Application of Microservice Architecture on B2B Processes

(IBM Watson Customer Engagement)

by

Vipin Dhonkaria

(Roll No. 2015274)

Supervisor(s):

External

Mr. Atul A. Gohad

(IBM ISL, Bangalore)

Internal

Dr. Manish Kumar Bajpai

(PDPM IIITDM Jabalpur)

Computer Science and Engineering

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, DESIGN AND MANUFACTURING JABALPUR

(9thst October 2018 – 19th October 2018)

Introduction

The International Business Machines Corporation (IBM) is an American multinational technology company headquartered in Armonk, New York, United States, with operations in over 170 countries. IBM manufactures and markets computer hardware, middleware and software, and provides hosting and consulting services in areas ranging from mainframe computers to nanotechnology.

IBM aims to bring Businesses closer and smarter than ever with the help of their state of the art enterprise software product called B2B Sterling Integrator. IBM B2B Integrator helps companies integrate complex B2B (Business to Business) / EDI (Electronic Data Exchange) processes with their partner communities. IBM aims to transform the B2B Sterling product into Microservice architecture.

Brief Overview

During the duration of my last report, I was involved in converting a few B2B APIs like **UserVirtualRoot** and **UserAccounts** to Spring Boot Rest API Framework and deploy it in the form of a WAR file on the liberty server of the Sterling Integrator. Also, I performed benchmarking of both versions of the API - SpringBoot and the normal TenX. SpringBoot was nearly 12 times faster for bulk level calls. This is a huge improvement. My team will consider pitching this idea to the stakeholders and then eventually converting all the B2B APIs to SpringBoot.

The target of this report is to create a new service for Sterling Integrator – JSON/XML Transformer. This will allow the customers to use inbuilt XML/JSON service for their data conversion needs. Also, I worked upon the Rest API client service (Pre-Midterm) for making it more production ready as it is going to be passed to the QA (Quality Assurance) stage for final testing.

Report on the Present Investigation

(Progress during this 10-days period)

Orchestration of JSON/XML Transformer service.

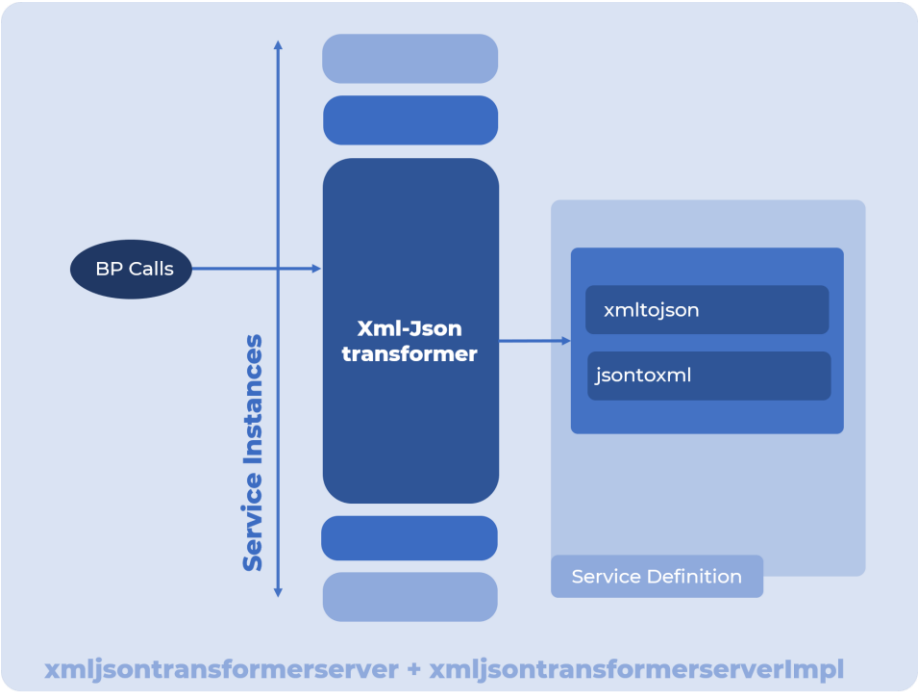
Structural Organization of the Service

The file xmljsontransformer.jar provides the class definition. This JAR file contains following three files (In the proper package hierarchy)–

```
com
   sterlingcommerce          woodstock
                services
                     xmljsontransformer
                                  xmljsontransformer.java
xmljsontransformerserver.java                    xmljsontransformerserverImpl.java
```

