Revealing the Mystery Behind Test Automation Framework Design

CQAA Webinar
August 27, 2014

Dean Carvin
Solutions Architect Manager
Checkpoint Technologies
Email: dcarvin@checkpointech.com

About Checkpoint Technologies

- Incorporated in January, 2003
- Experts in Quality Assurance and Quality Control!
- Testing Specialization include; Functional, Performance, Security and Mobile. *Manual, Automated and Scriptless!*
- HP Software Gold Partner & Authorized Training Partner
- HP Authorized Software Support Partner
- QAI Training Partner

Three Learning Objectives

- Learn the fundamental elements of a successful automation framework
- Understand the pros and cons of a complex framework design in contrast to a simple design
- Learn practical tips to design and implement an automation framework in your organization

Fundamentals of a Successful Framework

- Definition, Goals, Characteristics & Common Elements

Automation Framework Definition

- Definition: An automation framework is a test automation methodology that separates the automation code from the data allowing for separate development and maintenance of the two assets.

Goals of a Well Designed Framework

- Decrease automated test creation time. Tests can be created before the framework is ready.
- Decrease maintenance
- Increase test automation coverage and the overall impact of automation on the testing effort
- Put the power of automation in the hands of the manual testers

Automation Framework Characteristics

- Modular Code – There should be no duplication of code
- Scalable – Little effort should be required to added business processes or new AUTs to the framework
- Error Handling – All unexpected application errors are managed by using exception handlers.
- Reliable – Designed to be robust and reliable to the point of having scripts run unattended.
- Data is managed externally to the code

Common Elements of Successful Frameworks

- **Simple front end** – The “front end” is the mechanism for creating the test scripts.
    - Know your audience/customer – Who will be using the framework to design the automated scripts?
        - Do they understand objects, classes, methods, etc.?
        - Do they understand test automation methodology?
        - Can they be trained on how to use the framework in 8 hours or less?

Common Elements of Successful Frameworks, cont.

- Data input should be done via a well known tool i.e. Excel, simple database, XML.
- The rules or syntax required to create the automated tests need to be easily understood and conveyed
- Incorporate simple keywords such as <Clear> to clear the content of a field or <Today+30> to enter a date 30 days in the future

Common Elements of Successful Frameworks, cont.

- **Scalable and maintainable back end** – The "Back End" consists of the code in the form of drivers, process routines, functions, recovery routines, etc. that read the data, execute the test and handles everything possible to ensure the most successful test run.
  - The code must be:
    - Modular in design
    - Well documented
    - Reporting needs to be detailed, clear & concise
    - Simple enough so that a master's degree in software development is not required for maintenance and support

Common Elements of Successful Frameworks, cont.

- Multiple applications and even multiple platforms need to be taken into consideration
  - A framework should be flexible enough to accommodate most of your organization’s testing projects
  - Know the scope and expectations of automation in your organization
- Data is the KEY!!!
  - The framework must be driven by DATA!
  - Data can be in the form of input data and/or Keywords.
  - Must allow for multiple data iterations

Fundamentals of a Successful Framework

- Simple Vs. Complex Design

Business Process Testing Framework
Simple vs. Complex

Complex Framework Pros	Complex Framework Cons
Better scalability	More time to develop
Typically less framework maintenance required	More senior resources required
	Overall more expensive to develop
	Typically more technical knowledge is required to use the front end

Business Process Testing Framework
Simple vs. Complex

Simple Framework Pros	Simple Framework Cons
Much less time to develop	Typically more maintenance required
Easier to troubleshoot and maintain	Less scalable – Typically requires more coding/scripting to onboard a new business process or a new application

Automation Framework Case Study – Complex

- Real-World Example

Case Study – Complex
Client Overview

- Diversified holding company providing financial services
- Over 6,200 financial advisers serving 2.5 million client
- Recent merger and acquisition by client created one of the country’s largest full-service wealth management and investment banking firms

Case Study – Complex Framework Requirements

- Must work seamlessly with multiple technologies (Web, VB/ActiveX, Infragistics Controls)
- Must handle regression on 5 unique, critical applications
- Must execute on multiple machines simultaneously in VM environment without assistance of a test management tool
- Test results must be traced to and reported in QC manual test that the automated script is replacing

Case Study – Complex Framework Requirements (cont.)

