|  |
| --- |
|  |
| Effort Estimate for Technology Refresh XXXX application |
| Analysis Document |

|  |
| --- |
| Kumar Sinha, Ajoy  6/14/2017 |

Version 0.6

This is a Work in Progress Document will be release with Version 1.0

### Content

1 Scope 2

2 Sequence of Work item Execution and Application Migration 3

3 Migration into JBOSS platform 7

4 Change JDK version from 1.5 and 1.6 to 1.8 9

5 Upgrade Spring framework 1.2.9 to Spring 4.x 10

6 Junit Test cases need to be developed 12

7 Convert ANT build into Maven Build 13

8 Convert code repository from Clear Case into GitHub 14

9 Database Migration (From Microsoft SQL Server to Oracle MySQL) 15

10 Effort Calculation 16

# Scope

Below work items I have considered as scope of work

* + - * 1. **AVF application need to be migrated into JBOSS platform**
        2. **Change JDK version from 1.5 and 1.6 to 1.8**
        3. **Upgrade Spring framework version to Spring 4**
        4. **Junit Test cases need to be developed**
        5. **Convert ANT build into Maven Build**
        6. **Shift code repository from Clear Case into GitHub**
        7. **Database Migration from Microsoft MSSQL to Oracle MySQL**

Following are the list of Projects / Applications / Components of AVF to be converted

* + - 1. BlueMobileEJB
      2. AVFExtranet
      3. BlueMobileScheduler
      4. AvfAdminConsole
      5. BlueMobileSale
      6. BlueMobileTestHarness

Following are the list of Projects which can be analysed and discarded if not required at all

1. proxyCheck
2. SwiftPassTest
3. TributeInterfaceLoader
4. ctrExample
5. BlueMobileHandsetChecker

# Sequence of Work item Execution and Application Migration

Work Item 0: **As is Application setup and run in local environment**

Work Item 1: **Convert ANT build into Maven Build**

Work Item 2: **Shift code repository from Clear Case into GitHub**

Work Item 3: **Change JDK version from 1.5 and 1.6 to 1.8**

Work Item 4: **Upgrade Spring framework version to Spring 4**

Work Item 5: **Migration to JBOSS**

**Migration sequence for Applications depending on internal dependency**

**Application 1:** BlueMobileEJB

**Junit Test Case for Blue Mobile**

**Application 2:** AVFExtranet

**Junit Test Case for AVFExtranet**

**Application 3:** BlueMobileScheduler

**Junit Test Case for BlueMobileScheduler**

**Application 4:** AvfAdminConsole

**Junit Test Case for AvfAdminConsole**

**Application 5:** BlueMobileSale

**Junit Test Case for BlueMobileSale**

**Application 6:** BlueMobileTestHarness

**Junit Test Case for BlueMobileTestHarness**

Work Item 7: **Database Migration from Microsoft MSSQL to Oracle MySQL**

Work Item 6: **Junit Test cases need to be developed as part of Application Migration Cycle**

**Alternate Plan - A**

**Phase 1:**

Work Item 0: **As is Application setup and run in local environment**

Work Item 3: **Change JDK version from 1.5 and 1.6 to 1.8**

Work Item 4: **Upgrade Spring framework version to Spring 4**

Work Item 5: **Migration to JBOSS**

**Migration sequence for Applications depending on internal dependency**

**Application 1:** BlueMobileEJB

**Junit Test Case for Blue Mobile**

**Application 2:** AVFExtranet

**Junit Test Case for AVFExtranet**

**Application 3:** BlueMobileScheduler

**Junit Test Case for BlueMobileScheduler**

**Application 4:** AvfAdminConsole

**Junit Test Case for AvfAdminConsole**

**Application 5:** BlueMobileSale

**Junit Test Case for BlueMobileSale**

**Application 6:** BlueMobileTestHarness

**Junit Test Case for BlueMobileTestHarness**

Work Item 7: **Database Migration from Microsoft MSSQL to Oracle MySQL**

Work Item 6: **Junit Test cases need to be developed as part of Application Migration Cycle**

**Phase 2:**

Work Item 1: **Convert ANT build into Maven Build**

Work Item 2: **Shift code repository from Clear Case into GitHub**

**Risks: Business also may not execute Phase 2 at all, as business functionality will be achieved by Phase 1 only. Phase 2 will mainly ensure reduction of Technology debt and align AVF with enterprise roadmap fully.**

**Alternate Plan - B**

**Phase 1:**

Work Item 0: **As is Application setup and run in local environment**

Work Item 3: **Change JDK version from 1.5 and 1.6 to 1.8**

Work Item 4: **Upgrade Spring framework version to Spring 4**

Work Item 5: **Migration to JBOSS EAP 7** (Migration sequence for Applications depending on internal dependency)

