Channel Archiver

Oct 2000

Kay-Uwe Kasemir, LANL

Channel Archiver

- Generic archiving system for EPICS
- Stores independent “Channels”

  = any Process Variable served by Channel Access
- Sampling options

  a) periodically

  b) on change

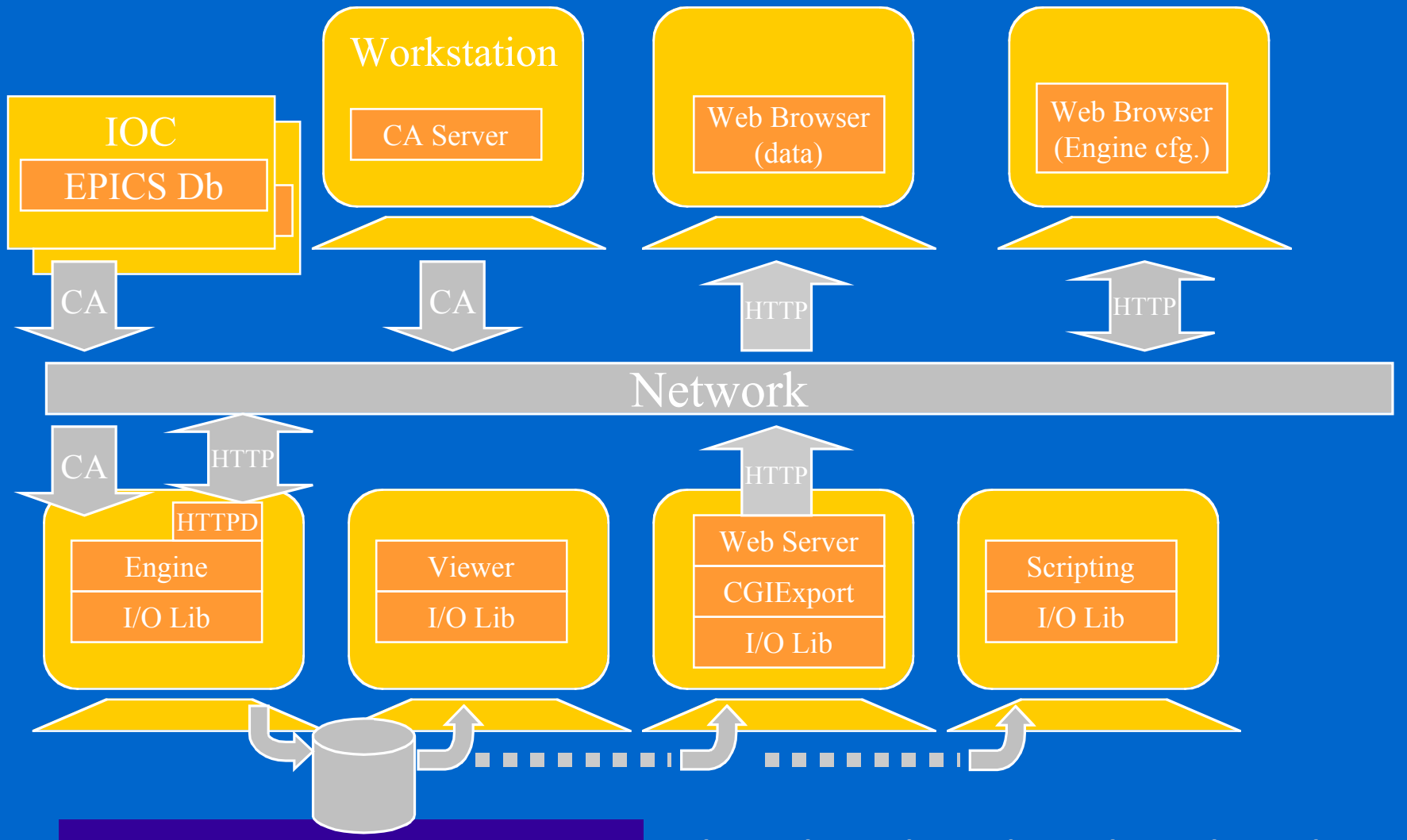
Goals

- Fast:
  - Hash-table channel lookup,
    binary data format with direct access to recent values.
- Generic and Portable:
  - Win32, Linux, Solaris, HPUX,…
  - Data archived: EPICS dbr_time_*, providing system-independent access
    to float, int, string, enum, … with time & status, units, limits, …
- Networked:
  - Remote access to both Engine status/configuration and archived data
- Extendable:
  - I/O library meant to be extended to make Engine, CGI Tool etc. work with different storage formats
- Attractive for both casual users and “experts”
  - Web interface, Win32 archive viewer for generic access to raw samples
  - scripting support (SWIG) for automated analysis, creation of input for other programs

Components

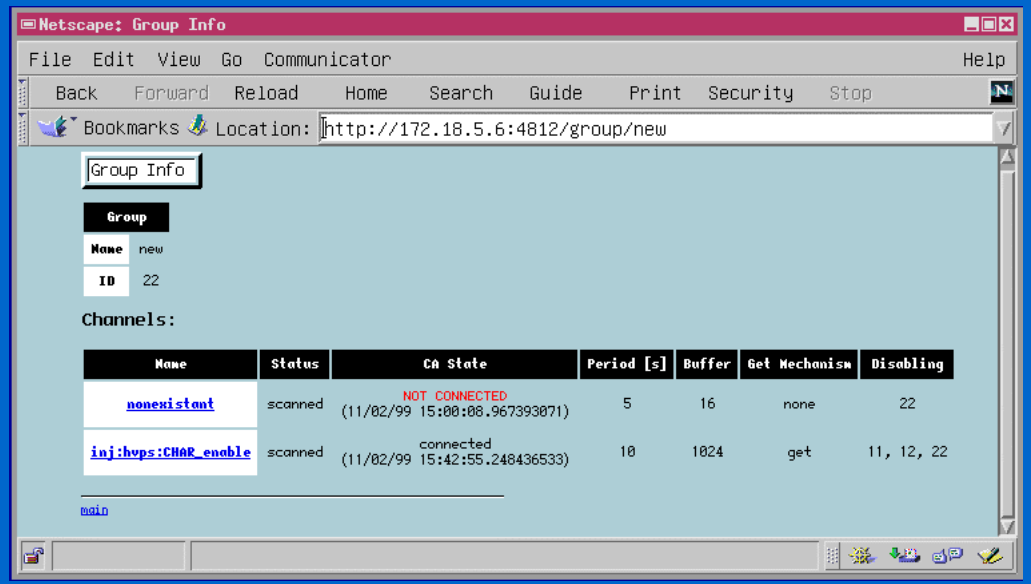
- Engine: Taking data from ChannelAccess
- CGIExport: Web access to data
- ChannelArchive ScriptingInterface: SWIG access for more sophisticated analysis
- WinBrowser: Win32 tool for fast archive browsing
- Archive I/O library: portable archive I/O, extendable for different file formats

