Subject: Re: java-idl connector & memory issues
Posted by apollomitqy on Fri, 13 Jul 2012 15:03:15 GMT
View Forum Message <> Reply to Message

On Tuesday, 13 May 2008 00:00:36 UTC+10, (unknown) wrote:
> On May 10, 1:36 pm, Jelle <p...@bio-vision.nl> wrote:
> > Hi All,
> >
> > Currently I am working on a processing line for eumetsat MPE data,
> > which comes in the very userfriendly -ahum- adjusted GRIB2 format. I
> > managed to find a java library that reads the files, and gives me an
> > array of raw doubles. I use the idl-java bridge to export this data to
> > IDL, where I project, subset, warp and save my data subsets.
> >
> > The java retrieval in itself (without actually calling IDL) runs fine.
> > The IDL script in itself run fine. Combined.. THey run fine for 20-30
> > files. Then the scripts starts to tos me errors:
> >
> > Caught an error
> > e:\MSG1-SEVI-MSGMPEG-0100-0100-20070421184500.000000000Z-907 242-9.grb
> > com.idl.javaidl.JIDLException[iErr=-1 sMsg=Could not get IDL error
> > information.]
> > at com.idl.javaidl.JIDLPAL.nativeThrowJIDLException(Native
> > Method)
> > at com.idl.javaidl.JIDLPAL.throwSpecificException(JIDLPAL.java:
> > 1039)
> > at com.idl.javaidl.JIDLPAL.throwJIDLException(JIDLPAL.java:
> > 1068)
> > Caught an error
> > at com.idl.javaidl.JIDLPAL.setIDLVariable(JIDLPAL.jav
> > a:702)
> > at com.idl.javaidl.JIDLObject.setIDLVariable(JIDLObject.java:
> > 588)
> > at GMPE.arrays_example.arrayManipulation(arrays_example.java:
> > 54)
> > at GMPE.arrays_example.main(arrays_example.java:137
> > e:\MSG1-SEVI-MSGMPEG-0100-0100-20070421190000.000000000Z-907 242-9.grb
> >)
> > com.idl.javaidl.JIDLException[iErr=0 sMsg=]
> > at com.idl.javaidl.JIDLPAL.nativeThrowJIDLException(Native
> > Method)
> > at com.idl.javaidl.JIDLPAL.throwSpecificException(JIDLPAL.java:
> > 1039)
> > at com.idl.javaidl.JIDLPAL.throwJIDLException(JIDLPAL.java:
> > 1068)
> > at com.idl.javaidl.JIDLPAL.setIDLVariable(JIDLPAL.java:702)
> > at com.idl.javaidl.JIDLObject.setIDLVariable(JIDLObject.java:
> > 588)

> > at GMPE.arrays_example.arrayManipulation(arrays_example.java:
> > 54)
> > at GMPE.arrays_example.main(arrays_example.java:137)
> >
> > At that point java just continues reading files, but skips the IDL
> > bridge. I am running this in windows, and in my memory overview I see
> > that when I include the IDL processing, memory fills a little more
> > with each file. I am not sure what specifically coul be going wrong.
> >
> > The function called to do the IDL work:
> >
> > ======>
> > private void arrayManipulation(String TheName, float[] Data)
> > {
> > try {
> > String a = "dataarray";
> > String b = "envi";
> > String d = "filename";
> > String g = "D:_software\\IDL_libraries\\MyScripts\
> > \Grib_javaIDlbridge_SettingsFiles";
> > String c = "doprestimate, dataarray, '"+TheName+"',
'"+g
> > +"'";
> > ostock.setIDLVariable(a, new JIDLArray(Data));
> > ostock.executeString(b);
> > ostock.executeString(c);
> > }
> >
> > catch (JIDLException e) {
> > System.out.println("Caught an error");
> > e.printStackTrace( );
> > }
> > }
> > <======
> >
> > The way I call this function from the .main() routine:
> >
> > ====>
> > for(int j=0; j<children.length; j++)
> > {
> > // Get filename of file or directory
> > String filename = children[j];
> >
> > try {
> > progInstance.SetCreateFileIndex(true);
> > progInstance.SetInFile(dir+filename);
> > float[] theData = progInstance.GetData();
> > example.arrayManipulation(filename, theData);

> > theData = null;
> > }
> > catch (IOException ex) {
> >
> > Logger.getLogger(arrays_example.class.getName()).log(Level.S EVERE,
> > null, ex);
> > } catch (NotSupportedException ex) {
> >
> > Logger.getLogger(arrays_example.class.getName()).log(Level.S EVERE,
> > null, ex);
> > }
> > }
> > <====
> >
> > Am I missing something crucial & simple to fix to make this run? I
> > have 14000 (!) files to run, so just running 20 files at a time and
> > adjusting the loop is no option..
> >
> > Thanks so much,
> >
> > Jelle
>
> Jelle,
>
> You are correct. There is a memory leak in the Java Export Bridges.
> It was fixed in the code base after the IDL 7.0 release. What version
> of IDL are you using? If you are using IDL 7.0, the fix is easy to
> apply -- it's just replacing two files in the IDL distribution.
> Unfortunately, there is nothing you can do in your Java code to get
> the IDL Java EB to release the memory.
>
> The best approach is for you to contact ITTVIS tech support and work
> the issue through them and possibly get updated files that fix the
> memory leak issue.
>
> Hope this helps.
> Abraham

Hello Abraham and all,

You mentioned "Unfortunately, there is nothing you can do in your Java code to get the IDL Java EB to release the memory." - Is this still the case for IDL 7.1???

We noted that every time Java calls the IDL Jave EB, the idl_opserver kicks in. And at the same time, the Java process builds up memory usage as shown in Windows Task manager. Afetr comleting the processing, Java destroys the bridge object and the idl_opserver disappears as expected. However, the Java process (and IDL Jave EB perhaps?) does not release memory so its memory usage keeps building up - This seems to match your above description.

Is there any way to get IDL Jave EB to release the memory in IDL7.1 or later version? Any tips/suggestions will be appreciated.

Many thanks.

Qing
