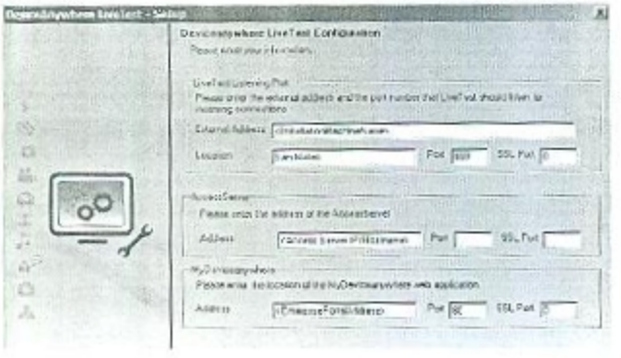
4) DeviceAnywhere Live Test and Live Monitor Server Installation Testing:

The Live Test and Live Monitor Servers was use to execute scheduled test runs and monitor scripts. In this step, I had to test the smooth installation of Device anywhere Live Test and Live Monitor Servers and its connectivity and integration with other component servers.

Element 4

Sample of DeviceAnywhere Live Test Configuration Screen



Note: Live Test Server and Live Monitor Server installation steps were same beside the configurations.

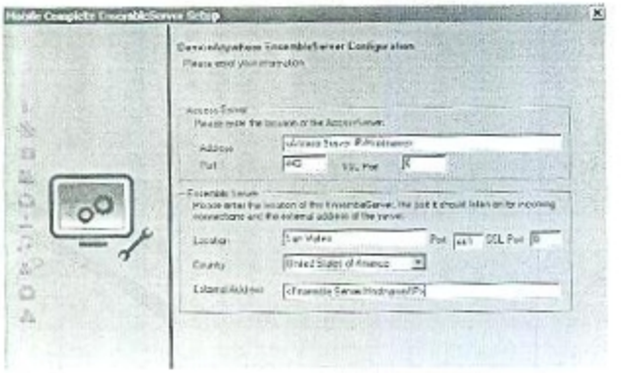
5) Ensemble Server Installation Testing:

The Ensemble Server was used to host one or more devices over USB, Wi-Fi, or Bluetooth. Each Server was able to hosts 2 to 6 devices. If the number of Ensemble Server were increased, then installation of each Ensemble Server had to perform in separate machines. Ensemble Server was responsible to establish communication between DeviceAnywhere Studio via Access Server and Hardware (Mobile Devices). In this step, I had to test the smooth installation of Ensemble Server and its connectivity with devices and integration with other component servers.

Element 4

Sample of Devices Attached in DeviceAnywhere Lab.

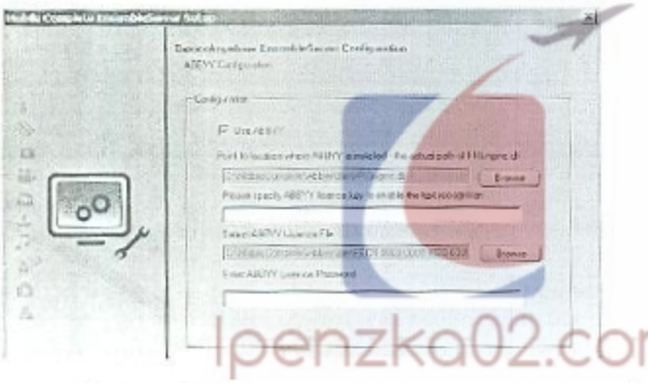
Sample of Ensemble Server installer configuration screen



Ensemble Server was also using third part ABBY Text recognition component that was using OCR engine to extract text. This feature of text extraction was used by text extraction command to extract text from device screen in Deviceanywhere studio while creating scripts.

Here, I had to test the installation of this third party software along with Ensemble Server installation in which it was integrated, and verify that text from device screen was extracted without any problem.

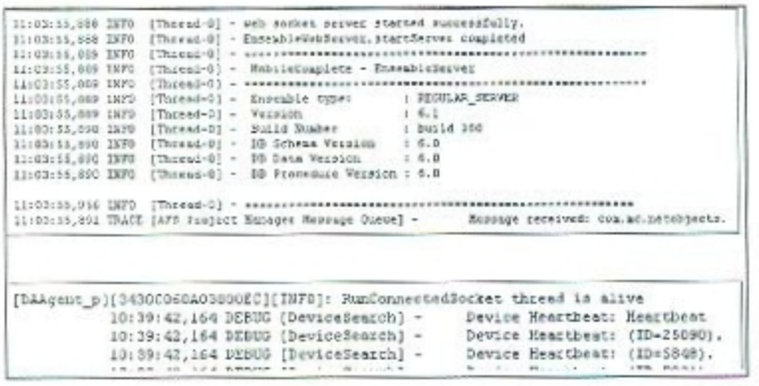
Sample of Ensemble server with ABBY configuration screen



If hardware devices were not recognized in log then I had to perform troubleshooting by doing RCA (Root Cause Analysis) by verifying system configuration and server restart processes. Most of the time, I had to communicate with the device lab administrators located globally to verify the physical cables of the devices and ports connectivity; and in parallel I had to work with hosted service to make the device working. This trouble shooting was very complex, if the devices were located in diverse locations. Here my communication skills and other inquiry and problem solving skills had helped me to carry out the work.

Element 1 & Element 6

Following is the sample Log of Ensemble server showing number of devices attached to it



```
11:03:55,886 INFO  [Thread-0] - web socket server started successfully.
11:03:55,888 INFO  [Thread-0] - EnsembleWebServer.startServer completed
11:03:55,889 INFO  [Thread-0] - ************************************************************
11:03:55,889 INFO  [Thread-0] -  MobileComplete - EnsembleServer
11:03:55,889 INFO  [Thread-0] - ************************************************************
11:03:55,889 INFO  [Thread-0] -  Ensemble type:        : REGULAR_SERVER
11:03:55,889 INFO  [Thread-0] -  Version               : 6.1
11:03:55,890 INFO  [Thread-0] -  Build Number          : Build 350
11:03:55,890 INFO  [Thread-0] -  DB Schema Version     : 6.0
11:03:55,890 INFO  [Thread-0] -  DB Data Version       : 6.0
11:03:55,890 INFO  [Thread-0] -  DB Procedure Version : 6.0

11:03:55,916 INFO  [Thread-0] - ************************************************************
11:03:55,921 TRACE [AFS Project Manager Message Queue] -       Message received: com.ad.dataobjects.
```

```
[DAAgent_p)[3430C060A03B00EC][INFO]: RunConnectedSocket thread is alive
          10:39:42,164 DEBUG [DeviceSearch] -      Device Heartbeat: Heartbeat
          10:39:42,164 DEBUG [DeviceSearch] -      Device Heartbeat: (ID=25090).
          10:39:42,164 DEBUG [DeviceSearch] -      Device Heartbeat: (ID=5848).
```

Element 4

<u>In order to ensure that Product is accurately working and complete environment is up and running, I had to execute following steps</u>

1. Login to studio application with credential access and verify that DeviceAnywhere studio launches appropriately.
2. Select the device packages and attempt to lock the device and release the device to ensure the both functions are working appropriately.
3. Perform some key inputs on live devices and verify that devices are taking input and the results are reflecting on device screens.
4. In order to ensure that key assets of New Scripting Mechanism is working properly, I had to create action, state and test cases; execute the action and test cases both separately and upload the results and verify that results are accurately displaying on portal.
5. To ensure that DeviceAnywhere Portal is correctly configured, I had to verify the portal configuration in DeviceAnywhere studio and Access Server

If all above steps worked without any problem then it mean that all the server components are intact and integrated. In this way, I completed the Integration testing of Core Component of the Product.

