

Arquitectura de Software

Conectores

Contenidos

- Introducción
- Ejemplo
- Categorías de conectores
- Tipos de conectores
- Ejercicios

Introducción

- Componentes
 - Diseño de la *funcionalidad* y la gestión de los datos
- Conectores
 - Integración de los componentes
 - Interacción de los componentes
 - *Transferencia de control y datos* entre los componentes

Introducción

- Además los conectores pueden
 - Proporcionar servicios de
 - Persistencia, mensajería, transacciones, control de concurrencia, ...
 - Se les conoce como *facilities components*
 - A veces los vemos/representamos como componentes (por su complejidad) pero son conectores en nuestro sistema
 - Los componentes se centran en la funcionalidad específica de nuestro sistema


Introducción

- Ejemplo
 - Distinguir componentes y conectores en un sistema de gestión de una clínica
 - Requisitos funcionales (*el qué?*): altas y bajas de pacientes, médicos, ...
 - *El cómo?*: con algún mecanismo de persistencia concreto (no es un requisito funcional)
- Otro contexto en el que aparecen conectores
 - Integración de componentes heterogéneos

Introducción

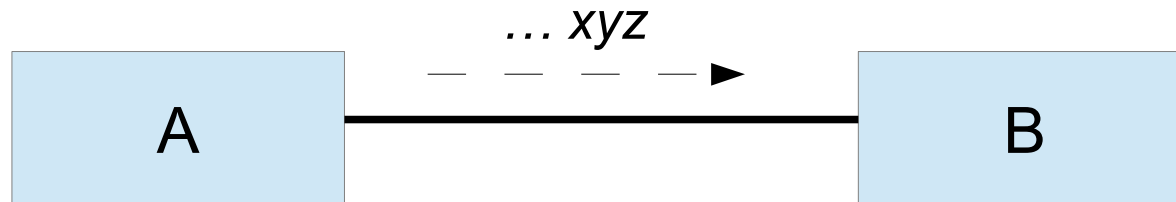
- Es responsabilidad del arquitecto de software:
 - Entender para cada componente sus necesidades de interacción
 - Identificar para cada una de esas interacciones sus atributos relevantes
 - A la vista de lo anterior seleccionar los conectores candidatos
 - Asesorar los *trade-offs* de cada candidato
 - **Analizar las propiedades clave de la interacción entre el componente y el conector**

Introducción

Los conectores nos permiten entender mucho sobre cómo lleva a cabo nuestro sistema sus tareas pero sin necesariamente saber qué hace el sistema 

- Conclusión: los conectores
 - Son elementos arquitecturales independientes de la aplicación que desarrollamos
 - Soportan dos principios básicos: abstracción y *separation of concerns*

Ejemplo



- Los componentes A y B se comunican a través de una “pipe” de Unix
 - Nos falta información concreta de esta “pipe”
 - Acompañaremos esto con una documentación particular de esta “pipe”

Ejemplo

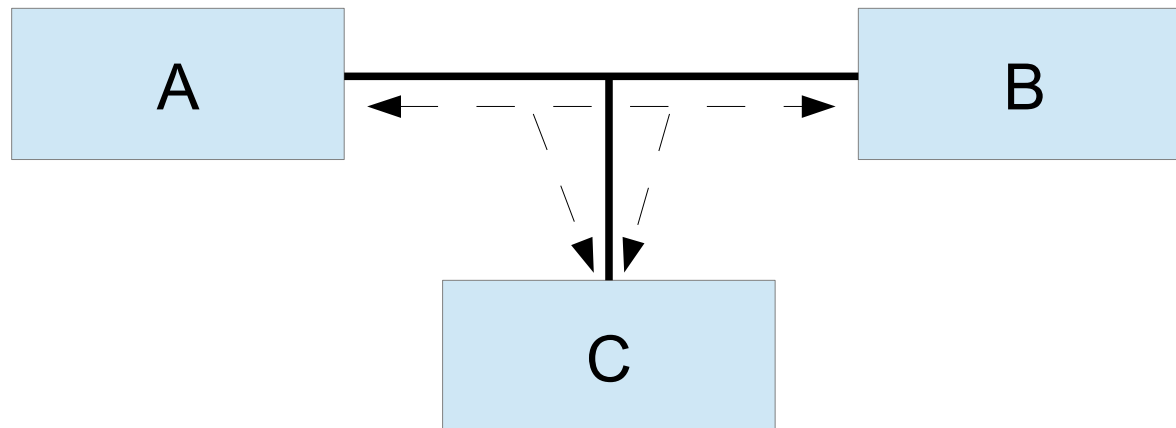
- Documentación general del comportamiento de una “pipe”
 - Pasa *streams* de datos sin formato
 - Es un canal unidireccional
 - Tiene un único emisor y un único receptor
 - No almacena datos
 - Sólo envía los datos una vez, si no se reciben se pierden

