BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Introduction to Enterprise Application Integration (EAI)

Madhu Venkat

Guest Faculty
BITS, WILP

In this segment

Introduction to Enterprise Application Integration

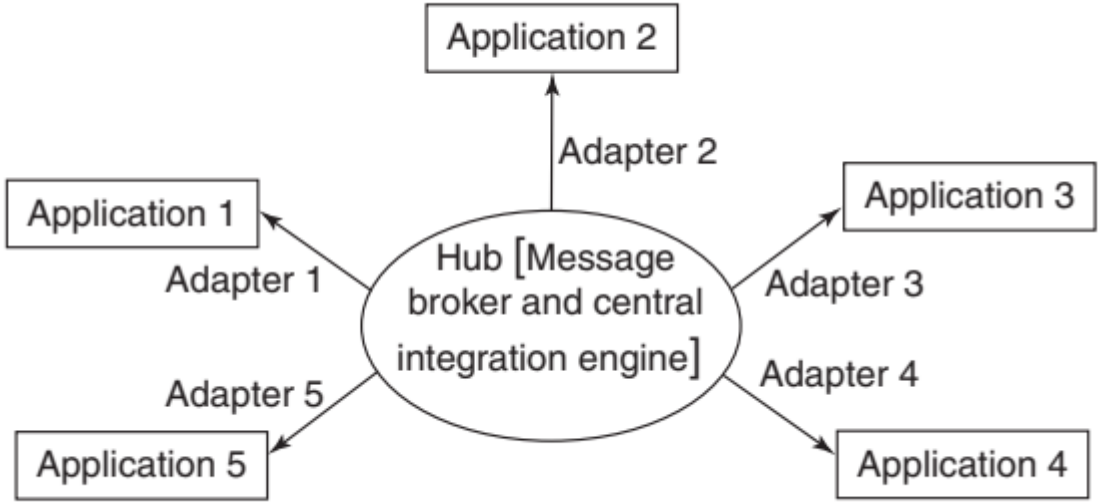
- EAI – Overview
- Message Channels
  - Point to Point messaging
  - Publish-subscribe model

Introduction to EAI

- Enterprise application integration allows diverse applications in an enterprise to communicate with each other to achieve a business objective in a seamless reliable fashion irrespective of platform and location of these applications.
- EAI comprises message acceptance, transformation, translation, routing, message delivery and business
- To achieve this,either Hub/Spoke or Bus Architecture can be used.

