

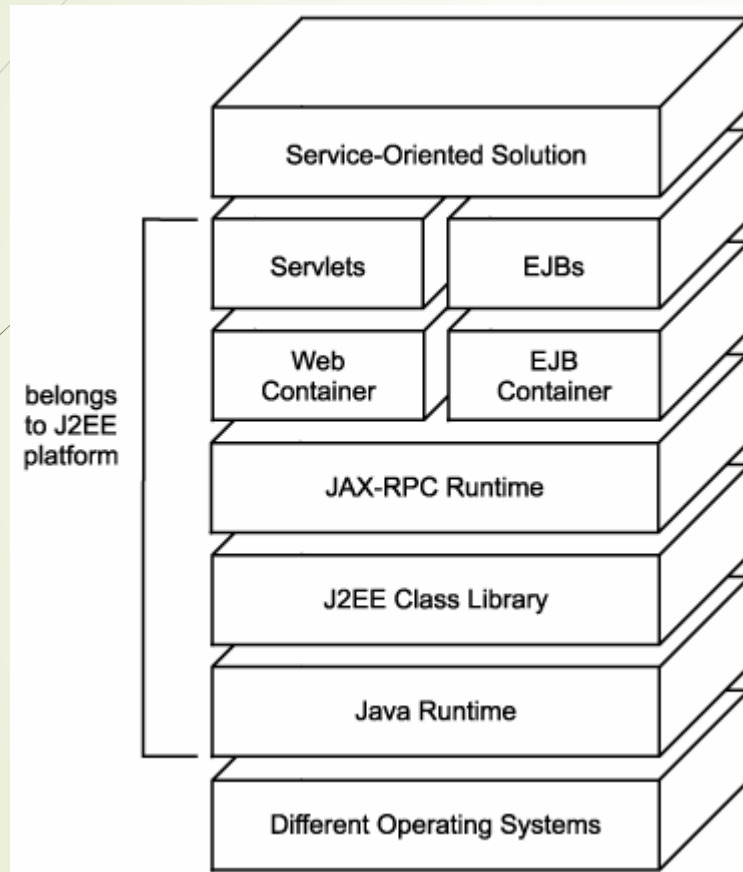


SOA Support in J2EE


Java 2 Platform Enterprise Edition (J2EE) – Used to develop enterprise solutions using Web services.

- Aneesh Gupta (2021UCS1720)

Layers of J2EE which relate to SOA



- Servlets, EJBs, Web Containers, EJB Container layers, JAX-RPC Runtime relate to the Web and Component Technology layers.
- They do not map cleanly to these layers because to what extent component and Web technology is incorporated is largely dependent on how a vendor chooses to implement this part of a J2EE architecture.



J2EE specifications that pertain to SOA -

- **J2EE Specification** - Establishes the distributed J2EE component architecture and provides foundation standards that J2EE product vendors are required to fulfill in order to claim J2EE compliance.
- **Java API for XML-based RPC (JAX-RPC)** - This document defines the JAX-RPC environment and associated core APIs. It also establishes the Service Endpoint Model used to realize the JAX-RPC Service Endpoint, one of the primary types of J2EE Web services.
- **Web Services for J2EE** - Defines the vanilla J2EE service architecture and clearly lays out what parts of the service environment can be built by the developer, implemented in a vendor-specific manner, and which parts must be delivered according to J2EE standards.

Architecture Components

Following components are used to build J2EE web applications -


- Java Server Pages (JSPs) - Dynamically generated Web pages hosted by the Web server.
- Struts - An extension to J2EE that allows for the development of Web applications with sophisticated user-interfaces and navigation.
- Java Servlets - These components also reside on the Web server and are used to process HTTP request and response exchanges.
- Enterprise JavaBeans - (EJBs) The business components that perform the bulk of the processing within enterprise solution environments. They are deployed on dedicated application servers and can therefore leverage middleware features, such as transaction support.

