on their laptop
    b. Phase2- QAI will send the feature file to the SCM - GIT for that product
4) Scripter Testing non-Tellus JEP
    a. Scripter points their Citrix at the Tellus Selenium Grid to do testing
5) Scripter Testing on Tellus Development Environment
    a. FactSet Scripter moves the test case to the development environment in the SCM for testing
6) Scripter Testing on Tellus Production Environment
    a. FactSet Scripter moves the test case to the production environment in the SCM for testing

CucumberMachineDirectoryStructure

Tuesday, March 08, 2016 10:45 AM

From Eyme

We discussed about where the feature files should be placed in GT/Perforce and also about the Tellus project structure once the tests cases and feature files are downloaded onto the VMs.

Project structure on VMs for Protractor:

Tellus
- Features
  - TestCase1.feature
  - TestCase2.feature
- PA3
  - TestCase1.spec.js
  - TestCase2.spec.js
- Pa
- geObjects
- Conf.js

Project structure on VMs for Ruby:

Tellus
- Features
  - TestCase1.feature
  - TestCase2.feature
  - PA3
    - TestCase1.rb
    - TestCase2.rb
- Utils
- PageObjects
- Rakefile

As you see above, the script/step definition level for Ruby is within the Features folder unlike Protractor and this is something Cucumber-Ruby enforces. We are looking into ways on how we can keep that uniform for both Protractor and Ruby.

Feedback

Thursday, March 31, 2016 8:46 AM

LB report of the test cases written in Gherkin (cucumber) http://is.factset.com/core/listbuilderportal/?wknljlkhaa

Regression feedback

General comments:

- Overall I think , the testers should definitely have an idea on the application, know the application related terminologies , dialog boxes and navigations. I sense initially this will be time consuming as testers will need to understand and get accustomed to these Gherkin test cases.
- The feature file and instructions are easy to understand and execute. But most importantly how to navigate/select to a particular path or chart is not described. It will be difficult for regression testers, and they will end up depending on others. Only a senior tester or the one who is familiar with the app can understand and navigate to the appropriate path. In the case of new/less tenured testers, there will be a possibility of executing instructions incorrectly. I agree with [other feedback] that cases are written in Gherkin demand reg testers to come with some knowledge, experience, terminology, and used to applications.
- Test cases written in the Gherkin, will definitely need testers to be aware of the different navigations, application specific terminologies & various dialog boxes related to a particular application. Otherwise it will consume more time to perform a test step. Also, I would like to add a point that this way of executing calls for assumptions as instructions are not explicitly outlined, thus leaving chances for wrong executions.

Good	Bad
http://is.factset.com/justifier/TestManager#/p4/p65/p223/p234/79135/641536 This was easy to follow, but only for the last step I was unable to perform the instructions as I couldn't find how to navigate to that particular link.(Test step Id: 558229).	http://is.factset.com/justifier/TestManager/#/p3645/p4299/p3405/79851/644368 (Bad)- I was unable to follow & execute. Instructions were not written per the application ,I was not getting the desired expected results, as what was given in the instructions was not available in the application. An e.g., it was instructed to click the options dropdown (three line icon) but this was not found anywhere in the app tool bar. Instead a wrench icon was there. Terminologies used are different and it will be important for a tester to be aware of certain application related terms.
http://is.factset.com/justifier/TestManager#/p4/p65/p223/p234/79135/641536 This was easy to follow, but only for the last step I was unable to perform the instructions as I couldn't find how to navigate to that particular link.(Test step Id: 558229).	http://is.factset.com/justifier/TestManager/#/p4/p60/p374/79798/644645 (Bad)- Though the instructions written could be understood, I was unable to meet the expected when followed those instructions as there is a variation in the actual behavior vs the mentioned behavior in the expected. The dialog box given in the expected vs actual output is different due to which the instructions cannot be performed.
http://is.factset.com/justifier/TestManager/ #/p4/p60/p368/79794/644504/958840	http://is.factset.com/justifier/TestManager/#/p4/p65/p223/p234/73664/611327 - Not sure about which application the scenario is about, such cases won't be easy to run. It requires to take a call on the application which needs to be opened. Executing such instructions will be a challenge to the testers.
http://is.factset.com/justifier/TestManager/#/p4/p60/p368/79794/644501	http://is.factset.com/justifier/TestManager/#/p4/p65/p223/p234/79197/641907 - Instructions in this feature file are easy but again involves figuring out navigations.
http://is.factset.com/justifier/TestManager/#/p4/p65/p223/p234/79197/641907	
http://is.factset.com/justifier/TestManager/#/p4/p60/p368/79794/644503 - This feature file was easy to follow. Everything was good about the instructions but there was no navigation on how to launch "Active graph" application & also tester would not be aware how to select "create my own" option. Thus, tester should have minimum knowledge about the application & different options under it while testing a particular application.	