Ejemplo

- Supongamos que queremos modificar la interacción entre A y B
 - B también envía información a A (*ack* de los datos)
 - Asegurar la recepción de la información: la “pipe” hace reintentos hasta que lo consigue
- Solución
 - Primero: necesitamos otra “pipe”
 - Segundo: necesitamos implementar un buffer de datos

Ejemplo

- Queremos pasar datos con tipos
 - Necesitamos otro tipo de conector q p.e. bus de datos



Categorías de conectores

- Describen el tipo de servicio que pueden proporcionar
 - Tipos de interacción entre los componentes
- Un conector puede pertenecer a más de una categoría
- Cuatro categorías:
 - Comunicación
 - Transfieren información entre los componentes

Categorías de conectores

- **Coordinación**
 - Dan soporte a la transferencia de control
- **Conversión**
 - Permiten que componentes que no han sido diseñados para interactuar puedan hacerlo
 - Posibilitan la interacción entre componentes heterogéneos
 - Ejemplos: Conversión entre formatos de datos. *Wrappers* para componentes legados
- **Facilitación**
 - Proveen servicios para facilitar y optimizar la interacción entre los componentes
 - Ejemplos: Balanceo de carga. Control de la concurrencia. Planifican servicios

Tipos de conectores

- Con las categorías no tenemos información suficiente para
 - Modelar y analizar los conectores en nuestras arquitecturas
 - Crear nuevos conectores
- Las características que vamos a presentar son “plantillas” para que el arquitecto seleccione el conector más adecuado
 - Un conector concreto no tiene que poseer toda la riqueza que aquí presentamos

Tipos de conectores

- Distinguiamos ocho tipos de conectores *simples*
 - Llamada a procedimiento
 - Evento
 - Acceso a datos
 - *Linkage*
 - *Stream*
 - Mediador
 - Adaptador
 - Distribuidor
- A partir de ellos podemos construir conectores *compuestos*, más complejos

Llamada a procedimiento

- Es el más común
- Comunicación
 - Parámetros y valores de retorno
- Coordinación
 - Diferentes técnicas de invocación
- Ejemplos
 - Métodos de los objetos en POO. Fork y exec en Unix. Llamada al SO
 - En conectores compuestos: RPC (llamada a procedimiento y facilitación)

Llamada a procedimiento

Servicio	Tipo	Dimensión	Subdimensión	Valor
<ul style="list-style-type: none"> - Comunicación - Coordinación 	Llamada procedim.	Parámetros	<ul style="list-style-type: none"> Transferencia Semántica Valores retorno Reg. Invoca. 	<ul style="list-style-type: none"> - Referencia - Valor - Nombre - Valores defecto - Param clave - Param inline - Push LR - Push RL - Hash
		Punto de entrada	<ul style="list-style-type: none"> - Múltiple - Unico 	
		Invocación	<ul style="list-style-type: none"> Explícita Implícita 	<ul style="list-style-type: none"> - Método - Macro - Inline - System call - Excepciones - Callback - Delegación
		Sincronismo		<ul style="list-style-type: none"> - Asíncrono - Síncrono
		Cardinalidad	<ul style="list-style-type: none"> - Fan out - Fan in 	
		Accesibilidad		<ul style="list-style-type: none"> - Privada - Protegida - Pública

Conectores (17)

Arquitectura de Software

Rubén Béjar y José Merseguer



Universidad
Zaragoza

Llamada a procedimiento

- Ejercicio: Llamada a método en java. Estudiar los valores específicos

Eventos

- Coordinación
 - El flujo se desencadena por la ocurrencia de un evento
 - Cuando el conector “percibe” el evento genera mensajes (notificaciones) para los componentes implicados y les pasa el flujo de control
 - Comunicación asíncrona en sistemas distribuidos
- Comunicación
 - El evento puede llevar información: tiempo y lugar de ocurrencia, datos

Eventos

- Diferencia con llamada a procedimiento
 - Se forman conectores virtuales entre componentes interesados en el mismo evento. Estos conectores aparecen y desaparecen dependiendo del interés de los componentes
- Ejemplo: Aplicaciones con GUI

