

# Sistemas Informáticos II

---

## Diapositivas extra de CORBA

[http://www.dia.eui.upm.es/asignatu/sis\\_dis/Paco/Comunicacion.pdf](http://www.dia.eui.upm.es/asignatu/sis_dis/Paco/Comunicacion.pdf)

Irene Rodríguez ([irene.rodriguez@uam.es](mailto:irene.rodriguez@uam.es))

---

# CORBA

MODELO para el desarrollo de sistemas distribuidos orientados a objetos

- Nace en 1991. Actualmente se va por la versión 3.

Desarrollado por  
el Object Management Group  
(OMG)  
para conseguir software:

Orientado a objetos  
Portable  
Reusable  
Distribuido  
Que funcione en  
entornos heterogéneos

- Implementaciones gratis y de pago: Orbacus, ORBIX, Visibroker

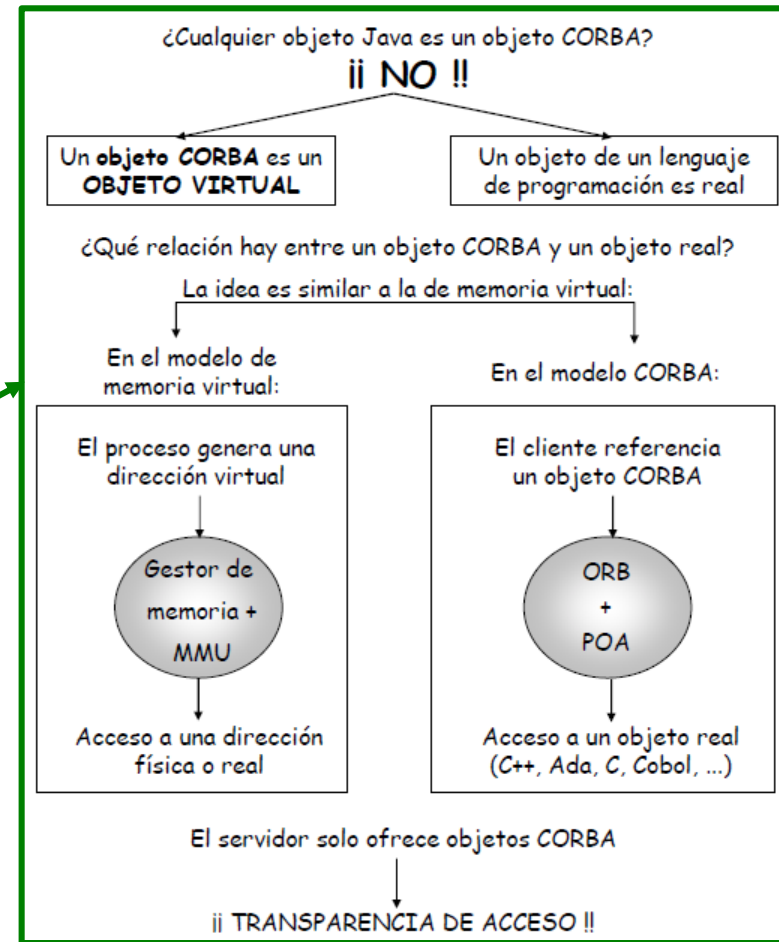
CORBA se compone:

El modelo de  
Objetos

Describe qué es un  
objeto CORBA

El modelo de  
referencia

Indica la arquitectura  
para que los objetos  
CORBA puedan  
relacionarse



# CORBA

MODELO para el desarrollo de sistemas distribuidos orientados a objetos

- Nace en 1991. Actualmente se va por la versión 3.