Cucumber Next StepsCucumber Next Steps

Wednesday, April 06, 2016 3:05 PM

For the Cucumber project, these are the next steps from Jay:

"Who, What, Why"

- I've read all your tweaks, questions, comments in the onenote, thanks
- Next step helps you own the vision, and collaborate on crafting both the plan to do it and the next version of the test plan standard
  - I suggest you each supply your favorite Cucumber feature file that your teams have written, in the context of the "who what why" and not the "how" spirit of the vision
  - Describe why this is your favorite example, what challenges remain with the example, if any
  - Iterate on improving to an ideal example
  - Incorporate into the next Standard as the way forward
  - Alongside this we need to create some short and medium term steps to implement the standard globally across the dept

What is our plan?
What changes are needed?

- Education
  - Workflow
  - Who will apply
  - Etc…

Where do we want to start

- Not ready for all to jump in
  - Not being written correctly
  - Expectations that they will not be run correctly by Regression
  - Setting ourselves up for failure

- **What are good candidates to start with**
  - [Alisa] start with tests that are already automated or go straight to Automation, no regression person, no manual testing,
  - [Kate] apps that have a scripter assigned but tests which are not already automated
  - [Dan] same as Kate
  - [Kristin] same as Kate + Agile projects + trained analyst (in cucumber standard format- tbd) , philosophy
  - [Ken] same as Kate + analyst which has already started writing FF, Simple cases (one page)

  - Other notes
    - Should we consider rewriting already automated? [Ken] - yes and no, could be confusing to scripter
    - [Alisa] - already have Automation for an App - can we mix Classic cases with Cucumber classic? (but not in the same plan)
      - [Kate]- yes, but pick more stable apps to start,
      - [Ken] - yes
      - [Jay] - start with apps that have no classic web automation yet, like OA
        - [Kate] - hard to find apps with scripter ready but no automation yet

        - [Kristin] -
        - [Alisa] - they would have to maintain functions for both since cucumber would live in one area separate from classic automation
        - [Dan] - don't mix them

    - Assumptions (consolidated):
        - Only **to be** automated , never to be manual
        - Has a scripter
        - Training available for Analyst and Scripter (and completed)
        - Standard format and philosophy
        - Web only (R&D for pc side code - phase 2??)
        - Classic Web and Cucumber can be mixed - try not to mix, but if we have to - start at test plan level
            - Test updates - are these good candidates to convert - maybe - need to assess - refactor to cucubmer

- **What tools?**
    - Technically
        - Editors with Feature File in QAI
            - Need to support both protractor and ruby (only ruby exists now) - how can we implement something that can scale for more options down the road.
            - Should these be different editors?
            - Ability to write in Jira and push to QAI
        - Relocate FF to source code repository
            - Figure out where QAI should put those files [find RPD]
        - Need Cucumber / GIT directory
        - Framework needs new path for GIT directory
        - SG and FP needs proper versions of cucumber (protractor and ruby)
        - How can we flip back and forth between Step Definition and Feature file?

