Subject: Re: Speeding up data crunching using IDL_IDLBridge with asychronous execution
Posted by Russell Ryan on Wed, 10 Jul 2013 18:12:47 GMT
View Forum Message <> Reply to Message

Hi Chip,

That's new. Two quick questions, in my test script, did you use the /nowait feature in the call to EXECUTE method? If not, try that. Again, that's the point of using the IDL_IDLBridge.

Second question, how exactly did you destroy the object if not with obj_destroy? I guess I didn't understand how you used the cleanup method. I never tried any of that, so I'm anxious to see what that does. I doubt this is the answer, because you can move the object creation and destruction *OUTSIDE* of my loop, and the memory problems do not go away.

Like I said, I don't have Linux. I sent that script to a friend who does use Linux, and he told me it did have problems, but I don't know which linux, which IDL, or which options he did/didn't use.

R

On Wednesday, July 10, 2013 2:05:25 PM UTC-4, Chip Helms wrote:
> I'm not actually seeing any memory leak. The memory lost during a single iteration remains zero at least through iteration 29. (The total memory lost stays constant...my interpretation is the value is the memory taken by the bridge object)
>
>
>
> The IDL docs on OBJ_DESTROY suggest using the cleanup method is better in the instance of recursive object definitions (and I've no clue if IDL_IDLBridge generates objects within itself) so I thought to try "o->Cleanup" (despite the docs saying that "o->cleanup" is the same as "OBJ_DESTROY, o").
>
>
>
> Also, in this example a new object is created and destroyed each iteration. If I rewrote this to mirror my own implementation, it would create the bridge once and then each iteration it would use setvar, execute, then getvar (or in this case just use execute each iteration). This also has the benefit of avoiding the overhead of creating and destroying the objects, but I'm unsure as to what impact it has on any memory leak.
>
>
>
> Regardless, here's my version structure if you're interested:
>
> { x86_64 linux unix linux 8.2.3 May 2 2013 64 64}