Following is the snap shot of DeviceAnywhere Enterprise suite showing locked device and log of inputs.

In this part, I used my skills of root cause analysis (RCA) and fundamental knowledge of computer engineering like setup and configuration, interacting with database schema. I also utilized my knowledge of operating system principals while installing the core services of product that also helped me to test the memory utilization of services and overall performance of the product. I utilized my specialized skills of integration testing of core services. My over all fundamental knowledge of software engineering skills, like knowledge of hosting web application and set up of DeviceAnywhere Enterprise portal used to carry out the work. I also provided training to my team to carry out the product deployment task where I utilized my skills of communication skills to convey the knowledge and training.

Element 3, 4 and 5

Part-2: Developing Understanding of New Dynamic Scripting Mechanism and Executing Functional Testing by Exploring the Product:

The new approach of scripting had leveraged building block to define script in modular, device-independent. It provided powerful tools for creating and organizing the reusable components of test scripts, where user could minimize repetitive coding and easily port test scripts across all devices and mobile platforms.

Element 4

There were various new concepts introduced along with New Scripting Mechanism in order to develop the foundation of new way of scripting.

The core of the project was being developed in USA head quarter. In order to accomplish the testing of this project, I had to use my communication skills to aggressively interact with US engineering team to develop functional understanding, referring product requirement document, get the identified functional issues resolved, and make the product stable.

Element 6

The New Scripting Mechanism was highly complex in testing as compared to the conventional scripting testing in the product. Pure **functional testing** was very difficult and time consuming to test each function without knowing its purpose. The New scripting Mechanism was, the unique way of scripting that was introduced in the product with the purpose of creating business cases and develops script as per the real time scenarios. Therefore, the product was required to test by considering how actual user would use it.

In order to test New Scripting feature, I recommended to use **Exploratory-testing methodology** to test such a complex and giant feature of the product with real time scenario that user would execute.

Element 4

Following were the key components of the New Scripting Mechanism.

Project,

A project was the organizational container for test scripts and test devices; generally specific to an application or functionality that user could wish to test. Project comprises of test assets as follows.

a) States
b) Actions
c) Test Cases
d) Test Cycles

The assets were the main building block used to automate the mobile testing. The meta data of project was also included with permissions, error definitions, project variables, and project dependency information.

I started to execute testing by creating a project, I found following dialogue box that specify the version control user wished to use. Here my knowledge of version controlling from configuration management work experience helped me to test this function. Here I identified various complex bugs like system was not able to save the version of the project locally because when I was importing the version again, it failed to import. This function was required if user wish to saved the project locally and share with other user. I reported that issue to software engineering team to resolve it.

Element 4

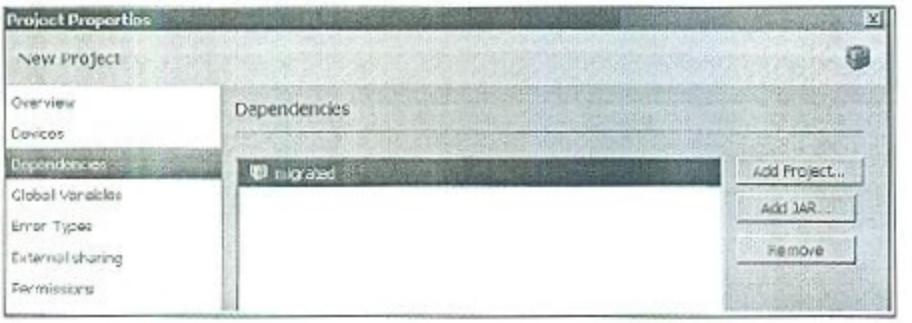
There was another option to save the version called DeviceAnyWhere version control, in which system itself maintained version of the project and I found it working correctly. This option was required, if user wishes to save the project in the system and maintain library for future use.

Initially the product was highly un-stable and had a lot of functional and requirement related bugs I reported, and gradually I found that engineering team was resolving those bugs and the product was going to stable.

I test the feature of Project by adding and removing various actions, states and again retrieving them and found it successfully imported.

After pressing the Create Project button, I found that project properties dialogue box appeared was showing assigned device to the project specifying the list of devices with which the project would be used. In property dialogue box, user could also defined dependent projects to specify to which project the current project is dependent. In property dialogue box, user could also defined global variables to be used by projects, error types and permission to user to use it.

I test these features by assigning various combinations of devices, providing dependent projects, test their dependencies by defining global variable, use them in other projects, and attempt to change the project with users having no permissions. In this testing, I found various functional issues and get them resolved by engineering team. I also executed test cases with various combinations of parameters to create projects. This way I tested all the key functions of project considering the user scenarios, identify the issues, and reported them to engineering team for resolution and verified them later on.



When I created Project, folder was automatically created in the project directory for different types of test assets. I performed further testing by creating 100 multiple projects by manual creation, verify the stability, check that product work smoothly, and do not halt. I verified that in each asset, sub folders could be created considering in mind that it could be performed by the application user because user would like to organize the test cases that deal with distinct functional area of an application.

State:

States were used to define known device conditions (text, audio or image based) that could be referenced to verify the result of an action or a sequence of device interactions. States defined a device mode, such as the device home screen or the Web browser home page in order to specify an expected result and could be reused across scripts and devices.