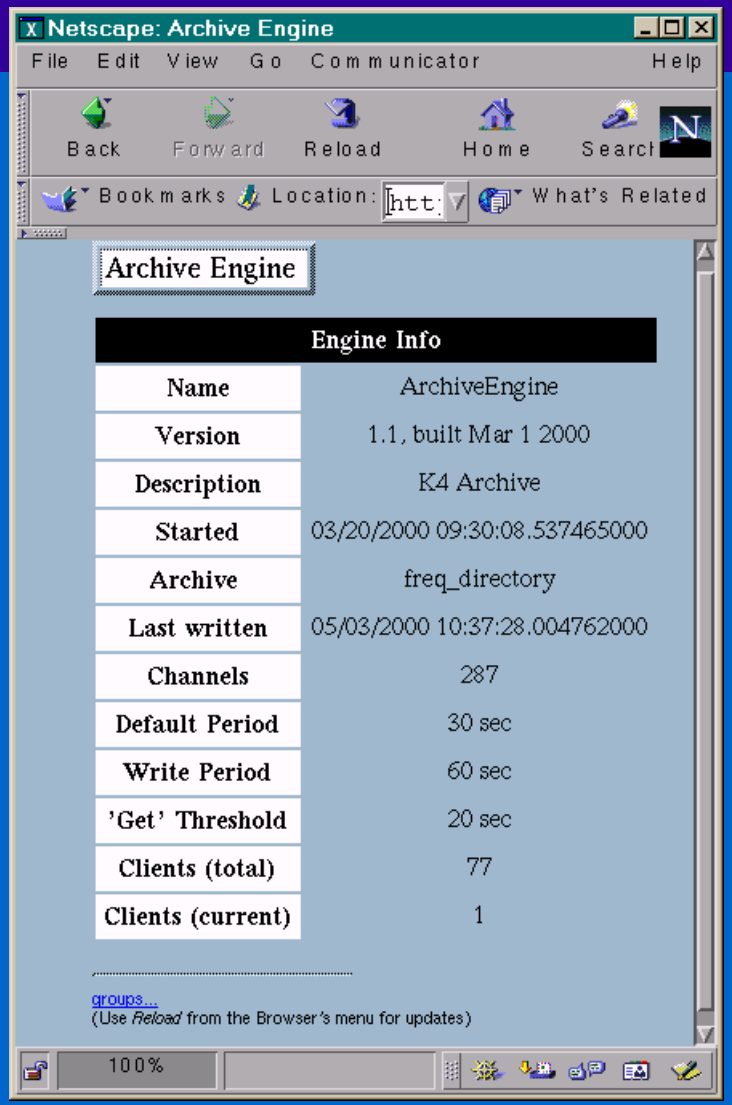
Interactions



Engine

- Networked via HTTP
- Up to 10000 values/sec (450Mhz PC)
- WIN32, Linux, Solaris





Engine: Start

USAGE: ArchiveEngine [Options] <config-file> [<directory-file>]

Options:

-port <port>	WWW server's TCP port (default 4812)
-description <text>	description for HTTP display
-log <filename>	write logfile

Default directory-file: 'freq_directory'

- Engine is ChannelAccess client: may have to set
  EPICS_CA_ADDR_LIST, EPICS_CA_SERVER_PORT
- TCP port has to be unique per machine
- Log-file: copy of messages/warnings

Engine: Configuration

- ASCII File:

  # Comments

  !default_period 30

  !group <Another config. file>

  – For building disabling groups, otherwise like an #include in C/C++

  <Channel Name> <Period [sec]> [Monitor] [Disable]

  – Scan period is also important for "Monitor" channels:
    It determines size of buffer.
    If more samples arrive than anticipated, "overwrites" occur.

- Example:

```
#Archive channels of example CA server (excas)
fred Monitor
freddy Monitor
jane 5
janet 10
```

Engine: Groups

- “disabling” channel !=0 $\Rightarrow$ group disabled:

- Main:

```
# Main archive file
!default_period 30
!write_period 60
!group power_supply
!group another_subsystem
```

- power_supply:

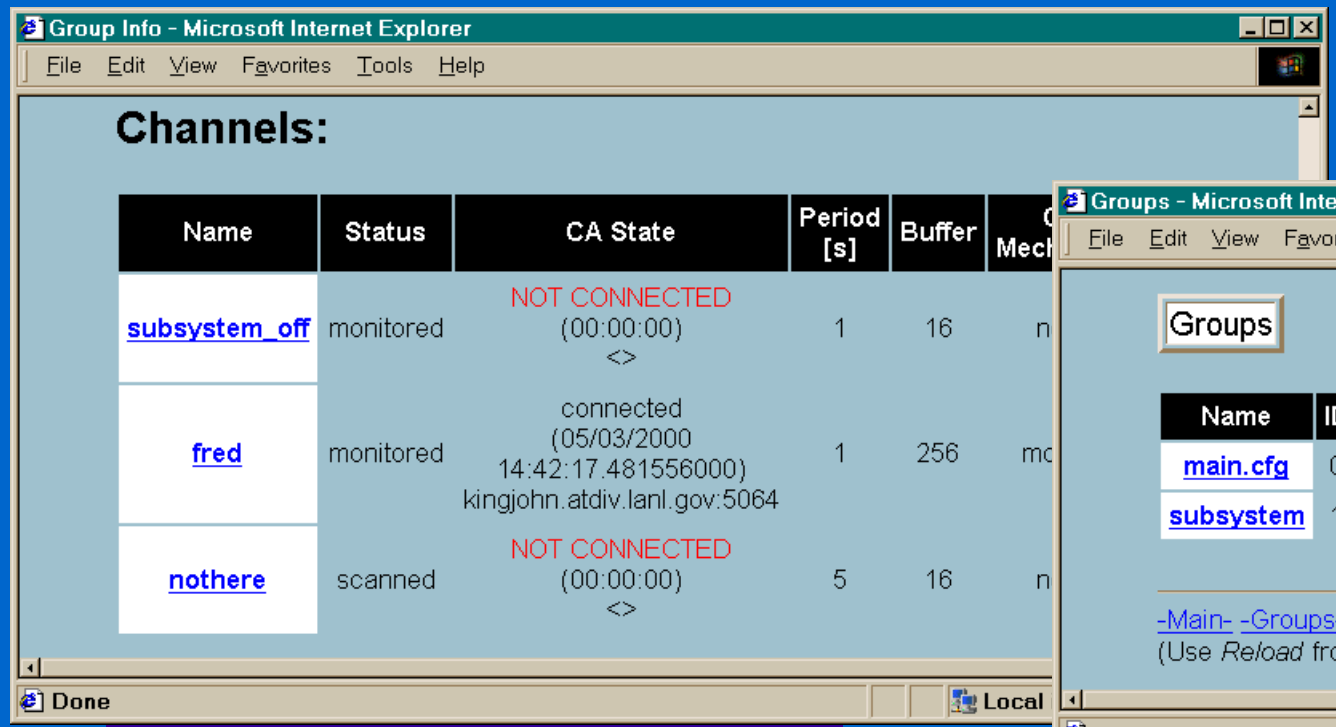
```
# Power Supply:
# Archive only when power is on!
power_off        Monitor  Disable
power_setpoint
power_readback
power_temperature
```

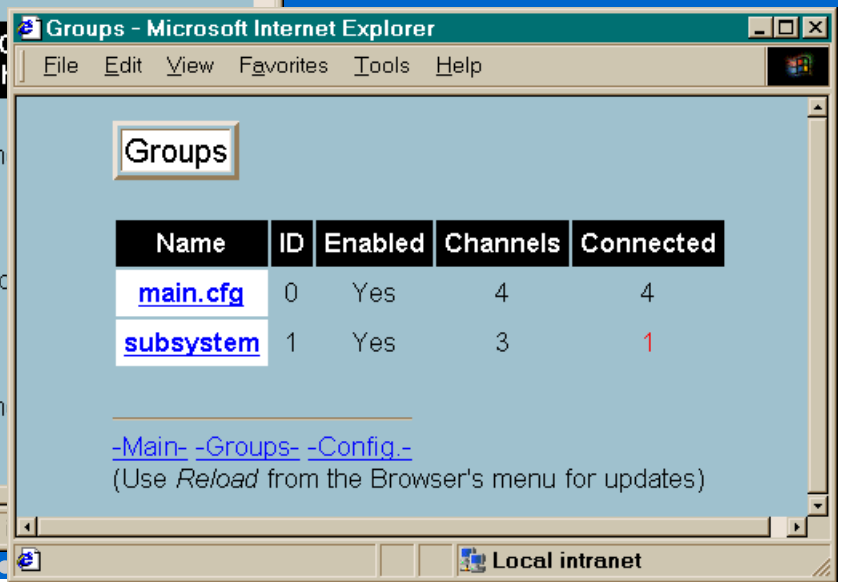
Engine: More Options

- !write_period <seconds>
  Time between writes to disk
- !get_threshold <seconds>
  Internally, CA 'get' is used for channels scanned at period > threshold, remaining channels are 'monitored'
- !buffer_reserve
  Engine keeps memory buffer per channel to buffer between writes to disk.
  Size: *buffer_reserve * write_period/scan_period*
  Since writes can be delayed by other tasks, disk activity etc., buffer is usually bigger than the minimum required (default: 3).
  If receiving "override" messages, one should
  - Check if the offending channel is tagged *Monitor*.
    In that case the *period* estimate might be too large.
  - Increase *buffer_reserve* (global for all channels)