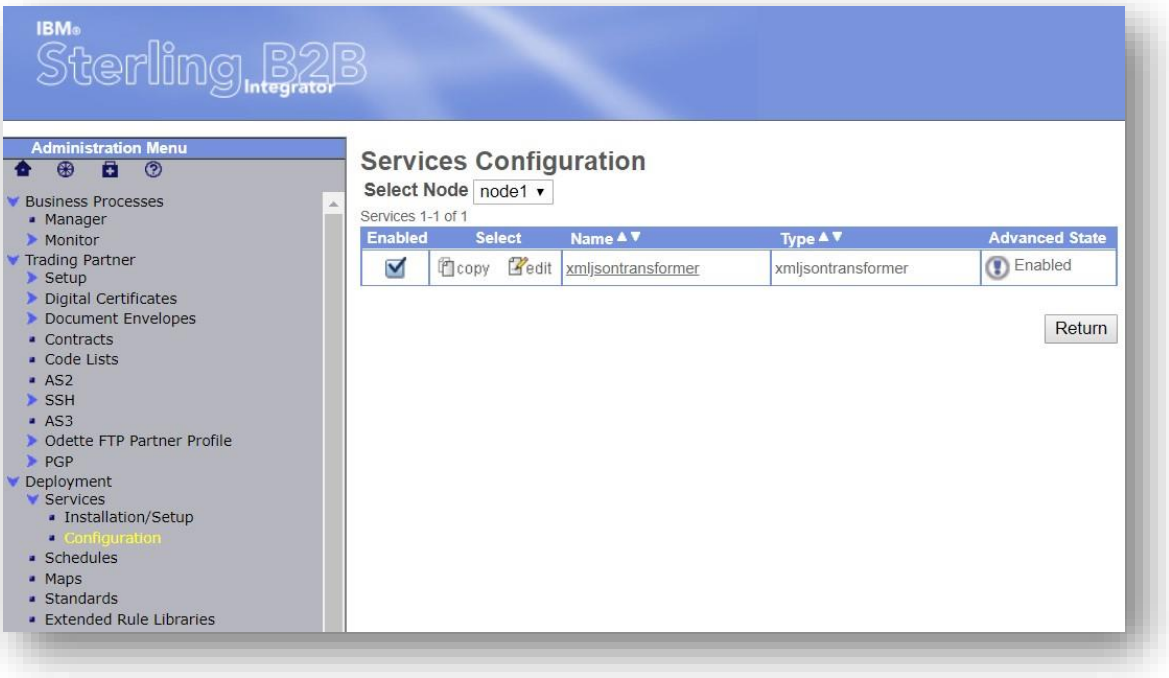
.

xmljsontransformer.java – The service which actually acts as a gateway to handle the upcoming converting requests depending upon the input parameters from the process data. **xmljsontransformerserver.java –** The java file that is responsible for restarting, shutdown; etc of the xmljsontransformer service. **xmljsontransformerserverImpl.java –** The java file that implements the xmljsontransformerserver class.



Running after Installing the Service to the SI Server.

After installation, on the UI, A service called **xmljsontransformer Service** will be shown created in the SI dashboard. This service can be seen by going to the **Deployment>>Services>>Configuration** from the left panel menu.



A workflow (business process) called **xmljsontransformer** will also be created. You can test the service by executing this business process. The response from the Service will be specified in the Process Data itself.

Business process has been created.

