Risk Measurement for the Real (and Imperfect) World

Darryl Nicholson
ContactDarrylNicholson@gmail.com

Agenda

- Introduction
- Context / Background
- The Problem
- Scenarios & Calibration
- Scenario Lifecycle
- Deliverable
- Questions

Introduction

- Who am I and why am I here?
- Risk and Regression Testing
  - Calibration of test plans
  - Minimalistic approach to deliver sw quickly
  - Our methods for Risk management
  - Designed to drive revenue
  - Fights natural instincts to be policeman/gatekeepers
- MRS – Minimum Regression Set
  - Our implementation of Code coverage
  - Controversial

Background / Context

- SaaS Environment
  - Our clients dictate schedules to sell services we build
  - Hybrid SOA Production environment
  - Process Billions of $$ in payments
  - We are built for speed; wired for changes.

- Speed to Market is key
  - Caution doesn't pay the bills
  - Compensation comes from driving revenue
  - Cost to fix a Production bug is roughly equal to QA

The Problem

- Continuous Test Case Growth
  - Customer review cycles and feedback
  - New Clients & New Features
  - Innovation in our product portfolio
  - SOA enhancements that magnify the test problem
  - Production test escapes

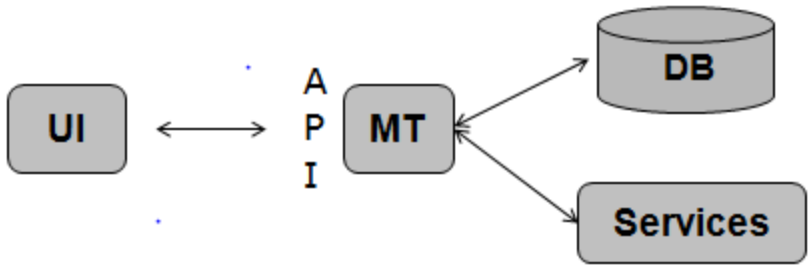
The Problem

- Result: Continuous test case growth in an unstructured quasi-subjective manner.
- Regression testing burden grows.
  - Each new release cycle needs additional time and/or resources to complete
  - Project Managers, Business Executives, Marketing and Customers never like this answer
- Not sustainable nor scalable

The Solution

- We chose to instrument our test cases using Code coverage techniques
- Resulting test case set from this analysis is the “Minimum Regression Set” (MRS)
- MRS easily maps to requirements, use cases, feature definitions, etc. All artifacts easily understood by key stakeholders.

Environment



- UI : User Interface Layer
- MT: Middle Tier (Java)
- DB: DataBase

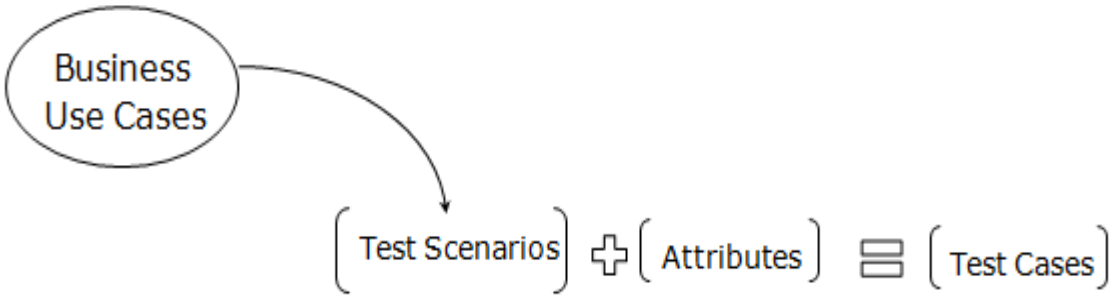
- Engineering team drives API & Code coverage unit tests with Cobertura
- Engineering has an extensive set of Unit tests that drive MT API's but do not include the UI
- All feature complete QA releases have an instrumented MT.