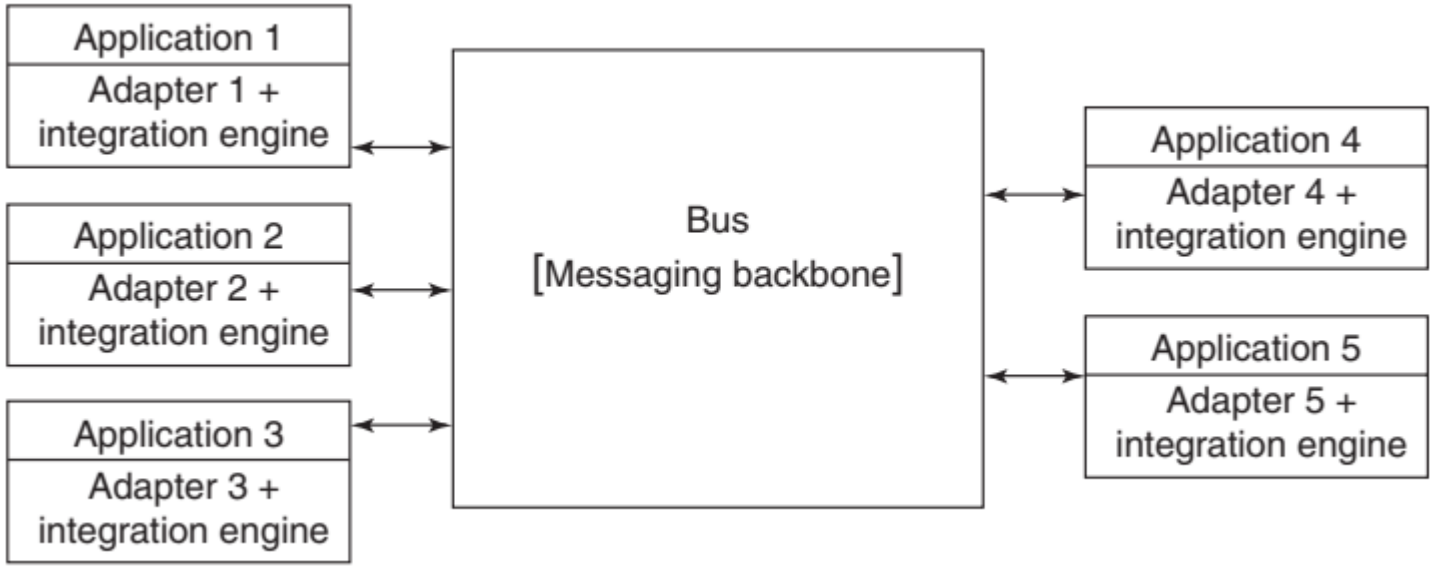
Hub/Spoke Architecture

- Hub has the centralized integration engine, which makes system with this architecture easy to manage and maintain but loses on scalability.



Bus Architecture

- In the bus architecture, adapters have integration engine and run on the same platform on which source and target applications run; and therefore it scales better but it is difficult to maintain.



Enterprise Application Integration (EAI)

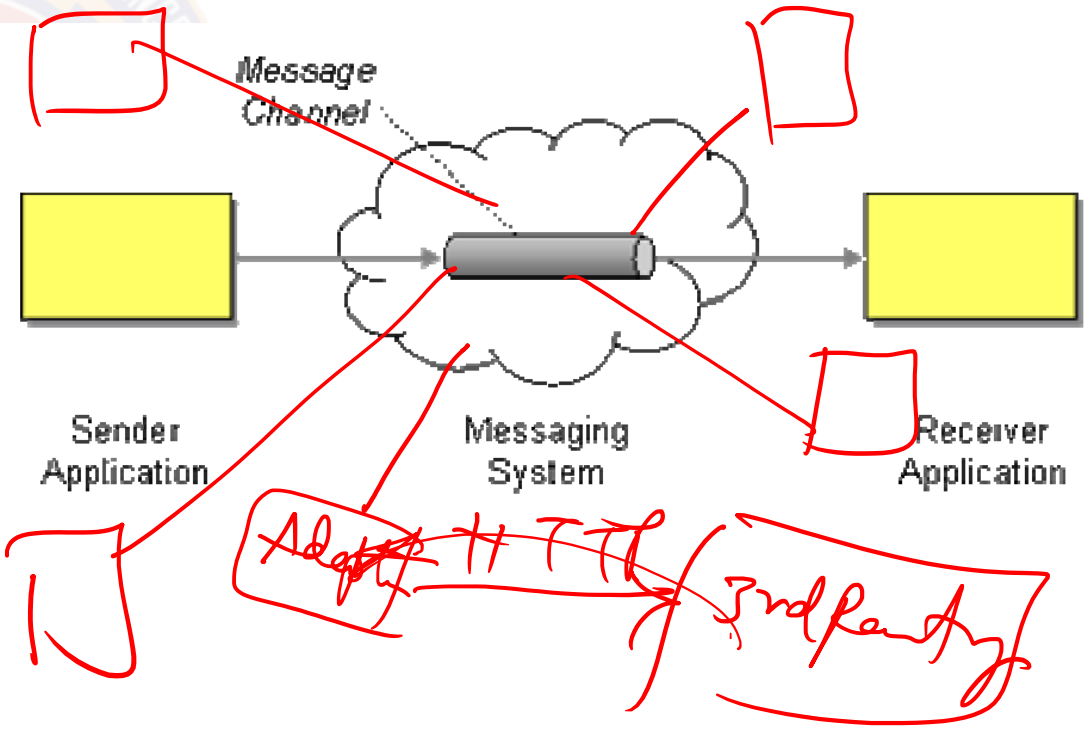
Introduction

- An integration framework composed of a collection of technologies and services which form a middleware or "middleware framework" to enable integration of systems and applications across an enterprise
- *"unrestricted sharing of data and business processes among any connected application or data sources in the enterprise" - Gartner*
- Application of EAI
  - Data integration
  - Vendor/3rd party independence
  - Common façade
- Patterns
  - Mediation
  - Federation