Engine: Status

- URL of engine's HTTPD: http://<machine>:<port>
- "Client Pull": Updates on reload
- Changes (added groups/channels) written to cfg subdirectory, original config. files unchanged

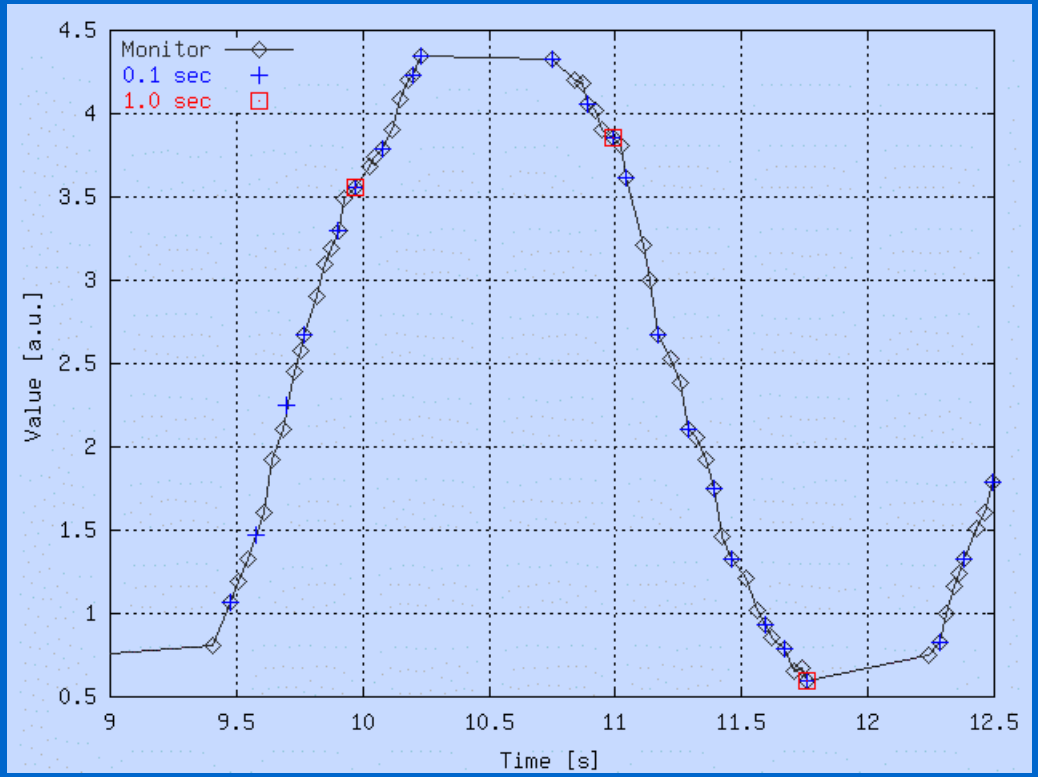




Engine: Sampling

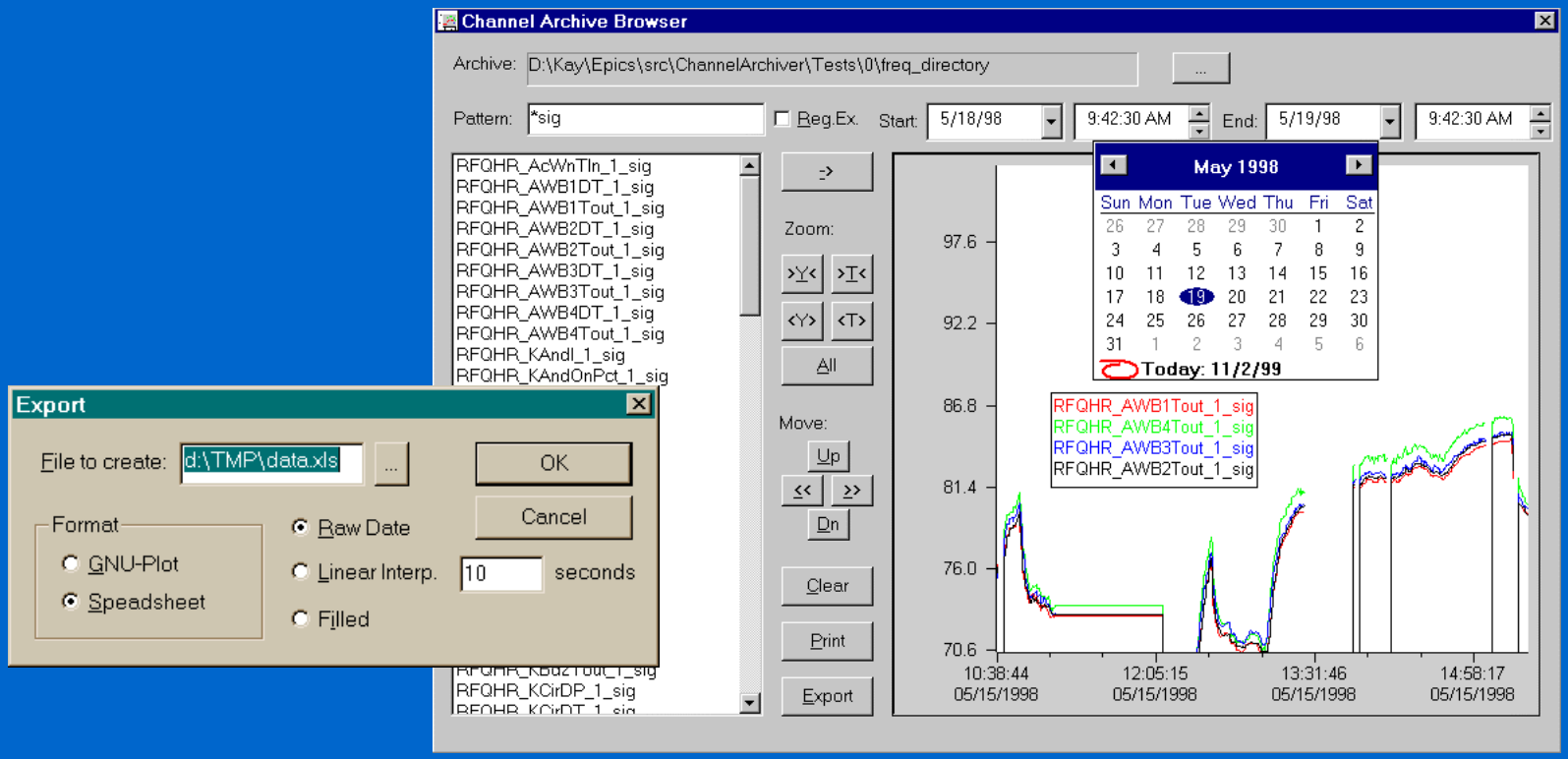
- Original time stamps of CA Server / IOC preserved, not adjusted/rounded to period!
- Example: "1.0 sec"