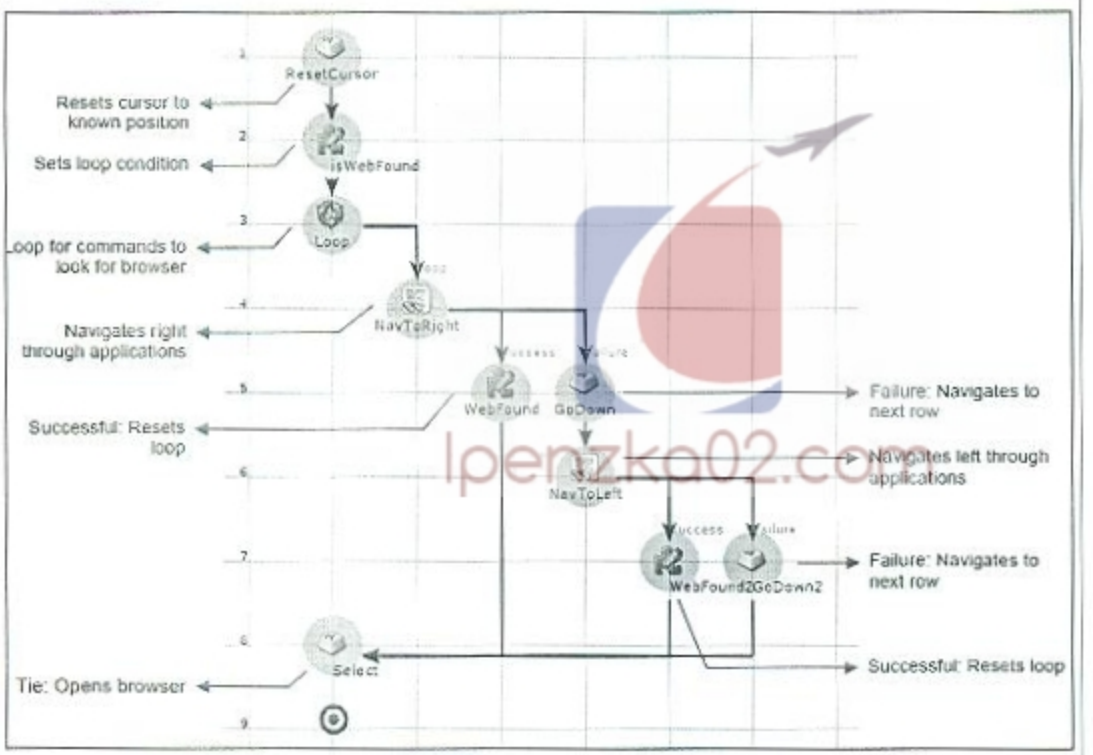
States were device independent and they were defined for all project devices and consist of device specific implementations to account for differences in interfaces. If an expected result changed over time, a state could be updated in one place, without having to update every script containing a reference to it.

Actions:

Actions were the basic building blocks and unit of scripting of a test case. Actions were use to accomplish a specific function. Actions were discrete procedures that made up the larger processes that were called as test case. Each broad step in a test case was corresponding to an action and consists of a series of device interactions and verifications. The size of actions were not limited, they were actually mini-scripts that could be reliably reused across devices and test cases. Actions were device-independent, it mean they were defined for all project devices and consisted of device specific implementations to account for differences in interfaces.

For Example actions in a test case that tests a Web site by opening a web browser, and navigate to a URL.

Below is the sample **action** that I implemented. This action script reset a BlackBerry device and then navigates to opens the web browser. This script exemplifies the use of set variable, loop, and Navigate To Commands.



Test Case:

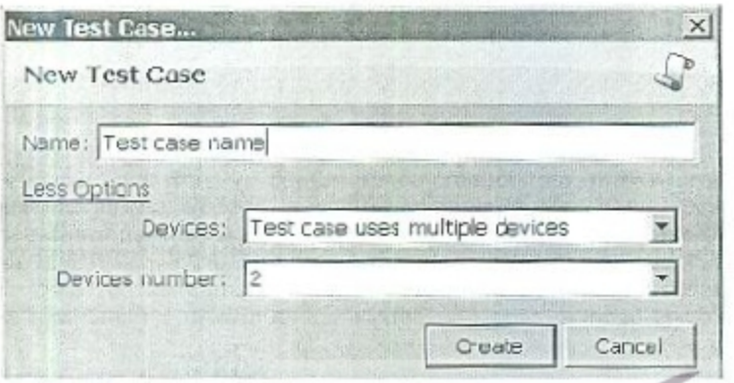
A test cases was the asset of project that describe the broad process to accomplish a test goal on mobile application. For example, a test case might consist of viewing and deleting call records from a device, searching for directions to a specified address in a mobile web browser, or getting the weather report for a specified zip code from a short-code weather service.

Test cases were device-independent. They were generally consisting of calls to previously defined actions and states with additional commands such as branches and loops to control

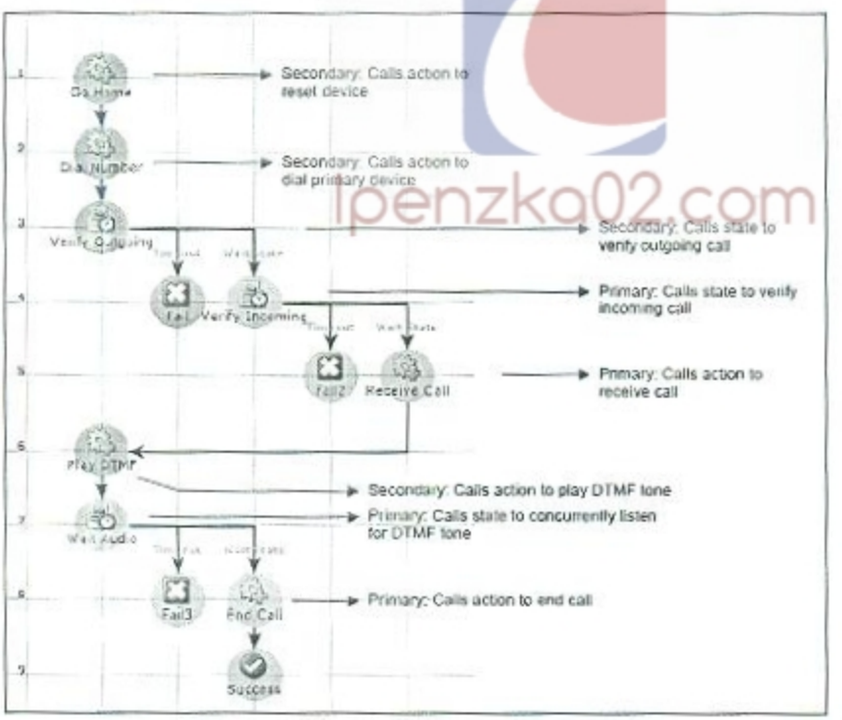
script logic. As I was already working with Enterprise product from where I already had gained extensive understanding and working of control commands like (Reset, Send Keys, Send SMS, Hardware extensions, Play Audio, Find and Touch, Upload Application, Find Text and Touch, Find state and Touch, Wait, Wait Text, Wait Image, Wait Event, Wait Audio, Wait State, Extract Text, Navigate To, Set Variable, Branch, Loop, Loop Data Set, Success, Fail, Execute Action, Execute Test Case). I utilized all knowledge and skills of those controls in test case feature testing.

Test cases were modular and generally consisted of calls to actions and states using controls to develop script logic. The larger test case goal could be broken down into discrete procedures, represented by actions.

I tested the asset of Test Case by creating various Test Cases in Project.



In following example, I use a reference point in a test case and call an action containing a reference point. I also drag in a state from the project directory, which automatically inserts a Wait Event command in the test case script



Test Cycle:

A test cycle defined test plan and consisted of a group of test cases assigned to be run on all or a subset of project devices. A test cycle defined a set of test cases to be executed in batch. It allowed to create device-test case groupings to focus testing on particular functional areas or test targets.

I tested this feature with various combinations of calling test cases in it and verified the results.

Conclusion:

In the whole project I used my knowledge and skills of scientific method and other inquiry and problem-solving processes along with that I also used my fundamental knowledge of computer engineering and specialized knowledge of requirement analysis by reviewing functional specification document; and based on it, I used my software testing skills to functionally test the complex scripting mechanism and used my software integration testing skills to test the core services of the product.

