1 : Introduction to Enterprise Application Development

IT4206 – Enterprise Application Development

Level II - Semester 4

Overview

- This topic will discuss the enterprise applications and the reasons to use JavaEE for the development of enterprise applications. It will gives a brief introduction to the component based application development and Enterprise JavaBeans (EJB).

Intended Learning Outcomes

- At the end of this lesson, you will be able to;
  - Explain enterprise application development.
  - Discuss the use of JavaEE in enterprise application development.
  - Compare JavaSE and JavaEE.
  - Explain the Component Based Architecture.
  - Discuss the architecture of Enterprise JavaBeans (EJB).

List of sub topics

What are Enterprise Applications?

- *An enterprise application (EA) is a large software system platform designed to operate in a corporate environment such as business or government. EAs are complex, scalable, component-based, distributed and mission critical. EA software consists of a group of programs with shared business applications and organizational modeling utilities designed for unparalleled functionalities. EAs are developed using enterprise architecture. ~techopedia*

https://www.techopedia.com/definition/24804/enterprise-application-ea

Enterprise Applications - Some Examples

- Business Process Management Systems
- Customer Relationship Management Systems
- Human Resource Management Systems
- Supply Chain Management Systems

Enterprise Applications

- An enterprise applications are software applications typically used in organizations such as businesses, schools, governments and so on.
- Enterprise applications often provide the following features:
  - Support for concurrent users and external systems.
  - Support for scalability to handle future growth.
  - Support for highly secure access control for different types of users.
  - Ability to integrate with back-end systems and web services.

Some Programming Languages and Frameworks for Enterprise Application Development

- JavaEE
- Spring
- PHP
- Django
- JavaScript

Java EE for Enterprise Applications…(1)

- Platform-independent applications can be developed and will run on many different types of operating systems.
- Applications are portable across Java EE compliant application servers due to the Java EE standard.
- The Java EE specification provides for a large number of APIs typically used by enterprise applications such as web services, asynchronous messaging, transactions, database connectivity, thread pools, batching utilities, and security. There is no need to develop these components manually, thereby reducing development time.

Java EE for Enterprise Applications…(2)

- A large number of third-party, ready-to-use applications and components that target specific domains such as finance, insurance, telecom, and other industries are certified to run and integrate with Java EE application servers.
- A large number of sophisticated tools such as IDEs, monitoring systems, enterprise application integration (EAI) frameworks, and performance measurement tools are available for Java EE applications from third-party vendors.

Java EE

- Java EE is developed under the Java Community Process (JCP), an organization responsible for the development of Java technology. JCP members include Oracle (the current steward of the Java platform), and the Java community at large.

Java Community Process

- The Java Community Process (JCP) allows interested parties to assist in developing standard technical specification for Java technology. Both companies and individuals can become members of the JCP and contribute to any technical specification they may be interested in. Each Java EE API specification is developed as part of a Java Specification Request (JSR).

JCP - https://www.jcp.org/en/home/index

Read more : Java EE 8 Application Development (Page 32)

Java APIs…(1)

- Java EE is a collection of API specifications designed to work together when developing server-side enterprise Java applications.
- Examples for Java EE 8 APIs:
  - Servlet 4.0
  - Enterprise JavaBeans (EJB) 3.2
  - JavaServer Faces (JSF) 2.3
  - Java Persistence API (JPA) 2.2

More examples : Java EE 8 Application Development (Page 33)

Java APIs…(2)

- Servlet 4.0 - The servlet API is a low-level API used to implement server- side logic in web applications.
- Enterprise JavaBeans (EJB) 3.2 - EJB's allow us to easily add enterprise features such as transactions and scalability to our Java EE applications.
- JavaServer Faces (JSF) 2.3 - JSF is a component library that greatly simplifies the development of web applications.
- Java Persistence API (JPA) 2.2 - JPA is the Java EE standard Object-Relational Mapping (ORM) API. It makes it easy to interact with relational databases.

More examples : Java EE 8 Application Development (Page 34)

One standard, Multiple Implementations…(1)

- Java EE applications are typically deployed to an application server; some popular application servers include JBoss, Websphere, Weblogic, and GlassFish.

