

Asynchronous sockets and pooled thread loading

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-05/msg05136.html>

- *From:* "User N" <UserN@xxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 28 May 2005 05:50:04 -0400
-

I'm working on a proxy which must support at least a dozen simultaneous connections from local clients to remote servers. It is conceivable that someone might want to run it in non-local mode, with as many as 3-4 dozen simultaneous connections from remote clients to remote servers. Supporting that is desirable but optional. The proxy will have to maintain client/server connections for up to several hours, but the traffic will be intermittent/bursty. I have no hard requirements in terms of performance. The proxy does a bit of massaging of the messages that get sent through it, and I haven't figured out the internal logic. But it isn't out of the question that I might want to do a BeginReceive and BeginSend on both the client and server sides. Which would mean four "outstanding" BeginX requests per proxy * N proxies.

I understand that when a send or receive completes, the callback will be executed on a some kind of pooled thread. I expect my callback routines to be fairly lightweight... just some simple text processing, optional debug oriented logging, and accesses to a shared Hashtable for purposes of recording/recalling that certain messages & events occurred. I have no idea of what kind of load the callback/completion routines will put on the system and can only hope for the best until I get some protocode up and running or someone points out an obvious problem. What I'm mainly concerned about ATM is whether the simple act of Beginning many sends/receives could tie up too many pooled threads.

Could someone give me some sense of how these pooled threads are actually used between the time a Begin is launched and the request completes? For example, does a BeginSend cause a thread to be allocated until all of the data can be written to the underlying socket's send buffer and the send callback completes? Does a BeginReceive cause a thread to be allocated only after data is available in the socket's receive buffer and until the receive callback completes? Thanks.

.