SOA vs MicroServices:

Microservices evolved from SOA. Share many characteristics

Service oriented intigration

App a | App B | App F 3rd Party propritory

| | |

\/ \/ \/

------------------------------------------------

Enterprise Service Bus

-------------------------------------------------

Service-Oriented Intigration(SOI). Applications communicate with each other through common intigration layer standards SOAP/XML

over HTTP or JMS.

OESB.

OSB deeply rooted into their products.

Either write heavy orchestration logic in the ESB layer or the

business logic itself in the service bus.

All enterprise level services are deployed and accessed via ESB.

For such organisations Microservices are altogether different from

SOA.

Possible questions asked:

What is spring used for? why is it used ?

-Lightweight for the types of functionality available,

-IOC & dependency injection

-Security

-Spring Boot

Examples of using multiple threads.

Multiple SSH connections - Concurrently creating multiple threads using either ParallelStreams or Completable futures.

SSH command is sent to server to compute some action, returns a string.

Why is multithreading used ?

To increase performance and efficiency. Tested:

In comparison to doing a task sequencially. Tasks that may have some latency such as receiving a response from a server,

will take a lot of time.

Utilising multithreading or doing tasks concurrently greatly reduces the time in which tasks are completed overall.

Threads do not have to wait for a pervious thread to complete in order to execute a task.

However this leaves the door open to risks such as deadlocks etc. Especially when attempting

to write to a shared resources. Synchronizing helps to a certain extent. Locks the resource until it is finished.

Describe the architecture of a project you work on?

Mention discussion about SOA and Microservices.

The differences between Microservices and SOA.

Microservices

Variant of SOA - A collection of loosely coupled services. Services are fine grained and protocals are lightweight.

Decomposing an app improves modularity - making apps easier to understand, develop

test, and become more resilient to architecture erosion.

Parallelizes development by enabling autonomous teams to develop, deploy and scale

servies independently.

Enables continous delivery and deployment.

Services form information barriers.

Statistics questions about how many possible paths from A to B.

Why is REST used so widely.

Representational State Transfer

-They are stateless and separate the concerns of client and server

Separation of cient and servers:

-Implementation of server can be done independently without

each knowing about the other.

-Code at client side can be changed at any time without affecting operation of server, and vice versa.

-As long as side knows what format of messages to send,

each other, they can be kept modular and separate.

-This improves flexability of interface across platforms

and improve scalability.

-Allows each component to evolve independently.

Statelessness:

-Server does not need to know anything about client and vice

versa.

-Both client and server can understand any message received

without seeing previous messages.

-constraint of statelessness is enforced through use

of resources, rather than commands.

-Do not require the implementation of interfaces.

-Constraints help RESTful apps become :

-reliable

-quick

-scalable

-Components can be managed, updated and reused.

Communication:

USING HTTP VERBS:

-Get

-Post

-Put

-Delete

Status code:

-200 (OK)

-201 (Created)

-204 (No Content)

-400 (Bad Request)

-403 (Forbidden)

-404 (Not Found)

-500 (Internal Server error)

REST vs SOAP:

SOAP:

-Designed to expose individual operations - or pieces

of operations as web services.

-Uses XML instead of HTTP do define content of th message.

-Require formal contracts between the API and consumer service

by using WSDL(Web services description language)

-Built in WS-Reliable messaging - increase security

in asynchronous execution.

-SOAP as built in Stateful Operations. :Dedigned to support con

versational state managenebt.

How do you keep up with technology news ?

-ThoughtWorks Tech Roadmap

-Books

-Communicating with colleagues

-Want to go to confrences

-Blogs

If you could work on any technology during the free time, what would it be ?

-Machine Learning

What steps would you take when you get a support issue.

How can you manage the memory of an application ? is this memory pattern good or bad ?

-Close Resources

-Resources are dereferenced properly so is eligivle for garbage

collection.

JVM switches:

-Xms : setting up heap size

-Xmx : setting up MAX heap size

-Xmn : for setting the size of the Young Generation - Rest of the space goes to for old Generation

-XX:PermGen - Setting up initial size of the Permanant Generation Memory

-XX:MaxPermGen - For setting maximum size of Perm Gen.

-XX: SurvivorRatio: Providing ratio of Eden space and survivor space

-If Young Generation size is 10m and VM switch is -XX:SurvivorRatio=2 then 5m will be reserved

for Eden Space and 2.5m each ofr both Survivor spaces. Default value is 8.

-XX:NewRatio: Providing ratio of old/new generation sizes. default value is 2

Questions on garbage collection.

-Objects created inside a method:

-When a mehod is called it goes inside the stack frame.

When the method is popped from the stack, all its members dies and if some objects were created

inside it then these objects become unreachable or anonymous after method execution

thus becomes eligible for garbage collection.

-Reassigning the reference variable:

-When reference id of one object is referenced to reference id of some other object then the

-previous object has no any longer reference to it and becomes unreachable. Becoming

eligible for GC.

-Nullifying the reference variable:

When all the reference variables of an object that are changed to NULL - it becomes unreachable

eligible for garbage collection.

-Anonymous object:

The reference id of an anonymous object is not stored anywhere. Hence, it becomes unreachable;

JDK vs JRE

JRE - JAVA RUNTIME ENVIRONMENT.

The environment which is needed for compiled programmes to run. Commands such as Java are available here.

Cannot develop in this environment,

Where JDK is the environment where you can develop or write Java programmes for the JRE to execute. JDK comes with JRE. Commands such

as javac - to compile a java programme is available here.