Desarrollado por  
el Object Management Group  
(OMG)  
para conseguir software:

Orientado a objetos  
Portable  
Reusable  
Distribuido  
Que funcione en  
entornos heterogéneos

- Implementaciones gratis y de pago: Orbacus, ORBIX, Visibroker

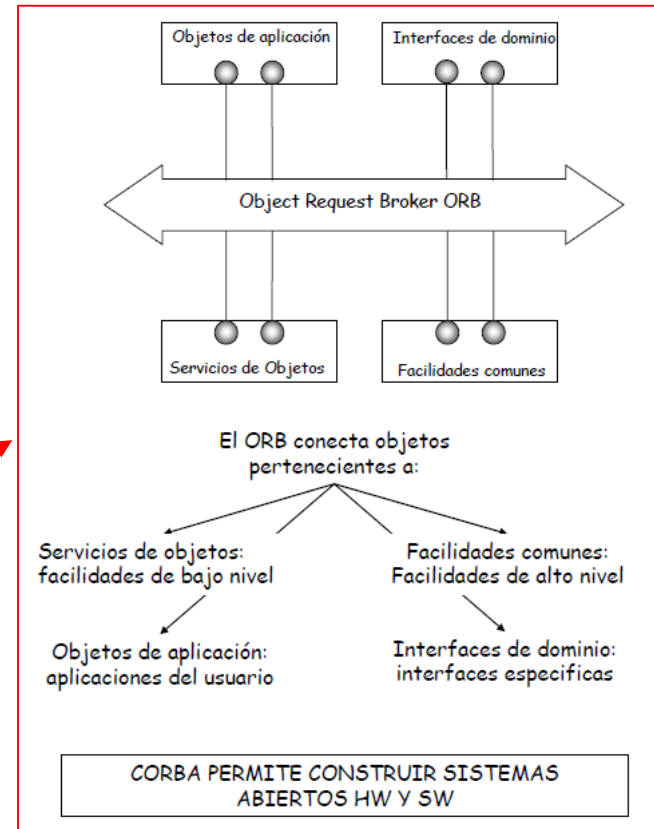
CORBA se compone:

El modelo de  
Objetos

Describe qué es un  
objeto CORBA

El modelo de  
referencia

Indica la arquitectura  
para que los objetos  
CORBA puedan  
relacionarse



# CORBA

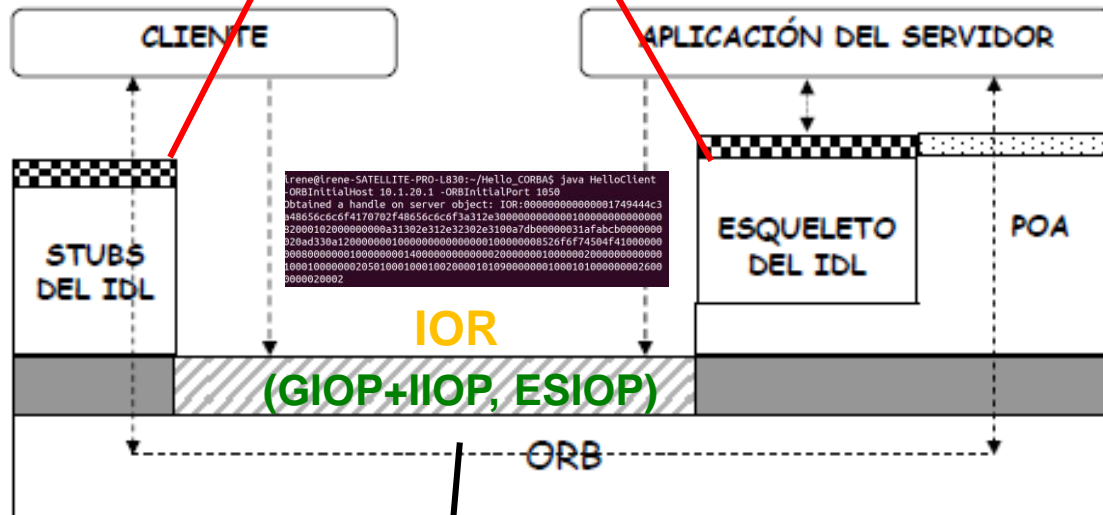
Servant=implementación de la interfaz

La especificación en CORBA con idl:

```
// Matematico.idl
interface matematico {
    double dividir (in double dividendo, in double divisor)
        raises(DivisionPorCero);
};
```

