What's New at BRTT

BRTT
BOULDER REAL TIME TECHNOLOGIES, INC.

Kent Lindquist

November, 2013

Papagayo, Costa Rica AUG

Overview

- Recap announcements
  - Personnel changes at BRTT
  - Solaris, Linux and Apple
- Operational changes at BRTT
  - Distribution, Installation, Licensing, Support, Project management, Manifest
- Antelope Python
- Antelope 5.3
- New products from BRTT

Recap Announcements

BRTT

November 2013

Recap: Personnel Changes at BRTT

- Kent Lindquist hired full time January 2012
- Dan Quinlan retired December 2012
- Kent increases BRTT's ability to undertake new development
- Kent is no longer available as a private consultant

Recap: Solaris, Apple, Linux

- Antelope 5.3 WAS THE LAST ANTELOPE RELEASE ON SOLARIS!
  - Ramp down in Solaris development
  - Solaris support through one year from 5.3 release
- Uncertainty of Apple hardware future
  - Cannot depend on Apple for enterprise-class hardware
  - Awaiting test of the new Mac Pro in December 2013
- BRTT fully supports Linux as a platform for enterprise-class systems
  - BRTT fully supports RHEL and CentOS 6.2 in Antelope 5.3 release

Operational Changes

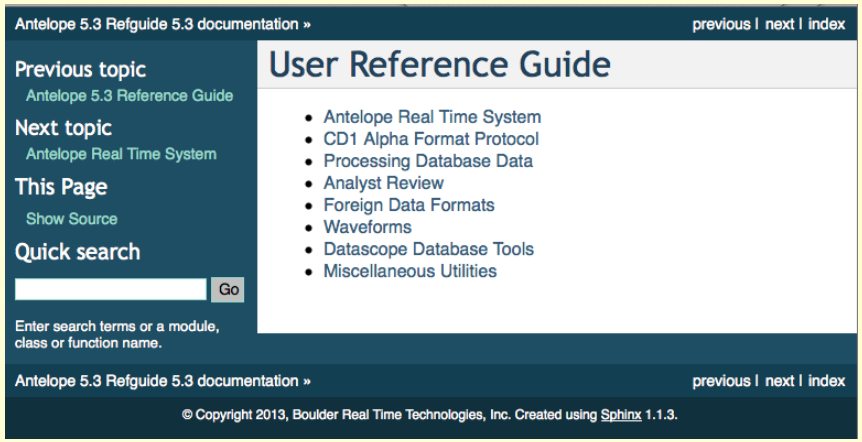
BRTT

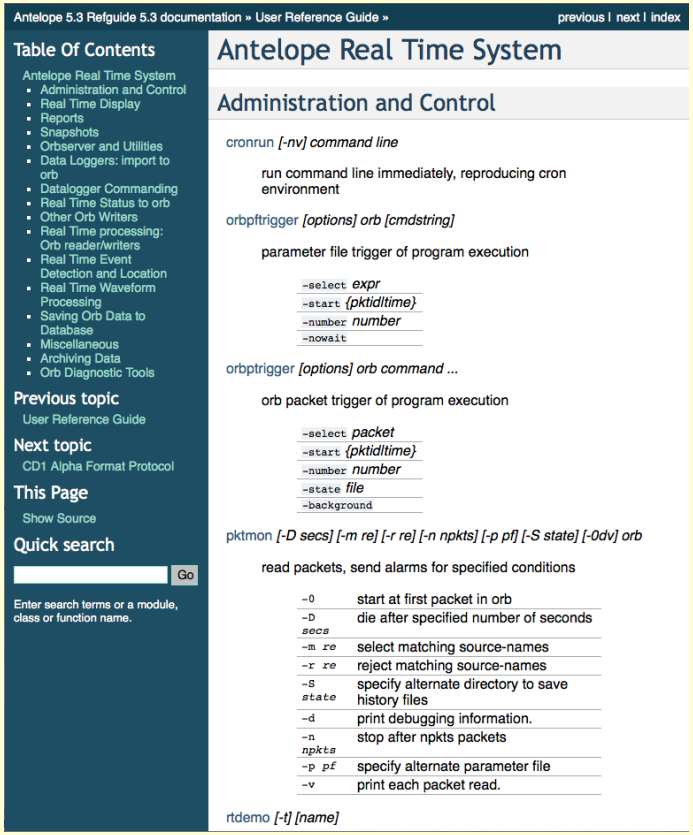
November 2013

Operational Changes at BRTT: Distribution

- Lots of problems in 2012 with CD publishing
- We are getting close to the CD size limit
- Kinemetrics is now responsible for CD (or DVD) publishing
- For 5.3, no more paper published manuals, including reference guides (they are web accessible)

Operational Changes at BRTT: Distribution



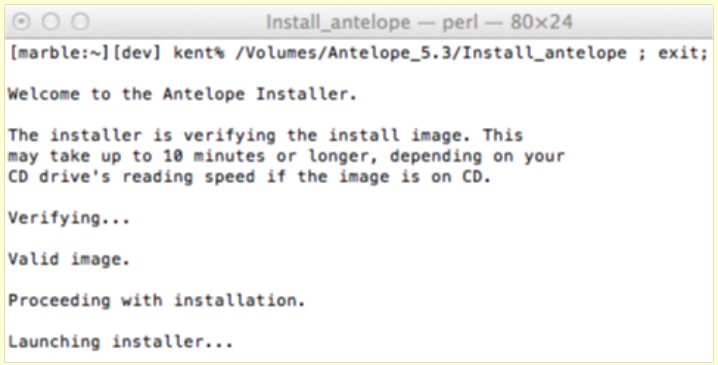


BRTT

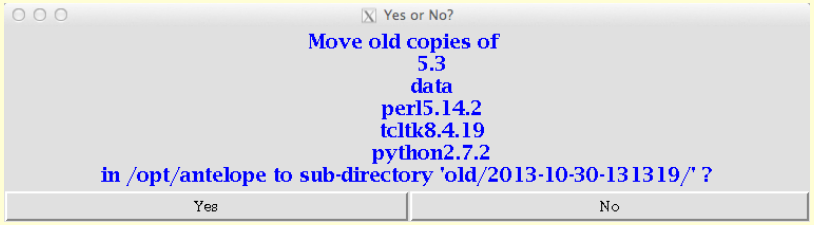
November 2013

Operational Changes at BRTT: Installation

- We have incorporated an automated verification check to identify duplication errors and stop installation



- Installation has been simplified (always installs everything, moves existing directories)



- No more -64 in directory names (e.g. “5.2-64”). All software is by default 64-bit.
  - This version is Antelope “5.3”
  - Next will be Antelope “5.4”

BRTT

Operational Changes at BRTT: Licensing

- Previous Antelope licensing mechanisms were antiquated, difficult to administer (on both ends) and prone to abuse
- We are overhauling Antelope licensing mechanisms (a lot of this will be invisible to our users)
- We will continue to provide node-locked licenses for the foreseeable future (under some restrictions)
- We have a server-based replacement for subnet licenses in 5.3. We think this will eventually provide the most convenient and flexible licensing system for our users (floating licenses).
- We are using this year's experience to improve the new system for Antelope 5.4 next year
- We ask for your patience as we transition into a new licensing system

