



**MODULE-**  
**13 SOA**

# Course Topics

## → Module 1

- » Introduction to Java

## → Module 2

- » Data Handling and Functions

## → Module 3

- » Object Oriented Programming in Java

## → Module 4

- » Packages and Multi-threading

## → Module 5

- » Collections

## → Module 6

- » XML

## → Module 7

- » JDBC

## → Module 8

- » Servlets

## → Module 9

- » JSP

## → Module 10

- » Hibernate

## → Module 11

- » Spring

## → Module 12

- » Spring, Ajax and Design Patterns

## → Module 13

- » **SOA**

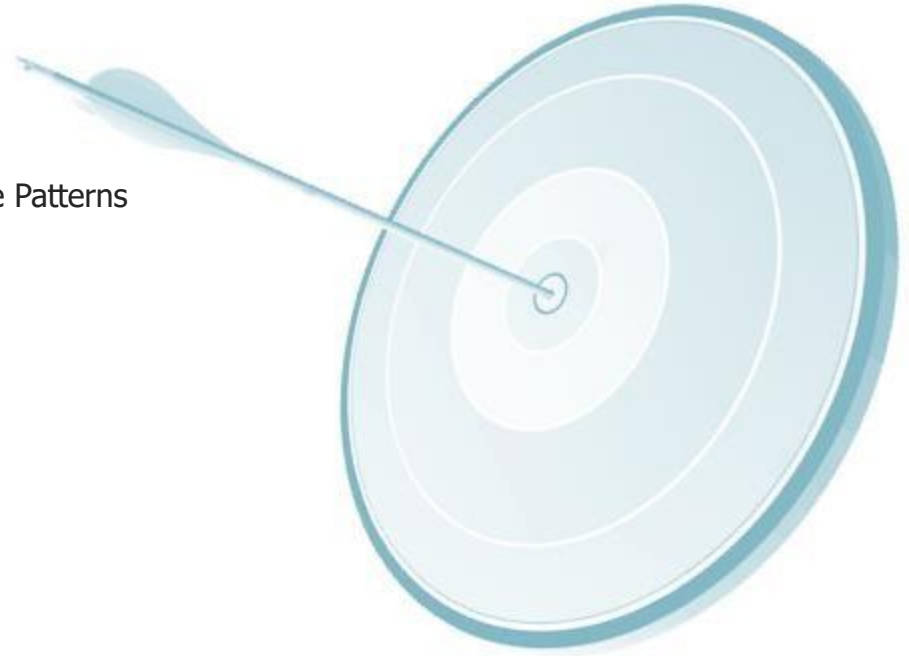
## → Module 14

- » Web Services and Project

# Objectives

At the end of this module, you will be able to

- Understand SOA and its concepts
- Learn SOA services and their life cycle
- Understand Enterprise Service Bus
- Learn Business Process Management
- Understand SOA architecture and Message Exchange Patterns



# Mark seeks Help!



John, one of my applications requires to process a request for online bank transactions. We have that functionality written in some other platform. How can we use it in our current platform?

# John helps Mark..

SOA is there to help you.  
It is software design  
methodology, which you  
can follow to provide  
your application  
functionality as services  
to other applications.



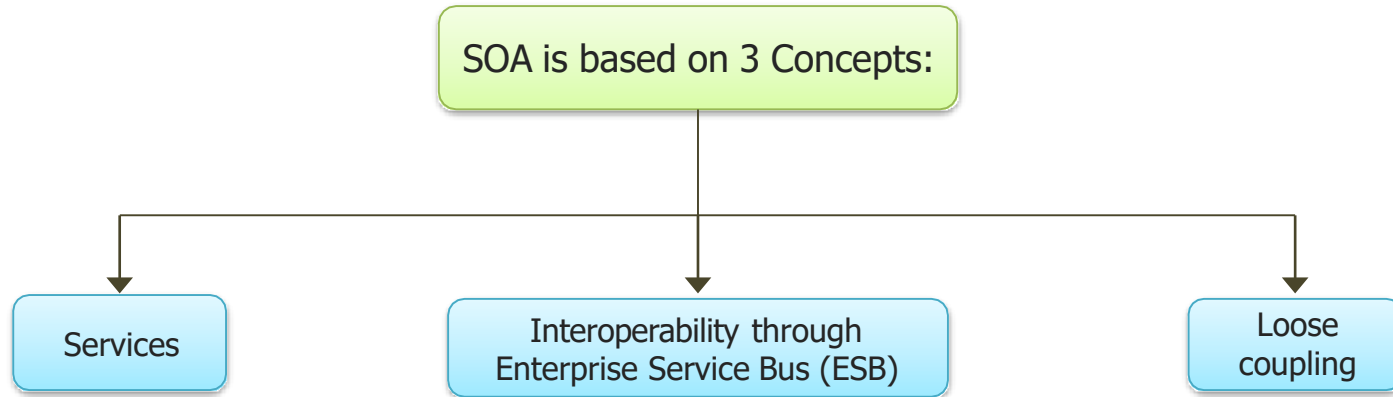
# John helps Mark..

So an application written by keeping SOA design in mind can be published as a Service, and others can use that regardless of the technology or platform they are using.



# Introduction to SOA

SOA helps to meet the needs of business processes of large distributed systems.



# SOA is based on 3 Concepts

- A service will have functionality/work to be done which eases the work of the user/client.
  - » This provided service could be as simple as getting some information from the user and fetching the required data from the db or it could be as complicated as an entire business process for an organization.
- Enterprise service bus which enables interoperability between systems for services. It makes it easier to have the services running in multiple systems with different platforms and technologies.
- **Loose Coupling**: When there is tight coupling between systems then the dependency is very high. If it is loosely coupled then the dependency is low.
  - » It is like writing a method and call this method in all the places. In the method specify the external dependency.
  - » If there is change in external dependency then only the method has to be modified.
  - » Loose coupling will help in minimizing the risk of modifications and failures.



# SOA – Policies and Processes

---

Introducing a new process for the organization is not a simple task. It has to be done by different teams in the organization.

After defining the policy, there is a need to have the roles and processes defined.

Take an example of new policy for credit card authorization. It involves various departments including Marketing, HR, Credit card authorization, Customer support etc.

# SOA Services

*The noun “service” is defined in dictionaries as “The performance of work (a function) by one for another.” - by OASIS SOA reference model.*

SOA focusses on business functionality. These processes are performed with different steps on different systems. A service should represent the real world business functionality.

A business person should be able to understand the functionality provided by the service.

Technically, a service takes the required input, processes it and returns the required information.

SOA is a software architecture that starts with interface definition and builds the entire application as interfaces so that the client can call the interface.

# SOA Services – Additional Attributes

**Self-contained:** They are self governed, independent, autonomous.

**Abstraction:** Services abstracts the implementation from the client. The issues of abstraction is, several calls has to be made. Hence, it is better to send the required data at once instead of sending it multiple times.

**Visibility of service:** You need to know that a service exists or it can be spread via “word of mouth”.

**Stateless:** Services will not maintain the state. Whatever the user asks for, it is served. At times, for reference purposes it is stored in the database.

**Idempotent:** You call a service and there is some response from the service. You do not know whether the response is right or wrong. You call the same service again to verify. This is called as Idempotent.

# SOA Services – Additional Attributes

---

**Reusable:** Service is written once and can be used many times.

**Composable:** A service can call another service as business process is broken into small functionalities.

**SLA:** A service can have Service Level Agreements like time to perform the given task, reliability etc.

**Pre and post conditions:** Pre and post condition will help to know resolve the issue(s) of the customer. Pre condition means customer would have come up with some issues and post condition is resolving the issue of the customer.

**Interoperability:** Will be able to communicate between different systems.

**Implemented as web services:** Web services are used to implement SOA.

# SOA – Loose Coupling

**Fault tolerance:** When the stock market system goes down for an hour there is a loss of 8 million dollars.

If the flight booking systems goes down for an hour there is a loss of 1 million dollar an hour.

Hence, the systems has to be robust.