**Application 1:** BlueMobileEJB

**Application 2:** AVFExtranet

**Application 3:** BlueMobileScheduler

**Application 4:** AvfAdminConsole

**Application 5:** BlueMobileSale

**Application 6:** BlueMobileTestHarness

Work Item 7: **Database Migration from Microsoft MSSQL to Oracle MySQL**

**Phase 2:**

Work Item 1: **Convert ANT build into Maven Build**

Work Item 2: **Shift code repository from Clear Case into GitHub**

Work Item 6: **Junit Test cases need to be developed**

**Migration sequence for Applications depending on internal dependency**

**Application 1: Junit Test Case for Blue Mobile**

**Application 2: Junit Test Case for AVFExtranet**

**Application 3: Junit Test Case for BlueMobileScheduler**

**Application 4: Junit Test Case for AvfAdminConsole**

**Application 5: Junit Test Case for BlueMobileSale**

**Application 6: Junit Test Case for BlueMobileTestHarness**

**Risks: Alternate Plan B will not support Test Driven Methodology as we are suggesting developing Test Cases in phase 2. This will happen after migration. Business also may not execute Phase 2 at all, as business functionality will be achieved by Phase 1 only. Phase 2 will mainly ensure reduction of Technology debt and align AVF with enterprise roadmap fully.**

# Migration into JBOSS platform

Following high level changes required for this migration

* EJB Conversion
* Queue Conversion
* Topic Conversion
* Web Service Conversion
* Remove Weblogic references

We have identified over 500 places where we need to do changes for this migration

EJB Conversion:

#### weblogic.jndi.WLInitialContextFactory need to be convered into org.jboss.naming.remote.client.InitialContextFactory

#### weblogic-ejb-jar.xml need to be convered into [jboss-ejb3.xml](https://docs.jboss.org/author/display/AS71/jboss-ejb3.xml+Reference)

#### Session ejb can to be converted into

#### Stateless(mappedName = "name")

#### @Remote(class name)

#### @TransactionAttribute(TransactionAttributeType.REQUIRED)

#### @Clustered

#### @Pool(value=”poolname”)

Queue Conversion

#### @MessageDriven(

#### name = "queuename",

#### activationConfig = {

#### @ActivationConfigProperty( propertyName = "destinationType",

#### propertyValue = "javax.jms.Queue"),

#### @ActivationConfigProperty( propertyName = "destination",

#### propertyValue ="/queue/queuename")

#### }

#### )

Topic Conversion

#### @MessageDriven(

#### name = "topicname",

#### activationConfig = {

#### @ActivationConfigProperty( propertyName = "destinationType",

#### propertyValue = "javax.jms.Queue"),

#### @ActivationConfigProperty( propertyName = "destination",

#### propertyValue ="/queue/topicname")

#### }

#### )

EJBGen can be changed using XDoclet. And can be compiled with the help of Maven.

Other changes required will be identified using windup and code analysis.

As we have found Spring and EJB (Two container framework) are co-existing into AVF application, we have considered migration of EJB into Spring framework fully. In that case EJB layer and related components features need to be re-developed. This will also include large amount of effort. After analysing scratch development and related effort we haven’t consider this approach in final migration approach. Although this is a point of discussion and can be considered if directed.

# Change JDK version from 1.5 and 1.6 to 1.8

As Java provide backward compatibility, hence major refactoring may not be required for this JDK upgrade.

If any major change required, that will be analysed and done during migration.

Upgradation to JDK 1.8 is considered to reduce technology debt for having older Java / JDK versions and also to ensure that all modules under AVF will have same execution platform [Currently majority of module is using JDK 1.5 and one module is using JDK 1.6]. This migration will only consider upgradation of platform; we do not consider any re-write of code to induce JDK features like Lambda Expression.

# Upgrade Spring framework 1.2.9 to Spring 4.x

Spring 1.2.9 version is an old framework version, this need to be upgraded to reduce technology debt. We are recommending changing those components which are required to support Spring 4.x version. Current stable release 4.3.9 of Spring Framework can be adopted for migration. Final Spring migration version will be confirmed during analysis and execution phase.

Basic requirement for Migrating to Spring Framework 4..x is depends on JDK 6. Hence JDK upgradation is primary task need to be executed during Spring upgradation.

Following need to be ensured during Spring Framework upgradation.

