business component & XML driven test automation framework - BXD

Test Automation is an accepted technique which is adapted by the industry for increasing the effectiveness of the testing phase.

Pradeep P P
QuEST Global
## contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>Introduction</td>
<td>01</td>
</tr>
<tr>
<td>0.2</td>
<td>Industry Practice and Lacunae</td>
<td>01</td>
</tr>
<tr>
<td>0.3</td>
<td>Suggested Solution</td>
<td>03</td>
</tr>
<tr>
<td>0.4</td>
<td>Business Component and XML Driven Test Automation Framework (BX-D)</td>
<td>03</td>
</tr>
<tr>
<td>0.5</td>
<td>Advantages</td>
<td>06</td>
</tr>
<tr>
<td>0.6</td>
<td>Case Study</td>
<td>06</td>
</tr>
<tr>
<td>0.7</td>
<td>Return on Investment</td>
<td>07</td>
</tr>
<tr>
<td>0.8</td>
<td>Future Work</td>
<td>07</td>
</tr>
<tr>
<td>0.9</td>
<td>Author Profile</td>
<td>07</td>
</tr>
</tbody>
</table>
1. Abstract

Test Automation is an accepted technique which is adapted by the industry for increasing the effectiveness of the testing phase. The recurring tasks are being automated by the tools thus simplifying the human efforts and results in increased quality of product under test. A study of test automation programmes in the industry reveals the fact that a good percentage of them fail to find the intended results. A pareto analysis reveals that the majority of the failures are the result of lack of capability overlap of test resources in the domain and test automation. In order to have success in a test automation program, the test engineers should be knowledgeable in the test automation tool as well as the domain under test. In practical scenario it is seen that, many a times, the test engineers who are experts in the domain are not fully capable of test automation due to lack of their awareness in testing tools.

The paper explains the experience of authors in creating an XML Driven business component test automation Framework – BX-D, which simplifies the test automation by concealing the complexity of script development. Test automation of a single product can be done using different test automation tools based on the customer demands using BX-D framework. It also ensures that multiple tools can be used for test automation of the same product which in turn reduces the limitations of the tools. The horizons of BX-D are wide open with the opportunities of cloud testing which is going to conquer the software testing world.

2. Industry Practice and Lacunae

The software industry has started embracing test automation as a practice for increasing the productivity. During the initial planning of the project itself, the effort and time for test automation is estimated. The tool for test automation will be identified in the initial period and procurement of licence and training as per the requirement is planned to ensure smooth test automation. The effort for test automation is a function of the number of tests which is planned to be automated. It is a well accepted fact that all the test cases are not planned to be automated due to the limitation of the tool which is used for test automation. The tests which are planned to be automated will be identified during the test design phase. Once the candidate features implemented in the product is received by the testing team, the test automation of identified test cases will be done by the test engineers.

When a product branching happens, test automation also takes a branching, but rarely as efficient as branching of source code. Separate scripts have to be maintained for each product line.

Even though it has been years since the initiative is in air, test automation is yet to come out of the childhood discrepancies. Even though test automation has been considered as one of the best methodologies that can be adapted for improving the productivity of the projects, there is reluctance in adapting test automation to the project. The lack of competency for performing test automation, intricacy in choosing the apt tool for automation and the demands of a highly ambitious market can contribute to one of the factors for this reluctance.
I. Competency Requirement

“For the most part, testers have been testers, not programmers. Consequently, the ‘simple’ commercial solutions have been far too complex to implement and maintain; and they become shelfware.”

- Carl J Nagle

Test automation includes designing test ware, implementation of the test cases and monitoring and closure of the results more than just executing the test cases. The test cases developed by the tester needs to be converted to machine executable code for performing test automation. Scripting has a very important role in test automation which in turn means that programming skills has become a vital factor in test automation. For developing the test cases domain knowledge is required and for automating those test cases programming skills are required. The competency required for a tester for a product which required test automation should be a blend of good domain knowledge and coding skills. This blend of competency seems to be lacking in the current industry which creates reluctance to the usage of test automation tools in the current industry.

Figure 1: Competency Requirement of a tester

ii. Choose the right tool

Apt tool selection for test automation is one of the challenges faced by the test engineers. The features supported by the different automation tools vary. It can also seen that many a times all the features of a product cannot be automated using a single test automation tool. It can also be noted that purchasing a costly tool for doing the automation of some features of the SUT is not cost effective. It all points towards having a framework available in such a way that different tools can be used simultaneously for the test automation of the SUT.