Eventos

Servicio	Tipo	Dimensión	Subdimensión	Valor
<ul style="list-style-type: none"> - Comunicación - Coordinación 	Evento	Cardinalidad	<ul style="list-style-type: none"> - Productores - Observadores - Patrones de eventos 	<ul style="list-style-type: none"> - Best effort - Exacto uno - Max. Uno - Min. uno
		Entrega		<ul style="list-style-type: none"> - Sincrono - Asincrono - TO sincrono
		Prioridad	<ul style="list-style-type: none"> - Salida - Entrada 	<ul style="list-style-type: none"> - Polled - P/S - Central update - Queue dispatch
		Sincronismo		<ul style="list-style-type: none"> - Absoluta - Relativa
		Notificación		<ul style="list-style-type: none"> - Page faults - Interrupciones - Traps
		Causalidad		<ul style="list-style-type: none"> - Señales - GUI-I/O - Triggers
		Modo	<ul style="list-style-type: none"> - Hardware - Software 	

Acceso a datos

- Comunicación
 - Permiten que los componentes accedan a fuentes de datos mantenidas por un componente
 - Los datos pueden estar almacenados de manera persistente o temporal
- Conversión
 - Pueden realizar conversión de formatos si difieren el formato almacenado del formato solicitado
- Ejemplo: Persistente: acceso SQL a BBDD.
Temporal: acceso a la pila de memoria o cacheo de

Acceso a datos


Servicio	Tipo	Dimensión	Subdimensión	Valor
<ul style="list-style-type: none"> - Comunicación - Conversión 	Acceso datos	Localidad		<ul style="list-style-type: none"> - Thread - Proceso - Global
		Acceso		<ul style="list-style-type: none"> - Accesor - Mutator
		Disponibilidad	Transitoria	<ul style="list-style-type: none"> - Registro - Cache - DMA - Heap - Stack
			Persistente	<ul style="list-style-type: none"> - Acceso repositorio - Fichero I/O - Intercambio din. - Acceso BBDD
		Accesibilidad		<ul style="list-style-type: none"> - Privado - Púb. - Protegido
		Ciclo de vida		<ul style="list-style-type: none"> - Inicialización - Terminación
Conectores (23)		Cardinalidad		<ul style="list-style-type: none"> - Defines - Uses

Stream

- Comunicación
 - Transferir grandes cantidades de información entre procesos autónomos
- Se combinan con los de
 - Acceso a datos: acceso a ficheros y bases de datos. Ejemplo: JDBC, ODBC
 - Eventos: para multiplexar gran cantidad de eventos
- Ejemplos: sockets, pipes Unix

Stream

Servicio	Tipo	Dimensión	Subdimensi ón	Valor
Comunicación Stream		Entrega		<ul style="list-style-type: none">- Best effort- Exacto uno- Max. Uno- Min. uno
		Bounds		<ul style="list-style-type: none">- Acotado- No acotado
		Buffering		<ul style="list-style-type: none">- Almacenado- No almacenado
		Throughput		<ul style="list-style-type: none">- Atómico- Alto orden
		Estado		<ul style="list-style-type: none">- Con estado- Sin estado
		Identidad		<ul style="list-style-type: none">- Nombrado- Sin nombrar
		Localidad		<ul style="list-style-type: none">- Local- Remoto
		Sincronismo		<ul style="list-style-type: none">- Sincrono- Asíncrono- TO sincrono
		Formato		<ul style="list-style-type: none">- Plano- Estructurado
		Cardinalidad		<ul style="list-style-type: none">- Binaria- N-aria
Conectores (25) Arquitectura de Software Rubén Béjar y José Merseguer				<ul style="list-style-type: none">- Multi-emisor- Multi-receptor- Multi E/R



Universidad
Zaragoza

Linkage

- Facilitación
 - Unir componentes durante su interacción
 - Habilitan “canales” para comunicación y coordinación que son utilizados por otros conectores
 - Cuando el “canal” se ha establecido pueden desaparecer
- Granularidad: nivel de detalle para establecer la unión
 - Unidad: especifica que un componente depende de otro. Ejemplo: Make
 - Sintáctica: relaciones entre variables, procedimientos, funciones, etc. dentro de los componentes unidos

Linkage

- Semántica: cómo deben interaccionar los componentes unidos
- Cardinalidad: número de lugares en los que el componente, procedimiento, etc, es definido, usado proporcionado o requerido

Linkage

Servicio	Tipo	Dimensión	Subdimensión	Valor
Facilitación	Linkage	Referencia		<ul style="list-style-type: none"> - Implícita - Explícita
		Granularidad	<ul style="list-style-type: none"> Unidad Sintáctica Semántica 	<ul style="list-style-type: none"> - Variables - Proc. - Funciones - Constantes - Tipos
		Cardinalidad	<ul style="list-style-type: none"> - Defines - Uses - Provides - Requires 	
		Binding		<ul style="list-style-type: none"> - Tiempo compil. - Run-time - Precompil.

