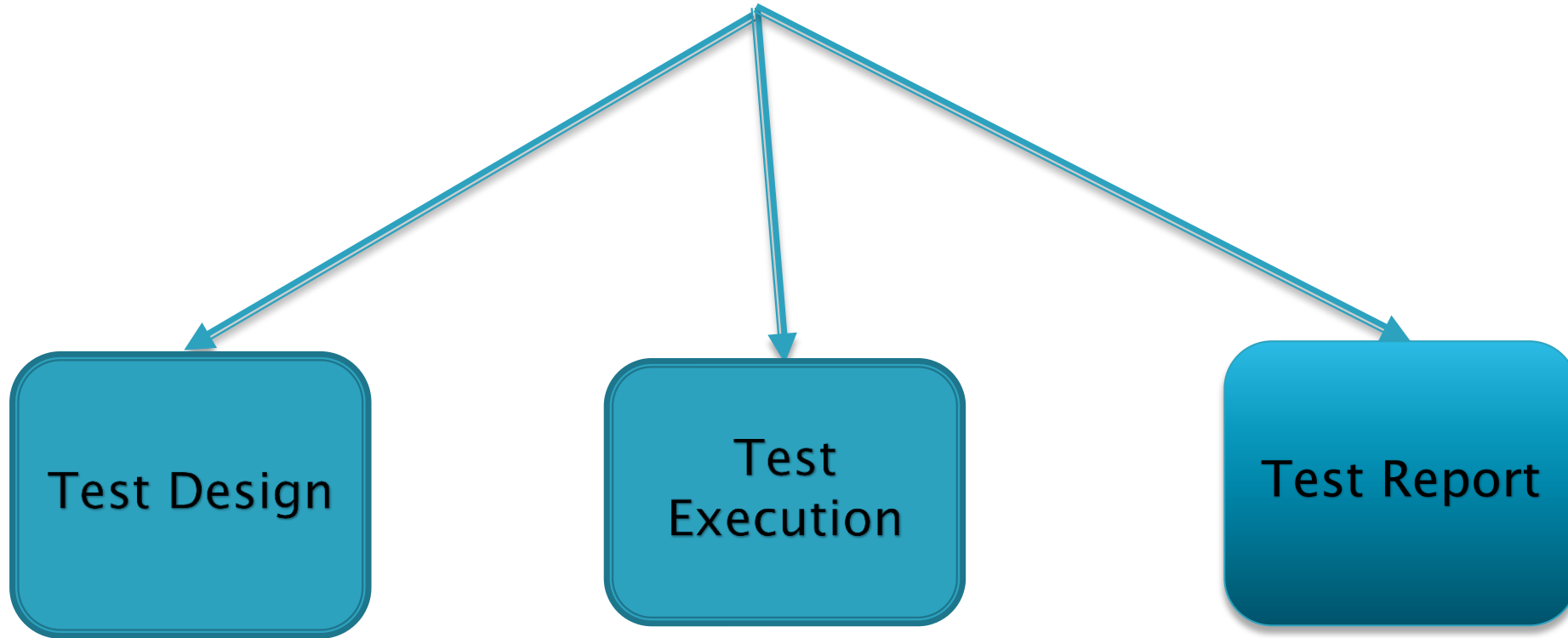


“Test Design” in “Test Automation”

- Deliver the effective “Test Design”

Jenny Jian

Test Automation



Success in automation is not as much as a technical challenge as it is a test design challenge.

– Hans Buwalda

Reality of “Test Design”



Test Case Template



Test Case Template

Test Case Document

Project ID:
Author of Test Cases:
Date of Creation:

Project Name:
Version No:
Date of Release:

Test Case ID	Req No	Version Number	Type of Test Case	Test Case Name	Action	Expected	Cycle #1			
							Actual	Status	Bug ID	Remarks
TC001										
M01 Module Name										
M01TC001										
SM01 Sub-Module Name										
M01SM01TC001										

Source from <https://strongqa.com>

Retrospective

An Agile retrospective is a meeting that's held at the end of an iteration in Agile Software Development. During the retrospective, the team reflects on what happened in the iteration and identifies actions for improvement going forward.

<https://searchsoftwarequality.techtarget.com/definition/Agile-retrospective>

What kind of problems have you experience
during
Test Design?