- Script design must be simple enough for a "non-technical" person to create
- Scripts must be designed without the designer having access to the AUT
- Screenshots must be saved for all steps as objective evidence of step completion

Case Study – Complex
Team Composition

- 1 automation lead
- 4 senior automation engineers to design the framework
- 4 junior to midlevel automation engineers to design the shared object repositories and the test scripts in Excel based solely on documented manual scripts

Case Study – Complex Framework Design Features

- Built using VBScript in HP-QuickTest Pro 11.00
- Key-word driven based on object class & action
  - Used a scripting dictionary to map object class & action combination to function that executed the step
  - Custom object classes could be defined so the automation tool could reliably interact with custom objects
- Effectively performs across applications developed in Web, .Net, VB and custom Infragistics controls
- Test Results updated in HP Quality Center automatically

Case Study – Complex Framework Design Features (cont.)

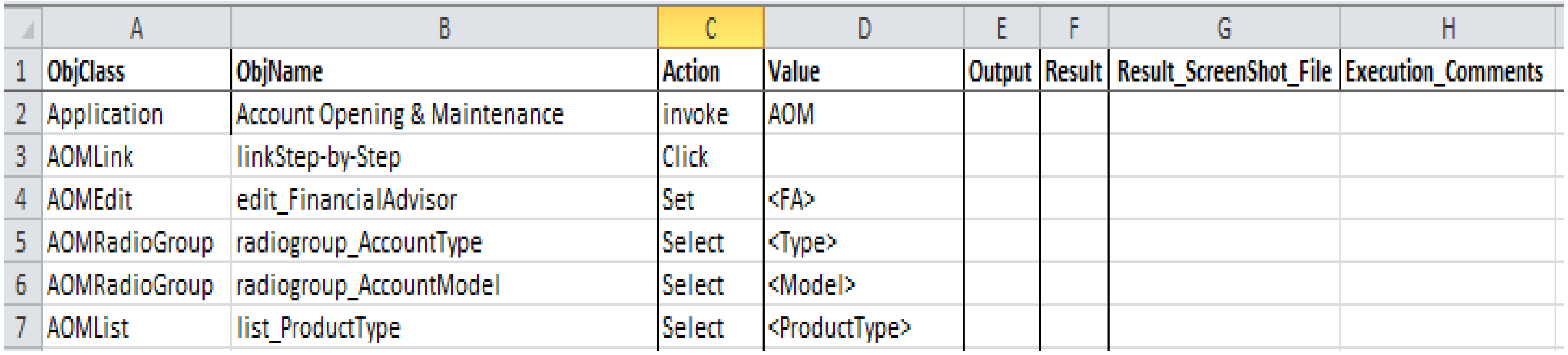
- Supports simultaneous execution with multiple machines utilizing a load balancing feature
- Requires no technical skills to create and maintain automated scripts
- Supports multiple sets of data (data-driving), providing unique results for each set of data
- Potential to work beyond the initial effort to act as a reliable regression testing framework

Case Study – Complex
Front End Design

- The automated scripts use Excel with simple syntax
- No object repository is required for Web applications. (Will find object based on object class and object name given in the test script.)
- The data can vary in such a way as to ignore steps which are <u>not</u> relevant to the current iteration of data

Case Study – Complex Test Script Sample

- The front-end used Excel with two sheets – "Test Steps" & "Data"



	A	B	C	D	E	F	G	H
1	ObjClass	ObjName	Action	Value	Output	Result	Result_ScreenShot_File	Execution_Comments
2	Application	Account Opening & Maintenance	invoke	AOM				
3	AOMLink	linkStep-by-Step	Click					
4	AOMEdit	edit_FinancialAdvisor	Set	<FA>				
5	AOMRadioGroup	radiogroup_AccountType	Select	<Type>				
6	AOMRadioGroup	radiogroup_AccountModel	Select	<Model>				
7	AOMList	list_ProductType	Select	<ProductType>				

Automation Framework Case Study – Simple

- Real-World Example

Case Study – Simple Framework Requirements

- Only three web-based applications in the initial scope
- Very limited resources were available to design and implement the framework (One senior engineer)
- Junior automation resource needed to be mentored throughout the project
- Off-shore automation resources needed to be engaged
- Front-end needed to be simple enough for manual testers to use for their everyday testing needs

Case Study – Simple Framework Design Features

- Built using VBScript in HP Unified Functional Test (UFT) 11.5
- Key-word & data-driven based on the AUT business process(*Key-words*) and parameters(*Data*)
- Parameter Types:
  - Input – Data to be entered into edit fields and drop-down lists
  - Verification – Expected values to be compared to actual values during execution
  - Decision/Action – Creates an action against an object
  - Output parameter – Place holders for data captured from the AUT during execution

Case Study – Simple Framework Design Features cont.

