PRIMERA PRUEBA EVALUACIÓN DE SÓFTWARE DE COMUNICACIONES

27 de octubre de 2022

|  |  |
| --- | --- |
| Apellidos |  |
| Nombre |  |

***Notas previas:***

* Mantenga a la vista algún **documento identificativo** durante todo el examen.
* **NO** se permite la utilización de ningún tipo de documentación en el examen.
* **Preguntas 1 a 6:**
  + **Responda en las cajas con V para verdadero y F para falso.**
  + **Cada pregunta puntúa un máximo de 1 punto.**
  + **Cada afirmación respondida correctamente de una pregunta puntúa 0,25 puntos.**

**Pregunta 1**: El concepto general de transparencia en un sistema distribuido incluye: El acceso a recursos locales y remotos con operaciones idénticas.

|  |
| --- |
| V |
| F |
| V |
| F |

El uso de mecanismos para garantizar la seguridad de la información.

El acceso a los recursos sin conocer su localización.

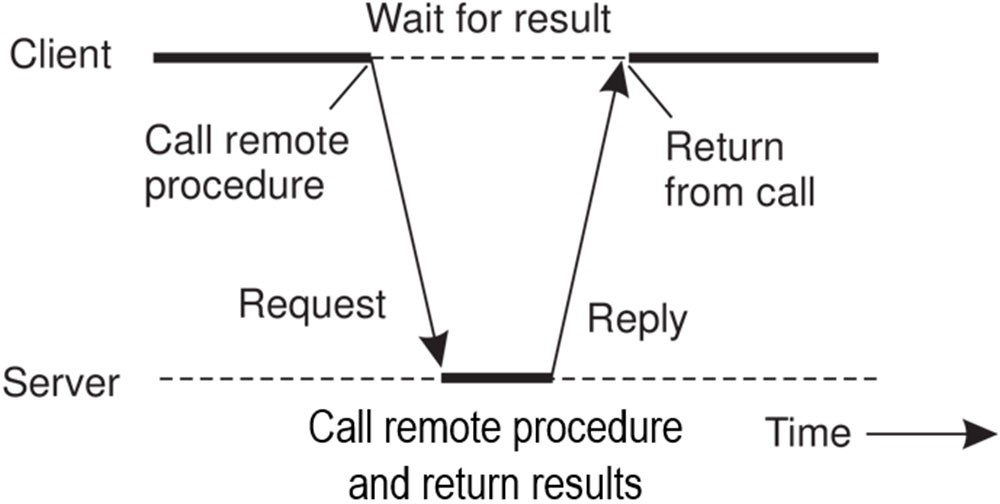
El uso de estándares conocidos en sistemas distribuidos.

**Pregunta 2**: Cuando un procedimiento remoto tiene una semántica “al menos una” (at‐least‐once):

El procedimiento remoto sólo permite el paso de, como mucho, un único parámetro dato. Se garantiza que el procedimiento remoto se ejecuta más de una vez.

|  |
| --- |
| F |
| F/V |
| V |
| F |

Un mensaje de petición de procedimiento remoto puede ser reenviado más de una vez. El servidor ignora las peticiones de procedimiento remoto duplicadas.

**Pregunta 3**: En una invocación síncrona a un procedimiento remoto con una secuenciación como la de la derecha:

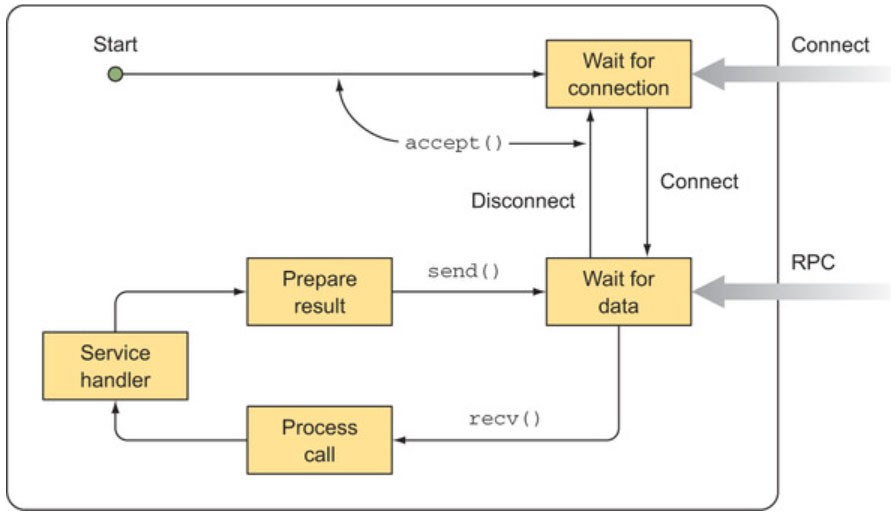
La recepción del mensaje de respuesta indica que el procedimiento remoto se va a ejecutar, pero no devuelve los parámetros resultado.

|  |
| --- |
| F |
| V |
| V |
| F |

El tiempo de ejecución del procedimiento remoto en el lado del cliente depende del tiempo de ejecución del procedimiento remoto en el lado del servidor.