Operational Changes at BRTT: Support

- Improved support responses via email and web https://brtt.zendesk.com
- Always get an automated reply with a ticket number
- Provides BRTT staff with coordinated support response tools
- You can go to the web site to see current and old support requests
- You can access your support requests from any web browser
- Our web site (www.brtt.com) describes this in more detail
- YOU ABSOLUTELY MUST USE

  support@brtt.com
- WE WILL NO LONGER RESPOND TO SUPPORT REQUESTS TO OUR INDIVIDUAL ADDRESSES

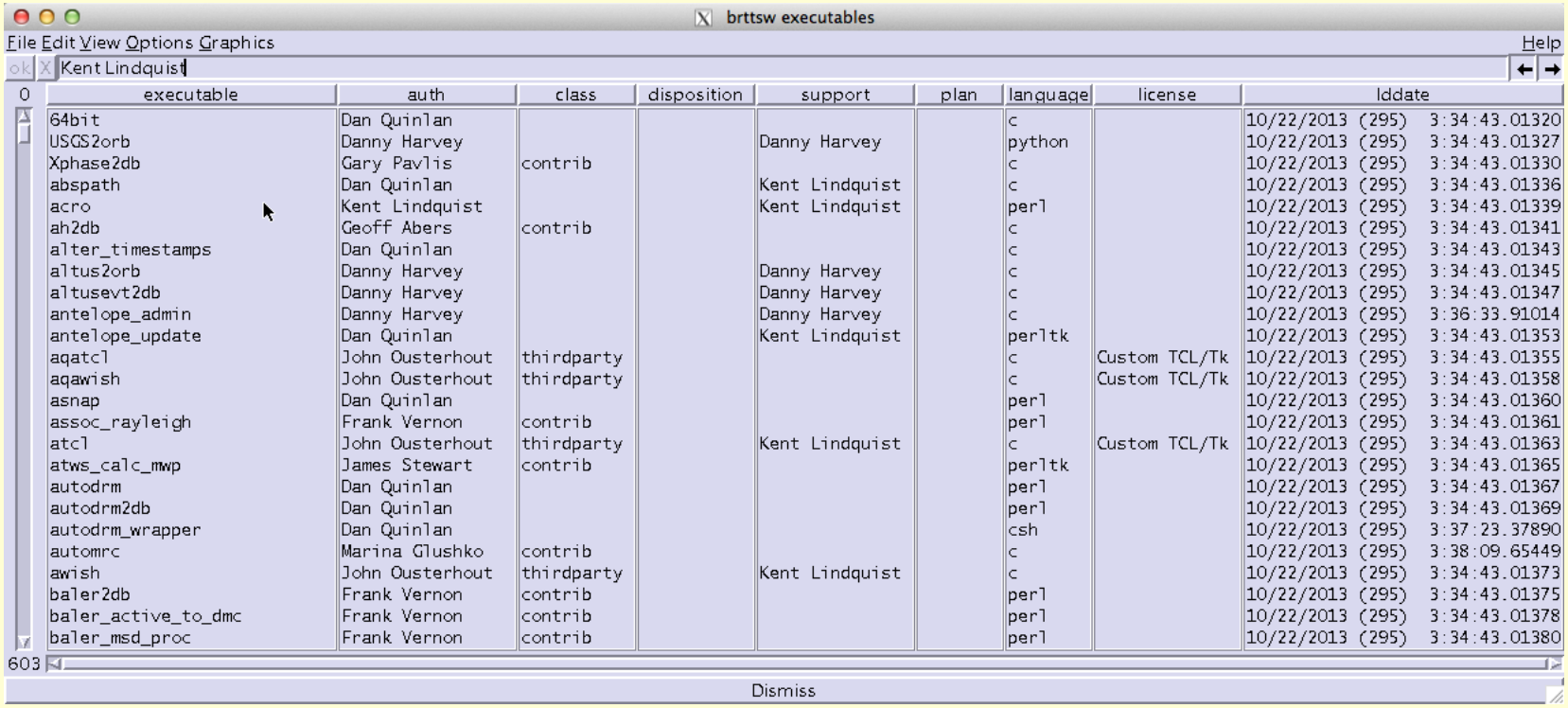
Operational Changes at BRTT: Project Management

- Project charters for each major project
- Requirements analysis for each major project
- Project portfolio management
- Team coherence
- Goals:
  - Continued regular delivery of new capabilities
  - Quality assurance
  - Triage of developments most valuable to the community
- Internal change

Operational Changes at BRTT: Software Manifest

- Developing internal manifest of software suite
- Contributed-code 'nobuild' area
- For Antelope 5.4:
  - New naming convention
    - Deprecated: *_dep
    - Preliminary: *_pre
    - Experimental: *_exp
- Auditing used and unused programs
- Identifying what is supported and by whom
- Wean out old code – drain on resources
- Focus on what's being used
- Comments welcome on what to keep / decommission
- Progressively rewrite, slowly eliminate *wish* and *Perl/Tk*

Operational Changes at BRTT: Software Manifest



executable	auth	class	disposition	support	plan	language	license	lddate
64bit	Dan Quinlan					c		10/22/2013 (295) 3:34:43.01320
USGS2orb	Danny Harvey			Danny Harvey		python		10/22/2013 (295) 3:34:43.01327
Xphase2db	Gary Pavlis	contrib				c		10/22/2013 (295) 3:34:43.01330
abspath	Dan Quinlan			Kent Lindquist		c		10/22/2013 (295) 3:34:43.01336
acro	Kent Lindquist			Kent Lindquist		perl		10/22/2013 (295) 3:34:43.01339
ah2db	Geoff Abers	contrib				c		10/22/2013 (295) 3:34:43.01341
alter_timestamps	Dan Quinlan					c		10/22/2013 (295) 3:34:43.01343
altus2orb	Danny Harvey			Danny Harvey		c		10/22/2013 (295) 3:34:43.01345
altusevt2db	Danny Harvey			Danny Harvey		c		10/22/2013 (295) 3:34:43.01347
antelope_admin	Danny Harvey			Danny Harvey		c		10/22/2013 (295) 3:36:33.91014
antelope_update	Dan Quinlan			Kent Lindquist		perltk		10/22/2013 (295) 3:34:43.01353
aqatcl	John Ousterhout	thirdparty				c	Custom TCL/Tk	10/22/2013 (295) 3:34:43.01355
aqawish	John Ousterhout	thirdparty				c	Custom TCL/Tk	10/22/2013 (295) 3:34:43.01358
asnap	Dan Quinlan					perl		10/22/2013 (295) 3:34:43.01360
assoc_rayleigh	Frank Vernon	contrib				perl		10/22/2013 (295) 3:34:43.01361
atcl	John Ousterhout	thirdparty		Kent Lindquist		c	Custom TCL/Tk	10/22/2013 (295) 3:34:43.01363
atws_calc_mwp	James Stewart	contrib				perltk		10/22/2013 (295) 3:34:43.01365
autodrm	Dan Quinlan					perl		10/22/2013 (295) 3:34:43.01367
autodrm2db	Dan Quinlan					perl		10/22/2013 (295) 3:34:43.01369
autodrm_wrapper	Dan Quinlan					csh		10/22/2013 (295) 3:37:23.37890
automrc	Marina Glushko	contrib				c		10/22/2013 (295) 3:38:09.65449
awish	John Ousterhout	thirdparty		Kent Lindquist		c		10/22/2013 (295) 3:34:43.01373
baler2db	Frank Vernon	contrib				perl		10/22/2013 (295) 3:34:43.01375
baler_active_to_dmc	Frank Vernon	contrib				perl		10/22/2013 (295) 3:34:43.01378
baler_msd_proc	Frank Vernon	contrib				perl		10/22/2013 (295) 3:34:43.01380

