Subject: How to obtain the process ID of the current IDL process in a platform-independent way?

Posted by tcburt on Fri, 04 Sep 2009 02:08:24 GMT

View Forum Message <> Reply to Message

*** Question
Is there a platform-independent "IDL way" to obtain the process ID of the current IDL process?

*** Background
I need the process ID (PID) of the current IDL process. Currently I have a working solution for a specific platform (Solaris 9 and 10), specifically

```
  IDLUnix> pid = CALL_EXTERNAL("/lib/sparcv9/libc.so", 'getpid')
```

The reliance on a library from the operating system library limits the applicability to that particular platform and installation, so I consider it only a provisional solution.

A recent discovery is the Unix libidl.so library that is in the IDL (v6.4 and v7.0) installation directory. Dumping the contents with

```
  shellUnix> elfdump -s libidl.so | less
```

revealed the existence of a 'getpid' function that returns the PID via

```
  IDLUnix> pid = call_external(!dlm_path+'/libidl.so', 'getpid', /
cdecl)
```

This is one step towards platform independence since the library is from IDL rather than the operating system and its location is stored in an IDL system variable. I have not yet tested on anything but the Solaris 10 systems, so this may not work on other Unix systems (e.g. linux).

I then turned to a Windows installation of IDL (v 6.2) and did not find a library called libidl.dll in the !dlm_path, but did find idl32.dll. I guessed[*] that this library would have 'getpid' as the entry symbol, so I tried

```
  IDLWindows> pid = call_external(!dlm_path+'\idl32.dll', 'getpid', /
cdecl)
```

The resulting error

```
  % CALL_EXTERNAL: Error loading sharable executable.
```

indicates that the problems go deeper than just whether the symbol is in the library. Even if the call_external() had worked under Windows, the method could have potential problems with internal changes to IDL (e.g. library name change from idl32.dll to idl.dll).

I seek the "IDL way" to obtain the PID. Searches in idlhelp, comp.lang.idl-pvwave, and Google have not revealed the way. It is a testament to the usefulness of this newsgroup over the past few years that other questions I had were already answered in the archives. I ask for your insight about the existence of robust solutions and pointers to more fruitful paths (such as writing specific external functions to determine the PID rather than using the libraries delivered with IDL).

In appreciation for benefits already obtained,
Tim

[*] I guessed because I do not know how to dump the contents of a Windows DLL. Local gurus will likely be able to help me remove this layer of ignorance.
[^] % CALL_EXTERNAL: Error loading sharable executable.