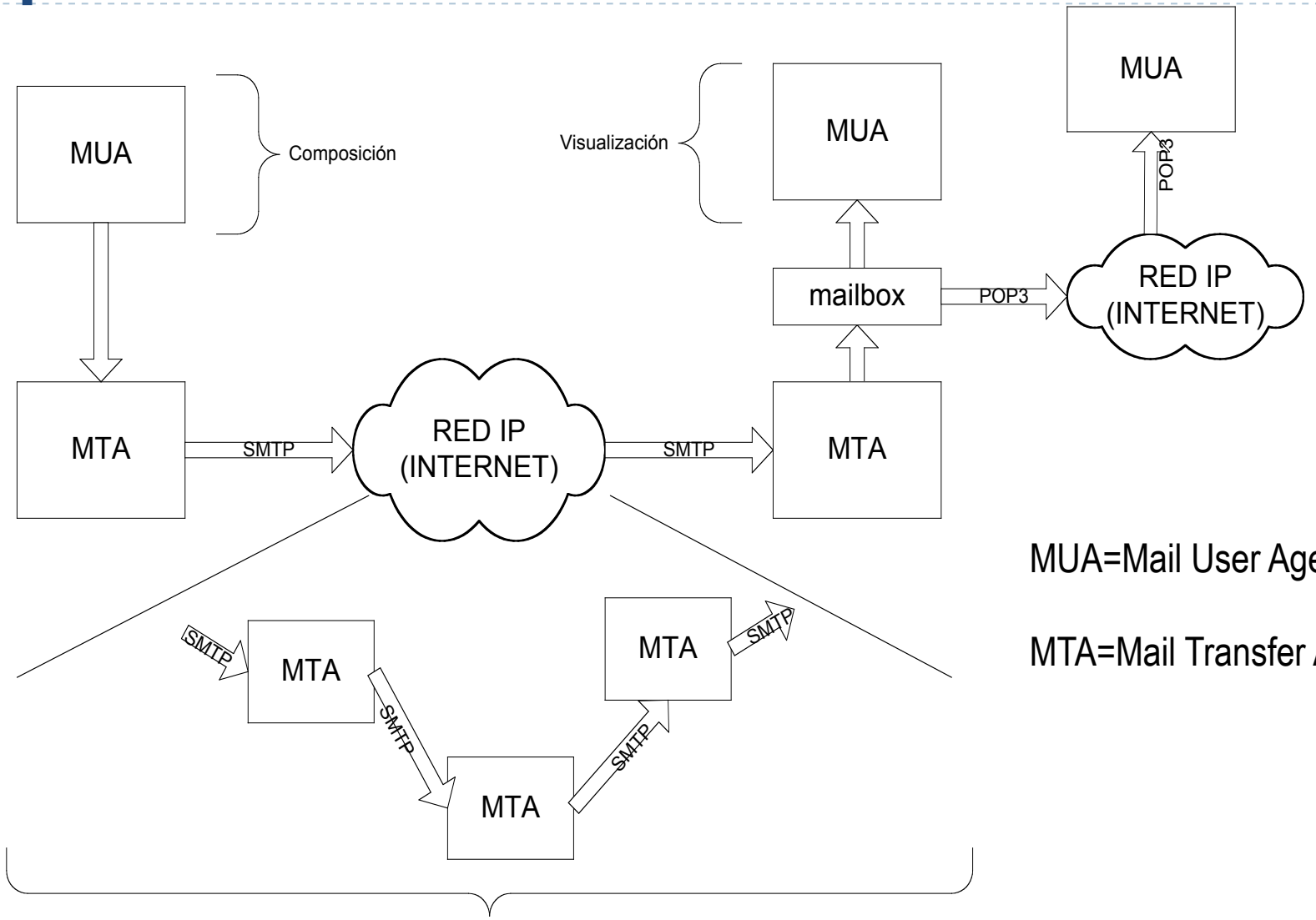
Tema 7 – Desarrollo de servicios en red

Humberto Martínez Barberá <humberto@um.es>

Contenidos

- Servicios de correo electrónico
- Servicios de directorio

Arquitectura de sistemas de email



MUA=Mail User Agent

MTA=Mail Transfer Agent

Formato de los mensajes de correo

- RFC 822
- Los mensajes sólo usan US-ASCII (7 bits)
- Formato de los mensajes:
  - Sobre (envelope).
    - Datos específicos del medio de transporte.
    - Los MTA pueden modificarlo
  - Encabezado.
    - Datos necesarios para la manipulación del mensaje por parte del MUA.
  - Cuerpo del mensaje
    - Separado por una línea en blanco de lo anterior

MIME: Multipart Internet Mail Extension

- El RFC 822 sólo describe el uso de texto ASCII en el cuerpo del mensaje. Esto plantea problemas:
  - Mensajes en idiomas con acentos (español)
  - Mensajes en alfabetos no latinos (ruso)
  - Mensajes en idiomas sin alfabetos (japonés)
  - Mensajes que no contienen texto (audio y video)
- Solución: ampliación del RFC 822 con los RFC 2045 y 2046, que definen nuevas cabeceras
  - Premisa básica: compatibilidad con el RFC 822. Sólo hay que cambiar los MUA.
  - Cabeceras MIME:
    - Content-Transfer-Encoding: 8bit, base64, quoted-printable, …
    - Content-Type: text/plain, image/jpg, messgae/rfc822, …

Mensajes Multipart

- Los mensajes de correo pueden incorporar varios objetos en diferentes “partes” del mismo mensaje.
- Para establecer la separación entre las partes se emplea un límite formado por una secuencia de caracteres
- El tipo MIME de los mensajes con varias partes es:
  - Content-Type: multipart/mixed; boundary=*cadena_separacion*
- A continuación se muestra un mensaje con un fichero gzip adjunto

Ejemplo (1)

Cabecera

```
Received: from unimur.um.es (unimur.um.es [155.54.1.1])
by gaia.fcu.um.es (8.8.8/8.7.3) with ESMTP id LAA28872
for <edumart@fcu.um.es>; Wed, 21 Apr 1999 11:03:47 +0200 (MET DST)
Received: from aries. (aries.dif.um.es [155.54.12.152])
     by unimur.um.es (8.9.1b+Sun/8.9.1) with SMTP id LAA05335
     for <edumart@fcu.um.es>; Wed, 21 Apr 1999 11:03:45 +0200 (MET DST)
Received: from dif.um.es by aries. (SMI-8.6/SMI-SVR4)
     id LAA27788; Wed, 21 Apr 1999 11:03:20 +0100
From: "Angel L. Mateo" <amateo@dif.um.es>
To: edumart@fcu.um.es
Sender: amateo@dif.um.es
```

Ejemplo (2)

Encabezado

```
Message-Id: <199904211003.LAA27788@aries.>
X-Mailer: exmh version 2.0.2 2/24/98
Subject: prueba de attachment
Mime-Version: 1.0
Date: Wed, 21 Apr 1999 11:02:13 -0100
Content-Type: multipart/mixed ; boundary="==_Exmh_18787698320"
Content-Length: 3751
X-UIDL: 924685439.000
X-Mozilla-Status: 8001

This is a multipart MIME message.
```

Ejemplo (3)

```
--==_Exmh_18787698320

Content-Type: text/plain; charset=us-ascii


Hola Eduardo:


Echale un vistazo al fichero que te paso


Salu2

Angel
```

