

Exploration & Production **Technology**

delivering breakthrough solutions



FreeUSP & FreeDDS

Richard Clarke

Outline

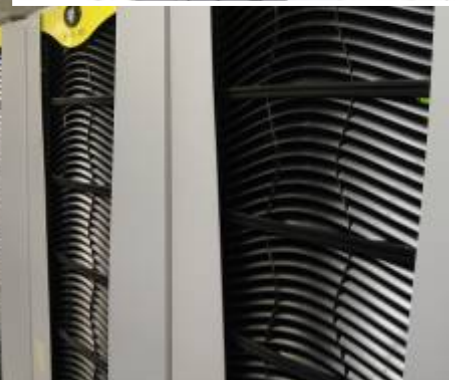
- Introduction
 - why open source?
- FreeUSP
- FreeDDS
 - using
 - developing new applications
- Looking to the future

Why Open Source ?

- FreeUSP was released as open source in 2001.
 - FreeDDS followed in 2003.
 - “to foster education, understanding and collaboration”
 - collaboration with academia
 - technology deployment via contractors
- fast application of technology

Industry scale research...

- USP & DDS enable ideas to be tested, verified, then scaled
- 99% of the usage of one of the world's largest research computing systems
 - used for applied research – NOT production processing
 - 450 teraflops, almost 5000 times bigger than 1999
 - 3500 computers with over 40,000 CPUs
 - 230 terabytes of physical memory
 - 15 petabytes of raw disk storage
 - optimized systems:
 - computers with big memory allow fast development
 - clusters to allow cost-effective scaling
 - team of geophysicists, mathematicians, computer scientists



FreeUSP

- “Unix Seismic Processing”
- Started in ~1980
- Used for processing Amoco/bp data for ~30 years
- Unix based
- Command-line driven (there is a GUI, but no-one uses it)
- Unix like syntax

```
utop -N input.usp -O output.usp -dt 4
```

- ~600 codes, mostly in signal analysis and preprocessing
- 3D cubes, designed for fast access
- Similar format to SEG Y
 - Line header, with processing history
 - 256 byte trace header, followed by samples

FreeDDS (next release coming in a few days)

- “Data Dictionary System”
- Started in ~1990
- Used for processing & imaging Amoco/bp data for ~20 years
- Unix based
- Command-line driven
- SepLib/SU/Madagascar like syntax
 - `editd in= input 3s=100 3e=110 out=records_100_110`
- ~100 codes in FreeDDS, ~400 not yet released, annual updates
- Covers signal analysis, preprocessing, imaging...
- Up to 9 dimensional cubes, designed for fast access
- There is no “DDS format”!

“There is no DDS format!”

- DDS is heavily influenced by SEPlib, so resembles Madagascar
- Normal use is with a dictionary file + binary file
 - Dictionary is text; line header type info describing the binary
 - Binary file can be fcube (samples only), USP, SEG Y, SU, ASP,...
- Autodetects format
 - No dictionary needed for USP, SEG Y.
- Every program can change format on the fly
 - `compute in= input.usp scale=10 out_format=seg y out_data=output.seg y`

Mixing Software Packages: DDS+USP+SU+...

- We can pipe between different software packages
 - DDS can act as the glue between processing packages
- Often preprocessing uses USP format because we need headers, then we use fcube (we separate the headers) when we are imaging.

```
editd \  
  in= input.segy 3s=100 3e=200 \  
  out_format=usp out_data=stdout: |\
```

DDS: select records 100-200

```
utop \  
  -dt 4 |\
```

USP: set the sample rate to 4ms

```
bridge \  
  out_format=su out_data=stdout: |\
```

