Introducción al Software basado en Componentes

Juan José Moreno Navarro
Curso de Doctorado LSIIS
(junto con Lars-Ake Fredlund)

Motivación

- Antecedentes: Sistemas distribuidos y el problema de la reutilización.
- Nuevos marcos de aplicación del software y nuevas formas de abordarlos.
- Dominios de aplicación

Un poco de historia.

- Sistemas Distribuidos
- Sistemas Abiertos
- El problema de la reutilización.
- Documentos abiertos.
- Evolución de los modelos de programación.
- WWW
- Comercialización: Software a la carta.

Sistemas Distribuidos

- Programas concurrentes ejecutados en diferentes procesadores sin memoria común
  - Extensiones y nuevos mecanismos de concurrencia (paso de mensajes, RPC, sockets).
  - Dependen de un medio físico de comunicación: cable serie, bus interno (transputers), red, etc.
  - Problemas de fiabilidad - Tolerancia a fallos hardware y software.

Sistemas Abiertos

- Sistemas distribuidos que integran programas heterogéneos e incluso aplicaciones ya existentes (legacy systems).
- Necesitan modelos nuevos ya que los mecanismos y las lógicas aplicables a la concurrencia no son válidos.

Sistemas Abiertos

- Actores: Objetos con un thread interno (activos)
  - mensajes asíncronos
  - comunicación punto a punto
  - una dirección que puede transmitirse lo que permite configuraciones dinámicas
  - los actores pueden crear otros actores (dinámismo)
  - Muchas versiones (sincronismo local, protocolos de interacción entre actores, tiempo real, agrupación de actores, etc.)

Sistemas Abiertos

- Modelos de coordinación
  - Construir programas pegando diferentes piezas activas: programa, tarea, thread, ...
  - Entidades a coordinar + medio de coordinación + leyes de coordinación
  - Lenguaje de coordinación: elementos lingüísticos para describir modelos de coordinación.

  - Orientados a datos: Modelos de pizarra - Linda
  - Orientados al control

Sistemas Abiertos

- Filtros: Permiten modificar el modelo de objetos separando los datos de la computación y sincronización

  Se añaden a los objetos manipulando los mensajes de salida y/o de entrada.
  - Patrón de aceptación para ver si lo manipula o los deja pasar.
  - Acciones con un mensaje:
    - Suspenderlo hasta que se verifique una condición.
    - Dejarlo pasar sin modificarlo.
    - Modificarlo de acuerdo al comportamiento del filtro.
    - Pasarlo a otro objeto distinto al original.
    - Responderlo.

El problema de la Reutilización

- Un viejo problema en la producción de software: Reutilizar elementos ya realizados.

  Ejemplo: UNIX - La búsqueda con expresiones regulares estaba repetido en varias aplicaciones (grep, ls, ll, ...)

  Ventajas:
  - Productivas (menor esfuerzo de desarrollo)
  - Los elementos reutilizados no hay que probarlos.

  Se reutiliza todo (análisis, diseño, código, documentación, ...)

  Comúnmente código en bibliotecas de programas.

El problema de la Reutilización

- Problemas de reutilizar bibliotecas:
  - Exige recompilar, reconfigurar, reinstalar la aplicación cuando se modifica una parte.
  - Difícil comercialización: exige entregar código (aunque solo sea objeto) que puede entrar en su proceso productivo. No puede facturarse por uso.
  - Ocupa espacio en nuestro sistema y exige un serio control de configuración.

¿Cuántas aplicaciones Windows utilizan el corrector ortográfico?
¿Cuántas lo usarán? ¿Netscape debe hacer el suyo?

El problema de la Reutilización

- Sugerencia:

Reutilizar binarios.

A favor:

- Más comercializables.
- Más fáciles de actualizar. No se recompila.

En contra:

- Más difíciles de documentar y especificar.
- Mecanismo de llamadas más complejo de usar.
- Los binarios no son compatibles.

Bibliotecas dinámicas o más allá.

Documentos abiertos

- OpenDoc: Uno de los primeros sistemas de componentes en un marco de aplicación concreto. Apple ▸ OMG
  - Documentos compuestos: documentos basados en elementos distribuidos que a su vez pueden ser documentos.
  - LiveObjects o Partes: Pertenecen a un documento compuesto y contiene datos (que pueden contener Partes). Cada parte tiene asociado un editor.

Documentos abiertos

- Mecanismos básicos para la gestión de documentos compuestos:
  - Estructura de archivos (Bento): Maneja versiones, permite asociar propiedades o etiquetas a los documentos que permiten manipularlos de cierta forma.
  - Transferencia de datos entre partes: Usa SOM (modelo de componentes de IBM) para empaquetar y distribuir las partes.
  - Arquitectura para configurar y manejar partes: Open Scripting Architecture (variante de Apple Script) basada en objetos lógicos, eventos semánticos e independencia del lenguaje de configuración (script) que se utilice.

Modelos de programación

- Evolución natural:
  - Programación modular.
  - Programación orientada a objetos.
  - Programación orientada a aspectos.
  - Programación orientada a componentes.

