

## *Getting the Most Out of U2 Connection Pooling*

BRUNEL: Welcome to another edition of Getting the Most Out of IBM U2. My name is Kenny Brunel, and I'm your host for today's episode. And today we're going to talk about connection pooling for U2: what is it, why do you need it, and what's the best application for your needs.

With me in the studio today, I have several guests. First of all, [Elizabeth Schmitt Thomas.] Elizabeth manages contracts and policies for U2 worldwide and she's been with the U2 group for over 12 years.

Dave Peters is my next guest, and he is the product manager for the IBM U2 data servers and also an IBMer for 12 years.

And my next guest, [Michael Byrne]. Michael has been involved with U2 for over 10 years and currently works as an IT specialist.

So, first of all, what is connection pooling? And Elizabeth, I'm going to come to you first. Can you give us an explanation of what connection pooling is?

THOMAS: Well, Kenny, connection pooling is a technique that allows multiple clients to send requests through a server to a shared set or a pool of persistent database connections. Connection pooling's generally used to improve performance.

## *Getting the Most Out of U2 Connection Pooling*

Once a connection's created, it's placed in a pool and can be reused, eliminating the overhead of establishing a brand new connection. If all the connection pools are being used, a new connection can be created and then added to that pool.

Connection pooling is very often done with Web applications, but it could be used with any environment that requires many short-term connections to a data server. Any of the connection pooling technologies, whether it's IBM's technology if you develop your code in house or even use third party code must be properly licensed. This is explicitly stated in our OEM transaction document, and in some cases it's in the actual license agreement that ships with our products.

BRUNEL:       Okay. Thank you, Elizabeth. So now I'd like to find out, why would someone use connection pooling? What do you gain from using it? And Michael, can you take that question?

BYRNE:        Sure, Kenny. I think in general the big bonus to connection pooling is it's obviously very expensive to open your database connections and your overhead going along with starting a session and the amount of time that that actually takes on the server. So in contrast to that, the actual amount of time that you spend either getting data, or calling the subroutine, or doing whatever you need to do, is

a short amount of time in comparison to that startup time.

So really what you're looking for in connection pooling is limiting that amount of the open and closing of the database connections continually and the use of the resources on the server itself. So if you could maintain that persistent connection there so that resources are actually just going through and using the database for just specific things that they need, you're going to save a lot of time from that standpoint.

So you're going to improve performance because you're not going to have that all the overhead of the opening of the sessions and managing all of that because that will be sitting there waiting, as well as you're going to reduce consumption of your server resources because you don't have all the overhead of starting the session, allocating memory, doing all the things that make that a very expensive transaction.

Also, you're going to be able to improve your scalability because if you can reduce that amount of time that's spend getting somebody to do the database, then you're going to be able to have a lot more throughput, so you're scalability is going to increase from that standpoint.

And more generally, I think the people that most benefit out

## *Getting the Most Out of U2 Connection Pooling*

of connection pooling are Web applications, because you're not always sure exactly how many users you're going to get at one point. You might get hit if you had tickets that go on sale or some special that runs for only an amount of time...

...everybody might be rushing to your Web page and you need some way to be able to scale that information without having database connections sitting there waiting one to one for every request that comes in. So that gives you a way to kind of multiplex those connections and make the most effective use of time.

BRUNEL: Dave, a question for you is, does it matter which database you're running, UniData or UniVerse, and which U2 interfaces will it work with?

PETERS: It will work with either of the U2 databases, UniData or UniVerse and you can use it in a number of different ways in the clients that you build. For example, U2 Web services developer would be one way. Or if you're a Java developer perhaps UniObjects for Java, UOJ. And if you're a .NET developer you could use UO .NET, IBM .NET or U2 .NET.

It's even possible to build your own connection pooling clients outside of these elements, but you'd still need a

connection pool license.

BRUNEL:       Okay. Elizabeth, Dave pointed out that it would work with both UniData and UniVerse. What releases of the databases must you be on to take advantage of connection pooling?