Figure 2: Choose the right tool

iii. Version Complexity

Different versions of a product will be created from base version of the product in a product line as shown in Figure 3 and will be delivered to multiple customers. The most important challenge in front of the tester is how to automate testing of the different versions of this product. Automation is been done for each version of the product which means that the automation of the core functionalities are repeated for each version which results in huge wastage of effort, resource and time. In this scenario if any change is incorporated in the base version of the product automation scripts needs to be repeated for all the versions in which the change is affected which in turns results in the huge effort for the maintenance.

Figure 3: Different versions delivered to different customers
3. Suggested Solution
The most imperative factor for the disinclination towards test automation is
- Huge effort which is contributed by the lack of expertise of the resource
- Maintenance effort for incorporating the change due to customization of product into the automation script
- Lack of skill in opting the right automation tool
- Cost involved and skill required for using different automation tools in single product

The resolution to all the concerns mentioned above can be achieved through bringing in a centralized test automation framework. This framework should be
- Easily adaptable to the customization done in the product
- Reduce the dependency to the programming skills needed by the tester
- Supports multiple test automation tool

BUSINESS COMPONENT AND XML DRIVEN TEST AUTOMATION FRAMEWORK (BX-D)

provides the solution for all the scenarios which are creating disinclination towards test automation and supports all the demands for a centralized test automation framework.

BX-D framework helps to utilize the skills of the persons who have good knowledge in the domain and test automation skills in a unique way. The testing team of an organization shall be divided into two teams – Domain experts and Automation experts. The responsibility of the domain expert is to study the product and domain and create the corresponding testing artifacts. Automation experts shall be responsible for implementing the business components received from the domain expert in the test automation tool. Domain experts can select the business components to create the test scenarios for test execution easily from the BX-D framework. The workflow of the BX-D framework is as mentioned in Figure 4.

Figure 4: Workflow of BX-D framework

I. BX-D Framework Architecture
BX-D has been built with four layer architecture – System under Test, Test Tool Layer, Test Case Layer and Framework Layer. Figure 5 shows the architecture for the test automation framework, BX-D.

Figure 5: Architecture Framework of BX-D

System under Test is the product for which the testing process is been conducted.
Business Component & XML Driven Test Automation Framework, BX-D

i. Framework Layer
As per the list of business components which has been given by the domain experts, business components and test scenarios are created in this layer. The test scenarios generated shall be saved for specific tool or specific customer. The main modules in this layer are Test Execution Configurator, Business Component Service, Test Scenario Generator, Test Scenario Collection and Test Result Poll Service.

a. Test Execution Configurator
BX-D framework can be configured for different test automation tools and for different customers. The tools which are configured in the test tool layer shall be listed in the Test Execution Configurator and test engineer can select the tools and customer for which the test scenarios are being generated. This module gives the provision to add, delete and save new tools and customers to the framework. Figure 6 shows the screenshot of the test execution configurator.

Figure 6: Test Execution Configurator

b. Business Component Service
Business Component Service Module has two components – Business Component (BC) Collection and BC editor. The business components as listed down by the domain experts shall be created using the BC editor. The BC editor gives the provision to select the tools for which the business components need to be added. Figure 7 shows the screenshot of BC editor for a login feature.

Figure 7: Business Component Editor

Business components are created and saved for specific tool using the business component editor and the business components thus created shall be listed down in the business component collection. BC collection lists down the business components and the tools in which the BC is implemented. Figure 8 shows the screenshot of BC collection.

Figure 8: Business Component Collection

c. Test Scenario Generator
The business components thus created can be dragged and dropped to the test scenario generator and can be customized as per the requirement. Figure 9 shows the screenshot of the test scenario editor.
Figure 9: Test Scenario Generator
The test scenarios generated can be saved as test cases in xml format to the test case layer of the BX-D framework as shown in Figure 10.

Figure 10: Scenario saved to XML layer

d. Test Scenario Collection
The scenarios and test cases created using the test scenario generator can be viewed at the test scenario collection of the BX-D framework as shown in Figure 11.

Figure 11: Test Scenario Collection

ii. Test Case Layer
The test cases which are generated using the test scenario generator shall be saved in XML format in the test case layer of the BX-D framework. These test cases shall be organized under each test automation tool as shown in Figure 14. This is an intermediate layer which will be generated when test cases are saved in the BX-D framework.

