

Funcionalidades de Base de Datos Orientada a Servicios en Microsoft SQL Server 2005

Gustavo Larriera
Microsoft MVP
gux@mvps.org

Resumen

Este artículo introduce los conceptos básicos de “arquitectura de base de datos orientada a servicios” implementados en SQL Server 2005. La “base de datos orientada a servicios” expone lógica de servicios asociada a mensajes procesados asincrónicamente, notificaciones de resultados de consultas realizadas, eventos generados en los datos, etc. Este comportamiento en una base de datos convierte a los datos en “datos activos” y permite integrar datos y lógica de la base de datos, dentro del contexto de una arquitectura orientada a servicios. Se muestran a nivel introductorio en este artículo, algunas funcionalidades de SQL Server 2005, como XML Web Services, Service Broker y Query Notifications.

1. Introducción

Las bases de datos tradicionalmente han sido utilizadas en el desarrollo de sistemas cliente/servidor orientados a sesiones, donde el programa cliente procesa la entrada y los eventos, y la base de datos actúa como un repositorio pasivo de información que es dirigida desde y hacia el programa cliente. En muchos casos, observamos que hay poca o nula asociación entre la lógica de datos y los datos en sí mismos. Una primera evolución fue en su momento acercar la lógica a los datos, mediante código en la propia base de datos (e.g. implementado en procedimientos almacenados y triggers).

Para hacer transparente el acceso a los datos y la lógica, los sistemas administradores de bases de datos (Database Management Systems, DBMS) ofrecen una variedad de librerías de programación como OLE-DB/ODBC, JDBC, ADO.NET, etc. Estas componentes habilitan la integración y reutilización de funcionalidades pero requieren una plataforma que soporte todos o algunos de los protocolos indicados para cada componente.

2. Datos orientados a servicios

En el desarrollo actual de sistemas distribuidos, la Arquitectura Orientada a Servicios (Service Oriented Architecture, SOA) es un estilo arquitectural cuya meta principal es lograr un acoplamiento escaso entre agentes de software, o servicios, que interactúan entre sí. Un servicio es una unidad de trabajo realizada por un proveedor de servicios, que logra los resultados finales deseados por el consumidor del servicio.

Dentro del contexto de SOA resulta de interés preguntarse cuál es el “nuevo rol” de los DBMS. Los DBMS están evolucionando; de ser sistemas pasivos donde se depositan datos, están pasando a ser agentes activos, que además de proveer acceso a datos relacionales, también deben integrarse con otras componentes y ser capaces de lidiar con información poco estructurada, mensajes, XML, protocolos Web, eventos, entre otros.

La “base de datos orientada a servicios” expone lógica de servicios asociada a mensajes procesados asincrónicamente, notificaciones de resultados de consultas realizadas, eventos generados en los datos, etc. La lógica de servicios puede ser disparada por diversas situaciones como son los cambios de valores en los datos, por la recepción de mensajes específicos o simplemente porque se llegó a un límite de tiempo prefijado. Este comportamiento en una base de datos convierte a los datos en “datos activos”.

Un DBMS es un lugar ideal para implementar funcionalidades que le permitan integrarse en forma flexible y simple en una arquitectura orientada a servicios. Para cumplir con el rol adecuado dentro de SOA, el DBMS debe disponer nativamente de las siguientes funcionalidades [Cam05]:

- Web services. El DBMS debe soportar diferentes tipos de “endpoints” (e.g. sockets TCP, HTTP GET/PUT, SOAP) que permitan el acceso a datos y la ejecución remota de lógica en la base de datos. Especialmente debe disponer de endpoints para web services.

- Conversaciones. El DBMS debe poder leer y procesar mensajes (comúnmente XML), participar en diálogos y conversaciones complejas.
- Lógica. El DBMS debe ser host de lógica de programación compleja y moderna, equivalente a la de cualquier servidor de aplicaciones (e.g. pooling, activación, escalabilidad de la lógica de proceso).

Un arquitectura de DBMS diseñada teniendo en cuenta los puntos anteriores, la denominamos “arquitectura de base de datos orientada a servicios” (Service Oriented Database Architecture, SODA). Microsoft SQL Server 2005 [Dum05] es un ejemplo de RDBMS construido en base a una arquitectura tipo SODA. Las principales características SODA implementadas en Microsoft SQL Server 2005 se describen a continuación.

