EXAMEN Nuestro

1.- Explica el paradigma de computación distribuida denominado SOA.

- Presenta la gestión de recursos de un sistema distribuido como servicios de red que son publicados y descubiertos por los procesos clientes para su consumo.

Un aspecto clave en este modelo es la aparición del agente intermediario (servicio de directorio) que posibilita la localización de los servicios disponibles.

Permite establecer un modelo completamente desacoplado entre el proceso que requiere un servicio y el proceso que lo provee. El proceso consumidor solicita una funcionalidad y se puede conectar a cualquiera de los servicios que la ofrezcan, sin conocer a priori ningún dato del proceso que provee dicho servicio.

·Enumera y describe sus elementos.

+Proveedor de servicio: Expone su información (recursos) como un servicio de red, en el servicio de directorio, para que este permita que se localice el servicio y se pueda acceder a él.

+ Consumidor de servicio: El consumidor se conecta a través de un servicio de red al servicio de directorio para localizar el servicio que requiere, y obtiene la información que busca (descubrimiento) para acceder a un servicio concreto.

+ Servicio de directorio: Almacena la información necesaria sobre los servicios para que un proceso consumidor pueda localizar y descubrir los servicios, y consumirlos. Este elemento provee una característica clave del modelo, transparencia de localización.

·Cuáles son las principales similitudes entre SOA y Cliente-Servidor convencional (Básate en los principios de SOA)

Reusabilidad: No depende de la arquitectura que utilicemos.

Contrato de servicio: En el cliente-servidor, el contrato de servicio no existe y en SOA el contrato viene definido en el WSDL.

Desacoplamiento: Gracias a los contratos de servicios en SOA se genera un desacoplamiento de la localización ya que no necesita saber dónde está ubicado el servidor. En cambio en cliente-servidor necesitamos conocer en todo momento dónde está localizado el servidor (ip+Puerto)

Abstracción: Tanto cliente-servidor como en SOA existen mecanismos de abstracción, por lo tanto los dos son abstractos.

Composición: Tanto cliente-servidor como SOA la composición no depende de la arquitectura sino de cómo esté compuesto el servicio.

Autonomía: No depende de la arquitectura, depende de cómo este implementado nuestro servicio.

Sin estado: En ambos modelos los servicios no suelen almacenar estados, tan solo reciben una entrada y devuelven un mensaje con el resultado de éste.

Descubrimiento: cliente-servidor no cumple con este principio ya que se debe conocer de antemano dónde se encuentra un servicio para poder acceder a él, por el contrario en SOA buscamos en el servidor de directorio para encontrar el servicio buscado.

· ¿Qué relación existe entre el modelo arquitectónico MatchMaker de SOA y el de Servicios WEB?

Los servicios web poseen un conjunto de características que permiten abrir todos los principios de la arquitectura orientada a servicios.

Podemos reproducir SOA con SOAP cómo lenguaje de intercambio, WSDL cómo lenguaje para la descripción de servicios y UDDI para la publicación o registro de estos servicios.

·¿Qué mecanismos de comunicación distribuida (RPC,RMI,ORB,Servicios WEB) utilizarás para implementar SOA? Justifica la respuesta.

El RPC, no cumple con el principio de desacoplamiento porque no dispone de agente intermediario dónde registrar el método y el cliente pueda buscarlo ahí.

El RMI, no cumple el desacoplamiento porque tienes que conocer dónde está el servidor.

Tanto ORB como Servicios WEB se podrían utilizar para implementar SOA porque tienen todos los componentes necesarios como Cliente-Servidor y agente intermediario que permite registrar referencias al objeto. Además cumple con las características principales de SOA que son la interoperabilidad (Comunicar distintos lenguajes o plataformas) y la reutilización.

2.-Dentro de las tecnologías web describe el modelo servlet.

Son programas escritos en Java, se pueden invocar desde un cliente. Se ejecutan en el seno del servidor WEB, que actúa como un contenedor de componentes, realizan una tarea y generan una salida que es recogida por el servidor web y reenviado al navegador.

·Indica sus principales características.

+Portabilidad: Cómo está escrito en java en cualquier máquina con máquina virtual java funciona.

+Rendimiento: Los servlet son instanciados una única vez.

+Concepto de sesión: Puede mantener información acerca de la sesión de un usuario.

+Software distribuido: Se pueden cargar de forma local o de forma remota de forma transparente.

+Concurrencia: Permite simultáneamente distintas peticiones.

+Otras ventajas debidas a estar implementado en java són la orientación a objetos, la modularidad y la reutilización.

·Realiza una breve comparativa con el modelo CGI.