El mensaje de petición debe contener un campo que identifique qué operación remota se desea ejecutar. Es obligatorio que el procedimiento remoto devuelva al menos un resultado.

**Pregunta 4**: Dada la arquitectura de un sistema de RPCs presentada en el esquema siguiente, donde se utiliza TCP para transportar las peticiones y respuestas de un cliente:



Una petición y su respuesta asociada pueden transportarse en diferentes conexiones TCP. Múltiples clientes pueden acceder al mismo procedimiento remoto de manera concurrente.

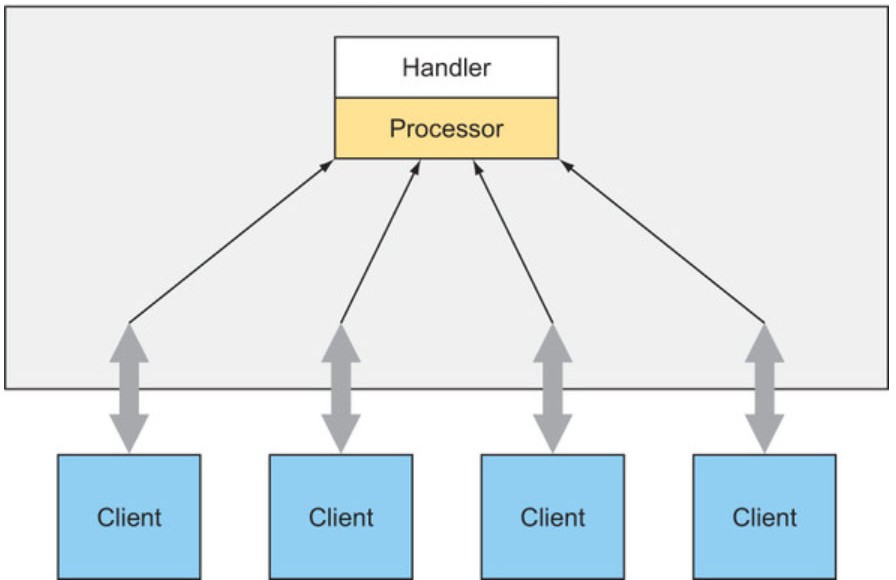
|  |
| --- |
| F |
| F |
| V |
| F |

Un cliente puede invocar consecutivamente a varios procedimientos remotos usando la misma conexión TCP.

Un cliente puede recibir una respuesta que no se corresponda con la última petición realizada a través del mismo flujo TCP.

|  |  |
| --- | --- |
| Apellidos |  |
| Nombre |  |

**Pregunta 5**: Dado un servidor de RPCs de Thrift de tipo TThreadPoolServer con un único procesador de un servicio y un manejador asociado que se presenta en el esquema siguiente:



Es posible la concurrencia de ejecución de procedimientos remotos.

|  |
| --- |
| V |
| F |
| F |
| F |

La compartición del estado del procesador por parte de los clientes impide la concurrencia de ejecución de los procedimientos remotos.

Hasta que no se cierra una conexión TCP de un cliente no se puede establecer la conexión TCP de otro cliente.

Si el manejador es un objeto con un estado que sólo se puede leer, la concurrencia de ejecución de procedimientos remotos producirá una corrupción en el estado dicho manejador.

**Pregunta 6**: En una arquitectura de objetos remotos:

Una referencia a un objeto remoto debe incluir la información de dónde se encuentra dicho objeto remoto.

|  |
| --- |
| V |
| V |
| F |
| V |

Si un método de un objeto invoca a un método de un objeto remoto, dicho método puede incluir el paso de una referencia a otro objeto remoto.

Si un método de un objeto invoca a un método de un objeto local (no remoto) no puede pasarle una referencia a otro objeto remoto.

Un método remoto puede disponer de una interfaz ampliada cuyos métodos pueden ser invocados en modo local.

**Pregunta 7 :**(puntuación máxima 2 puntos) Dado el diagrama de la derecha correspondiente con las capas de un sistema de RPCs de Thrift, indique:

Manejador de servicio

Cliente

Servidor

1. Qué subsistemas son generados automáticamente a partir del IDL.

Procesador

Stub cliente

1. Qué subsistemas se deben codificar.

TBinaryProtocol

TBinaryProtocol

1. Qué subsistemas pertenecen a la biblioteca de Thrift.

TSocket

TSocket

TCP/IP

1. **Stub del cliente:** Se genera automáticamente a partir del IDL, responsable de hacer las llamadas remotas desde el cliente al servidor

**Procesador :**  También se genera automáticamente y maneja las solicitudes entrantes en el servidor,llamando al manejador correspondiente

1. **Manejador de servicio :** Es el código del servidor que se debe implementar para realizar las operaciones necesarias