Work/Study Episode 2
(April-2013 to June-2013)

Elements

Overview of the project

To Evaluate Software Test Automation tool for the purpose of Automated Functional and Regression Testing of Core Banking System that was developed using Java Swing library, having backend database of IBM DB2, and to propose best tool along with the POC (proof of concept) for further procurement.

Your role and responsibilities

While working as Software Quality Assurance engineer I was given an assignment to identify a suitable automated testing tool that could be integrated with Core Banking System (with JDK1.3 support).

My role was:

1) To do research and provide analysis of various functional testing tools, which support Java Swing Application and IBM DB2 Database.
2) To integrate the evaluated tool with Core Banking System.
3) To do comparative analysis based on Cost and Benefit.
4) To recommend appropriate tool for procurement.
5) Provide POC (proof of concept) to justify the selection.

Complexities (using the complexity definitions) and challenges of the project

While working with Core Banking System in Bank Al Habib Ltd, I found that whenever any new functionality is added to system or existing functionality is modified, I had to test the entire application, making sure that existing functionalities of the system were stable and had not been broken. Technically, this is called **Regression Testing (Re-Testing),** At this stage, usually there were high chances that existing functionalities may not work as per expectation.

In this situation, I had to execute several manual testing cycles whenever any change occurred. It caused me to put a lot of effort to test the same functionalities in the system repeatedly to ensure that system functionalities are stable and not broken due to recent fix of the bugs. Beside this, the last minute changes in application also required me to test the entire application, that most of the time was not possible due to unavailability of time.

In order to efficiently execute the testing effort, with maximum test coverage in minimum time (comparatively less than manual), I recommended IT management to allow research various

tools available commercially and identify tool that could integrate with Core Banking System and made us to automate functional test cases. After my several verbal and in written discussions and communication with management, this assignment was given to me to carry out research and analysis and identify tool. Here my leadership and communication skills has helped me to justify the management about my ability to undertake this assignment.

Element 6

While doing the assignment I faced following major complexities and challenges.

a) I found that the Core Banking system was developed using JDK 1.3 and was being used with JRE1.3 for run time environment. At that time, JDK had been upgraded to version 1.7 but the core banking system was still using version 1.3 that was the quite older version or out dated version. Therefore, most of the third party tools required for integration was not able to support that older version of Java. In addition, the Core Banking system database was IBM DB2 for which very few tools were available with limited support.

b) While doing research and analysis, I also observed that there were very less number of tools available that could support Java swing application and support older version of java.

c) Another hurdle was that the core Banking System was desktop based and therefore the components used in application interface were highly complex, especially Java Swing components. Many tools that I found were able to record application components based on X and Y coordinates only. In such situation whenever any change in the position of components occurred, the test scripts failed to recognize those components. After doing a rigorous search I found that there were very few tools were available that could capture the components with its entire properties and methods and have support of IBM DB2 database.

d) An another challenge, I found was that the available Java supported tools were highly complex in configuration and Test Scripts were not easy to develop and execute smoothly. I also found that those tools were less interactive with complex GUI based controls and script were not easy to maintain and tools were difficult to learn and were not cost effective.

How does this project demonstrate application of your engineering knowledge?

This was the major challenge for me to identify best tool for a highly complex Core Banking System, that could be used to develop a Test Automation Framework on which entire Functional Test Cases could be automated and used in Bank for prolong period. As I had already worked as Software Quality Assurance Professional for several years in various organizations from where I had gained rich of experience and knowledge of Software Quality Engineering, Best practices and Relevant Tools and Technologies, therefore this project assignment was an opportunity to apply those engineering knowledge and experience and undertake this project more confidently. I performed in depth research and analysis, and solved engineering challenges based on various judgments while evaluating automation tool for such a complex system.

In order to evaluate automated software testing tools, I did thorough research by performing following steps.

1) Applied for Trial versions using my official Email ID i.e. nizar.karim@bankalhabib.com as most of the vendor do not send trial license to private email address.
2) I setup virtual machine and Installed and configured the software in test environment.
3) Integrate the tools with Core Banking System and evaluate the tool for its core features.
4) Escalated technical issues to vendor, corresponds with questions, and get the support.
5) Finally, I performed comparative analysis of the tools, which were evaluated in detail; and among them, a tool was recommended as the best tool for procurement.

Element 4

This was my first experience, which gave me an opportunity to explore tools and technologies and in particular, I had to identify a tool for Java based systems. I learned to handle different

complexities of Java code and its application interface in order to develop test scripts as part of evaluation exercise. In addition, I used my extensive communication skills in the form of writing to explain technical issues and problems to vendor in a clearer and understandable manner and I learned to communicate with vendors to acquire support and develop good relations that I could use later, if the tool is acquired.	**Element 6**
I had to undertake an extensive evaluation exercise based upon following criteria due to complexities of Java technology and the Core Banking Desktop Application developed using Java Swing library. 1) **GUI Object Recognition:** Based on GUI objects of Java based application, I evaluated that tool • Has ability to record each swing object on Core Banking system interface. For Example, Text Box, Buttons, Recognizing Java Object in Drop down Menu and List boxes etc • Has ability to deal and record effectively with dynamically generated objects. 2) **Platform Support** I evaluated that tool support all the required platforms like Windows XP, Windows Vista with the • Ability to record test on core banking system but different OS platform. • Ability to play back the recorded test cases on core banking system but different OS platform in which actual recording has not occurred. 3) **Recording Non Standard Java Objects** In this, I evaluated if the tool has the ability to record objects against non-standard Java classes that normally required by the Java application. For example, third party controls toolset used in application. 4) **Java Playback** • Tool is reliable to repeat playback the evaluated test cases again on same platform. • Provide some types of generic capability to deal with application. For example, some time application is not ready for execution, there for tool should be able to correctly synchronize code execution. 5) **Visual Test case Recording** • Provide ability to visually record the test cases by interacting with application interface as a real user would do such that tools used its own controls and develop scripts while user is in interaction with the system. • Provide the ability to insert a validation statement while interaction with application interfaces during recording to visually select validation properties. For example, tool should be able to sense contents of a text field, focus on a control, control enabled and disabled etc. 6) **Recovery System** Provides support of built-in recovery system by some control, such controls are normally written by programmer by writing code. The control drives the application under test back to a known state, especially in the case where model dialogs often left open while execution of script, when a test case failure occurred. 7) **Custom Objects** Provide capability to deal with unrecognized objects on window. 8) **Technical Support** While using the trial versions and communicating with the vendor support site, I evaluated the quality of technical support. Because some time, vendor sale their product and later on just provided limited help. 9) **Reports** Verify that essential reports are generated by the application after different set of tests are	**Element 6**

executed.

10) Training Requirement
I went through the tools in detail to check if it is easy to learn for other test engineers considering timelines to acquire core competencies.

11) Un-attended Test Suite Execution
Ability to drive multiple test suites from GUI interface and command line, especially, when scripts are required to be executed un-attended.

12) Test case management support
Provide support for test case management, allowing each test engineer to execute single test or any combination of tests in full test suite for a given project.

13) IBM DB2 Data Base Support
Integrate with Core Banking system and can communicate with DB2 Database with the support of intermediate Data driver.

