**LÓGICA DEL PROCESO DEL RF15:**

Se consume desde Postman el servicio *Registrar la compra de un abonamiento (POST)* el cual es atendido por la clase ClienteResource que es la que recibe todas las solicitudes REST. A continuación esta llama al Transaction Manager para que lleve a cabo este proceso y lo que hace es ejecutar primero el requerimiento para la unidad de negocio local, es decir llama el método realizado para el RF11 el cual realiza el abonamiento sobre la base de datos con la que se hicieron las anteriores iteraciones. Luego se procede a llamar el método correspondiente al requerimiento pero ahora de la case FestivAndesDitributed que se encarga de ejecutar el requerimiento para las otras dos unidades de negocio, esto lo hace usando la clase específica creada para este requerimiento que recibe el nombre de RegistrarAbonamientoMDB la cual tiene un método que utiliza colas de mensajes como mecanismo de comunicación implementando la interface MessageListener. En cada una de las etapas se verifica que exista al menos una entrada disponible para todas las funciones sobre las cuales se desea registrar el abonamiento.

**ANÁLSISI DE POSIBLES RESULTADOS:**

Ya se describieron las estrategias requeridas para implementar RF15 con el método Colas de mensajes, ahora procedemos a describir como sería con Two phase Commit y compararemos resultados.

Si desarrolláramos este requerimiento utilizando Two Phase Commit deberíamos incluir un Java Naming Directory Interface mediante el cual tuviéramos acceso desde un componente a otros componentes. También deberíamos tener en cuenta que se requiere de un coordinador que realice la petición del commit a los participantes.

En este caso los resultados cambiarían en tiempo porque el coordinador debería esperar que las otras dos bases de datos realicen las consultas y respondan mientras que con colas de mensajes no es así, sería más demorado el procesamiento de los datos y la integración de resultados y el diagrama de secuencias cambiaría. La mayor desventaja del protocolo commit en dos fases es el hecho de que es un protocolo bloqueante. Un nodo estará bloqueado mientras esté esperando un mensaje por lo tanto, como lo sugiere la guía, para este requerimiento es mejor usar colas de mensajes

Para el RF16 si se implementa con colas de mensajes tenemos básicamente la misma estrategia que usamos para el RF15 pero ahora lo que cambian son las sentencias sobre las unidades de negocio. Para este caso los resultados no variarían si se usara 2PC o si se usara colas de mensajes debido a que en el requerimiento no influye la espera entre recursos, simplemente consiste en eliminar de la base de datos la compañía y todas las funciones asociadas a esa compañía en las tres unidades de negocio.

Para los dos requerimientos de consulta RFC13 y RFC14 sí habría una gran diferencia en su implementación con colas de mensajes y con 2PC. Si se realizara con 2PC tendríamos el problema de bloqueo, mientras que con colas de mensajes los clientes no se bloquean mientras producen contenidos; los servidores son notificados de manera asíncrona de los nuevos contenidos mientras realizan otra actividad. La comunicación asíncrona que permite las colas de mensajes produce mejores resultados para estos requerimientos ya que puede realizar la tarea en cada una de las unidades lógicas de manera simultánea y debido a que estas consultas tienen bastantes accesos a las bases de datos, el modelo de colas de mensajes agilizaría el proceso.