Universidad de los Andes > INGENIERÍA

Documento de diseño
Protocolo de comunicación cliente-servidor en capa aplicación

Integrantes: Natalia Espínola - Cristian Funck
Fecha: Martes 05 de noviembre 2019
Profesor: Claudio Álvarez

Tabla de contenido

Introducción

En este documento se presenta la propuesta de diseño de un protocolo de capa aplicación, el cual utilizando socket ip con protocolo TCP de la capa de transporte permitirá la comunicación interprocesos, más específicamente dirigido a la comunicación entre los usuarios y la base de datos relacional de la tarea uno de este curso utilizando una arquitectura cliente - servidor. Los aspectos definidos en este documento de diseño contemplan los requerimientos del protocolo, los tipos de mensaje, los formatos de cada tipo de mensaje, las acciones posibles, los estados posibles y finalmente los manejos de excepción.

Requerimientos

1. Generales

1.1. El protocolo debe permitir comunicación entre los usuarios y la base de datos no relacional de la tarea uno del curso utilizando una arquitectura cliente-servidor pero esta vez sobre sockets de internet, permitiendo así que usuarios de distintas redes ip puedan conectarse a un servidor.

1.2. El protocolo debe permitir al cliente realizar operaciones de lectura y escritura sobre la base de datos.

1.3. El protocolo debe permitir identificar los diferentes estados que puedan resultar de una petición.

2. De conexión

2.1. El protocolo debe permitir establecer una conexión persistente entre cliente y servidor utilizando una sola conexión TCP para enviar y recibir mensajes.

2.2. El protocolo debe permitir establecer conexión para múltiples clientes concurrentes.

2.3. El protocolo debe contar con un puerto dedicado conocido entre cliente y servidor para su uso.

3. De tipos de mensaje

3.1. El protocolo debe contar con un tipo de mensaje que permita enviar peticiones desde un cliente a un servidor.

3.2. El protocolo debe contar con un tipo de mensaje que permita enviar respuestas desde un servidor a un cliente.

4. De formato de mensaje

4.1. Formato de petición

4.1.1. El formato de mensaje de petición debe contar con un campo para especificar la acción de petición.

4.1.2. El formato de mensaje de petición debe contar con un campo para especificar un identificador o clave de dato que será escrito o leído.

4.1.3. El formato de mensaje de petición debe contar con un campo para indicar el tipo de representación de la data que es enviada para escritura.

4.1.4. El formato de mensaje de petición debe contar con un campo para indicar el tipo de representación en que espera sea recibida la data para lectura.

4.1.5. El formato de mensaje de petición debe contar con un campo de contenido para indicar la data que será escrita.

4.1.6. El formato de mensaje de petición debe contar con un campo para indicar la versión usada del protocolo.

4.2. Formato de respuesta

4.2.1. El formato de mensaje de respuesta debe contar con un campo para indicar el mensaje de estado y/o código de estado.

4.2.2. El formato de mensaje de respuesta debe contar con un campo para indicar la versión usada del protocolo.

4.2.3. El formato de mensaje de respuesta debe contar con un campo para indicar el tipo de representación de la data que está siendo enviada en el contenido de la respuesta.

4.2.4. El formato de mensaje de respuesta debe contar con un campo de contenido para indicar la data que está siendo enviada en la respuesta.

4.2.5. El formato de mensaje de respuesta debe contar con un campo para indicar el largo en cantidad de bytes de la data enviada en la respuesta.

5. De acciones

5.1. El protocolo debe contar con alguna acción que permita establecer una conexión persistente entre cliente y servidor hasta que alguno de los host determine cerrarla.

5.2. El protocolo debe contar con alguna acción que permita realizar una desconexión cerrando alguna conexión que haya estado pre establecida entre cliente y servidor.

5.3. El protocolo debe contar con alguna acción que permita realizar la inserción de algún dato en la base de datos enviando solo el dato y asumiendo que el servidor creará una clave generada automáticamente.

5.4. El protocolo debe contar con alguna acción que permita realizar la inserción de alguna dato en la base de datos enviando el dato y la clave asociada.

5.5. El protocolo debe contar con alguna acción que permita leer el valor asociado a una clave en particular.

5.6. El protocolo debe contar con alguna acción que permita reconocer si cierta clave ya está registrada en la base de datos.

5.7. El protocolo debe contar con alguna acción que permita actualizar algún registro de la base de datos proporcionando el nuevo dato y la clave asociada.