DDS: convert to SU

```
suximage \  
  legend=1 ...
```

SU: display the data

Multi-file Support

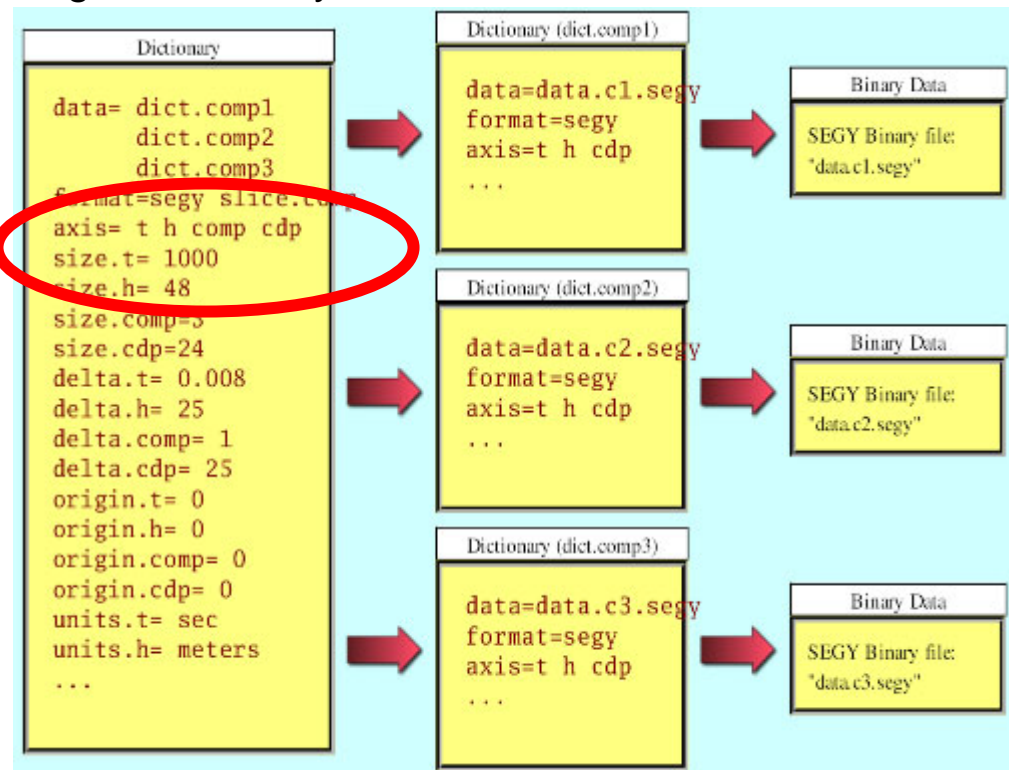
- The data can be in several different files
 - “sliced” across 1 axis (e.g. separate offset volumes),
 - or across several axes (e.g. organized as many sub-cubes)
- The dictionary allows to access it as 1 big file, or many small files.

Change the axis order* to change the effective order of the data:

axis=t comp h cdp

axis=t h comp cdp

axis=t h cdp comp

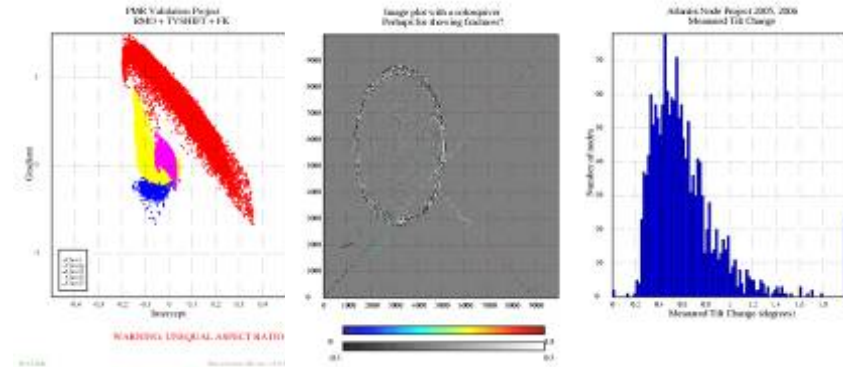


*on the command-line or by editing the dictionary. For example:

editd in=input.segy axis=t x y h size.y=10 size.h=10 4e=10 | compute ...

Viewing Data: command line

- ddsplot
 - Emulates most matlab plots, plus a few extra
 - Can create plots on top of plots
 - Reads DDS/ascii; writes post-script
- Joe Dellinger uses vplot
- SU Tools
 - Wrapper scripts that convert the data to SU format:
“bridge in=\$in out_format=su out_data=stdout: | suximage legend=1...”



