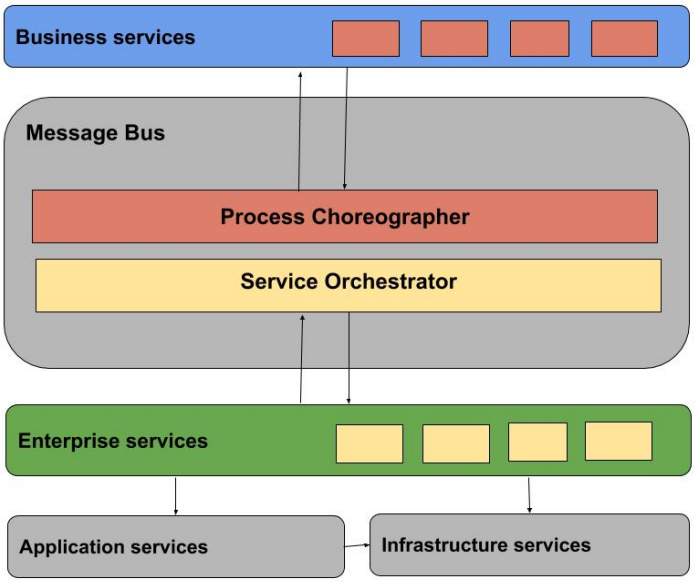
What is Service Oriented Architecture

Description

Introduction

Service oriented architecture or also known as SOA is a design approach where multiple services collaborate to provide some set of features/capabilities. A service here is being referred to as a separate operating system process. Mostly the communication between the services is via network rather than method calls within the process boundary.



Types of abstraction in SOA

Before diving into the explanation of each component in SOA architecture, let’s discuss theabstractions that SOA provides. There are 5 types of abstraction.

–**Location transparency** – If I am calling the service either as a client or service, I don’t need to know the address of the service. I need to locate without knowing the address
-Name transparency – I as a caller to that service either as a client or service don’t have to be concerned or aware of the name of that particular service or the name of the actual operation that I am invoking. A Good way to ensure that is I can refactor my service to change the class name and method name and not impact the caller.
-Implementation transparency – I am not concerned about the language in which the service is written and also the platform. E.g. I make a call to some weather service and don’t have to know the underlying implementation and only concerned about the weather
-Access Decoupling – Access decoupling means we donâ€™t need to care or donâ€™t need to know the access protocol that particular service is using. E.g. We can choose RESTful to invoke another service even though that service is using some different access protocol.
-Contract Decoupling – Contact decoupling allows us to pass different information than what that service is asking for. Allows me in some cases to change the contract in the underlying service and not impact the caller.

Levels of abstraction

Business services

-All the services that come under Business services are abstract Enterprise level coarse grained services owned and defined by Business users.
-We only have name, input and output data represented as wsdl, bpel, xml etc. Business services do not have any implementation.

Enterprise services

Enterprise services are concrete enterprise-level coarse grained services owned by shared services team.

Application services

-These are concrete application level fine grained services which are bound to a specific application context.
-These services are owned by application teams.

Infrastructure services

-These are Concrete enterprise-level fine grained services owned by infrastructure or shared service teams.
-These services represent the cross cutting concerns (Logging, security etc) and also they implement

non-business functionality to support both enterprise and business services.