* Servlet 3.0 *(2.5 supported for deployment)*
* JPA 2.0
* Bean Validation 1.0
* JSF 2.0
* JCache 1.0

#### Servers

* JBoss EAP 6 recommended

#### Libraries

* Hibernate Validator 4.3
* Hibernate ORM 3.6.10 (note: phasing out as of Spring Framework 4.2, with Hibernate 4.2/4.3 recommended)
* Apache HttpComponents 4.3 (required for Spring's http.client package, and for all of Spring as of 4.1.4)
* EhCache 2.4.7 (note: minimum 2.5 as of Spring Framework 4.1, with EhCache 2.8 or later recommended)
* Quartz 1.8.6 (note: minimum 2.1.4 as of Spring Framework 4.1, with Quartz 2.2.1 recommended)
* Jackson 1.8.6 (note: minimum 2.1 as of Spring Framework 4.1, with Jackson 2.3 or later recommended)
* Joda-Time 2.1 (note: 2.3 or later recommended)
* XStream 1.4
* Apache POI 3.8
* JUnit 4.7

Other changes required will be identified using code analysis and migration

# Junit Test cases need to be developed

Junit version need to be upgraded from **4.1 to 4.7**

**As Spring 4.x recommend to use Junit 4.7, we are also planning to migrate into Junit 4.7**

1. BlueMobileEJB

Around 400 Test cases already been developed and those need to upgraded to 4.7

1. BlueMobileHandsetChecker

Around 130 Test cases need to be developed; these are for Service implementation classes only.

1. BlueMobileTestHarness

Around 20 Test cases need to be developed; these are for com.atosorigin.bluemobiletestharness.dal classes only.

1. AVFExtranet

Around 50 Test cases need to be developed; these are for Service implementation classes only.

1. AvfAdminConsole

Around 60 Test cases need to be developed; these are classes within com.worldline.avf.adminconsole.logic only.

1. BlueMobileScheduler

Around 25 Test cases need to be developed; these are classes within com.atosorigin.bluemobile.scheduler.events only.

1. BlueMobileSales

Around 180 Test cases need to be developed; these are classes within com.atosorigin.bluemobile.helpers only.

Other changes required will be identified using code analysis and migration

# Convert ANT build into Maven Build

**During conversion of build technology we have considered Maven and Gradle Build tool. But as few of existing project already follow Maven build, hence we suggest migrating all the projects into Maven. This consideration will reduce Build tool conversion effort.**

Following conversion is required for different projects.



Steps to be followed:

1. Create a pom with groupId, artifactId and version (packaging: war)
2. Add the required dependencies to the pom
3. move the
   1. java sources to src/main/java,
   2. resources to src/main/resources,
   3. webapp content to src/main/webapp,
   4. test content to src/test/java and src/test/resources
4. Update Project
5. Build project

Repeat number 2 to 5 till project compiled and builds

Other changes required will be identified using code analysis and migration

# Convert code repository from Clear Case into GitHub

1. Baseline Codebase in Clear Case
2. Get the latest version from Clear Case.
3. Clear codebase and remove reference of Clear Case
4. Check in codebase into Git.

History of the Clear Case migration into Github has not been considered. This will be a fresh start of code management after Technology Refresh of AVF and migration into github. We can use clearcase in read only mode and refer to history whenever required. And retire it after certain amount of time

Other changes required will be identified using code analysis and migration

# Database Migration (From Microsoft SQL Server to Oracle MySQL)

**Oracle MySQL 5.7 Enterprise Edition** will be used as AVF database.

Existing Microsoft SQL Server 2005 objects and data will be migrated into **MySQL 5.7 Enterprise Edition**.

Following migration steps will be followed for Database Migration

1. Analysis and Define Data Dictionary (MSSQL)
2. Analyse and compare Target Data Types and Compatibility (MySQL)
3. Configure Migration Tool
4. Run Migration scripts
5. Fix Migration Errors and execute
6. Object and Data Validation

**Migration Tool: MySQL workbench will be used for data migration**

Application retrofit for DB Migration

1. Analyse and change (if required) in java files ( 361 files need to be revisit )
2. Test Application

# Effort Calculation

Above tasks have been analysed by following manners

* + - * 1. JBOSS Windup migration tool: Windup is a tool to simplify Java application migrations. The tool analyzes application artifacts (such as Java code, JSPs and XML) and produces an HTML report highlighting areas that require changes.
        2. High Level Code Analysis
        3. Using KT documents
        4. Discussion with Avanish and Pooja
        5. Past Experience
* **Migration Task**

Work Item 1: **Convert ANT build into Maven Build**

Work Item 2: **Shift code repository from Clear Case into GitHub**

Work Item 3: **Change JDK version from 1.5 and 1.6 to 1.8**

Work Item 4: **Upgrade Spring framework version to Spring 4**

Work Item 5: **Migration to JBOSS**

Work Item 7: **Database Migration from Microsoft MSSQL to Oracle MySQL**

**Effort Required ~ 1979 PH (*Around 300 Change points identified)***

* **Junit Creation Task**

Work Item 6: **Junit Test cases need to be developed**

**Effort Required ~ 2314 PH ( *Around 430 test case need to created )***