ACTIVIDAD 1 (Conceptos varios)

Proporcione una breve descripción **comprensible** (máximo de 50 palabras) para cada uno de los siguientes conceptos:

Concepto	*Descripción*
Sockets	Es un punto de comunicación entre procesos en una red de ordenadores
Servicios Web	Sistema software diseñado para proporcionar interacciones ordenador-ordenador utilizando la red
Colas de mensajes	Sistema de mensajería en el que los nodos de un sistema distribuido envían mensajes a colas y también los reciben de colas. Las colas las gestiona un bróker de comunicaciones.
RPC	Remote Procedure Call. Precursora de la invocación a método remoto. Permite de forma transparente que un programa llame a un procedimiento que se ejecutará en un ordenador distinto al del programa llamante.
XML	Extensible Markup Language. Es un lenguaje de marcas desarrollado que permite representar información de forma legible por las personas pero procesable por las aplicaciones. Se ha convertido en un estándar del envío de información en la WWW
JSON	JavaScript Object Notation, es un estándar abierto que utiliza texto legible para transmitir objetos JavaScript. Se utiliza principalmente para transmitir datos entre un servidor y un cliente web, como alternativa a XML
Serialización	Procedimiento que permite empaquetar un objeto en un nodo y transmitir dicho objeto "serializado" a un nodo destino, de tal forma que se puede crear en el nodo destino un nuevo objeto copia del original.

ACTIVIDAD 2 (ROI)

1. Ordene, según el orden en que tienen lugar, los siguientes pasos de una ROI. ¿Falta algún paso para que todo el mecanismo ROI funcione? Si es así, descríbalo.

A. El método llamado finaliza y se desbloquea el esqueleto

B. El proxy desempaqueta los resultados y los devuelve al proceso cliente

C. El proceso cliente invoca el método del proxy local relacionado con el objeto remoto

D. El esqueleto empaqueta los resultados y llama al ORB, el cual hace llegar el mensaje al proxy

E. El proxy empaqueta los argumentos y, utilizando la referencia al objeto, llama al ORB

F. El ORB gestiona la invocación, haciendo que el mensaje llegue al esqueleto.

Solución:

C, E, F, A, D, B

Entre los pasos F y A, falta "El esqueleto desempaqueta los argumentos e invoca el método solicitado, quedando a la espera de que finalice"

ACTIVIDAD 3 (ROI)

Sobre el paso de argumentos en el mecanismo de comunicación ROI, indique si las siguientes afirmaciones son Verdaderas (V) o falsas (F), justificando su respuesta:

V	En un paso de parámetros por referencia, ésta puede pertenecer a uno nodo que no sea ni el invocador ni el invocado. JUSTIFICACIÓN: La referencia apuntará al nodo donde esté el objeto, no importa en qué nodo resida, invocador, invocado u otro.
V	Los argumentos se pueden pasar por valor, no sólo mediante referencias a objetos. JUSTIFICATION: En estos casos se serializan para poder transmitir una copia del objeto.
F	Los argumentos que se pasan por referencia se serializan antes de transmitirlos al nodo destino. JUSTIFICATION: Cuando son por referencia, como el término indica, se pasa una copia de la referencia como argumento, que apunta directamente al objeto y por tanto no es necesario serializarlo
V	En un paso de parámetros por referencia, ésta puede pertenecer al nodo invocador. JUSTIFICATION: La referencia apuntará al nodo donde esté el objeto, no importa en qué nodo resida, invocador, invocado u otro.

ACTIVIDAD 4 (ROI)

En el mecanismo ROI, la creación de objetos puede realizarse mediante dos procedimientos distintos. ¿Cuáles son? Indique, para los siguientes pasos, a qué procedimiento se corresponde y en qué orden tienen lugar (Nota: un mismo paso puede pertenecer a los dos procedimientos).

A. El servidor obtiene una referencia al objeto.

B. Un proceso (cliente) solicita a un servidor (factoría) que cree un determinado objeto.

C. El proceso servidor usa la referencia del objeto para registrarlo en un servidor de nombres, proporcionando una cadena de texto como nombre del objeto.