Presentation
layer

Web
Services



Runtime environments

- The J2EE environment relies on a foundation Java runtime to process the core Java parts of any J2EE solution. In support of Web services, J2EE provides additional runtime layers that, in turn, supply additional Web services specific APIs (explained later).
 - Most notable is the JAX-RPC runtime, which establishes fundamental services, including support for SOAP communication and WSDL processing.
- 



APIs

J2EE contains several APIs for programming functions in support of Web services. The classes that support these APIs are organized into a series of packages.

- Java API for XML Processing (JAXP) This API is used to process XML document content using a number of available parsers.
- Java API for XML-based RPC (JAX-RPC) The most established and popular SOAP processing API, supporting both RPC-literal and document-literal request-response exchanges and one-way transmissions.
- Java API for XML Registries (JAXR) An API that offers a standard interface for accessing business and service registries. Originally developed for ebXML directories, JAXR now includes support for UDDI.
- Java API for XML Messaging (JAXM) An asynchronous, document-style SOAP messaging API that can be used for one-way and broadcast message transmissions
- SOAP with Attachments API for Java (SAAJ) Provides an API specifically for managing SOAP messages requiring attachments.
- Java Architecture for XML Binding API (JAXB) This API provides a means of generating Java classes from XSD schemas and further abstracting XML-level development.