    - [Ken]
        - to create RPD for technical work next week (4/11)
        - Versioning (QA/stage/live) will work

    - What do we need to get cucumber running in Tellus
        - Standard format and philosophy
        -

- **Plan**
    - Get Feedback from Cygnet - rpds for each topic
        - Research editors
        - Feature Files during training weeks - Kristin's team to get a few identified and listed in an RPD [add category to view]
        - Training
        - Logging
        - What testing tools to integrate - Inventory / Testing apps / Bug tracking
    - Derrick conversation - how to utilize GIT - Ken, Alisa, Me
        - can we add an editor(s) to GIT
        - Is the directory structure used by protractor the same as ruby
    - Ken to mockup Git view/flow
    - List of RPDS already open for Cucumber efforts
    - Schedule next meeting for week of 9th

- **What process?**
  - [Kate]- Training (with both analyst and scripter together)
  - [Ken] - same process for Scripter between Protractor and Ruby

- **Standards**
  - [Kate] Using variables

India - Regression, Analysts, Automation - want to know more, what is BDD, what changes are coming, how will it affect them. They want to move forward. [jay] - ask them to hold up while we figure more out.

6/2 meeting notes

Thursday, June 02, 2016 2:08 PM

(Chris, Alisa, Bryan, Ken, Angela)

1. *What do we need from QAI*

1. Do we want to version Feature files (live, stage, QA, devel)
    a. Not for phase1

2. *Cucumber*:

Ken is working on creating doc (living doc) similar to QAID doc.

- Feature files will be edited in QAI using the Ruby format (still needs testing)
- QAI will push the Feature file to the GIT repository for that product
    - Link to GIT API - https://gitlab.factset.com/help/api/README.md
    - The location will be product(/QA, Devel, Released, mainline)/tests/e2e/*featurefiles*
    - Also inline with *featurefiles* will be folders
        - Associatedfiles (needed other files)
        - Stepdefinitions
        - PageObjects
- In a separate Repo
    - Utilities and Rakefile folder(not versioned)

Action items:

1. Have Sravan start learning about Gherkin
2. Eyme needs to get Protractor-Cucumber-Framework node.js module working. (need ETA)
3. Create a FF for RPD product in Ruby (or reach out to charting group ?)
4. Create RPD GIT repo
5. QAI to use GITlab API to save (use TCID) FF to RPD repo

6/9 notes

Thursday, June 09, 2016 4:18 PM

6/14 Meeting Notes:

Action items:

1. Get Protractor-Cucumber-Framework node.js module working. [Chandana didn't make any progress, Eyme to start next week, need ETA)]
2. Set up meeting with group +Jay
    i. Eng to create repo first, even if code isn't being developed yet.
        1) Sample Git projects
            a) https://gitlab.factset.com/groups/analytics-html-apps
        2) Display repo in QAI (create field in TTM and TFS to map)
3. Do we need a process for new projects.
    i. 3 amigos should talk and create directories.
4. Create a FF for RPD product in Ruby (or reach out to charting group ?) [deferred]
5. QAI to use GITlab API to save (use TCID) FF to RPD repo [differed]
6. TTM
    i. Add field for GIT and TFS Mapping in TTM
7. Notify ENG that QA will be creating e2e directory under each repo

Move Automation from Perforce to Git

6/9 Meeting Notes:

Action items:

1. Get Protractor-Cucumber-Framework node.js module working. [Chandana didn't make any progress, Eyme to start next week, need ETA)]
2. Set up meeting with group +Jay
    i. Eng to create repo first, even if code isn't being developed yet.
        1) Sample Git projects
            a) https://gitlab.factset.com/groups/analytics-html-apps
3. Talk to Eng teams to find out how they use GIT.
4. Do we need a process for new projects. – set up meeting David Seah, Brian Chaiklin
5. Create a FF for RPD product in Ruby (or reach out to charting group ?) [deferred]
6. QAI to use GITlab API to save (use TCID) FF to RPD repo [differed]

6/2 Meeting Notes

Cucumber:

Ken is working on creating doc (living doc) similar to QAID doc.