D. El servidor (factoría) devuelve al cliente una copia de la referencia del objeto que ha solicitado crear.

E. Un proceso que conozca el nombre utilizado para registrar el objeto contacta con el servidor de nombres y obtiene una referencia al objeto.

F. Un proceso crea un objeto y lo registra en el ORB.

G. El servidor crea el objeto (que le han solicitado crear) y lo registra en el ORB.

Procedimiento 1: A iniciativa del cliente

Pasos: B, G, A, D,

Procedimiento 2: A iniciativa del servidor

Pasos: F, A, C, E

ACTIVIDAD 5 (Java RMI)

F	Los objetos remotos deben residir en la misma JVM. JUSTIFICACIÓN: Java RMI que los objetos remotos estén distribuidos a lo largo de todas las JVMs donde se ejecuta la aplicación distribuida, no tienen por qué estar todos en la misma JVM
V	Java construye automáticamente los esqueletos y los proxies a partir de la especificación de la interfaz del objeto remoto . JUSTIFICACIÓN: A partir de la interfaz puede construirse completamente el código del proxy, que implementará la interfaz, así como el del esqueleto, que atenderá las peticiones del proxy y las convertirá en las llamadas a la interfaz del objeto

F	Todos los objetos que se pasan como argumentos en Java RMI deben ser remotos, no permitiéndose por tanto pasar objetos locales. JUSTIFICACIÓN: Los objetos locales también pueden pasarse como argumentos, pero eso sí, no pueden pasarse por referencia sino serializados.
F	El servidor de nombres de Java RMI almacena, para cada objeto registrado, su nombre y esqueleto. JUSTIFICACIÓN: Almacena la referencia al objeto, no su esqueleto.
F	El mecanismo de comunicación de Java RMI no tiene nada que ver con el mecanismo de comunicación ROI (invocación a objeto remoto). ***Justificación***: Java RMI puede verse como una instanciación del mecanismo ROI.

ACTIVIDAD 6 (Java RMI)

Indique brevemente todos los pasos que se deben realizar, tanto para el desarrollo del cliente como para el desarrollo del servidor, para implementar la aplicación como Java RMI.

- Hacer que la interfaz Hola extienda "java.rmi.Remote"
- Añadir "throws RemoteException" al método "saluda"
- Hacer que la clase ImpleHola extienda "java.rmi.server.UnicastRemoteObject"
- En el servidor, crear un objeto de la clase "ImpleHola" y registrarlo en el servidor de nombres (rmiregistry) de JavaRMI
- En el cliente, buscar el objeto creado por el servidor en el "rmiregistry" antes de utilizarlo, en vez de crearlo

ACTIVIDAD 7 (Java RMI)

1.- Actualice la siguiente definición del servicio "eco", para que pueda ser utilizado de forma remota.

```
import java.rmi.*;

interface ServicioEco extends java.rmi.Remote {
    String eco (String s) throws RemoteException
}
```

2.- Actualice la clase *ServicioEcoImpl* (que se muestra a continuación) para que implemente el servicio remoto.

```
import java.rmi.*;


class ServicioEcoImpl  implements ServicioEco
extends java.rmi.server.UnicastRemoteObject {
   public String eco (String s){
          return s.toUpperCase();
}
```

3.- En la clase *ServidorEco*, que actúa como servidor, indique cómo se inicia el servicio remoto y cómo se hace accesible usando *rmiregistry.* ¿Con qué nombre se ha registrado el servicio?

El servicio remoto se inicia creando el objeto "srv" y registrándolo con reg.rebind. El nombre utilizado para registrarlo ha sido "Eco".

4. Finalmente, se muestra a continuación el código del cliente (fichero *ClienteEco.java*). Actualice dicho código para que el cliente obtenga una referencia remota asociada al servicio "eco" que hemos implementado.

SOLUCIÓN:
http://laurel.datsi.fi.upm.es/~ssoo/SD.dir/practicas/guiarmi.html

ACTIVIDAD 8 (REST)

1. Para las siguientes llamadas, indique:

a) Si siguen el estándar REST o no. En caso de no ser REST, realice los cambios necesarios para que sean REST.

b) ¿Qué es lo que pretende realizar esa llamada?