Variation in Workflow (BPs) to carry out diverse Operations

Depending upon the requirement, Business Processes can be modified to perform requests. All the BPs must provide following 2 mandatory parameters:

o **Outputpath:** Path of some directory on the server o **Processtype:** XMLtoJSON or JSONtoXML o **Input:** Data to be converted

(Input assign statement can be omitted when supplying input data from file. Instead of Input assign statement, just add the following line to the BP)

```
<assign to='.' from='PrimaryDocument/@SCIObjectID'></assign>
```

XML to JSON Request

- BP Structure

Note: In case of XML as an Input, the Angular Brackets have to be replaced with escape characters as follows:

1. '<' with '<'
2. '>' with '&gt'

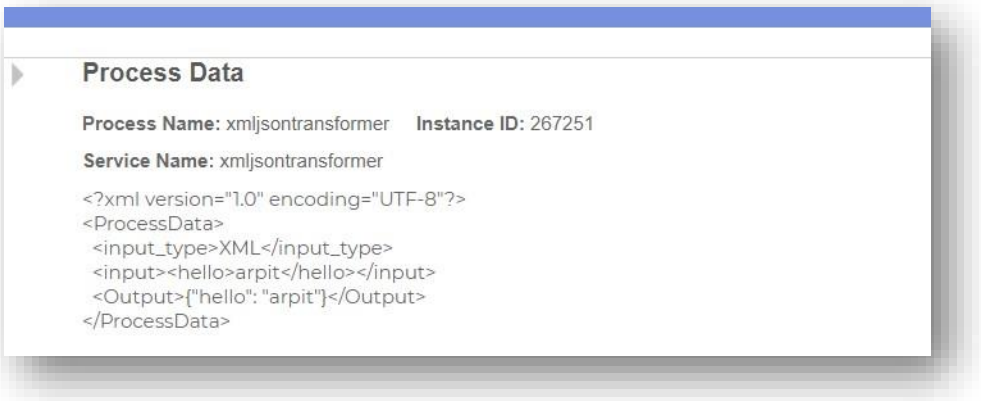
This has to be done because if a tag is inputted as **<tag>tagdata<tag>**, then it causes XML Validation error and causes BP Syntax error.

This has to be written as **<**tag**>**tagdata**<**tag**>**

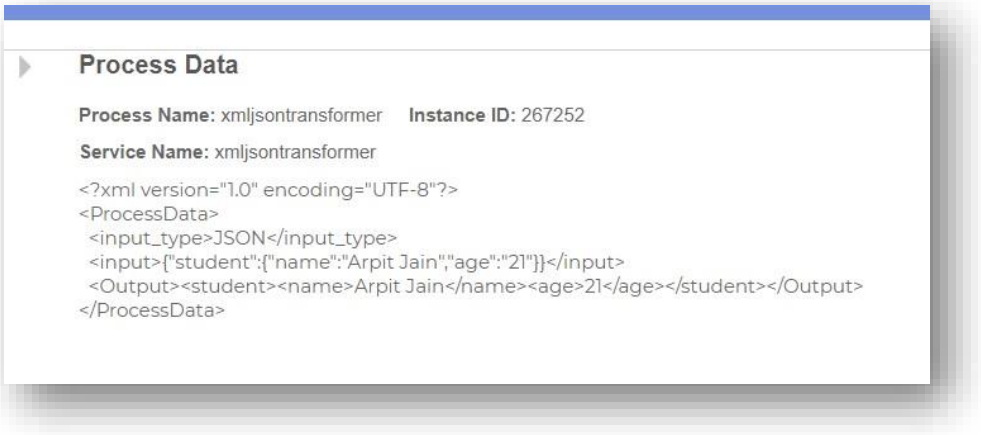
```
<process name="xmljsontransformer">
 <sequence>
      <operation name="Request">
       <participant name='xmljsontransformer'/>
       <output message='xout'>
        <assign to='input'>      &lt;hello&gt;arpit&lt;/hello&gt;     </assign>
        <assign to='proces        stype'>                    </assign>
        <assign to=’outputpath          >XMLtoJSON  </assign>
                                          ./output.json
        <assign to='.' from='*' />
       </output>
       <input message="xin">
        <assign to='.' from='*' />
       </input>
                                                    </operation>
 </sequence>
</process>
```

