UDEM

Unidad 3. Elementos Básicos de EAI

Integración de Aplicaciones Computacionales

Dr. Irving Cruz

Departamento de Ciencias Computacionales - Universidad de Monterrey (UDEM)

Contenido Unidad 3

1. Elementos Básicos
   - Elementos Básicos de EAI
2. Modelo de comunicaciones
   - Modelo de comunicaciones
   - Comunicación síncrona
   - Comunicación asíncrona
3. Métodos de Integración
   - Métodos de Integración
   - Integración de mensajería
   - Definiciones de Interfaz
   - Conectores
4. Middleware
   - Middleware
   - Middleware de acceso a bases de datos
   - Middleware orientado a mensajes (MOM)
   - Tecnología de objetos distribuidos
   - Monitores de procesamiento de transacciones
5. Servicios
   - Elementos Básicos de Servicio

Repaso

- ¿Qué es un modelo de integración?
- ¿Que es el acoplamiento?
- ¿Que es la integración de caja negra y caja blanca?
- ¿Cuáles son los modelos de integración?
  1. Integración de presentación.
  2. Integración de datos.
  3. Integración funcional.
- ¿En que se diferencían?
- ¿Cuáles son los enfoques de la integración funcional?
  1. Consistencia de datos.
  2. Procesos de múltiples pasos.
  3. Componentes plug-and-play.

Elementos básicos de EAI

- La implementación de EAI en una empresa requiere tanto de la metodología y como de la tecnología.
- La metodología es la combinación de las definiciones, procesos y criterios que proporcionan una estructura para organizar y ejecutar el desarrollo de una solución.
- La tecnología se utiliza para poner en práctica las soluciones.
- La arquitectura EAI es una combinación de tecnologías reunidas estructuradamente y se basa en cuatro elementos básicos fundamentales:
  - Modelo de comunicaciones.
  - Método de integración.
  - Middleware.
  - Servicios.

Modelo de comunicaciones

- La manera en que los sistemas interactúan es fundamental para para su flexibilidad.
- Dos opciones básicas para el modelo de comunicación: síncrona y asíncrona.
  - La comunicación síncrona requiere que el emisor de una solicitud espere hasta recibir una respuesta del receptor antes de continuar con el proceso.
  - La comunicación asíncrona permite al emisor continuar el proceso después de que se envía la petición o contenido.
- En determinado momento de la integración, la comunicación sincrónica puede ser obligada a actuar de forma asíncrona y viceversa.

Modelo de comunicaciones...

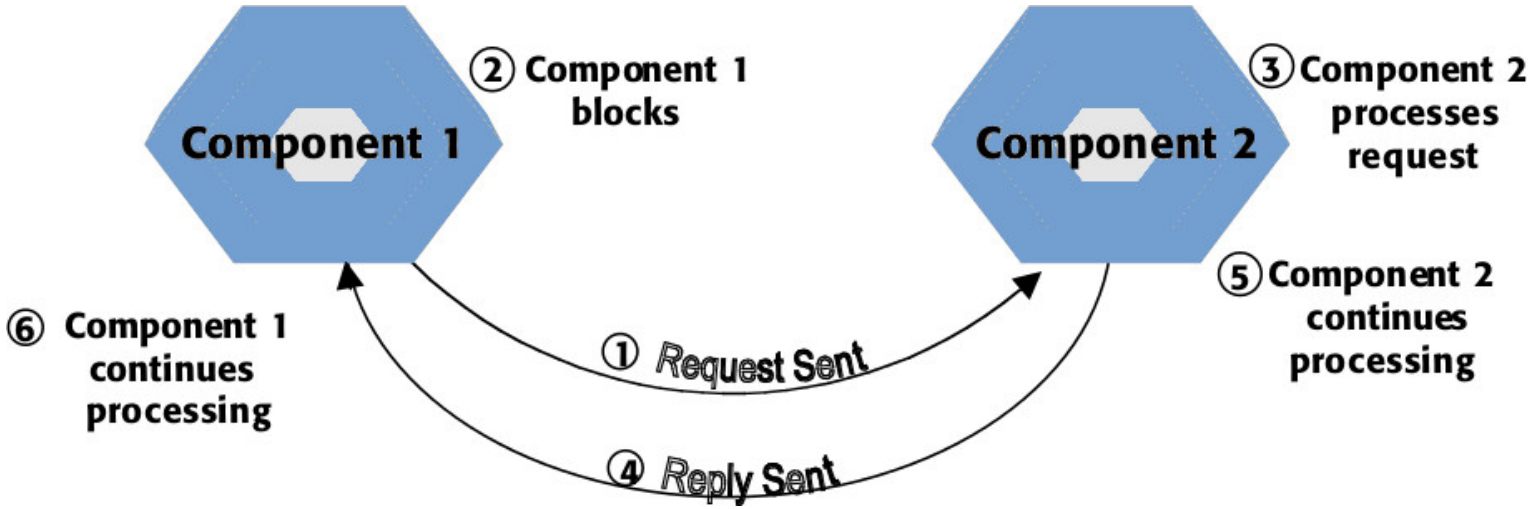
- **Receptor**. Un receptor es un software que recibe una solicitud de un emisor. El receptor puede enviar una respuesta al emisor como resultado de la solicitud.
- **Emisor** Un emisor es un software que envía una petición a otro componente de software.
- **Solicitud**. Una solicitud es un conjunto de acciones y datos enviados desde un módulo de envío a un módulo receptor.
- **Respuesta**. Una respuesta es un conjunto de datos y acciones posiblemente asociadas que son enviadas como como resultado de una solicitud.

