README for the biopython-corba module

Brad Chapman (chapmanb@arches.uga.edu)

Last Update–15 April 01

This module contains python code to help with connecting to various CORBA related bioinformatics services.

This contains the python implementation of biocorba.idl, an idl definition for connecting biopython, bioperl and biojava. In addition to this README, which is a really quick overview of things, there is more extensive documentation available in the Doc directory in the files biocorbahowto.pdf and biocorbahowto.html.

Brad Chapman (chapmanb@arches.uga.edu) wrote all of the trash in this module, so if you hate anything, go ahead and send him and e-mail to tell him off :)

1 Requirements to Run This

1. Python 2.0 or better, available from http://www.python.org
2. Distutils 1.0 or better, available from: http://www.python.org/sigs/distutils-sig/download.html This is the standard build system in python 2.0, so if you've got python 2.0, you are all set.
3. An ORB implementation with python bindings. Right now this has been tested to work with three different ORB implementation:
   (a) omniORBpy - The python bindings are available from http://www.uk.research.att.com/omniORB/omniORBpy/ and omniORB itself is available from http://www.uk.research.att.com/omniORB/index.html. This has been tested to work with omniORBpy 1.0 and later and omniORB 3.0 and later. Older versions of omniORB will probably work fine, although this has not been tested with them. However, older versions on omniORBpy will not work, due to a change in the python mapping. Sorry about this... Anyways, Installing omniORB can be pretty tough, so be sure to read the install instructions for it. This is the most supported ORB implementation, and should work like a charm with everything in biopython-corba.
   (b) Fnorb 1.1 - available from http://www.fnorb.org/ Fnorb is fairly easy to install because it is written in almost all python. The support for Fnorb is fairly good, but I am getting some random lockups and crashes using Fnorb that I haven't been able to debug fully yet. If you are an Fnorb expert and can help me figure out the problems, that would be greatly appreciated!
   (c) orbit-python – The python bindings are available from http://sourceforge.net/projects/orbit-python/ and ORBit itself is available from ftp://ftp.gnome.org/pub/GNOME/stable/sources/ORBit. This has been tested with orbit-python-0.2 and the current version of ORBit. Right now ORBit support is really shakey, which I believe is due to bugs in orbit-python. Again, I would really appreciate any help shaking out these bugs from people experienced with orbit-python.
4. Biopython. The latest code is available for download at http://www.biopython.org/Download/ and the bleeding edge stuff is available via cvs at http://cvs.biopython.org. Everything in here should work with the latest biopython release.

5. Bioperl. If you want to work with bioperl you'll need to download their code. Bioperl releases are available from http://bioperl.org/Core/Latest/index.shtml. To use bioperl-corba you'll need the most recent version from CVS (it has bug-fixes for interoperability with Biopython). You'll want the bioperl-corba-server module, see the CVS instructions at http://cvs.bioperl.org.

6. Biojava hasn't been upgraded to the latest version of the spec yet, so there is no Biojava support yet.

2 How To

2.1 Getting set up

1. Get all of the appropriate stuff you need (see above) and decide what orb implementation you want to use.

2. Get biopython-corba, which is available by anonymous cvs at biopython (see http://cvs.biopython.orgfor instructions), or via download at http://www.biopython.org/Download/. Presumedly you already have this though, if you are reading this :)

3. Edit the configuration file. This file is located in BioCorba/biocorbaconfig.py, and contains the options available for building and using biopython-corba.

   - `orb_implementation`- Set this variable to either 'Fnorb', 'omniORB' or 'ORBit' (make sure the case is right) depending on what you want to use. The default is 'omniORB'
   - `supported_interfaces` - Right now support is available for 'biocorba,' the main BioCorba interfaces, and 'ensembl,' the bindings to the Ensembl system (http://www.ensembl.org).

2.2 Building and install everything

Since this uses distutils, it is easy to do a full install which will put everything into your site-packages directory (on UNIX). To do this, just type `python setup.py install`. Hopefully this should do everything, and you shouldn't get any error messages. Note that you'll need to be root to do this. The install should compile all of the IDL files, get everything setup, and move them to the right place. Be sure you've specified your ORB implementation of choice before doing this.

I haven't yet tested this on other platforms (Windows/Mac), so any install help for these platforms from other people would be very appreciated.

2.3 Testing the biopython-corba intallation

There are quite a few test scripts in the Tests directory of biopython-corba. The relevant ones for testing only biopython-corba are:

- `test_EMBLserver.py` – Tests a python client to the BioCorba EMBL server.
- `test_PythonBioEnv.py` – Tests a python BioEnv server with the regular CORBA interfaces.
- `test_PythonBioInterfaces.py` – Tests a python BioEnv server using the Bio-like interfaces.