JBoss®

by Red Hat

WebSphere®

Application Server

ORACLE®

WebLogic

One standard, Multiple Implementations…(2)

- Each application server is considered to be a Java EE implementation. Application server vendors either develop their own implementation of the several Java EE API specifications or choose to include an existing implementation.
- Application developers benefit from the Java EE standard by not being tied to a specific Java EE implementation. As long as an application is developed against standard Java EE APIs, it should be portable across application server vendors.

Java EE, J2EE, and the Spring framework

- Java EE was introduced back in 2006; the first version of Java EE was Java EE 5.
- Java EE replaced J2EE; the last version of J2EE was J2EE 1.4, released back in 2003.
- Even though J2EE can be considered a dead technology, replaced by Java EE , the term J2EE refuses to die.
- Spring applications are referred to as J2EE applications, but it has never been, J2EE.

More description : Java EE 8 Application Development (Page 38)

Comparing Java Enterprise edition (Java EE) with Java Standard Edition (Java SE)

- Java SE is generally used to develop stand-alone programs, tools, and utilities that are mainly run from the command line, GUI programs, and server processes that need to run as daemons.
- The Java EE specification is a set of APIs built on top of Java SE. It provides a runtime environment for running multi-threaded, transactional, secure and scalable enterprise applications. It is important to note that unlike Java SE, Java EE is mainly a set of standard specifications for an API, and runtime environments that implement these APIs are generally called as application servers.

Component Based Development…(1)

- Component based development (CBD) is a branch of software engineering which emphasizes the importance of building software with the aid of reusable software components.
- CBD involves in developing software systems by assembling these components in well defined software architectures.
- Components can be
  - Off the shelf software
  - Commercial tailormade
  - Self-developed

Component Based Development…(2)

- Object-oriented modeling results in
- Classes
- Objects
- Relationships

- Common Issue : It is very hard to discover reusable parts among these smaller units.

- CBD integrate the related parts and reuse them collectively. These integrated parts are known as components.

Separation of Concerns…(1)

- Is a design principle for separating a computer program (when considering software engineering) into distinct sections in such a way that each such section addresses a separate concern.
- Generally can be considered in three perspectives when considering software architecture,
  - Functions
  - Classes
  - Components

Separation of Concerns…(2)

- What would happen if separation of concerns where not adhered to even at a minimal level?
  - Spaghetti Code
  - God Classes / Objects
  - Monoliths

Separation of Concerns…(3)

- Spaghetti Code - are unstructured and difficult to read. Consider the following sample code.

```
1  x=0
2  x=x+1
3  PRINT x; “Number = “;x
4  IF i>=10 THEN GOTO 6
5  GOTO 2
6  PRINT “Counter Completed”
7  END
```

- It can rewrite as follows. Which is more structured and readable?

```
1  FOR x=1 TO 10
2     PRINT x; “Number = “;x
3  NEXT x
4  PRINT “Counter Completed”
5  END
```

Separation of Concerns…(4)

- God Classes / Objects - *knows too much and does too much*.

```
public class employee {
        public int ID;
        public string Firstname;
        public string Lastname;
        public string Address;
        public string City;
        public string Postalcode;
        public string County;
}
```

- This can be refactor to smaller and manageable parts.

Separation of Concerns…(6)

- God Classes / Objects

```
public class employee {
        public int ID;
        public string Firstname;
        public string Lastname;
        public Address Address;
}
```

```
public class Address {
        public string City;
        public string Postalcode;
        public string County;
}
```

Component Based Development…(3)

- CBD is done within a middleware infrastructure which supports the component based software construction process.
- E.g. Enterprise Java Beans

- Advantages of CBD
  - Minimized Delivery
  - Improved Efficiency
  - Improved Quality
  - Minimized Expenditure

The Current Context

- If a business application works to its specification only in the modern day, is it considered good software?

- Some characteristics of good software (modern context)
  - Secure
  - Interoperable
  - Scalable
  - Robust

- When considering the present day software, all code can be broadly categorized into following aspects.
  - Core Concerns
  - Cross-Cutting Concerns
  - Plumbing

Core Concerns

- Represents the core business logic of the application which is developed by a team internal to the organization/ business.

Cross-Cutting Concerns

- These are the secondary operations that are necessary to keep the overall system running correctly and efficiently.

Core business - Window cleaning,
Cross-cutting concern - Safety

Plumbing

- Plumbing enables getting the data and invocations from one point in the software to another.