1ª parte: texto

Ejemplo (4)

```
--==_Exmh_18787698320
Content-Type: application/x-gzip ; name="prueba.gz"
Content-Description: prueba.gz
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="prueba.gz"

H4sICA+m7zYAA3JlbnVuY2lhLnBzALVYbY/bNhL+rl/BwyFAesBaEkW9LYoFmmxTFEjbRZJe
AxzugyxRjhpZUiXZa8fY/37PDClZ3iR3SNrDWjY5M5z3GY72yd/uXl99V7RrfRWsPOfJ2+6u
r5pR91c/tYWur8UPutF9lYu7dhhf533VjcJSOE+ePO91Nrb9tfitagr3V/G2MyjxT90PVduI
YOWvPPG0rprd4Rvx9Hm77apai9ts1Nfiu91GyFj4aZoIP7n21XUoQfS2u91tt8dfB91/Axlv
fXacIKwZH+C9R9OC8UoUcOHRu459e3zTZ1WNd8x+ePzG5ydIf0+EMhB4SXTotbbDAfpHvD0z
/+/c57fh//Ewf925WAlZhgAAA==

--==_Exmh_18787698320--
```

2ª parte: fichero gzip

SMTP: Simple Mail Transfer Protocol

- Los MTA son servidores que escuchan en el puerto 25. Usan el protocolo SMTP (sobre TCP) definido en RFC 821
  - El servidor es el primero en anunciarse:

```
220 juanita.um.es ESMTP Sendmail 8.8.5/8.7.3
  (IRIS 1.1); Wed, 21 Apr 1999 09:36:40 GMT
```

  - El cliente responde enviando su dominio:

```
HELO alu.um.es
```

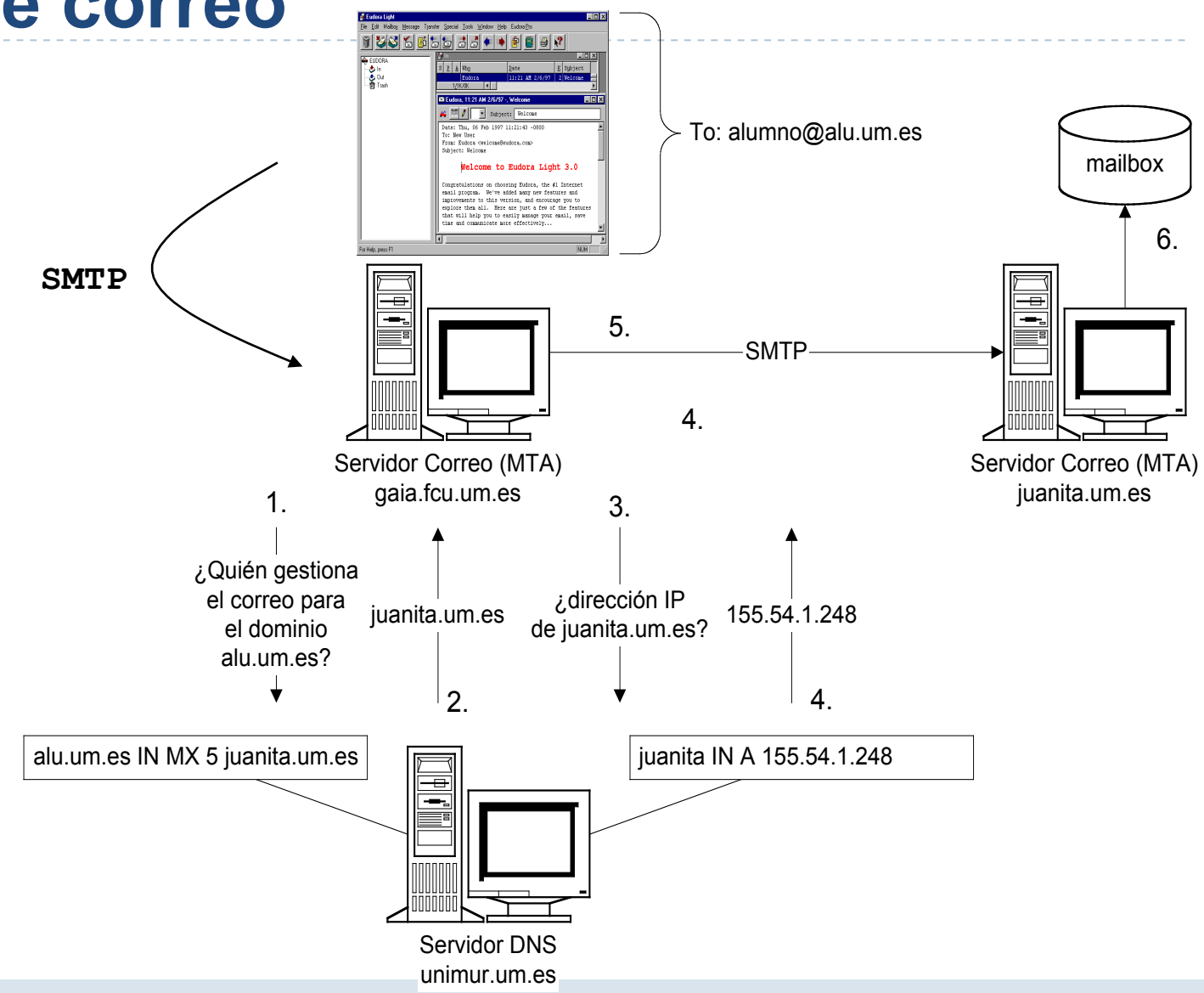
  - El cliente va enviando las distintas partes del mensaje:

```
MAIL FROM:<edumart@fcu.um.es>
```

  - El servidor confirma si la petición es correcta

```
250 <edumart@fcu.um.es>... xxxxxxxxxxxx
```