Test Scenarios

- Our clients tend to describe changes in terms of business use cases, marketing ideas or product delivery strategies rather than traditional software requirements.
- Client definition, in whatever form it arrives, is used to describe “Test Scenarios”
- Segregate out the test case data and refer to these elements as “Attributes”.

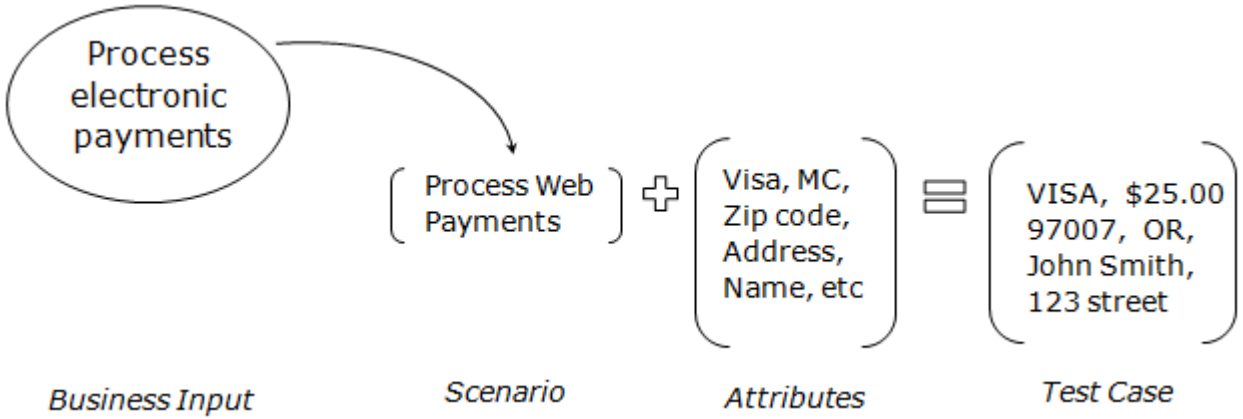
Test Scenarios

- Process looks like this:



- Example: process credit card transactions from all states for different amounts and payment methods

Test Scenarios



- A typical review for one of our web products will create 700-900 Scenarios.
- Creates Joint Ownership

- Are all defined Scenarios truly needed ?