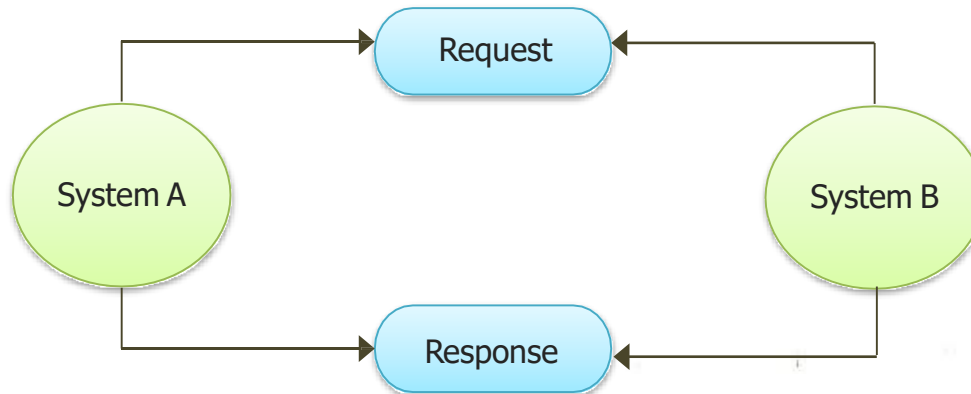
Fault tolerance has to be implemented in the systems. This is achieved through loose coupling.

# Asynchronous Communication

**Asynchronous communication:** Sender of a message and receiver of a message is not synchronized.

Consider the following example of an email:

Sender sends an email. Receiver receives it. Receiver checks the email at his own pace and responds back. In this form of communication, sender sends a message to the receiver. It is in receiver queue. When the receiver sees the message, sender gets an acknowledgement. Receiver can send the response back and the sender gets the notification.



# Loose Coupling – Forms

**Heterogeneous Data systems** – When the data types for two systems are different, then one wrapper has to be written to bind these two systems.

Like first name and last name as name in one system and just name in one system. This has to be binded by a wrapper.

# Enterprise Service Bus (ESB)

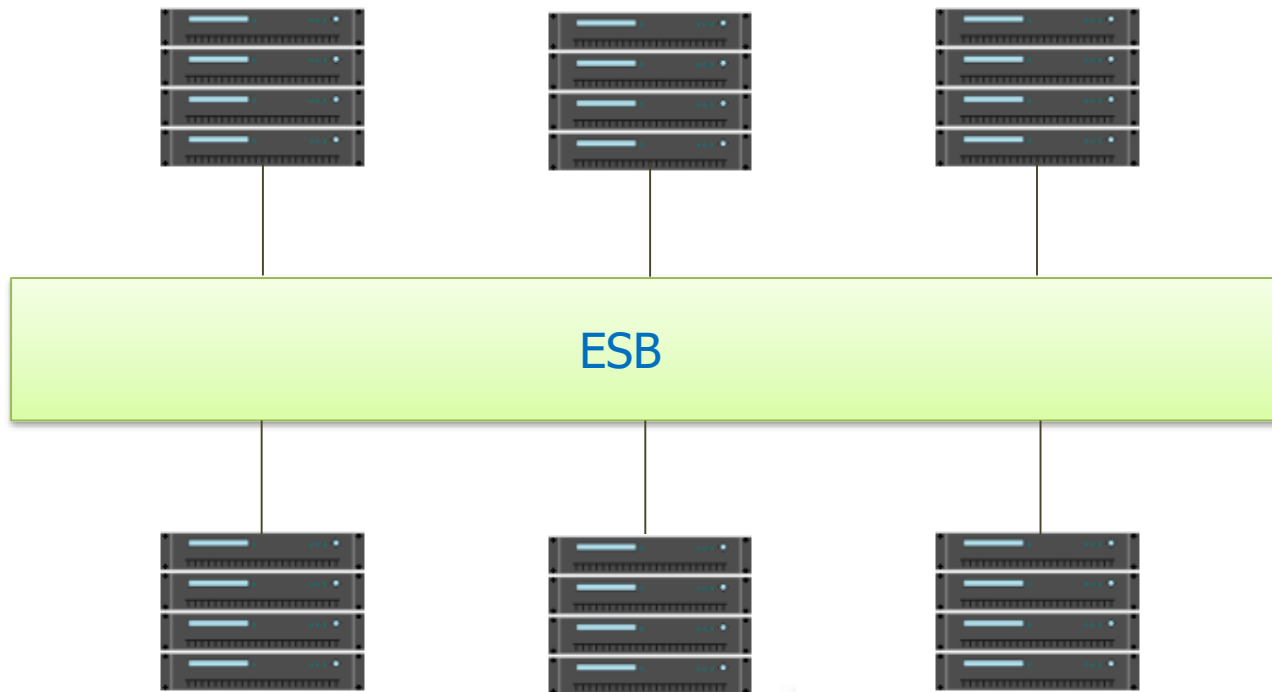
An enterprise service bus (ESB) is a software architecture model used for designing and implementing communication between mutually interacting software applications in a service-oriented architecture (SOA).

The concept has been developed from bus architecture from the computer hardware. The motivation was to develop a standard, structured and general purpose component which is of loosely coupled that are expected to run independently.

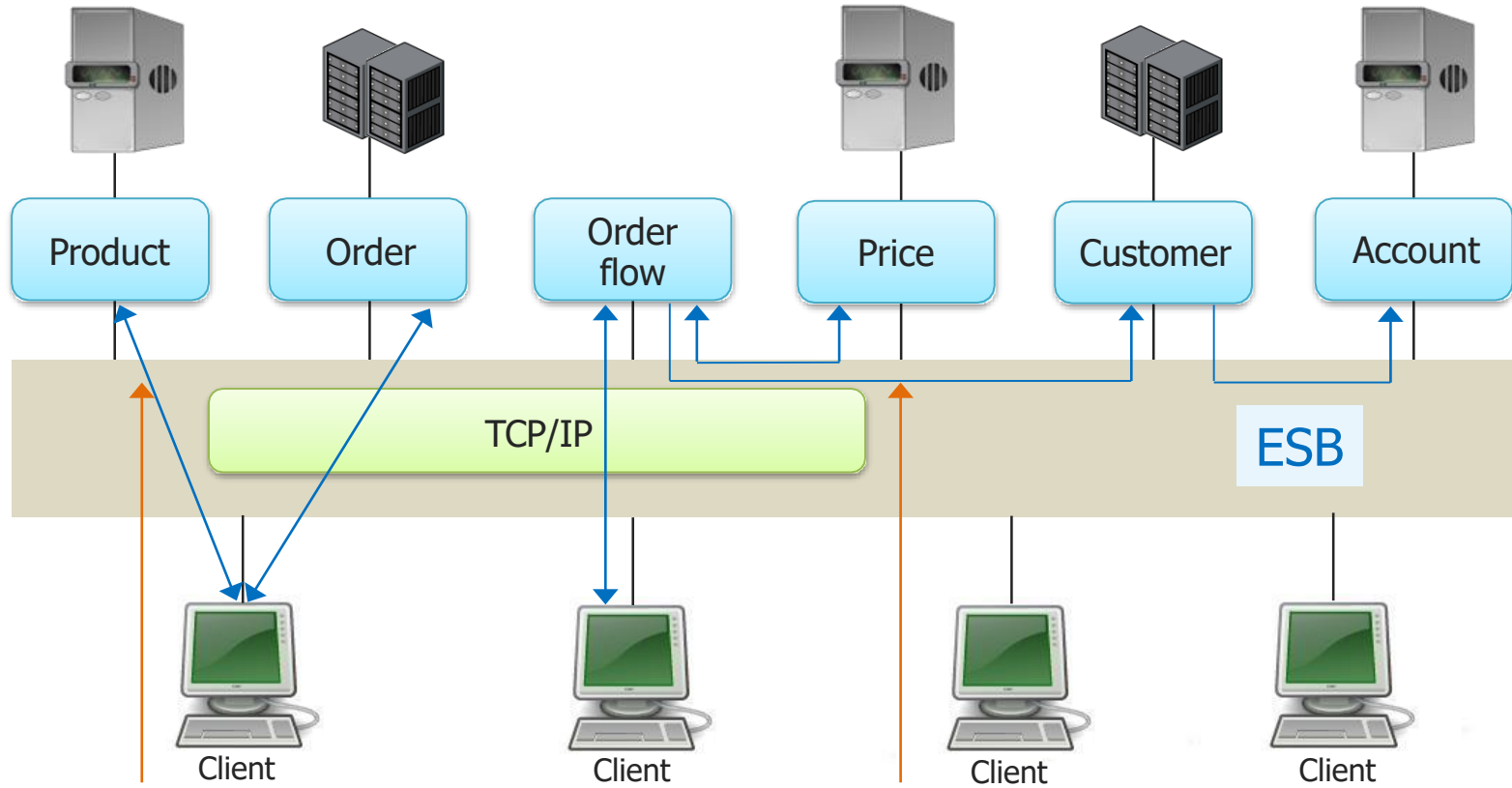
It is like a management layer for technical services of SOA.