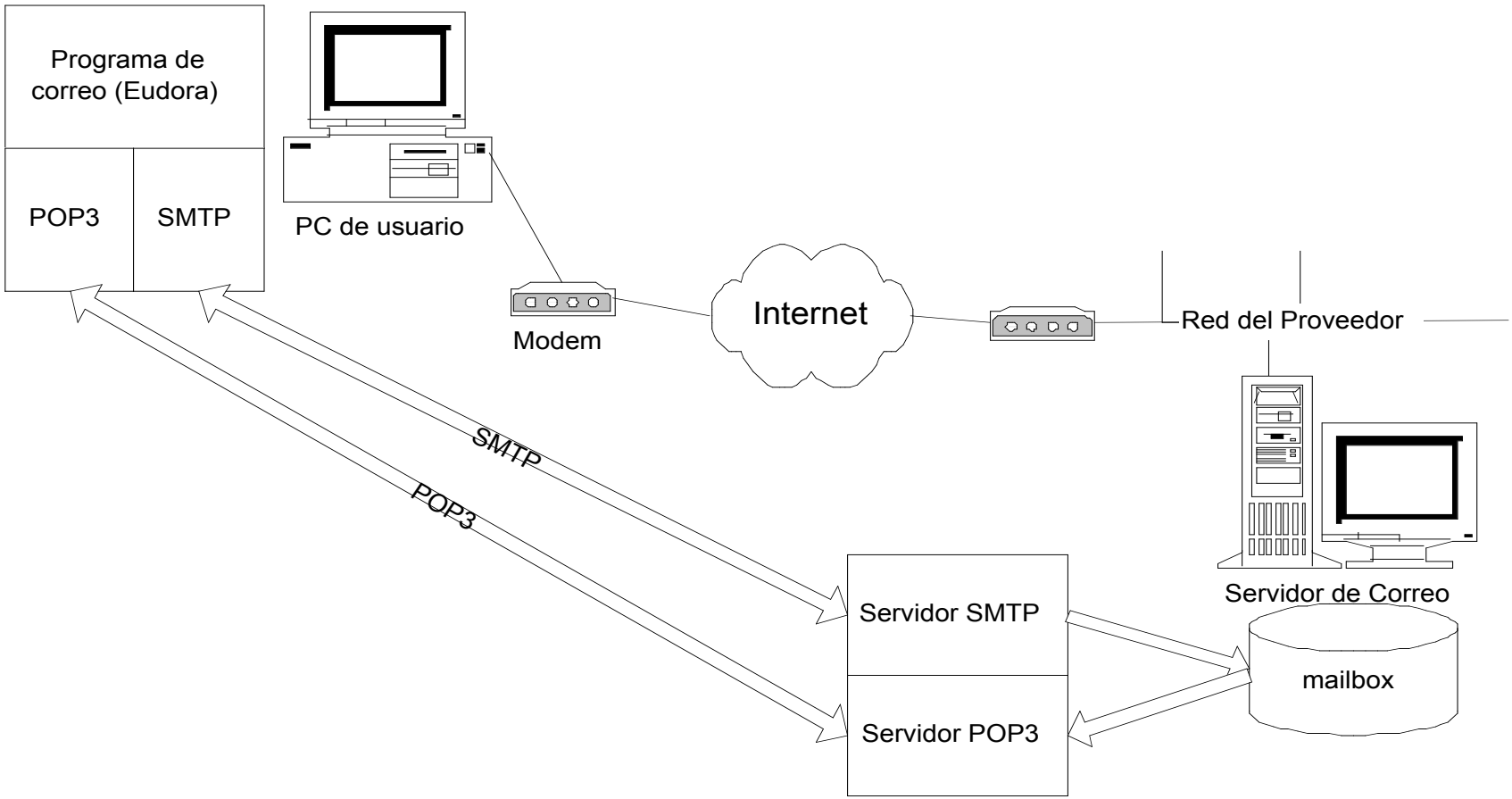
Envío de correo



POP3: Post Office Protocol

- El mailbox de un usuario está en un equipo que está permanentemente conectado a Internet, con entrada en el DNS como servidor de correo de un dominio => no puede estar en el PC del usuario.
- Hace falta un mecanismo para acceder al mailbox, especialmente cuando el PC se conecta a través de una conexión telefónica con un proveedor.
- Solución: protocolo POP3 (Post Office Protocol)
- POP3 trabaja sobre TCP, y el servidor utiliza el puerto 110.

Consulta de correo



IMAP: Internet Mail Access Protocol

- POP3 plantea problemas a los usuarios nómadas
- IMAP, definido en RFC 2060, permite tener en el servidor carpetas y mensajes.
- IMAP proporciona comandos para:
  - Mover mensajes entre carpetas
  - Buscar mensajes
  - Obtener partes de cada mensaje

JavaMail

- Extensión estándar de Java para leer y componer mensajes de correo
- Soporta los protocolos SMTP, POP3 e IMAP
- Clases principales:
  - Session: sesión con un servidor de correo
  - Message: mensaje de correo
  - Address: dirección de correo
  - Transport: protocolo para enviar mensajes
  - Store: protocolo para recuperar mensajes
  - Folder: carpeta en el servidor de correo
- Paquetes: javax.mail, javax.mail.internet, javax.activation

JavaMail: Session

- Emplea java.util.Properties para disponer de información (ej: servidor de correo).
- Creación de una sesión compartida:

```
Properties props = System.getProperties();

props.put("mail.smtp.host", servidor_correo);

Session session = Session.getDefaultInstance(props, null);
```

- Creación de una sesión única:

```
Session session = Session.getInstance(props, null);
```

- Segundo parámetro: Authenticator

JavaMail: Message

- Construcción de un mensaje (javax.mail.internet)

```
MimeMessage message = new MimeMessage(session);
```

- Asignación de contenido:

```
message.setContent("Hello", "text/plain");

message.setText("Hello");
```

- Asunto del mensaje:

```
message.setSubject("Asunto");
```

JavaMail: Address

- Creación de una dirección:

```
Address address = new InternetAddress("edumart@um.es");

Address address = new InternetAddress("edumart@um.es","Eduardo Martinez");
```

- Emisor: Address direcciones[] = { … };

```
message.setFrom(address);

message.addFrom(direcciones);
```

- Receptor:

```
message.addRecipient(tipo, direcciones);
```

- Tipo es:

```
Message.RecipientType.[TO|CC|BCC]
```

JavaMail: Transport

- Mecanismo de transporte por defecto: cierra la conexión

Transport.send(message);

- Alternativa: mantiene la conexión hasta close

```
message.saveChanges();

Transport transport = session.getTransport("smtp");

transport.connect(host, username, password);

transport.sendMessage(message, message.getAllRecipients());

transport.close();
```

JavaMail: Store

- La recuperación de mensajes (POP3 o IMAP) comienza también con un objeto Session
- Store representa al protocolo de acceso al mailbox:

```
Store store = session.getStore("pop3");

store.connect(host, username, password);
```

JavaMail: Folder

- Obtención de mensajes:

```
Folder folder = store.getFolder("INBOX");

folder.open(Folder.READ_ONLY);

Message message[] = folder.getMessages();
```

- En servidores POP3, hay un folder INBOX solamente.
- Recuperación del mensaje:

```
String m = ((MimeMessage)message).getContent();
```

- Cierre del folder (booleano que indica actualizar y borrar):

```
folder.close(bool);

store.close();
```

JavaMail: Ejemplo de envío

```
String host = …;
String from = …;
String to = …;
Properties props = System.getProperties();
props.put(“mail.smtp.host”, host);
Session session = Session.getDefaultInstance(props, null);
MimeMessage message = new MimeMessage(session);
message.setFrom(new InternetAddress(from));
message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));
message.setSubject(“Hola JavaMail”);
message.setText(“Bienvenido a Java”);
Transport.send(message);
```

JavaMail: Ejemplo de recuperación

```
String host = …; String username = …; String password = …;

Properties props = new Properties();

Session session = Session.getDefaultInstance(props, null);

Store store = session.getStore("pop3");

store.connect(host, username, password);

Folder folder = store.getFolder("INBOX");

folder.open(Folder.READ_ONLY);

Message message[] = folder.getMessages();

for (int i = 0, n = message.length; i < n; i++) {

    System.out.println(i + ": " + message[i].getFrom()[0] + "\t" +
    message[i].getSubject());

}

folder.close(false);

store.close();
```

JavaMail: Borrar mensajes

- Abrir el folder en modo lectura escritura:

```
folder.open(Folder.READ_WRITE);
```

- Marcar el mensaje para ser borrado:

```
message.setFlag(Flags.Flag.DELETED, true);
```