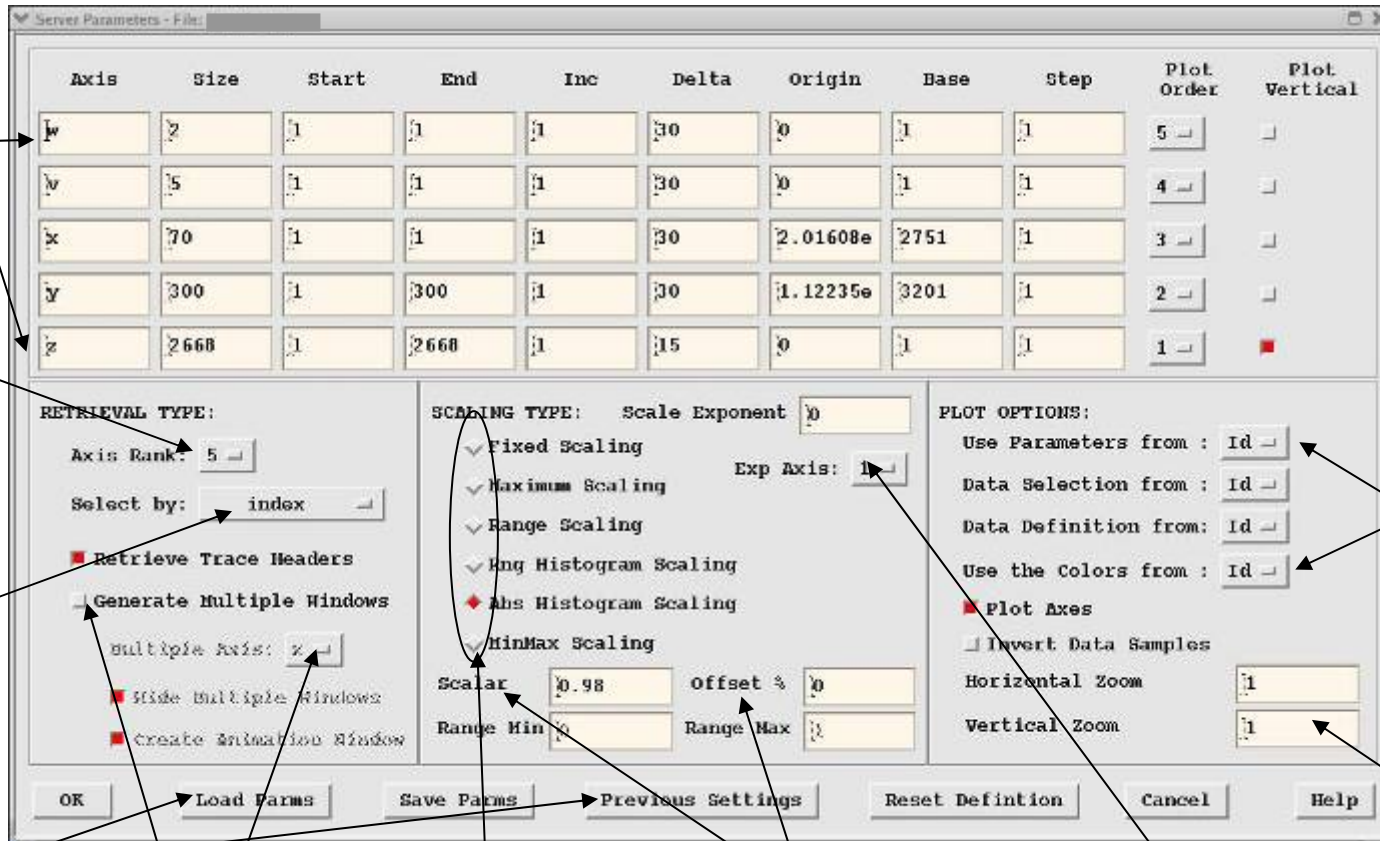
Viewing Data: GUI

- XSD
 - 2D Graphical viewer
 - Data server/client – designed for use with a remote HPC and low bandwidth
 - Interleave, animations, picking,...