Figure 14: Test Case Layer

iii. Test Tool Layer
Automation Engineers shall be responsible for the activities which are done in the test tool layer. The XML test cases are parsed into the test automation tool. The automated test executions will happen as defined in the test automation scripts developed by the automation engineers.

e. Test Result Poll Service
The test scenarios shall be selected and executed by clicking the Run button as shown in Figure 12.

Figure 12: Test Scenario Execution
The test results of the test cases shall be listed in the test results viewer of the BX-D framework as shown in Figure 13.
1. Advantages
One of the most important factors for the disinclination towards test automation is the expertise which is required for test automation and complexity involved in the maintenance of the automation scripts in a product line. BX-D framework provides a well defined platform for the testing resources to express their testing skills and automation skills. The test engineers can showcase their testing skills without having the fear of lack of awareness in automation whereas automation engineers have an ideal platform to showcase their automation skills. The distinction of the test engineers has brought in reduction of the effort which was spend for the maintenance of the automation scripts done by less skilled resources where customization of the products were done for different customers.

The high cost involved in using different automation tools for testing of one product is solved through BX-D framework. Different features in one product can be tested using different automation tools and the effective combination of open source and licensed tools resulted in cost optimization and optimized utilization of the tools without affecting the budget allocated for the project. Testing can be scheduled overnight using different tools and test engineers shall be receiving the test results in the morning.

The demands of multiple customers for test automation of the same product can be achieved by using the BX-D framework. The versions are configured in such a way that by choosing a particular version the test cases corresponding to that shall be executed. This has enhanced customer satisfaction without putting additional effort, cost and complexities.

Even though it is seen in the architecture diagram that it is a complex diagram, it is an easily adaptable framework. Test cases can be saved for each customer and also for different versions of the product. Also easy drag and drop feature in the framework allows the test engineer to easily create the test cases needed for a particular version or customer thereby increasing the productivity.

2. Case Study
The product is a web based solution for element management. There is a framework for the product and customized versions are delivered to different customers across the globe. It was really painful to ensure quality multiple product lines simultaneously since different customers are using different versions and new features are added to the latest version for competing in the market. The test team dedicated for the product was lacking expertise in test automation tools but has expertise in manual testing and in the domain.

On considering the situation, it was decided by the QA department to design an automated test automation framework which gave birth to BX-D. This enables the testing team to develop automated tests even without the knowledge of test automation tools. The goal was to develop a test framework which segregates the scripting part and test case design so that test automation experts will handle complex scripting part and domain experts will create test cases using the business components created by the test automation experts. It was also decided to bring the flexibility to integrate multiple testing tools to the framework so that maximum usage of open source tools can be ensured and licensed tools can be used only for those areas where open source tools fails to support some technologies/ user interface controls. It also helped to ensure the optimum usage of licensed tools across the projects and increased the return on investment on tool purchases across the organization. Domain experts were also given the freedom to choose the version of product against which the test has to be executed and the complexity of the implementation of the same is masked from them.

About 80 percentage of the tests could be implemented using the open source tool Selenium and remaining 20 percentage were done with the licensed tool Testcomplete. An existing floating license of Testcomplete could be reused for the project without affecting the work of other projects. It was also decided to institutionalize BX-D across the organization which helped to share the licenses of test automation tools effectively.
3. Return on Investment
By institutionalizing BX-D across the organization we got multidimensional advantages. It created a huge impact on the direct cost of tools and indirect cost of effort.

About 80% of the test cases could be automated using free tools and the usage of licensed tools could be minimized to 20%. This helped to reduce the cost of tool licenses by 80% across the organization.

Training cost on testing tools also reduced by 70% since a dedicated team of test automation only needs to be trained.

There is huge impact on maintenance cost since it is easy to adapt product changes into test cases. Maximum reuse of test scripts against multiple product lines also gained.

4. Future Work
BX-D framework can be made more cost effective when it is integrated to the evolving concept of cloud network. BX-D framework gives the flexibility to use different automation tools in one product. Rather than an organization buying a tool for doing the test automation of some features in a product, using BX-D the tools can be taken from cloud for testing and the organization had to do the payment only for the time in which the tools is been used from cloud. This will widen the spectrum of usage of test automation tools thereby improving the productivity of the organization in a highly cost effective method.

6. Author Profile
Pradeep is working as Program Manager at QuEST Trivandrum. He has 10 years of experience in software testing industry and is an Engineering graduate from College of Engineering Trivandrum under Kerala University. He is a Certified Test Manager by STQC and