Subject: Re: Multithreading in IDL
Posted by tdaitx on Tue, 26 Jul 2005 12:06:20 GMT
View Forum Message <> Reply to Message

IDL itself can be set to run multithread even on a single processor system. Simply run CPU, TPOOL_NTHREADS = NumThreads and you're set.

So far my experience says that on Linux there'll be a process for each thread and on Windows all I could get was a single process, no matter what - maybe I'm doing something wrong, but my guess is that is the way multithreading works on Windows (note that I neither own a SMP system nor have access to one).

Now, if you're asking about threading support for your code, I must say that IDL doesn't provide any means to do it (someone please correct me if I'm wrong here). If you want your code to spawn some threads and get a handle of it you'll have to wait till RSI decides to support thread programming.

A creepy way I can think of to let a piece of code running in the background while a main program runs (or not if you want so) is creating a base widget with Map 0 and register it with Xmanager using the No_Block keyword, since Xmanager can run tasks in the background. It's an ugly workaround and I didn't dare to try it (yet), but it's there if someone wants to give it a shot.

And, finally, you can always choose to use call_external or link_image, but it's almost the same as using the Xmanager trick, what you get is multiprocessing, not multithreading. Take a look at the Multithreading topic
(http://groups-beta.google.com/group/comp.lang.idl-pvwave/bro wse_thread/thread/f49fb0d9810cd530/b51116bb6b96cf71)

Regards,
Tiago S Daitx

Subject: Re: Multithreading in IDL
Posted by Marc Reinig on Tue, 26 Jul 2005 14:25:27 GMT
View Forum Message <> Reply to Message

news:1122379580.769915.254950@g43g2000cwa.googlegroups.com.. .
> IDL itself can be set to run multithread even on a single processor
> system. Simply run CPU, TPOOL_NTHREADS = NumThreads and you're set.
>
> So far my experience says that on Linux there'll be a process for each
> thread and on Windows all I could get was a single process, no matter

> what - maybe I'm doing something wrong, but my guess is that is the way
> multithreading works on Windows (note that I neither own a SMP system
> nor have access to one).

On Windows, processes are ~the equivalent of a program that you run. The process can, in turn, have multiple threads. They all share the same address space and resources. They can also run simultaneously on multiple processors.

IDL supports multiple threads internally on Windows for some operations. FFT is one. We have an 8 processor machine. When it hits the FFT's all 8 come alive. When it goes back to regular code, only one is active. There are probably some Matrix operations that are also mutli threaded.

You can do your own multi threading using a DLM to spawn separate threads in Windows, but I haven't done it to use IDL functions.

Marc Reinig
System Solutions

Subject: Re: Multithreading in IDL
Posted by Ricardo Bugalho on Fri, 29 Jul 2005 13:20:11 GMT
View Forum Message <> Reply to Message

On Tue, 26 Jul 2005 05:06:20 -0700, Tiago Stï¿½rmer Daitx wrote:

> So far my experience says that on Linux there'll be a process for each
> thread and on Windows all I could get was a single process, no matter what
> - maybe I'm doing something wrong, but my guess is that is the way
> multithreading works on Windows (note that I neither own a SMP system nor
> have access to one).

Actually, that's a flaw of old threads implementation in Linux. You should not see one process per thread. Threads should be part of the process. The modern threads implementation doesn't have that bizarre effect.

>
> A creepy way I can think of to let a piece of code running in the
> background while a main program runs (or not if you want so) is creating a
> base widget with Map 0 and register it with Xmanager using the No_Block
> keyword, since Xmanager can run tasks in the background. It's an ugly
> workaround and I didn't dare to try it (yet), but it's there if someone
> wants to give it a shot.

Doesn't work, XMANAGER is a loop running everything in the same thread.