5.8. El protocolo debe contar con alguna acción que permita borrar algún registro (el par clave - valor) de la base de datos indicando alguna clave existente.

5.9. El protocolo debe contar con alguna acción que permita conocer el listado de las claves ya registradas en la base de datos.

6. De estados y manejo de excepciones

6.1. El protocolo debe contar con códigos o mensajes de estado que permitan reconocer cuando se produce una espera prolongada y el motivo.

6.2. El protocolo debe contar con códigos o mensajes de estado que permitan reconocer los diferentes motivos cuando una petición es exitosa.

6.3. El protocolo debe contar con códigos o mensajes de estado que permitan reconocer los diferentes motivos cuando se produce un error por parte del cliente

6.4. El protocolo debe contar con códigos o mensajes de estado que permitan reconocer los diferentes motivos cuando se produce un error por parte del servidor.

6.5. El protocolo debe contar con códigos o mensajes de estado que permitan reconocer los diferentes motivos cuando la conexión es interrumpida voluntariamente por el servidor.

6.6. El protocolo debe contar con algún mecanismo de intento de reconexión cuando esta sea interrumpida.

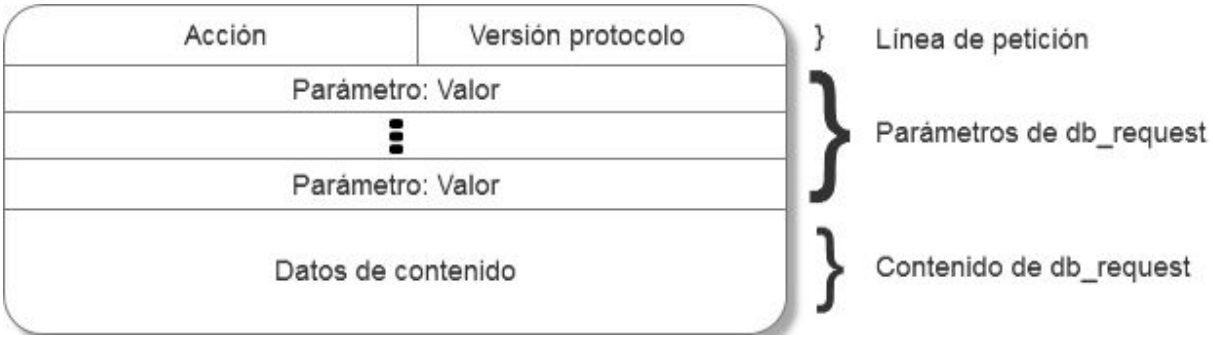
6.7. El protocolo debe contar con algún mecanismo de reenvío de petición cuando no reciba respuesta.

Propuesta de mensajes

Mensajes tipo petición

En este protocolo los mensaje de petición serán llamados db_request

Formato



La línea de petición dentro del un db_resquest considera dos campos importantes, el primero es la acción que se solicita realizar en la base de datos, esta puede ser insert, peek, entre otras acciones (están definidas más adelante en el documento) y el segundo es la

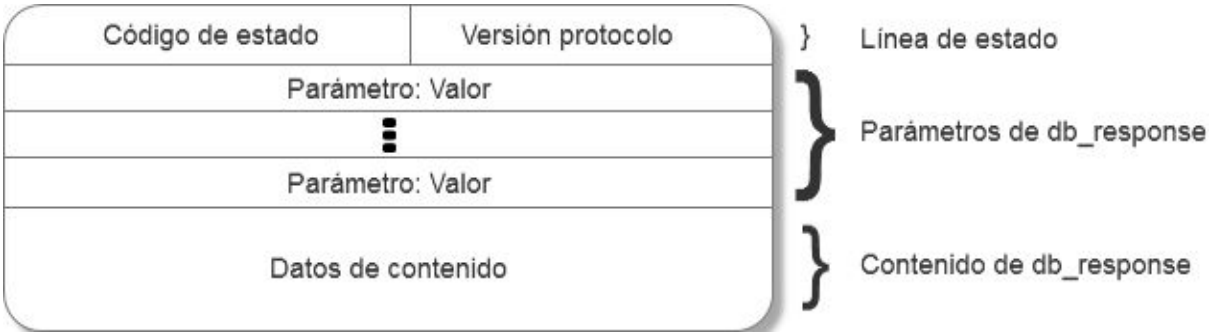
versión utilizada de nuestro protocolo, ambos campos de la línea de petición son siempre obligatorios.