BRTT

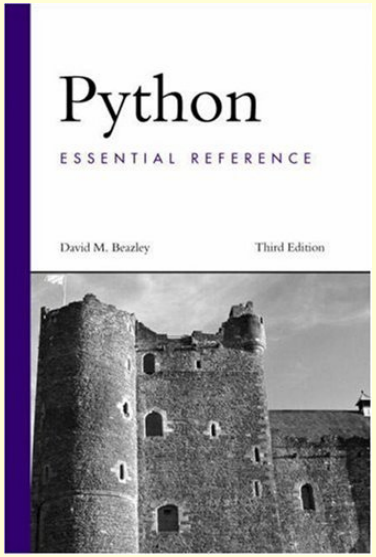
Antelope 5.3

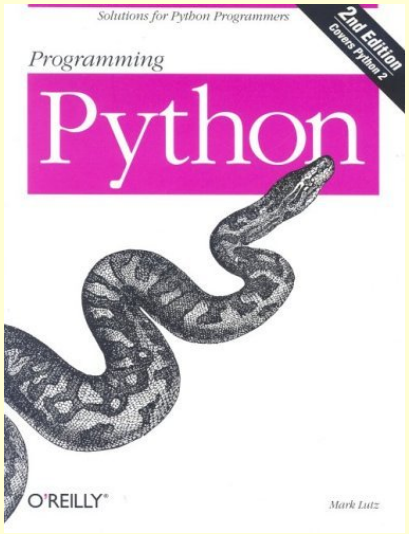
BRTT

November 2013

Python

- Python: Object-oriented scripting language
  - http://www.python.org
  - Dynamic
  - Powerful
  - Extensible
  - Fast





About Python

- http://www.python.org/about:
  - Very clear, readable syntax
  - Strong introspection capabilities
  - Intuitive object orientation
  - Natural expression of procedural code
  - Full modularity, supporting hierarchical packages
  - Exception-based error handling
  - Very high level dynamic data types
  - Extensive standard libraries and third-party modules for virtually every task
  - Extensions and modules easily written in C, C++ (or Java for Jython, or .NET languages for IronPython)
  - Embeddable within applications as a scripting interface

Why Bother With Python?

- Heavily used “modern” object oriented scripting language
- Used extensively in Australia and US Antelope communities
- Can find young software developers who know Python (not so much the case for Perl and TCL)
- Has a large and comprehensive set of public-domain extensions, including scientific/engineering extensions
- Performs similar to Perl
- Unlike Perl, is inherently OO
- Unlike Perl, provides a simple path for Tk widget extensions
- Helps to prevent BRTT fossilization

Python in Antelope: History I

- Initial impetus: PASSCAL Instrument Center
  - Some pieces; Not a generic interface
- 2007: First open-source version, IRIS/ANF
  - Datascope; waveform plotting, orbtopo
  - Good proof-of-concept; lots of routines missing
  - Advice from Alex Clemesha, Rob Newman
- 2008: GA Consulting on Python
  - Ole Nielsen, Nariman Habili, Phil Cummins, Spiro Spiliopoulos, Michael Potter
  - Thin C layer with Python intelligence in script
  - Better architecture; warts and missing pieces

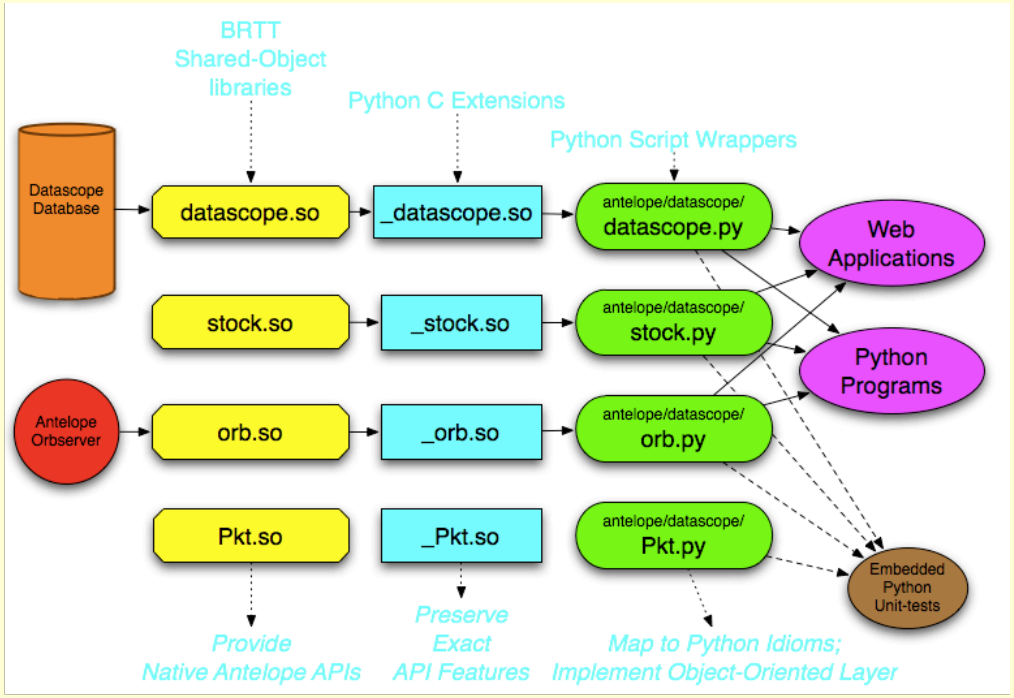
Python in Antelope: History II

- 2009: Added python orb, Pkt functions for UCSD
  - Experiment with AMQP for OOI
  - Filling out interfaces
  - open-source and integration issues
  - Discussions of heavy rewrite / expansion through GA
- 2010-2011: pre-release *Oryx*
  - Rtwebserver, rtcache
  - Headed towards Lindquist Consulting, Inc. Product
  - Never materialized as independent product: KL->BRTT

Python in Antelope: History III

- 2012: BRTT, first commercial version.
  - Python interpreter shipped with Antelope
  - raw, scripted layers separate
  - Docs; functional basic toolkit
  - Peregrine
  - Solid raw layer; glitches in scripted layer, divergent open-source developments
- Beg. 2013: Script-layer rewrite by Jeff Laughlin, Laughlin Consulting
  - Pkt, stock, orb, brttpkt, elog
  - In Antelope 5.3
- Summer 2013:
  - More Jeff Laughlin rewrites: Datascope, coords
  - Advanced Tk utilities, buplot

Python Interface Structure



BRTT

Multiple Layers

- Raw layer
  - _function naming convention: not for general use
  - Slavish adherence to C return values and structure
  - No Python intelligence
- Scripted layer
  - Intended for general user
  - On top of raw layer
  - Implements the ‘feel’ of Python

Antelope 5.3: Python

- Standard python 2.7.2 64-bit interpreter in Antelope 5.3 release
- Also included are a set of public-domain Python modules
  - numpy, matplotlib, twisted, setuptools
- Also included are a set of Antelope extensions to Python in the same vein as the TCL and Perl Antelope extensions
  - "raw" interfaces that closely following C calling syntax and use
  - New OO interfaces that follow standard python paradigms
- Most new GUI tools will be developed using python