# ESB – Architecture



# ESB



SOA Interactions

Event Notifications

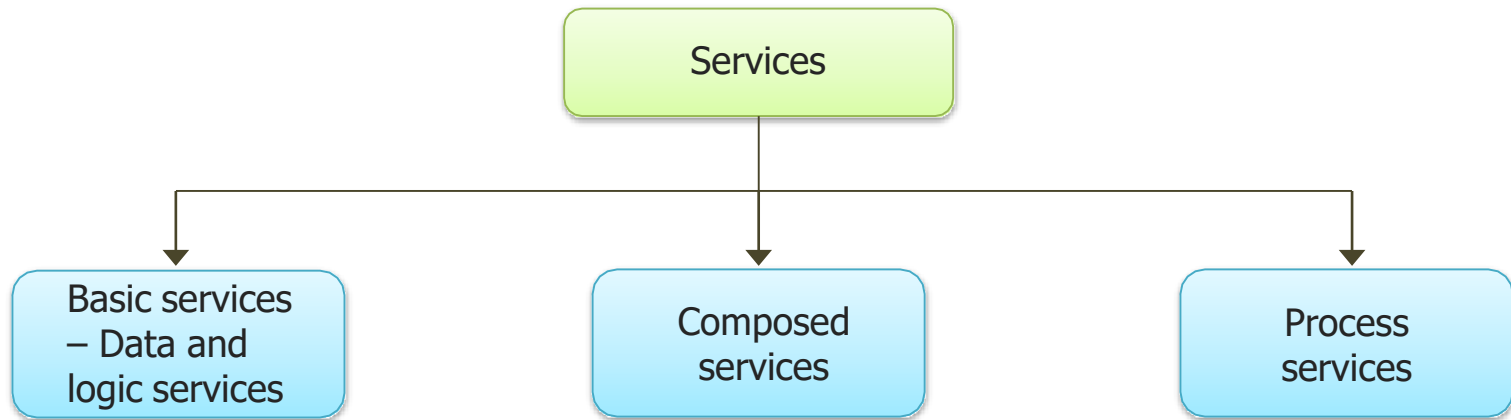
# Responsibilities of ESB

---

- Providing connectivity
- Data transformation – As discussed earlier, can define how to couple different data types into compatible one
- (Intelligent) routing – Supporting load balancing and failover support
- Dealing with security
- Dealing with reliability – Has to define how to reliably deliver services and defines how to handle the technical issues
- Service management – When the number of services grow then management comes into the picture. What processes and policies has to be implemented should be part of this component.
- Monitoring and logging – All the activities are monitored and logged in

# SOA – Services

Services is of 3 types:



# Services - Basic Services

Basic services are of two types: They are (a) Basic Data Services (b) Basic Logic Services

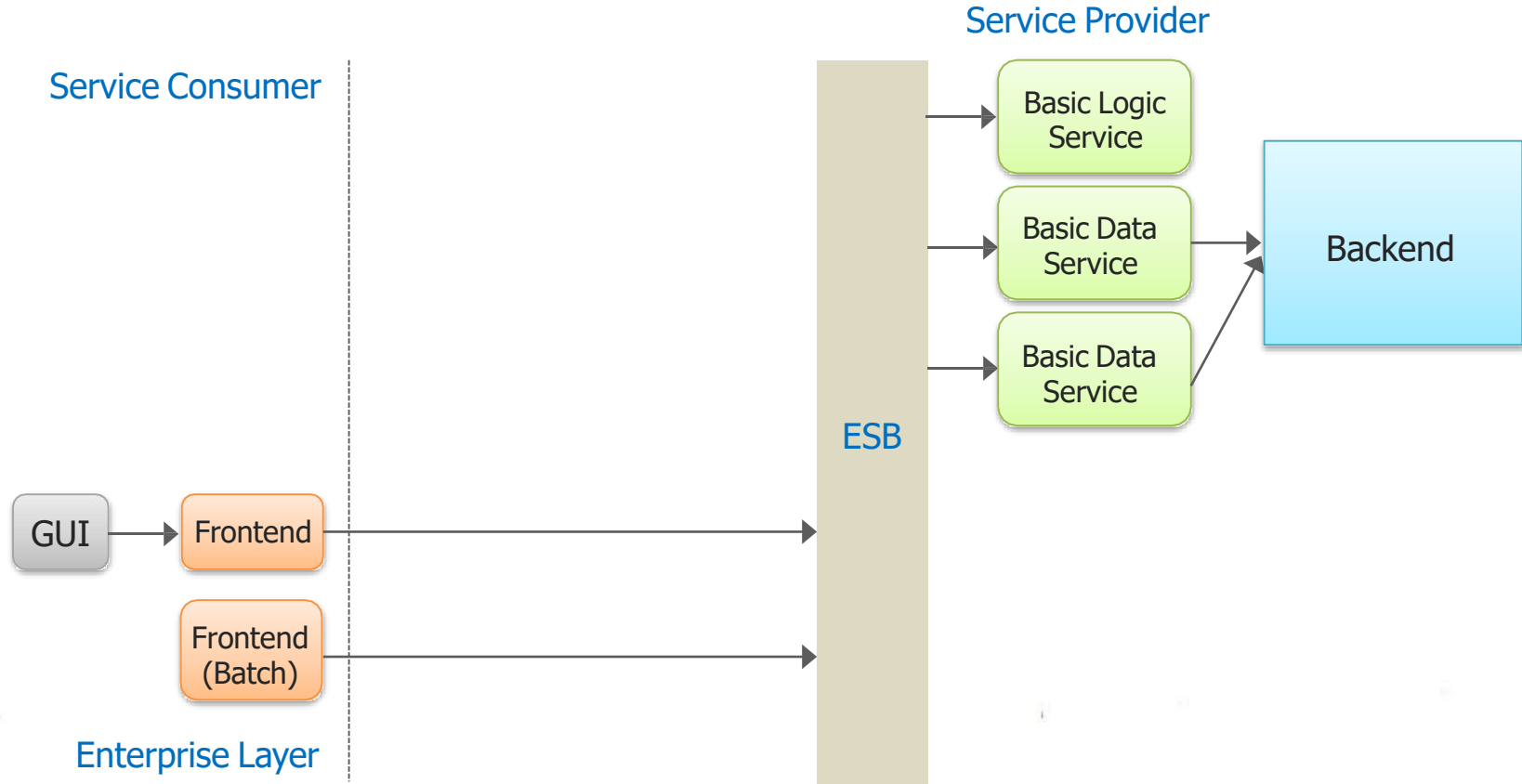
**Basic Data Services:** This is the service which is used to read or write basic business processes. Some of the examples of this type of service are given below:

- Create a new customer.
- Change the customer's address.
- Details about the calls made by mobile in that month.
- Informing about the credit card balance.

**Basic Logic Services:** This service represents business rules. These services take the data and take a decision and responds. For example:

- Defining product catalogs and price lists based on customer needs.
- Providing the solution to a problem based on the input like change in the customer contracts.

# Basic Services



# Services - Composed Services

The second service is composed services.

It uses the combination of basic services or other composed services.

In SOA, composing new services using the existing services is called [Orchestration](#).

Like in an orchestra, many music instruments can be combined to generate the required music. These services are also called as orchestrated services.

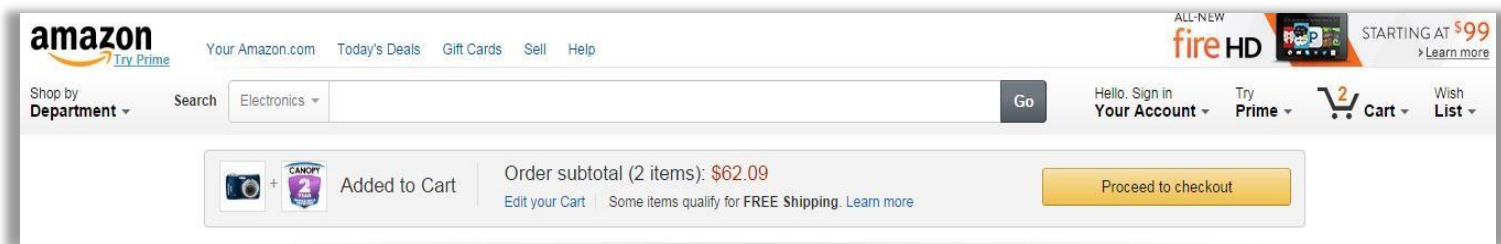
These services operate at a higher level than basic services.