14) Data Driven Scripting
Provide ability to execute script calling external data source allowing to execute a test case with multiple types of data for each iteration.

15) Debugging Support
Provides debugging capability to help isolate scripts and run time errors.

16) Functionality and intended Audience
To evaluate the intended audience for the tool, whether it is purely for Test Technician who lack programming skills, Test Developers that have limited training with programming to develop tests or Test Architects who are quite competent in developing test framework and are recognized as an expert.

Comparative Analysis,

I engaged myself to examine the behaviour of different tools with Core Banking system that helped me to identify the required features and their applications. In this way, it also helped me to study various researches on latest technology and their applications that ultimately added value to my over all knowledge. Based on above analysis and thorough study and evaluation, I found three tools that supported Java swing application and support older version of JDK 1.3.

Element 8

a) Squish,

After evaluation, I found that It is a test automation tool for functional and regression testing of GUI application. It captures the properties of Java Swing Application components used in our application and develops script in python language. Python (Programming Language) expertise was required to use this tool to make complex tests that our applications demand. It had good support to record components on Java based GUI application and I was able to integrate with core banking application having support of JRE1.3 and above. This tool was highly complex and difficult to use, as it was able to record scripts in python language only.

b) Testing Anywhere by Automation Anywhere,

During Evaluation, I found that it records the script in Core Banking system but after carrying out in-depth analysis I found that it only supported JRE v 1.5 and above. It had an option called Smart Recorder that was used to record Java web objects (For Java Web Application). It had limited support to record Java Swing components for desktop Application therefore the script get recorded but failed to re-execute on our core banking system.

c) Quality First Test (QFT),

QFT was another tool for GUI test automation, which I evaluated in detail for functional GUI regression tests. It's important feature was that it properly captured the properties of Java Swing Application components of Core Banking system and was able to develop user-friendly GUI based JAVA script that was easy to understand and simple in modification. It also provide support for all JDK versions from 1.3 and onwards. It was also reliably recognise complex and dynamic Java UI components on screen. It also provided a good Test Management mechanism. It had a variety of UI controls, having built-in programming logics that could be used to implement complex scripting. Another good feature was its debugging and reporting mechanism. It also had support for IBM DB2 database, which was a core requirement in our banking system.

In this project, I learned to think rationally and applied judgment based on technical knowledge and exploration of tools and developed a comprehensive comparison among these tools given in below table.

Element 6

S.No	Key Features	QFT	Squish	Testing Anywhere
1	Support Script Recording on Core Banking System	Yes	Yes	Yes
2	Script Re-execution	Yes	Yes	Failed to execute
3	User Friendly Scripting (Easy to understand and modify)	Yes (Scripts are recorded using varieties of UI Controls having built-in programming logics)	No (Script recorded in Python Language. Expertise required)	No Script is recorded in proprietary language. Language expertise required.
4	Java Support 1.3 and above	Yes	Yes	No (Support JDK1.5 and above with limited Java swing support)
5	IBM DB2 Support	Yes (Using DB2 Data Driver)	No	No
6	Script Parameterization	Yes	Yes (Script recorded in Python. Language. Expertise required)	No
7	Recording Mechanism Used	Component Based	Component Based	X and Y Coordinate base and Limited Support for Components
8	Reusability of recorded script	Yes	Yes (Script recorded in Python. Language, Expertise required)	No
9	Robustness in Script Execution	Yes	Less than QFT	No
10	Adaptability - Easy to Use	Yes	No	Yes
11	Generate Test Report	Yes	Generate test results in XML File that can be utilized in report	Generate Test Results only
12	Support for Test Management	Yes	No	No
13	Email Support	Yes	Yes	Yes
14	Detailed Product Documentation available	Yes	Yes	Yes
15	Support in Pakistan	No	No	No
16	License Cost Per User	1995 EUR	2400 EUR	NA
17	Annual Maintenance Cost Per User	480 EUR	800 EUR	NA

Recommendation:

I learned and applied my analysis skills to identify pros and cons of above tools and based on their features, I applied my engineering knowledge and judgment to recommend the best solution by doing comparative analysis and cost benefit analysis. Finally I recommended Quality First Test tool for procurement with following key points,

Element 6

- Support Integration with JRE1.3 and above.
- Smart capturing of java based components while recording.
- Smart recognition of java based components on re-execution of scripts.
- It provides parameterization support in script that can be used to provide dynamic data from external source.
- In QFT, procedures developed can be re-used in entire Test Suite.
- It also provides Test Script Management mechanism.
- It produces execution report that provides details about the entire results of test suite execution.
- It supports data base connectivity with IBM-DB2.
- Also, provide support of External Data Source to develop data-driven scripts, for example MS Excel, CSV format, SQL script etc.
- It is easy to learn, adaptable and can help to create complex test cases using GUI controls.

Proof of Concept:

I gave detailed presentation and demonstration with proof of concept for which I prepared following test environment and scripts. Here my fundamental knowledge of database, operating systems has helped me to carry out the work.

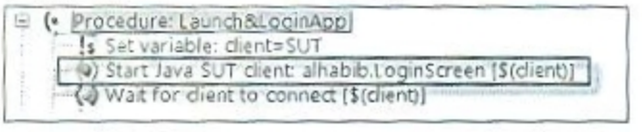
I developed a Test Environment that Contained following installations and configurations and establish connection with Core Banking system.

Element 3

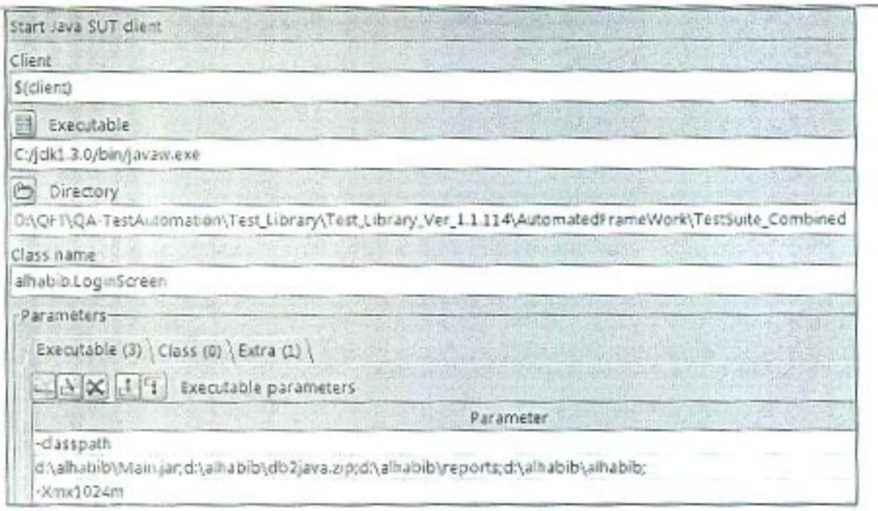
1. A system having Microsoft Windows 7 Installed.
2. Dummy Data Base of Core Banking System with IBM DB2 Setup.
3. Installation and Setup of Core Banking System with JDK 1.3
4. Installation and setup of Quality First Test (QFT) as trial version
5. Establish connection with Core Banking system with following steps

Step-1:

I created following script to establish QFT Connection with Core Banking System. This script is the part of Launch and Login Application script.