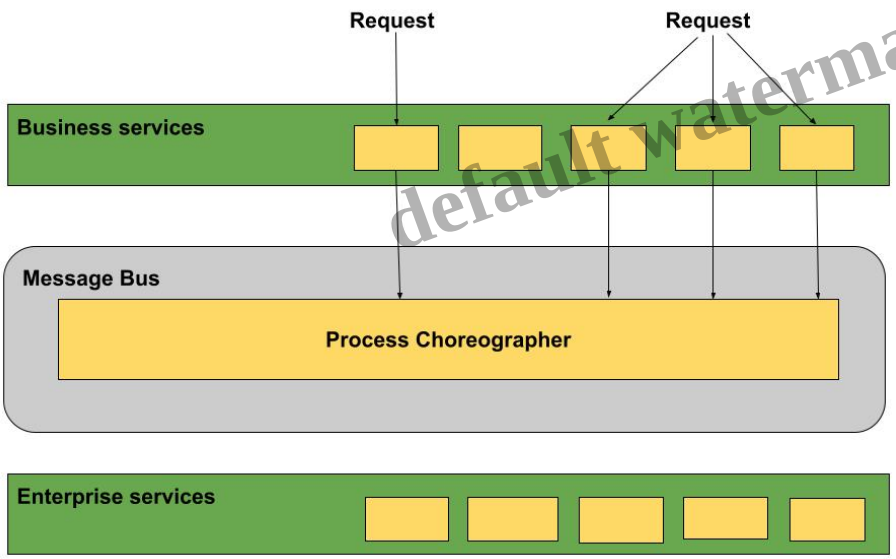
Message Bus

-Binding all the services i.e. Infrastructure, Application, Enterprise, Business services together is the middleware and that is the Message bus.
– The message bus performs different type of functions like Routing, Orchestration, Choreography, Message & protocol Transformation
– Routing – The whole point of Routing in the message bus is to find out which Enterprise service is implementing the business service and how to get to those.

Lets see the detailed explanation of each feature provided by Message Bus.

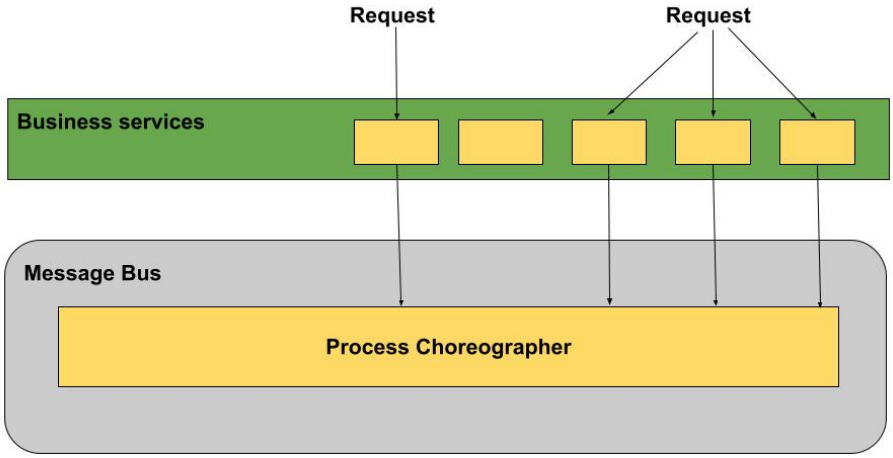
Routing

When a business service is executed, it needs to find the corresponding enterprise services which implements that business functionality.



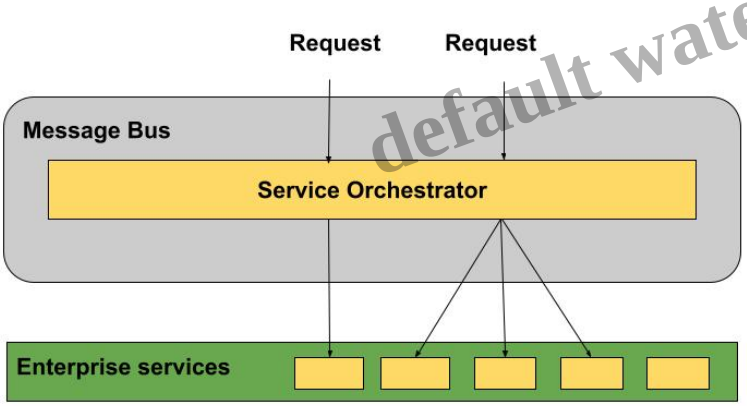
Process Choreography

Here a request can either be to a single business service or for a particular type of request it can span to multiple business services and coordination is required between multiple business services.