Comunicación síncrona

Se produce cuando la comunicación entre un emisor y el receptor se lleva a cabo de manera coordinada. Esto requiere que el emisor y el receptor operen dependiendo del procesamiento de la solicitud.

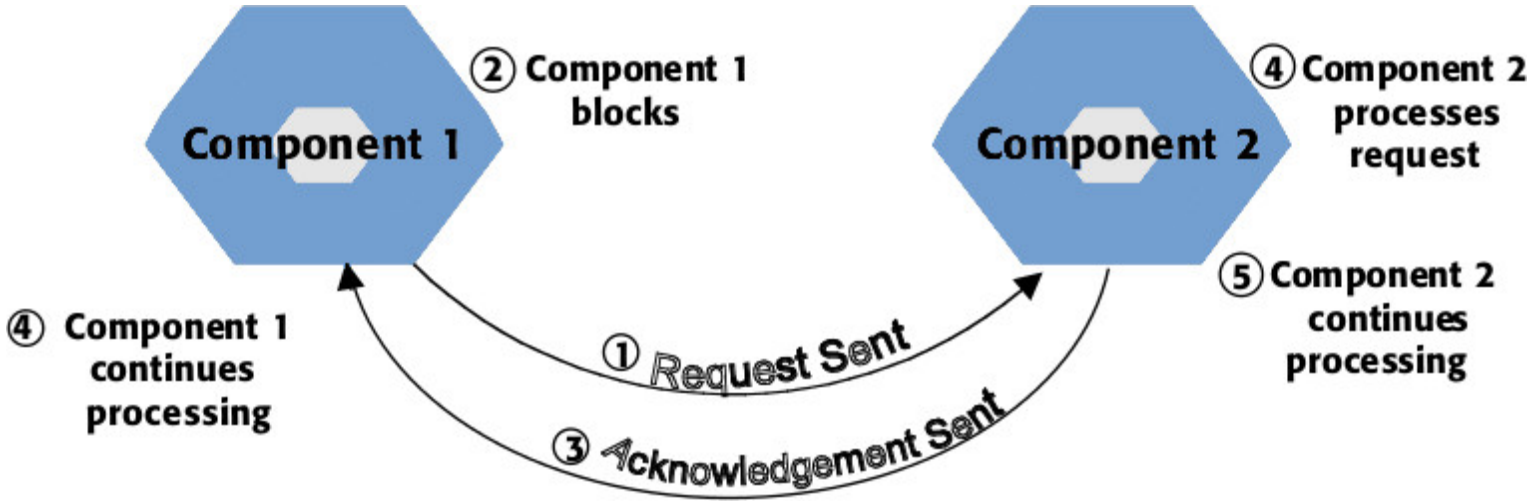
- Los sistemas interactivos requieren comunicación síncrona.
- Se requiere una infraestructura de red fiable para evitar que el emisor quede en modo de espera si la solicitud pierde durante el envío.
- Tres tipos populares de comunicación síncrona:
  - Solicitud / respuesta.
  - Unidireccional.
  - Sondeo.