Antelope 5.3: Python

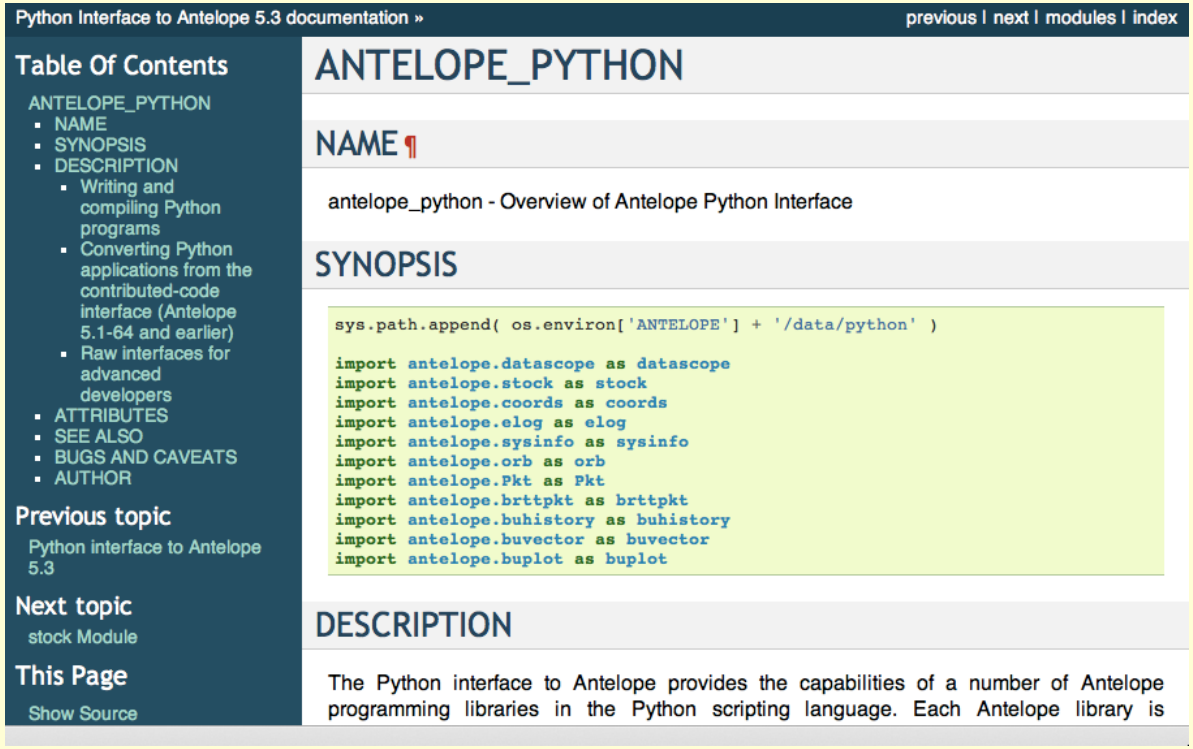
- Rewrites by Jeff Laughlin, Laughlin Consulting
- Scripted layer libraries for Antelope 5.3:
  - Orb
  - Pkt
  - BRTTPkt
  - Stock (parts)
- Object Oriented
- Context Managers
- Iterators
- Heavily documented examples
- For Antelope 5.4:
  - Datascope
  - Coords

Requirements for project pyext:update;1
General goals:

- State-of-the-art Python interface for Antelope that hides C ugliness from Python programmer
- Appropriate object structure and behaviors
- Appropriate exception hierarchy and behavior
- Seamlessly handle memory management issues
- Seamlessly hide C-structure interaction, copying, passing, allocation/ deallocation
- Succeeds at adoption by Python / Antelope community (inspires app development, not interface rewrites)
- Result must be straightforward to maintain and extend by BRTT (i.e. passes BRTT acceptance)
- Provides demonstration template, model for how to write wrappers for remaining Antelope libraries

Hyperlinked Sphinx Docs

- file:///opt/antelope/5.3/html/antelope_python_overview.html



Python Interface to Antelope 5.3 documentation » previous | next | modules | index

Table Of Contents

ANTELOPE_PYTHON
- NAME
- SYNOPSIS
- DESCRIPTION
  - Writing and compiling Python programs
  - Converting Python applications from the contributed-code interface (Antelope 5.1-64 and earlier)
  - Raw interfaces for advanced developers
- ATTRIBUTES
- SEE ALSO
- BUGS AND CAVEATS
- AUTHOR

Previous topic

Python interface to Antelope 5.3

Next topic

stock Module

This Page

Show Source

ANTELOPE_PYTHON

NAME ¶

antelope_python - Overview of Antelope Python Interface

SYNOPSIS

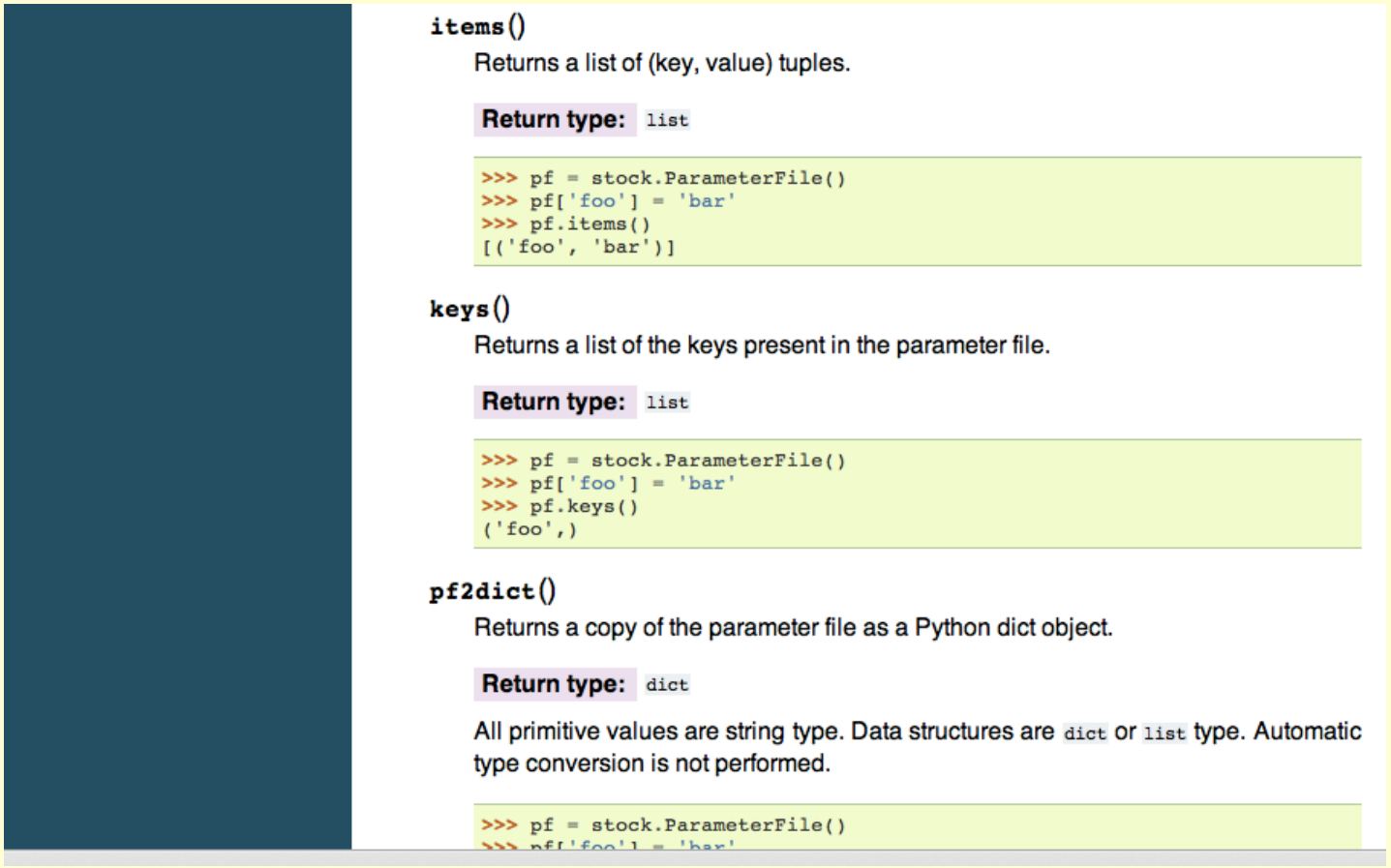
```
sys.path.append( os.environ['ANTELOPE'] + '/data/python' )

import antelope.datascope as datascope
import antelope.stock as stock
import antelope.coords as coords
import antelope.elog as elog
import antelope.sysinfo as sysinfo
import antelope.orb as orb
import antelope.Pkt as Pkt
import antelope.brttpkt as brttpkt
import antelope.buhistory as buhistory
import antelope.buvector as buvector
import antelope.buplot as buplot
```

DESCRIPTION

The Python interface to Antelope provides the capabilities of a number of Antelope programming libraries in the Python scripting language. Each Antelope library is

Hyperlinked Sphinx Docs



items()

Returns a list of (key, value) tuples.

Return type: `list`

```
>>> pf = stock.ParameterFile()
>>> pf['foo'] = 'bar'
>>> pf.items()
[('foo', 'bar')]
```

keys()

Returns a list of the keys present in the parameter file.