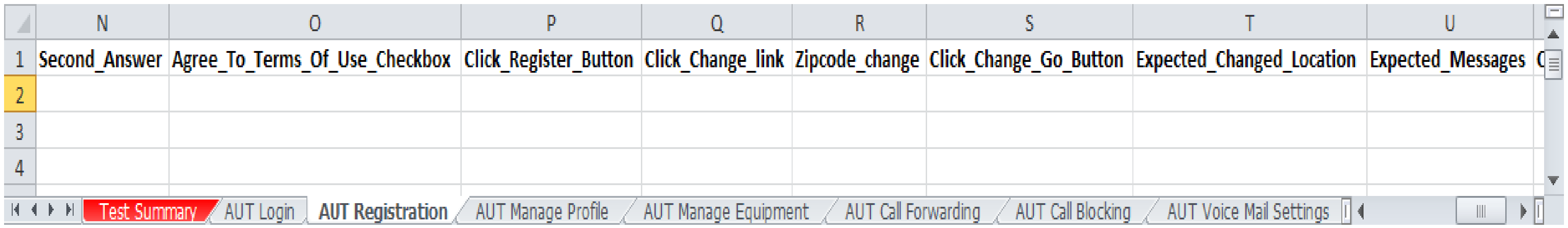
- Low-level common functions handle all the complexities of the detailed data management, object identification and manipulation
- Reporting is filtered through **one function** which captures screen shots, writes detailed messages to the test results and handles critical failures stopping test execution

Case Study – Simple Front-End Design

- The front-end utilizes Excel for script design
  - Each tab name represents the name of a business process
  - Each BP tab contains all the necessary parameters to create both positive and negative test scenarios
  - Output is captured to Output Parameters and can be used in subsequent iterations by using a Excel formula
- Requires No technical skills to build.
- Multiple AUTs can be easily incorporated into an integration test by adding tabs associated with other AUTs

Case Study – Simple Front-End Design

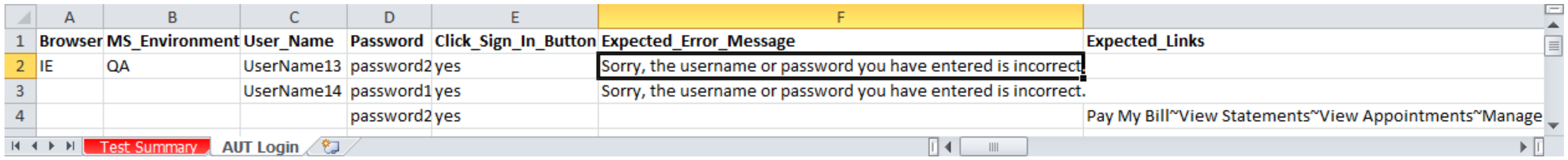
- A Master Excel Template contains all the business process for a given application.



	N	O	P	Q	R	S	T	U
1	Second_Answer	Agree_To_Terms_Of_Use_Checkbox	Click_Register_Button	Click_Change_link	Zipcode_change	Click_Change_Go_Button	Expected_Changed_Location	Expected_Messages
2								
3								
4								

Test Summary | AUT Login | AUT Registration | AUT Manage Profile | AUT Manage Equipment | AUT Call Forwarding | AUT Call Blocking | AUT Voice Mail Settings

- To create a new test, simply save the Master Template as another file, remove the tabs not required for the test and reorder in the order they will execute, then apply the data.



	A	B	C	D	E	F	
1	Browser	MS_Environment	User_Name	Password	Click_Sign_In_Button	Expected_Error_Message	Expected_Links
2	IE	QA	UserName13	password2	yes	Sorry, the username or password you have entered is incorrect	
3			UserName14	password1	yes	Sorry, the username or password you have entered is incorrect.	
4				password2	yes		Pay My Bill~View Statements~View Appointments~Manage