Se admite más de un parámetro de entrada y/o salida

Los errores del servidor se asocian a excepciones

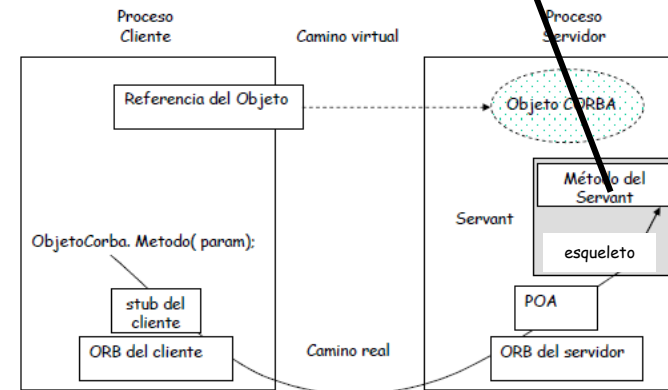


- Interfaz dependiente de la implementación del ORB
- Interfaz idéntica en el lado del cliente y servidor
- Interfaz idéntica en todas las implementaciones de ORB

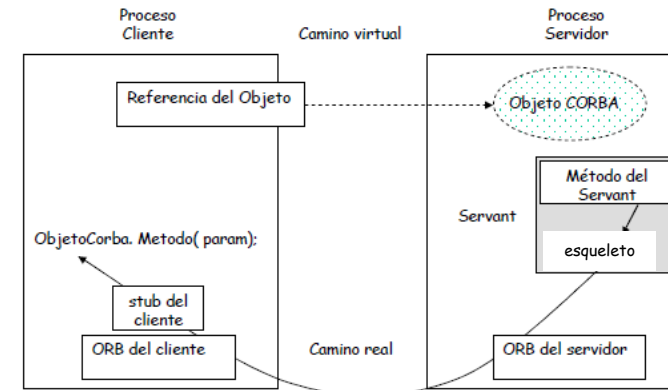
- Un objeto CORBA se identifica de manera única en todo el sistema mediante su **referencia de objeto**. Una referencia de objeto es un identificador de comunicación que el cliente usa, pero no interpreta, para poder comunicarse con el servidor.
- La asociación entre un objeto CORBA y el objeto real que lo implementa la realiza el **Portable Object Adapter(POA)**.

**POA(IOR)=objeto real que lo implementa**

Llamada a un método CORBA:



Envío de la respuesta:



# CORBA

## La especificación en CORBA con idl:

```
// Matematico.idl
```

```
interface matematico {
```

```
    double dividir (in double  dividendo, in double divisor)
```

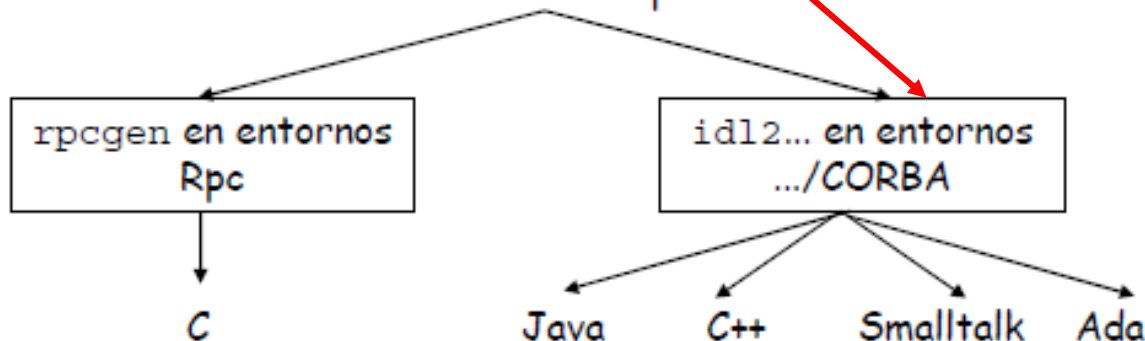
```
    raises(DivisionPorCero);
```

```
};
```

Se admite más de un parámetro  
de entrada y/o salida

Los errores del servidor  
se asocian a excepciones

Los interfaces se someten al compilador de interfaz:



Objeto remoto CORBA (genérico)