In above diagram on selected node, following is the configuration defined to establish connection with Core Banking system.



In this case, system is using JDK 1.3 version. If core banking system is enhanced to more upgraded version of JDK then QFT will also support.

Step-2,

After executing the above script, connection of QFT established with Core Banking System that is indicated by Red Mark Button enabled.

In above case, Core banking system is integrated with QFT with JDK1.3

Strategy:

I adapted following strategy to develop script as part of Proof of Concept.

1. Record script
2. Parameterized the script
3. Call the script in Test Sets

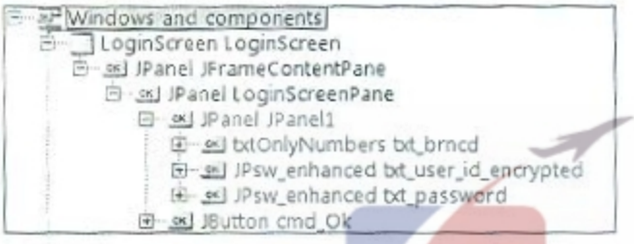
Business Case-1: Launching and Login Application:

I created following script to launch and Login the Core Banking System

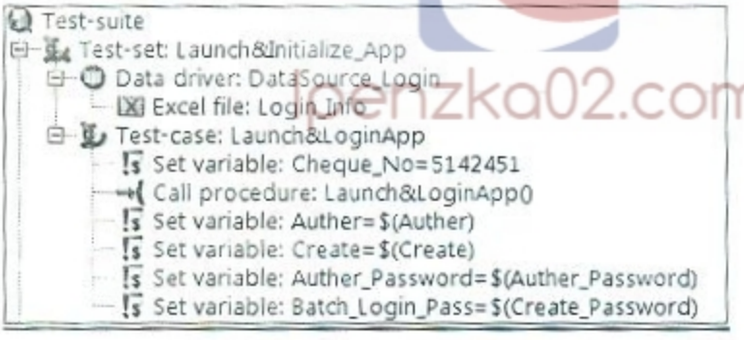
1. Record Launch and Login Procedure by interacting with application and providing Branch Code, ID and Password. After this I parameterized the script by replacing the values with variables as defined in external data source having input values, in this case external data source is excel sheet.

In above case, Script is parameterized that provide best mechanism to develop data driven scripts by defining various combination of transaction data and make script execute in one go that will call individual record of data and execute script producing separate result for each test data.

2. In this case, the Java components of Core Banking system are captured in the form components with all the properties.



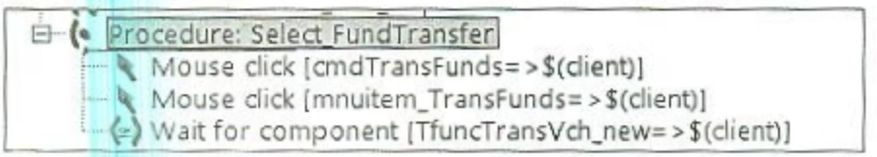
3. I created Test Set that contained the Data Driver and Source information in which I created a test case and called the above procedure.



In this case Test Suite contain Test Sets, in which test cases are created that called procedures recorded separately, provide best mechanism to organize the entire Test Automation Framework and make user convenient to locate to any script conveniently.

Business Case-2: Test Script of Fund Transfer from one Customer to Another:

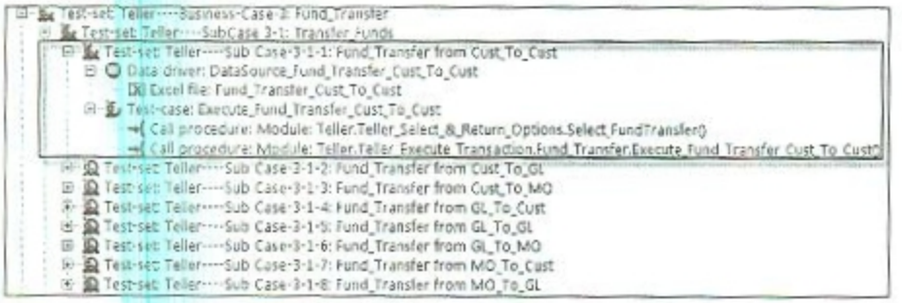
1. I created following script to execute Fund Transfer Activity for which first I created a procedure to navigate to fund transfer interface.



2. After this I created a procedure to record fund transfer activity as follows

In above script I used various programming logic controls to verify the business logics to make the script more complex to cover maximum test of fund transfer in a single script.

3. After that, I created Test Set that contained the Data Driver and Source information in which I created a test case and called the above procedure.



Business Case-3: Test Script of Booking Financial Deal:

1. I created following script to book financial deals.

In above case, Database connectivity with DB2 database (i.e. Core Banking db) was established. Following is the DB2 Driver configuration used with connection string.

In above case, script used input data directly from database using sql query and executed for each data set input.

Conclusion:

After several weeks of effort, this project was successfully completed and organization had carefully considered the need of test automation tool and finally procured the recommended software that is currently being used in organization for test automation framework.

Attached Documents:

- Proposal for Automation tool
- Recommendation Letter
- Capital Expenditure Request
- Email for Approval of Software Purchase.

Work/Study Episode 3:
Project Duration (April,2014 to April,2015)

Element

Overview of the project:

Bank Al Habib Ltd had procured third party Financial system for its Home Remittance Banking Business in order to facilitate transfer of money from an individual living overseas to any other individual living in Pakistan.

The third party system implementation project was initiated in Bank and it required Integration with Al Habib Core banking system (AHBS) for all modes of transactions. The Integration testing was the key part of the entire project implementation that required thorough testing of all modes of transactions through third part system integrated with AHBS.

Your role and responsibilities:

Following were my roles and responsibilities

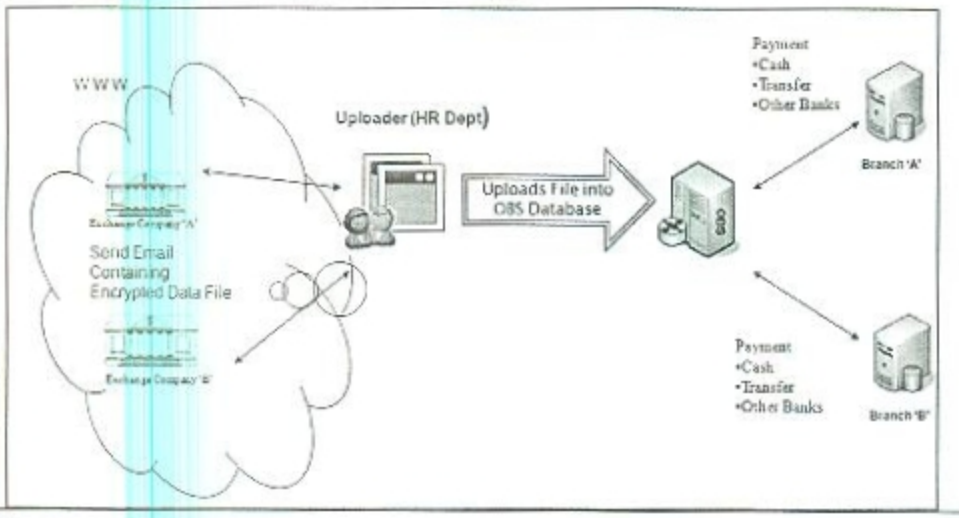
1. Develop sufficient understanding of Home Remittance business requirement and business process.
2. Deploy, Setup and configure the system on UAT Environment.
3. Execute functional testing of standalone third party system, verify all processes, and engage vendor to resolve issues in system.
4. To execute integration testing of third party system with Al Habib Core Banking System (AHBS) and resolve issues working with in-house integration developers.
5. To conduct User Acceptance Testing of the entire system with Business Users.
6. Deploy, Setup and configure the system on Live Servers.
7. Provide on premise training to users with third party system functionalities.
8. Resolve system operational issues.