Solicitud / respuesta



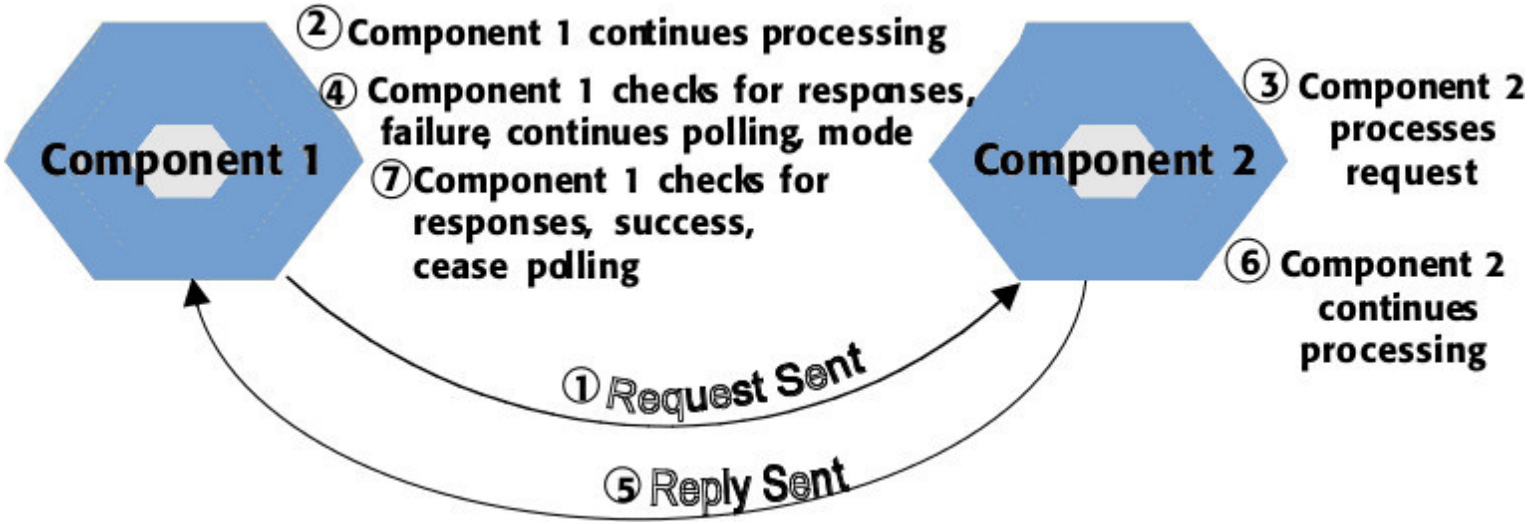
- Es el tipo básico para la comunicación síncrona, donde un emisor realiza una solicitud de un receptor y espera una respuesta antes de continuar con el proceso.
- Si el procesamiento del receptor toma mucho tiempo, el emisor no puede continuar con el proceso y el rendimiento se vuelve inaceptable.

Unidireccional



- Comunicación síncrona donde un emisor hace una petición a un receptor y espera una respuesta que confirma la recepción de la solicitud.
- Un inconveniente de este enfoque es que se depende del acuse de recibo.
- Otro inconveniente está en el rendimiento. La recepción del acuse debe ser rápida y no esperar a que se complete el proceso.

Sondeo



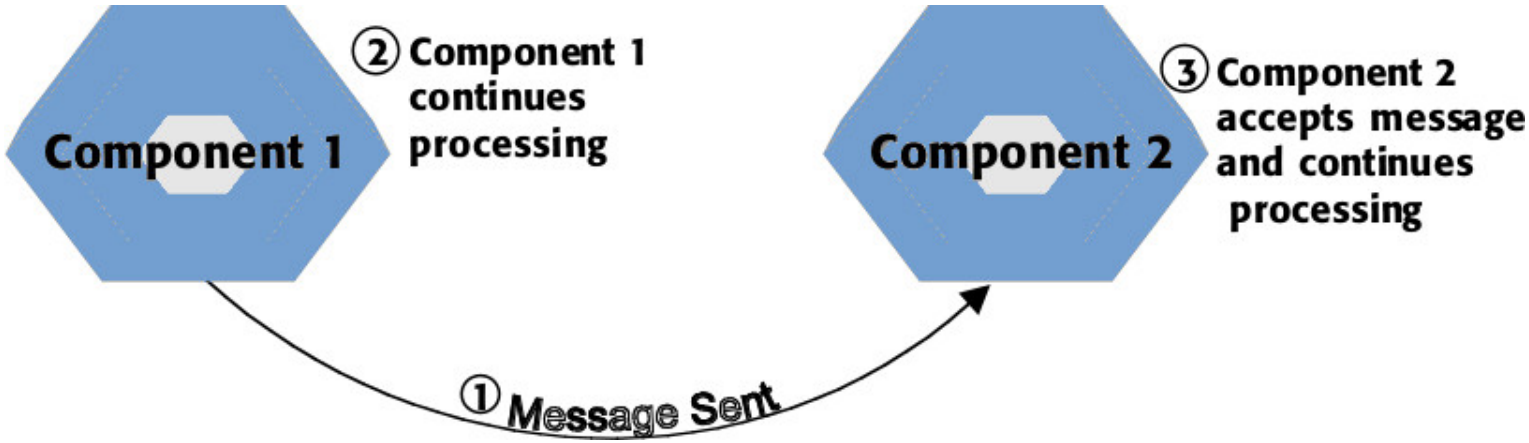
- Un emisor comunica una solicitud a un receptor pero en lugar de bloquearse, sigue con su proceso. En intervalos definidos, el emisor comprueba si alguna respuesta ha sido enviada. Cuando se detecta una respuesta, procesa y detiene los futuros sondeos de respuesta.
- Se utiliza en situaciones en las que el emisor necesita una respuesta, pero que se puede continuar el proceso sin la respuesta.
- Mejora el rendimiento, pero es más compleja de implementar debido a la necesidad del sondeo.

