a

MicorServices

===:

=>This needs the knowlege of webservices (especially restful webservices) So we learn Spring Boot Rest first to get into Micro Services Subject

=> Spring Boot Rest is used to develop restful web services nothing but Distributed App Development (App to App interaction)

Back ground preparation for Distributed Apps / webServices Java is platform independnt and Architecture neutral language (OS) (Computer Architecture)

=> The plan of manifacturing vehicle is called vehicle architecture like that four wheeler architectures are

a) Car architecture

b) Jeep architecture

c) Van architecture

d) Mini Bus Architecture

e) Mini Lorry archicture

and etc..

Spring boot rest spring boot mvc++ Spring Rest = spring mvc++

(officially there is no module called spring Rest/spring boot rest .. it is spring mvc/spring boot mvc which we use to develop the Restful webservices)

=> The plan /process of manifacturing computer is called computer archicture and the popular architectures are

=> IBM Architecture (all our reqular laptops and desktops fall under this archicture) => Apple/MAC architecture

=> Sun Architecture

and etc..

=> c,c++ languages are not only Platform dependent (OS) and they are also

archicture dependent where java is platform(OS) and archicture independent

.c/cpp (source file)

↓compile

.obj (object file)

.exe (executable file)

[Windows OS - IBM architecture)

.java (sourcefile)

compile

.class (byte code)

(Windows OS -IBM architecture)

=>browser to App interaction is called website

eg: browser to flipkart.com (client is browser)

=> App to App interaction is called distributed App eg: flipkart.com to phone App (Here client can eg:

phoneApp to Our Bank App

This indicates, we can use

spring MVC/spring Boot MVC to develope both web applications and the Distirbuted Apps

OS = Operating System like windows, linux and etc..

be any software app)

Execution

Computer1

yes

Java is platform independent

yes

Windows OS -IBM Architecture

and architecturally neautral/ architecture independent

Computer2

yes

no

MAC OS --Apple

architecutre

Computer3

no

yes

Sun Solaris OS -

Sun Architecture

Computer 4

yes

no

Linux OS Architecture

-IBM

Computer 5

no

yes

window OS - Sun Architecture

execution

=>Computer platform is nothing but Computer OS => Computer architecture is nothing but the way hardware comps are assembled following certain process/principles is called computer architecture

browser -- websiste /App interaction :: web application

eg: browser to nareshit.com

App to App interaction:: Distributed App nareshit.com ---> gpay/

gpay ---> BankApp

Enterpise Application :: web application + Distributed App

eg:: flipkart with card payment

nareshit with UPI Payment

web application

browser ------

Enterprise App

distributed App

(consumer App) Flipkar.com (e-commerce site)

Distrubuted App

(consumer App)

I cel

(producer App)

payPall comp

(payment broker)

Distributed App

(producer App)/(consumer App)

--> VISA/Master/.. App -------- --> SBI/ICICI/.. App (Payment gateway)

(producer App)

(Banking Apps)

nimo pay

payazor

are

=>VISA/Master/... and etc.. Payment gateway apps providing world wide infrastructure to perform cards based (debit card/credit card) based Transactions

=> The cards issued by payment gateway will come

to customer through Banks by linking with bank Accounts

paypal/pay u money/ pay and etc.. are payment broker acting as bridge b/w e-commerce Apps and payment gateways

either

=>Distributed Computing / App developnent makes two Apps of two different servers belonging to same machine or different machines talking each other and also exchanging the data

browser

note:: Since the xml /JSON data is platform idependent, OS independnet, language and technology independent .. so we can say XML/JSON data global data

Server1 (weblogic) machine20 App1 (Flipkar.com)

server2 (tomcat)

machine10

App2 (paypall)

request

XML/JSON Data

response

method calls based interaction

(web application)

(Client App)

(Data in global Formats) (Distributed App)

(server App)

(Consumer App)

(Producer App)

JSON is better alternate to Xml global data.