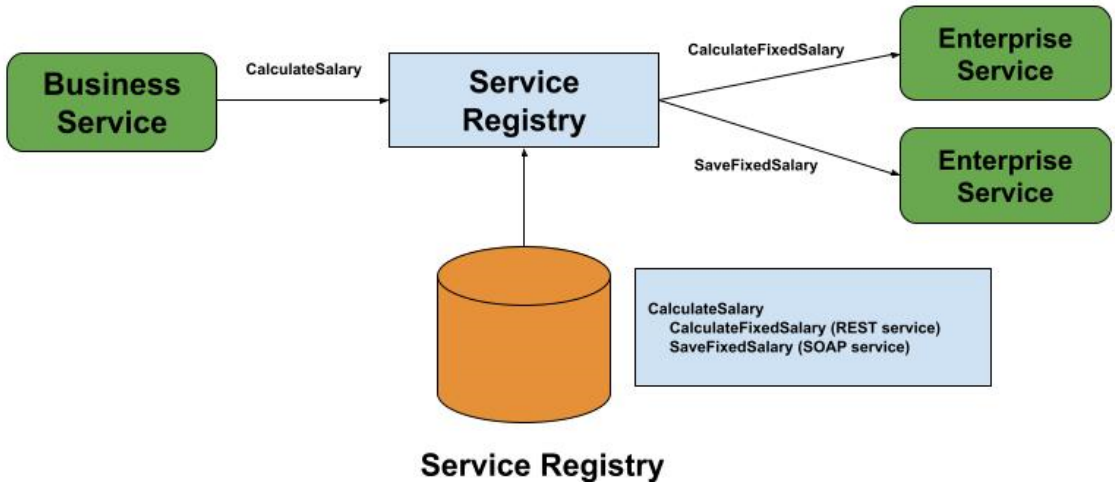
Service Orchestration

A request comes into the message bus , the service orchestrator knows how many enterprise services it has to invoke or need to be executed in order to meet a business request. It can either be one to one or one to many.



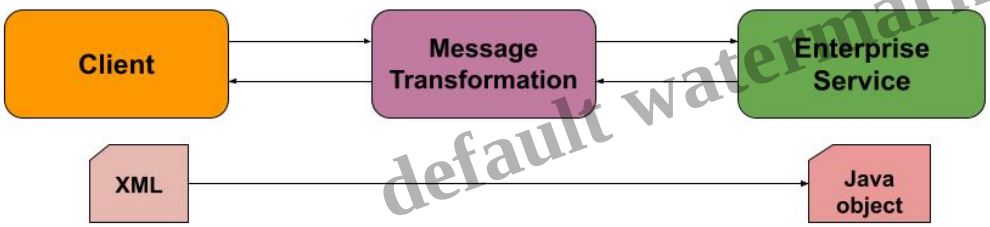
Service registry

This is used to locate which enterprise service has implemented the business service. Service registry knows what enterprise service it has to call and where to go and find those.
E.g. Here the business service wants to calculate the salary but is unaware which enterprise services have implemented the calculateSalary which it has invoked. This calculateSalary is associated with CalculateFixedSalary and SaveFixedSalary methods in two different enterprise services and all this mapping information is present in the service registry.



Message Transformation

If the client is sending a request payload in the xml format but the enterprise services need the data in the java format, then the service bus will transform the xml message format to the Java object format.



Protocol Transformation

If a client is invoking via a different protocol say SOAP and the service is accepting either RMI or REST, the protocol transformation is done from SOAP to REST. The client does not have to care on what protocol the enterprise services operate on.

I hope you liked this article on Service oriented Architecture. I would like to give most of the credits to Marks Richards because of which I was able to write this article.

Keep Learning!

Category

1. Design
2. Microservices

Date Created

March 29, 2021
Author
kk-ravi144gmail-com