[For example](#): If the customer address needs to be updated in many databases like CRM, RDBMS (MySQL) etc., then composed services can be used.

# Services - Process Services

Unlike basic and composed services, process service maintains the state of the user. One of the example of process service is shopping cart service.

State of the customer's selection is maintained in shopping cart over the multiple sessions.





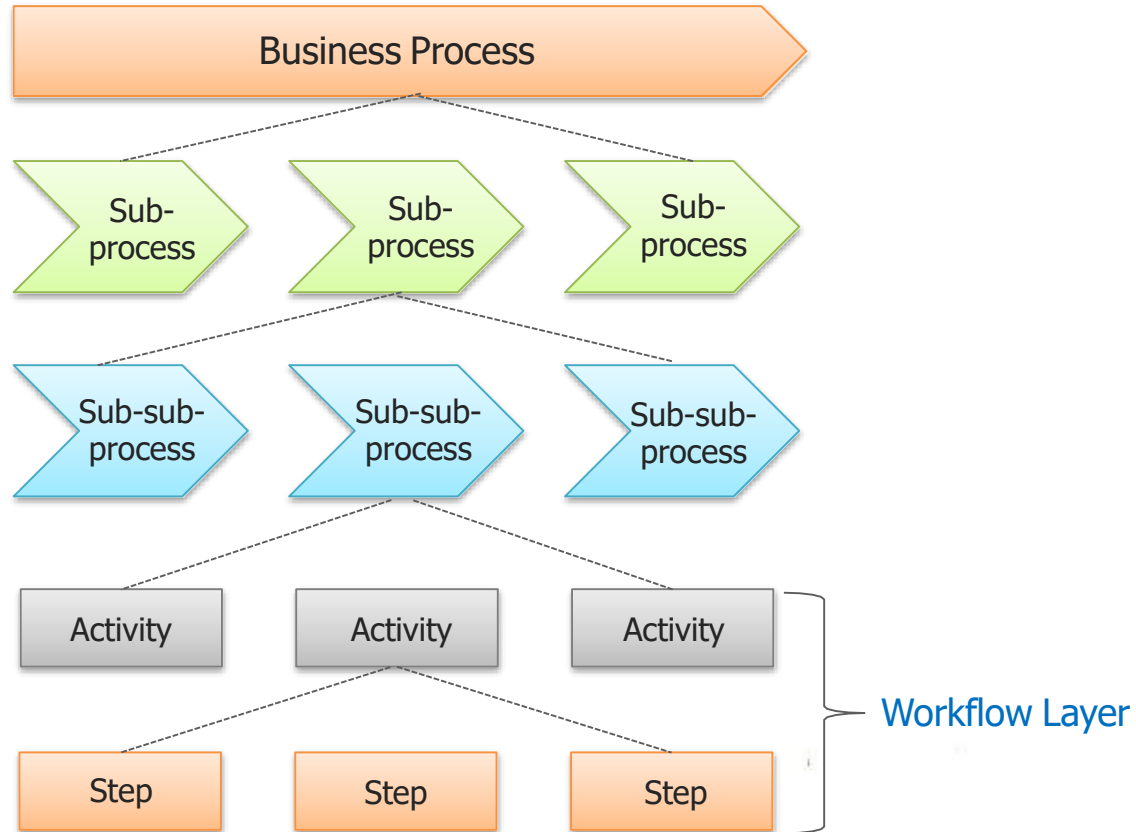
# Business Process Management (BPM)

---

Business Process refers to the work that has to be done including what is the input taken and what is the output given. It might include manual activities or kind of resources.

A workflow describes how certain result can be achieved. It looks into the detailed level of producing the results like steps or activities which needs to be performed.

# BPM and Workflow



# BPM and SOA

---

Business process management is very huge.

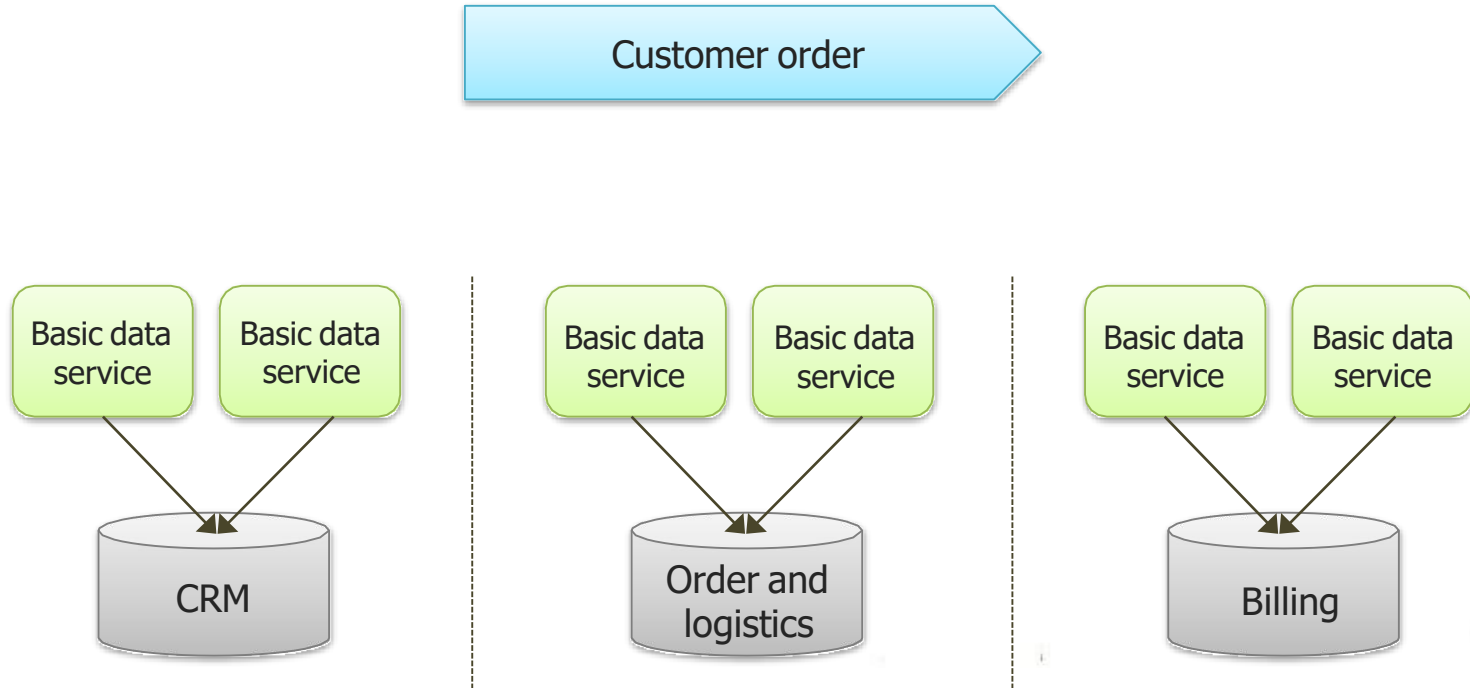
It deals with analyzing the business (Including needs and opportunities), implementing business strategies, monitoring and optimizing business processes, establishing corresponding tools and culture and aligning business with IT.

BPM in relation with SOA.