Servidor CORBA

Casting a la clase del objeto

Para ofrecer un servicio, el servidor

1. Crea un objeto real (servant)

2. Crea un objeto Corba

3. Los asocia

4. Da de alta el objeto CORBA

```
// Server.java
import org.omg.PortableServer.*; import org.omg.CosNaming.*;

public class Server {
    public static void main(String[] args) {
        try {
            // 1. Inicialización del ORB
            org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);

            // Obtención del identificador del POA raíz
            POA rootPOA =
            POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
            // Activación del POA manager
            rootPOA.the_POAManager().activate();

            // Creación de una implementación del servicio (servant)
            MatematicoImpl managerServant = new MatematicoImpl();

            // Se crea el objeto CORBA y se asocia al servant
            org.omg.CORBA.Object CORBAObj =
            rootPOA.servant_to_reference(managerServant);

            // 2. Localización del servicio de nombres
            org.omg.CORBA.Object rootObj =
            orb.resolve_initial_references("NameService");
            NamingContextExt root =
            NamingContextExtHelper.narrow(rootObj);

            // 3. Alta del s. matematico en el servicio de nombres
            root.bind(root.to_name("MAT_PROGRAM"), CORBAObj);


            // 4. Espera de peticiones
            orb.run();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

1. Inicialización del soporte de transmisión (ORB).
- Obtención del identificador del **POA raíz** (mediante el servicio de nombres **global**)  
Activación del POA manager
- Creación de una implementación (REAL) del servicio
- Se crea el objeto CORBA (genérico) y se asocia al servant. Esto se hace a través del POA, que le asignará la referencia al objeto CORBA (IOR).
2. Localización del servicio de nombres (global).  
En tiempo de ejecución, el cliente pregunta al ORB dónde está el servicio de nombres (transparencia ubicación del servidor de nombres)
3. Alta del servicio en el servidor de nombres, el servidor da de alta en el servidor de nombres global el objeto CORBA MAT\_PROGRAM que ofrecerá el servicio Matematico.
4. Espera de peticiones. Se cede el control al esqueleto/dispatcher que se encarga de recoger peticiones.

# Servidor CORBA

---

```
// matematicoImpl.java
import org.omg.PortableServer.*;
public class MatematicoImpl extends MatematicoPOA {
    public int dividir (int dividendo,int divisor)
        throws MatematicoPackage.DivisionPorCero {
        if (divisor == 0)
            {throw new MatematicoPackage.DivisionPorCero();}
        else
            {return (dividendo/divisor); }
    }
}
```



Implements MatematicoOperations (interface)  
Interface Matematico extends MatematicoOperations

# Cliente CORBA

```
// Client.java
import org.omg.CosNaming.*;

public class Client {
    public static void main(String[] args) {
        try{
            // 1.Inicialización del ORB ①
            org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);

            // 2.Se Obtiene el identificador del servicio de nombres ②
            org.omg.CORBA.Object
            rootObj = orb.resolve_initial_references("NameService");
            NamingContextExt
            root = NamingContextExtHelper.narrow(rootObj);

            // 3.Se Obtiene el identificador del servicio matemático ③
            org.omg.CORBA.Object
            mgrObj = root.resolve(root.to_name("MAT_PROGRAM"));
            Matematico servidor = MatematicoHelper.narrow(mgrObj);

            // Se obtienen los operandos
            int dividendo = Integer.parseInt(args[0]);
            int divisor = Integer.parseInt(args[1]);

            // 4. Se invoca la operación de división ④
            int cociente = servidor.dividir(dividendo,divisor);

            // 5. Se imprime el resultado ⑤
            System.out.println(dividendo + "/" + divisor+ " = " + cociente);
        }

        // Se controlan las excepciones del servicio
        catch (MatematicoPackage.DivisionPorCero e) {
            e.printStackTrace();
        }

        // se controlan los errores de comunicación y del sistema
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

1. Inicialización del soporte de transmisión (ORB).

2. Localización del servicio de nombres (global). En tiempo de ejecución, el cliente pregunta al ORB dónde está el servicio de nombres (transparencia ubicación del servidor de nombres)

3. Obtención del identificador de objeto remoto (IOR). Se obtiene al invocar la operación resolve del servicio MAT\_PROGRAM sobre el servicio de nombres. Si el cliente guarda el identificador, puede usarlo en sucesivas ejecuciones.

4. Invocación al método del servicio.  
**objetoCorba.método(parámetros)**

Control de las excepciones propias del servicio

Control de las excepciones de comunicación y del sistema