xensight

Publication Date: 12 July 2010

Governance - Test Automation Project Plan

By Randy Raymond

Winston Churchill once said "He who fails to plan is planning to fail" and this statement could not be more true when it comes to test automation. Too often companies start automation wanting quick results then experience less than desired benefits. Much of that less than desired benefit is created by the lack of planning and governance of the automation project.

The Internet is full of resources that tell you that you need to plan your automation projects. There is a scarcity of examples that make planning simple. This paper provides an actual real-world example of a test automation project plan to help you get started with governance of your automation development efforts which leads to a successful automation project.

Engineers and project teams new to automation will benefit from having expert guidance from the beginning. Veteran automation engineers will appreciate the adherence to the fundamentals of automation, project management, and software development.

Scenario in the Example

The example project plan contained in this paper is for a fictitious company named WorldCo that has an internal application called WinBig. An offsite contracting firm was hired by WorldCo to develop a test automation framework and test automation suite.

After developing the automation framework and test suite the offsite contracting firm will transfer development and maintenance responsibilities for both assets to WorldCo. This strategy could easily work for an offshore to onshore arrangement where the assets are developed offshore then transferred to the onshore customer.

1. INTRODUCTION

1.1. Project Overview

The WinBig project is WorldCo's next step toward increasing test automation for all products at WorldCo. A consulting firm, has been engaged to provide the design architecture for a repeatable and portable test suite that contains both a solid framework from which to expand automation and test scripts for testing the WinBig application.

1.2. Objectives

There are several objectives of the project. All are designed to support these WorldCo strategic business goals:

- Reducing costs for overall testing
- Increase product quality on all products
- Expand automation to more of Pearson's products

1.3. Acronyms

Term	Description
LADC	Los Angeles Development Center
AUT	Application Under Test
SUT	System Under Test or Software Under Test
GUI	Graphical User Interface
SME	Subject Matter Expert
BA	Business Analyst
SVP	Software Verification Plan

1.4. Referenced Documents

Title	Document
WinBig Functional Specification Document (FSD)	WB_FSD_Introduction.doc

Use Cases 1 thru 26	26 different documents
WinBig Acceptance Test Plan	WB_ATP.doc
WinBig System Product Features Summary	WinBig_Features.doc

1.5. Limitations

The limitation of this first pilot project is that most of the cost of start-up in creating the reusable library framework will be incurred during the project. This project will build the infrastructure that will significantly reduce scripting and costs of future projects.

Other first time costs will be in the learning curve to manage central repository Object Map and refinement of the configuration management process for making test suites available for the team at WorldCo.

2. Scope

The following activities have been identified as in-scope and out-of-scope for the WinBig Test Automation Architecture Project.

2.1. In Scope

In scope activities for the Project are broken down into these different areas:

- Framework Standards and Procedures
- Framework Implementation
- Test Scripts
- Metrics Definition
- Smoke Test Plan
- Test Environment Requirements
- Script Development Environment Definition
- Decompose Use Cases to Specifics That Can be Automated

2.1.1. Framework Standards and Procedures

Maintainability is crucial to the continued success of an automated test strategy. These components represent software development best practices as applied to automated test suite development. These deliverables help ensure the test automation suite is maintainable on a long-term basis, expandable, and repeatable from release to release.

Deliverable	Description
Metrics & Measures	Definition of the metrics and measures that will be used to monitor the effectiveness of test automation
Automation Requirements and Goals	Test automation requirements and goals are stated are documented to keep the project team working toward the same results. These dovetail with metrics and measures to objectively evaluate progress towards meeting the business goals and technical requirements
Framework, Script, and Test Data Configuration Management Plan(s)	Written plan for managing framework library code, script source code, and test data as a single coordinated test suite related to specific releases of the AUT
Script Code Writing Standards	Establishes consistency within code and the code of all team members. Consistency leads to code that is easier to understand, which means it is easier to develop and to maintain while reducing reduces the overall cost of the test suites. New team members get up-to-speed faster with less learning curve
Test Data Standards	Establish consistency for the construction, management, storage, and reuse of test data
Test Environment Requirements	Documents the environment including equipment, software, connectivity, setup, and tear-down required to script tests and execute the test cases and test suites. This may be an individual document or a major section in an individual test plan. See SVP section 2.3
Script Development Environment Definition	Documents the workstations, software, and other tools necessary to write and execute automated test scripts
Smoke Test Plan (Automation)	Test plan specifically written to determine what Use Cases and functionality will be automated for WinBig and how the automated test will be conducted. This document will be similar to the WinBig Acceptance Test Plan except it will be for the functionality that is to be automated. Smoke Tests exercise the most critical functionality of an application to determine if the AUT is ready for acceptance into general testing and represents a mini-regression test. The larger regression test can be built from the foundation created in the Smoke Test