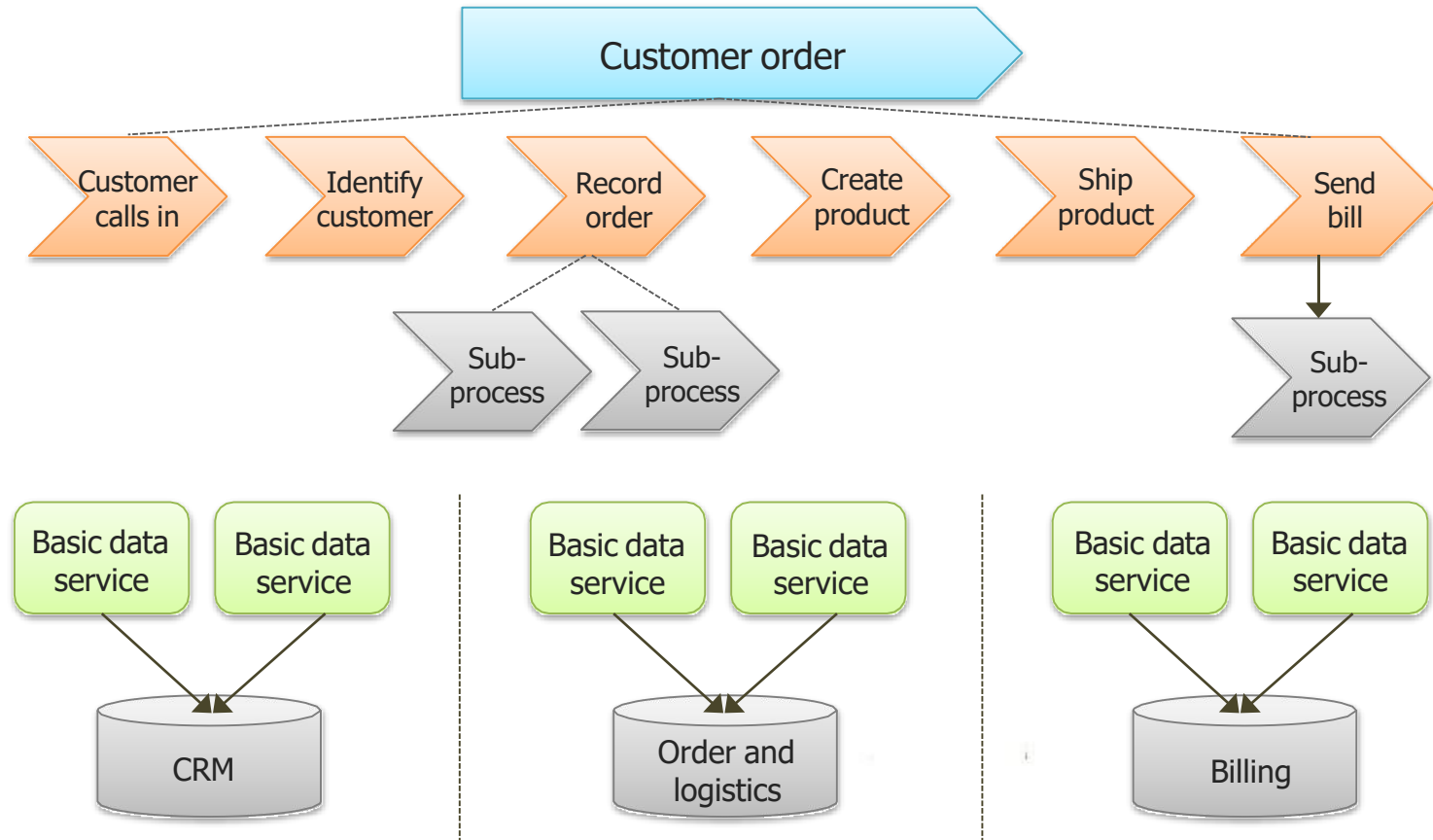
Lowest level of activities of a business process are services. From the business perspective, it is really not an issue whether the service is basic or composed. What matters is whether the service is doing the required work / task or not.

# Example of BPM with Services

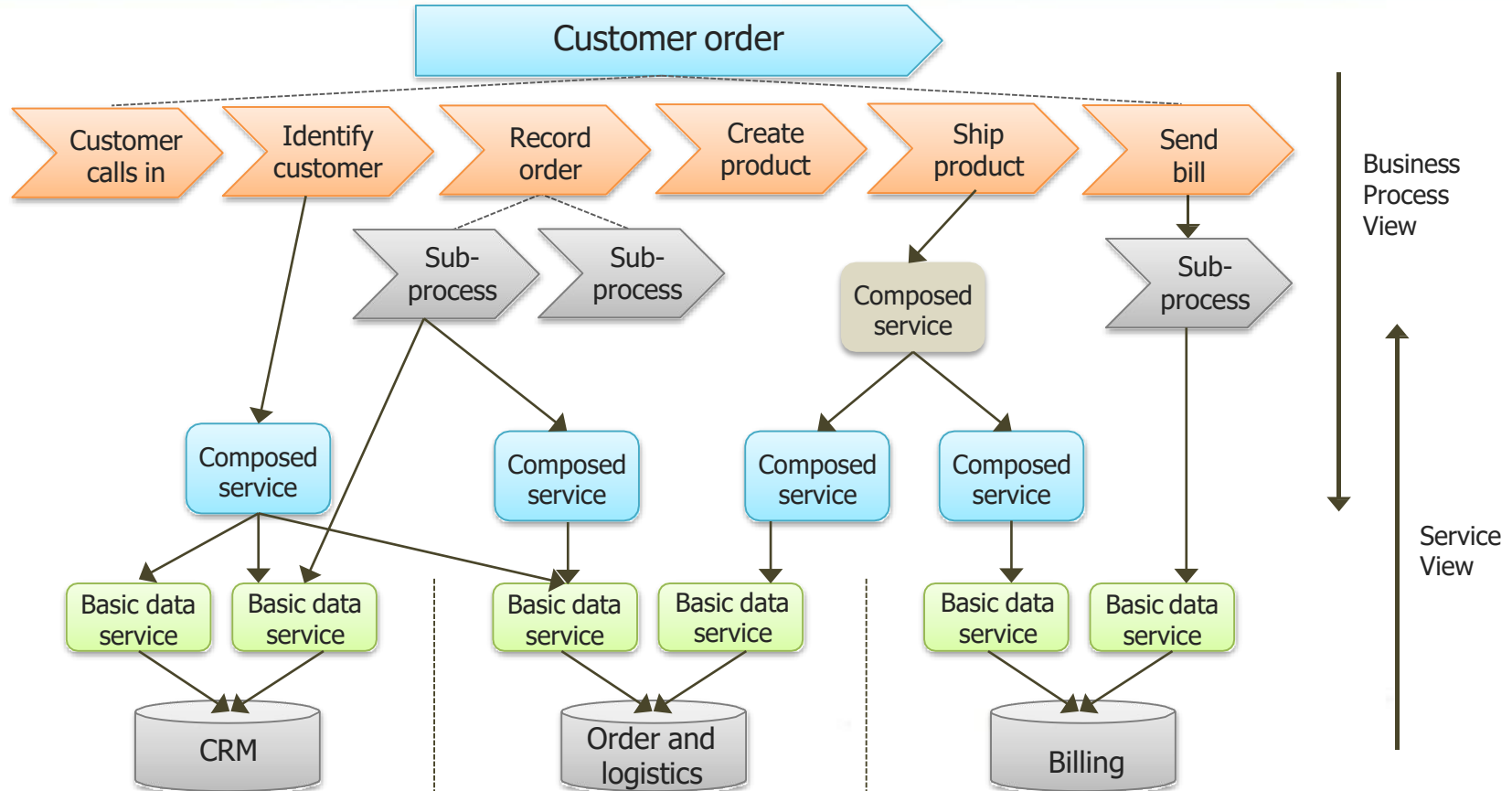
Starting point for a new business process