What are your two biggest pain-points?

Reality/Situation – 1

Vague documentation written in Natural English, BA and Developers have a lack of skill to write precise requirement

“Why documentation is often unusable”
–David L. Parnas, Software Fundamentals

Leads to False Positives

Prose Requirement – Example

- Drive safely according to traffic light.
- Update Parameter A, B, C after receiving message when car is in driving mode
- Do A, B, C according to EngineOilLevel
- Support xxx feature

Solutions – 1

Work with BA/Developers to use the below methods to write the precise requirement

- Table Format
- Flow Chart
- State Diagram

Table Function – 1

Conditions	Results
Traffic_Light_Green	Drive
Traffic_Light_Yellow	Slow down
Traffic_Light_Red	Stop

If Traffic_Light_Green
then Drive

Elseif Traffic_Light_Yellow
then Slow_down

Else Traffic_Light_Red
then Stop

Table Function – 2

Condition		Result
Conditions	Sub_Conditions	
Engine_on	EngineOilLevel_low	Do A
	EngineOilLevel_Med	Do B
	EngineOilLevel_High	Do C
Engine_off		Do D

If (Engine_on) and (EngineOilLevel_low)
then do A

elseif (Engine_on) and (EngineOilLevel_Med)
then do B

elseif (Engine_on) and (EngineOilLevel_High)
then do C

Else
then do D

In which adjoining cells are interpreted as being “anded”.