2.1.2. Framework Implementation

Framework implementation deliverables are the reusable, common code library routines written to the published code standards and managed by the configuration management plan. Framework code is essentially functions similar to the test tool's programming language that allow script developers to program independent of the AUT/SUT user interface.

The framework library scripts written for WinBig will have utility in several applications shortening the time to develop automated test suites for those applications. The goal of the framework implementation is to create a library that becomes a fast script creation warehouse. In its ultimate and ideal form, creating scripts for test cases ideally becomes calling existing functions in a particular order.

The following functionality will be coded as framework library components:

Functionality	Description
Error trapping / handling	Generalized error trapping and handling functionality to be used in all scripts to gracefully recover from application errors cause by code defects, operating system issues, or other unknown sources
Error logging	General functionality used within the error trapping / handling routines to log error to file, screen captures, or other test control applications
Exception handling	General functionality that handles page errors such as 404 page not found and other unexpected events. These may retry or reset to a condition before invoking error logging.
Data access for test data	General functionality for reading, adding, editing, deleting test data in from files for data-driven tests. Separate strategies for CSV and Excel files
Application launch	General function to launch the application in a browser and navigate to the opening page or other designated pages
Application navigation	General functions for basic navigation through the application where hyperlinks, command buttons, or menus are used.
Populate to or Read data from controls	Wrappers for the most-used standard controls that put data into or select / de-select a control. Wrappers for the most-used standard controls that retrieve the content or state of controls
Pre-set / Reset conditions	General functions to set data in a database or establish often-used application states required by other scripts. Reset the application and application data (restore an entire database? Yes, if necessary) to a known state in order to be ready for the next test
Post-test verification conditions	General functions to verify data in a database or other files where output is produced

Dialog handlers	General functions to handle dialogs that are used or appear in an application. Examples are File Save, File Open, Print, Warnings with OK/Cancel buttons, etc.
GUI Map Utilities	To the extent the selected automation tool will allow, write functions to dynamically load or change objects in the automation tool data store GUI Object Map

The list can grow time permitting in the Pilot Project.

2.1.3. Test Scripts

The test scripts are the code that are specific to the WinBig user interface or backend components and that validate the functionality being tested manually with the scripts derived from the Use Cases. The test scripts use the library code provided by the framework and have code specific to the WinBig user interface or backend. The test scripts will be data-driven to the extent possible depending on the requirements of the individual test case.

Several things may occur that will cause a Use Case to be de-selected for automation:

- Functionality was implemented with custom controls, custom objects, and/or custom widgets and requires too much effort to develop automated test scripts. Depending on the technology there may not be a work-around for the automated test tool
- The particular functionality of the application that may be too unstable to script. Ideally, automation only should take place after a script has been executed successfully manually
- The functionality may be too difficult to script in the time allowed
- Test data is not available, cannot be created, or does not exist

Guidance for automated script development will come from the same scripts used for Functional Test, Subsystem Test, and Integration Test types as described in the Software Verification Plan.

2.1.4. Metrics & Measures Definition

Metrics and measures will be defined that assist management to determine the effectiveness of the automated tests once automated testing commences against the application under test. Metrics and measures will be proposed by the Test Automation Architect and approved by WorldCo project and senior management.

2.1.5. Automated Smoke Test Plan

The Automated Smoke Test Plan will specify which of the critical functionality of WinBigs will be automated. This automation will become the first scripting that uses the framework to create application specific test scripts. The Smoke Test is a miniature regression test in that it only tests critical functionality required to determine if the AUT is ready to accept into testing.