Conversion of java primitive types from one type to another.

byte – 8 bits and signed

short – 16 bits and signed

char – 16 bits and unsigned, so that it may represent Unicode characters

int – 32 bits and signed

long – 64 bits and signed

float – 32 bits and signed

double – 64 bits and signed

boolean – it’s not numeric, may only have true or false values

Widening primitive conversions:

int myInt = 127

long myLong = myInt

float myFloat = myLong;

double myDouble = myLong;

narrowing primitive conversions:

int myInt = (int)myDouble;

byte myByte = (byte)myInt;

Special cases:

when we want to convert byte to char.

byte myLargeValueByte = (byte) 130; //0b10000010 -126

System.out.println(myLargeValueByte);

char myLargeValueChar = (char) myLargeValueByte;

System.out.println(myLargeValueChar);

int myLargeValueInt = myLargeValueChar; //0b11111111 10000010 65410

System.out.println(myLargeValueInt);

byte myOtherByte = (byte) myLargeValueInt; //0b10000010 -126

System.out.println(myOtherByte);

char myLargeValueChar2 = 130; //This is an int not a byte!

System.out.println(myLargeValueChar2);

int myLargeValueInt2 = myLargeValueChar2; //0b00000000 10000010 130

System.out.println(myLargeValueInt2);

byte myOtherByte2 = (byte) myLargeValueInt2; //0b10000010 -126

System.out.println(myOtherByte2

Inheritence and polymorphism related programs and questions on their behaviour

Inheritence: When a class is derived from another class. "Subclass extends Superclass"

Types of inheritence:

1: Single Inheritence

-Subclass inherits features of one class

2: MultiLevel inheritence:

-Derived class will be inheriting from superclass as well as acting as baseclass for another derived class

A -> B -> C

3: Hierarchical inheritence:

-One class serves as a superclass for more than one subclasses

A

|--------|---------|

B C D

4: Multiple inheritence:

-One class has more than one superclass and inherit from all parent classes.

JAVA does not support multiple inheritence:

Can be achieved using interfaces.

Class one

Class two

Iterface three extends one, two

Child class implements three

5: Hybrid inheritence:

A subclass cannot inherit private members of its superclass (can use getters and setters to access it).

Constructor and initializer blocks cannot be inherited by subclass.

Polymorphism :

-The ability of an object to take many forms.

-Method overloading

Class constructors

Inner and outer classes.

Interfaces vs abstract classes:

-Interfaces can only have abstract methods (Method without a body). (From java 8 it can have default and static methods)

-variables declared in interfaces are final - Abstract class may contain non final variables

-Abstract class can provide implementation of intervase !vice versa

-Interface implemented using implement method - Abstract is extended

-Members in interface are public by default, abstract can have any access modifiers.

STACK VS HEAP memory

Java heap space is used by java runtime to allocate memory to

Objects and JRE classes.

Whenever we create any object, it is always created in the Heap Space.

Garbage collection runs on heap memory used by objects that don't have any

reference.

Stack memory: Used for execution of a thread - Contain method specific values

short lived and references to other objects in the heap.

Stack is last in first out.

we can use -XMS and -XMX : determine startup size and maximum heap memory.

-Xss define the stack memory size.

---------------------------------------------

JRE vs JDK

JRE

Package of everything necessary to run a compiled Java program.

- Including java virtual machine, java class library and the Java command.

JDK

Has everything JRE has but also the compiler. Capable of creating and

compiling programs.

If you just want to run java programs then you want JRE.

If you are planning to programme in java JDK.

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

Java primitive types.

byte : 8

char : 16

short : 16

int : 32

long : 64

float : 32

double : 64

boolean : true false

Business Intelligence and OLAP.

OLAP - Online Analytical Processing.

Performs multidimensional analysis of business data.

Provides the capability for complex calculations, trend analysis, and

sophisticated data modelling.

Foundation for - Business performance management, planning, budgeting,

Forecasting, Financial Reporting, Analysis, simulation Models,

knowledge discovery, and data warehouse reportint.

Provices end users to perform ad hoc analysis of data in multiple dimensions,

thereby providing the insight and understanding they need for better decision

making.

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

Precomputing all totals and subtotals needed for reporting when the database

server is normally idle.

Stored in OLAP cube.

Don't have to loop through any transaction - all pre-calcualated providing

instant access.

OLAP cube is a snapshot of data at a specific point in time.

-End of day, week, month or year.

Business Intelligence - analysis and presentation of data, stored in an OLAP cube.

Dimension X Dimension X Measures

Category | Time | Thing thats being totalled.

Retrieving a value from the cube - retrieving value from intersection.

This is an interaction of a Measure and Dimensions.

To get data from OLAP cube to pivot table it uses MDX (Multi dimensional expressions).

Pivot requests MDX query from OLAP cube.

Data requested by MDX query returned form cube.

SOA vs MicroServices:

Microservices evolved from SOA. Share many characteristics

Service oriented intigration

App a | App B | App F 3rd Party propritory

| | |

\/ \/ \/

------------------------------------------------

Enterprise Service Bus

-------------------------------------------------

Service-Oriented Intigration(SOI). Applications communicate with each other through common intigration layer standards SOAP/XML

over HTTP or JMS.

OESB.

OSB deeply rooted into their products.

Either write heavy orchestration logic in the ESB layer or the

business logic itself in the service bus.

All enterprise level services are deployed and accessed via ESB.

For such organisations Microservices are altogether different from

SOA.