Los servlet se pueden invocar desde un cliente de forma similar a como se invocan los programas CGI. El objetivo es obtener el HTML dinámico.

La diferencia fundamental respecto al modelo CGI se encuentra en que un servlet se ejecuta dentro del contexto, y por lo tanto del control del servidor o el contenedor web. Mientras que en el CGI no existe control sobre la ejecución de la aplicación.

Otra diferencia es en cuanto al rendimiento, los servlet sólo se instancian una vez pudiendo a partir de entonces atender diferentes peticiones y los CGI deben ser instanciados cada vez que se les infoca.

· Comparativa con las páginas activas atendiendo a las recomendaciones de uso.

Las páginas activas se suelen utilizar cuándo no hay que procesar mucha información, y los servlets cuándo se procesa mucha y además hay que manipularla.

El servlet está dedicado a la parte del control, mientras que las páginas activas a la presentación.

Es más fácil realizar código en páginas activas que en los servlets, ya que estos últimos necesitan más código.

3.- ARA 2013- En referencia a la capacidad de distribución de la información y su gestión de un servicio de directorio (básate en el caso de LDAP), enumera y explica, de forma breve los principales motivos que pueden llevar a dicha distribución.

+LDAP es un protocolo ligero para acceder al servicio de directorio tanto local como global y servicio de nombres.

+ Se ejecuta en protocolo TCP/IP con lo que nos permite orientar la conexión y obtener más fiabilidad.

+ Abarca un ámbito más general que aumenta las capacidades de búsqueda.

+ Permite buscar por cualquiera de los atributos. (Se puede hacer un filtrado de búsqueda).

+Refuerza la seguridad para proteger los objetos de intrusos.

· Que técnica es utilizada para realizar las diferentes partes del espacio de nombres cuando es distribuido.

Se utilizaría DNS, que establece una jerarquía de nombre separa nodos en redes TCP/IP en las que asocia cada nombre a una dirección IP. Es un sistema basado en base de datos distribuida que permite obtener una IP a partir de un nombre de dominio (Resolución directa) o viceversa (Resolución inversa). El espacio de nombre DNS tiene una arquitectura jerárquica, un nombre

de dominio está formado por una o más cadenas separadas por puntos llamadas componentes de nombres o etiquetas. No hay delimitadores de principio y fin, un prefijo de un nombre es una sección inicial del nombre que contiene 0 o más componentes complejos.

·Cómo funciona en relación con la resolución de peticiones.

Resolución directa: Dado un DNS localizar su dirección IP asociada.

Resolución inversa: Dado una dirección IP localizar el nombre DNS asociado.

Resolución iterativa gestionada por el cliente: Se pregunta los distintos servicios DNS hasta que uno de ellos responda la petición. Éste sistema está obsoleto.

Resolución iterativa gestionada por el propio servidor: El cliente pregunta su servidor preferido y este va obteniendo partes de la petición iterativamente.

Resolución recursiva gestionada por el servidor: El cliente mira en la cache si la dirección esta resuelta a nivel local. Si no está resuelta conectamos con el servidor DNS el cual conectará con su base de datos de zona y en caso de tampoco encontrarlo aquí se mirara en la cache del servidor para ver si ha sido traducida. Cómo último recurso se conectará con un DNS externo haciendo el mismo paso para buscar.

4.-Dentro de la práctica guida de sockets y RMi que habéis realizado explica esquemáticamente tu arquitectura desde el controlador hasta ls sensores y actuadores.

Controlador: Monitoriza cada 2 segundos el estado de las máquinas. Se inicia al inicio del servidor. Se dispone de un fichero donde el controlador coge la IP con la que buscará los datos de esa máquina. Una vez obtenidos los datos se comprobará que están dentro de los márgenes establecidos y en caso de ser incorrecto se llamara al actuador para que realice los cambios pertinentes. (Corrección de errores, temperatura).

Registro: Sirve para registrar las máquinas en el servicio de nombre. Para que el controlador pueda invocar a los métodos de los objetos remotos.

Máquina: La clase máquina implementa los métodos que han sido definidos en la interfaz de la máquina. Sus métodos serán invocados por el controlador para obtener los datos.

Sensor: Los sensores dispondrán de los valores del estado de las máquinas. Al utilizar RMI estamos llamando a la referencia del objeto, pero no el objeto en sí, por lo que tendríamos los valores sin inicializar. Para solucionar este problema se utilizar un fichero para almacenar los valores.

Actuador: Se encargará de enviar los avisos por el Display cuando hay un error en la máquina. Activar el refrigerador, activar y desactivar la máquina. Es llamado por el controlador.

·Que importancia tiene el RMI Registry

El RMI Registry nos va a permitir localizar las máquinas desde el controlador.