=>App to App interaction is called Distributed App =>They generally deal with methods, calls/method requets based communication where data can be exchanged through XML/JSON

Format

=>Generally every Application we develop contains 4 layers

a) presesentation layer /UI Layer ---> contains UI logics (struts,jsf, spring mvc, servlet,jsp, angular, react js,html,js,css) (view and controller)

b) Serivce Layer

Model layer

-->contains b.logics (calculations, analyzations, filterings, sortings and etc..)

[using java class or spring beans(spring core+spring Tx)]

In Distirbuted App development,

The client App will haveits own four layers

dealing with MVC Architecture and

the server app will have its onw four

layers again dealing with MVC architecture.

c) Persinstence Layer/DataAccess Layer ------> contains persistence logic like performing CURD Operations

[using jdbc, hibernate, spring jdbc, spring data jpa, spring data jdbc, spring data mongodb,spring orm and etc..]

d) Data Layer

contains the data of the Application (The real persistence store)

[DB s/w (SQL or No SQL), Files (Text files, properties files, Xml files,JSON Files,...)]

add

if want to link one App with another App using Distributed Computing or App development env.. we need to an addtional layer develop

in both cosumer and producer App that is "Integration Layer" we can the logics of this Integration layer using Distributed Technologies or frameworks like RMI,EJB, webservices and etc...

Consumer App ---> Client App that cosumes the services of server App/Producre App Producer App --->

Server App that developes the services of server App/Producre App

and keeps them ready for consumption

DB s/w

Data Layer

(weblogic)

Server1 on machine1

App1 (Consumer App) (Flipkar.com)

presentation ayer PL service

Server2

(Tomcat) on machine10 App2 (Producer App)

resentation ayer PL service

(paypall)

DataAcessLayer(DAL)/ Persistence Layer

Layer($L

DataAcessLayer(DAL)/ Persistence Layer

network

Layer($L

Integration Layer (IL) (Stub

http response

http request (method call) XML/JSON Formats based Data Exchange (method result)

Integration Layer (IL) (Skelton

Db s/w

Data Layer

In web services env, The stub and skelton will communicate with each other using request -response model in which the inputs will be sent in the form of request body and the outputs will be received in the form of response body

In non-webservices env (like RMI,EJB), the stub and skelton will communicate with other in method calls model in which the inputs will be given in the form of method call args and the outputs will be given in the form of method return values

=> To develop Integration Layer Logics of both Consumer And Producer Apps we can take the support of Distributed Technologies or Frameworks

=> The integration layer logic of consumer App is technically called as

Stub logics

=> The integration layer logic of Producer App is technically called as Skelton logics

Different ways of Integration logics in Java env.. /Different Distributted Technologies of Java

===========

developing

layer

=============

(a) RMI (Remote Method Invocation) (java based)

(b) EJB (Enterprise Java Beans)

(d) CORBA (can be implemented in multiple languages)

(c) Http Invoker (from spring JEE module) (java based)

__/Frameworks_========

(e) WebServices (can be implemented in multiple languages) (Best)

and etc..

RMI (outdated)

=====

=>Given by Sun Ms

=> part of Jdk s/w (JSE module)

=>Java based Distributed Technology/framework is nothing but both consumer App and server App must be there in java eg: RMI, EJB, Http Invoker

=> Multi Language Distributed Technology/Framework means the client

and server App can be there in any one language or in two different languages eg:: webservices, CORBA

=>It is language dependent i.e both Consumer and Server App must be there in java

=> It is platform independent and Architeture neautral

=> Can not use Internet network as the communication channel b/w consumer and producer Apps

=> Uses JRMP (Java Remote Method Procotol) as the protocol for communication =>Outdated becoz of EJB

=> Does not give any built-in middleware services

(The ready made secondary logics like security, Transaction mgmt and etc..) =>Here data will be exchanged in binary format b/w consumer and producer apps