Note: watch for missing X libraries during installation that can cause XSD to not compile!

Displaying data: XSD

- Once seismic data is selected to be opened, the following window appears:



data selection for display

dimensions of data file

option to select data based on axis index, base/step or delta/origin

use previous or saved parameters

to open an animation window

scaling method

for fixed scaling display

scaling exponent along any axis

to select parameters from another window already open

data zoom options

Current DDS applications... +30 in next release

AutoHTMLIndex	Automatically build HTML index	fd_deriv	Frequency domain	nmo	Apply/re move hyperbolic normal moveout	wtstk	apply weights and stack traces (DDS)
ChkMCHheaders	check the headers for anomalies	fill	Read surfaces and fill	obsazcheck	vector fidelity correction of multi-component	xsd	an X window seismic display program (DDS)
asciidump	Utility to dump binary data	fkdm0	3D common-offset	pad3d	Pad first 3 axes of an N-dimensional dataset	z2d_horizon	depth correct a 3D volume (DDS)
boxdesign	An interactive aid to design	fkidm0	3D common-offset	pick2dds	interpolate xsd picks over 2D surface (DDS)		
bridge	Convert seismic data between	freq	frequency transform	radar	Interactive Box Wave Test Analysis (DDS)		
checkpolarity	check that the hydrophone	gensort	General sorting on	resetlive	Reset live header flag in geometry file (DDS)		
chgaxis	Rename dds axis names	green	traveltime maps from	restore_hdr	Restore trace headers from a separate (DDS) dataset		
chkdata	Check/Correct data for errors	grid2lndmrk	Create Landmark a	reverse	reverse order any selected axes (DDS)		
compute	general trace computation	gridt2d	Grid Time-to-Depth	rotatexy	3D coordinate rotation (DDS)		
create_hdr	Create trace headers to	horizon_extract	extract either a wide	rsamp2d	resample first 2 axes of ND dataset (DDS)		
ddsServer	remote data server for x	interp_headers	interpolate arbitrary	rsampND	resample any axes of ND dataset (DDS)		
dds_defn	- Utility to return a definition	lndmrk2grid	create dds grid from	save_hdr	Save trace headers to a separate (DDS) dataset		
dds_if	Conditionally change header	makeditct	- Utility to make a	skewrot	Apply a skew rotation on traces (DDS)		
dds_in	- Utility to return axes in	makemodel	make simple test volume	slicecube	slice a cube from the values in a surface (DDS)		
ddsarg	a utility program for building	mbs-list	- Utility to list all	slicevol	Slice horizons out of a volume (DDS)		
ddscan	Interactive dds scan utility	mbs-show	- Utility to list any	smooth3d	smooth on any of 1st 3 axes (DDS)		
ddsplo	emulates the matlab plot	mbs-which	- Utility to list the	sort4coamig	sort data for Vxyz 3D common-offset migration (DDS)		
ddstats	print out statistics of a	mccampbalance	record consistent s	surfs2volume	interpolate volumes from surfaces (DDS)		
dmo3d	3D dmo (DDS)	mccappfill	apply a multi-channel	taper3d	Apply a taper on any of first 3 axes (DDS)		
dmofold	pre 3D dmo fold determination	mcclean	kill multi-component	tdconv	Time-to-Depth/Depth-to-Time Conversion (DDS)		
dmute	Apply mute (DDS)	mccrecon	record consistent n	transpose	General transpose program (DDS)		
dnoise	spectral balancing noise	mccsplit	split multiplexed O	trcpik	Interactive first break picking on traces (DDS)		
dstack	Stack data along any selected	memorysort	in memory sorting	tscan	Tape Scan to Determine Tape/Data Format (DDS)		
editd	General N-dimensional editing	mergeND	Multidimensional d	varotate	rotate H1 and H2 components to radial and transverse. (DDS)		
efficientz	Resample Vxzy file for efficiency	mutegather	apply mute to gather	velfill	Fill velocity between two surfaces (DDS)		
expandz	resample between irregular	mutesalt	mute selected traces	vzd2t	V(z) time to depth/depth to time moveout (DDS)		
extractND	extract traces from one	mutestk	apply mute and velocity	vzfkcoamig	3D Vx/3A common offset common midpoint non stack depth migration (DDS)		

- 
- If you are not a software developer, then you can start to nap now....