- Feature files will be edited in QAI using the Ruby format (still needs testing)
- QAI will push the Feature file to the GIT repository for that product
    - Link to GIT API - https://gitlab.factset.com/help/api/README.md
    - The location will be product(/QA, Devel, Released, mainline)/tests/e2e/*featurefiles*
    - Also inline with *featurefiles* will be folders
        - Associatedfiles (needed other files)
        - Stepdefinitions
        - PageObjects
- In a separate Repo
    - Utilities and Rakefile folder(not versioned)

Action items:

1. Have Sravan start learning about Gherkin [done]
2. Get Protractor-Cucumber-Framework node.js module working. [Chandana didn't make any progress, Eyme to start next week, need ETA)]
3. Create a FF for RPD product in Ruby (or reach out to charting group ?) [deferred]
4. Create RPD GIT repo [done - https://gitlab.factset.com/PDIS/RPD]
5. QAI to use GITlab API to save (use TCID) FF to RPD repo [differed]

RPDs & Trello cards

Monday, April 11, 2016 3:12 PM

RPDs

How to display Cucumber Feature Files scenarios in QAI with Version Control - Pasted from <http://is.factset.com/rpd/summary.aspx?messageId=18197835>

Trello

https://trello.com/c/FqGiExNY/60-cucumber-test-cases-in-qai-how-to-display-feature-file-scenarios-in-qai-with-version-control

Video notes

Friday, April 15, 2016 11:01 AM

Video 2 - Your First Scenario (Ruby)

1st specify behavior in FF

Scenario = 1 single aspect
- Given = context
- When = action
- Then = outcome (1 or more)

2nd - Write Step definitions

Milestone is getting to first failed step - never trust a test that doesn't ever have a failure.

Video 3 - Matching Steps (Ruby)

Step definitions - enable readability

Compare Gherkin steps with step definitions = does each scenario have a step definitions

Use regex (wildcards)

Analysts need to understand that capability/flexibility exists

Careful not to be too flexible, be expressive

https://en.wikipedia.org/wiki/Regular_expression

[Jay] Dev community needs to also be trained.

Video 4 - Cleaning up (Ruby)

Cucumber created to bridge communication gap

Drive solution designs from step definitions.

Keeping code and scenarios up to date

Add user story and acceptance criteria (rules), and maybe todo list.

Background keywords can be used to replace repeated Givens

Video 5 - Loops (Ruby)

Excess scenarios -
Scenarios where When and Then are the repeated

Use tabular data tables to make less clunky

Refactor code to improve design of existing code without changing behavior (basic housekeeping)

2 levels of feedback - inner loop of unit tests (build thing right) , out loop of acceptance tests (build the right thing)

Video 6 - Configurations (Ruby)

To run specific scenarios of Feature File using Filters or Tags

Tags - to manage/organize scenarios. (ex: smoke testing scenarios)

Cucumber can generate reports to different formats (html, json, junit, etc)

Video 7 - Details (Ruby) - would be great for analysts

How much detail to use??

Do we want to define some standard tags

Remove incidental details (when scenarios are heavy in details) - irrelevant to scenario.

Several "when" steps suggest testing more than 1 business rule at a time.

Have "Rules" to summarize details of tests

Have "Questions" to answer

Have tests that test one business rule at one time, then consider if needing others that combine.

Video 8 - Problems and Solutions (Ruby)

How to prevent messy FF in first place:

Meetings with all parties to flush out requirements - Assuming ignorance.
Documents user stories and acceptance criteria
Have small user stories
Run sessions in time box of 25 minutes
No need for gherkin scenarios - could slow process down

Problem and Solution domain

Video 9 & 10 - Acceptance Tests vs Unit Tests (Ruby)

[something interesting was said, near beginning - go back and listen]

Acceptance tests are warning lights - what user can't do
Unit test - tells programmers why problem occurred.
Ideally one unit test failure per Acceptance test failure

Mock objects are design tool to hash out ideas. Focus on interaction which is true behavior.