JSON to XML Request

- BP Structure

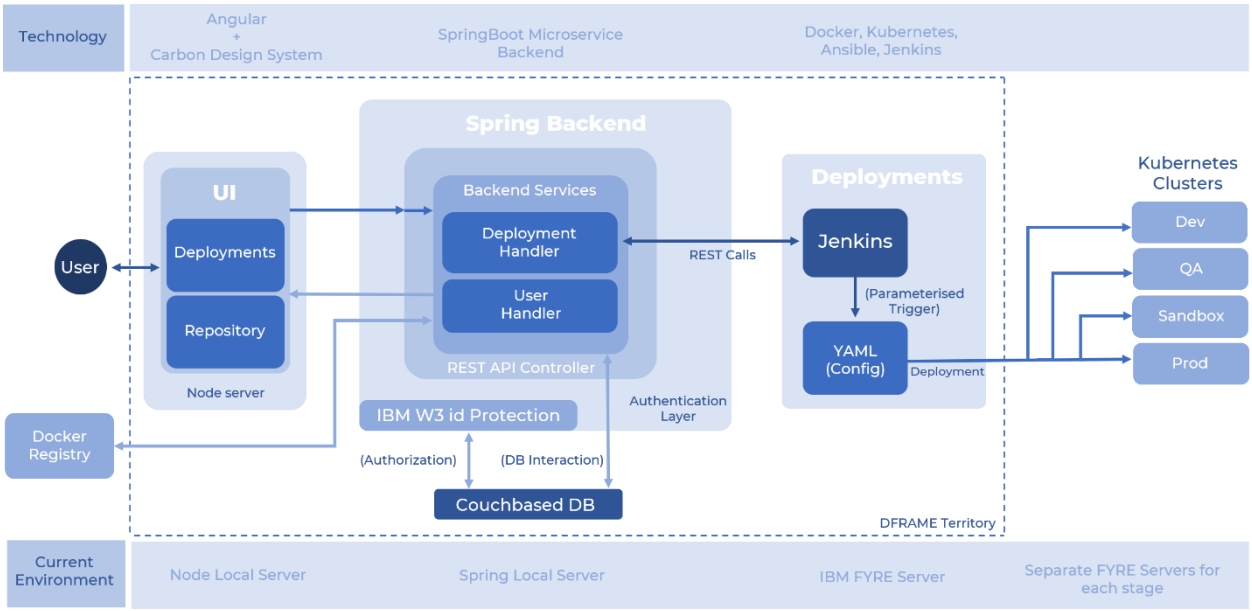


```
<process name="xmljsontransformer">
 <sequence>
      <operation name="Request">
       <participant name='xmljsontransformer'/>
       <output message='xout'>
        <assign to='input'>{"student":{"name":"Arpit Jain","age":"21"}}</assign>
        <assign to='processtype'>JSONtoXML</assign>
        <assign to='outputpath'>../output.xml</assign>
        <assign to='.' from='*' />
       </output>
        <input message="xin">
         <assign to='.' from='*' />
        </input>
      </operation>
 </sequence>
</process>
```