- Cerrar el folder con parámetro *true*:

```
folder.close(true);
```

JavaMail: Mensajes Multipart

- Cada parte es una instancia de la clase MimeBodyPart
- Las partes se agrupan en un contenedor de la clase MimeMultipart, que es el contenido del mensaje
- Las partes que contienen ficheros emplean un FileDataSource:

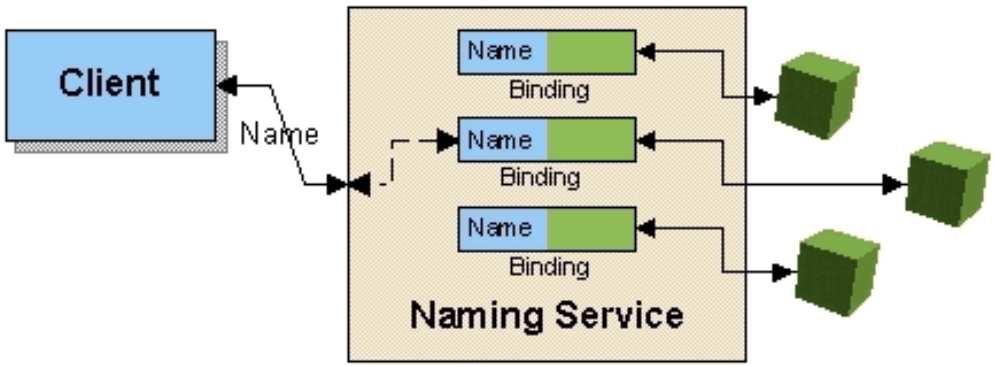
```
String fichero = "…";

messageBodyPart = new MimeBodyPart();

FileDataSource source = new FileDataSource(fichero);

messageBodyPart.setDataHandler(new DataHandler(source));

messageBodyPart.setFileName(fichero);
```

JavaMail: Ejemplo correo Multipart

```
Message message = new MimeMessage(session);
message.setFrom(new InternetAddress(from));
message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));
message.setSubject("Hola con attachment");
BodyPart messageBodyPart = new MimeBodyPart();
messageBodyPart.setText("Hola mundo");
Multipart multipart = new MimeMultipart();
multipart.addBodyPart(messageBodyPart);
messageBodyPart = new MimeBodyPart();
DataSource source = new FileDataSource(fichero);
messageBodyPart.setDataHandler(new DataHandler(source));
messageBodyPart.setFileName(fichero);
multipart.addBodyPart(messageBodyPart);
message.setContent(multipart);
Transport.send(message);
```

Servicios de nombres (1)

- Herramienta fundamental en cualquier sistema de computación
  - Nombres
  - Bindings
  - Referencias y direcciones
  - Contextos
  - Sistemas de nombres y espacios de nombres



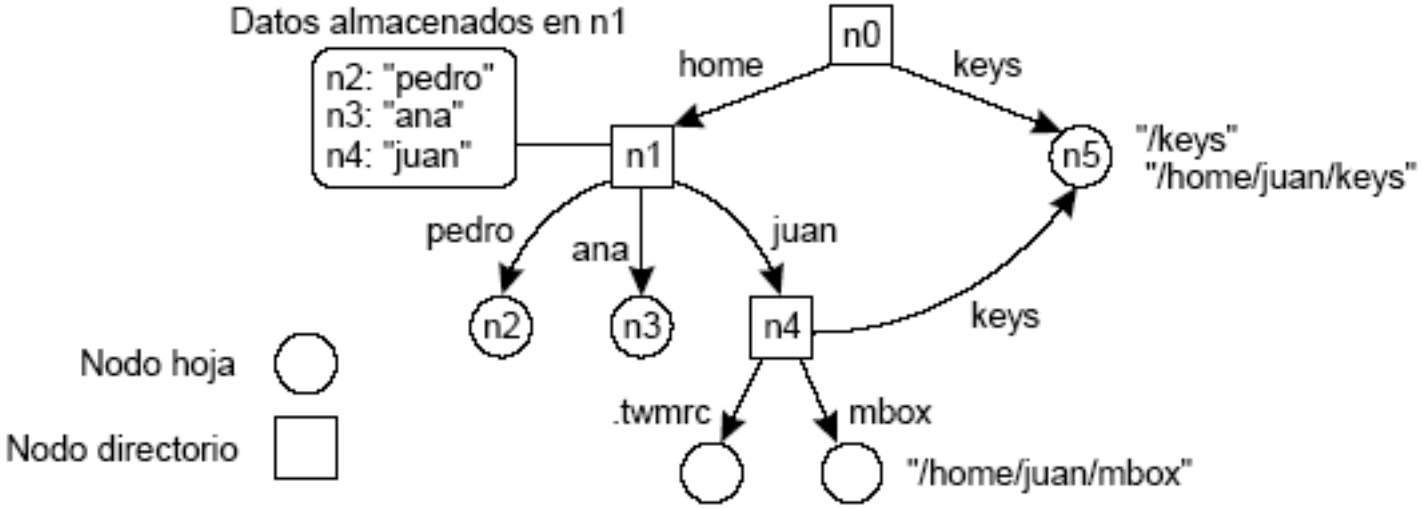
Ejemplos:
DNS (Domain Name Service)
Registro RMI
Servicio de nombres de CORBA

Servicios de nombres (2)

- **Identificador**. Un nombre que cumple las siguientes propiedades:
  - Un identificador se refiere como mucho a una entidad.
  - Cada entidad tiene un único identificador.
  - Un identificador siempre se refiere a la misma entidad.
- Las direcciones y los identificadores se representan en forma adecuada a las máquinas (secuencia de bits).
- Un nombre de entidad pensado para ser empleado por humanos (human-friendly) se suele representar con una secuencia de caracteres.