2.2. Out of Scope

The following activities are out of scope for the Project.

- Writing generic wrappers for the command language of the scripting tool that isn't specifically needed to support the scripts on WinBig
- Writing generic wrappers for the workarounds that may be required to handle custom controls, custom objects, custom widgets, or other application features not directly and easily recognized by the scripting tool AND that may be too difficult and/or time-consuming to create during the Pilot Project
- Scripting against any application other than WinBig
- Scripting against non-Internet Explorer browsers.

3. Project Plan

3.1. Use Case Dependencies

The Use Cases have dependencies on other Use Cases in that some of the Use Cases perform preparatory work for others. A certain set of Use Cases must be automated first or we must manage the data states to support latter Use Cases. There is also an order that the Use Cases must be performed.

The choice is to manage the order Use Cases are automated or manage the data states in the database to facilitate out-of-order Use Case automation and execution.

The automation project plan will be to automate the Use Cases in order of dependency. Once the core Use Cases have been automated then the high-value Use Cases can be selected for automation.

3.1.1. Dependency Collateral Benefit

The collateral benefit of automating the Use Cases according to dependency is the first Use Cases are small, non-complicated, and provides the right proving ground for the framework. In short, this provides the opportunity to work out any of the issues with the framework while implementing in small increments. The steepest part of the project learning curve will occur while implementing the least complex automation laying the groundwork for fewer and less complex problems with future Use Cases all of which accelerates automated script development.

3.2. Plan for Flexibility

It is difficult to understand what objects will be recognized by selected automation tool until scripting against the object actually occurs.

Project Plan: script the selected test cases and engage the WinBig developers for the details of any objects that appear problematic.

- Identify opportunities to work with development to open/create interfaces into the back-side code allowing for testing independent of the user interface
- Decide whether the workaround is easy enough to create script code or flag the particular functionality as a better candidate for manual testing then focus automation efforts on a different application test case
- Flag difficult work-arounds for later scripting where a dedicated resource can expend significant time to develop a robust solution

3.2.1. Scope Control

Successes will happen and unexpected barriers will arise both of which could affect the project. Extra effort (features or test coverage) could be requested where successes are occurring – scope creep. Too much time could be spent dealing with unexpected technical barriers. The Trade-off Matrix and agreement statements from Appendix A will be published once decisions are made in either formal or informal status meetings to control the scope of the automation that will occur from that point forward in the project.

3.3. Defect Tracking

OnTime by Axosoft (www.axosoft.com) will be used to track defects and other issues found during script development. Sixty to eighty percent (60% to 80%) of defects found with automation is estimated to be found during script development.

3.4. Configuration Management

WorldCo's company standard source control tool will be used as the configuration management tool to manage and version test automation project artifacts.

3.5. Resource specialization

There are several items and/or activities that will require through understanding so as to avoid coding a solution that has technical issues preventing the automation effort from reaching its goals. The script development resources will divide up the work with each resource specializing in one or more of the technical topics in order to support the other developers and to keep the script development going in a safe direction.

Specialization by team members to advise other team members on areas of risk reduces the overall risk of the project.

3.5.1. Object Map (GUI Map)

One script development resource will be assigned the task of becoming the expert in automation tool's implementation of the Object Map.

- Primary objective: keep scripts from conflicting from one another due to Object Map issues
- Understand and advise on the impact of the central Object Map repository and how to deploy or move databases for use by the team at WorldCo
- Understand and advise what is required from a configuration management perspective to store and maintain one or more Object Maps for use by several scripts and deploy for use by the team at WorldCo
- Understand and advise what is necessary to create, manage, and deploy custom object definitions for specialized use by scripts

3.5.2. Test Data Management and the Database

One script development resource will be assigned the task of becoming the expert on the WinBig database and the data needs for testing.