3. Soporte nativo para Web Services en SQL Server 2005

SQL Server XML Web Services permiten la llamada remota a lógica de la base de datos mediante acceso estándar SOAP nativo en el servidor. Para ello la aplicación SQL Server 2005 se registra al “HTTP listener”, un nuevo modo del kernel de Windows Server 2003. Una vez registrado, los requests http son despachados al servicio adecuado. Esta forma de trabajo no requiere el uso de un servidor web como Internet Information Services.

Los administradores definen los endpoints SOAP y asocian métodos (i.e. WebMethods) al endpoint. Los métodos invocados pueden ser procedimientos almacenados programados en T-SQL o en cualquier lenguaje .NET

```
-- creo base para pruebas
create database testws
go
use testws
go
-- hola mundo: un sproc simple
create procedure hola_mundo
(@inmsg nvarchar(256))
as
begin
select 'hola mundo ' + @inmsg
end
-- creacion de un endpoint de acceso
use master
go
create endpoint hola_mundo_endpoint
state = started
```

```
as http (
authentication = ( integrated ),
path = '/sql/demo',
ports = ( clear )
)
for soap (
webmethod
'http://tempuri.org/'. 'hola_mundo'
(name = 'testws.dbo.hola_mundo'),
batches = enabled,
wsdl = default
)
-- permisos de acceso a traves del endpoint
grant connect on
endpoint::hola_mundo_endpoint to
[contoso\administrator]
```

Cada servicio expuesto puede generar automáticamente su contrato vía Web Service Description Language (WSDL), lo cual define un mecanismo RPC basado en estándares, con mecanismos modernos de autenticación para el cliente SOAP-compatible. Para solicitar el WSDL en el ejemplo mostrado, se debe acceder de la forma estándar:

```
http://servername/sql/demo?wsdl
```

4. Service Broker

El Service broker es un middleware transaccional integrado en el motor de SQL Server 2005, que define nuevos objetos en la base de datos (e.g. servicios, colas, contratos y mensajes) y brinda un ambiente de ejecución concurrente asincrónico, con manejo de estado, desacoplado y distribuido. Los administradores disponen de T-SQL específico, con sentencias como BEGIN/END DIALOG, SEND y RECEIVE.

Todas las operaciones de Service Broker tienen lugar en el contexto de una transacción de base de datos, lo cual preserva la integridad transaccional de la mensajería [Wol05]. En el siguiente ejemplo se muestra cómo usar en forma simple al Service Broker.

```
-- service broker simple: Hola Mundo
create database testsb
go
use test
go
-- creo dos colas de mensajes
create queue targetq
go
create queue initq
go
```

```

-- creo dos servicios
create service targetservice on queue
targetq ([DEFAULT])
go
create service initSERVICE on queue initq
go
-- establezco una conversación
declare @h uniqueidentifier
begin dialog @h
    from service initSERVICE
    to service 'targetSERVICE'
    with encryption = off;
send on conversation @h ('Hola Mundo SB')
-- consulto el servicio
select * from targetq
-- recibo mensaje del servicio 2
declare @m varchar(max)
declare @h uniqueidentifier;
receive top(1)
    @m = convert(varchar(max), message_body),
    @h = conversation_handle
from targetq;
print @m;
send on conversation @h ('Hola')
end conversation @h
-- recibo mensaje del servicio 1
declare @m varchar(max)
declare @h uniqueidentifier;
receive top(1)
    @m = convert(varchar(max), message_body),
    @h = conversation_handle
from initq;
print @m;
end conversation @h

```

5. Notificaciones de consultas

Esta nueva funcionalidad de SQL Server 2005 permite a un programa requerir una notificación del servidor cuando los resultados de una consulta cambian, lo que habilita a consultar a la base de datos solamente cuando se ha producido un cambio en determinado dato (versus hacer consultas periódicamente para ver si el dato ha cambiado). Esta modalidad reduce las las idas del programa hacia la base de datos. Los pasos que se deben realizar son: a) El programa envía una consulta al servidor SQL y solicita una notificación; b) El programa guarda en caché el resultado de la consulta; c) Cuando el programa recibe la notificación de la consulta, limpia el contenido del caché; d) El programa entonces vuelve a enviar la consulta y el pedido de notificación cuando necesita los resultados refrescados.

