Subject: Re: Python and IDL
Posted by JD Smith on Fri, 28 May 2004 20:43:52 GMT
View Forum Message <> Reply to Message

On Fri, 28 May 2004 10:38:08 -0500, Michael Wallace wrote:

> I'm probably the only person here who'd be interested in this, but I guess
> I'll throw it out anyway.
>
> I have recently started using Python and love the language. Python is
> interpreted, interactive, object oriented, and has dynamic data types and
> dynamic typing. From this perspective, IDL and Python are very similar.
> There are even projects such as SciPy (http://www.scipy.org) which add
> some scientific analysis to the language.
>
> Python does have a couple nice benefits, namely the huge number of modules
> available where there's everything from operating system commands to
> networking to XML to email etc. One thing I've found myself doing
> recently is writing my core analysis processing code in IDL and then
> writing a Python wrapper around the particular code. For example, in one
> of my programs, I use Python to automatically download files from an FTP
> site and then call IDL to do the specific processing on the files. In
> another program, I use Python to process command line arguments and then
> call the appropriate IDL code based on the arguments.
>
> Currently, in my Python programs I open a pipe to an IDL process and write
> the IDL commands to the pipe. What I'm wondering is if there is an
> efficient way to send data from IDL back to Python. The only thing I've
> found so far is to have IDL write data to stdout and set up the Python
> side to read this stream. However, this can get pretty inefficient at
> times. Without any direct conversion between the two languages available,
> are there any other (i.e. better, efficient) ways to set up bi-directional
> interprocess communication between Python and IDL?
>
> What'd be really nice is if RSI provided some mechanism for IDL and Python
> to communicate at a much closer level than interprocess communication.
> Karl, that's a hint. ;-)

Seems like this one is coming up a bit recently: you could use shared memory. This is typically the fastest IPC available, but does require some coordination when both sides are reading & writing. It's also fairly flexible, in that data in any format can be written. Actually interpreting that data on the Python or IDL side would require the same tricks implicit in writing DLMs (map integers to shorts, etc.), but without needing to compile. Ideally you could write a glue module which used shared memory to map arrays to arrays, etc.

JD