- Primary objective: keep script developers from conflicting with one another through data entry, modification, or deletion

- Understand and advise what is necessary to restore test data to a known condition so tests can be re-executed
- Understand and advise what is necessary to have for a data pool to minimize the need to restore the entire database instance
- "Expert on the database" refers more to the content of data in the database and the needs of testing rather than table construction, triggers, stored procedures and the technical architecture of the database itself

Testing, in general, is destructive to data so we must have a careful and controlled plan for restoring back to known conditions. Automated testing uses large volumes of data and will have a tendency to need frequent resets to the known conditions.

3.5.3. Configuration Management

One script development resource will be assigned the task in becoming the expert on how the automation tool manages scripts in configuration management.

- Primary objective: use the existing power of the automation tool management features to manage test scripts, test data, and other test artifacts as necessary so that entire purpose-built automated test suites can be made available to the team at WorldCo
- Relate how scripts and other artifacts are managed in configuration management and advise on code organization so as to not create deployment or portability problems
- Artifacts managed in the CM process could include but is not limited to:
    - Test scripts, individual and entire suites
    - Test data for individual scripts and entire suites
    - Test plans for which individual scripts and suites were created
    - Standards including script coding, test data, and anything else listed in the Framework Standards and Procedures section

3.6. Test Environment Coordination

One script development resource, the lead designer, will be assigned the task of coordinating the environment used for script automation and execution. It is expected that the automation team will share an environment with other testers and there is potential for data destruction between the various test teams.

3.7. Deployment & Portability Planning and Shakedown

One script development resource will be assigned the task of validating the test scripts and test suites can be deployed or otherwise transferred to the team at WorldCo and can be successfully executed by that team. Shakedown activities could include but are not limited to:

- Test scripts and test suites
- Test data
- Processes and procedures along with anything else stored in configuration management
- Environment setup and definition

The basis if the shakedown is that we can organize tests and execute at the script development site in Los Angeles but we need to make sure that the test site at WorldCo can do the same. We will practice moving scripts from where they were developed to where they will be used in testing.

Note	This paragraph describes that the scripts are being developed at an offsite location in Los Angeles and will be executed at the WorldCo facilities that are at a location other than Los Angeles. This works equally well for an offshore/onshore model.

3.8. Script Execution Control Point – Metrics and Measures

One script development resource, the lead designer, will be assigned the task of understanding the Rational tools and how and where the scripts should be run to generate metrics and measures so progress can be monitored. The main things to watch are architectural which drive script design and suite organization and include:

- Run scripts from automation tool test manager to log results. The automation tool test manager is the “driver” and results can be recorded in the test management tool suite for automatic metrics reporting
- Run scripts from driver scripts – scripts running other scripts. Individual scripts are the “drivers” an may require additional work/code/reporting to generate metrics
- Combination of the other two where automation tool test manager is the driver and runs/schedules driver scripts

3.9. Application Development Coordination

All script developers will be responsible for communicating when it would be more appropriate to coordinate with application development to open or create interfaces

that will make testing easier and more comprehensive. Ideas, needs, and proposed solutions will be presented to the Test Automation Architect for discussion with the developers.

Hooks into the application source code will be coordinated by the Test Automation Architect with the application development representative. The Test Automation Architect is to keep the big picture in perspective and understand when hooks into application source code can benefit other testing activities.

Other opportunities could include identifying tests that could be automated and used as part of the development team's Unit Tests.

3.10. Script Code Peer Reviews

The script developers will hold periodic evaluations of one another's code to determine if the code is robust and ready for deployment. A second script developer will review and approve code before it is saved in the configuration management system for future use and maintenance.

3.11. SME and Business Analyst Reviews

Scripted tests, whether individual or in a suite will undergo scrutiny at a functional level by a SME or Business Analyst for the AUT. This helps to ensure that the scripts are testing the right functionality in the application. A specific time to review script execution will be scheduled for the review when the script and/or suite is ready for deployment.

3.12. Knowledge Transfer

The test suites, all associated artifacts, and the knowledge of how to maintain, enhance, and deploy need to be transferred to WorldCo personnel. Training may be required for one or more resources at WorldCo that were not part of the script development team in order to take ownership of the framework and test suites.