# Business Process – High level or Solution Design



# Business Process with Service View



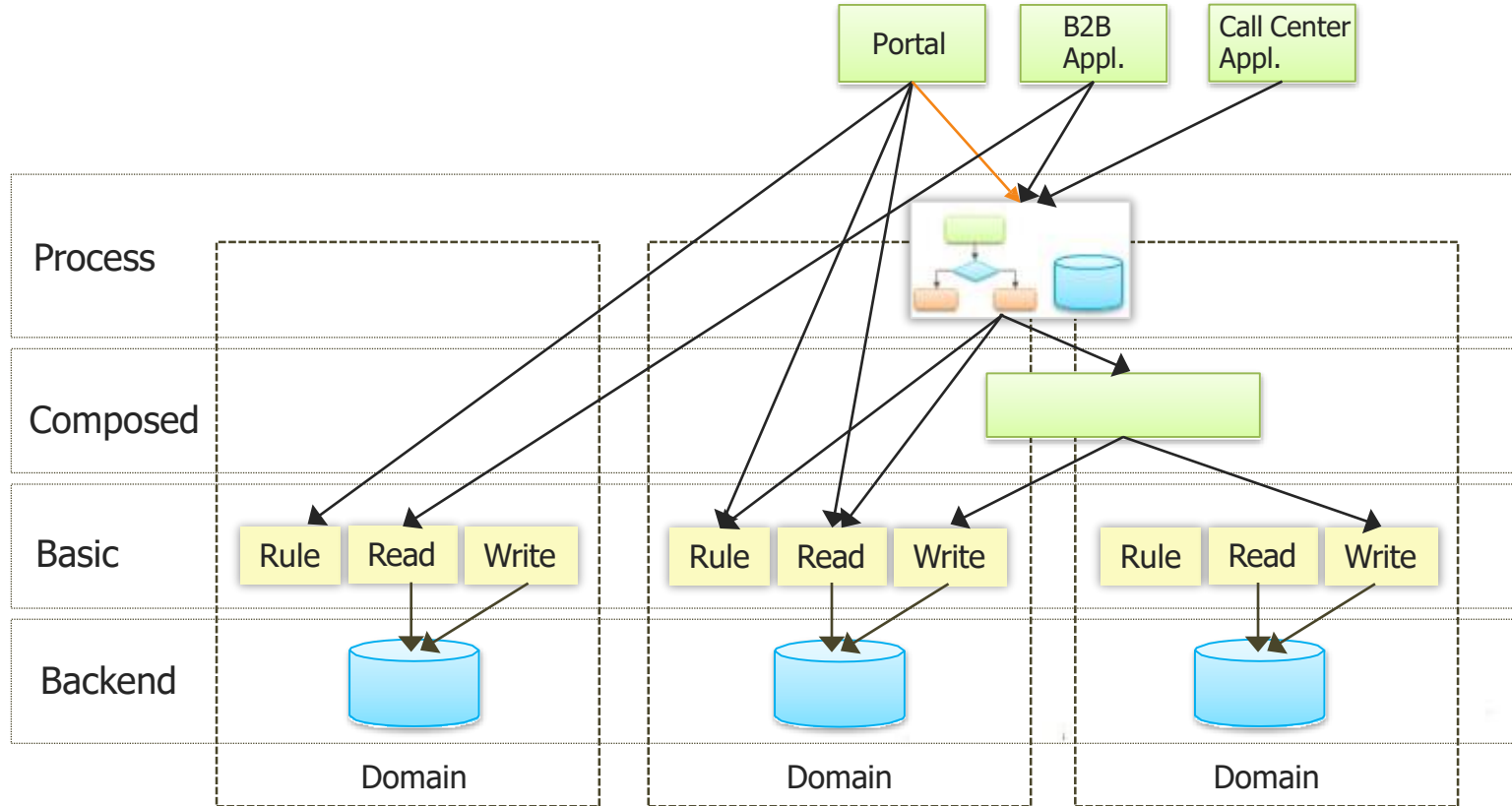
# SOA Based Architecture Models

---

SOA Architecture can be seen from different perspective based on the need. They are:

- Business or Logical based
- Mixed
- Technical

# SOA – Logical Architecture Model





# SOA – Logical Architecture Model

---

- In the Process section, all the processes are implemented.
- In the Composed section, all the composed services are defined.
- In the Basic section, all the basic data services are defined.
- In the back end section, all the back end databases are listed defining the type of business data to be stored.

# Mixed Architecture Models

If some technical details has to be included as part of providing domain / logical view of SOA then mixed model can be used.

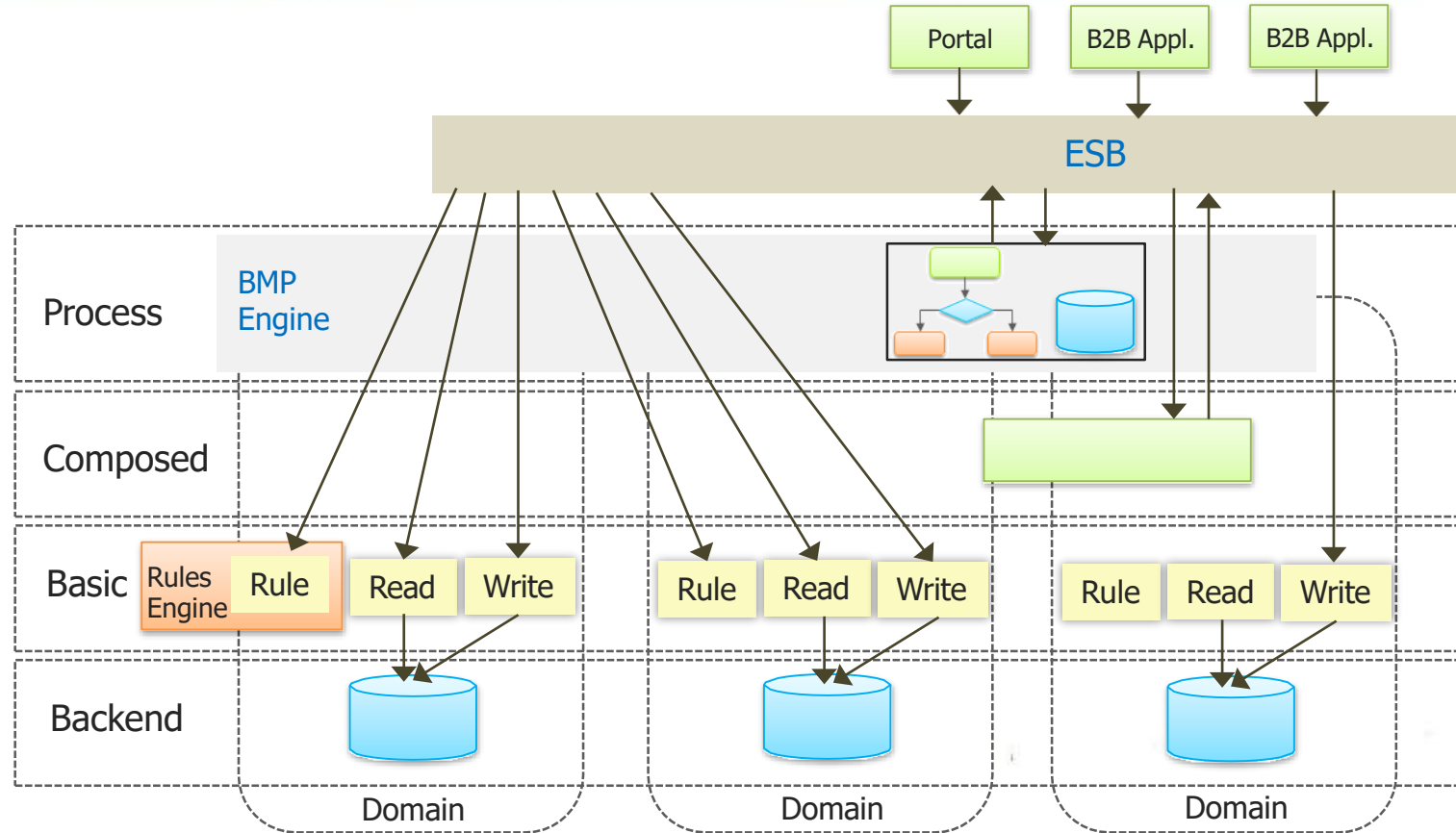
In the figure (in the next slide) you can see both the technical and logical view aspect.

- All the service calls are routed through ESB.
- Process services are implemented through BPM Engine.
- All basic logic services providing business rules are implemented in a rules engine.

Although technical information is there in the view, it also demonstrates the overall picture of SOA architecture model.

Logical view is not contradicting with mixed view and vice versa. They are just different views of the same landscape.