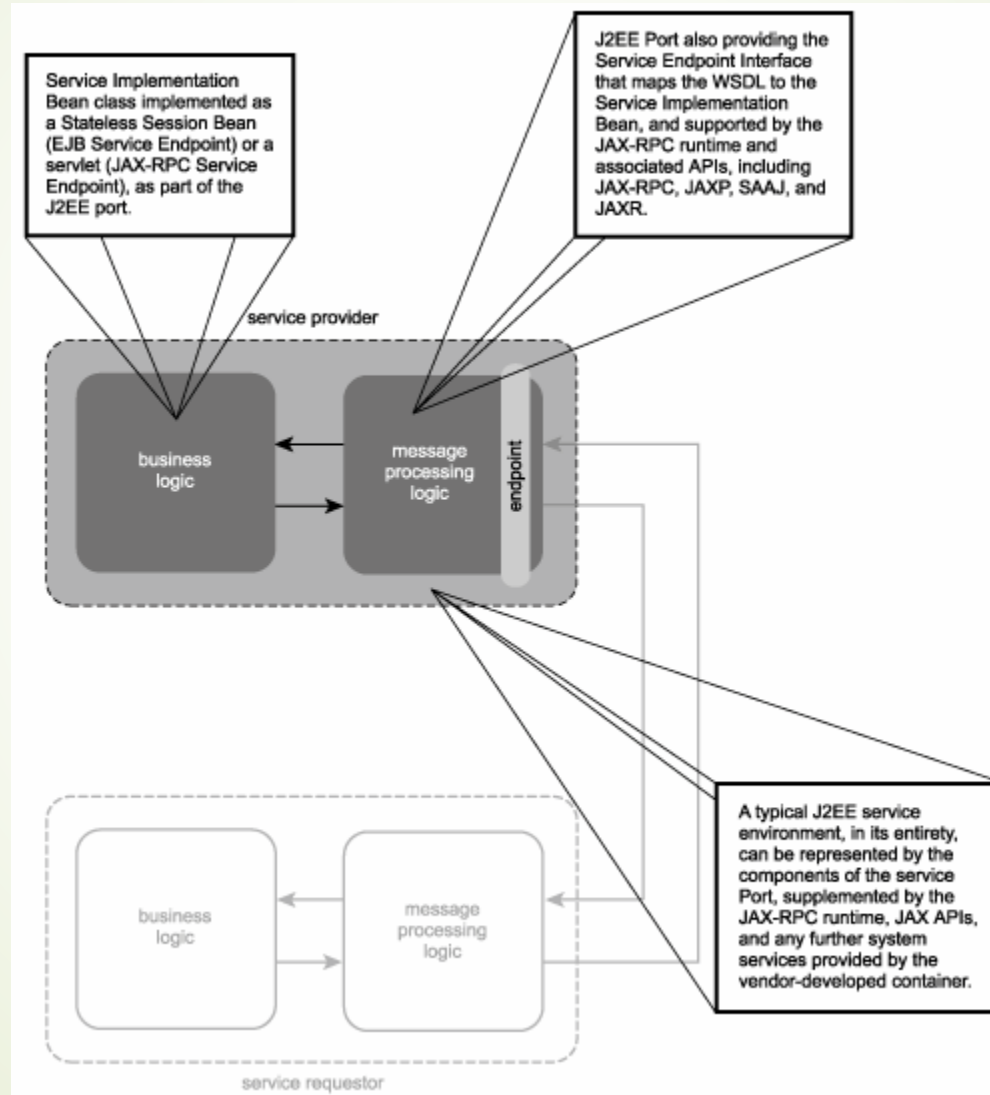


Service Providers

- JAX-RPC Service Endpoint When building Web services for use within a Web container, a JAX-RPC Service Endpoint is developed that frequently is implemented as a servlet by the underlying Web container logic.
- EJB Service Endpoint The alternative is to expose an EJB as a Web service through an EJB Service Endpoint. This approach is appropriate when wanting to encapsulate existing legacy logic or when runtime features only available within an EJB container are required.

Regardless of vendor platform, both types of J2EE Web services are dependent on the JAX-RPC runtime and associated APIs.