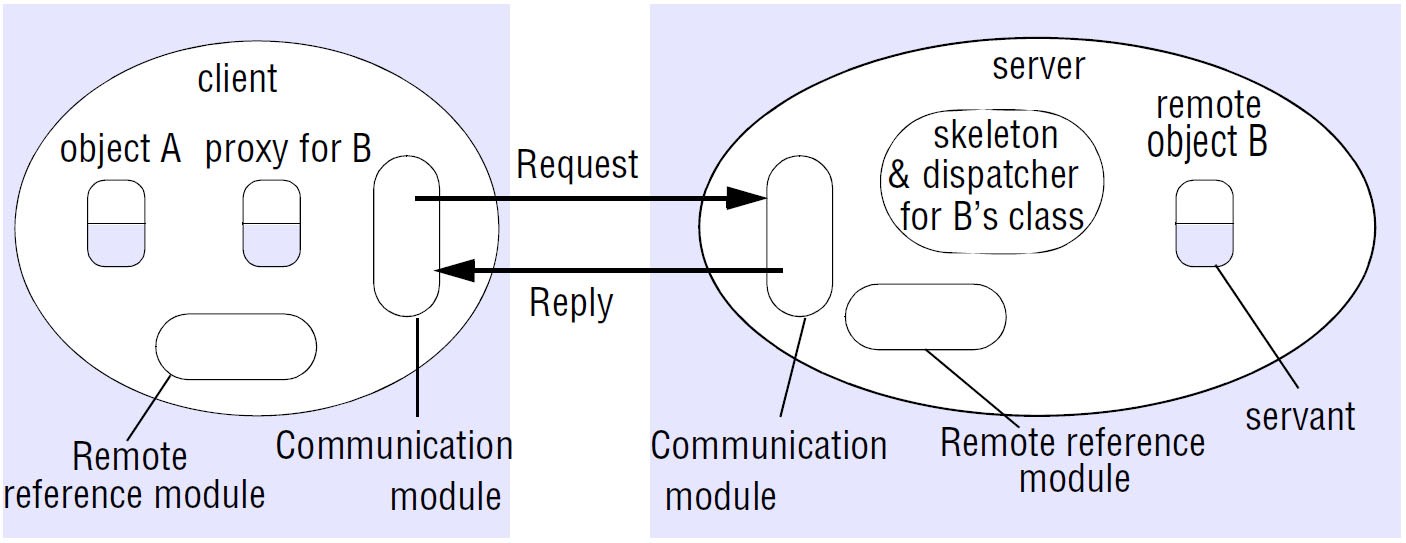
**Cliente :** Debe codificarse para manejar las iteracciones de la aplicación en el lado del cliente

1. **TBinaryProtocol :** Es uno de los protocolos de serialización que viene con el Thrift y es utilizado para codificar los datos que se envían entre el cliente y el servidor

**TSocket :** Se encarga de gestionar las conexiones de red entre cliente y servidor

|  |  |
| --- | --- |
| Apellidos |  |
| Nombre |  |

**Pregunta 8 :**(puntuación máxima 2 puntos) Describa de manera resumida cuál es la funcionalidad de cada uno de los elementos de la arquitectura de objetos distribuidos que aparece a continuación:



**Cliente**

* **Object A** : Es un objeto local del cliente que puede invocar métodos en otros objetos locales o remotos.

En este caso esta intentando invocar un método en el objeto remoto B a través de su proxy.

* **Proxy for B** : Es un intermediario en el lado del cliente , que representa al objeto remoto B.

El proxy actúa como si fuera el objeto remoto, gestionando la serialización de los parámetros y el envio de solicitudes al servidor . Recibe la respuesta y la deserializa antes de enviársela al cliente.

* **Remote reference module :** Este modulo gestiona las referencias a los objetos remotos. Se encarga de crear y resolver las referencias remotas , permitiendo que el cliente pueda acceder al objeto remoto B mediante su proxy.
* **Communication module :** Se encarga del envio de las peticiones al servidor y recibir las respuestas.

Actua como el canal de comunicación entre el cliente y el servidor

**Servidor**

* **Skeleton & Dispatcher for B´s class :** El **esqueleto** es el intermediario en el lado del servidor que traduce las solicitudes recibidas por el cliente y se las pasa al objeto remoto.

El **dispatcher** decide que método invocar en el objeto remoto B, basándose en la solicitud recibida.

* **Remote Object B** : Es el objeto remoto al que el cliente intenta acceder , Contiene los métodos que pueden ser invocados remotamente por el cliente.El cliente interactúa con B como si fuera un objeto local.
* **Servant :** Es el objeto en el servidor que contiene el código real que se ejcuta cuando el cliente invoca un método en el objeto remoto B.
* **Remote Reference Module (servidor) :** Es el modulo en el servidor que gestiona las referencias a los objetos remotos. El modulo traduce las referencia remota en una referencia local para que pueda ser invocada-
* **Communication Module (servidor):** encargado de recibir las solicitudes del cliente y enviar las respuestas de vuelta