EJB (outdated)

===

=>given by Sun Ms

(0,1)

(No xml or json is used)

(non-java)

other domain distributed Technologies

a) RPC (using c/c++)

b) DCOM (using micro soft technologies)

c) .Net Remoting (using .net technologies)

d) Corba (can be implemeted in muliple languages)

e) webServices (can be implemented in multiple languages) (best)__

RPC:: Remote Procedure Calls

CORBA: Common Object Request Broker Archicture

DCOM :: Distributed Computing

(or) Disributed Component Object Model

Consumer App === Client App Producer App === Server App

webServices env.. is the best env.. for developing the Distributed Apps becoz it makes Distibuted Apps as the a) extension of web applications using http/https protocols b) interoperability (the client and consumer apps

on

can be there in two different laguages or technologies or frameworks running any OS having any computer architecture)

Hero in the specification

and Zero in the

implemetation

=> Enhancement of RMI (EJB = RMI++)

=>It is language dependent (Both consumer App and Producer App must be there in Java)

=> It is platform and Architecture independent

=>can use both LAN (Local Arean network) and Internet(WAN) as communication channel

=> needs the heavy weight EJB Container to execute EJB Comps

s/w

[EJB container is part of another heavy weight Applicaiton SErver like weblogic, glassFish and etc..)

=>Gives lots of built-in middlewares (It was popular for banking apps earlier for this reason) (Tx Mgmt is very popular) => Outdated becoz of webServices and its hevyness, complexity

=> It is technology of JEE module and EJB Containers of applicaiton server softwares will come as

the Implementations..

=>We must deploy the devleoped EJB comps in the EJB containers of Application server for execution.. note:: Though Tomcat is called as Applicaiton server from version 7 .. It is still not providing EJB container.

=> Very complex to lean and execute

=>Here data will be exchanged in binary format b/w consumer and producer (no xml/json is not used) CORBA (Common Object Reqquest Broker Archicteture)

=======

=> Platform, Architecture and language indepndent

(not succedded)

=> we can develop produce and consume services in multiple languages like java, c++, c and etc..

Consumer and Producer can be there

The addtional,configurable and optional services that can

be applied on the Applications are called middleware services.

eg: security, Logging,auditing and etc...

EJB comps mean:

the server/provider app of

Distributed computing that is developed using EJB

in two different languages or platforms or architectures

=> CORBA is specification and it will be implemented as IDL (Interface definitation language)

=> CORBA is complex¶earn and apply

=> CORBA is heavy weight to implement

=> CORBA looks great conceptwise.. But gives problems towards the implementations.

WebServices

(Best)

================

=>WebServices is a mechanism of linking two apps as consumer and producer apps

using the protocol http.

=> WebSerevices is platform independent and archicture independent and language independent..

A producer developed in "Java" can be used/consumed in

.net, java, php, phyton, Java script and etc.. (Even reverse possible) the service

the service

=> WebServices says Develop/produce any where and consume any where. (This feature is called interoperability)

=> Webservices make the consumer and producer Apps exchanging data either

data

in XML or in JSON (Global formats)

other

=> Java and languages have provided multiple apis/Technologies and frameworks to

Laver

implement web service logics as skelton logics (Integration Logics Producer)

Layer

and as stub logics (Integration logics of consumer)

=>WebServices support http request and http response based method invocation

the

i.e as http request the cosumer App invokes method of producer App and the

results method execution goes back to consumer App as http response from the Producer App (having xml/json data)

Two ways of implementing web services

a) SOAP WebServices (going down)

b) RESTfull WebServices (best)

SOAP :: Simple object Access protocol (protocol) REST :: Representational State Transfer

web services (It is not a protocol

a) SOAP WebServices

on

=> This runs based 3 component priciple a) SErivce Provider(producer /Server)

c) Service Registry (where ervices

The Registries in SOAP based webServices env..

is UDDI (Universal Description Discovery and Integration)

b) Sevice Consumer (Consumer/Client)

will be registered to to expose)

it is kind of architecture/ mechanism for app to app interaction)

=>Microservices is an architectures to develop every module as seperate project and integrating them as needed.