Complexities (using the complexity definitions) and challenges of the project :

Background:

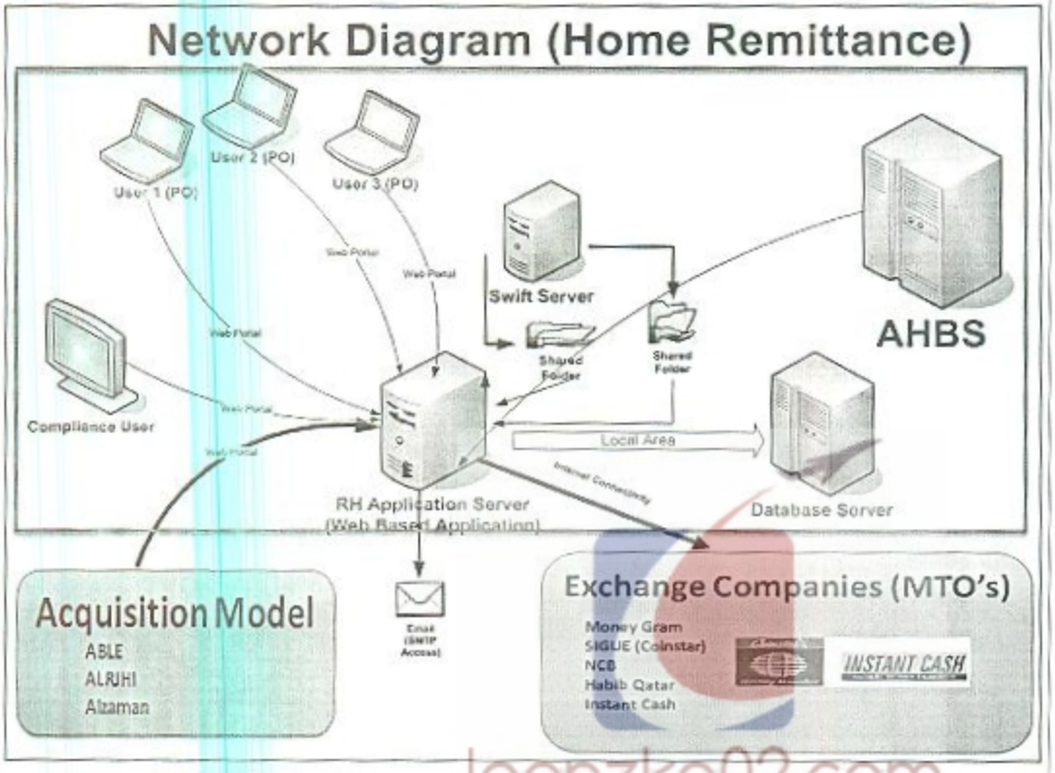
Home remittance is banking business. The business refers to transferring of money from a person (individual) to another individual through a bank or banking channel. Home Remittances are typically person-to-person payments via bank and in small amounts for personal use at their origin country.

Pictorial diagram of Home Remittance Existing Setup in Bank.



Bank Al Habib Ltd was already using a desktop-based application in order to execute this business. In year 2014 Bank Management had decided to expand the business by acquiring third party Home Remittance system, a web based solution that was capable to adapt Acquisition model with other exchange companies that desktop-based solution was not able to adapt. The main plan of adapting acquisition model was to start after the successful implementation of new Home Remittance (Web based) solution by replacing existing desktop based system

Pictorial diagram of New Home Remittance Application setup in Bank (Scope of this project is highlighted in blue box)



Major Complexities and Challenges:

Following were the major complexities and challenges of entire Project implementations.

1. To develop complete and in-depth understanding of entire Home Remittance business operations.
2. Conduct User Acceptance Testing of the entire system with Business Users and handle all face to face system related questions and queries as per the business aspect.
3. Perform functional testing of the third party system and ensure that it fulfils all the Home Remittance business needs that existing system provides.
4. Perform integration testing with in-house developed core-banking system (AHBS) and verify all modes of financial transactions with their financial entries and other financial details.
5. Deploy, Setup and configure the entire system on Live Servers.
6. Provide training and system understanding to users (Business Users, Compliance Users) within their premise and resolve operational issues.

How does this project demonstrate application of your engineering knowledge?

Understanding Business:
In order to functionally test the software system, it was essential for me to gained complete business process and requirement understanding. I found it very challenging to acquire the required knowledge and system understanding from business users, as they were non-technical having minimum IT background and there was a lot of resistance to accept new system. Here my analytical, leadership and communication skills had really helped me to gain the required knowledge that I utilized while working in close collaboration with business users. For which I went through following steps

Element 6

- Studied various system and procedure manuals about Home Remittance business.
- Involve with business users within their premise and gained the practical understanding of the business process and system flow of existing Home Remittance Software.
- Setup and configure the new system at UAT Environment and explore the system as per the understanding developed.

Here my specialized knowledge of requirement engineering and software system architecture and design has helped me to understand the business requirement and system architecture. Also my knowledge of inquiry based learning and problem solving ability, fundamental engineering knowledge like working with database, operating systems, system deployment had helped me to carry out the work.

Element 1, Element 3 and Element 4

After going through above steps, I found that in Home Remittance business, the exchange companies in different countries who have business ties with our bank send system generated transactions files (encrypted format) to our bank's Home Remittance department via authenticated communication channels e.g. Formal Emails or shared web folders with authenticated access.

The transaction files contain various numbers of different modes of transactions such that

1. Account to Credit Transactions (AC)
2. Cash over the counter transactions (COC)
3. Other Bank transactions (BT)

Account to Credit Transactions:
In this mode of transaction, the detail about the beneficiary account number with beneficiary title that exists in any branch of Bank Al Habib Ltd across Pakistan is provided with other customer details. When this transaction is process via system, it directly credits the beneficiary account with the remitting amount.

Cash over the Counter Transactions:
In this mode of transaction, A unique reference number (hidden) is provided in file that is already shared by the remitter (Sender) to beneficiary and other details. In this case, beneficiary visit to the branch and share the reference number providing all the secret details that branch user verify, after verification payment is delivered to beneficiary.

Other Bank Transactions:
In this mode of transaction, the details about beneficiary account that does not exist in Bank Al habib Branch network, however it belong to any other bank i.e. other then Bank Al Habib. In this case Bank, attempt to process such transaction via following ways.