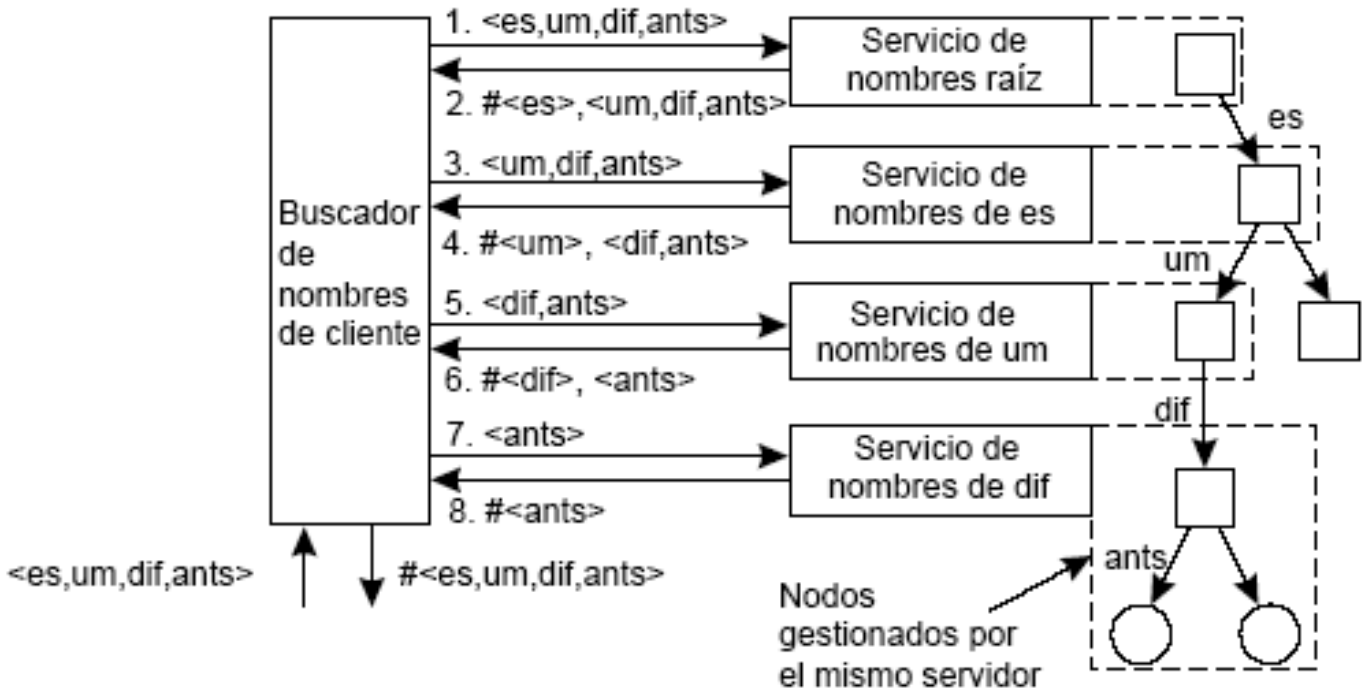
Espacios de nombres

- **Espacio de nombres**. Conjunto de nombres de entidades que siguen una misma convención de nombres.
  - Un espacio de nombres puede representarse como un grafo dirigido con *nodos hoja* y *nodos directorio*



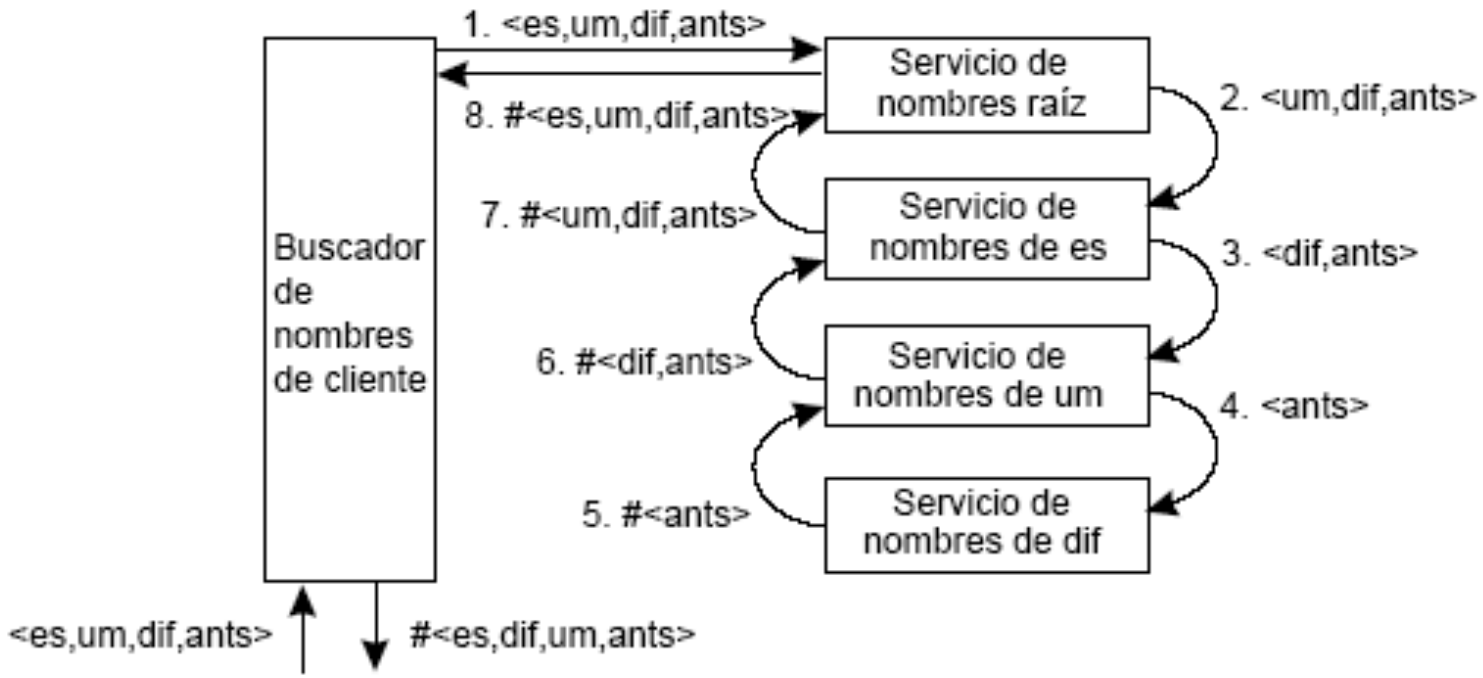
Implentaciones (1)

- **Resolución de nombres iterativa**: el buscador de nombres local consulta iterativamente a los servicios de nombres, siguiendo la secuencia de etiquetas desde el nodo raíz.



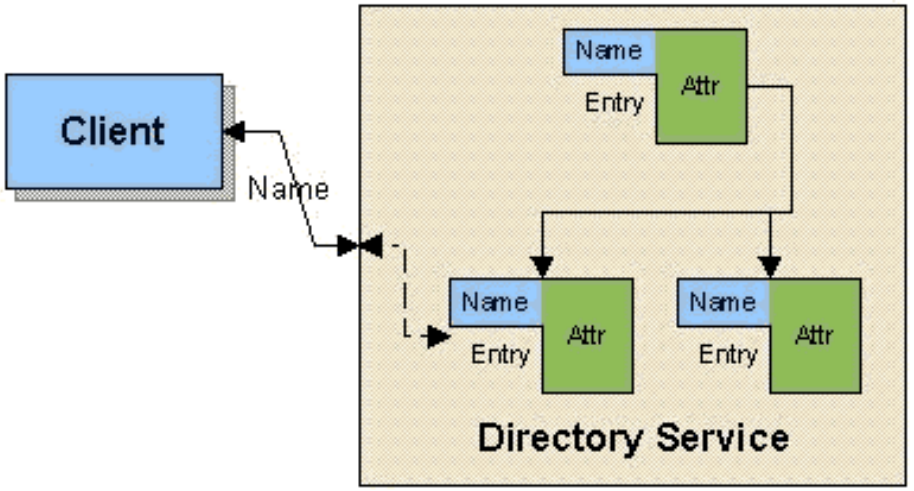
Implementaciones (2)

- **Resolución de nombres recursiva**: los servicios de nombres delegan la resolución de parte del nombre a otros servicios de nombres



Servicios de directorio (1)

- Amplían la funcionalidad de los servicios de nombres asignando atributos a los objetos
  - Atributos
  - Búsquedas y filtros



Ejemplo: directorios LDAP (Lightweight Directory Protocol)

Servicios de directorio (2)

- DNS es un ejemplo de servicio de nombres tradicional: dado un nombre estructurado, resuelve el nombre recuperando un nodo cuyo contenido es un registro de recurso.
- **Servicio de directorio**. Es un tipo especial de servicio de nombres en el que un cliente puede buscar una entidad mediante una descripción de sus propiedades, en lugar de su nombre.
- En esta sección se describe el servicio de directorio OSI X.500, y su versión más ligera para Internet, LDAP (Lightweight Directory Protocol).