J2EE Service Provider



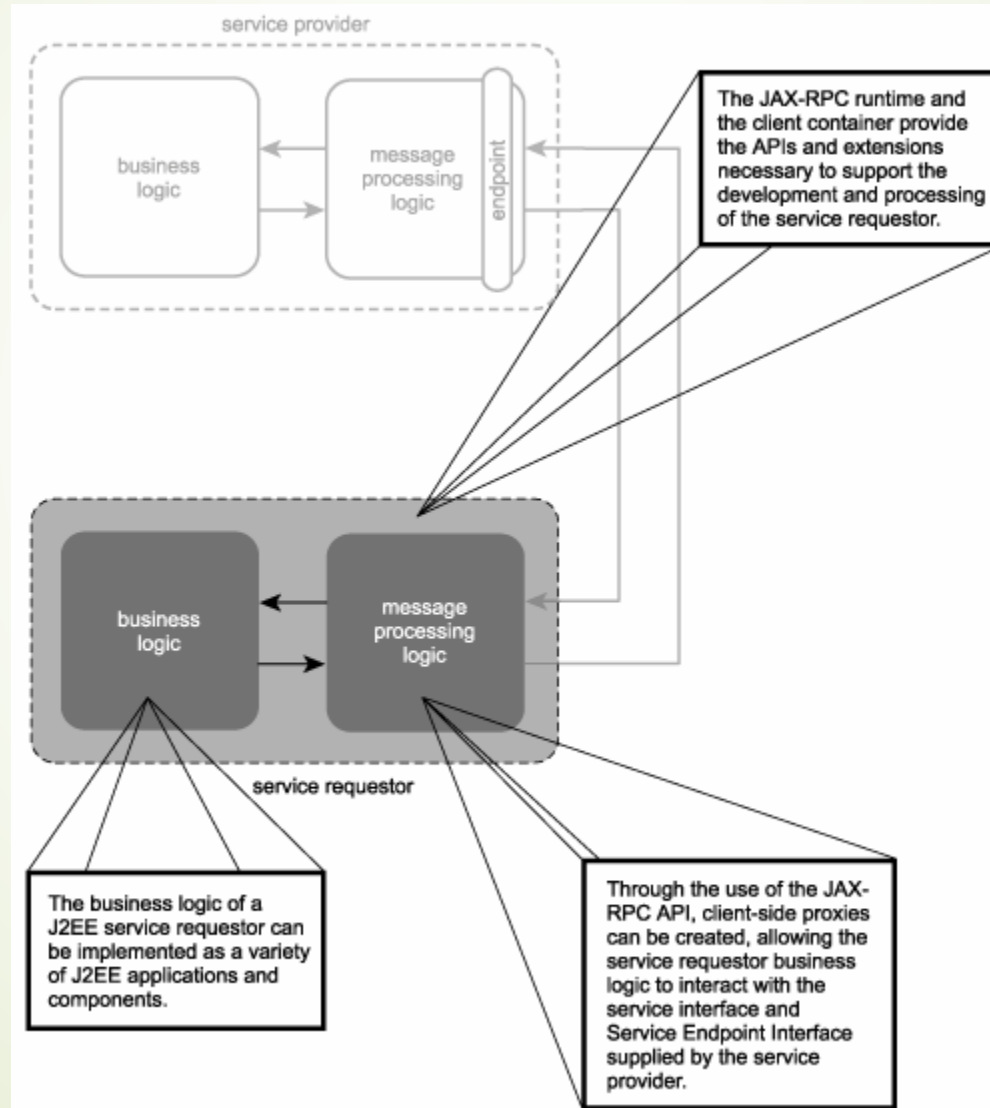


Service Requesters

The JAX-RPC API can be used to develop service requestors. It provides the ability to create three types of client proxies, as explained here:

- **Generated stub** The generated stub (or just "stub") is the most common form of service client. It is auto-generated by the JAX-RPC compiler (at design time) by consuming the service provider WSDL, and producing a Java-equivalent proxy component. Specifically, the compiler creates a Java remote interface for every WSDL portType which exposes methods that mirror WSDL operations.
- **Dynamic proxy and dynamic invocation interface** Two variations of the generated stub are also supported. The dynamic proxy is similar in concept, except that the actual stub is not created until its methods are invoked at runtime. Secondly, the dynamic invocation interface bypasses the need for a physical stub altogether and allows for fully dynamic interaction between a Java component and a WSDL definition at runtime.