Comunicación asíncrona

Se produce cuando la comunicación entre un emisor y el receptor se lleva a cabo de una manera que permite a cada uno operar independientemente. El receptor no tiene ninguna obligación de manejar la comunicación o responder al emisor. El emisor continua una vez que envía la solicitud sin tener en cuenta como continua el receptor.

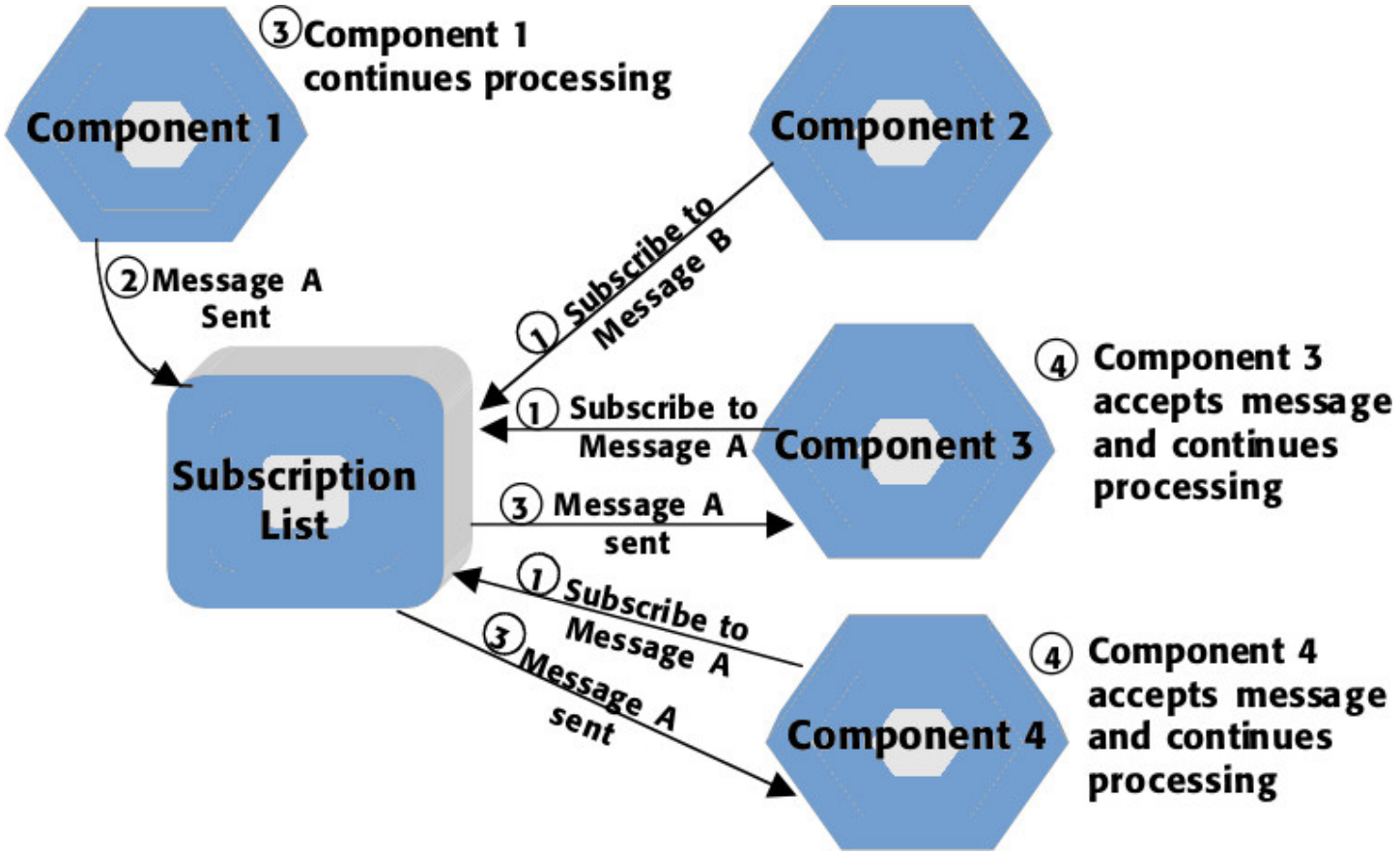
- Se utiliza cuando se requiere la comunicación de la información sin la necesidad de coordinar las actividades o respuestas.
- Tres tipos populares de comunicación asíncrona:
  - Paso de mensajes
  - Publicación / suscripción
  - Emisión