Return type: `list`

```
>>> pf = stock.ParameterFile()
>>> pf['foo'] = 'bar'
>>> pf.keys()
('foo',)
```

pf2dict()

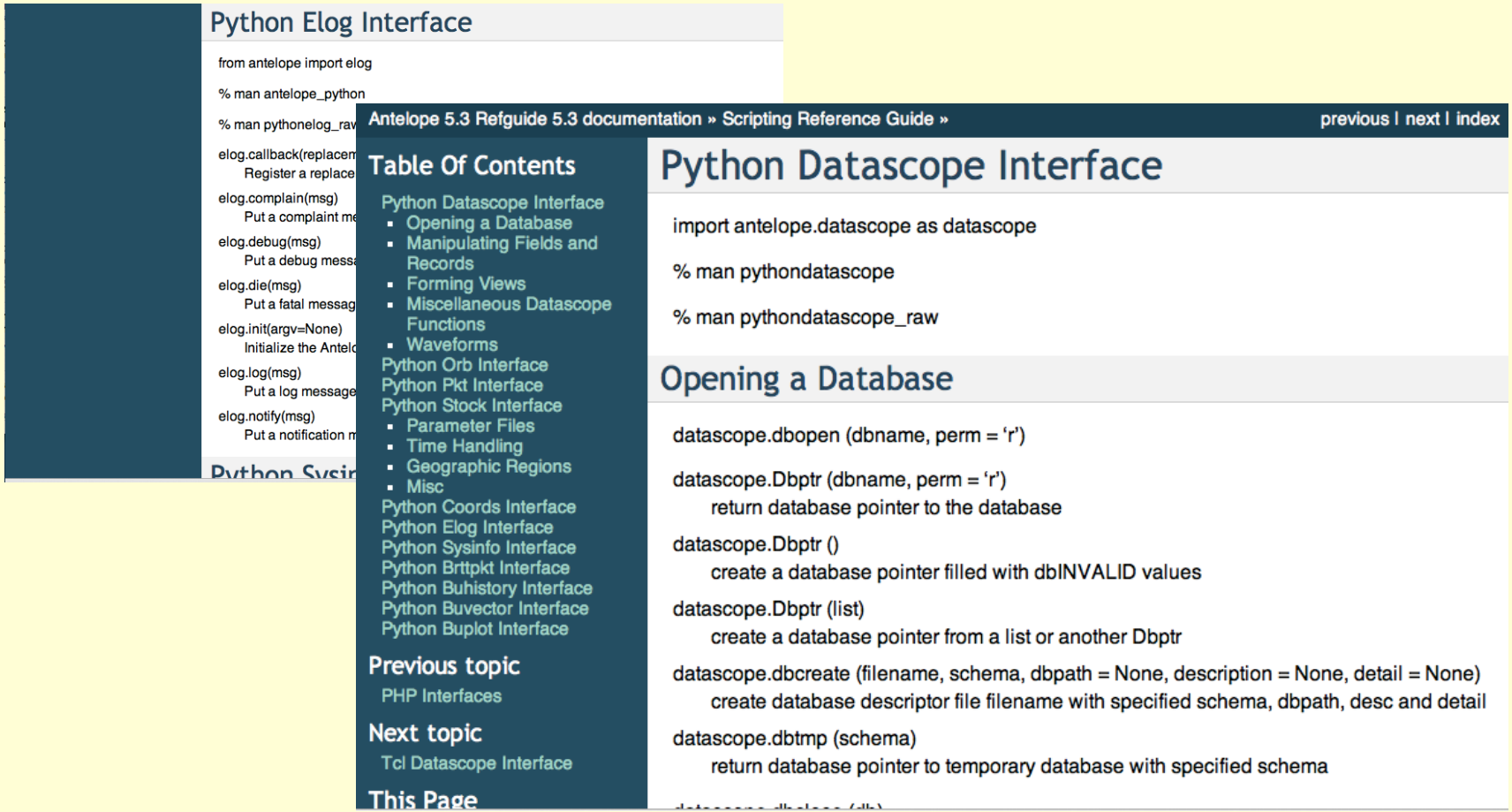
Returns a copy of the parameter file as a Python dict object.

Return type: `dict`

All primitive values are string type. Data structures are `dict` or `list` type. Automatic type conversion is not performed.

```
>>> pf = stock.ParameterFile()
```

Online Refguides



Python Elog Interface

from antelope import elog

% man antelope_python

% man pythonelog_raw

elog.callback(replacem
Register a replace

elog.complain(msg)
Put a complaint me

elog.debug(msg)
Put a debug messa

elog.die(msg)
Put a fatal messag

elog.init(argv=None)
Initialize the Antelo

elog.log(msg)
Put a log message

elog.notify(msg)
Put a notification m

Python Sysin

Antelope 5.3 Refguide 5.3 documentation » Scripting Reference Guide »

previous | next | index

Table Of Contents

- Python Datascope Interface
  - Opening a Database
  - Manipulating Fields and Records
  - Forming Views
  - Miscellaneous Datascope Functions
  - Waveforms
- Python Orb Interface
- Python Pkt Interface
- Python Stock Interface
  - Parameter Files
  - Time Handling
  - Geographic Regions
  - Misc
- Python Coords Interface
- Python Elog Interface
- Python Sysinfo Interface
- Python Brttpkt Interface
- Python Buhistory Interface
- Python Buvector Interface
- Python Buplot Interface

Previous topic

PHP Interfaces

Next topic

Tcl Datascope Interface

This Page

Python Datascope Interface

import antelope.datascope as datascope

% man pythondatascope

% man pythondatascope_raw

Opening a Database

datascope.dbopen (dbname, perm = 'r')

datascope.Dbptr (dbname, perm = 'r')
return database pointer to the database

datascope.Dbptr ()
create a database pointer filled with dbINVALID values

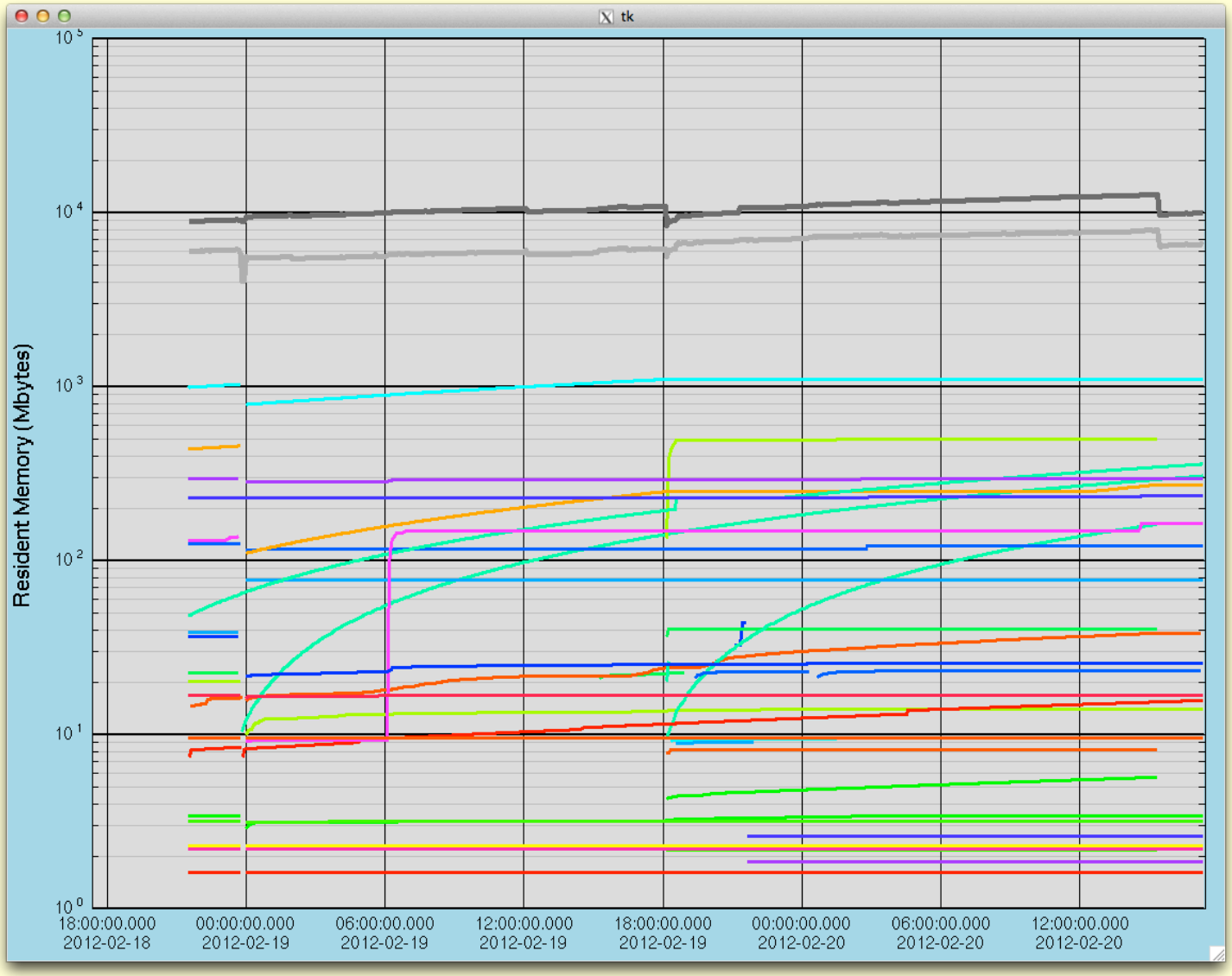
datascope.Dbptr (list)
create a database pointer from a list or another Dbptr

