Documentación Iteración 5

Universidad de los Andes, Bogotá, Colombia
{hs.hernandez,a.gracia10,js.beltran14,ks.estupinan,ma.puentes,am.valero10}@uniandes.edu.co

Tabla Contenido

1. Análisis de la implementación de transacciones distribuidas

1.1 Implementación de transacciones distribuidas XA y colas de mensaje.

Las transacciones distribuidas usando la interfaz XA de X/OPen es un gestor de transacciones global y administrador de recursos, que preserva las propiedades ACID en sus transacciones, esta interfaz implementa el protocolo 2 Phase Commit(2PC), en este caso la primera restricción en nuestra aplicación es la necesidad de que los recursos se encuentren comprometidos y siempre disponibles,ya que en el caso que algún participante no esté disponible o por algún motivo no responda en alguna de las fase del 2PC, la transacción será revertida. Otra restricción es la necesidad de que las 3 Rotondas utilicen este protocolo de comunicación e implementan la interfaz de XA y una última restricción es que las transacción tienen que ser respondidas en un tiempo estimado y considerado para las consultas, ya que este tipo de protocolo al trabajar de manera sincrónica necesita un timeout para preservar las fases del 2PC, en el caso de que en el pre-commit o en la primera fase no responda la pregunta de poder hacer commit, como en la segunda fase de la solicitud de hacer el commit, lo cual requiere que las consultas se hagan en un tiempo estimado y optimo, para no comprometer el 2PC y este finalice correctamente.

Las transacciones distribuidas con colas de mensajes pueden tener restricciones de un tiempo de respuesta adecuado para la solicitud de los requerimientos de la rotonda, ya que aunque este tipo de protocolo trabaja de manera asincrónica, si es necesario un tiempo de respuesta adecuado para la solicitud de algún tipo de producto del restaurante o la información de cierto restaurante. Otra restricción podría ser la de dar cierta prioridad en la solicitudes a determinado restaurante, ya sea porque la solicitud fue realizada en un determinado restaurante y este posee en sus menús los productos solicitados o porque hay algún tipo de favoritismo del cliente hacia determinada rotonda.

a. Mismo valor para las constantes en las 3 aplicaciones:
   i. GLOBAL_TOPIC_NAME
   ii. LOCAL_TOPIC_NAME
   iii. TIME_OUT
   iv. REQUEST
   v. REQUEST_ANSWER
   vi. Modo AUTO_ACKNOWLEDGE

b. Para RF18:
   i. Si el TIME_OUT es violado, se manda un mensaje a la cola de respuesta informando del problema y se hace rollback de la transacción local.

c. Para RF19
   i. La cadena debe existir en las 3 aplicaciones.

ii. Cada aplicación recibe la cadena de restaurante que se desea retirar.
    iii. Utilizando colas de mensaje
        1. Siempre se debe retorna una respuesta positiva.
    iv. Utilizando Two-phase Commit
        1. Si todas las Rotondas están lista para retirar una cadena, se retira de todas las Rotondas. En caso de que una no esté lista o presente algún error, no se retira ninguna Rotonda.

d. Para RFC13

i. Para este requerimiento se va a utilizar Two-phase Commit dado que es necesario mostrar la información de las 3 Rotondas o en caso contrario mostrar mandar un mensaje de error.