Servicios de directorio (3)

- Un servicio de directorio contiene un conjunto de registros, denominados entradas.
- Cada entrada está formada por una colección de pares (*atributo*, *valor*), que describen características de una entidad.
- Cada atributo tiene un tipo asociado, y puede ser mono-valuado o multi-valuado.
- Los servicios de directorio proporcionan operaciones para crear, añadir, borrar y modificar atributos de una entrada.
- La operación más potente es la de búsqueda, empleando expresiones lógicas en las que se especifican los atributos de las entradas buscadas.

X.500

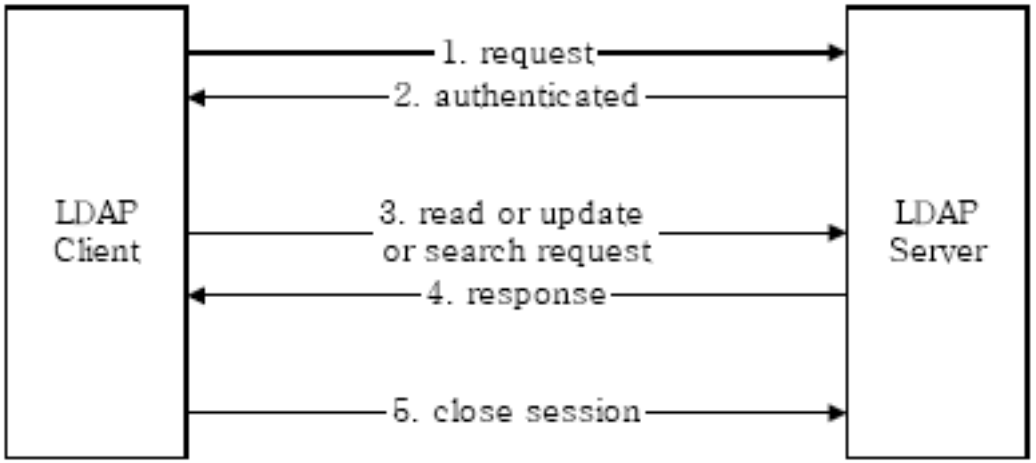
- Servicio de directorio
  - Data Communications Network Directory, Recommendations X.500-X.521
  - Organiza las entradas del directorio en un espacio de nombres jerárquico
  - Define potentes capacidades de búsqueda
  - Directory Access Protocol(DAP)
    - Comunicación entre el cliente y el servidor de directorio
    - Usa la pila de protocolos OSI
    - Es un protocolo con un gran consumo de recursos

LDAP (Lightweight Directory Access Protocol)

- LDAP se diseñó en la Universidad de Michigan como alternativa *ligera* a DAP, empleando la pila TCP/IP (RFC 2251). Se definió una API C estándar de facto (RFC1823).
- LDAP es un estándar abierto, incorporado en muchos productos software, como Microsoft Active Directory o clientes de correo.
- Método estándar de acceso y actualización de la información del directorio
  - Simplifica algunas operaciones X.500
  - Es un protocolo de comunicaciones
  - No define un interfaz de programación

Interacción básica LDAP

- La interacción entre un cliente y un servidor LDAP sigue estos pasos:
  - El cliente establece una sesión con el servidor (*binding*).
  - El cliente solicita la ejecución de operaciones de lectura, actualización y búsqueda.
  - El cliente cierra la sesión con el servidor (*unbinding*).

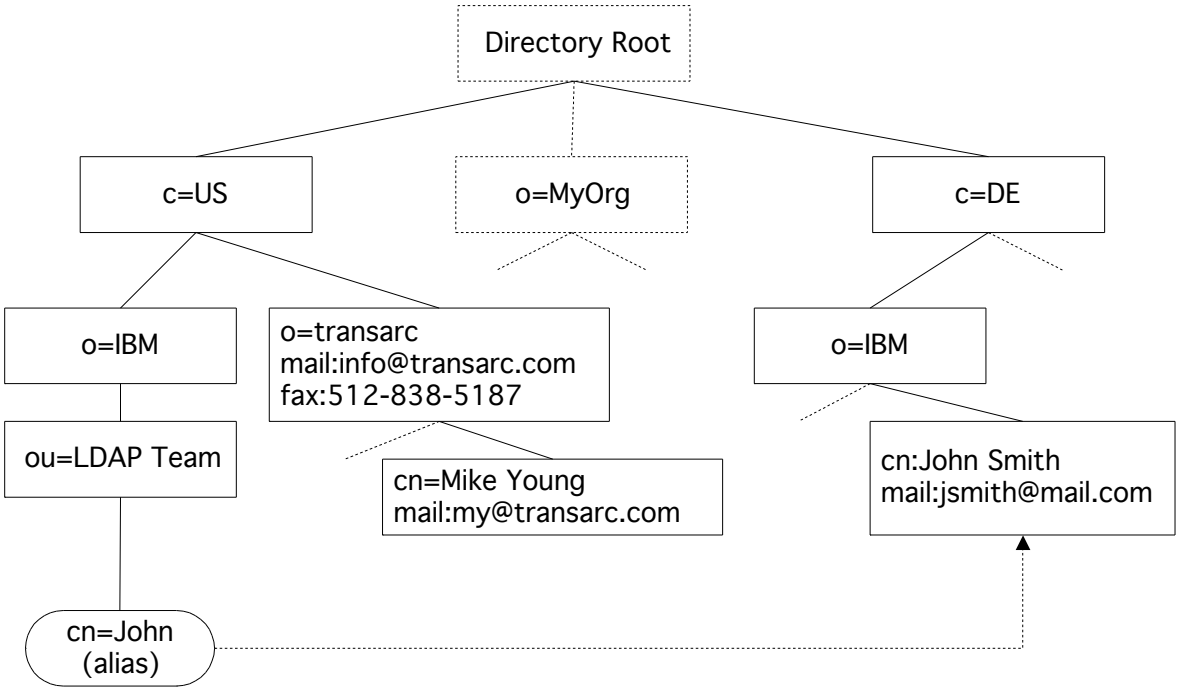


Modelos de LDAP

- **Modelo de información**. Describe la estructura de la información almacenada en un directorio LDAP (tipos de datos): *bin, ces, cis, …*
- **Modelo de nombres**. Describe el espacio de nombres de LDAP, es decir, cómo se organiza e identifica la información: *DIT, DN, RDN*.
- **Modelo funcional**. Describe qué operaciones se pueden ejecutar sobre la información de un directorio LDAP: *consultas, actualización, autenticación*.
- **Modelo de seguridad**. Describe cómo se protege la información de accesos no autorizados: *kerberos, SSL*.