datascope.dbcreate (filename, schema, dbpath = None, description = None, detail = None)
create database descriptor file filename with specified schema, dbpath, desc and detail

datascope.dbtmp (schema)
return database pointer to temporary database with specified schema

BRTT

Python-based Tool for Monitoring Memory

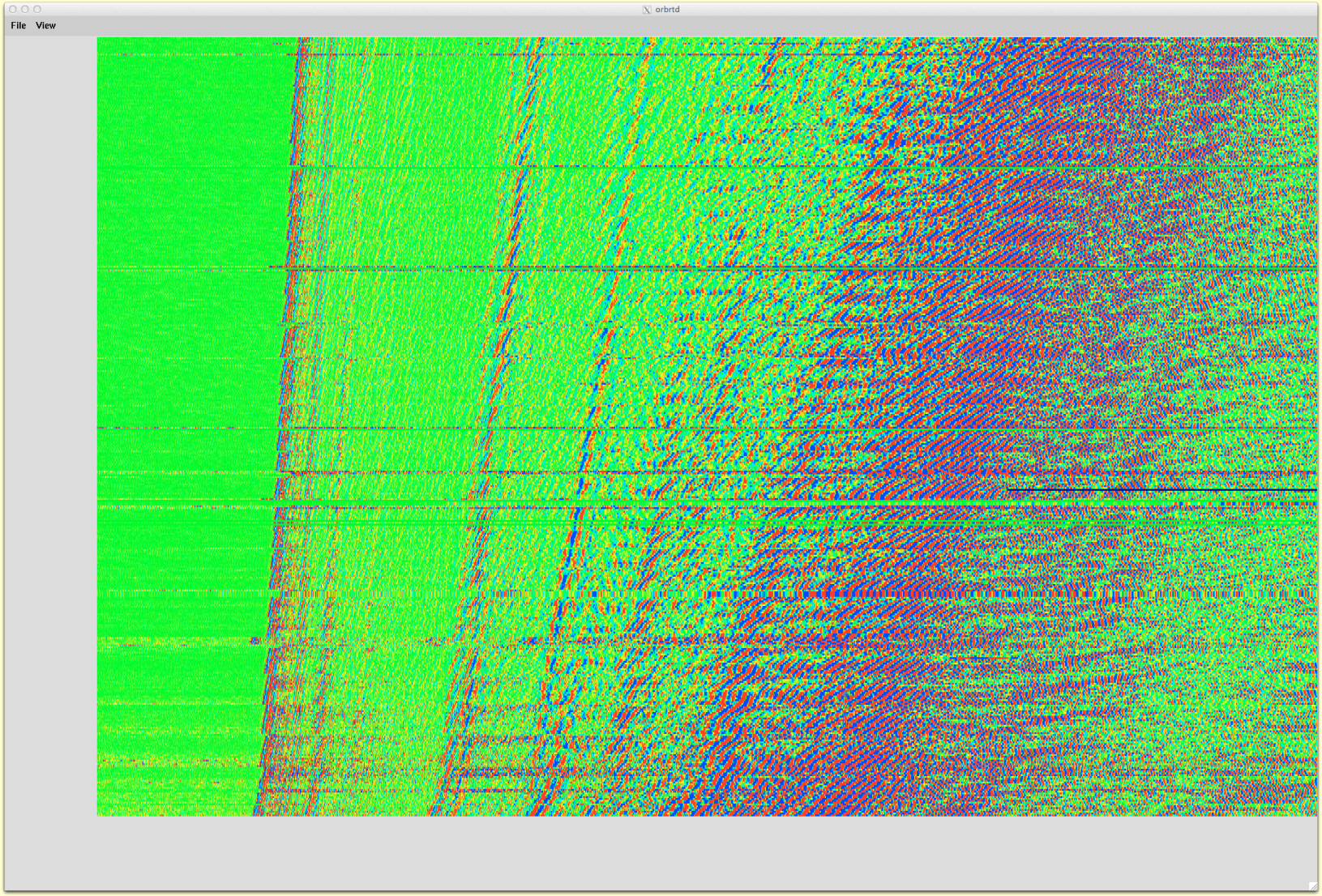


BRTT

orbrtd

- **orbrtd** is a complete rewrite of **orbmonrtd**
- Rewrite of old TCL/Tk script as a Python script
- Adaptation of *buplot bptrace* Tk canvas item extension available in python
- Provides enhanced trace amplitude plotting options (color, log scales, etc.)
- Provides capability to plot color-contoured spectragram style time-scrolling spectra plots
- Introduces a number of new features, including dynamic automatic channel configurations
- First stage in converting **dbpick** display graphics