THOMAS:       Well, Kenny, IBM's connection pooling technology -- and this is an actual product -- is available for UniVerse 10.1.18 and later, and for UniData, 7.1 and later. These releases have been available for over two years now. Of course, user connection pooling can also be done with any release. User connection pooling is what you do if you wrote your own connection pooling.

BRUNEL:       So now we understand a little bit more about what connection pooling is and which interfaces it will work with. What are the performance gains with it? And Michael, I'll come back to you for this.

Byrne:        Sure. Essentially if you could take the same application, let's say a UniObjects for job application that was running without connection pooling and then eventually add that connection pooling in, you're going to see huge performance gains. We've seen everything from 5X to up and above for the just amount of throughput that you could get. And that's going to depend on...

## *Getting the Most Out of U2 Connection Pooling*

BRUNEL: 5X, you're saying five times faster?

BYRNE: Yes.

BRUNEL: Wow. Okay.

BYRNE: It's all going to depend on if you have a lot of requests that need to go through at a certain time, then you're going to be able to service more of those requests versus if your load is smaller, then you might not be as small. But essentially your throughput is going to be a lot greater and it's vastly improved.

There are some things to consider if you're developing a new application. Performance is going to be affected on that. Essentially if you're using connection pooling, the time that the resource that has the connection at that time spends on the back end doing something is time that other connections can't actually use that resource.

So the amount of time that you spend in server teams, if you're doing a lot of business logic or making external calls to third party applications or something like that, and you're waiting on other things to happen, that's time that can't be used on that.

So you want to make sure you optimize everything you can on the back end so indexing your files for selects, making sure

that, you know, file open routines and all those things are taken care of ahead of time so that you could minimize the amount of time that you actually spend on the back end, all those things will greatly increase your throughput and you know allow the performance gains even further.

BRUNEL: Dave, can you tell us, I wouldn't have any idea how many connection pools I would need for my, for any given application. Can you give us some information on that?

PETERS: Sure. Michael's brought up a lot of good points when you're trying to tune an application for connection pooling but really you have to look at what your application needs are.

If response time is critical, then you'd probably do better with more connection pools available so they could be ready to respond if needed. For example, if at your peak hour you expect to have, say, 20 simultaneous requests going to the database, you'd want to think about having 20 connection pools so that all requests could be serviced at the same time.

If response time isn't so critical and you know your configuration runs well with 10 simultaneous database requests, then you might just want 10 connection pools. It really depends on what your application needs are and what

your experience is with your application in tuning it.

BRUNEL:       Okay. Well, that brings us to my favorite part of the podcast, which is how we learn to get the most out of connection pooling. So, Michael, it's back to you again. I know you had some points that you wanted to cover here.

BYRNE:        Sure. So a couple things to just mention here. I'll go over a couple of points that, number one, you're not ever guaranteed that you're going to get the same thread the next time you come back on the connection pool because you might have a pool of five or 10 connection pools waiting. You might get a different account.

So you need to make sure that you do any of your setup for your [actual] application in any type of setup routines or shutdown routines before you actually add it, because things like file open routines or setting up common or cleaning up certain members that you might do, you want to actually make sure you take a look at that so that you don't leave anything hanging around for the next person that might not be you that gets the connection.

Secondly, from a performance standpoint, I like to just go on the open late, close early policy, which would be.... And this is not a Taco Bell policy [LAUGHTER] open your database connection as late as possible, so before you need

it, right before you're actually going to do anything with the actual connection, that's when you want to open it.

And then you want to close it as soon as possible when you're done with it, because the sooner that you can release that connection that gets put back in the pool for whoever else might need it. And at large load time somebody else might be waiting on that connection, so you want to release that as soon as you could.

Also, make sure you are closing any connections when you're navigating away from a page, perhaps, or things that might take you away from your, you know, for exceptions and things like that that could pass that might lead you to exit a page without closing it, you want to make sure you take a look at that.