4. Phase II Timeline

See MS Project file 'winbig_project_plan.mpp' for detailed breakdown and schedule.

5. Critical Success Factors

- Must have strong management sponsorship
- Simple architecture understandable by all
- Implementation of a framework-based architecture
- Implementation of a data-driven architecture
- Development of standards and procedures for automated script/function development
- Development of configuration management procedures for automated script/function implementation
- Disciplined adherence to the standards, procedures, and configuration management plans
- Reusable, maintainable, expandable, and portable test scripts
- Availability of talent/resources
- Developed and deployed incrementally in phases
- Automation created as a full-time effort, not on a time-permitting schedule
- Robust understanding and management of the Object Map
- Good teamwork and coordination between Mesa and India

6. Assumptions

- Script development is no different than any other software development and must be treated like any other coding activity
- Framework library code will be developed before specific test script code
- Resources used in script development have script development coding experience
- The test environment is separate from the development environment and remains relatively stable during scripting. This includes both the application and databases servers
- Tests that drive in data can have the data restored to know pre-test conditions. This includes completely restoring an entire database if necessary
- The Trade-off Matrix will be used to adjust resources, features, and schedule should trades become necessary
- Script development environment definition document (run books) is built over time as the environment for the application becomes known. This does not appear as a task item in the MS Project plan since it is created a little each day.
- Test environment requirements document is built over time as the test environment for application becomes known. This does not appear as a task item in the MS Project plan since it is created a little each day.

7. Risk Analysis

Risk	Description	Impact	Mitigation
Older version of the automation test tool	The automation team is using an older version of the automation test tool instead of latest and greatest release the automation tool vendor. This version may have problems with the AUT that cannot be solved with any amount of workarounds that may have been solved in automation tool's latest release.	Medium	Upgrade the test automation tool to the latest version from the vendor
Loss of Automation Development Discipline	Loss of discipline to follow the standards and processes listed in the Framework Infrastructure & Governance section due to time and budget pressures. Lack of standards and failing to follow processes are leading contributors to software project failures.	Medium	Assignment of a seasoned Test Automation Architect and project sponsor to both monitor the development process. Use of the Trade-off Matrix to make decisions on features keeping the pressures that lead to cutting corners at a minimum.
Application too Unstable	The application may be changing too frequently to support automation. Too unstable and cannot be scripted. Ideally, automated scripts are developed from test cases that have been executed manually.	High	Concentrate the Smoke Test scripting activities on the areas of the application that are more stable.
Object Recognition	Many objects (controls, widgets, etc.) may have been custom-written for the application and Robot nor any other test tool can recognize the object. Work-arounds may be difficult if not impossible.	High	These become the first candidates for coordinating with development to provide interfaces or hooks that the test tools can recognize. In the meantime, manual testing is the only available choice.
HTML Objects	The automation tool may not provide direct access to many important properties and text of HTML objects.	High	The team is anticipating writing code to load page source into XML and the access individual properties of each object using XML DOM.
Underestimated Effort to Interpret Test Results	The application may behave differently than expected during the test. Effort will be required to determine if the application behaved correctly and the test is wrong or the AUT failed and the test is correct.	Medium	Level-setting with SME's or BA's as part of the review process for the scripts.
Too Few	WorldCo may not have enough	High	Investigating obtaining more

Risk	Description	Impact	Mitigation
Licenses	test automation tool licenses for the test automation team.		licenses.
Conflicts with Other Testing Groups	The test environment is shared with other test groups that may interfere with automated script development or automated test development may interfere with other groups. Data will be the main area of contention.	High	Assignment of a script development resource to coordinate with other test groups on issues pertaining to the test environments.
Service Level Agreement with Development	The application may not be in an optimal testing condition in that many hooks or code elements that can be present to facilitate testing are not present. Testing can coordinate with development and the developers can agree to implement the hooks but the turn around time could be too slow to meet testing's implementation schedule. Knowing the code changes are in the pipeline, testing will have to work on other areas.	High	A resource from the script development team is being specifically assigned to coordinate with development. Development turnaround is determined by their existing queue and schedule.
Disbursed Team Composition	Team members WorldCo working on deliverables puts additional burden on all team members to communicate in a timely fashion over time and geography. Response time to deliver project deliverables may increase jeopardizing making schedule.	Medium	Investigating bringing additional resources from WorldCo to co-locate all team members.