Test Summary | AUT Login

Case Study – Simple
Back-End Design

- Master driver script controls the flow of the script based on the data.
  - Calls AUT specific driver scripts which in turn calls process-specific groups of steps (actions) that contain all the steps required to complete **ALL** of the business process (BP) needs both positive and negative

```
'Evaluate if the current sheet (action). If data exists in the current iteration the sub-driver will be called
 If ExecuteAction(LocalActionExcelObject,objActionWorkSheet,Environment("IterationRow")) Then
    'Call the application specific parent action based on the current sheet name prefix
    Select Case Ucase(sheetNamePrefix)
        Case "MS"
            RunAction "My Services Driver [My Services Driver Script]", oneIteration
        Case "ATIM"
            RunAction "ATIM Driver Script Action [ATIM Driver Script]", oneIteration
        Case "CRM"
            RunAction "CRM Driver Script Action [CRM Driver Script]", oneIteration
        Case Else
            Reporter.ReportEvent micFail,"Process " & sheetName & " is not yet accounted…
    End Select
 End If 'End If ExecuteAction(LocalActionExc
```

Case Study – Simple Back-End Design

- Code in the form of actions are mostly devoid of logic and only contain simple steps referencing the object with the related parameterized data

```
Browser("Networks").Page("Networks").WebElement("Register for a My Services").VerifyExist "Register Page"

Browser("Networks").Page("Networks").WebEdit("Email Address").Set Parameter("Email_Address")
Browser("Networks").Page("Networks").WebEdit("Re-Enter Email Address").Set Parameter("ReEnter_Email_Address")
Browser("Networks").Page("Networks").WebEdit("Account Number").Set Parameter("Account_Number")
Browser("Networks").Page("Networks").WebEdit("Customer Code").Set Parameter("Customer_Code")
Browser("Networks").Page("Networks").WebEdit("Username").Set Parameter("Username")
Browser("Networks").Page("Networks").WebEdit("Choose Password").Set Parameter("Choose_Password")
Browser("Networks").Page("Networks").WebEdit("Question 1").Set Parameter("First_Security_Question")
Browser("Networks").Page("Networks").WebEdit("Your Answer 1").Set Parameter("First_Answer")
Browser("Networks").Page("Networks").WebEdit("Your Answer 2").Set Parameter("Second_Answer")
Browser("Networks").Page("Networks").WebCheckBox("Iagree").Set Parameter("Agree_To_Terms_Checkbox")
Browser("Networks").Page("Networks").Link("Register").Click Parameter("Click_Register_Button")

Browser("Bright House Networks").Page("Bright House Networks").VerifyMessages Parameter("Expected_Messages")

Browser("Bright House Networks").Page("Bright House Networks").ClickLinkOnPage Parameter("Link_To_Click")
```

Simple Framework – The Planning Process

- Determine the front-end design before starting on the back-end!
  - Who will be tasked with using the framework to design the automated tests?
  - In what format will the tests be designed? What tool will manage the data? (Excel, DB, XML, etc.)
  - Utilize Input, Output, Decision and Verification parameters
  - What is the syntax or rules governing the data input?
    - “<Clear>” – Clears the data in a field or expects the field to be empty for verification
    - “<Today+30>” – Adds 30 days to the current date

Simple Framework – The Planning Process (cont.)

- Build the back-end to support the front-end
  - How does the data get fed into the framework?
  - How does runtime data get captured and reused?
  - How are multiple iterations handled?
  - What type and level of reporting is created – tool generated and/or custom? (screenshots)
  - How are unexpected events handled?

Simple Framework – The Design Process–

- Master Driver Script
  - Copies the Input File to the Execution folder
  - For each row of data, each sheet is evaluated whether or not there is data in the current row.
  - Reads the sheet name and calls the appropriate Application Driver
- Application Driver
  - The Application Driver calls the automated business process related to the name of the current sheet
- Business Processes
  - Each step is evaluated for execution based on the data
  - Report steps are written to the test results

Thank you!

Author
Dean Carvin
Solutions Architect Manager
Checkpoint Technologies, Inc.
Email: dcarvin@checkpointech.com

Questions & Answers

- "Q: Have you ever created a framework that would do a query search for the test data so the automation is truly touchless?
- A: I personally have not done this. Many of the same framework principles would apply. My only concern with this method of extracting data is how predictable the results will be. Since in a data-driven FW environment the data drives the execution, your queries would have to be VERY refined.
- Q: maximum how many iterations can we have for business process testing?
- A: For those who do not know, Business Process Testing is a test automation methodology that is uses both HP’s UFT and ALM in conjunction (I assume that is what you are talking about). It can be considered a framework to an extent but still requires a good bit of low level functions to make it very robust. I had a colleague give it a try and he got to 321 iterations with no problem. We could not find any documentation on that question but I would be surprised if you find the limit. Let me know if you do.