A couple more points is you want to probably use a common set of user accounts when you're making the database connection. Those are based on user roles. Each of these account and user combinations actually is what defines a connection pool.

So a certain set of accounts with the specified permissions and things like that would probably be a good idea to take a look at doing that rather than having each person use their own specific individual connection.

There are a couple other things like tracing is provided as far as troubleshooting goes as well as performance counters on all our.NET APIs and providers. And then, I think in general the biggest thing when you're programming is you just never assume that the connection is going to get the same thread, so you need to be aware of that for any state management or common or any of those issues.

BRUNEL:       Okay. Thank you, Michael. Elizabeth, a question for you now. I understand that there's different ways of using connection pooling. Can you expand on that a bit?

THOMAS:       Yes, Kenny. As I alluded to before, there are actually three ways that connection pooling can be employed with the U2 data servers. The first is simply to purchase the U2 connection pooling program from IBM. The second is to develop connection pooling technology yourself or to purchase third party code. Now, we labeled this type of connection pooling user connection pooling to differentiate it from the IBM's product.

Now, the third way is to use a tool such as a business intelligence application. Many of these sorts of applications utilize a connection point to query the U2 database. In this case, the end user of the application is

actually responsible for purchasing user connection pools from their database supplier to remain compliant with all licensing.

So for U2 OEM partners there can be a revenue opportunity here if they allow third party tools provider such as business intelligence tools, but certainly not limited to that, to sell into their install base. And they should take a look at that.

BRUNEL: Thank you, Elizabeth. So, Dave, I'm sure as the product manager, you're going to want to get your two cents in on how to get the most out of connection pooling, so what can you tell us?

PETERS: I think the most important thing would be to learn as much as you can. And the best way to do that, I believe, is to go to U2 university. Not only do you get to go to sessions that cover the clients that I mentioned before, but they cover connection pooling in each of them.

In addition, you get a chance to mingle with the experts and really get some indepth knowledge that would be difficult for you to get any other way. So you can find out more about U2 university on the IBM Web site.

In addition, you can look for developerWorks articles on the

*Getting the Most Out of U2 Connection Pooling*

IBM Web site. There's a particularly good one on developing applications with connection pooling in .NET. So look for those on the IBM Web site, too.

BRUNEL: When you mention the IBM Web site, Dave, are you referring to the specific site for U2?

PETERS: Yes. Well, you can actually go to just [www.ibm.com](http://www.ibm.com) and search for U2 space connection pooling and you'll get a lot of good information that way.

BRUNEL: Now we know how to get the most out of connection pools and we need to find out how do we...how does somebody obtain connection pools? Elizabeth, can you take that one?

THOMAS: Well, Kenny, IBM's connection pooling technology can be obtained through our U2 Business connect Application, just like any other U2 technology, such as our data servers.

Now, for people who have written their own connection pooling or happen to be using third party code, or if they're using a third party tool that employs connection pooling, they should contact their IBM U2 sales representative to work out a plan that puts them in compliance with their contract. If you don't know who your

*Getting the Most Out of U2 Connection Pooling*

IBM sales rep is, you can always e-mail [u2bc@us.ibm.com](mailto:u2bc@us.ibm.com) to obtain this information.

BRUNEL:       Okay. Thank you, Elizabeth. Dave, thank you, and Michael. I would also like to thank our listeners for joining in today. We hope that you have found today's podcast useful.

Be sure to e-mail us with any feedback or comments, suggestions for future episodes that you may have. Our e-mail address is [U2askus@us.ibm.com](mailto:U2askus@us.ibm.com), or you can simply click on the feedback link just below this podcast download on our Web page.

Transcripts for today's podcast which include all relevant links are available on our Web site by following the U2 podcast link from the main page. And you can get to our main page by connecting to [www.ibm.com/software/U2](http://www.ibm.com/software/U2). Be sure to stay tuned in for future episodes of how to get the most out of IBM U2.

[END OF SEGMENT