Sapient Exercise Code Example

Lift Engine Simulation

0 Overview

0.1 Purpose of this document

This document is meant to give detailed information of the Lift Engine Simulation Coding and Testing Process Process. It provides a brief explanation application and software stack and coding practices.

## 0.2 Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| LFE | Lift Engine Simulation |
|  |  |
|  |  |
|  |  |
|  |  |

# Application stack

## IDE

Eclipse Version: Kepler Service Release 2

Build id: 20140224-0627

## build Tools

Maven 3.X

Reporting and Cobertura Plugins

Dependency Administration

JAR Package and Console Application

## Java version

Compiles 1.7+

## Java Features

Java Core

Java Collections and Concurrent

List, ArrayList in Thread Safe Operations

CopyOnWriteArrayList in Concurrency Operations

Java Threads

## other frameworks

Spring 4.x

Dependency Injection

Memory Administration

Java Based Configuration xml-less

Unit Testing support

JUnit and Mockito

Unit Testing

Apache Commons

String Thread Safe and Null Safe Operations

Log4J

Log diagnostics to Console

## Some Desing Patterns

Singleton based in Spring Beans

Listener based in Shared Objects

MVC Approach with some variations

Consumer-Producer approach and synchronization

## Some coding best pratices

Uses of Interfaces rather than Abstract Classes

Adding Unit Testing to most of the modules

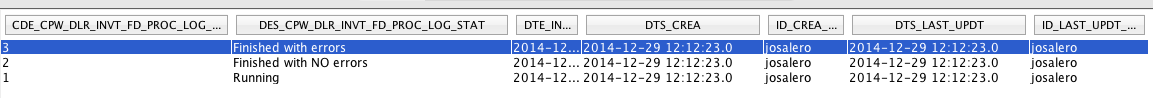
Use of Concurrent Data Structures and Coding in Thread Safe Mode

# Deployment process

## pre-conditions

### Validate the existence of 3 rows corresponding to the different log status definition for every of the feed processor executions in the table "CPWDBADM"."CPW\_DLR\_INVT\_FD\_PROC\_LOG\_STAT\_CDE"

- If Not: Please include 3 rows like the following example:

**

* The *audit username* will be used in the filter property ***audit.user*** for every maven profile. In the sample provided, **josalero** corresponds a user that actually is used however a more accurate one can be created.

## Deployment steps

Check out the branch <http://s171a046.mbusa.corpintra.net/svn/repo1/code/third-party/inventory-feed-processor/branches/0.1.0/> in to a local folder.

* + - This step should be done only once and when you have the source code already checked out, please run the command “svn update” to get the latest.

To build the feed processor and create a war file ready to be deployed please consider the following steps:

* + - mvn clean install -P [Maven Profile (defined in 1.3.2)].

Once the step b) finishes, please navigate through mb-fp-web/target and you will find the **mb-feed-processor.war** file.

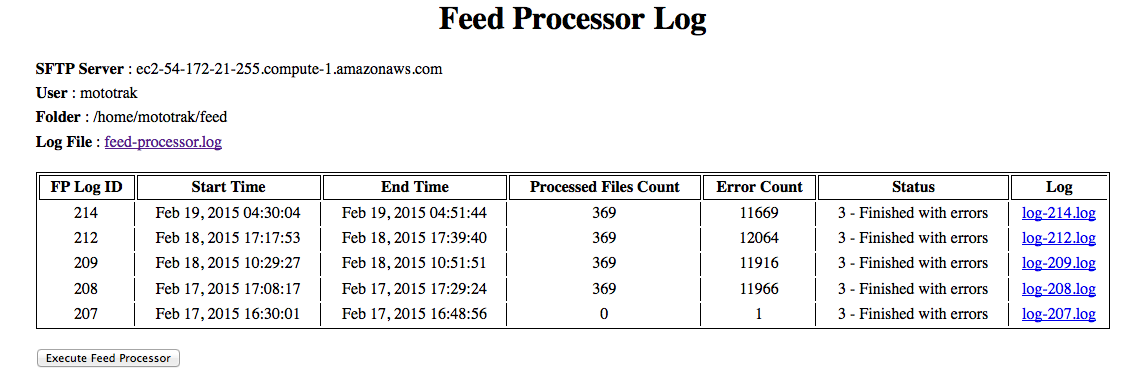
Please make sure tomcat server is down. If not, please stop the tomcat server.

Take the file **mb-feed-processor.war** and move it in your tomcat/webapps folder

Configure the application to run in ROOT.

Start the tomcat server

In order to validate if the application was correctly deployed and working, please consider to browse the page <http://127.0.0.1/feed-processor>. You will get a landing page like the following:

**