Is it possible to share a transaction between a .Net application and a COM+ object?

Asked 16 years, 4 months ago Modified 11 years ago Viewed 1k times

4

I did some tests a while ago and never figured out how to make this work.

The ingredients:

- COM+ transactional object (developed in VB6)
- .Net web application (with transaction) in IIS that...
  makes a call to the COM+ component
  updates a row in a SQL database

Testing:

Run the .Net application and force an exception.

Result:

The update made from the .Net application rolls back.
The update made by the COM+ object does not roll back.

If I call the COM+ object from an old ASP page the rollback works.

I know some people may be thinking "what?! COM+ and .Net you must be out of your mind!", but there are some

places in this world where there still are a lot of COM+ components. I was just curious if someone ever faced this and if you figured out how to make this work.

.net database transactions com+

Share

Improve this question

Follow

edited Dec 2, 2013 at 13:37

e-sushi

14.2k ● 10 ● 39 ● 57

asked Aug 22, 2008 at 1:39

bubbassauro

4,249 ● 2 ● 48 ● 47

2 Answers

Sorted by: Highest score (default)

2

Because VB and .NET will use different SQL connections (and there is no way to make ADO and ADO.NET share the same connection), your only possibility is to enlist the DTC (Distributed Transaction Coordinator). The DTC will coordinates the two independent transactions so they commit or are rolled-back together.

From .NET, EnterpriseServices manages COM+ functionality, such as the DTC. In .NET 2.0 and forward, you can use the System.Transactions namespace, which makes things a little nicer. I think something like this should work (untested code):

```
void SomeMethod()
{
    EnterpriseServicesInteropOption e = EnterpriseServ
    using (TransactionScope s = new TransactionScope(e
    {
        MyComPlusClass o = new MyComPlusClass();

        o.SomeTransactionalMethod();
    }
}
```

I am not familiar enough with this to give you more advice at this point.

On the COM+ side, your object needs to be configured to use (most likely "require") a distributed transaction. You can do that from COM+ Explorer, by going to your object's *Properties*, selecting the *Transaction* tab, and clicking on "*Required*". I don't remember if you can do this from code as well; VB6 was created before COM+ was released, so it doesn't fully support everything COM+ does (its transactional support was meant for COM+'s predecessor, called MS Transaction Server).

If everything works correctly, your COM+ object should be enlisting in the existing Context created by your .NET code.

You can use the "Distributed Transaction Coordinator\Transaction List" node in "Component Services" to check and see the distributed transaction being created during the call.

Be aware that you cannot see the changes from the COM+ component reflected on data queries from the .NET side until the Transaction is committed! In fact, it is possible to deadlock! Remember that DTC will make sure that the two transactions are paired, but they are still separate database transactions.

Share Improve this answer

Follow

answered Sep 15, 2008 at 20:17

Euro Micelli

33.9k ● 8 ● 53 ● 72

1

How are you implementing this? If you are using EnterpriseServices to manage the .NET transaction, then both transactions should get rolled back, since you're using the same context for them both.

Share Improve this answer

Follow

answered Aug 22, 2008 at 1:42

Eric Z Beard

38.4k ● 27 ● 101 ● 147