=> This MicroService architecture is built on the

top WebServices architecture especially

For every SOAP web service (producer app)

that is registered in service Registry

on the top of Restfull webservices.

there will be one WSDL doc (xml document)

MicroServices = Services/modules as

having multiple details about web service (producer app)

like how to consume, the services names

Service

Registry (UDDI) (WSDL docs)

lookup

(2)

(3)

register

wsdl doc

and etc (endpoint details)

(Flipkart AppY

(Phone pe App)

(xml)

Consumer

SOAP over http request Producer

(1)

(4)

75)

(3)

(Serivce consumer)

(Service Provider)

SOAP over http response

(8)

(7) (xml)

is

SOAP message strictly typed or complex XML message

=> The http request carries xml baased SOAP 'message

but

as request body nothing inputs

=> The http response carries xml baased SOAP message

WSDL :: WebService Description Language

Projects using restful websevices + lots of Design Patterns impl

+ lots third party tools

for Integration

as response body nothing but outputs

which will be conveted java obejcts and reverse using

JAXB

jaxb :: Java archicture for xml binding

(nothing but converting java object data to xml content and vice-versa)

java object data to xml conversion is called marshalling and reverse is called

unmarshalling JaxB is used for this marshalling and unmarshalling

Product object

xml content

(java obj)

pid:101 pname: table price: 300

marshalling jaxb code

<product>

unmarshalling

<pid>101</pid> <pname> table </pname> <price> 300 </price> </product>

=>The consumer sends inputs to producer as SOAP message(xml) in http request (as the request structure body) =>The Poducer sends outputs to cosumer as SOAP message(xml) in http response (as the response structure body)

=>The http request and http response contains two parts

a) HEAD part

|---> contains two sections

b) Body part

a) Initial Line

b) headers

Structure of http request

======

=======

request

Initial line http method path protocol & version

http methods or modes are (9)

GET

request headers

POST

HEAD

HEAD

part

user-agent: chrome

TRACE

accept: text/html, text/plain,... accept-language: en-US

PUT (full update)

Body

part

...

>>>> blank line >>>>

payload/body query params or SOAP message (xml)

Requests

POST/Wish

Http request with POST mode

DELETE

OPTIONS

CONNECT (reserved for future)

PATCH (for partial Updates)

In Normal http requeust, the request body will be query String having request param names and values like sno=101&sname=raja&sadd=hyd

In soap over http request, the request body will be xml based SOAP message (xml tags)

```
HTTP/1.1 Host: localhost:8000 User-Agent: Mozilla/5.0 (Macintosh;) Firefox/51.0
```

```
Accept: text/html,application/xhtml+xml,*/*;q=0.8
```

http

(normal request)

initial line/request line

```
Accept-Language:
```

```
en-US,en;q=0.5
```

```
Accept-Encoding: gzip, deflate
```

request headers

```
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=-12656974
Content-Length: 345
```

Body

sno=101&sname=raja&sadd=hyd

blank line request body/ pay load

Request headers carry more information about client (browser) to server.. along with the generated request

• Normal

Http request

HEAD

structure example

=>POST mode reuqest contain request body representing query String where as GET Mode request does nonquest body becoz it carries inputs as query String appended to the request url.

