Scenario Focused Test Design
Presenter

April Ritscher

• Over 13 years experience in Software development
• QA Lead on project described in this paper
What is Scenario Focused Test Design?

SFTD is a methodology for testing applications from a business process and customer usage perspective. The goal is to break application functionality into measurable components that can be assembled into multiple scenarios. This process uses flowcharts to visually represent the user action and application interaction. This results in more in-depth feedback from users and other disciplines during reviews.
Business Value

Customer expectations not met

Move upstream

QA Analysts, Designers & SDET

Sustainable Automation
Business Value

We think:
- On Budget: 89%
- On Time: 89%
- To Spec: 94%

Our customers say:
- On Budget: 70%
- On Time: 56%
- To Spec: 48%
48%

Our customers say we only deliver to spec 48% of the time
Problem Statement

- Business Value
- Proactive Quality

Customer expectations not met

Test Capacity

PM  Dev  Automation  Test  Product

Sustainable Automation

QA Analysts, Designers & SDETs
Capacity

![Bar chart showing relative cost to fix with Requirements = 1.]

- Requirements: 1
- Coding: 6.5
- Test: 15
- Deployment: 80

Legend: Relative Cost to Fix (Requirements = 1)
Problem Statement

Customer expectations not met

Test application meets FS

Proactive Quality

Business Value

Test Capacity

QA Transformation

Sustainable Automation

PM
Dev
Automation
Test

Product
Problem Statement

- Business Value
- Proactive Quality
- Customer expectations not met
- Test Capacity
- Test application meets FS
- Automation Cost
- QA Transformation
- Automation

Requirements
PM
Dev
Automation
Test
Product

PM Dev Test
Requirements
Product

QA
Transformatio
n
Automation
Cost
Goals

- **Business Value**: Product satisfies business needs.
- **Proactive Quality**: Move upstream.
- **SAQ Analysts, Designers & SDETs**: Sustainable Automation.
- **Automation**: Automation.

Additional notes:

- **PM Dev Test**: Requirements.
- **Product**: Framework.
- **Cases**: Suites.
- **QA Transformation**: Prevention.
From Conventional to Innovative

Ensure Solution meets Business Needs

QA Transformation

Creating Proactive Quality (Drive QA Process Upstream)

Increase Manageability of test code

Functional Test Cases
- Scenario Focused Test Design

Automation Framework dependent
- Automation Framework Neutral

One test case per variation
- Data Matrix drives multiple variations per test case
Scenario Focused Test Design

User Scenario

Jane Doe (PID Analyst) starts her work day by creating a new Exhibit. She examines the Exhibit and assigns the Exhibit to herself as she will ultimately sign off on the Sample Group. She examines the lead ID from the Continuity form attached to the sample and searches the records for existing parties that may be associated with the test purchase of the Exhibit. She finds a Party associated with the Exhibit test purchase and records the Party Action (PA) and Lead (PL) on the Continuity form as well as the Target Party information.

Jane Doe examines the Outside Investigator (OI) report for a sales invoice. She identifies the invoice and records the presence of a proof of purchase. She searches the Party records for a source; she then assigns the party found as the sources. She records all her finds and reviews them on the form and secures the finds so no one outside the PID analyst group has access to it.

Jane Doe realizes that the morning is over and she heads out to lunch.
Process Flow

Previous - Linear process

New - Parallel Process

Scenario based Enables Parallel process
What is a Scenario?

- a concrete narrative story told from the customer’s point of view that explains their situation and what they want to achieve
- does not say HOW to solve the problem
- describes the customer’s “dream state”
  - what can they now accomplish that they couldn’t before?
- success criteria allow you to measure and evaluate the quality of the solution
User Scenario

Jane Doe is a system analyst at Microsoft working in Redmond. She needs to create a functional spec for a 2.0 version of her application. This project includes enhancements to existing screens in the current 1.0 application. As she is examining the requirements she realizes that there is an existing area of the 1.0 application that needs to change in order to accommodate the business request. She decides she needs to confirm with the business representative that they want this change implemented. Since her business representative is in China, she needs to be able to send the draft document to her business representative so that they can collaborate on the design. Since they are working in different time zones, they may need to send multiple revisions back and forth before they settle on a final design.

Use Case
65% of the population are Visual learners
Visio

- On a single tab shapes should be **numbered** in proper sequence so that the pseudo code is generated in meaningful order.
- **User Action\Step:** Used to describe an action performed by the user of the application. Equivalent to a step in a test case.
- **System Action\Step:** Used any time there is a System action or response to the users Action. Equivalent to step level Expected results or Validation Step. Many User actions will be directly followed by a System response.
- **Conditional attributes** – All the decision statements need to be followed by the proper if-else construct.
- **Dynamic Connector:** Used to connect steps in logical order
- **Off-page Reference:** Used to point to another tab in Visio.
- **Terminator:** Used any time there is a step or action that ends the current path of a function\feature.

All the **tabs** should be named so that the generated pseudo code outlines clear and unambiguous path to be followed for the entire module.