Paso de mensajes



- Se envía una solicitud desde un emisor a un receptor. Una vez que el emisor ha hecho la solicitud, esencialmente se olvida que la ha enviado y sigue procesando. La solicitud se entrega al receptor y se procesa.
- Esta comunicación debe implementarse en una red fiable o con un servicio de entrega garantizada
- Su principal inconveniente es que sin el servicio de entrega garantizada, una solicitud se puede perder debido a un fallo.

Publicación / Suscripción

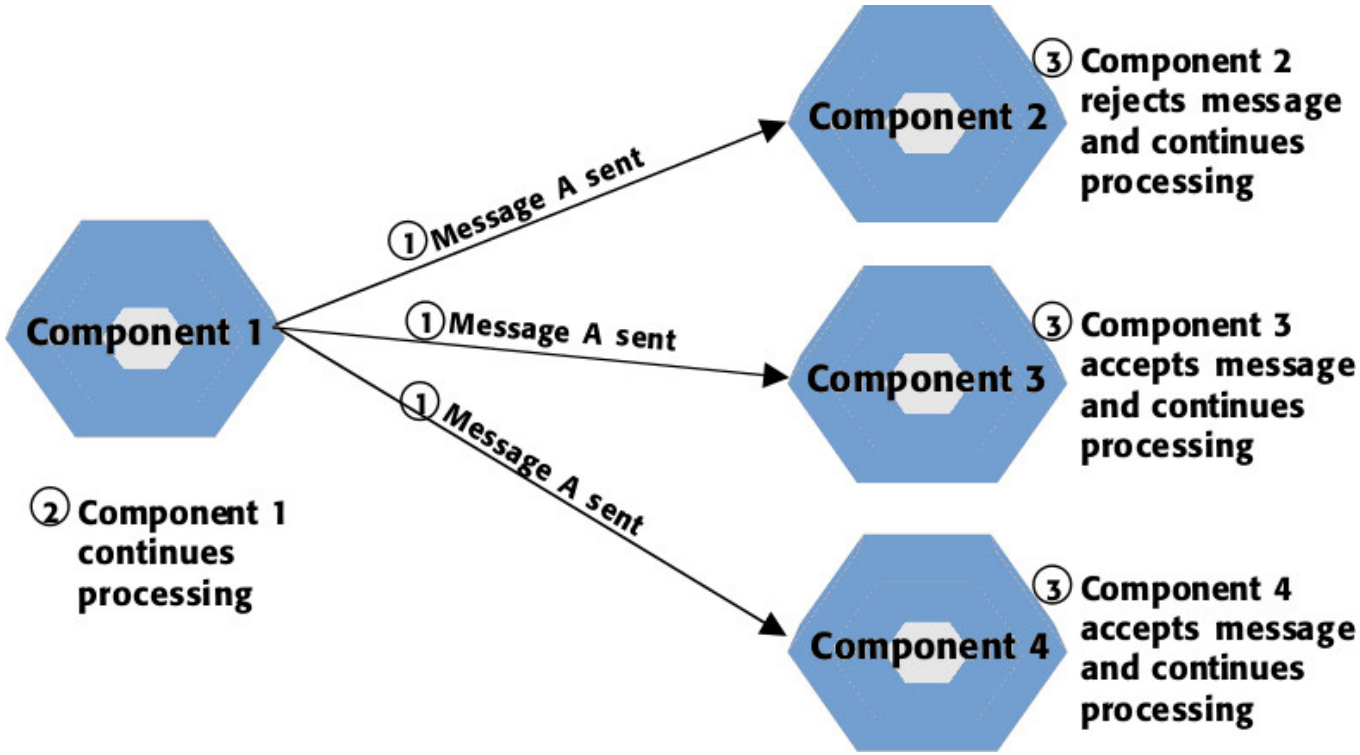


- Una solicitud es enviada por el emisor y el receptor se determina mediante una declaración de interés por el receptor de la solicitud.