⇒ every sec., last value is saved



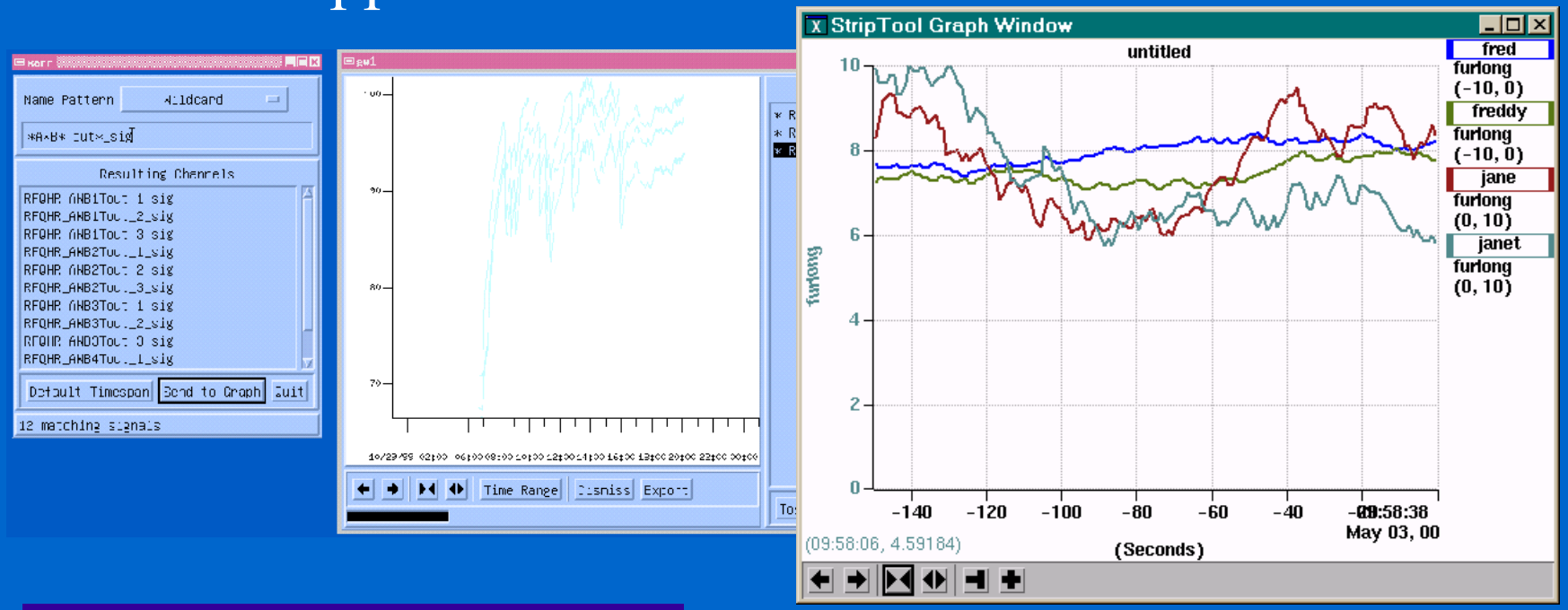
WIN32 Browser

- Familiar User Interface
- Win. Only

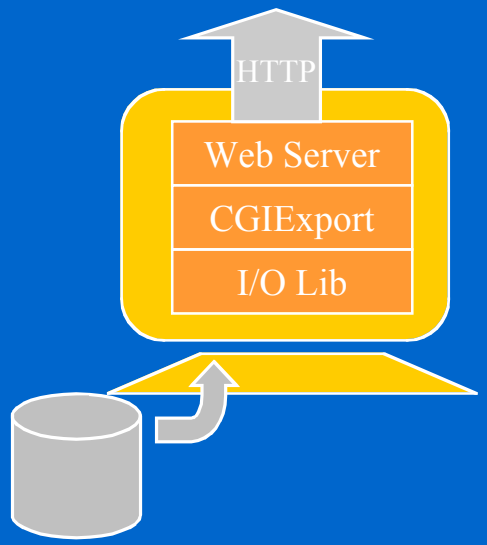


XARR, StripTool (Chris Larrieu, JLab)

- View/Export Tools for UNIX (X11/Motif)
- XARR: access to archive via older lib.
- Plan: Support LibIO for both