**Main path** as well alternate paths must be included in the Flow. (Main Green, Alternate White)
Putting it together

- Visio diagrams are the building blocks
- Putting them together forms the scenarios
- These could be rearranged to recreate different scenarios
Test Suites & Test Cases

- User scenarios equates with test suites (1:1)
- Use Cases equates with test cases (1-∞)
- Data Validation through data matrix
Shape Meta-Data

- **Main Flow**: Used to distinguish whether step is part of the main flow for the selected function. This value is used to determine data graphic color.

- **Function**: Name of the test function that tab represents. Used to group steps by Function in Pseudo-code. Since all shapes on a given tab will have the same function name.

- **Conditional**: Conditional is used to provide hints to automation engineers on location of If\Then blocks and provide information on which step to go to if it is not the next chronological Step.

- **Shape Number**: Use Visio add-on to number shapes within functions to determine order in pseudo-code.

### Table

<table>
<thead>
<tr>
<th>Main Flow?</th>
<th>Function</th>
<th>Conditional</th>
<th>Shape Number</th>
<th>Shape Number Text</th>
<th>Hide Shape Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Flow</td>
<td>1.1 Select File for Upload</td>
<td>Else(Enter filename or Browse)</td>
<td>4</td>
<td></td>
<td>FALSE</td>
</tr>
</tbody>
</table>
# Pseudo-Code

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>File upload Main</td>
<td>Open File upload Application</td>
</tr>
<tr>
<td>2</td>
<td>Display File upload application with focus in filename field</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>User selects action to perform</td>
<td>Main Flow</td>
</tr>
<tr>
<td>4</td>
<td>If(Action)</td>
<td>Cancel</td>
</tr>
<tr>
<td>5</td>
<td>Go to step 1 of Function= 2 Cancel</td>
<td>Alternate Flow</td>
</tr>
<tr>
<td>6</td>
<td>If(Action)</td>
<td>Select File for upload</td>
</tr>
<tr>
<td>7</td>
<td>Go to step 1 of function=1.1 Select File for Upload</td>
<td>Select File</td>
</tr>
<tr>
<td>8</td>
<td>Else(Action)</td>
<td>Upload</td>
</tr>
<tr>
<td>9</td>
<td>Go to step 1 of Function=1.2 Upload process</td>
<td>Upload Process</td>
</tr>
</tbody>
</table>
C#-Code

```csharp
public override void ExecuteWF()
{
    foreach(DataVariation dv in CurrentTest.DataContext.DataVariations)
    {
        CurrentDataVariation = dv;
        Execute(false, null);
    }
}

internal void Execute(bool isMainPath, SampleGroupHubVM sgHubVm)
{
    if (null == sgHubVm)
    {
        sgHubVm = new SampleGroupHubVM();
        sgHubVm.OpenEMISHomeFaceWithSGHub();
        CurrentTest.EnqueueCallback(() =>
            {
                CurrentTest.Logger.LogMessage("Executing SG-CreateNew test");
                CurrentTest.AddStep("Check default state of Create New button");
            });
        EntityCommandSampleGroup sgCommands = sgHubVm.LoadEntityCommandSampleGroup(false);
        CurrentTest.EnqueueCallback(() =>
            {
                bool state = sgCommands.OnCreateNewCommandCanExecute(null);
                if (!isMainPath)
                {
                    {
                        if (state != true)
                            throw new Exception("Create New button should be enabled");
                        LogMessage("Create New button is enabled as expected");
                    }
                    else
                    {
                        if (state == true)
                            throw new Exception("Create new button is enabled for user role : " + CurrentTest.CurrentUserRole.ToString());
                        LogMessage("Create New button is disabled as expected");
                        // Test finished here for this variation
                        CurrentDataVariation.DataVariationComplete = true;
                    }
                }
            });
    }
    Click Create New Button
    Verify that Details tab is loaded by default
}
```
Possible Path
1: New Sample Group, No Sample
2. Existing Sample Group, Existing Sample
3: Existing Sample Group, No Sample
Data Matrix flow

1. Create Sample Group
   - Create Sample
     - With One Sample
       - Random Sample
         - Last (max) Sample
     - Random Sample
   - Last (max) Sample

2. Select First (min) sample Group
   - With One Sample
     - Random Sample
       - Last (max) Sample
   - With One Sample

3. Select Last (max) Sample Group
   - With One Sample
     - Random Sample
       - Last (max) Sample
   - Create Sample
Data Matrix Approach

- **Data-driven testing** is the creation of interacting test scripts together with their related data results in a framework used for the methodology. In this framework, variables are used for both input values and output verification values: navigation through the program, reading of the data sources, and logging of test status and information are all coded in the test script. The logic executed in the script is dictated by the data values.