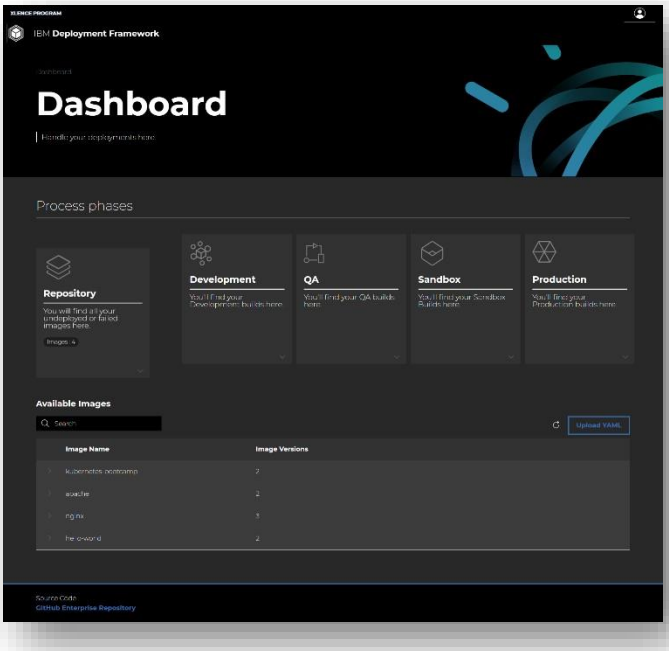
XLENCE Program – Deployment Framework

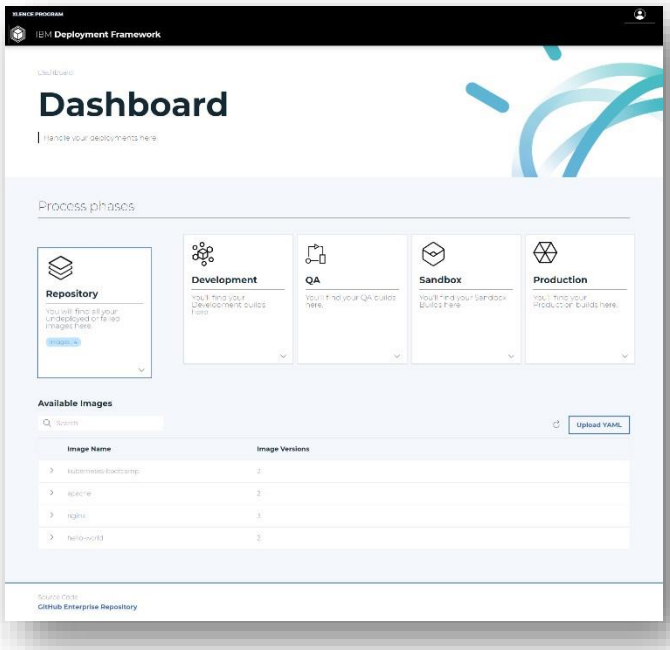
As a part of the XLENCE Program for the IBM interns, we are working on a Docker image deployment framework that builds and deploys an Application Image in just a few clicks from the intuitive UI. Following figure (Updated from the last report) will illustrate the exact functioning and architecture of this application.



We have had meetings with the Stakeholders- Product Architects, Managers and Program Director where we demonstrated this product in an extensive demo session. In further meetings after more refinement to the functionality, this product might be sent for production.

Following are the screens for this product.





Results and Discussions

Before Midterm, I built a REST Client Adapter for the Sterling Integrator which allowed the users to place REST calls over Business APIs. Earlier, these APIs only accepted JSON as input. Using this XML JSON Transformer service as an intermediate layer, users will have the versatility to provide input in both formats. Although JSON is the accepted standard for API inputs and outputs, still the users of Sterling Integrator require this feature as Business Processes and some other important components of the product follow XML standards.

Conclusions

During past few weeks, I was involved in creating a new service for the Sterling Integrator. This Service is called XML JSON Transformer. This service would allow the user to transform the data from XML to JSON and vice-versa. Using this service, input could be taken from the user in any format and using this service, it can be converted to the opposite format and then be supplied as input to some other service. This small but really useful service is an integral part of what the users were really demanding.

Next Target

My target for the next 15 days is to transform **UserAccounts** to the Spring Boot framework. Currently, SpringBoot version of the UserAcounts API is just capable of a GET Call. My task would be to implement and add other calls like POST, PUT and DELETE to this API (All the CRUD operations).