# COM asynchronous interfaces, part 5: The unreliable server

devblogs.microsoft.com/oldnewthing/20220218-00

February 18, 2022

Raymond Chen

So far, we've been making asynchronous calls on the assumption that the server is well-behaved, but just a little slow. But what if the server is not being responsive at all?

Let's break down the steps that take place on the main thread and classify which ones involve synchronous calls to the server. These are the ones that are going to cause trouble if the server is unresponsive.

> Querying for the call factory.

The call factory lives on the client, so obtaining the call factory for a proxy does not call out to the server.

> Creating a call object.

To create the call object, COM needs to verify that the remote object does support the interface you are trying to call.

In our case, we have the proxy in the form of an `IPipeByte`, and in order to have obtained that, we must already have confirmed that the remote object supports that interface. So creating the call object does not call to the server.

On the other hand, if our proxy had been in the form of an `IUnknown`, then creating the call object for `AsyncIPipeByte` would require calling out to the server to `QueryInterface` the server object for `IPipeByte`. This query has to be done only once per proxy, because COM proxies cache `QueryInterface` results.

The `Begin_` methods on the asynchronous interface call out to the server but do not wait for a response. They return immediately, and the arrival of a response from the server can be queried by polling the `ISynchronize::Wait` method.

So if you want to avoid synchronous calls to a possible-unresponsive server when you make your asynchronous call, make sure that you have obtained the interface on the remote object ahead of time.

One way to improve the likelihood that you can get that interface is to query for it the moment the server gives it to you. The server is almost certain to be responsive at that point, seeing as it just responded to your previous query. If you obtained the object via `CoCreate-Instance`, you can request multiple interfaces at once by calling `CoCreateInstanceEx`, thereby avoiding an extra round trip to the server for each additional interface. (We'll look more at `CoCreateInstanceEx` in a few weeks.)

Raymond Chen

**Follow**