Publicación / Suscripción ...

- Utilizado cuando la respuesta no es necesaria y el receptor se determina por el contenido de la solicitud.
- Es el desarrollador de la aplicación receptora quien define los intereses de la aplicación.
- Útil en un sistema de integración de procesos de múltiples pasos donde la solicitud es en realidad la notificación de que ha ocurrido un evento.
- El principal inconveniente está en la complejidad del diseño en determinar qué receptores reciben una solicitud y cuáles no.

Emisión



- Se envía una solicitud a todos los receptores de una red. Cada participante determina si la petición es de su interés examinando los contenidos.
- Se debe tener cuidado para no crear cuellos de botella de rendimiento, ya que cada posible receptor debe buscar en cada mensaje de difusión.

Métodos de Integración

- Un método de integración es el enfoque utilizado para construir una solicitud de un emisor a un receptor. Existen dos métodos principales de integración:
  - Mensajería.
  - Definiciones de interfaz.
- Ambos requieren un **conector** para crear la conexión en la aplicación a través de la cual se transmiten las solicitudes.
- Un conector es una interfaz dentro de la aplicación que define las peticiones que aceptará el receptor y que oculta la complejidad de la integración.
- Una arquitectura EAI robusta a menudo requiere la existencia de ambos modelos de comunicación y muchas veces de los dos métodos de integración.

Integración de mensajería

- En esta integración, el emisor construye un mensaje con información de las acciones a realizar (información de control), y los datos para realizar las acciones.
  - Ejemplos: Mensaje de movimientos en una cuenta bancaria, jugada en un juego de ajedrez.
- Esta integración ayuda en la construcción, uso y tratamiento de los datos. El acoplamiento de la información sobre datos y control ayuda a reducir el acoplamiento entre aplicaciones.
- El mensaje debe ser codificado y descodificado de la misma manera por todos los emisores y receptores para evitar malas interpretaciones y causar inconsistencias.
- Una buena arquitectura de EAI debe proporcionar las herramientas de diseño necesarias para definir los mensajes.

Integración de mensajería ...

- Cualquier aplicación que envíe mensajes debe ser capaz de:
  - Crear el mensaje en el formato apropiado.
  - Colocar el mensaje en el sistema de comunicaciones.
  - Recibir el mensaje del sistema de comunicaciones.
  - Analizar y separar la información de control y datos del mensaje.
  - Determinar qué hacer con el mensaje.
- Un problema con la mensajería es que no siempre es visible qué aplicaciones pueden responder a qué mensajes.
- Una buena documentación ayuda a determinar estos vínculos.

Definiciones de interfaz

- En este enfoque , el emisor se comunica a través de una interfaz que define las acciones que pueden ser invocadas por una aplicación.
  - Cualquier dato que se procese se envía a través de la interfaz.
- Diferencias con la integración de mensajería:
  - La interfaz está asociada con una aplicación y los mensajes no.
  - En las interfaces las acciones son fáciles de leer y están indicadas explícitamente. Los mensajes esconden las aplicaciones que los utilizan.
  - Las interfaces requieren menos procesamiento que los mensajes para decodificar datos, por lo que los errores se detectan antes.
  - La mensajería genera un menor grado de acoplamiento que las interfaces, pero es más propensa a errores y no permite el reuso de soluciones.

Definiciones de interfaz...

- Las interfaces hacen que una aplicación se asemeje a un procedimiento o un objeto en un lenguaje de programación.
- Proceso para el uso de interfaces:
  1. Crear una invocación o llamada al componente.
  2. Ejecutar la acción basada en la interfaz que se está utilizando.
- El receptor debe realizar lo siguiente:
  - Recibir una invocación o llamada remota.
  - Ejecutar la acción basada en la interfaz que está utilizando.
- Si las definiciones de interfaces son aplicadas correctamente, las aplicación verían las peticiones como solicitudes internas en vez de peticiones provenientes de sistemas externos.

Ventajas y desventajas de las interfaces

- Ventajas de la integración basada en interfaces:
  - A largo plazo, la integración basada en interfaces es fácil de reutilizar y mantener, ya que se definen de manera explícita y visible para el desarrollador.
  - No se requiere del código para saber si una aplicación responde a una solicitud.
  - Las interfaces se auto-describen en términos de las acciones que pueden llevar a cabo.
- Desventajas:
  - Pueden ser más difíciles de modificar y ampliar dependiendo de la implementación.
  - Para que los cambios en la interfaz surtan efecto, se requiere de una compilación de todas las aplicaciones involucradas.
  - Las interfaces requieren de mayor conocimiento para una correcta definición y se puedan utilizar como plug-and-play.