Modelo de nombres

- Ejemplo de DIT

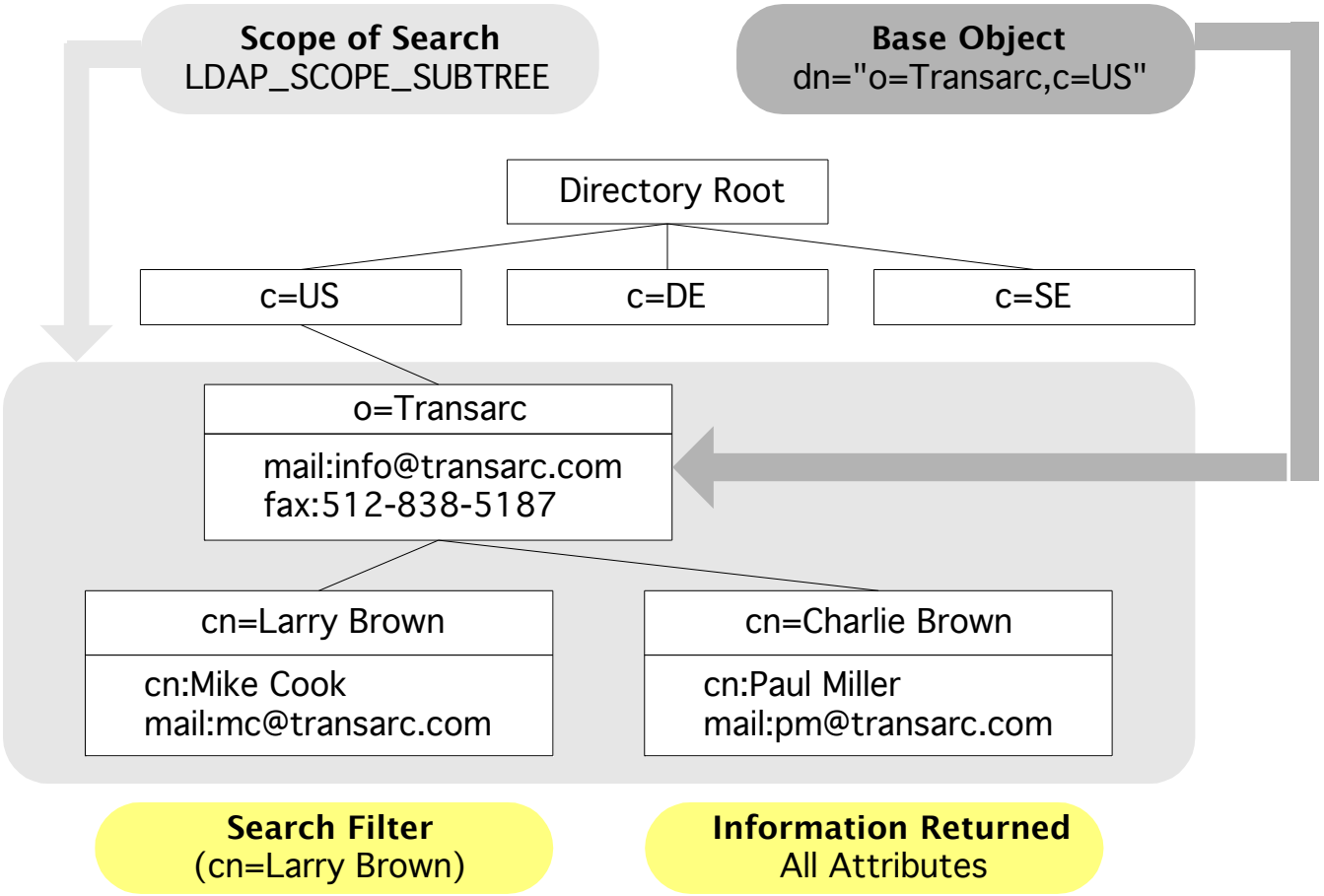


cn=John,ou=LDAP Team,o=IBM,c=US

cn=John Smith,o=IBM,c=DE

Modelo funcional

- Operación de consulta



Formato LDAP URL (1)

- Formato de URLs para recursos LDAP

  ldap[s]://[<host>[:<port>]][/[<dn>[?[<attributes>][?[<scope>][?[<filter>][?<extensions>]]]]]]

  - *host*: equipo en el que se encuentra el servidor
  - *port*: puerto utilizado por el servidor (por defecto 389).
  - *dn*: entrada base
  - *attributes*: atributos devueltos, separados por comas.
  - *scope*: alcance de la búsqueda
    - *base*: entrada base sólo
    - *one*: primer nivel
    - *sub*: subárbol
  - *filter*: filtro de búsqueda.
- Los caracteres no admitidos en URL se especifican con %

Formato LDAP URL (2)

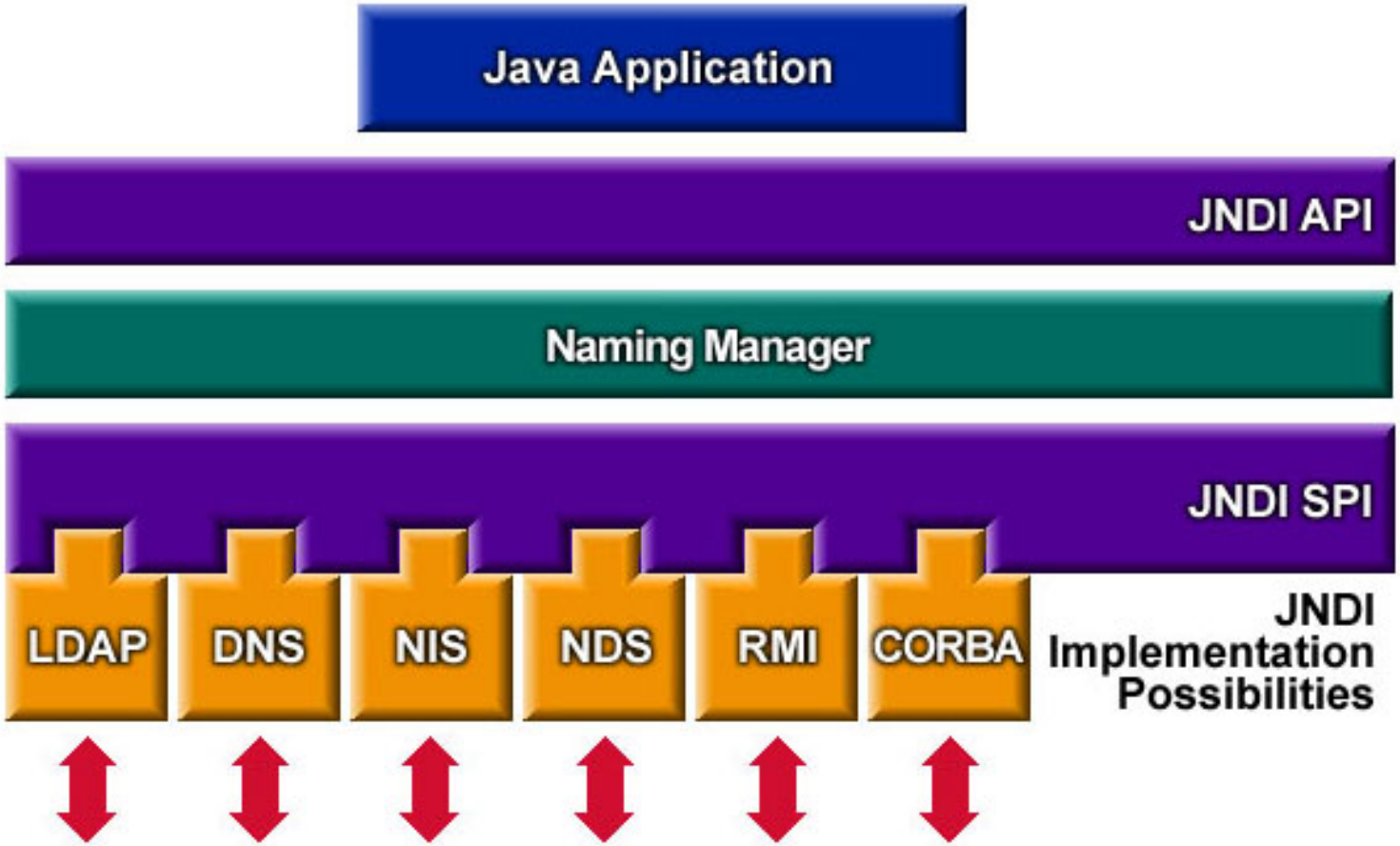
ldap:///o=Universidad%20de%20Murcia,c=ES

ldap://ldap.um.es/o=Universidad%20de%20Murcia,c=ES

ldap://ldap.um.es/o=Universidad%20de%20Murcia,c=ES?postalAddress

ldap://ldap.um.es/o=Universidad%20de%20Murcia,c=ES??sub?
(cn=Eduardo%20Martinez)