State Diagram

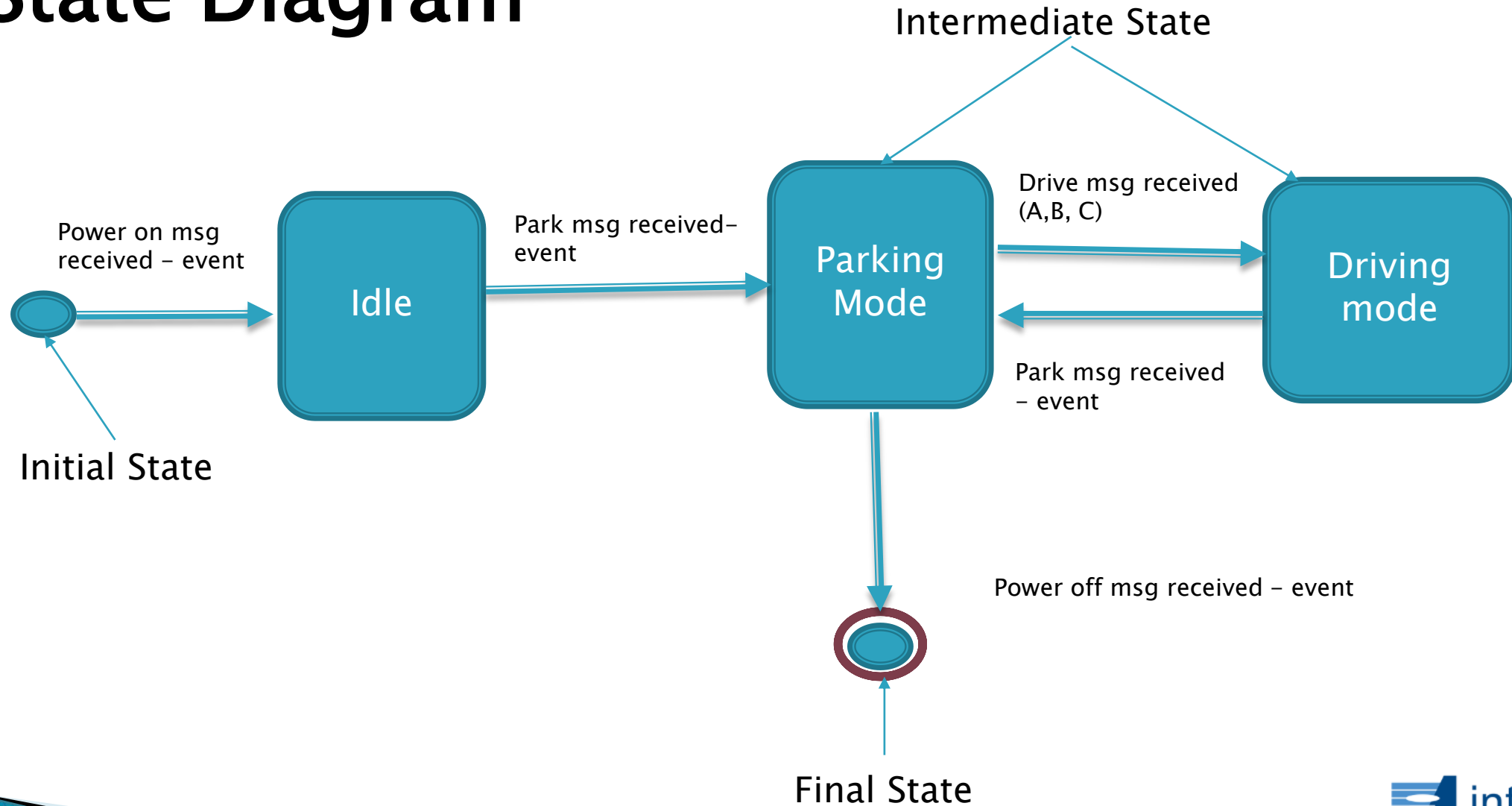


Table Function for Driving Mode

Parameter A	Parameter B	Parameter C	Driving Mode	
			Upon Receiving xxx msg	
			Sub_State_1	Sub_State_2
mismatching	mismatching	mismatching		
mismatching	mismatching	match		
mismatching	match	mismatching		
mismatching	match	match		

Reality/Situation – 2

Traceability from the automation tests to the requirements require manual effort and goes stale fast.

Reality/Situation – 3

Test cases are written manually and fail to keep up with the speed of agile.

Reality/Situation - 4

Automated Tool which generated test cases from requirement is not well accepted by Industry yet.

People aren't aware of them, there are tools available in the market.

People are not trained well to use it effectively.

Solutions – 2

Documentation-based testing tools, using table function that represent the specification of a program's behavior can be used:

- To generate test cases for that program
- To evaluate test coverage for that program

Comformiq Tools

The screenshot displays the Comformiq software interface, which is used for modeling and testing business processes. The main workspace shows an activity diagram for a car booking process. The diagram starts with a 'Register Request' activity, followed by a 'Merge' activity. The flow then goes to a 'Review Request' activity, which leads to a decision diamond labeled 'Approved?'. If approved, the process moves to 'Request Picked' and then to a final state 'Failure'. If not approved, it goes to 'Allocate Car', which leads to another decision diamond 'Allocated?'. If allocated, the process moves to 'Deliver' and then to a final state 'Success'. If not allocated, it loops back to the 'Merge' activity.

Below the diagram, there is a 'Requirements' table and a 'Properties' panel for the 'Review Request' activity.

Name	Priority	In model	Priority in model
Ad-hoc requirements			
Ensure car available		✓	
Evaluate requests		✓	
Deliver car		✓	
Reject invalid req		✓	

The 'Properties' panel for 'Review Request' shows the following details:

- Name: Review Request
- Business Actions:
 - When: Review Request is performed where request is <request>
 - Then: Review Request completes with outcome <outcome>
 - Do: Evaluate requests
 - Requirement: Evaluate requests
 - Priority: <click to specify>
- Parameters:
 - Input: request
 - <click to add input parameter>
 - Output: outcome
 - <click to add output parameter>
 - Combinators: <click to add new output parameter> <click to specify>

- We need to raise awareness of the requirement formalization approaches and related tools
- We need to learn how to think about formalism in our test design
- We need to understand the cost/benefit tradeoff, the price of the tools vs. the increased quality or decreased effort/time for testing.

Summary

It is crucial decision to determine
what to automate, how to automate,
Or
should use any tool to automate
“test design” process.

Tools are expensive and team needs to be trained to use it effectively. Even though without the tool, if developers, BA and testers can improve their capability to communicate, it will help to deliver effective “Test Design”.

Reference:

- Parnas on why documentation is often unusable – except from
“A Rational Design Process: How and Why to Fake It”
- Wise Integrity, SCRTTool Release Information & Introduction
- A. John van Schouwen, The A-7 Requirements Model
- David L. Parnas, Software Fundamentals
- <https://www.conformiq.com/resources/getting-started/>