Conector

Es la lógica programada en una aplicación cuyo objetivo es proporcionar acceso a la presentación, datos, o funcionalidad de la aplicación de una manera estructurada. Oculta la complejidad de traducción y comunicación de un mensaje o llamada a una interfaz.

- Un conector es más que una simple interfaz. Proporciona ventajas adicionales, por ejemplo:
  - Manejo de errores y comprobación de validación.
  - *Marshalling* (serialización) y *unmarshalling* de datos del mensaje u objeto.
  - Conversión y transformación de los datos en el formato de la aplicación receptora.
  - Gestión de información del estado de la comunicación para proporcionar una entrega garantizada.

Conector ...

Marshalling

Proceso de convertir secuencias de parámetros, estructuras de datos u objetos en cadenas de bytes para ser transmitidas a través de un puerto de comunicación. *Unmarshalling* es el proceso de restaurar correctamente las estructuras originales en la aplicación receptora.

- Cuando las aplicaciones no tienen conectores, se pueden utilizar archivos, bases de datos, interfaces de usuario, o memoria como punto de entrada de una solicitud.
- Se debe primero seleccionar un modelo adecuado de integración para construir el conector correcto.
- Es importante asegurar que una solicitud es válida y está en el formato correcto antes de ser enviada.

Tipos de middleware

Middleware

Software que facilita la comunicación de las solicitudes entre los componentes de software mediante el uso de interfaces o mensajes definidos. Proporciona además, el entorno de ejecución para administrar las solicitudes entre los componentes de software.

- Para la EAI, el middleware es la tecnología que permite que sistemas basados en diferentes tecnologías estén interconectadas.
- Existen básicamente cinco tipos de middleware:
  1. Llamadas a procedimientos remotos.
  2. Middleware de acceso a bases de datos.
  3. Middleware orientado a mensajes.
  4. Tecnología de objetos distribuidos.
  5. Monitores de procesamiento de transacciones.

Llamadas a procedimientos remotos (RPC)

Middleware que se basa en la noción de desarrollar aplicaciones distribuidas que se integran en el nivel de procedimiento. Tienen la capacidad de hacer llamadas de procedimiento a través de una red.

- RPC existe desde 1970, en 1980 alcanzó su punto máximo, pero disminuyó con el surgimiento del uso de la tecnología OO.
- Utilizado en el área de definiciones de interfaz, utilizada por la mayor parte de la tecnología de middleware de objetos distribuidos (CORBA, COM/DCOM).
- RPC aún se utiliza pero sólo en actividades de desarrollo que utilizan lenguajes procedurales como C, que no requieren integración heredada.

Middleware de acceso a bases de datos

Tipo de middleware que se basa en la noción de acceso a los datos distribuidos ya sea en archivos o bases de datos. Se integra en el nivel de datos y permite las consultas o transmisión de datos que se produzcan a través de una red.

- Desarrollado en la época cliente/servidor a finales de los 80's. Cada proveedor de BD desarrolló su propio middleware.
- SQL era un lenguaje estándar para consultas, pero el acceso a la BD no era estándar.
- El middleware de acceso a BD proporciona las bases para la integración de datos. Es decir, permite acceder directamente a los datos sin pasar por la presentación y la funcionalidad de una aplicación.

Middleware de acceso a bases de datos ...

- Los mecanismos estándar como la tecnología ODBC y JDBC proporcionan mecanismos estándar para el acceso a BD.

Open Database Connectivity (ODBC)

Significa conectividad abierta de bases de datos. Es una interfaz estándar, originalmente destinado para sistemas gestores de bases de datos relacionales. Se ha aplicado a otras fuentes de datos.

- ODBC es un estándar y al igual que SQL compite con soluciones propietarias, como las interfaces de Oracle.
- El middleware de acceso a BD es un complemento de la EAI, pero no es el núcleo de su arquitectura.

Middleware orientado a mensajes

MOM es un middleware que utiliza mensajes como método de integración; proporciona la capacidad de crear, manipular, almacenar y comunicar estos mensajes.

- Se hizo popular en los 90's cuando surgió la necesidad de integrar aplicaciones antiguas en mainframes con aplicaciones nuevas.
- MOM es adecuado para problemas de consistencia de datos y proceso de múltiples pasos, pero no en la integración de componentes.
- La integración de componentes implica reuso y plug-and-play, y los mensajes no son visibles como interfaces para los desarrolladores de aplicaciones.