Reorganize unit tests by responsibilities

When refactoring - work in small sections

Video 11 & 12 - Web Automation (Ruby)

Avoid Automation that goes through UI.
Use domain model first gives fast feedback without distractions from web.

Stay focus on core domain helps build more stable core
Helps reduce incidental details.

Majority of tests should be Unit tests.
GUI tests should be small percentage.

Acceptance tests should not be full stack tests. Work with developers to push as many tests to lowest level unit tests.

High-Risk
High-Impact

Page objects to

Feedback

Thursday, March 31, 2016 8:46 AM

LB report of the test cases written in Gherkin (cucumber) http://is.factset.com/core/listbuilderportal/?wknljlkhaa

Regression feedback

General comments:

- Overall I think , the testers should definitely have an idea on the application, know the application related terminologies , dialog boxes and navigations. I sense initially this will be time consuming as testers will need to understand and get accustomed to these Gherkin test cases.
- The feature file and instructions are easy to understand and execute. But most importantly how to navigate/select to a particular path or chart is not described. It will be difficult for regression testers, and they will end up depending on others. Only a senior tester or the one who is familiar with the app can understand and navigate to the appropriate path. In the case of new/less tenured testers, there will be a possibility of executing instructions incorrectly. I agree with [other feedback] that cases are written in Gherkin demand reg testers to come with some knowledge, experience, terminology, and used to applications.
- Test cases written in the Gherkin, will definitely need testers to be aware of the different navigations, application specific terminologies & various dialog boxes related to a particular application. Otherwise it will consume more time to perform a test step. Also, I would like to add a point that this way of executing calls for assumptions as instructions are not explicitly outlined, thus leaving chances for wrong executions.

Good	Bad
http://is.factset.com/justifier/TestManager#/p4/p65/p223/p234/79135/641536 This was easy to follow, but only for the last step I was unable to perform the instructions as I couldn't find how to navigate to that particular link.(Test step Id: 558229).	http://is.factset.com/justifier/TestManager/#/p3645/p4299/p3405/79851/644368 (Bad)- I was unable to follow & execute. Instructions were not written per the application ,I was not getting the desired expected results, as what was given in the instructions was not available in the application. An e.g., it was instructed to click the options dropdown (three line icon) but this was not found anywhere in the app tool bar. Instead a wrench icon was there. Terminologies used are different and it will be important for a tester to be aware of certain application related terms.
http://is.factset.com/justifier/TestManager#/p4/p65/p223/p234/79135/641536 This was easy to follow, but only for the last step I was unable to perform the instructions as I couldn't find how to navigate to that particular link.(Test step Id: 558229).	http://is.factset.com/justifier/TestManager/#/p4/p60/p374/79798/644645 (Bad)- Though the instructions written could be understood, I was unable to meet the expected when followed those instructions as there is a variation in the actual behavior vs the mentioned behavior in the expected. The dialog box given in the expected vs actual output is different due to which the instructions cannot be performed.
http://is.factset.com/justifier/TestManager/ #/p4/p60/p368/79794/644504/958840	http://is.factset.com/justifier/TestManager/#/p4/p65/p223/p234/73664/611327 - Not sure about which application the scenario is about, such cases won't be easy to run. It requires to take a call on the application which needs to be opened. Executing such instructions will be a challenge to the testers.
http://is.factset.com/justifier/TestManager/#/p4/p60/p368/79794/644501	http://is.factset.com/justifier/TestManager/#/p4/p65/p223/p234/79197/641907 - Instructions in this feature file are easy but again involves figuring out navigations.
http://is.factset.com/justifier/TestManager/#/p4/p65/p223/p234/79197/641907	
http://is.factset.com/justifier/TestManager/#/p4/p60/p368/79794/644503 - This feature file was easy to follow. Everything was good about the instructions but there was no navigation on how to launch "Active graph" application & also tester would not be aware how to select "create my own" option. Thus, tester should have minimum knowledge about the application & different options under it while testing a particular application.	