e. Para RFC14

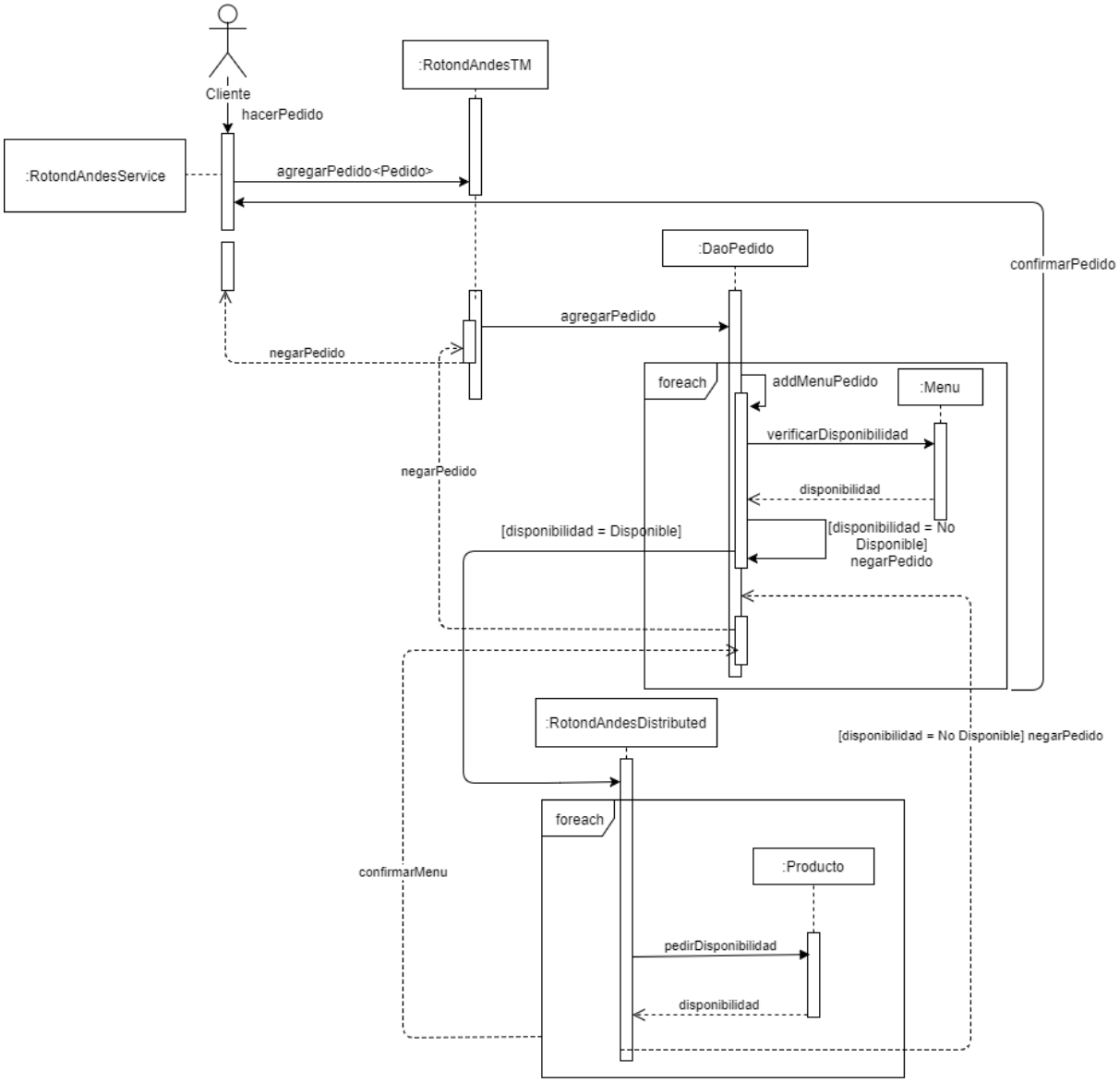
i. Para este requerimiento se utilizará Two-phase Commit dado que es necesario que la información debe ser completa en cuanto a facturación de todas las Rotondas. En caso de que se presente un error se hace RollBack de todas las bases de datos de las Rotondas.

1.2

1.2.1 SubGrupo C-07

- Se creó el nuevo atributo de Cadena (Varchar 30 NN) en la tabla de Restaurante, con el fin de facilitar la implementación del requerimiento FR19.
- Se crearon los packeages de StartUp, JMS y DTM para el manejo de transacciones distribuidas.

1.3



1.4 Analisis requerimientos para casos de colas de mensaje y Two Phase Commit.

RF18 - REGISTRAR PEDIDO DE PRODUCTOS A UNA MESA: Al realizar este requerimiento con el protocolo de colas de mensajes, al momento de realizar la solicitud esta podría verse afectada por el tiempo de respuesta entregada por los servidores, ya que al no trabajar sincrónicamente ellos podrían no estar disponibles para realizar el registro del pedido a la mesa de manera oportuna, y ademas que al no requerir que los servidores se encuentren conectados al momento de la peticion, incumpliria con una de las restricciones del requerimiento, la cual es la de que si algun restaurante no puede confirmar de forma inmediata la disponibilidad de algún

producto, la solicitud quedaría en estado rechazada. Para este requerimiento es mejor implementar el protocolo Two Phase Commit(2PC) ya que este para su correcto funcionamiento compromete los recursos involucrados en la transacción, en este caso los restaurantes para el registro de varios productos necesitan estar disponibles en el momento de la transacción, además que este protocolo asegura la integridad de la transacción y sus sub-transacciones de una manera sincrónica y esto nos permite tener un tiempo de respuesta adecuado para cumplir a cabalidad el requerimiento.