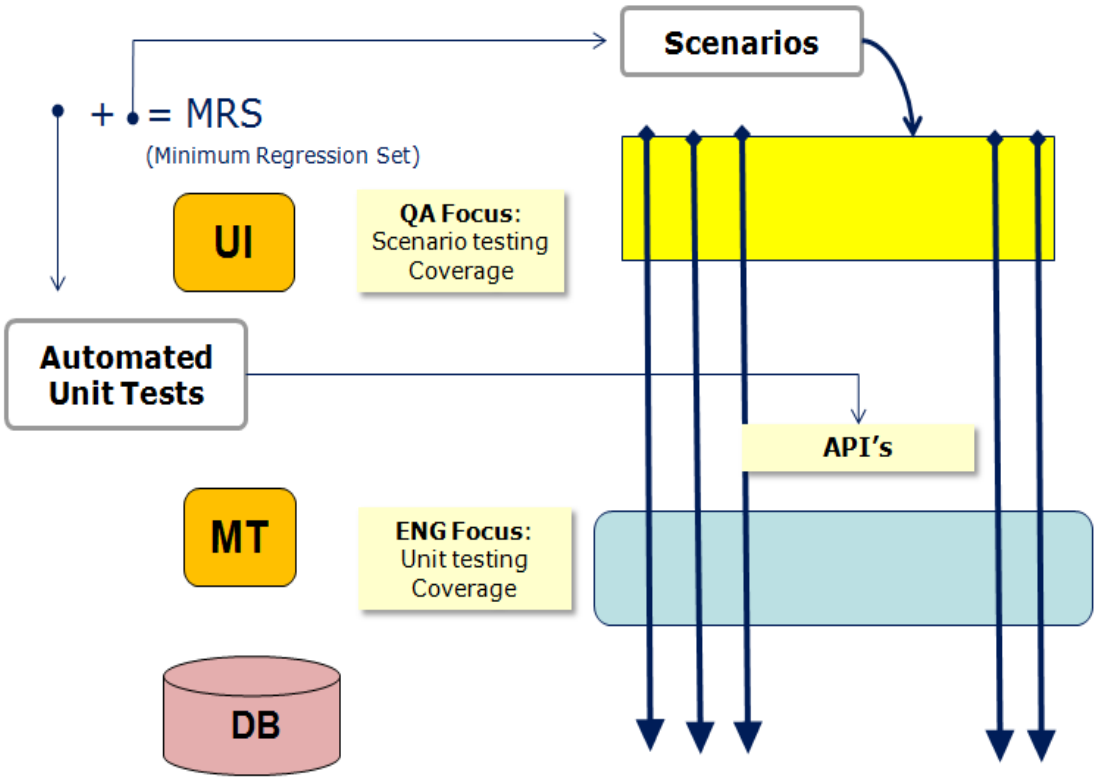
Test Calibration

- Test Calibration is the process by which we create an MRS from the large set of Scenarios

- Classify in 3 categories:
  - Cat 1: **The MRS.** Single Scenario that exercises a unique code path, is repeatable and measured
  - Cat 2: A scenario that does not add code path uniqueness but adds unique data sets based on attributes
  - Cat 3: A scenario that has neither code path uniqueness nor adds unique attribute data.

Test Calibration

- MRS Definition of Category 1
- Instrumented MT-JAR file in the System Under Test
- Run each scenario to increase code coverage



Cobertura Code Coverage

Coverage Report - All Packages

Package	# Classes	Line Coverage		Branch Coverage		Complexity
All Packages	55	75%	1625/2179	64%	472/738	2.319
net.sourceforge.cobertura.ant	11	52%	170/330	43%	40/94	1.848
net.sourceforge.cobertura.check	3	0%	0/150	0%	0/76	2.429
net.sourceforge.cobertura.coveragedata	13	N/A	N/A	N/A	N/A	2.277
net.sourceforge.cobertura.instrument	10	90%	460/510	75%	123/164	1.854
net.sourceforge.cobertura.merge	1	86%	30/35	88%	14/16	5.5
net.sourceforge.cobertura.reporting	3	87%	116/134	80%	43/54	2.882
net.sourceforge.cobertura.reporting.html	4	91%	475/523	77%	156/202	4.444
net.sourceforge.cobertura.reporting.html.files	1	87%	39/45	62%	5/8	4.5
net.sourceforge.cobertura.reporting.xml	1	100%	155/155	95%	21/22	1.524
net.sourceforge.cobertura.util	9	60%	175/291	69%	70/102	2.892
someotherpackage	1	83%	5/6	N/A	N/A	1.2

- Example from Cobertura home page

- Simply run Scenarios and verify coverage is increasing
- Goals: 100% API & code coverage.

MRS Findings

- Generally after execution of approximately a third of the defined Scenarios, the code coverage needle will stop incrementing far short of 100% coverage.
- This is the moment where we realize that the Scenarios analysis done as an intellectual exercise has missed a number of valid cases.
- Validation of the method!

MRS Findings

- Typically what is missed and overlooked:
  - the error handling routines
  - obscure use cases
  - available functionality that was not obvious at review or “Snuck in”
- When running with code coverage enabled, these potential test escapes are very obvious.

