

Re: Asynchronous socket operations and threadpool

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework/2006-03/msg00206.html>

- *From:* "Chris Mullins" <cmullins@xxxxxxxxxx>
 - *Date:* Mon, 6 Mar 2006 15:05:04 -0800
-

"William Stacey [MVP]" <william.stacey@xxxxxxxxxx> wrote

| The scenario I keep seeing is:
|
| 0 – You get the Socket.BeginRead callback, and get your data. You're now
| on
| an IOCP thread.
| 1 – Perform an async operation (say, lookup MX or SRV records in the
| DNS).
| Pass in a delegate for the callback.
| 2 – While that operation is under way, your IOCP thread keeps going
| doing
| whatever it can do.
| 3 – Eventually the IOCP thread hits a WaitHandle and has to sync up with
| the
| async operation you kicked off.

I thought we are talking about a pure async server? This is more like a
thread per connection server because your blocking on a Wait.

The wait is on a WaitHandle for an operation performed during a callback.
For example, you get a big chunk of data for the user (via the async call)
and realize you need to perform a database lookup to satisfy the request.
You kick off the DB Request async, do as much more of the user request as
you can, then wait for the DB Request to complete. Once it's done, you send
the user back his data, and put the socket back into BeginRead.

I tend to do either this, or just do everything synchronous once I'm on the
IOCP thread in my callback.

At various times, I've tried doing this a number of other ways and always
come back to this approach.

In a full async server, you would not block at all (or for very short
times).

Re: Asynchronous socket operations and threadpool

In `Socket.EndRead`, you would get data, update state, and Begin your next async operation, and on down the line like walking a "virtual" task list controlled by state.

An additional complication is that your socket is back in `BeginRead` mode so you may get another request in on that socket. Now you need to decide which request to process first – is it key requests are processed in order? If so, then more logic is needed (a state machine, as you allude to).

One thing I'm not clear on, given what you describe – when you get data from a socket, and realize you have something significant to do (and you want to do it async), how do you do it? You can't just post it to the `ThreadPool`, as there aren't enough threads in there. You don't want to manage a ton of threads manually if you can help it...

So most the time, your system will be waiting on `BeginReceives` and `EndReceives` to fire.

Agreed – if you have 10K connected TCP sockets, almost all of them are going to be stuck in "BeginReceive" at any particular moment in time. This is by design and one of the biggest strengths of the IOCP infrastructure – it manages which threads are awake, what processors they run on, what socket data they have, and all of the other good stuff.

--

Chris Mullins

.