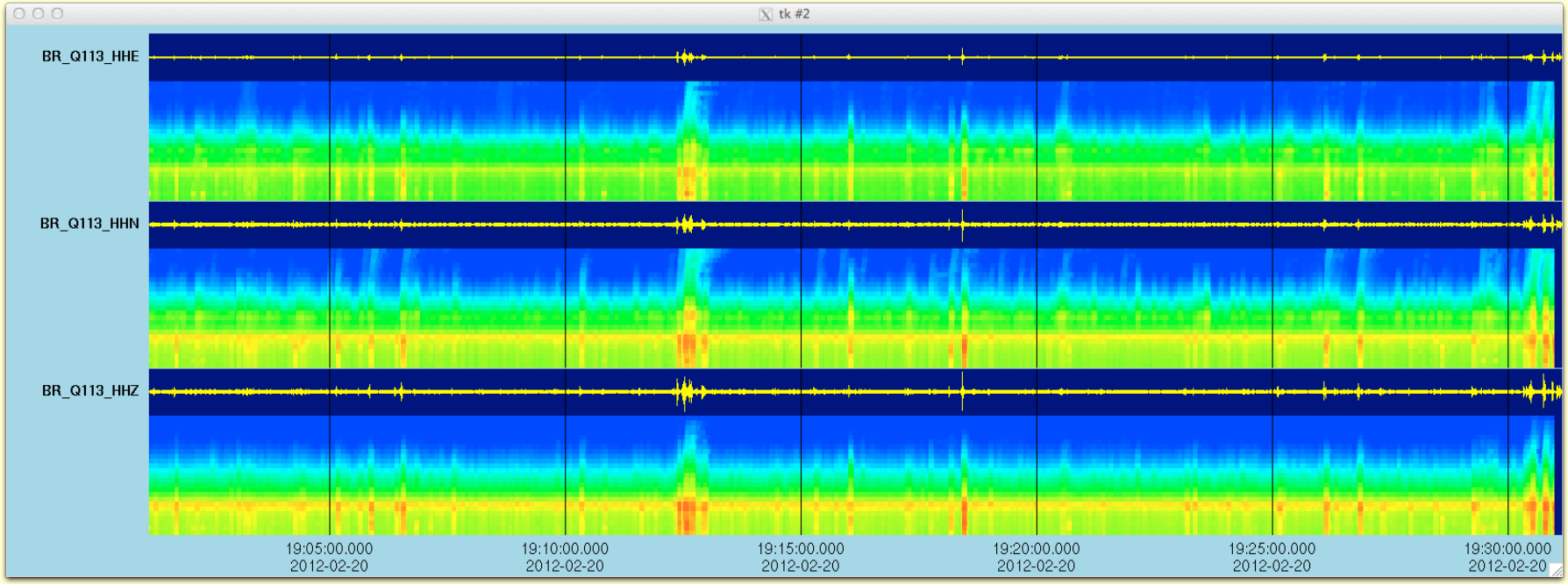
Orbrtd: colorscale display



BRTT

November 2013

orbrtd: spectragram display



BRTT

November 2013

Antelope 5.2 **dbpick** Capabilities

- Complete rewrite of underlying data handling middleware
- Overall data access performance increased by a factor of 10+ (with caveats)
- Highly dynamic response to changing underlying database
- New type-in commands, `dbrefresh` and `dbreopen`
- New type-in command, `batch`, to increase graphics performance and to control display flashing when changing events
- Can keep a **dbpick** window open continuously when running **dbevents**
- First phase toward a complete rewrite of **dbpick**

Future Antelope Development

- **`orbrtd`**
  - Add display of arrivals and detections
  - Enhanced autoscaling
- **`dbpick`**
  - Next phase is to migrate the GUI to our new tk-based graphics extensions
  - Develop new **`dbpick`** main program as a python script
- **`dbloc2`**
  - Kent will head a complete rewrite
  - Taimi Mulder and Trilby Cox led comments/suggestions solicitation for new **`dbloc2`**
  - Python script with embedded **`dbpick`** functionality
- **`dbe / dbhelp`**
  - Jeff Laughlin will rewrite

New Products from BRTT!

BRTT

November 2013

Peregrine

Web enhanced version of Antelope

Bighorn

Strong motion/structure monitoring version of Antelope also including web enhancements

Peregrine

- Python-based web server
- Along with the various python extensions, provides a comprehensive toolkit for developing custom web servers that are highly integrated into the Antelope environment (configuration, connectivity, etc.)
- Along with other components, will be sold as a separate BRTT product or as an add-on to existing Antelope site licenses
- Current version is not ready for release. We are probably one year away from a production version.

Bighorn - `orbsmrsp`

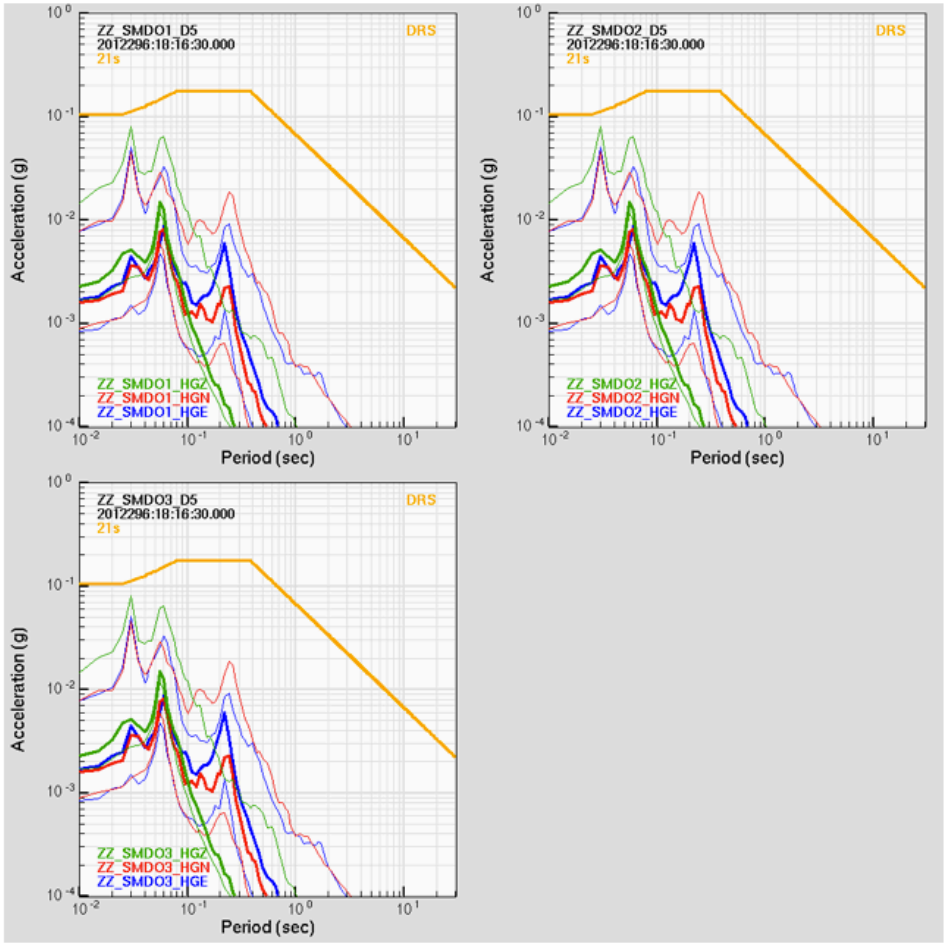
- New ability developed for producing continuous time-dependent strong motion response spectra
- Expanded floating point data representations within ORB packets and Datascope waveform files
- Pf ORB packets to represent time continuous strong motion response spectra
- Provides a very fast method for computing continuous time-dependent response spectra for large numbers of channels