Enterprise Application Integration (EAI)

Message Channels - Overview

- Message Channel works as a logical addressing system in Messaging
- Messaging Channel acts as a medium of communication between two specific applications in a complex enterprise, where in the sender application selects which particular ”channel” of the messaging system to use, that designates the target application/ group of applications.
- Design challenges
  - One to one or one to many
  - Data type channel
  - Invalid and dead message handling
  - Crash proof design / Guaranteed delivery
  - Non-messaging client access - Channel adapter
  - Communications backbone – Message bus



Enterprise Application Integration (EAI)

Message Channels – Example (JMS)

- Configuring Message channels (Queues and Topics)

```
j2eeadmin -addJmsDestination jms/mytopic topic
j2eeadmin -addJmsDestination jms/myqueue queue
```

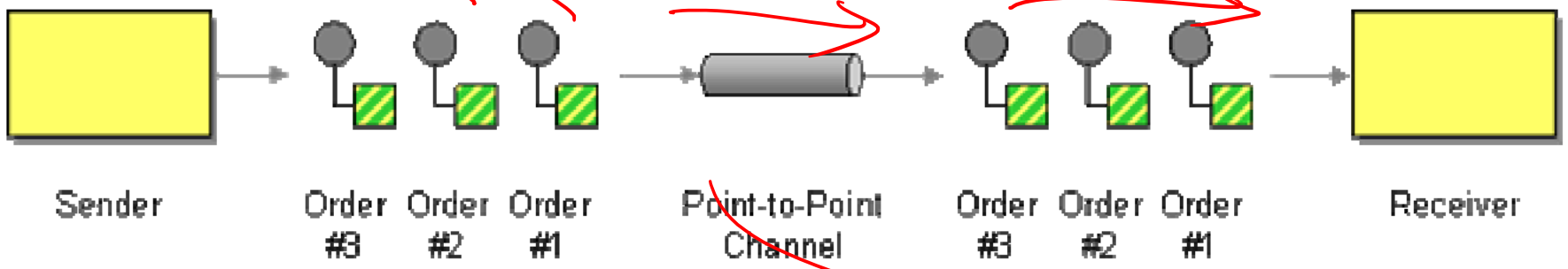
- Accessing Message channels

```
Context jndiContext = new InitialContext();
Queue myQueue = (Queue) jndiContext.lookup("jms/myqueue");
Topic myTopic = (Topic) jndiContext.lookup("jms/mytopic");
```

Enterprise Application Integration (EAI)

Point to point channel

- Point-to-Point Channel ensures that only one receiver consumes any given message, irrespective of number of receivers in that channel
- Increases scalability in multi consumer applications
- JMS Implementation uses 'Queue' interface to implement Point to point channel
  - QueueSender – send messages
  - QueueReceiver – receive messages



Enterprise Application Integration (EAI)

Point to point channel – Example (JMS)

- Sending message

```
Queue queue = // obtain the queue via JNDI
QueueConnectionFactory factory = // obtain the connection factory via JNDI
QueueConnection connection = factory.createQueueConnection();
QueueSession session = connection.createQueueSession(true, Session.AUTO_ACKNOWLEDGE);
QueueSender sender = session.createSender(queue);

Message message = session.createTextMessage("The contents of the message.");

sender.send(message);
```

Enterprise Application Integration (EAI)

Point to point channel – Example (JMS)