1) GET https://api.github.com?type=users&id=captainkidd

Solución: NO es REST. Corrección: GET https://api.github.com/users/captainkidd
Devuelve la información sobre el usuario captainkidd de github.com

2) GET https://api.githum.com/users/captainkidd/edit

Solución: NO es REST. Corrección: PUT https:// api.githum.com/users/captainkidd y asumimos que en el cuerpo del mensaje PUT se proporciona el contenido actualizado del recurso captainkidd. Actualiza la información del usuario captainkidd de github.com

3) GET https://api.githum.com/gists/page/22/per_page/2

Solución: GET https://api.githum.com/gists?page=22&per_page=2

Obtiene la información de los "gists", indicándole por parámetro cuáles nos interesan.

4) POST https://weatherapp.com/messages

Solución: OK. Crea un nuevo mensaje en el servidor. Asumimos que en el cuerpo del mensaje POST se proporciona el contenido del mensaje a crear.

5) GET weatherapp.com/wheaterLookup.do?zipcode=46017

Solución: No es REST. Corrección: GET weatherapp.com/zipcodes/46017. Devuelve la información (del tiempo) del código postal 46017.

6) GET weatherapp.com/getMessages.do?id=10

Solución: No es REST. Corrección: GET weatherapp.com/messages/10. Devuelve el mensaje con identificador 10.

7) GET https://myapp.com/deleteOrder.do?id=10

Solución: No es REST. No podemos especificar una acción (delete) en la URI. Deberíamos realizar DELETE https://myapp.com/orders/10 que borra la "order" con id=10

8) DELETE https://myapp.com/messages/10

Solución: OK. Elimina el mensaje con identificador 10.

9) GET https://myapp.com/messages

Solución: OK. Devuelve todos los mensajes.

ACTIVIDAD 9 (REST)

Dadas las siguientes URIs, indique el resultado que se obtiene al aplicar sobre ellas los métodos HTTP indicados:

Método	URI	Resultado
GET	/messages	Obtiene todos los mensajes
POST		Añade un nuevo mensaje (que se proporciona en el contenido del POST)
GET	/messages/10	Obtiene el mensaje con id=10
PUT		Actualiza el contenido del mensaje con id=10
DELETE		Elimina el mensaje con id=10
GET	/messages/10/comments	Obtiene todos los comentarios del mensaje con id=10
DELETE		Elimina todos los comentarios del mensaje con id=10
POST		Crea un nuevo comentario para el mensaje con id=10
PUT		Reemplaza todos los comentarios del mensaje 10 con una nueva lista de comentarios

ACTIVIDAD 10

1. Generalmente es preferible usar JMS frente a Java RMI cuando es necesario que todos los componentes de la aplicación estén simultáneamente en ejecución.	F
2. La comunicación se considera débilmente acoplada.	V
3. Un cliente JMS es un objeto administrado.	F
4. Las colas de mensajes se crean normalmente utilizando las herramientas administrativas del proveedor JMS.	V
5. Un proveedor JMS es una empresa que ofrece servicios de consultoría relativos a JMS.	F
6. Los objetos que implementan la interfaz Queue se crean llamando a métodos de la interfaz JMSConsumer.	F

7. Los objetos que implementan la interfaz JMSProducer se crean llamando a métodos de la interfaz JMSContext.	V
8. La comunicación normalmente es persistente.	V
9. La comunicación es sincrónica en la respuesta.	F
10. El direccionamiento empleado es del tipo directo.	F

ACTIVIDAD 11 (Mecanismos de Comunicación)

Complete la siguiente tabla sobre características de comunicación:

Mecanismo	Utilización	Estructura y Contenido de los mensajes	Direccionamiento	Sincronización	Persistencia
ROI	2	C	Directo	C	No
Java RMI	2	C	Directo	C	No
Servicios Web RESTful	1	A	Directo	B	No
Java Message Service	1	B	Indirecto	A	Sí

Utilización: 1. Con primitivas básicas de comunicación; 2. Llamadas a procedimientos o métodos remotos

Estructura de los mensajes: A. No estructurados; B. Cabecera + contenido; C. Estructura transparente al programador (determinada por middleware)

Contenido: 1. Bytes; 2. Texto

Sincronización: A. Asíncrona; B. Sincrónica (entrega); C. Sincrónica (respuesta)