Writing a DDS Program

- API's in C & Fortran
 - Template examples with “core” DDS functions on the website
 - Matlab API in progress
- Compilers
 - Requires a recent version of gfortran (that supports pointers) or Intel, PGI,...
- Not thread safe
 - “Those geophysicists would shoot themselves with it anyway” - Comp. Sci Guru
 - Many codes use both OpenMP & MPI
- MPI
 - Template, convenience routines
 - Parallel input and output
 - Largest OpenMP+MPI job to date had 20,000 threads/processes.

Cat Herding...

“It compiled and ran. I’m done!”

- Documentation
 - Man pages required!
 - Automated template man page
- Regression testing
 - On-line test results (run with memory checking)
- Run-time Logging
 - Internal only!
 - Connections (*who has used the program I need help with?*)
 - “Risk” warning (*production/test/where did you get this?*)

What you are probably thinking...

“That all sounds complicated.”

- It's not!
- Let's look at a simple Fortran example ignoring headers...
 - record by record processing
 - can read input files of different formats
- Using headers is only slightly more complicated
 - requires extra trace buffer, and calls to extract headers/samples

Reading and writing parameters...

```
#include <fdds.h>
```

```
if( fdds_openpr(prog,") .gt.0 ) call help()
```

Open the print file, unless the user specified help= or -?

```
ier=fdds_dict('par:', 'scan')
```

Scan from the parfile or command-line (not an input file)

```
verbose = (fdds_switch('verbose',0) .eq.1)
```

Logical parameter: yes/no/true/false

```
ntaper=10
```

```
ier=fdds_scanf('ntaper nt', '%d\0', ntaper)
```

```
ier=fdds_printf('ntaper=%d\0', ntaper)
```

Read integer parameter, with synonym. Write value to log file.

```
ptr_taper=fdds_malloc8(dble(ntaper)*SIZEOF_REAL)
```

Allocate (maybe >2Gb)

Open the input and output...

```
fd_in=fddx_in('in','stdin:', 'title')
if (fd_in.lt.0) then
  ier=fdds_prterr('Unable to open input data file!\n\0')
else
  ier=fdds_scanf('size.axis(1)', '%d\0', n1)
  ier=fdds_scanf('size.axis(2)', '%d\0', n2)
  n3=fdds_axis_prod(3)
endif
```

Open the input, look for in= *filename*.
Default to stdin:

```
fd_out=fddx_out('out','stdout:', 'title', fd_in)
if (fd_out .lt. 0) then
  ier=fdds_prterr('Unable to open output!\n\0')
endif
```

Open the output, look for out= *filename*.
Default to stdout: Use the input file for the
history, and defaults.

```
If ( fdds_errors().gt.0 ) fdds_closepr()
```

Stop if there were errors.

Reading and writing data...

```
do i3 = 1,n3
```

```
  if (fddx_read(fd_in,record,n2).ne.n2) then  
    ier=fdds_prterr('Reading record %d\n\0',i3)  
    ier=fdds_closepr()  
  endif
```



Read n2 traces into array *record*.

```
  call do_some_stuff()
```

```
  if (fddx_write(fd_out,record,n2).ne.n2) then  
    ier=fdds_prterr('Writing record %d\n\0',i3)  
    ier=fdds_closepr()  
  endif
```



Write n2 traces from array *record*.

```
end do
```

```
ier=fdds_closepr()
```



Close any open files. Stop.

Simple makefile...