Tecnología de objetos distribuidos (DOT)

DOT es un tipo de middleware que se extiende los conceptos de la tecnología orientada a objetos para el procesamiento distribuido. Se desarrollan interfaces OO para que las aplicaciones actúen como objetos y sean accesibles desde cualquier otra aplicación.

- Los lenguajes de definición de interfaz y la comunicación síncrona son elementos importantes de una solución DOT.
  - El lenguaje de definición de interfaz permite la creación de un objeto distribuido, mientras que la comunicación síncrona permite la transmisión de invocaciones a las interfaces.
- DOT es adecuado para la creación de sistemas basados en componentes.
- La desventaja de la tecnología DOT frente a MOM es su complejidad, ya que requiere un mayor grado de acoplamiento entre aplicaciones.

Monitores de procesamiento de transacciones (TPMs)

Los TPM's son un tipo de middleware que conserva la integridad de una transacción. Soportan características como reversión (roll-back), tolerancia a fallos (failover), auto-reinicio, registro de errores, y replicación para eliminar puntos simples de fallo.

- Los TPMs garantizan que una transacción mantenga las propiedades ACID a lo largo de su vida.
- Las propiedades ACID son propiedades deseables del software en el proceso de transacciones:
  - Atomicidad.
  - Coherencia.
  - alslamiento
  - Durabilidad.

ACID

- **Atomicidad**. Una transacción debe ser una unidad atómica de trabajo. Se ejecutan todas o ninguna de sus operaciones.
- **Coherencia**. Cuando finaliza, una transacción debe dejar todos los datos en un estado coherente.
- **Aislamiento**. Las modificaciones realizadas por transacciones simultáneas se deben aislar de las modificaciones llevadas a cabo por otras transacciones simultáneas. Una transacción reconoce los datos en el estado en que estaban antes o después de otra transacción, pero no reconoce un estado intermedio.
- **Durabilidad** Una vez concluida una transacción, sus efectos son permanentes en el sistema. Las modificaciones persisten aún en el caso de producirse un error del sistema.

Monitores de procesamiento de transacciones (TPMs)...

- TPMs permiten que una transacción se cree desde el remitente y aseguran que se coloque en el lugar adecuado, en el momento adecuado, y se complete en el orden correcto.
- TPMs es el más complejo de todos los tipos de middleware.
- Pocas herramientas de EAI utilizan TPMs como su tecnología base. La funcionalidad de los TPMs se han integrado en las tecnología MOM y DOT.
- Una arquitectura robusta de EAI requiere que las tecnologías MOM, DOT, y TPM estén integradas en una estructura consistente.
- No existe un producto de EAI que cubra todas las necesidades de una empresa, por tanto se necesita del middleware para conseguir una arquitectura robusta de EAI.

Elementos Básicos de Servicio

Un servicio es una extensión funcional para la comunicación básica o capacidad de middleware.

- El modelo de comunicación, el método de integración y el middleware son el núcleo de cualquier solución EAI y los servicios son el elemento final.
- Los servicios pretenden reducir el trabajo de la aplicación de la tecnología. También ayudan a la seguridad y la fiabilidad.
- Existe una amplia gama de posibles servicios.

Servicios

- Algunos de los servicios más importantes para una EAI exitosa:
  - **Directorio**. Rastrea todos los componentes y los datos fundamentales sobre el sistema.
  - **Ciclo de vida**. Automatiza la creación de objetos o mensajes y garantiza que se usen adecuadamente y sean eliminados cuando terminen.
  - **Seguridad**. Proporciona todas las capacidades necesarias para asegurar cualquier integración, específicamente de autenticación, autorización y comunicaciones seguras. Es complejo.
  - **Conversión y transformación**. Convierte y transforma los datos en el formato correcto para completar adecuadamente cualquier integración.

Servicios...

- - **Persistencia**. Asegura que la información no se pierde. Proporciona la capacidad de guardar la información de estado y los datos.
  - **Eventos**. Proporciona la capacidad de identificar y realizar un seguimiento de eventos. Identifica cuando se produce un problema en particular o un evento único.
  - **Notificación**. Cuando se detecta un evento, el servicio de notificación alerta al componente interesado que se ha producido el evento.
  - **Flujo de trabajo**. Gestiona un conjunto de peticiones o mensajes a través de una serie de componentes en un orden prescrito como una sola acción.
- Algunos servicios casi siempre se requieren como el directorio y ciclo de vida. El uso de los otros se basa en el uso de la arquitectura EAI.

¿ Preguntas