Cucumber Next StepsCucumber Next Steps

Wednesday, April 06, 2016 3:05 PM

For the Cucumber project, these are the next steps from Jay:

"Who, What, Why"

- I've read all your tweaks, questions, comments in the onenote, thanks
- Next step helps you own the vision, and collaborate on crafting both the plan to do it and the next version of the test plan standard
    - I suggest you each supply your favorite Cucumber feature file that your teams have written, in the context of the "who what why" and not the "how" spirit of the vision
    - Describe why this is your favorite example, what challenges remain with the example, if any
    - Iterate on improving to an ideal example
    - Incorporate into the next Standard as the way forward
    - Alongside this we need to create some short and medium term steps to implement the standard globally across the dept

What is our plan?
What changes are needed?

- Education
    - Workflow
    - Who will apply
    - Etc…

Where do we want to start

- Not ready for all to jump in
    - Not being written correctly
    - Expectations that they will not be run correctly by Regression
    - Setting ourselves up for failure

- **What are good candidates to start with**
    - [Alisa] start with tests that are already automated or go straight to Automation, no regression person, no manual testing,
    - [Kate] apps that have a scripter assigned but tests which are not already automated
    - [Dan] same as Kate
    - [Kristin] same as Kate + Agile projects + trained analyst (in cucumber standard format- tbd) , philosophy
    - [Ken] same as Kate + analyst which has already started writing FF, Simple cases (one page)

    - Other notes
        - Should we consider rewriting already automated? [Ken] - yes and no, could be confusing to scripter
        - [Alisa] - already have Automation for an App - can we mix Classic cases with Cucumber classic? (but not in the same plan)
            - [Kate]- yes, but pick more stable apps to start,
            - [Ken] - yes
            - [Jay] - start with apps that have no classic web automation yet, like OA
                - [Kate] - hard to find apps with scripter ready but no automation yet

        - [Kristin] -
        - [Alisa] - they would have to maintain functions for both since cucumber would live in one area separate from classic automation
        - [Dan] - don't mix them

    - Assumptions (consolidated):
        - Only **to be** automated , never to be manual
        - Has a scripter
        - Training available for Analyst and Scripter (and completed)
        - Standard format and philosophy
        - Web only (R&D for pc side code - phase 2??)
        - Classic Web and Cucumber can be mixed - try not to mix, but if we have to - start at test plan level
            - Test updates - are these good candidates to convert - maybe - need to assess - refactor to cucubmer

- **What tools?**
    - Technically
        - Editors with Feature File in QAI
            - Need to support both protractor and ruby (only ruby exists now) - how can we implement something that can scale for more options down the road.
            - Should these be different editors?
            - Ability to write in Jira and push to QAI
        - Relocate FF to source code repository
            - Figure out where QAI should put those files [find RPD]
        - Need Cucumber / GIT directory
        - Framework needs new path for GIT directory
        - SG and FP needs proper versions of cucumber (protractor and ruby)
        - How can we flip back and forth between Step Definition and Feature file?

    - [Ken]
        - to create RPD for technical work next week (4/11)
        - Versioning (QA/stage/live) will work

    - What do we need to get cucumber running in Tellus
        - Standard format and philosophy
        -

- **Plan**
    - Get Feedback from Cygnet - rpds for each topic
        - Research editors
        - Feature Files during training weeks - Kristin's team to get a few identified and listed in an RPD [add category to view]
        - Training
        - Logging
        - What testing tools to integrate - Inventory / Testing apps / Bug tracking
    - Derrick conversation - how to utilize GIT - Ken, Alisa, Me
        - can we add an editor(s) to GIT
        - Is the directory structure used by protractor the same as ruby
    - Ken to mockup Git view/flow
    - List of RPDS already open for Cucumber efforts
    - Schedule next meeting for week of 9th

- **What process?**
  - [Kate]- Training (with both analyst and scripter together)
  - [Ken] - same process for Scripter between Protractor and Ruby

- **Standards**
  - [Kate] Using variables

India - Regression, Analysts, Automation - want to know more, what is BDD, what changes are coming, how will it affect them. They want to move forward. [jay] - ask them to hold up while we figure more out.