Modelos de programación

- El problema de la incompatibilidad de los binarios se resuelve volviendo a las máquinas abstractas que ejecutan código intermedio:
  - Primeros compiladores.
  - Programación declarativa
  - Java

Modelos de programación

- Java:
  - Lenguaje orientado a objetos casi puro.
  - Herencia simple.
  - Recogida de basura automática - no hay punteros.
  - Máquina abstracta de Java (JVM) independiente de la plataforma que ejecuta código intermedio (bytecodes).
  - Applets: Pequeñas aplicaciones que pueden ser transferidas por la red y ejecutadas en la máquina cliente. Diferente a los CGIs - Los clientes traen el código y lo ejecutan ellos en vez de ser el servidor el que ejecuta el código.

Modelos de programación

- Java:
  - Applets: la seguridad se comprueba en la carga y en la ejecución impidiéndoseles acceder a los recursos locales.
  - Los objetos son serializables: se transforman en texto en una única cadena, con lo que pueden transmitirse.
  - RMI (Remote Method Invocation) implementa un modelo cliente-servidor en donde el cliente puede invocar de forma remota los métodos del servidor.
  - JDBC: permite a los applets y aplicaciones Java un acceso a bases de datos usando SQL.
  - Reflexión: Capacidad limitada para metaprogramación: inspección de objetos y métodos

Modelos de programación

- Arquitectura de SUN de aplicaciones WEB/Java

Cliente: Interfaces de usuario

Aplicaciones (Lógica del negocio)

Almacenamiento persistente de datos

•Limpia y sencilla.

•Ineficiente: los applets no pueden acceder a los recurso locales luego muchas utilidades básicas requieren mucho intercambio de información entre los clientes y los servidores.

•WebTop servers

Modelos de programación

- Progamación orientada a aspectos
  - Los programas se descomponen en diferentes aspectos, encargándose cada uno de ellos de un requisito independiente (prestaciones, seguridad, concurrencia, fiabilidad)
  - Los aspectos se desarrollan de forma separada e, incluso, en diferentes lenguajes.
  - Weaber (tejedor): herramienta que se encarga de juntar todos los aspectos ordenadamente entre sí.
  - Modelo atractivo pero complicado/poco desarrollado.

WWW

- Primera generación: Publicación de contenidos con HTTP/HTML.
- Segunda generación: Posibilidad de crear páginas dinámicamente, formularios, bases de datos.

  CGI (Common Gateway Interface) que define una forma estándar para que los procesos servidores puedan ejecutar aplicaciones y procesar la información introducida por el usuario.

WWW

- Tercera generación:
  - Se asume que todo es WEB: lo de dentro (corporativo), lo de fuera y del medio (soporte, nodos, etc.)
  - Asume la movilidad de los usuarios: reciben los mismos servicios desde diferentes lugares y a diferentes terminales.
  - Asume la seguridad del mundo conectado
  - Mayor estructuración de los datos (XML).
  - Servicios web

Software a la carta

- Las empresas de software quieren vender productos ampliables sin necesidad de reinstalar las aplicaciones cuando se adquiere un producto suyo:

Plug and play

No requiere ningún cambio en el sw/hardware.

Software a la carta

- Windows:
  - Nuevo hardware / drivers.
  - Nuevo producto Office: Reutiliza componentes anteriores (corrector ortográfico, inserción/manejo de hojas de cálculo, macros en Visual Basic, etc.)
  - Impresoras disponibles para todas las nuevas aplicaciones.

Mismo proveedor.

Software a la carta

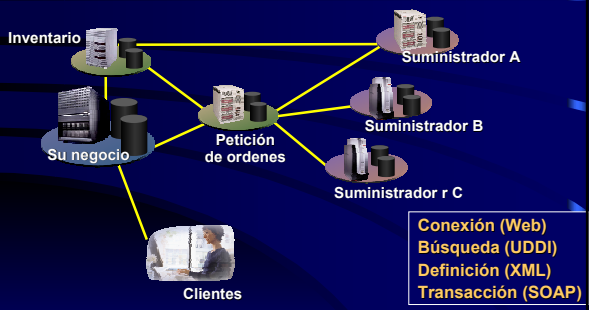
- Netscape:
  - Admite ampliaciones con nuevos
  - El repertorio de plug-ins no está prefijado de antemano. Surgen nuevos que se adaptan.
  - En el desarrollo de netscape no se han previsto todos los posibles.
  - Los desarrolladores de estos plug-ins no saben cómo está hecho netscape.

Diferente proveedor.

Dominios de aplicación

- Sistemas operativos y sus aplicaciones: Windows / Gnome
- Comercio electrónico: Pocas variaciones.
- Comercialización de componentes: COTS (*Components on the self*)
- Servicios web: Componente que puede usarse en los programas propios accesible via protocolos de web estándares.

Nuevo modelo de desarrollo: Basado en estándares WEB



Componentes en la WWW

Organización del curso

- Programación basada en componentes
- Arquitecturas Software
- Un nuevo modelo de desarrollo de software
- Formalización del software basado en componentes
- Contratos: Validación y verificación
- Erlang: Ejemplo de lenguaje
- Servicios Web