BIGHORN WEB DISPLAY SYSTEM

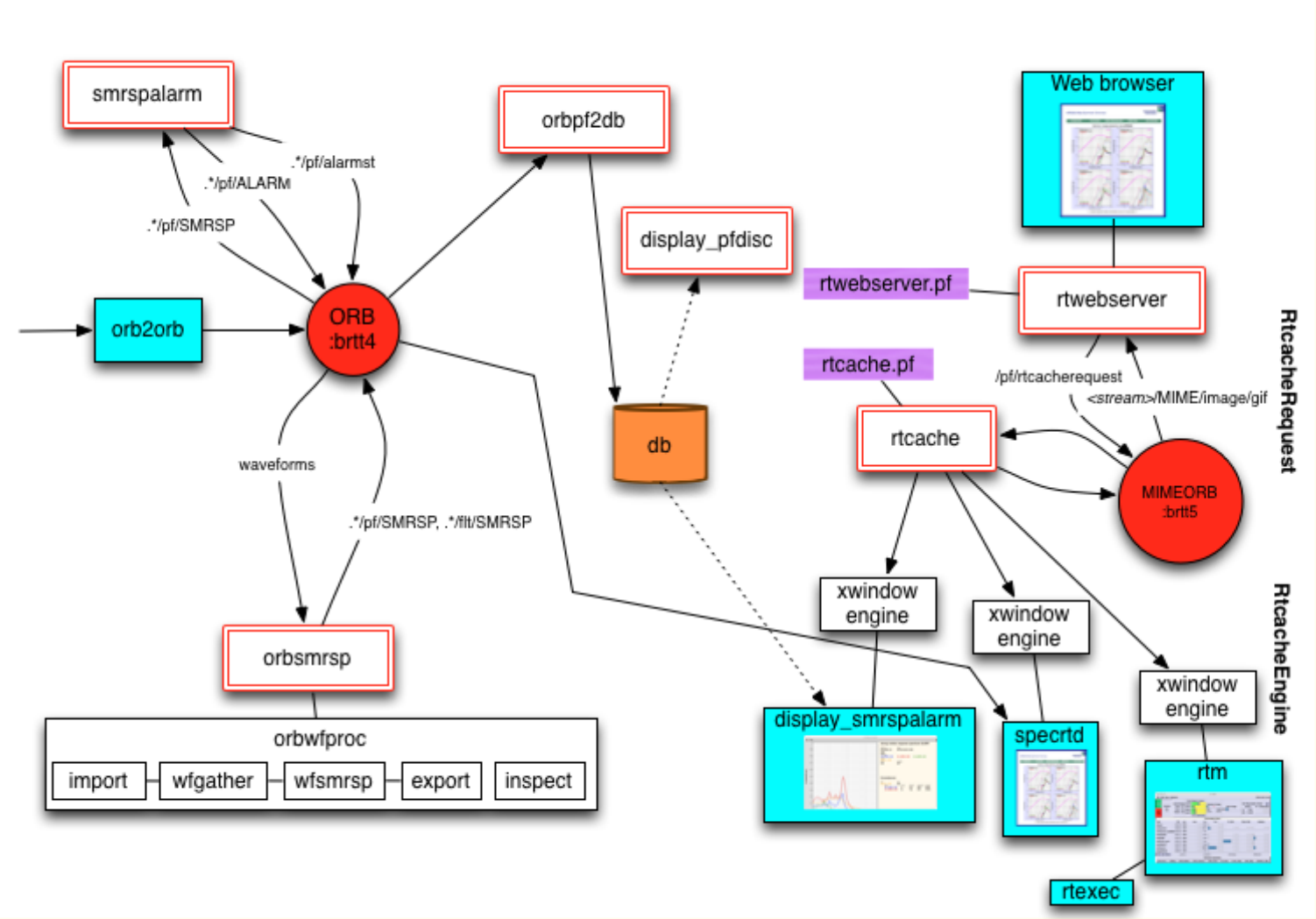
OVERALL | FACILITY MAP | SPECTRA | FACILITY SOH

SMDEMOO FACILITY STATUS: NORMAL

OUTPUT FROM SPECRTD FOR FACILITY: SMDEMOO FACILITY



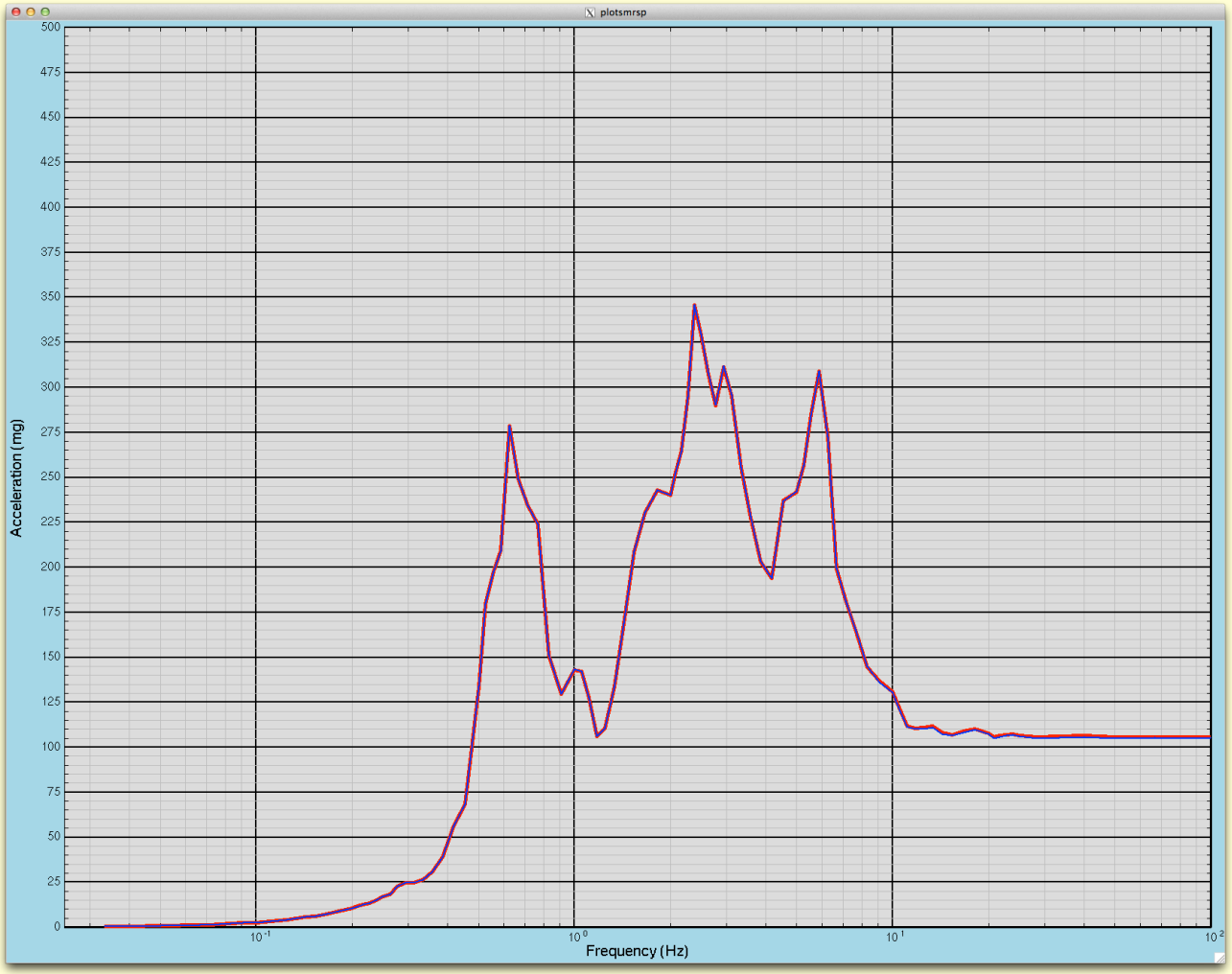
Bighorn: architecture



BRTT

November 2013

Bighorn:
overlay of new vs traditional processing



BRTT

November 2013

Thank You

css3.1 v. 5.2

- New incremental changes to **css3.0** as we discussed last year
  - Mainly increased lengths of various attributes
  - All id attributes from 8 to 12 characters
  - All epoch times have microsecond precision
  - dnorth, deast in site have 5 digits precision
  - lat, lon have 7 digits precision
  - Increase dir to 80 characters and dfile to 48 characters
  - Increase sta to 8 characters and chan to 14 characters
- Increased sta attribute size means that SEED to css aliasing can be done with consistent snet_ssta type naming convention
- Intention to use **dbconvert** to convert from **css3.0** to **css3.1**
- New `gsn_demo` uses **css3.1**
- Addition of `prefmag` attribute in `event` table
- Not all critical programs are compatible were compatible

`css3.1` v. 5.3

- Lots of problems with initial implementation
- Initial implementation hard wired SEED to css name aliasing and dropped snetsta and schanloc tables
- This insured incompatibility between existing **`css3.0`** and **`css3.1`** databases
- Cannot use **`dbconvert`** to convert from **`css3.0`** to **`css3.1`**
- Fixed problems by re-introducing snetsta and schanloc tables. Issues described in **`cssconversion(5)`** man page. Helper script, **`cssconvert`**, provided for dealing with changes in SEED to css name aliasing during conversion.
- New `gsn_demo` now goes back to using **`css3.0`**
- Now works with all critical programs