Service Requester



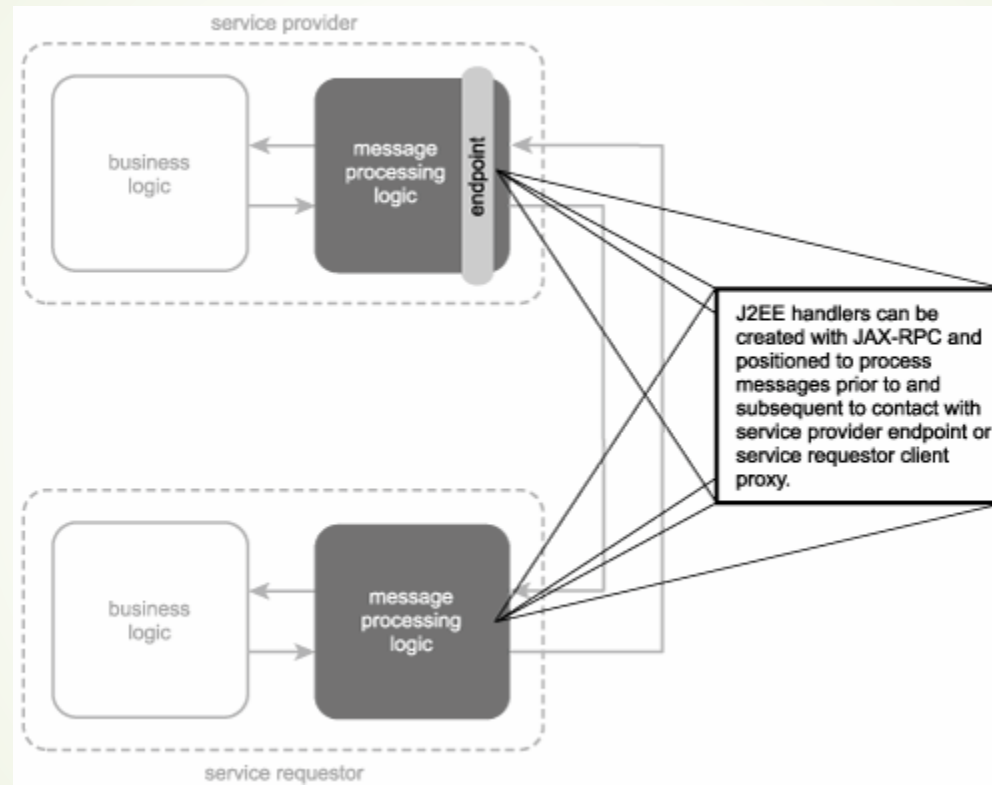


Service Agents



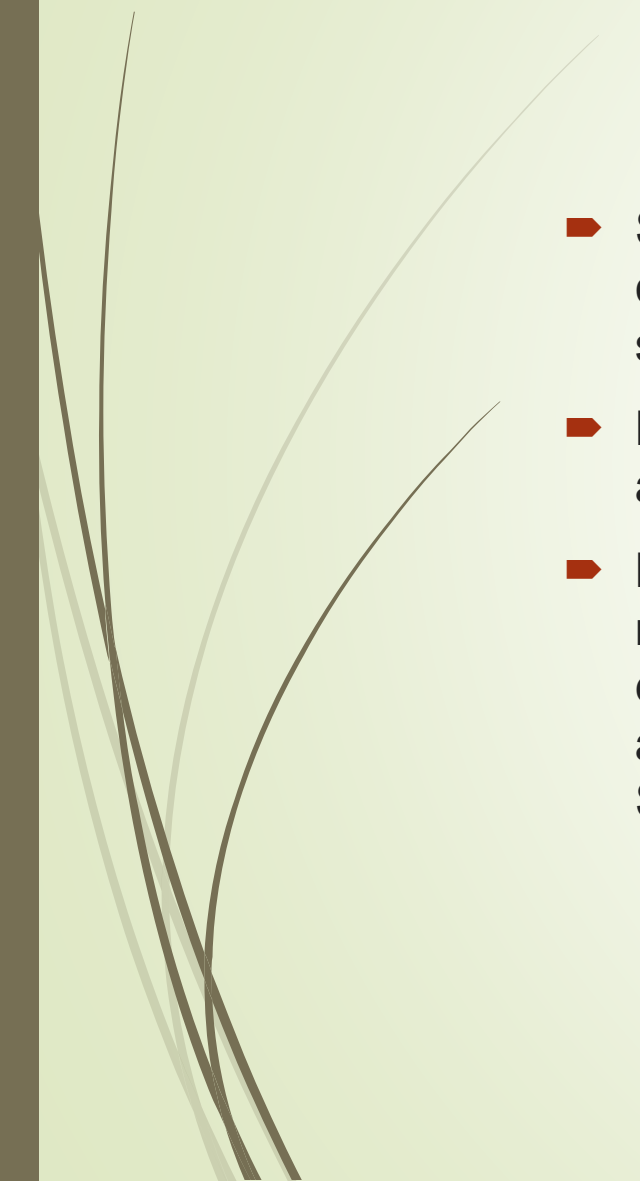
- Vendor implementations of J2EE platforms often employ numerous service agents to perform a variety of runtime filtering, processing, and routing tasks. A common example is the use of service agents to process SOAP headers.
- To support SOAP header processing, the JAX-RPC API allows for the creation of specialized service agents called handlers (Figure 18.17) runtime filters that exist as extensions to the J2EE container environments. Handlers can process SOAP header blocks for messages sent by J2EE service requestors or for messages received by EJB Endpoints and JAX-RPC Service Endpoints
- Multiple handlers can be used to process different header blocks in the same SOAP message. In this case the handlers are chained in a predetermined sequence (appropriately called a handler chain).

J2EE Service Agents





Primitive SOA support

- Service encapsulation - The distributed nature of the J2EE platform allows for the creation of independent units of processing logic through Enterprise Java Beans or servlets.
 - Loose coupling - The use of interfaces within the J2EE platform allows for the abstraction of metadata from a component's actual logic.
 - Messaging - Prior to the acceptance of Web services, the J2EE platform supported messaging via the JMS (Java Message Service) standard, allowing for the exchange of messages between both servlets and EJB components. With the arrival of Web services support, the JAX-RPC API provides the means of enabling SOAP messaging over HTTP.
- 



Thank You...