Enterprise JavaBeans (EJB)

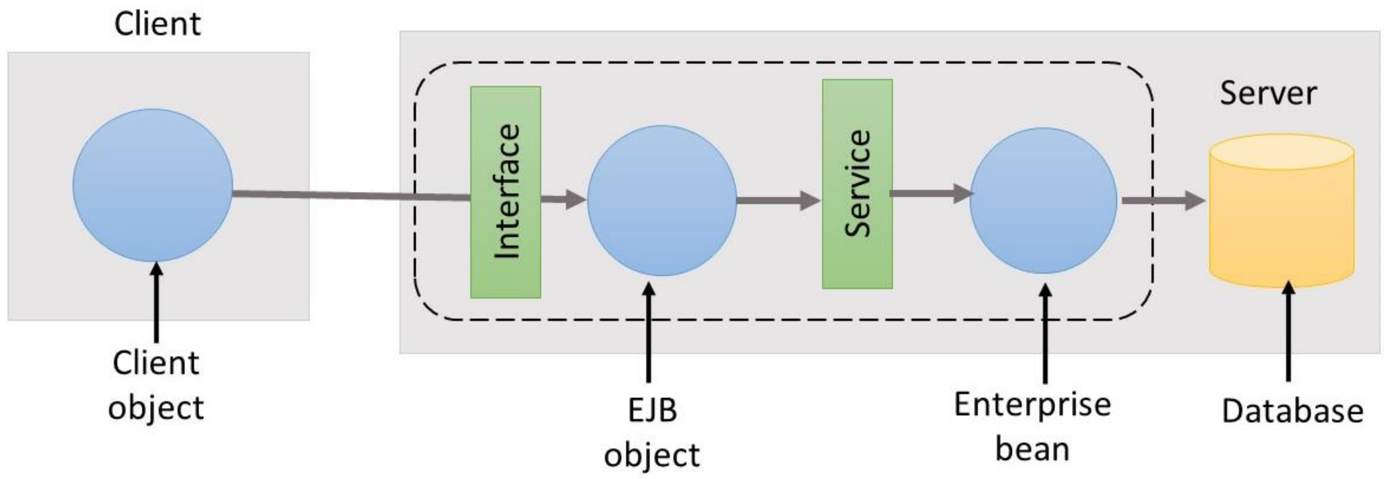
- EJB components (beans) can be assemble and reassemble into different applications.
  - Eg : Customer bean - represents a customer in a database.
- Customer bean can be used in accounting program, e-commerce shopping cart system, tech support application, etc.
- Developers can build different beans (payroll bean, shopping cart bean, order bean etc) and used in their own applications or sell them to other developers.
- Instead of reusing Java classes, developers can reuse a bidder chuck of functionality when using beans. It con be modified without ever touching its java code.

What can we get from EJB

- EJB let the developer to concentrate more on the business logic, and leave the underlying services to the EJB server vendor.
- Services provided by EJB servers - Transaction management, Security, Concurrency, Networking, Resource management, Persistence, Messaging, Deploy-time customisation.
- Developer can customise and configure reusable components at deployment time without changing the source code.
- EJB are portable. Can be used in different JVMs and EJB servers. Write-once-deploy-anywhere (WODA).

How does EJB works?

- The Beans run under the control and protection of the EJB server.
- When the client object request service from the bean, the server virtually manage the transactions, security, persistence and other services.



Types of Beans…(1)

- Entity - It represent a thing in the persistent store. An instance of entity beans is a row in a table.
  - Eg: a customer (with a customer id, name. Etc).
- Message-driven - This beans can listen for messages from Java message service (JMS). Client cannot directly call to a message driven bean, It has to send a message to a messaging service. So the server get the request from the messaging service, not a direct call from client to the message-driven bean.

Types of Beans…(2)

- Session - Session beans represent a process. Any kind of a back end service should begin with a session bean.
    - Eg: shopping session, credit card processing system. There are stateless and stateful session beans. Stageful bean can remember the conversational state between method calls. Stateless beans won’t remember.
- Singleton session beans are session beans that are instantiated once per application and exists for the lifecycle of the application. Every client request for a singleton bean goes to the same instance. Singleton session beans are used in scenarios where a single enterprise bean instance is shared across multiple clients.