- **Keyword-driven testing** is similar except that the test case is contained in the set of data values and not embedded or "hard-coded" in the test script itself. The script is simply a "driver" (or delivery mechanism) for the data that is held in the data source.
Existing Data Matrix Examples

<DV Function DV_Function="00 Sample" >
  <DV DV_SG_DatavariationID="" DV_VariationDescription="Min Samplegroup with Max SampleID"
      DV_Decision_Shape="(NeworExisting)" DV_Decision_Branch="New" NeworExisting="Existing">
    <SG SampleGroupID="SQL:SELECT SG.SamplegroupID FROM AntiPiracyOLTP.EM.SampleGroup SG LEFT
      JOIN AntiPiracyOLTP.EM.Sample S ON SG.SampleGroupID = S.SampleGroupID WHERE SG.SampleGroupID =
      (SELECT Min(SampleGroupID) FROM AntiPiracyOLTP.EM.SampleGroup) and S.SampleID = (Select
      MAX(SampleID) from AntiPiracyOLTP.EM.Sample where samplegroupid = S.SampleGroupID )" />
  </DV>

  <DV DV_SG_DatavariationID="" DV_VariationDescription="MAX Samplegroup with Random SampleID"
      DV_Decision_Shape="(NeworExisting)" DV_Decision_Branch="New" NeworExisting="Existing">
    <SG SampleGroupID="SQL:SELECT SG.SamplegroupID FROM AntiPiracyOLTP.EM.SampleGroup SG LEFT
      JOIN AntiPiracyOLTP.EM.Sample S ON SG.SampleGroupID = S.SampleGroupID WHERE SG.SampleGroupID =
      (SELECT MAX(SampleGroupID) FROM AntiPiracyOLTP.EM.SampleGroup) and S.SampleID = (Select TOP 1
      SampleID FROM AntiPiracyOLTP.EM.Sample where samplegroupid = S.SampleGroupID ORDER BY NEWID())"
      />
  </DV>

  <DV DV_SG_DatavariationID="" DV_VariationDescription="Min Samplegroup with Random SampleID"
      DV_Decision_Shape="(NeworExisting)" DV_Decision_Branch="New" NeworExisting="Existing">
    <SG SampleGroupID="SQL:SELECT SG.SamplegroupID FROM AntiPiracyOLTP.EM.SampleGroup SG LEFT
      JOIN AntiPiracyOLTP.EM.Sample S ON SG.SampleGroupID = S.SampleGroupID WHERE SG.SampleGroupID =
      (SELECT Min(SampleGroupID) FROM AntiPiracyOLTP.EM.SampleGroup) and S.SampleID = (Select TOP 1
      SampleID FROM AntiPiracyOLTP.EM.Sample where samplegroupid = S.SampleGroupID ORDER BY NEWID())"
      />
  </DV>
</DV_Function>
Generated Data Matrix Examples

<DV DV_SG_DataVariationID="" DV_VariationDescription="SampleGroup with sample details but no components"
    DV_Decision_Shape="(Sample data exists)"
    DV_Decision_Branch="Yes" NeworExisting="Both">
    <SG SampleGroupID="" State="Submitted" SG_Finding="Random"
        Generated_Finding="Random" PIDAnalyst="Random" AnalysisType="Random"
        Method="Random" TPType="Random" AcquisitionNotes="Random">
        <Sample SampleType="Random" DatereceivedbyMS="Null" Analystreceiveddate="Null"
            productDivision="Null" ProductName="Null" Language="Null" DistributionType="Null"
            ExactUnitsAquired_Bit="True" TotalUnitsAcquired="Random" ProductNotes="Null"
            PriceperUnit="Null" PriceperunitLocalcurrency="Null" PriceperunitCurrency="Null"
            PriceperunitUSD="Null" priceperSystem="Null" PricepersystemLocalcurrency="Null"
            PricepersystemCurrency="Null" PricepersystemUSD="Null" PricingNotes="Null" SKU="Null"
            SKUDescription="Null" QualityAnalyzed="Null" SampleFinding="Null" Characteristics="Null"
            AnalysisNotes="Null">
            </Sample>
    </SG>
</DV>
Benefits of testing Scenarios.....

scenarios focus entire team on the priorities

• design phase
  • identify design gaps early
  • stimulate discussion to find feature requirements

• during coding
  • discover cross-feature and cross-app (ETE testing) bugs earlier
  • ensure team is fixing scenario blocking bugs

• at major milestones
  • verify scenario implementation is complete
ROI

- Increase test automation:
  - Pre 56% Manual 44% Automated
  - Post 28% Manual 72% Automated
- $150K saving in vendor cost
  - By changing from SDET 3 to SDET 1
- 4000 Hours of savings in test case maintenance cost
  - By reducing test cases from over 1048 to 128
- 38% reduction of defect cost in stabilization phase
  - By finding defects in build phase saving 5.2 weeks of person work on one drop, equivalent to $18K/per drop
Lessons for Others

- Use User scenarios as a base for all testing
  - Much higher customer Satisfaction on projects using Scenario focused test design. No Surprises during UAT
- Visually represent your test cases
  - First project to have 100% test cases reviewed
- Move QA Process Upstream
  - Able to execute automation code before receiving official drop. On average found 10 bugs per drop.
- Extract data variations into data Matrix
  - Reduces Automation maintenance cost
Q&A