Path to the source

on Web Server

The HTTP Method

Http Request with GET Mode (Normal http request) Protocol Version Browser supports

Parameters to the server

HEAD

The Request Headers

GET /RegisterStudent.asp?user=jhon&pass=java HTTP/1.1

Host: guru99.com

User-Agent: Mozilla/5.0

path with query String

Accept-text/xml,text/html,text/plain, image/jpeg

Accept-Language: en-us,en

Accept-Encoding: gzip,deflate

request line

Accept-charset: ISO-8859-1, utf-8

no

HEre no Blank Line and Request body

Keep-Alive: 300 Connection: keep-alive

Http response structure (Normal)

====

response line /initial line

protocol

& version

HEAD

response

status code status code and message

response haders server :: apache Tomcat

Body

host:

...

contentType: ...

contentLength:

>>> blank line >>>>>

[response body/payload

becos GET mode request does not contain them more

over GET mode request carrries the data in the form

of query String appended to the URL

Normal Http response

HTTP/1.1 200 OK Date: Thu, 20 May 2004 21:12:58 GMT

Connection: close

HEAD

Server: Apache/1.3.27

`Accept-Ranges: bytes`

(initial line/response line) Status Line

Content-Type: text/html

Content-Length: 170 Last-Modified: Tue, 18 May 2004 10:14:49 GMT

blank line

response

headers

<html> <head>

<title>Welcome to the Amazing Site!</title>

Body

</head>

<body>

<p>This site is under construction. Please come

response body/pay load

back later. Sorry!</p>

</body>

</html>

Http response status codes

====

1xx (100-199) :: Informational 2xx (200-299) :: Success

3xx (300-399) :: Redirection

4xx (400-499) :: Incomplete (Client side errors)

5xx (500-599) :: Server side errors

=> response headers given additional

instructions to browser towards displaying

the recieved response as the webpage (eg: auto refreshing the web page)

web comp results or SOAP message

(any thing)

(xpl)

Normal http response contains the web comp genertad ouput (generally html code

or plain text) as response body where as soap over http response contains

the soap message(xml) as the response body

soap over http request

======

(SOAP messages will go as http request body)

POST http://127.0.0.1:8088/mockServiceSoapBinding HTTP/1.1 request line

Accept-Encoding: gzip,deflate

Content-Type: text/xml; charset=UTF-8

HEAD

req

part

headers

SOAPAction: "http://www.soapui.org/sample/login"

Content-Length: 505

Host: 127.0.0.1:8088

SOAP over http response

Connection: Keep-Alive

User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

| blank line

response line

<soapenv:Envelope xmlns:sam="http://www.soapui.org/sample/" xmlns:soap

HTTP/1.1 200 OK

<soapenv:Header> <wsse:Security xmlns:wsse="http://docs.oasis-open.org/v

<soapenv:Body>

X-Powered-By: Servlet/2.5 Server: Sun GlassFish Enterprise Server v2.1.1

response

HTTP Headers

<sam:login>

<username>Loginn0.196</username> message password>Loginn123</password>

soap

Body

</sam:login>

as

</soapenv:Body>

req kedysoapenv:Envelope>

=> In normal http request/http response the body/payload will be

BODY

plain text content where as in soap over http the body/payload

will be SOAP protocol based messages

HEAD

Content-Type: text/xml; charset="utf-8"

Transfer-Encoding: chunked

Date: Wed, 05 May 2020 21:10:14 GMT

blank line

```
<?xml version="1.0"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<ns2:sayHelloResponse xmlns:ns2="http://soap/">
<return>Hello, Shawn!</return>
</ns2:sayHelloResponse>
</S:Body>
</S:Envelope>
```

=>SOAP over http means the http request and http response contain SOAP messages (xml) as boyd/payload

=>In SOAP based webservices, if the consumer and producer apps are avaiable in the same machine then

the communication protocol is SOAP over HTTP

=>In SOAP based webservices, if the consumer and producer apps are avaiable in two different machines then

the communication protocol is SOAP over HTTP over TCP/IP

=>TCP/IP is network protocol to get interaction between two physical computer s

to

=> SOAP over http is application protocol get interaction b/w cosumer and producer app of SOAP based

webservices.

Message Body

(soap message

as response body)