# Mixed Architecture Model



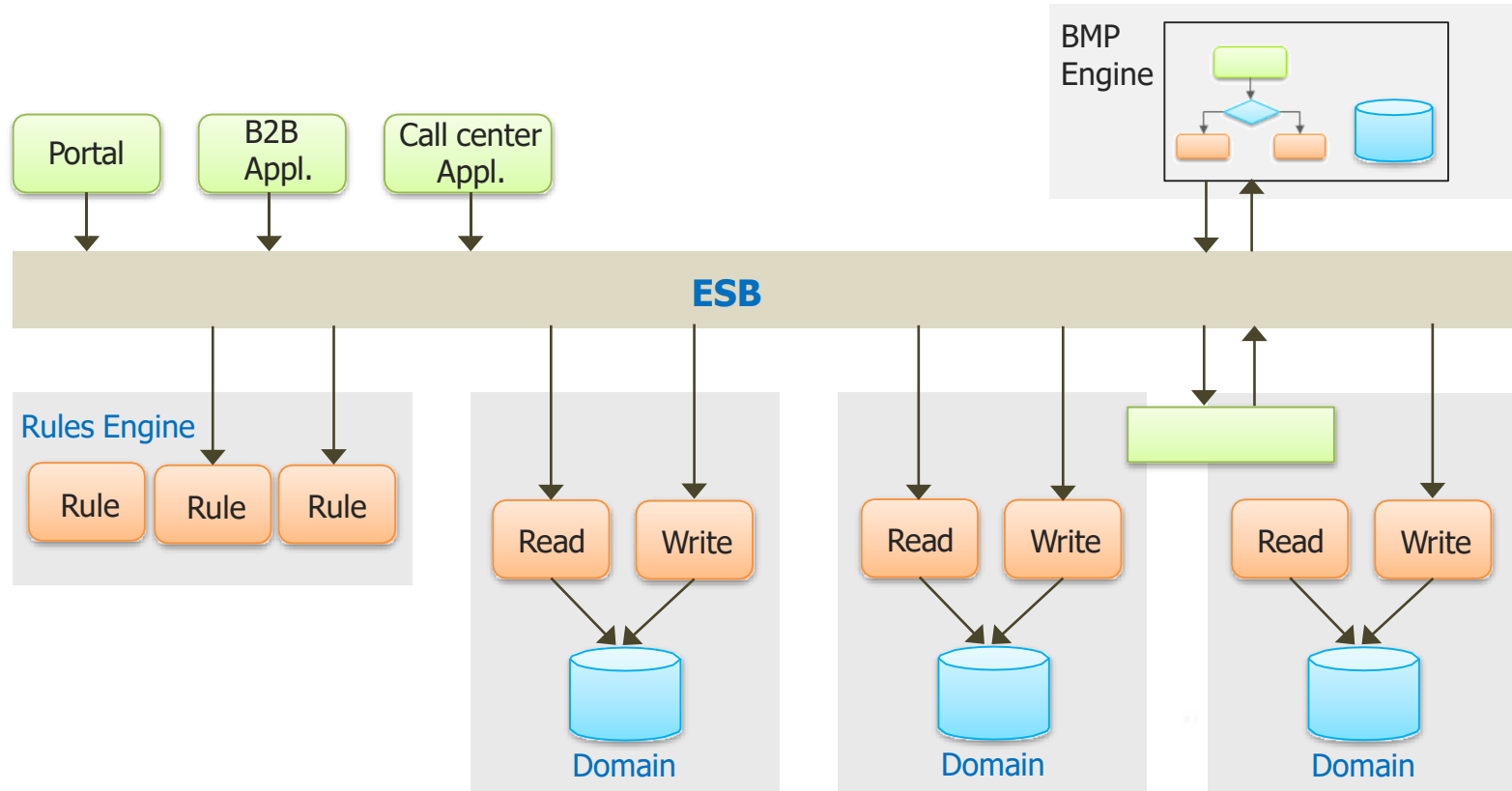
# Technical Architecture Model

In the technical architecture model, more technical details are provided in the view.

The changes in this view are given below:

- ESB is in the center.
- Domains are given with basic and composed services.
- Processes are separated as it is a technical view.

# Technical Architecture Model



# Message Exchange Patterns

---

There are different ways of exchanging data between distributed systems.

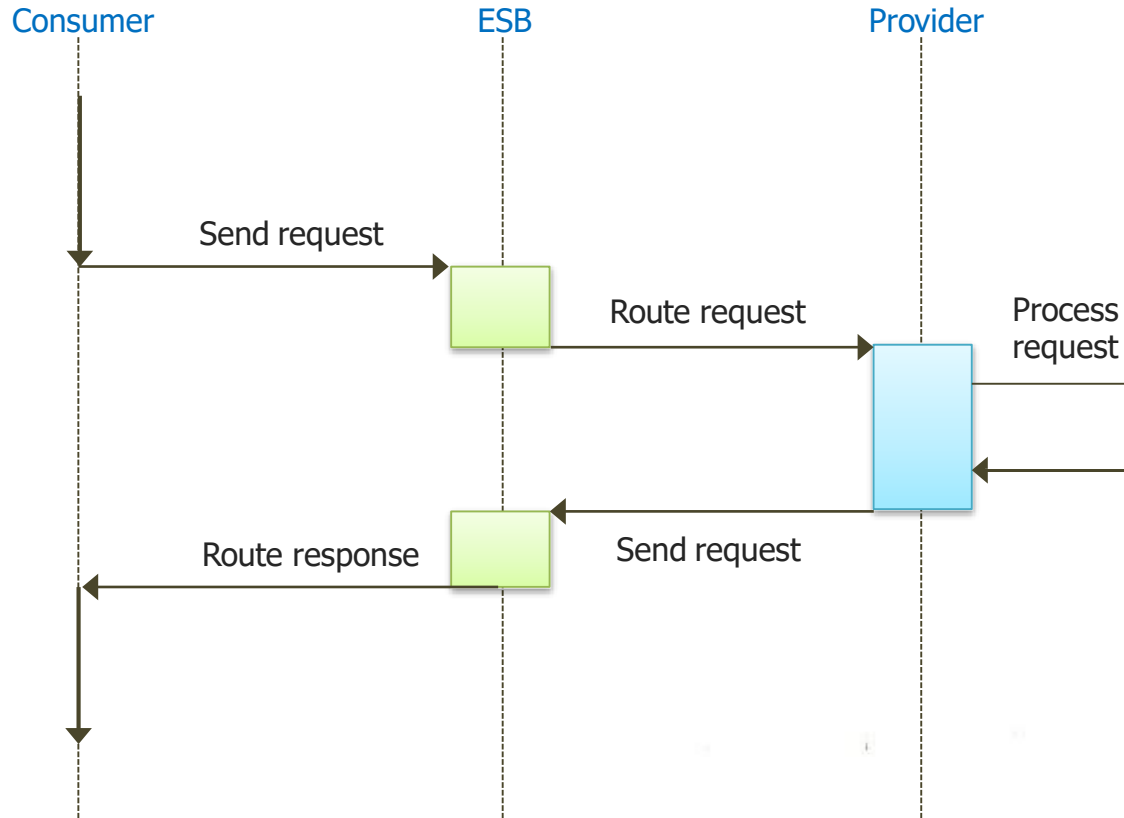
One approach of dealing with these differences is to categorize the way chunks of data are exchanged.

These chunks of data are called messages.

Different ways of exchanging messages is called Message Exchange Patterns.

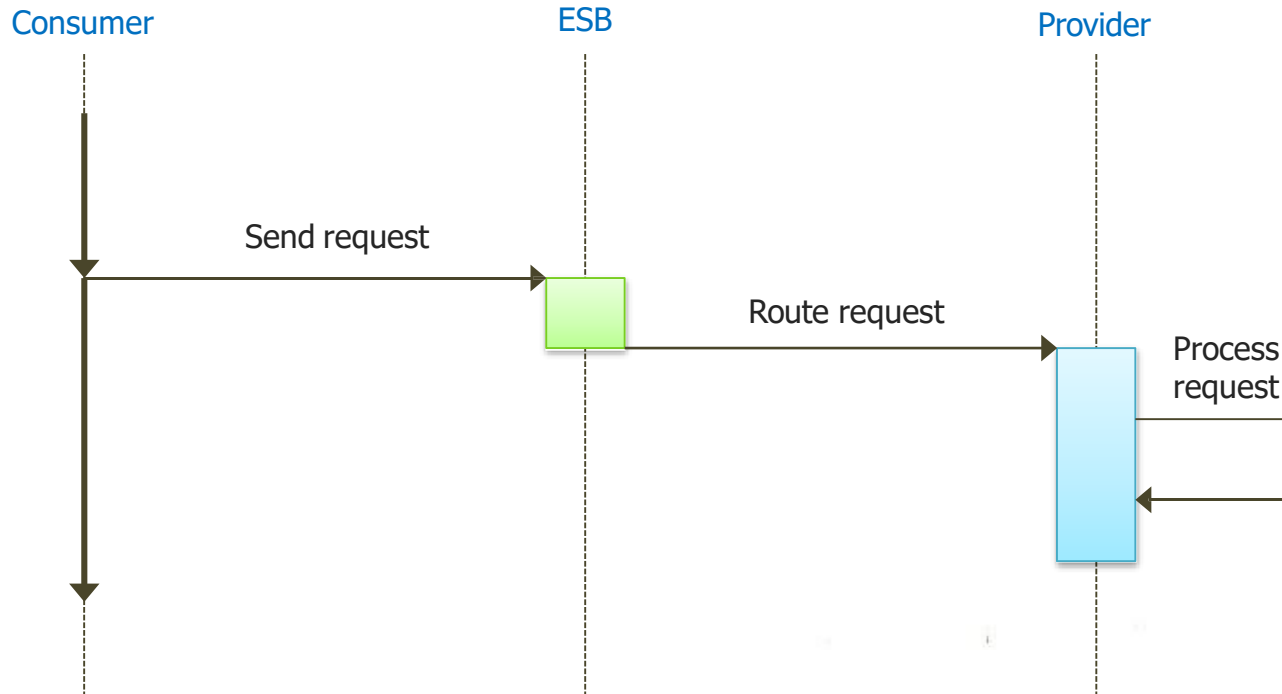
MEPs define the sequence of messages being called. It will specify the order, direction of those messages.