6/2 meeting notes

Thursday, June 02, 2016 2:08 PM

(Chris, Alisa, Bryan, Ken, Angela)

1. *What do we need from QAI*

1. Do we want to version Feature files (live, stage, QA, devel)
    a. Not for phase1

2. *Cucumber*:

Ken is working on creating doc (living doc) similar to QAID doc.

- Feature files will be edited in QAI using the Ruby format (still needs testing)
- QAI will push the Feature file to the GIT repository for that product
    - Link to GIT API - https://gitlab.factset.com/help/api/README.md
    - The location will be product(/QA, Devel, Released, mainline)/tests/e2e/*featurefiles*
    - Also inline with *featurefiles* will be folders
        - Associatedfiles (needed other files)
        - Stepdefinitions
        - PageObjects
- In a separate Repo
    - Utilities and Rakefile folder(not versioned)

Action items:

1. Have Sravan start learning about Gherkin
2. Eyme needs to get Protractor-Cucumber-Framework node.js module working. (need ETA)
3. Create a FF for RPD product in Ruby (or reach out to charting group ?)
4. Create RPD GIT repo
5. QAI to use GITlab API to save (use TCID) FF to RPD repo

6/9 notes

Thursday, June 09, 2016 4:18 PM

6/14 Meeting Notes:

Action items:

1. Get Protractor-Cucumber-Framework node.js module working. [Chandana didn't make any progress, Eyme to start next week, need ETA)]
2. Set up meeting with group +Jay
    i. Eng to create repo first, even if code isn't being developed yet.
        1) Sample Git projects
            a) https://gitlab.factset.com/groups/analytics-html-apps
        2) Display repo in QAI (create field in TTM and TFS to map)
3. Do we need a process for new projects.
    i. 3 amigos should talk and create directories.
4. Create a FF for RPD product in Ruby (or reach out to charting group ?) [deferred]
5. QAI to use GITlab API to save (use TCID) FF to RPD repo [differed]
6. TTM
    i. Add field for GIT and TFS Mapping in TTM
7. Notify ENG that QA will be creating e2e directory under each repo

Move Automation from Perforce to Git

6/9 Meeting Notes:

Action items:

1. Get Protractor-Cucumber-Framework node.js module working. [Chandana didn't make any progress, Eyme to start next week, need ETA)]
2. Set up meeting with group +Jay
    i. Eng to create repo first, even if code isn't being developed yet.
        1) Sample Git projects
            a) https://gitlab.factset.com/groups/analytics-html-apps
3. Talk to Eng teams to find out how they use GIT.
4. Do we need a process for new projects. – set up meeting David Seah, Brian Chaiklin
5. Create a FF for RPD product in Ruby (or reach out to charting group ?) [deferred]
6. QAI to use GITlab API to save (use TCID) FF to RPD repo [differed]

6/2 Meeting Notes

Cucumber:

Ken is working on creating doc (living doc) similar to QAID doc.