Los parámetros del db_request son opcionales y dependen siempre de la acción que se está solicitando realizar, por ejemplo en el caso de estar realizando una acción peek un parámetro sería “clave” y el valor sería “29840” donde valor es la clave que se consulta a la base de datos si es que está ya asociada a algún registro.

El contenido de db_request será opcional, dependiendo si la solicitud requiere insertar algún dato en la base de datos, en tal caso, es en este campo en donde estará indicado. Los datos podrán estar representados en texto plano o bien en formato binario codificado en base64.

Mensajes tipo respuesta

Formato



La línea de estado dentro de un db_response considera dos campos importantes, el primero es el código de estado, el cual permitirá representar si la petición fue resuelta correctamente o resultó en error por parte del cliente o por parte del servidor, entre otros estados (estos están definidos más adelante en el documento) y el segundo es la versión utilizada de nuestro protocolo, ambos campos de la línea de estado son siempre obligatorios.

Los parámetros del db_response son opcionales y dependen siempre de la acción que fue procesada por el servidor, por ejemplo en el caso de haber procesado una acción peek entonces un parámetro será “clave_existente” y el valor será “verdadero” en caso de ser una clave existente o bien “falso” en caso contrario.

El contenido de db_response será opcional, dependiendo si la solicitud que fue procesada por el servidor requiere retornar los datos de un registro de la base de datos, en tal caso, es en este campo en donde estará indicado. Los datos podrán estar representados en texto plano o bien en formato binario codificado en Base64.

Propuesta de acciones

Connect

Utilizar esta acción permite establecer una conexión persistente con el servidor, de manera que este procesa a crear un client socket ip dedicado a escuchar mensajes de este cliente.

Parámetros db_request:

- **IP_Cliente:** Este parámetro indica la dirección ip del cliente para establecer la conexión por medio de socket**.** Este parámetro es obligatorio.
- **Puerto:** Este parámetro indica el puerto que utiliza el cliente para establecer la conexión por medio de socket**.** Este parámetro es obligatorio.

Contenido db_request:

Esta petición no posee contenido.

Parámetros db_response:

- **IP_Servidor:** Este parámetro indica la dirección ip del servidor para establecer la conexión por medio de socket**.** Este parámetro es obligatorio.
- **Puerto:** Este parámetro indica el puerto que utiliza el servidor para establecer la conexión por medio de socket**.** Este parámetro es obligatorio.

Contenido db_response:

Esta respuesta no posee contenido.

Disconnect

Esta acción permite al cliente desconectarse del servidor.

Parámetros db_request:

La petición no posee parámetros.

Contenido db_request:

La petición no posee contenido.

Parámetros db_response:

La respuesta no posee parámetros.

Contenido db_response:

La respuesta no posee contenido.

Insert

Esta acción permite insertar una nueva tupla clave-valor en la base de datos. Se genera error si la clave ya existe en la base de datos.

Parámetros db_request:

- **Tipo**: Este parámetro representa el tipo de dato que se quiere insertar en la base de datos. Es obligatorio.
- **Largo**: también es obligatorio. Indica el lago del contenido en cantidad de octetos que se va a insertar en la base de datos.
- **Clave**: Este parámetro es opcional. En caso de que no se reciba, se le otorga una clave autogenerada.

Contenido db_request:

Es el Valor a insertar en la base de datos.

Parámetros db_response:

- **Clave**: Este parámetro es obligatorio. Se utiliza para demostrar al cliente que el valor ha sido insertado exitosamente en esta clave.

Contenido db_response:

La respuesta no posee contenido.

Get

Esta petición retorna el valor asociado a la clave recibida.

Parámetros db_request:

- **Clave**: Parámetro obligatorio. Indica la clave asociada al valor que se solicita obtener de la base de datos.

Contenido db_request:

La petición no posee contenido.

Parámetros db_response:

- **Tipo**: Este parámetro es obligatorio. Representa el tipo de dato asociado a la clave solicitada.
- **Largo**: También es un parámetro obligatorio. Indica el lago del contenido en cantidad de octetos.

Contenido db_response:

Lleva el valor asociado a la clave solicitada.

Peek

Esta acción permite conocer si una clave está registrada en la base de datos. retorna “verdadero” si la clave está registrada y “falso” en caso contrario.