# Basic MEP – Request / Response (Two way)



# MEP – One Way

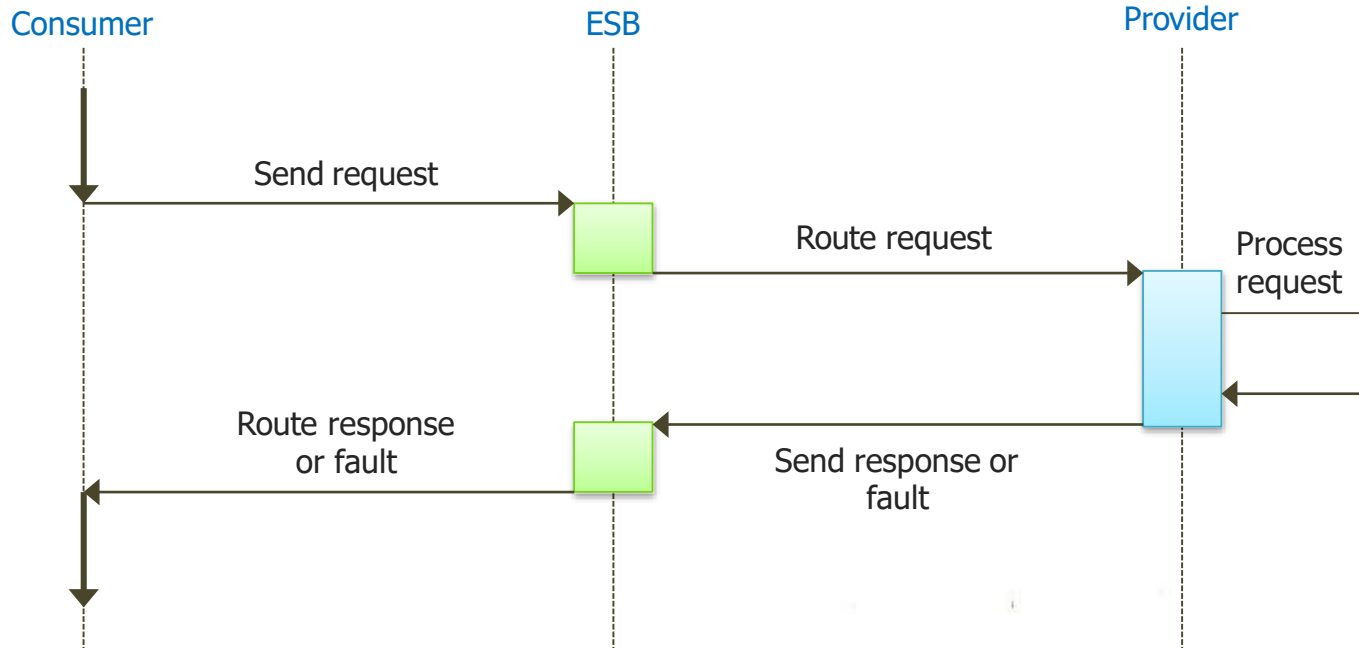
Here you can see the communication is happening in only one way. Response is not returned.





# MEP – Fault Messages

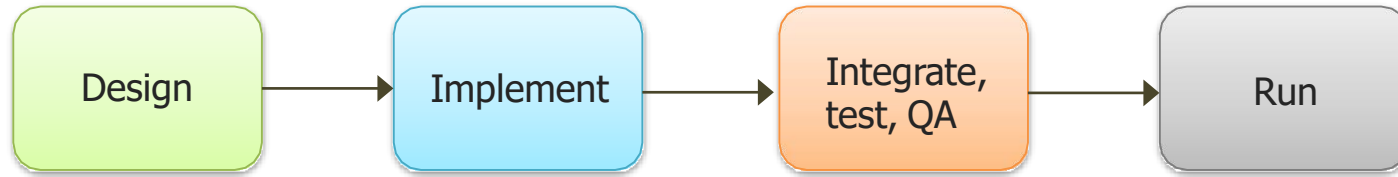
When there is error in the given input or when there is an error after processing, the service provider can specify the faulty message rather than generic message which helps the consumer to fix it.



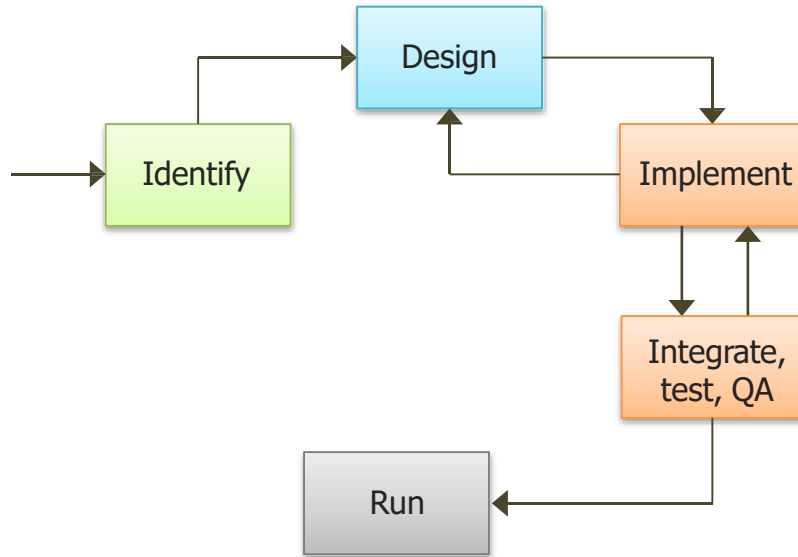
# Service – Life Cycle

Implementing Services: It has usual software development Life cycle as:

Analysis, design, implementation, Integration and deploying in production.



# Service Identification



As some companies want to provide some services. These services will be identified and developed.

Service should be identified based on its need.

In the picture, Service is identified. Service is designed. It is implemented.

It is iterated again for the changes as in iterative process development. After the completion of coding service is tested and bugs are fixed and deployed for usage.

# Service – Withdraw

---

If a service is no longer required as the new version of it is available.

Then the old version of service should be marked as deprecated and should be informed to the customers.

In the next release deprecated services can be removed when all the customers are migrated to the new version of the service.

In large companies, this process takes time as many customers would be using the old services and they will not be willing to move to the new version of the services.

# SOA – Performance

---

There are different places where performance comes into play in the context of services and SOA architecture.

If the performance becomes critical then it is usually time to execute the service is high.

Care must be taken while designing the components of SOA (Business and Technical) which should take as less time as possible.

# QUESTIONS



# Assignment

Assume that you are creating and developing a service for Credit Card customers.

Customers can check their credit card information/amount to be paid using internet. For this functionality, develop the Business Functions and Technical Service Level Functions using SOA.



# Topics of the Next Modules


In the next modules, you will be able to:

- Understand Web Services and its types
- Implement WSDL File
- Create a SOAP based web service
- Create a RESTful web service
- Get the glimpse of the final Project





# Pre-work

 Preparation for the next module: Go through the concepts of web Services.



Thank you!