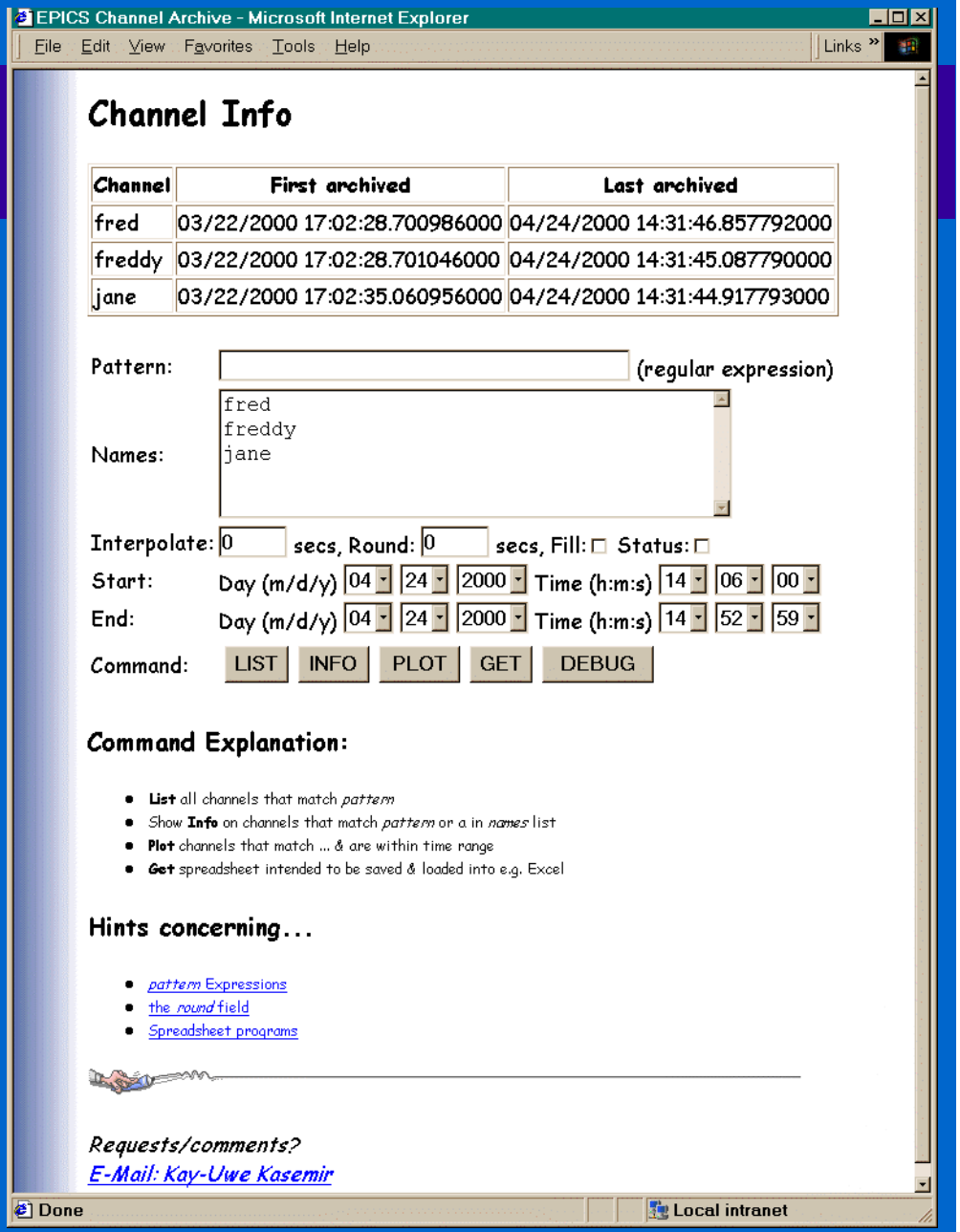


CGIExport

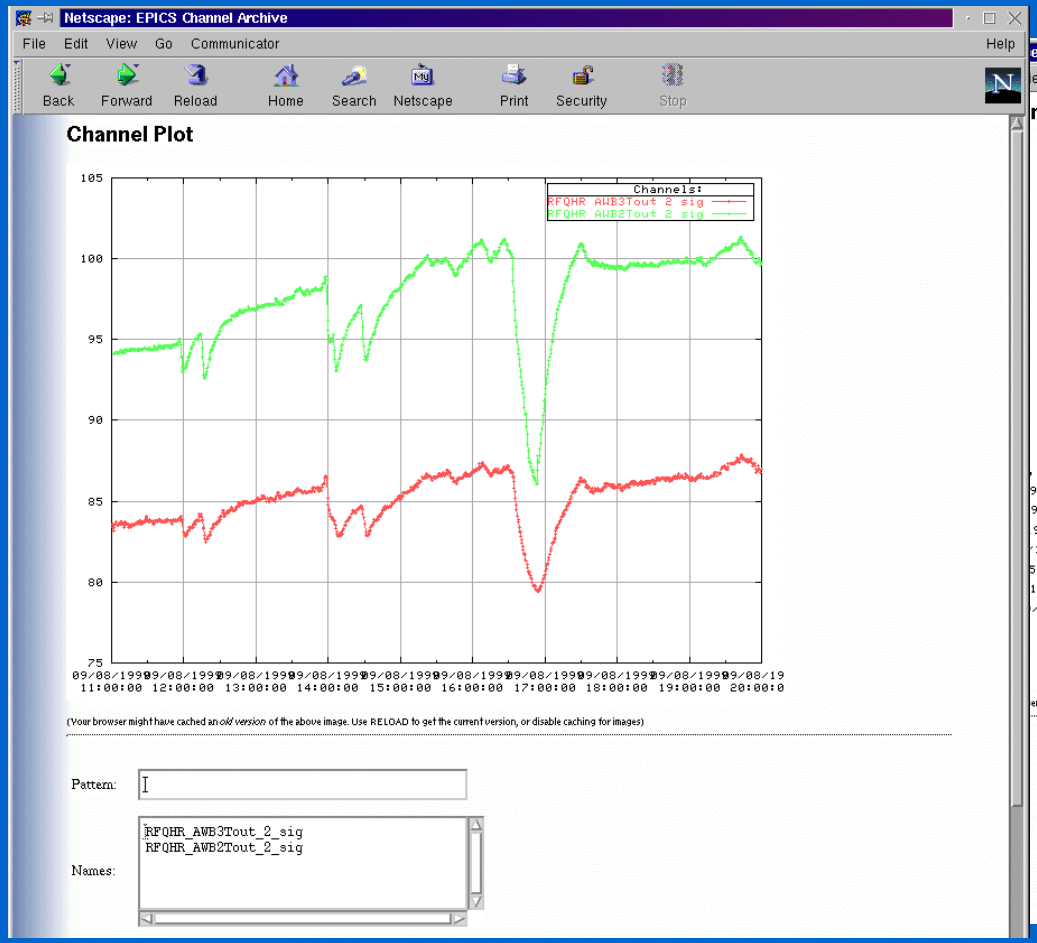


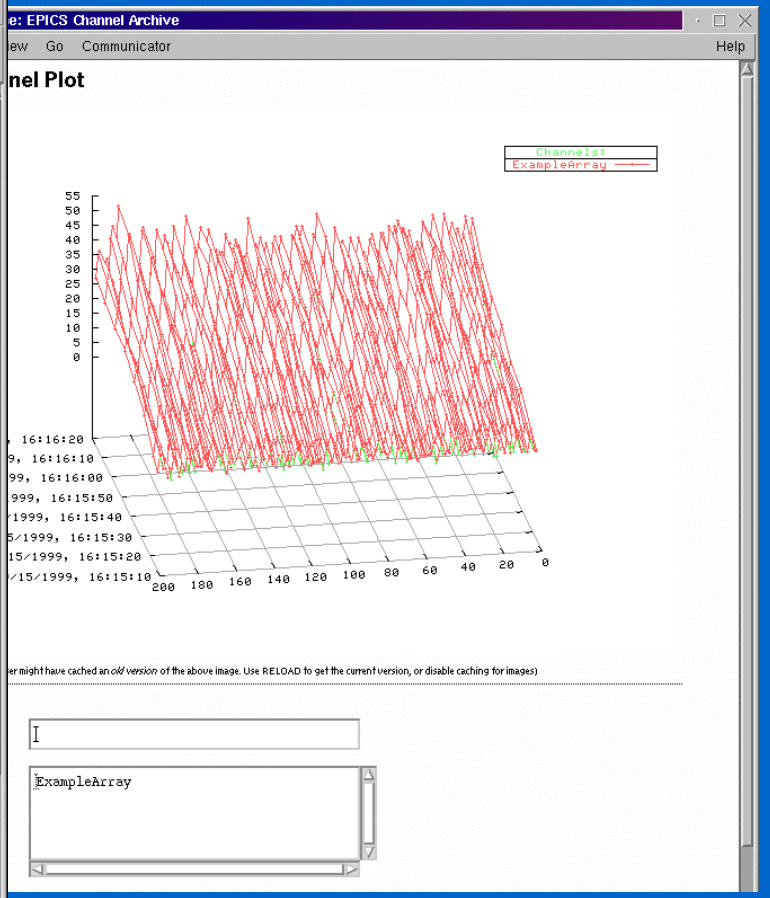
- For any Web Browser, any Web Server w/ CGI

- Archive Info
- Simple Plots (GNUPlot)
- Export in Spreadsheet format



CGIExport: Plots





CGIExport: Spreadsheet Options

- Original Time Stamps:
  ill-suited for Spreadsheets
- “Fill” missing values
  by repetition
- Linear Interpolation
  for given period

Time	A	B
3/22/00 17:02:28.700	0.071824	#N/A
3/22/00 17:02:28.701	#N/A	-0.08601
3/22/00 17:02:37.401	0.054358	#N/A
3/22/00 17:02:37.511	#N/A	-0.11178

...

Time	A	B
3/22/00 17:02:28.700	0.071824	#N/A
3/22/00 17:02:28.701	0.071824	-0.08601
3/22/00 17:02:37.401	0.054358	-0.08601
3/22/00 17:02:37.511	0.054358	-0.11178
3/22/00 17:02:39.411	0.139948	-0.11178