Parámetros db_request:

La petición no posee parámetros.

Contenido db_request:

La petición no posee contenido.

Parámetros db_response:

- **clave_existente**: Es un parámetro obligatorio. El valor será “verdadero” en caso de ser una clave existente o bien “falso” en caso contrario

Contenido db_response:

La respuesta no posee contenido.

Update

Utilizar esta acción permite actualizar un registro de la base de datos, en particular actualizar el valor asociado a la clave indicada.

Parámetros db_request:

- **Clave:** Este parámetro indica la clave asociada al registro que se solicita actualizar en la base de datos**.** Este parámetro es obligatorio.
- **Tipo:** Este parámetro indica el formato de representación del contenido, el cual puede ser texto plano o binario**.** Este parámetro es obligatorio.
- **Largo:** Este parámetro indica el tamaño del contenido expresado en cantidad e bytes**.** Este parámetro es obligatorio.

Contenido db_request:

El contenido es el valor que se utilizará para actualizar el registro de la base de datos asociado a la clave indicada.

Parámetros db_response:

La respuesta no posee parámetros.

Contenido db_response:

La respuesta no posee contenido.

Delete

Utilizar esta acción permite eliminar un registro de la base de datos. (eliminar una par clave-valor)

Parámetros db_request:

- **Clave:** Este parámetro indica la clave asociada al registro que se solicita eliminar de la base de datos**.** Este parámetro es obligatorio.

Contenido db_request:

La petición no posee contenido.

Parámetros db_response:

- **Clave:** Este parámetro indica la clave que se intentó eliminar y es retornado en una respuesta del servidor cuando no existe un registro asociado a la clave. Este parámetro es opcional.

Contenido db_response:

La respuesta no posee contenido.

List

Utilizar esta acción en un mensaje de solicitud db_request permite obtener una respuesta db_response con un listado de todas las claves asociadas a registros existentes en la base de datos.

Parámetros db_request:

La petición no posee parámetros.

Contenido db_request:

La petición no posee contenido.

Parámetros db_response:

- **Tipo**: Este parámetro indica el formato de representación del contenido, por ejemplo este podría ser texto plano o binario. Este es un parámetro obligatorio.
- **Largo**: Este parámetro indica el tamaño en cantidad de bytes del contenido. Este parámetro es obligatorio.

Contenido db_response:

El contenido de una respuesta a la acción List es la lista de claves asociadas a registros existentes en la base de datos.

Propuesta de estados

De peticiones exitosas

El rango reservado para peticiones exitosas es del 0 al 9.

- 0: OK , respuesta estándar para acción exitosa.
- 1: Conexión exitosa.
- 2: desconexión exitosa con el servidor.
- 3: Valor insertado exitosamente.
- 4: Actualización exitosa.

De error de parte del cliente

El rango reservado para errores de parte del cliente es del 10 al 19.

- 10: timeout, el cliente ha superado el tiempo máximo de 10 segundos para conectarse a un servidor.
- 11: IP_Cliente ingresado erróneo.
- 12: Puerto ingresado erróneo.
- 13: Clave ingresada ya existe en la base de datos al tratar de insertar dato.
- 14: Clave no registrada en la base de datos al tratar de obtener un dato.
- 15: Clave no registrada en la base de datos al tratar de actualizar.
- 16: request_timeout, el cliente ha dejado de utilizar la conexión, por lo tanto el servidor envía una respuesta indicando que cierra la conexión.
- 17: Acción ingresada inválida.
- 19: Error no determinado.

De error de parte del servidor

El rango reservado para errores de parte del servidor es del 20 al 29.

- 20: Error interno al servidor, falla de la base de datos.
- 21: El servidor no soporta la funcionalidad necesaria para responder a la solicitud del cliente.
- 22: El servidor está congestionado o realizando otras tareas, por lo que no puede responder a la solicitud del cliente.
- 23: Desconexión fallida.
- 29: Error no determinado.

Manejo de situaciones de excepción

- Conexión interrumpida:
  En caso de que se pierda la conexión es responsabilidad de la aplicación que implemente el protocolo realizar intentos de reconexión.

- Tiempo de espera de respuesta excedido:
  Debido a que los tiempo de respuesta de una base de datos es muy variable en función de la cantidad de solicitudes concurrentes a la misma, la responsabilidad de definir un tiempo de respuesta y de manejar la superación de este tiempo es de la aplicación que implemente el protocolo.