RF19 - RETIRAR RESTAURANTE: Este requerimiento al ser implementado con colas de mensaje cumpliria con la solicitud de este, ya que este tipo de protocolo es el indicado y podría ser resuelto en un tiempo prudencial, más no necesariamente en tiempo corto o con restricción de timeout, al trabajar asincrónicamente la solicitud de la cola puede ser resuelta cuando el servidor esté disponible y cumpla con la solicitud del requerimiento de retirar un restaurante y sus productos asociados. Por otra parte, la implementación del protocolo Two Phase Commit(2PC) también servirá para cumplir con las condiciones del requerimiento asegurando su integridad, específicamente donde se debe dar garantía de que se deben eliminar los restaurantes de todas las rotondas, por tal razón, es mejor utilizar el protocolo two phase commit, pues si se logra ejecutar correctamente, nos da la garantía de que efectivamente el requerimiento se cumplió, o de lo contrario se haría un rollback global de la transacción.

RFC13. CONSULTAR LOS PRODUCTOS DISPONIBLES EN ROTONDANDES: Al implementar en este requerimiento con el protocolo de colas de mensaje podría responder con la solicitud del requerimiento de forma correcta, pero en el momento que los servidores se conecten y respondan la solicitud, lo cual para la consulta es de vital importancia la extracción de la información de los productos disponibles de los tres restaurantes en el momento en el que los usuarios realicen una consulta sobre estos productos, por esta razón es
mejor implementar el protocolo Two Phase Commit que necesita que los servidores se encuentren conectados en el momento de realizar la consulta, lo cual permite en un tiempo óptimo la extracción de la información y además compromete a los servidores para responder esta solicitud al tener que trabajar de manera sincrónica.

RFC14. CONSULTAR LA RENTABILIDAD DE UN RESTAURANTE – RFC5 v2: Este requerimiento al ser implementado con colas de mensajes cumpliria con la solicitud pero solo los servidores o restaurantes conectados responderían con esta solicitud o la responderían cuando se encontraran conectados, por este motivo es mejor la implementación del protocolo Two Phase commit, ya que el usuario-Restaurante que realice la consulta necesita que la solicitud sea respondida y que su rotonda se encuentre disponible para la solicitud, además que la transacción necesita los valores facturados por cada rotonda participante,por este motivo es

necesario un protocolo que comprometa los servidores para que la respuesta sea entregada con toda la información especificada en el requerimiento.

2. Especificación e implementación de transacciones distribuidas

2.1. Este requerimiento fue cumplido por cada subgrupo.

2.2.

Para RF18:

i. Dada la propiedad asincrónica de las colas de mensaje se facilita en la agilidad para el cliente el hecho de hacer un pedido teniendo en cuenta que la Rotonda que responda primero es quien atiende el pedido. El hecho de aplicar Two-phase Commit implica que todos los restaurantes deben estar listos a la hora de recibir un pedido ralentizando el proceso.

Para RF19

i. Teniendo en cuenta que la cadena de restaurantes debe existir en las tres rotondas, las colas de mensajes podrían ser una solución viable para responder a este requerimiento, pues al trabajar asincrónicamente estas podrían resolver las solicitudes cuando el servidor esté disponible, ya que el mensaje queda en la cola de solicitudes. Sin embargo, al necesitar la garantía de que el restaurante sea retirado de todas las rotondas, la mejor opción es utilizar el protocolo 'two phase commit', porque nos permite saber cuándo el requerimiento ha sido cumplido para todas las rotondas, ya que si se logra realizar correctamente, no daría garantía de que efectivamente fueron eliminadas de las tres rotondas, de lo contrario, es decir, si hubo algún tipo de fallo, se realiza un 'RollBack' global.

Para RFC13

i.Para este requerimiento es necesario utilizar Two-phase Commit dado que se vuelve fundamental mostrar mostrar la información de las 3 Rotondas o en caso contrario mostrar mandar un mensaje de error. Esto con el fin de no excluir los productos servidos en una Rotonda. Un contra de usar 2PC es el tiempo de demora en la respuesta por su propiedad sincronica.

Para RFC14

i.Dada la especificación del método, es necesario que todas las Rotondas brinden la información de cada restaurante para poder responder a la solicitud del usuario. Por la propiedad sincronica de 2PC esto se cumple en cualquier caso.