User Interface White Space

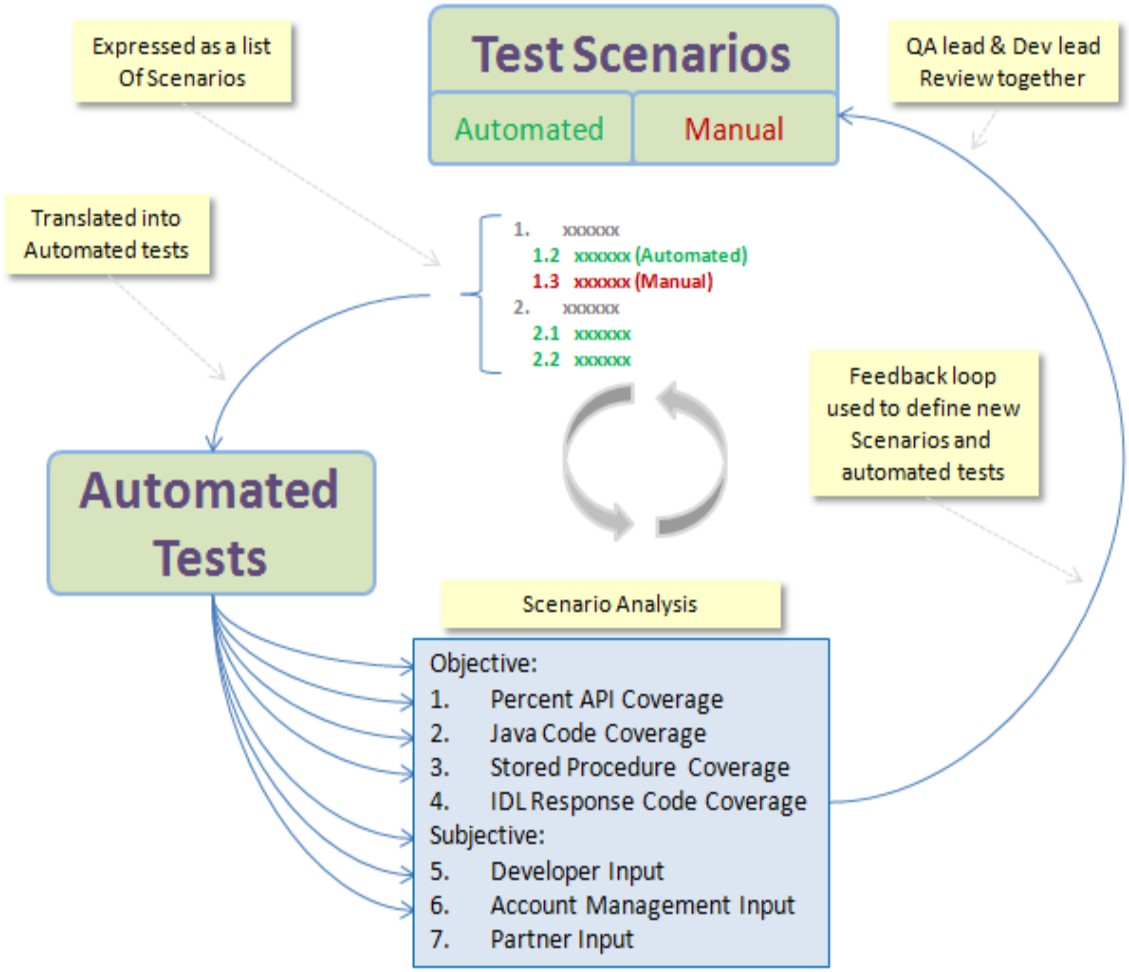
- After MRS is defined, a final UI Code review is required
- The White space is the UI code structures not measured since their scope is entirely in the UI Framework

- Examples: JQuery elements, Analytic web tags, form validation logic

- These are manually added to the MRS

MRS Lifecycle

- Feedback loop
- Catch “feature Creep”
- Iterative and keeps conversation flowing



Test Escapes

- They happen. Root cause expressed as an MRS

- In our system, test escapes are generally:
  - Automated test failure
  - MRS Definition inaccuracy (missed)
  - White Space analysis incorrect
  - Scenario not executed
- First 3 = MRS additions
- 4th Case is the price of too much speed & Risk

The Deliverable

- We live in an imperfect world.
- Accept – Deliver code with the “Sun & Moon alignment method”
- If we “Have to …” when QA has not finished testing then QA has a simple message for the team MRS = 45%.

Questions ?