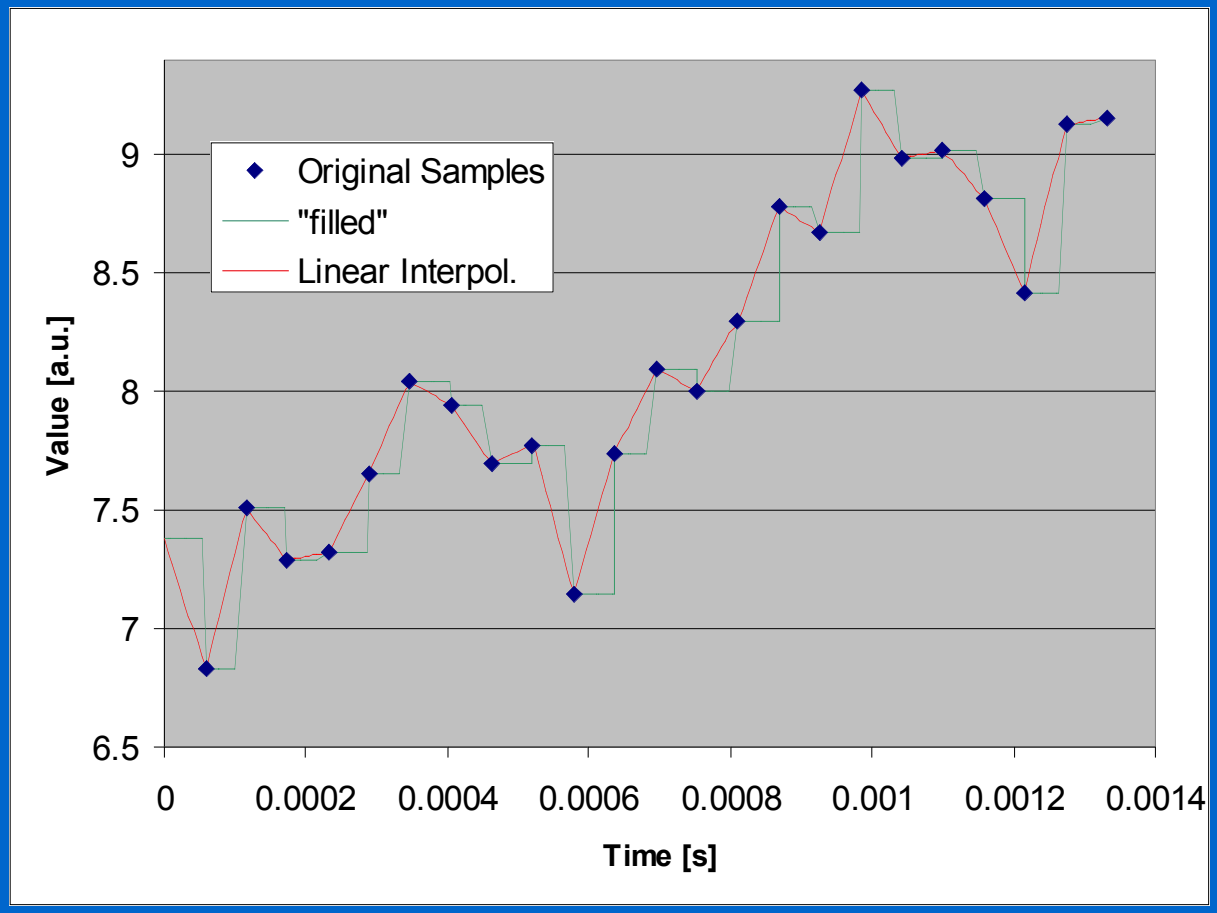
...

Time	A	B
3/22/00 17:02:28.700	0.071824	#N/A
3/22/00 17:02:28.701	#N/A	-0.08601
3/22/00 17:02:30.000	0.069216	-0.08981
3/22/00 17:02:32.000	0.065201	-0.09566
3/22/00 17:02:34.000	0.061186	-0.10151
3/22/00 17:02:36.000	0.057171	-0.10736

...

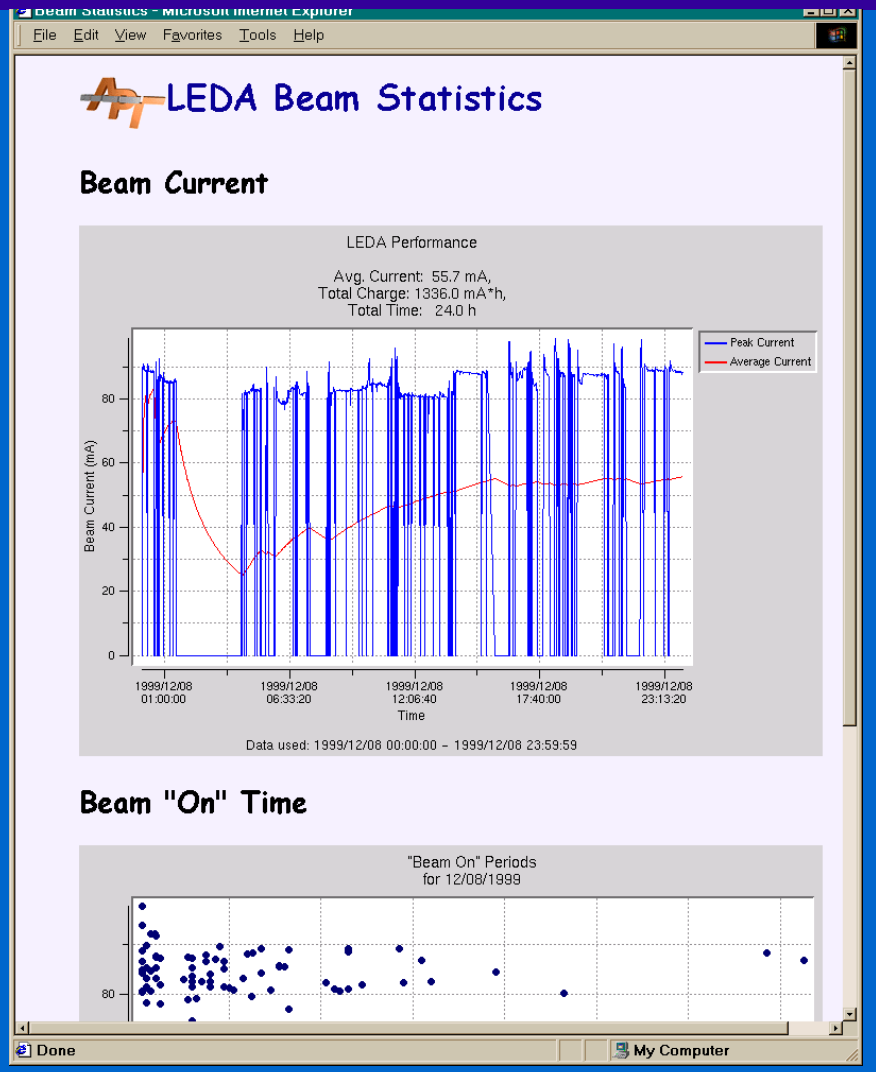
(initial #N/A until all channels have valid value)

CGIExport: “Fill”, “Interpol.”, …



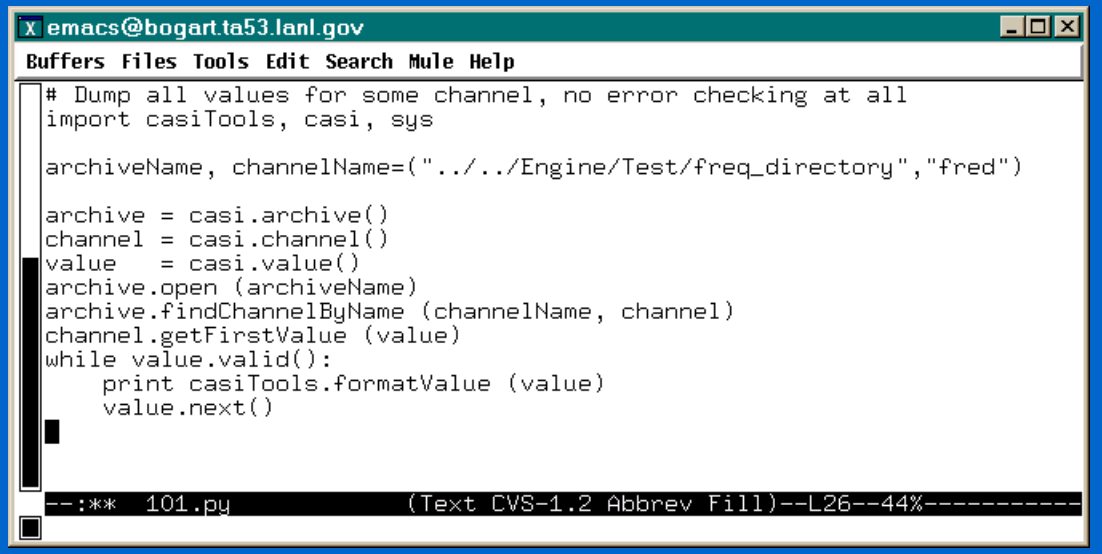
Scripting Interface

- Not optimal for “end users”, but allows programmers to
  - provide adjustable scripts: Time/Y Plot, X/Y Plot, List,...
  - automatically generate e.g. daily statistics for Web
  - write filters for Matlab, Mathematica, …
  - answer questions like: “How often was XX below 10.0 and for how long?”