El motor de SQL Server 2005 utiliza un mecanismo de suscripciones de notificaciones para hacer el seguimiento de las notificaciones de consultas. Cuando un comando contiene una solicitud de notificación, la base de datos registra el pedido como una suscripción y ejecuta el comando. Para entregar los mensajes XML de notificación, SQL Server 2005 utiliza al Service Broker. Desde el punto de vista de la programación cliente, el siguiente código C# utiliza ADO.NET 2.0 para suscribirse a una notificación de una consulta [Cho04, Bea05]:

```

using System;
using System.Data;
using System.Data.SqlClient;
class dbnotif
{
    public static string connectionstring =
"server=SERVER1;          database=PUBS;
trusted_connection=YES";
    public void HacerDependencia()
    {
        using (SqlConnection conn = new
SqlConnection(connectionstring))
        {
            conn.Open();
            Console.WriteLine("Connexión
abierta...");

            SqlCommand cmd =
conn.CreateCommand();
            cmd.CommandText = "SELECT title,
price FROM dbo.titles";

            // código específico de la
notificación
            SqlDependency dep = new
SqlDependency(cmd);
            dep.OnChange += delegate(Object
o, SqlNotificationEventArgs args)
            {
                Console.WriteLine("Ocurrió
un evento!");
                Console.WriteLine("Info:" +
args.Info);
                Console.WriteLine("Origen:"
+ args.Source);
                Console.WriteLine("Tipo:" +
args.Type);
            };

            SqlDataReader r =
cmd.ExecuteReader();
            // leo acá los datos y cierro el
datareader

```

```

        r.Close();
        Console.WriteLine("DataReader
leyó...");
    }
}

public static void Main()
{
    try
    {
        // inicio la escucha del cliente

SqlDependency.Start(connectionstring);
        dbnotif q = new dbnotif();
        q.HacerDependencia();
        Console.WriteLine("Esperando
evento de notificación...");
        Console.WriteLine("Pulsar
ENTER...");
        Console.ReadLine();
    }
    finally
    {
        // limpieza (opcional)

SqlDependency.Stop(connectionstring);
    }
}
}

```

Cuando se ejecuta un comando asociado a un objeto `SqlDependency`, solicita al servicio `SqlQueryNotificationService` para ser notificado si se produce un cambio en el resultado del comando. Si se produce un cambio en sus datos, el servicio pone un mensaje de notificación en una cola y el servidor SQL envía un mensaje desde la cola hacia el cliente. Entonces el cliente hace el call-back del evento adecuado.

6. Conclusiones

SQL Server 2005 dispone de múltiples protocolos y tecnologías de acceso a datos, no solamente basadas en OLE-DB/ODBC, JDBC y soporte nativo XML/XQuery, sino que además brinda soporte para integrarse nativamente en una arquitectura orientada a servicios (SOA). Esta integración está implementada en Web Services, servicios de notificación, servicios de mensajería, eventos y colas, todos nativamente implementados en el producto. Todas estas funcionalidades que componen la arquitectura de base de datos orientada a servicios de SQL Server 2005, habilitan fluidamente la integración de las bases de datos dentro del contexto de SOA.

7. Referencias

- [Bea05] B. Beauchemin. Query Notifications in ADO.NET 2.0. Developmentor. Abril 2005.
- [Cam05] D. Campbell. Service Oriented Database Architecture: App Server-Lite? Microsoft Research Technical Report MSR-TR-2005-129. Junio 2005.
- [Cho04] S. Chordia. Notification Support in SqlClient Managed Provider. Weblog del autor. Diciembre 2004.
- [Dum05] M. Dumler. Microsoft SQL Server 2005 Product Overview. Microsoft. Abril 2005.
- [Olo05] C. W. Olofson. Putting Enterprise Information to Work with Microsoft SQL Server 2005. IDC Whitepaper. Septiembre 2005.
- [SS04] B. Sarsfield, Raghavan, S. Overview of Native XML Web Services for Microsoft SQL Server 2005. Microsoft. Marzo 2004.
- [Wol05] R. Wolter. Building Reliable, Asynchronous Database Applications Using Service Broker. Microsoft. Febrero 2005.