Scripting McIDAS-X in a Python environment

Kevin Hallock
SSEC Datacenter Programmer
September 10, 2013
What problems are we trying to solve?

• Running a mcenv instance within a bash script requires redirection of stdin (i.e. "<< 'EOF'")
• Parsing text in bash can be clumsy, requires piping commands together to extract useful information.
• Manipulating dates in bash is messy, code can often depend on having a particular version of GNU date installed, for example.
What is Python?

- Python is a **general-purpose** interpreted programming language that supports multiple programming paradigms, including both object-oriented and procedural programming.
- Python is very **flexible** and can be used to write anything from simple scripts to an entire application with a fully graphical interface.
- Because of the flexibility it provides, Python has been one of the most **popular** programming languages over the past 10 years.
What makes Python a good alternative?

http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Where is bash?

https://sites.google.com/site/pydatalog/pypl/PyPL-Popularity-of-Programming-Language
What makes Python a good alternative?

- Existing Python modules such as NumPy and SciPy offer advanced mathematical/scientific analytical tools comparable to MATLAB or R.
- As implied by the name, Python modules are modular, allowing a user to import whichever modules they deem necessary for their project.
- Because of the nature of Python modules, code implementing the McIDAS-X module can easily be added to an existing project; similarly, any McIDAS-X Python script can be extended to use other modules.
What makes Python a good alternative?

Python is multi-platform
How our solution works

• The Python 'subprocess' module is used to spawn an instance of the mcenv shell in the background.

• mcenv shell commands are passed to the mcenv session using pseudo-native python functions, as in the following examples:
  – bash:
    
    `imglist.k DATA/SET FORM=ALL`
  – python:

    `mcenv.imglist("DATA/SET FORM=ALL")`

• In this case, the imglist method is not explicitly defined, but rather is interpreted such that the example bash command is run in the mcenv sub-shell.
#!/bin/bash
PATH=/home/mcidas/bin:$PATH
MCPATH=$HOME/mcidas/data:/home/mcidas/data
export PATH MCPATH
mcenv << 'EOF'
    logon.k ABC 1234
    dataloc.k ADD GROUP SERVER.DOMAIN
    imglist.k GROUP/DESCRIPTOR TIME=12:00
EOF
#!/usr/bin/env python
import mcidasx
import os
os.environ['PATH'] = "%s:%s" % ('/home/mcidas/bin', os.environ['PATH'])
os.environ['MCPATH'] = '%s/mcidas/data:~mcidas/data' % os.environ['HOME']

m = mcidasx.mcidas.mcenv()

m.logon('ABC 1234')
m.dataloc('ADD GROUP SERVER.DOMAIN')
m.imglist('GROUP/DESCRIPTOR TIME=12:00')

Setting environment variables could be further abstracted into the module import step in order to simplify code.
GOES East Full Disk over a range of days

#!/usr/bin/env python
import mcidasx
import os
import sys
from datetime import datetime
from mcidasx.utilities import date_range
mcidas_home = '/home/mcidas'
o.s.environ['PATH'] = "%s:%s" % (mcidas_home + '/bin', os.environ['PATH'])
o.s.environ['MCPATH'] = '%s/mcidas/data:%s/data' % (os.environ['HOME'], mcidas_home)
def main(start_date, end_date):
    mcenv = mcidasx.mcidas.mcenv(frame_size='900x1200')
    mcenv.logon("ABC 1234")
    mcenv.dataloc("ADD AGOES13 GEOARC.SSEC.WISC.EDU")
    for day in date_range(start_date, end_date):
        d = day.strftime('%Y%j')
        mcenv.imgdisp("AGOES13/FD MAG=-10 -20 BAND=1 DAY=%s TIME=17:45" % d)
        mcenv.frmsave("1 test-%s.jpg" % d)
if __name__ == "__main__":
    start_date = datetime.strptime(sys.argv[1], '%Y-%m-%d')
    end_date = datetime.strptime(sys.argv[2], '%Y-%m-%d')
    main(start_date, end_date)
Relevant python modules

• NumPy
• SciPy
• netCDF4
• GDAL
• matplotlib
• PyTables
• pandas
How can users install the McIDAS-X python module?

• Care was taken to ensure that the McIDAS-X python module could be installed and imported just like any other python module.


• Decompress the archive, and run 'python setup.py install --user' in the new directory (or use a Python tool such as pip or easy_install to install the .tar.gz directly).

• Now, within python, add the command 'import mcidasx', and all of the functionality of this module should be available.
Future work

• Additional modules which offer more direct access to underlying McIDAS-X commands
  – For example, the imglist command could return a Python object with callable methods and accessible attributes, rather than returning a multi-line string (which needs to be manually parsed to be useful).

• Additional specialized/specific utility functions
  – Perhaps functions that generate commonly used McIDAS time/date formats

• Although we are giving this "McIDAS-X/Python" presentation at the MUG meeting, this project is not currently funded by MUG.
  – Funded by another project at SSEC where the use of Python was necessary

• Disclaimer – This project was initially designed as a proof-of-concept, and as such has not been thoroughly tested for every McIDAS-X command.
Questions?