To run these tests, you first need to start up the python BioEnv server. This server is located in the Scripts directory. If you change into this directory and type:

```
$ python bioenv_server.py &
```

This will start the server in the background. You can then run these tests and check for errors. The tests are very extensive, and everything should pass with omniORB. With other ORB implementations, there are currently problems (seg-faults, errors, etc) that need to be worked out, so I wouldn't expect everything to pass.

2.4 Testing with Bioperl

There is an extensive test suite to check interoperability with Bioperl, contained in the file `test_PerlServers.py`. There are two different ways to use this test script.

1. Run all of the tests – The script can start up a number of servers, run the tests on them, and then kill the servers; this will all happen automatically. To run the script like this, you need to have bioperl-corba-server installed, and then run the test script with:

   `python test_PerlServers.py </path/to/bioperl-corba-server>`.

   The `</path/to/bioperl-corba-server>` argument specifies the entire path to the location of the unpacked bioperl-corba-server directory.

2. Run only one test – If you don't want the automatic start up of all test scripts, you can start up just a single perl server and test it. Here's how to do it:

   (a) Start up the perl server, you want to run. For example, if you wanted to start up the simpleseq.pl server, you would need to change to the directory where you installed bioperl-corba-server and type:
   `perl servers/simpleseq.pl < AFastaFile.fasta`.

   (b) Run the python test script. You do it as before, but now you need to specify a `--test=` flag that gives the test you want to run. For our simpleseq.pl example, you would then run:
   `python test_PerlServers.py --test=simpleseq</path/to/bioperl-corba-server>`

Hopefully this new test script format will make testing with bioperl much easier.

2.5 Testing with Ensembl

There is now an IDL specifying interoperability with the Ensembl code-base. Ensembl (http://www.ensembl.org) is a comprehensive system for annotating and dealing with the human genome. Being able to interoperate with this is very nice; Ensembl is a great system and being able to access it from python is a big bonus.

Right now, Ensembl interoperability is coming along very well. The ensembl CORBA bindings are newly written, but already work for most things.

We have a Test script for working with Ensembl, in the file `test_Ensembl.py`. To use this, you'll need to install the ensembl-corba-server, available in Ensembl CVS(http://www.ensembl.org/cgi-bin/cvsweb/cvsweb.cgi/ensembl-corba-server/), along with bioperl-corba-server, bioperl and Ensembl itself.

There is also a test server available for developers interested in doing testing. Contact me (chapmanb@arches.uga.edu) if you would like access for testing purposes.

To run the Ensembl tests, you'll need to start up the ensembl server, and write the IOR to a file somewhere. The server can be either local or remote to the python client (if it is remote, you'll need to put the IOR on an http or ftp site where the client can get it). Then you can run the test script with:

```
python test_Ensembl.py </path/to/ensembl/ior>
```

The path can either be to a local file, or to an http or ftp site. An extensive test suite should than run.

To get an idea of what is working with Ensembl right now (and how to do things), check out the code in the test script.

2.6 Testing with Biojava

XXX Biojava doesn't have the biocorba-0.2 implentations ready, so this entire section is outdated and meaningless. It will be updated when things are ready over there.

3 Module Organization

- BioCorba - The main directory for all of the code.
  - biocorbaconfig.py - The configuration file to edit when you start using this.
  - biocorba.idl - The idl interface describing the bioperl/biojava/biopython connection.
  - ensembl.idl - The idl interface describing connectivity with Ensembl server.
  - Bio - Interface classes which make coding with the biocorba server work like coding with the normal Biopython code. This will make code portable between the two, and also allow you to only have to learn one interface.
  - Client - The client side corba code which looks exactly like the definitions in idl (except in python :). These provide wrappers around the functions so you don't have to do boring repetitive stuff like narrow interfaces and check for nil references.
  - Server - The server code which connects the main Biopython stuff (in the normal Bio directory) into corba. This allows you to run a full fledged server using Biopython.
  - Adapters - Biopython-corba uses the standard python mapping for CORBA (available from http://www.omg.org/cgi-bin/doc?ptc/00-04-08. These adapters make orb implementations which don't follow this mapping (yet) act like they do.
- Doc - The documentation for biopython-corba
  - biocorbahowto.pdf, biocorbahowto.html – The main documentation that describes how to do lots of things.
  - examples – A bunch of documented examples – the code from these is used in the documentation.
- Scripts - Scripts to start up servers.
- Tests - Test code to make sure everything is working properly.