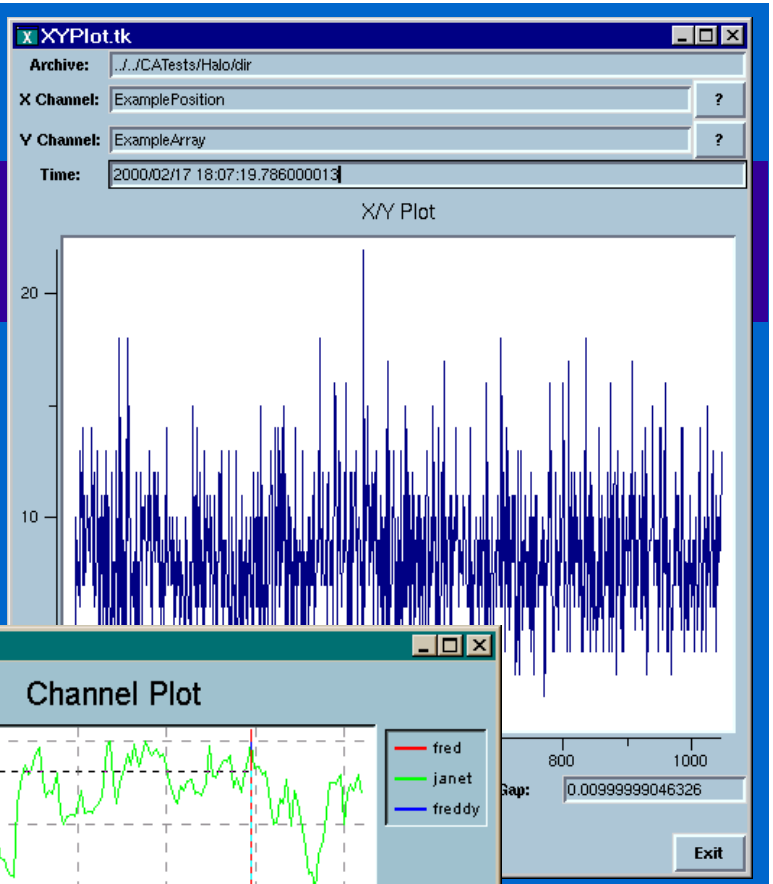


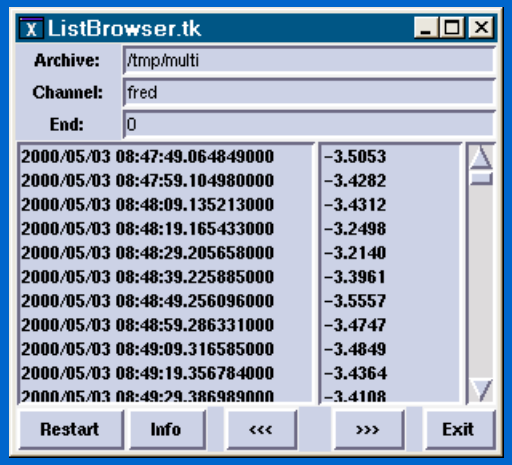
CASI: Channel Archive Scripting Interface

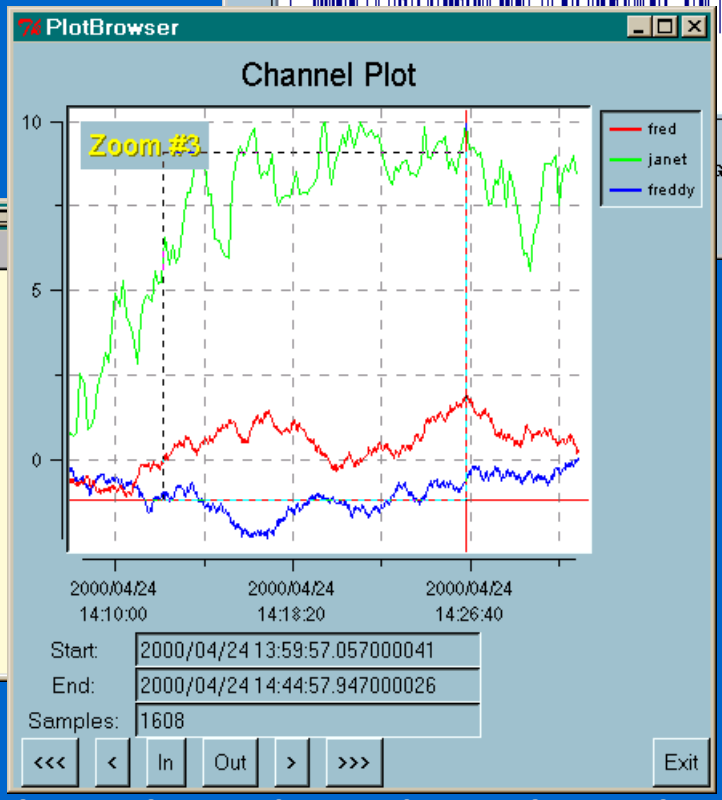
- SWIG - based:
  - allows access from tcl, perl, python
  - available for Win32 and Unix
  - Available Examples: tcl/tk and python
  - for Win32 and Linux: loadable module
- API:
  - Plain adaption of ChannelArchive LibIO

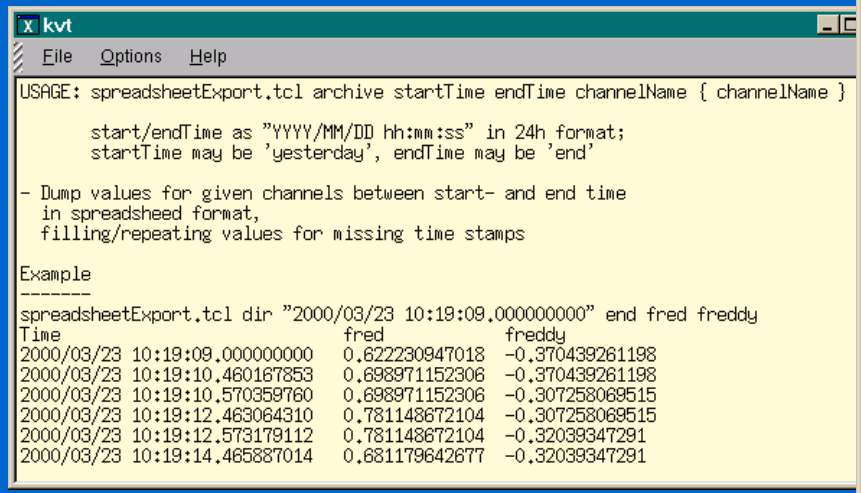


Generic Examples









Channel Archiver LibIO

- Portable C++ code (handling e.g. byte swapping)
- Based on generic “Iterator” interface:
  - Archive: list channels
  - ChannelIterator/Channel:
    first/last time available, find value before/after/near time
  - ValueIterator/Value:
    get time, status, value both as string and “raw”
- Supported: BinArchive, MultiArchive