Mediadores

- Coordinación y facilitación
- Ejemplo: Un sistema multithread necesita memoria compartida y control de concurrencia
- Los componentes saben que existen otros componentes pero no conocen su estado
 - Este conector organiza la ejecución del sistema, resuelve conflictos, balanceo de carga

Mediadores

Servicio	Tipo	Dimensión	Subdimensión	Valor
- Facilitación - Coordinación	Mediadores	Fault handling		- Voto - Semáforo - Rendezvous - Monitor - Lock
		Concurrencia	Mecanismo	- Ligero - Pesado
			Peso	
		Transacciones	Anidamiento	- Uno - Múltiple
			Awareness	- Nada - Soportado - Requerido - Nuevo
			Isolation	- Lect. - Escritura - RW
		Seguridad	Autenticación	- Capacidades
			Autorización	- Listas control acceso
			Privacidad	- Encrip.
			Integridad	- Screeninig
Planificación	Duración	- Check. Redund. - Certificados		
	Tiempo	- Single ses.		
Conectores (30)			Peso	- Multi ses.

Adaptadores

- Hacen que interoperen componentes en entornos heterogéneos
- Conversión
 - Proporcionan utilidades para que interactuen componentes que no han sido diseñados para ello
 - Crean las políticas de comunicación y los protocolos de interacción

Adaptadores

Servicio	Tipo	Dimensión	Subdimensión	Valor
Conversión	Adaptador	Conversión de invocación Conversión de empaquetado Conversión de protocolo Conversión de presentación	Address Mapping Marshaling Translation	Wrappers Packagers

Distribución

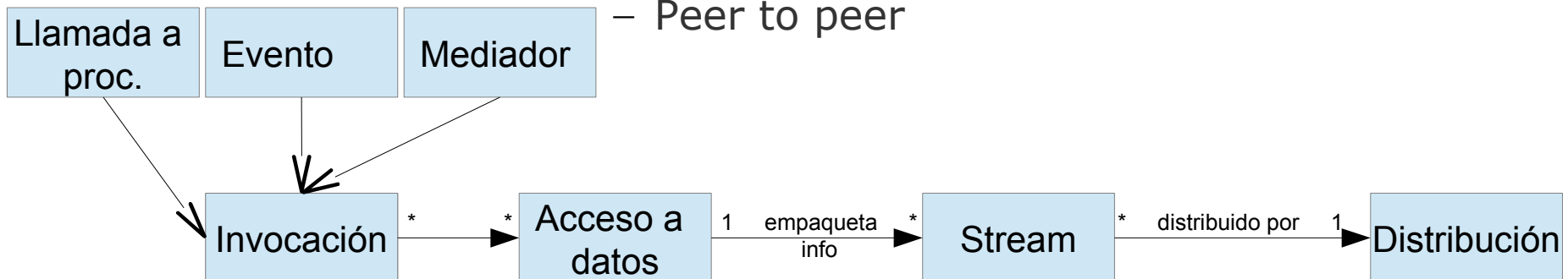
- Facilitación
 - Identifican caminos de interacción entre componentes y después establecen comunicación y coordinación
- Nunca aparecen solos: Asisten a llamada a procedimiento o a stream
- Los sistemas distribuidos intercambian información usando este conector para dirigir el flujo de datos
- Ejemplos: DNS, routing, switching, diferentes servicios de red

Distribuidores

Servicio	Tipo	Dimensión	Subdimensi ón	Valor
Facilitación	Distribuidor	Nombrado	Basado en estructura	- Jerárquico - Plano
			Basado en atributo	
		Entrega	Semántica	- Best effort - Exacto uno - Max. Uno - Min. uno
			Mecanismo	- Unicast - Multicast - Broadcast
		Enrutado	Membership	- Acotado - Adhoc
			Path	- Estático - Caché - Dinámico

Ejemplos

- Usualmente los conectores se componen entre sí
- Ejemplos de conectores compuestos para “distribución” de grandes cantidades de información en Internet:
 - Basado en eventos
 - Grid
 - Cliente/Servidor
 - Peer to peer



Ejercicios

- Cada grupo analiza un conector (Event-based, Grid-based, C/S, P2P). Expone las características del mismo con respecto a las dimensiones que hemos visto
- Caracterizar completamente los conectores utilizados en el trabajo de la asignatura