Arquitectura de JNDI



Proveedor de servicio

- Componente de JNDI para integrar el acceso a algún tipo de servicio de nombres o directorio:
  - Lightweight Directory Access Protocol(LDAP)
  - CORBA services(COS) naming service
  - Java Remote Method Invocation(RMI) Registry
  - Network Information System(NIS)
  - File System
  - Domain Name System (DNS)
  - Novell NDS

Paquetes JNDI

- Parte del Standard Development Kit 1.3, jndi.jar
- javax.naming
  - Acceso a los servicios de nombes
- javax.naming.directory
  - Acceso a los servicios de directorio
- javax.naming.event
  - Soporta notificación de eventos en servicios de nombres y directorios.
- javax.naming.ldap
  - Características propias de LDAP v3 no cubiertas en el paquete javax.naming.directory
- javax.naming.spi
  - Interfaz de proveedor de servicio

Contexto inicial (1)

- Punto de inicio para la resolución de nombres

1. Especificar el proveedor de servicio

```
Hashtable env = new Hashtable();
env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
```

- Otros proveedores:
  - Sistema de archivos:
    - com.sun.jndi.fscontext.RefFSContextFactory
  - Registro RMI:
    - com.sun.jndi.rmi.registry.RegistryContextFactory

Contexto inicial (2)

2. Especificar otra configuración:

```
env.put(Context.PROVIDER_URL, "ldap://ldap.wiz.com:389");
env.put(Context.SECURITY_PRINCIPAL, "user");
env.put(Context.SECURITY_CREDENTIALS, "password");
```

3. Creación del contexto inicial:

```
Context ctx = new InitialContext(env);
DirContext ctx = new InitialDirContext(env);
```

Contexto inicial (3)

- Uso del fichero de propiedades jndi.properties

```
java.naming.provider.url="ldap://localhost:389/o=jnditutoria“

java.naming.factory.initial="com.sun.jndi.ldap.LdapCtxFactor"
```

- jndi.properties en:
  - algún directorio del classpath
  - JAVA_HOME/lib/
- Constructor sin parámetros

```
Context ctx = new InitialContext();
```

Ejemplo servicio de nombres

```
try {
     // Contexto inicial
     Context ctx = new InitialContext(env);
     // Buscar un objeto
     Object obj = ctx.lookup(name);
     // Casting y uso
     System.out.println(name + " ligado a: " + obj);
} catch (NamingException e) {
     System.err.println("Problemas con " + name + ": " + e);
}
```

Ejemplo servicio de directorio (1)

```
try {
    // Contexto indicial de directorio
    DirContext ctx = new InitialDirContext(env);
    // Atributos del objeto
    Attributes attrs =
       ctx.getAttributes("cn=Eduardo Martinez, ou=DIIC");
    // Encontrar el nombre de pila ("sn") e imprimirlo
    System.out.println("sn: " + attrs.get("sn").get());
} catch (NamingException e) {
    System.err.println(“Problemas recuperando atributo:" + e);
}
```

Ejemplo servicio de directorio (2)

```
DirContext ctx = new InitialDirContext(env);
String base = “o=UM, c=ES”;
String filter = “(|(cn=Mart*)(cn=Per*))”;
SearchControls contraints = new SearchControls();
contraints.setSearchScope(searchControls.SUBTREE_SCOPE);
NamingEnumeration results = ctx.search(base,filter,contraints);
while (results.hasMore())
{
    SearchResult sr = (SearchResult)results.next();
    System.out.println(sr.getName());
    Attributes attrs = sr.getAttributes();
    if (attrs != null)
    {
```

Ejemplo servicio de directorio (3)

```
        for (NamingEnumeration ne=attrs.getAll(); ne.hasMore();)
        {
             Attribute attr = (Attribute)ne.next();
             String id = attr.getID();
             for (Enumeration vals = attr.getAll(); vals.hasMoreElements();)
                  System.out.println(id + ":"+ vals.nextElement()));
        } // for attrs.getAll()
    } // if attrs!=null
} // while results.hasMore()
```

Ligar y desligar objetos (1)

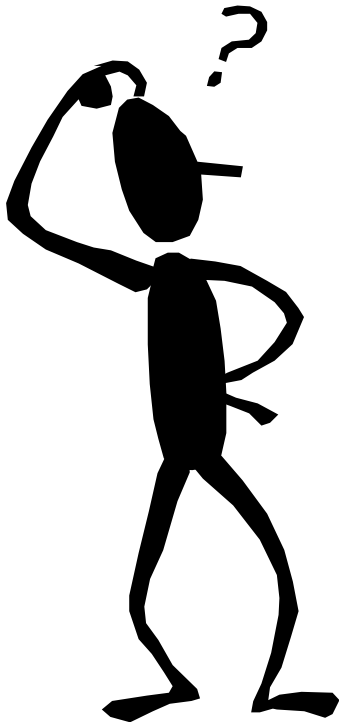
```
Context ctx = ……

…….

// Crear un objeto para ser ligado

VideoServer vs = new VideoServer();

// Ligar

ctx.bind(“VS1”, vs);

…….

// Desligar

ctx.unbind(“VS1”);
```

Ligar y desligar objetos (2)

```
DirContext ctx = ……

// Crear una entrada para un objeto RMI

VideoServer vs = new VideoServer();

// Ligar

ctx.bind(“cn=VideoServer”, vs);

// Crear una entrada para un objeto serializado

String codebase = ...;

Flower f = new Flower("rose", "pink");

ctx.bind("cn=Flower", f, new BasicAttributes("javaCodebase", codebase));
```

Bibliografía

- A.S. Tanenbaum (1998) *Computer Networks*. Prentice-Hall
- H. Johner (1998). *Understanding LDAP*. IBM Corporation, International Technical Support Organization
- T. Howes, M. Smith (1997) *LDAP. Programming Directory-Enabled Applications with Lightweight Directory Access Protocol*. Macmillan Technical Publishing
- R. Weltman, T. Dahbura (2000) *LDAP Programming with Java*. Addison Wesley



Humberto Martínez Barberá (humberto@um.es)
Área de Ingeniería Telemática
Departamento de Ingeniería de la Información y las Comunicaciones (http://www.diic.um.es:8080/diic/)
Facultad de Informática (http://www.um.es/informatica)