- Feature files will be edited in QAI using the Ruby format (still needs testing)
- QAI will push the Feature file to the GIT repository for that product
    - Link to GIT API - https://gitlab.factset.com/help/api/README.md
    - The location will be product(/QA, Devel, Released, mainline)/tests/e2e/*featurefiles*
    - Also inline with *featurefiles* will be folders
        - Associatedfiles (needed other files)
        - Stepdefinitions
        - PageObjects
- In a separate Repo
    - Utilities and Rakefile folder(not versioned)

Action items:

1. Have Sravan start learning about Gherkin [done]
2. Get Protractor-Cucumber-Framework node.js module working. [Chandana didn't make any progress, Eyme to start next week, need ETA)]
3. Create a FF for RPD product in Ruby (or reach out to charting group ?) [deferred]
4. Create RPD GIT repo [done - https://gitlab.factset.com/PDIS/RPD]
5. QAI to use GITlab API to save (use TCID) FF to RPD repo [differed]

RPDs & Trello cards

Monday, April 11, 2016 3:12 PM

RPDs

How to display Cucumber Feature Files scenarios in QAI with Version Control - Pasted from <http://is.factset.com/rpd/summary.aspx?messageId=18197835>

Trello

https://trello.com/c/FqGiExNY/60-cucumber-test-cases-in-qai-how-to-display-feature-file-scenarios-in-qai-with-version-control

Video notes

Friday, April 15, 2016 11:01 AM

Video 2 - Your First Scenario (Ruby)

1st specify behavior in FF

Scenario = 1 single aspect
- Given = context
- When = action
- Then = outcome (1 or more)

2nd - Write Step definitions

Milestone is getting to first failed step - never trust a test that doesn't ever have a failure.

Video 3 - Matching Steps (Ruby)

Step definitions - enable readability

Compare Gherkin steps with step definitions = does each scenario have a step definitions

Use regex (wildcards)

Analysts need to understand that capability/flexibility exists

Careful not to be too flexible, be expressive

https://en.wikipedia.org/wiki/Regular_expression

[Jay] Dev community needs to also be trained.

Video 4 - Cleaning up (Ruby)

Cucumber created to bridge communication gap

Drive solution designs from step definitions.

Keeping code and scenarios up to date

Add user story and acceptance criteria (rules), and maybe todo list.

Background keywords can be used to replace repeated Givens

Video 5 - Loops (Ruby)

Excess scenarios -
Scenarios where When and Then are the repeated

Use tabular data tables to make less clunky

Refactor code to improve design of existing code without changing behavior (basic housekeeping)

2 levels of feedback - inner loop of unit tests (build thing right) , out loop of acceptance tests (build the right thing)

Video 6 - Configurations (Ruby)

To run specific scenarios of Feature File using Filters or Tags

Tags - to manage/organize scenarios. (ex: smoke testing scenarios)

Cucumber can generate reports to different formats (html, json, junit, etc)

Video 7 - Details (Ruby) - would be great for analysts

How much detail to use??

Do we want to define some standard tags

Remove incidental details (when scenarios are heavy in details) - irrelevant to scenario.

Several "when" steps suggest testing more than 1 business rule at a time.

Have "Rules" to summarize details of tests

Have "Questions" to answer

Have tests that test one business rule at one time, then consider if needing others that combine.

Video 8 - Problems and Solutions (Ruby)

How to prevent messy FF in first place:

Meetings with all parties to flush out requirements - Assuming ignorance.
Documents user stories and acceptance criteria
Have small user stories
Run sessions in time box of 25 minutes
No need for gherkin scenarios - could slow process down

Problem and Solution domain

Video 9 & 10 - Acceptance Tests vs Unit Tests (Ruby)

[something interesting was said, near beginning - go back and listen]

Acceptance tests are warning lights - what user can't do
Unit test - tells programmers why problem occurred.
Ideally one unit test failure per Acceptance test failure

Mock objects are design tool to hash out ideas. Focus on interaction which is true behavior.

Reorganize unit tests by responsibilities

When refactoring - work in small sections

Video 11 & 12 - Web Automation (Ruby)

Avoid Automation that goes through UI.
Use domain model first gives fast feedback without distractions from web.

Stay focus on core domain helps build more stable core
Helps reduce incidental details.

Majority of tests should be Unit tests.
GUI tests should be small percentage.

Acceptance tests should not be full stack tests. Work with developers to push as many tests to lowest level unit tests.

High-Risk
High-Impact

Page objects to