- Receiving message

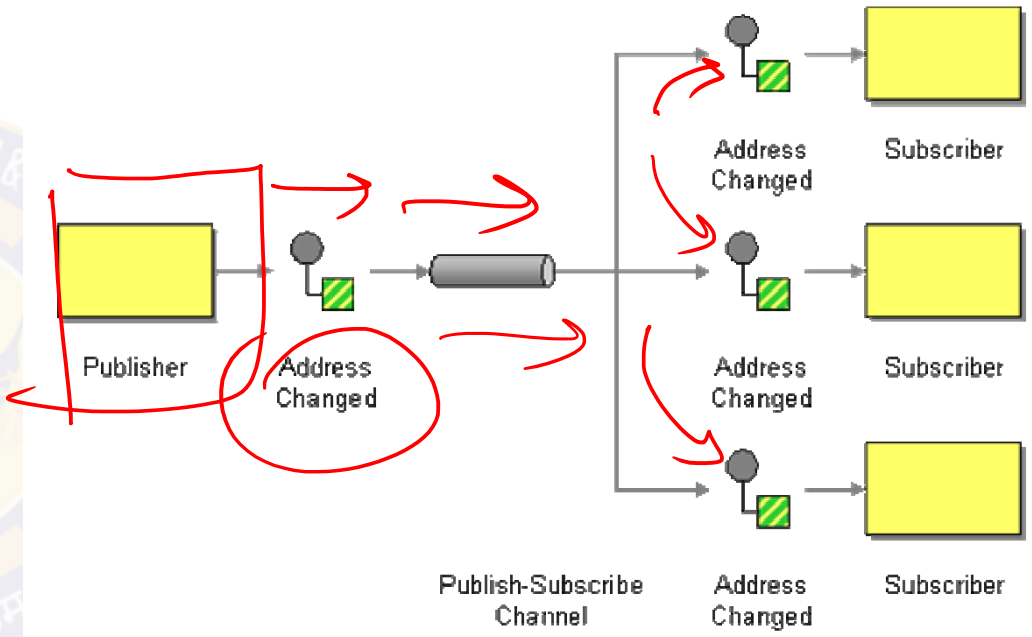
```
Queue queue = // obtain the queue via JNDI
QueueConnectionFactory factory = // obtain the connection factory via JNDI
QueueConnection connection = factory.createQueueConnection();
QueueSession session = connection.createQueueSession(true, Session.AUTO_ACKNOWLEDGE);
QueueReceiver receiver = session.createReceiver(queue);

TextMessage message = (TextMessage) receiver.receive();
String contents = message.getText();
```

Enterprise Application Integration (EAI)

Publish - subscribe channel

- Used for broadcasting messages to each recipient in the channel
- Ensures that each recipient gets only one copy of the message
- Helps in debugging channel without disturbing traffic
- JMS Implementation uses 'Topic' interface to implement Point to point channel
  - TopicPublisher – send messages
  - TopicSubscriber – receive messages



Enterprise Application Integration (EAI)

Publish - subscribe channel – Example (JMS)

- Sending message

```
Topic topic = // obtain the topic via JNDI
TopicConnectionFactory factory = // obtain the connection factory via JNDI
TopicConnection connection = factory.createTopicConnection();
TopicSession session = connection.createTopicSession(true, Session.AUTO_ACKNOWLEDGE);
TopicPublisher publisher = session.createPublisher(topic);

Message message = session.createTextMessage("The contents of the message.");
publisher.publish(message);
```

Enterprise Application Integration (EAI)

Publish - subscribe channel – Example (JMS)

- Receiving message

```
Topic topic = // obtain the topic via JNDI
TopicConnectionFactory factory = // obtain the connection factory via JNDI
TopicConnection connection = factory.createTopicConnection();
TopicSession session = connection.createTopicSession(true, Session.AUTO_ACKNOWLEDGE);
TopicSubscriber subscriber = session.createSubscriber(topic);

TextMessage message = (TextMessage) subscriber.receive();
String contents = message.getText();
```

Thank You!

In our next session:

Middleware Security