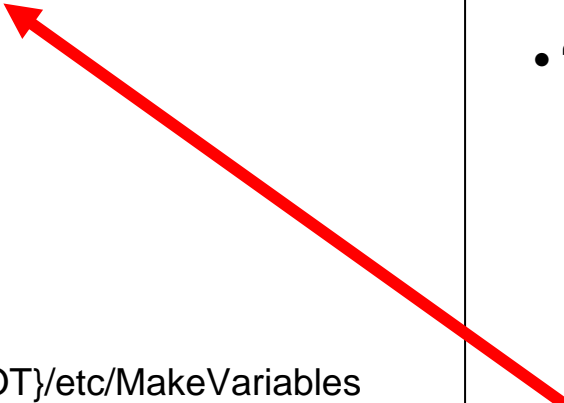
```
Name := my_program
FSrcs := my_fortran_code.F

#DEBUG := TRUE
#WARN := TRUE
#CHECK := TRUE


MP := TRUE
MPI := TRUE

include ${DDSROOT}/etc/MakeVariables
include ${DDSROOT}/etc/MakeRules
```

- “make” to compile
- “make help” for options
- “make newestest” to create a test template
- “make tests” to run the tests
- “make man” to create a template man page



Useful flags for debugging
(commented out here)

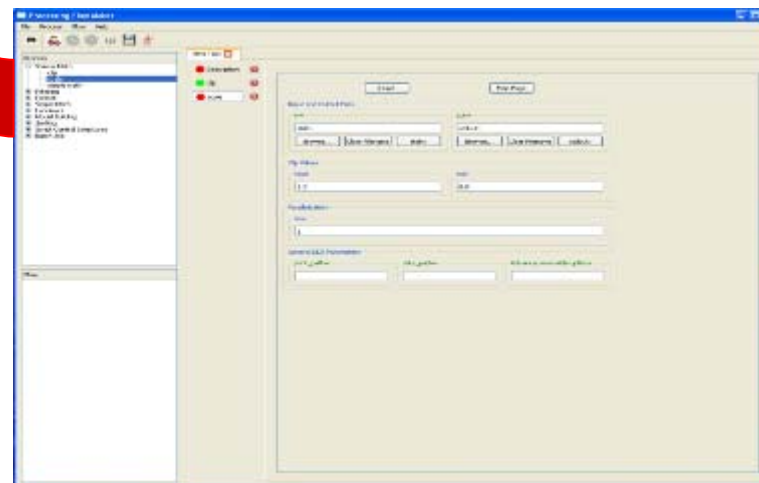
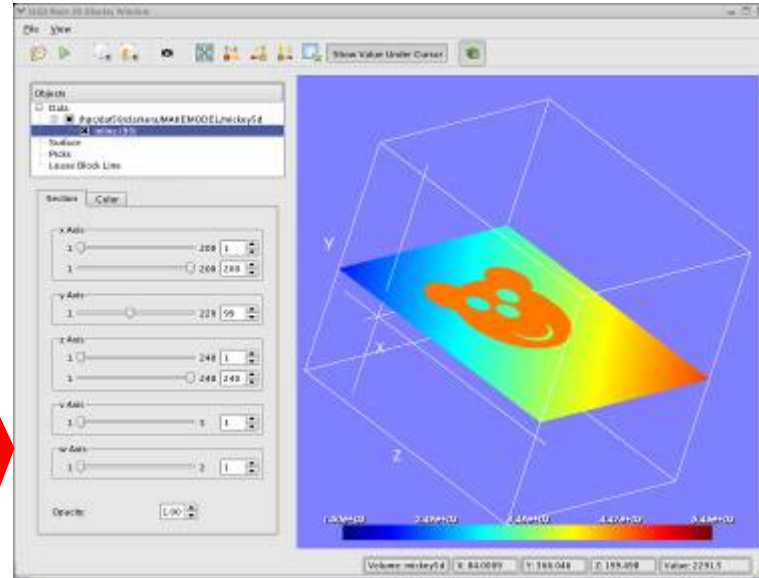
- 
- If you were napping, you might want to wake up now...

Looking ahead...

