Subject: Re: Idl error handling - is there a way to force a time out?
Posted by lecacheux.alain on Wed, 05 Sep 2012 22:31:52 GMT
View Forum Message <> Reply to Message

Le mercredi 5 septembre 2012 16:44:12 UTC+2, Brian Devour a écrit :
> Hey all, I've got a project where I am running an analysis routine on a ton of image files. Since it takes a couple minutes per image, I was setting it up to run overnight and therefore wanted to add error handling so that if it exploded halfway through it would go on to the next image rather than just stop. I did that with CATCH, but when I returned today to see if it had finished I discovered that it had still managed to hang halfway through. The problem was that somehow or other it had managed to get stuck in some sort of effectively infinite loop, and since it didn't explicitly throw an error, catch didn't force it to move on.
>
>
>
> I am of course trying to figure out how/why it got stuck, but since the place it got stuck was somewhere in the guts of mpfit, that is taking a while. However, this suggested to me that my error handling should include some sort of time out condition. I know roughly how long each image *should* take, so I wanted to have it halt and move on if it was taking significantly longer than it should. However, I haven't found any simple way to make IDL do this.
>
>
>
> The closest I've been able to come up with is to include some sort of statement like this in my outermost procedure right before the call to the analysis routine for each image:
>
>
>
> start_time = systime(1)
>
> timeout_limit = 500
>
>
>
> and then paste this line everywhere I can inside the guts of my program:
>
>
>
> if ((systime(1) - SCOPE_VARFETCH('start_time', LEVEL=2)) gt (SCOPE_VARFETCH('timeout_limit', LEVEL=2))) then message, error_message
>
>
>
> which would actually explicitly throw an error if it was taking too long so that catch could force it to skip that image and move on to the next one. The problem I have with that approach is that execution has to actually pass through the if-then statement to detect a timeout, so if it gets stuck in a place where I hadn't anticipated, or gets stuck inside something where it's not feasible to paste that statement, it won't work. It also seems like a terribly clumsy solution, and it really feels

like there ought to be a better way to do it. Ideally there'd be some sort of way to call a function/procedure with a generic timeout condition that *doesn't* require execution to pass through it in order to trigger, but if there is, I haven't been able to find it. Does anyone know of a way to do something like this?

A possible way might be by processing each of your image files in an asynchronous thread, by launching a specific IDL_IDLbridge in NOWAIT state from you main IDL session.
By checking the state of this thread (completion, error, etc...) within a loop in you main session (by using the IDL_IDLbridge.Status() function), you should be able to restart a new one when finished, or abort the pending one after some exceeding delay, and so on.
alain.