- Bank Perform IBFT (Inter Bank Fund Transfer), a real time transaction via 1link that directly credit beneficiary account of another bank.
- If IBFT is not supported or IBFT failed to execute with Beneficiary bank, then system generate PRI files (Pakistan Remittance Initiative) and send it beneficiary bank via communication channels like email or shared folders.

Example of Exchange Company File

Following is the snap image of transaction file from an Exchange Company

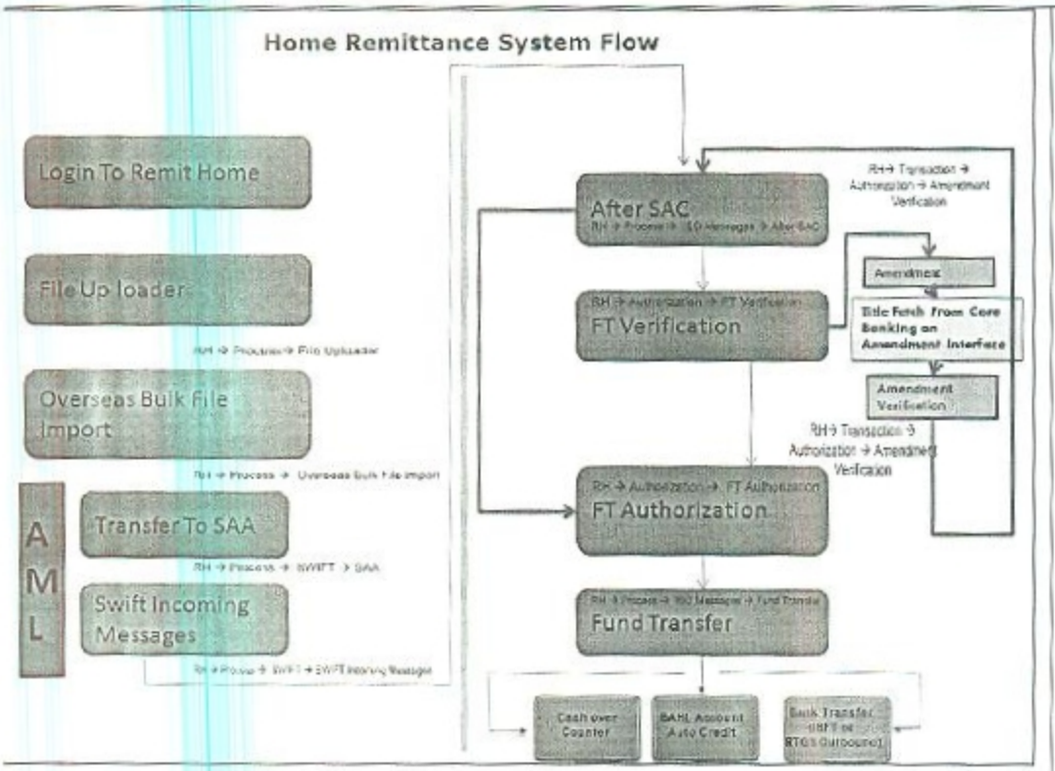
.....|| Bank AL Habib Limited/Able Express International Inc- Home Remittance ||.....
.....|| Export Activity Report ||.....
<Date/Time :8/5/2014 10:30 PM>

Total Exported Rows :50
Total Amount(in PKR) :2411852.00

Transaction Ref #:HR*********33
Remitter Name :Minhaj Uddin Shaikh

Developing System Understanding and Executing Functional and Integration Testing:

After gaining business understanding, I explored the new system after configuration and settings in User Acceptance Test Environment (UAT) and started testing the third party system functionally. During exploration I found that new system, support same business process flow as existing system. However, the new system was developed in Dot Net technology and it was web-based application, having various User Interfaces to perform business activities. I found it that this was the main reason bank was going for new web based system. Because due to web, system can directly be access by exchange company (if access is provided) and logged the transaction from their client interface over the internet that was not possible in desktop-based application (existing system) for which exchange companies had to send the encrypted files via emails. However, the plan of this acquisition model by providing direct access to exchange company was to be followed later on.



Steps of Developing System Understanding and Executing Functional and Integration Testing:

In each functional and UAT testing phase, my fundamental knowledge of computer engineering and software engineering with the capacity of Software testing and quality assurance knowledge and experience was fully applied.

Step-1: User Login to System:

User Login to system, and navigate to exchange file loading panel. This step also covered all the roles verification in the form of available options provided by the system administration. The roles may be Verifier or Authorizer of transactions.

Functional Testing and UAT:

During functional testing, I verified that

- User is able to login system
- Maximum five incorrect attempts to system login were allowed, after that password had to reset by Administrator.
- Application can be launch and login in any browser Internet Explorer, Google Chrome, Firefox etc. Here I executed browser based certification testing.
- The password of the user must follow complex requirement, For Example it must contain alphabets, numeric and special character.
- Application interfaces are allowed to assign to users according to their business role

Element 4

by application administrator.

I tested this feature with all the positive and negative test cases, verifying all possible scenarios. During UAT with business users, all major business cases were executed and business users were satisfied at this stage.

Step-2: File Up loader:

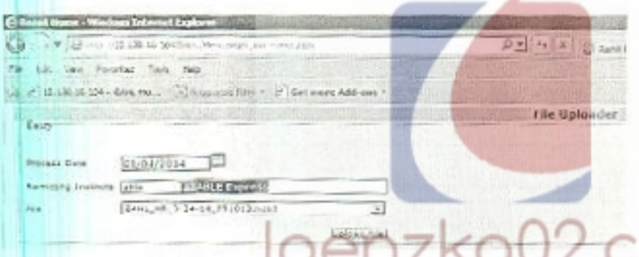
Here user load exchange company files in the system by navigating to File Up-loader panel. Doing File uploading, the exchange files placed in a specific folder defined in system parameter. The exchange company files contained AC, COC and Other banks transactions records.

Following is the sample of transaction information file.

.....|| Bank AL Habib Limited/Able Express International Inc- Home Remittance ||.....
..|| Export Activity Report ||.....
<Date/Time :24/3/2014 11:10 PM>

Total Exported Rows :43
Total Amount(in PKR) :2413194.00

Transaction Ref # HR********33
Remitter Name :Afia Ali Khan
Remitter ID :3467
Currency PKR
Remit Amount :686.13
Remit Amount(PKR) :67620.00
Beneficiary A/C# :01530000004731
Pay Mode :3
Transaction Date :24/03/2014



Functional Testing and UAT:

During functional testing, I verified that system

Element 4

- Allow to load files with various combinations of exchange companies.
- System produce proper error, if an exchange company file is loaded in incorrect Remitting Institute and provides proper way to user to correct the mistake. If any error occurs due to incorrect file format, then system provide proper way to rectify those errors.
- System allows loading multiple files in appropriate time.

In this file loading panel, user were satisfied with the file loading mechanism but not getting satisfied with the errors due to any mistake that user could do. However, when I explain them with proper justification about the reason of those errors occurred and shown them the mechanism to rectify those errors, they got satisfied. After detailed discussion, it was