- R&D platform for seismic processing & imaging in bp
- Simplification
 - We are trying to move from USP to DDS
 - Less maintenance, training; more flexible formats (less use of integer2!)
- More user-friendly
 - GUI integrating a script builder with more powerful graphical QC tools
- Improvements to DDS
 - Parallel I/O with compressed data?
 - Thread safe?
 - Stub libraries for other packages?
 - Looking for opportunities to collaborate

“DDSGUI” – in progress

- 3D viewer for:
 - Slices of N-dimension data
 - Very large datasets
 - Surfaces
 - Well paths and Markers
 - Lease block lines
 - Scriptable
- Script builder
 - Ability to save and reload scripts
 - Very easy to add new functions (through XML)
 - Foreach loops
 - SGE job submission
 - Framework for error checking
- Based on Qt and VTK




Acknowledgments

- USP developers:
 - Paul Garossino, Paul Gutowski, Don Wagner, Martin Smith, Greg Partyka, Joe Wade, Ganyuan Xia, Bryan Helvey, Richard Crider...
 - Many people who worked at the Amoco Research Center in Tulsa.
- DDS developers
 - Randy Selzler, Jerry Ehlers
 - John Etgen, Joe Dellinger, Mike O'brien, Dan Whitmore,...
 - Many members of the bp Advanced Seismic Imaging Team
- BP management for permission to make USP & DDS open source and for permission to deliver this presentation.

For information about USP and DDS:

FreeUSP.org



The screenshot shows a web browser window titled "Data Dictionary System: Home - Microsoft Internet Explorer". The address bar shows "http://FreeUSP.org/DDS/index.html". The website header features the "DDS Data Dictionary System" logo and a navigation menu with links: Home, Users, Applications, Developers, Code Slinger, Maintainers, Download, and DDS Wiki. A search box is located in the top right of the page content.

WELCOME TO DDS

The **Data Dictionary System** is an I/O System (and much more) capable of handling multi-dimensional seismic datasets (up to 9 dimensions; eg. "axis= t offset shot x y") in various formats including [v30](#) (BP Amoco's Un*x Seismic Processing), [seg2](#) (Society of Exploration Geophysicists SEG-Y format), [seg21](#) (SEG's SEG-Y Rev 1.0 format), [seg](#) (Stanford Exploration Project's format), [sji](#) (Colorado School of Mine's Seismic Un*x format) as well as other flexible trace definitions (with or without trace headers) such as [fcube](#). DDS allows different processing systems to be used together by piping directly together as well as through actual emulation of different processing systems.

A DDS dataset normally consists of a Dictionary and Binary data. A DDS Dictionary is a text file consisting of free-form Definitions, each given by a Definition name, an equal symbol ("=") and a value which can be nearly anything without an "-" character. The command line can be used to override any Dictionary Definitions. Also, a "par" file containing command line definitions can be specified on the command line by `par=filename`.

The Dictionary defines the Binary data format and contains any accumulated processing history. The underlying data model for the Dictionary uses data structural syntax similar to the "C" programming language. The Binary data is normally in a separate file from the Dictionary and is pointed to by the Dictionary. The Binary can, however, be attached to the end of the Dictionary in order to keep the dataset contained in a single physical file. Specific formats like `usp` or `seg2` are already predefined within DDS and can therefore exist as a single binary file without the need for a Dictionary in which case they default to 3D data with `axis= t x y`.

DDS also has multi-file support where multiple conformable files can be arbitrarily merged for processing by simply creating a master Dictionary file pointing to the various individual files. It has been used in production on many different systems including CMS, Cray, Sun, SGI, Linux, Lunix clusters (32 & 64 bit); and even tested on several others. It is currently used in production on large Terabyte datasets with MPI and OpenMP parallelization.

DDS is being offered in open source by BP America Inc under [FreeCDS](#) along with a small collection of seismic imaging programs and associated utilities.

We reserve the rights to make changes to this Site "for any reason or no reason at all", including the temporary or permanent cessation of all or any portion of this Site. Such changes may be made without notice. You should visit this page from time to time to review any changes.

Need additional information? Contact [DDS Maintainer](#).

Copyright 2006 © BP America Inc, Houston, TX, USA. All Rights Reserved.

Note: FreeDDS is currently hosted on FreeUSP.org