8. Resource Requirements

#	Designation	Team Size
1	Test Manager	1
2	Test Automation Architect	1
3	Test Automation [Lead] Engineer	2 / 3

WorldCo will be solicited to provide a resource to be part of the script development team who could join the team in 4 or 5 weeks into the project.

9. Roles and Responsibilities

Role	Responsibility	Organization
Test Manager	• Review of detailed Test Plan of the project. • Identify training needs as per project team member specific to the assigned project(s). • Set-up and maintain proper communication within WorldCo and vendors. • Regular review/monitoring of deliverables and escalate issues for timely resolution. • Review of deviation and corrective/preventive measures taken up during automation development and execution of the testing cycle • Review and approve the metrics and measures to be used to monitor the effectiveness of test automation	WorldCo
Test Automation Architect	• Design and enforce framework processes, standards, and rules • Design and enforce test data processes, standards, and rules • Maintain the big picture view identifying opportunities to create common code and methods across all automation efforts • Scope level of effort and activities associated with the automation of applications • Preparation of the periodic test progress/status reports • Propose metrics and measures to be use to monitor effectiveness of test automation. Design systems for the approved metrics and measures • Keep the big picture in mind when identifying custom objects that require script work-arounds	Contracting firm
Team Automation Lead Engineer / Test Automation Engineer	• Write scripting code for the framework • Write scripting code for application-specific tests • Write script code for custom objects and other work-arounds • Preparation of the periodic test	WorldCo Contracting firm

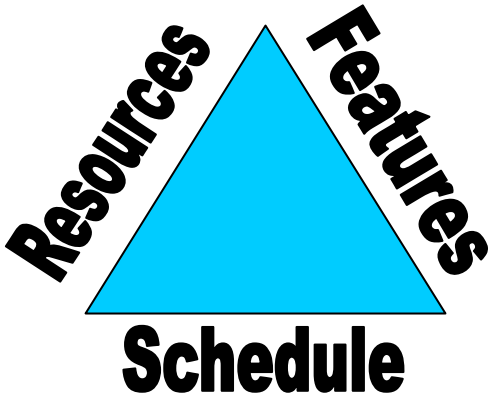
Role	Responsibility	Organization
	progress/status reports. • Preparation of Use Cases for automation • Preparation of Test Data • Log defects discovered during testing	

Appendix A Trade-off Triangle and Matrix

The Trade-off Triangle shows the challenges project teams face when completing a project. The Triangle is obvious to most project team members but is rarely seen in a project plan. Teams make trades “on-the-fly” without deliberately documenting what they traded.

The Trade-off Matrix formalizes the decision making process for the trades made when team members decide to add or reduce components of the project. Just as important, it helps to clearly document the decisions that were made.

Trade-off Triangle



Resources: people and money
Schedule: time
Features: scope

Of these, failing to manage Features (scope) is the #1 cause of project failure.

Project Trade-off Matrix

	Fixed	Chosen	Adjust
Resources		X	
Schedule	X		
Features			X

The following are sentences that could be stated in meeting minutes or status reports to clearly describe what team members decide to trade and how to proceed forward in the project.

Resources
- Given fixed resources, we will choose a schedule and adjust the feature set as necessary.
- Given fixed resources, we will choose a feature set and adjust the schedule as necessary.

Features (framework, scripts)
- Given a fixed feature set, we will choose a level of resources and adjust schedule as necessary.
- Given a fixed feature set, we will choose a schedule and adjust resources as necessary.

Schedule
- Given a fixed schedule, we will choose a level of resources and adjust the features set as necessary.
- Given a fixed schedule, we will choose a feature set and adjust resources as necessary.

<u>Agreement between parties and published in meeting minutes or status reports</u>

Given a fixed _____, we will choose a _____ and adjust _____ as necessary.

Example: Given a fixed schedule, we will choose to add more scripts and adjust programming resources by adding personnel as necessary.