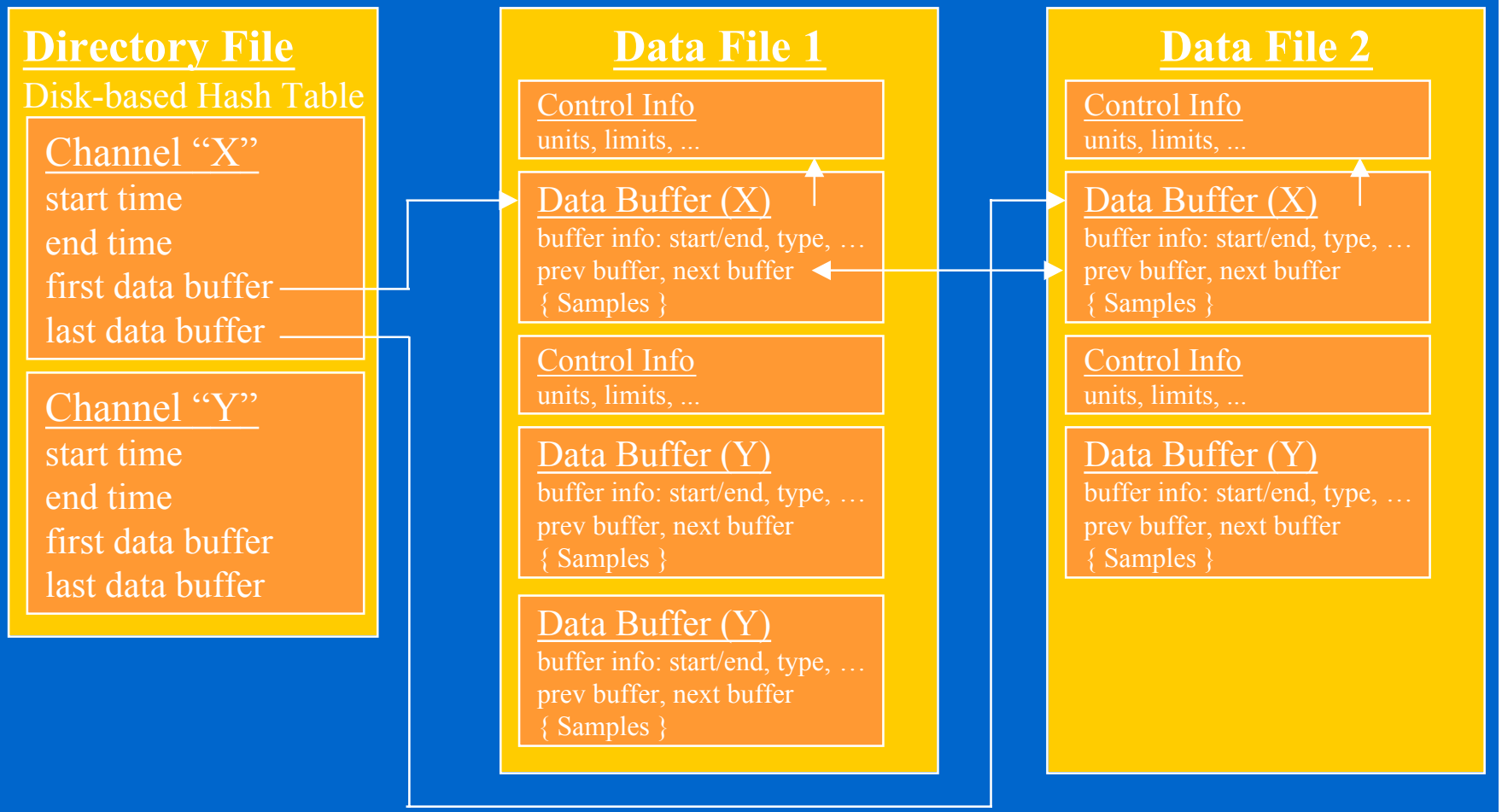
LibIO: Example

```
#define ARCHIVE_TYPE BinArchive

void list_values
        (const stdString &archive_name, const stdString &channel_name, const osiTime &start, const osiTime &end)
{
        Archive         archive (new ARCHIVE_TYPE (archive_name));
        ChannelIterator channel(archive);
        ValueIterator   value(archive);

        if (! archive.findChannelByName (channel_name, channel))
                return;
        channel->getValueAfterTime (start, value);
        while (value && value->getTime() < end)
        {
                cout << *value << endl;
                ++ value;
        }
}
```

BinArchive File Layout

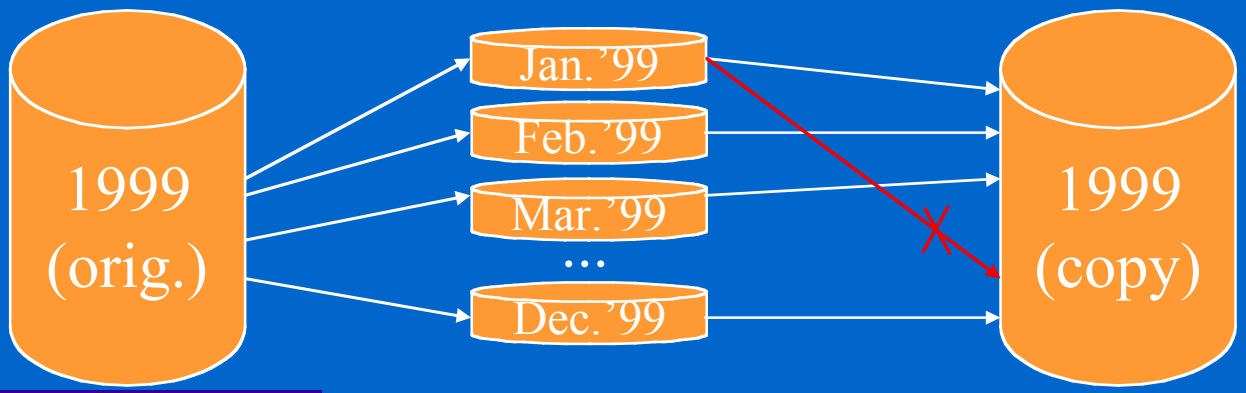


BinArchive Features

- Hash Table
  - + fastest lookup by name
  - – less suited for sorted listing or wildcard lookup
- Binary Data File, Multiple Channels per File
  - + minimized open/close calls, fastest read/write access
  - – byte swapping required, harder to maintain
- Double-Linked Data Blocks
  - + fastest access to most recent values
  - – links must not be broken

ArchiveManager Program

- Extraction of channels and time range into new archive
  - attempt is made to skip/repair “broken” values while copying
- Channels/time ranges can be appended to existing archive
  - no “insert” nor “prepend”!
- Possible Approach:
  - Create extracts of reasonable size for backup (e.g. monthly CD-ROM)
  - Original archive can be recreated from extracts, starting with the oldest one, not going back in time



MultiArchive

- Allows read-access to list of archives
  - compile-time option for CGIExport, Tcl extension, WinBrowser
  - first archive that holds requested data is used
  - when iteration meets end of data, archive list is searched again
  - no sophisticated merging, i.e. archives should not overlap in time
  - for now each individual archive has to be a BinArchive
- Format:

```
# ChannelArchiver master file
master_version=1
# Order in which archives are checked for data:
/archives/fast/dir
/archives/main/SinceJan2000/freq_directory
/archives/main/Jul99-Dec99/freq_directory
/archives/main/Jan99-Jun99/freq_directory
/home/fred/xyzarchive/dir
```

More Information

- “ChannelArchiver” under
  http://mesa53.lanl.gov/lansce8/epics/PC
- Documentation is part of sources:
  ChannelArchiver/doc
- Bob Dalesio: ldalesio@lanl.gov
  Kay-Uwe Kasemir: kasemir@lanl.gov