- Q: How much time did you spend testing your complex framework? i.e. verifying that there were no false positive or false negative results?
- A: That is a tough question to answer directly. We tested every aspect of the framework from the object interaction to the flow of data to the reporting, etc. As we found issues and areas for improvement we made the modifications. I am a strong proponent of taking screenshots for both pass and fail conditions. It is always good to have objective evidence to back up your results. It is safe to say that once we got the basics in place it was a couple months until we could say it was ready for ALL applications.
- Q: Are you a proponent of masking the implementation of a system from the test? Your screenshot for the complex framework frontend required knowledge of the AUT UI. In your experience, have you found benefit in hiding the UI – i.e. just provide the data to enter and not specifically say which UI component it goes to
- A: I do not fully understand your question unless you are talking about API testing. If you are talking about API testing then absolutely. As far as GUI testing, it is best to simulated manual interaction as much as possible using the automation tool. For screenshots, just take screenshots of the entire desktop. That way you don't miss something peripheral happening that may be impacting your execution.

- Q: how does qtp compare to microsoft test manager or robot framework?
- A: I have not used all these tools so I can’t speak to the question personally. There are many sites online that compare test tools. The challenge is to find an objective site. I would recommend getting trial versions and trying them all out to see what works best for you and your environment.
- Q: are there more detailed examples online of the drivers you've described here? so that I could build a POC relatively quickly? (like in a two weeks)
- A: The only pre-built framework I tried that we found online was so complicated it wasn’t worth trying to implement and it was only for Web. The front-end it supported was also not what we were looking for. It is more than just the driver that needs to be developed. There is significant design in the form of functions that handle the variations of data in the background. But this should be no problem for someone with 5+ years of experience in automation design.

Below is a simple function that handles selecting an item from a list object. This can be a model for others.

```
Public Function SelectList(ListObj,itemToSelect)
    If itemToSelect = "" Then
        Exit Function
    End If
    'If the parameter = "<blank>" then change it to "#0"
    If Lcase(itemToSelect) = "<blank>" Then
        itemToSelect = "#0"
    End If

    On Error Resume Next
    'Get the object name from the OR to report to the test results
    objName = ListObj.ToString
    If Not ListObj.Exist(1) Then
        ReportEvent micFail, objName & " does not exist.","Cannot set '" & itemToSelect & "'"
        Exit Function
    End If
    'Select the item in the list
    ListObj.Select itemToSelect
    'If there is an error report it.
    If err.number <> 0 Then
        ReportEvent micFail,objName & " select",itemToSelect & " - " & err.Description
        err.clear
    Else
        ReportEvent micDone,objName & " select", itemToSelect
    End If
End Functions
```

- Q: Can using nightly regression tests be run easliy over multiple environments DEV, INT, QA, Prod
- A: This can be done but define "easily". If you are not using a strong test management tool that can schedule runs to start over night and point the execution to different machines then this will be extremely challenging. As far as the tests themselves being able to run on multiple environments, simply parameterize the environment and change the url or executable in the login routine depending on what the value is. The other challenging aspect is setting up and maintaining the systems to support the testing. It is best to use systems separate from the manual testing if possible.
- Q: Can one tests iterations call multiple web browsers for testing over different versions?
- A: Again the browser would be parameterized and the correct browser would be invoked. You could have separate tests for each browser type or have multiple iteration in the same test; each iteration pointing to a different browser. Your question seems to ask if one test iteration can execute on multiple browsers. For the Login routine the parameters would be "Browser", "Environment", "UserName", "Password", "Click_Login_Button"
  - Login Parameters for Multiple Browsers & Environmenst

Browser	Environment	UserName	Password	Click_Login_Button
IE	QA	Username1	Password1	Yes
IE	DEV	Username1	Password1	Yes
FireFox	QA	Username1	Password1	Yes

- Q: What are the challenges with VMs? We are trying to establish a lab of VMs for running test automation but teams claim they are very slow and unusable. Any tips on how to effectively utilize VMs
- A: VMs are great for manual testing but not so much for automation! Where I am currently working we experience about a 40% reduction in speed running against the VM environment. And this is not unique. Everywhere I go the results are similar. I am not a VM expert as far as how to set them up but I have had some VM "experts" that were not able to get the performance up to par.
- At a client a few years ago they bought 4 high end machines to sit under our desk area (they barely fit) that we maintained to suite our automation needs. Using these machines compared to running on our laptops we were about to DECREASE the execution time of 350+ tests by over 20%. We were able to get the final results and get the report out hours earlier each week. So consider me old-school on this subject.