UT-BATTELLE

ChannelArchiver Update

2006 EPICS Meeting @ APS

Kay-Uwe Kasemir

SNS
SPALLATION NEUTRON SOURCE

SNS Statistics

- Using one "sampling" and one "serving" computer.
  - More "sampling" machines available to be configured.
  - They happen to be 64 bit.
- About 30 archive engines.
- Collected each day: >=6GB.
- Accumulated on data server: 1500GB
  - Compare: IOC Boot Server 32GB,
  - SNS Project Oracle Database: 15GB.

Recent News

- V2.8: Different "Example Setup"
  - Separation of sampling (engines) and serving (indices, data server).

- V2.9 (planned): Engine Rewrite
  - Code Cleanup and more unit-tests.
  - Index recovery tool.

Random Notes

- 64-bit Linux systems are supported since V2.6.
  - Used: RedHat Enterprise Linux WS 4
    (2.6.9 kernel on AMD Opteron).
  - Binary archive file compatibility, i.e. same 2GB size limits.
- Under Fedora Core 2 (Linux 2.6.5, gcc 3.3.3, …) there was a memory leak in the engine.
  - Engine grew with each HTTD access.
  - Didn't show under *valgrind*, I think because it's in the *pthread* library, which *valgrind* replaces with its own version.
  - Didn't investigate further. Avoid FC2/2.6.5/3.3.3/...
- EPICS Base R3.14.8.2 Timestamp Issues
  - Asserts and/or exceptions for non-normalized time stamps with nanoseconds > 1e9.
  - A bug in the Java viewer created such time stamps, the pre-V2.8 code passed it on to EPICS base, which crashed the network data server.
  - Get the more robust V2.8 and/or the recent SNS version of the java viewer.

V2.8: Example Setup

- Before: All on one computer
  - "ArchiveDaemon" used to start/stop engines and update indices.
  - Indices were updated periodically (example: every 20 min).
  - Network data server ran on the same computer.
  - Result: Constant disk access bogged computer down.
- Now: Different "Sampling"
  - "ArchiveDaemon" handles engines; updates a mailbox directory when new sub-archives are created.
- … and "Serving" computers
  - Copies the older sub-archives over, with md5sum check, updates indices
  - Runs the data server.
  - Read-Access to the "current" sub-archives on "sampling" computer via NFS.

V2.8: Example Setup Details

- **What’s to run where gets configured in XML file.**
- **At SNS, currently used with 2 computers.**
  - **In theory, should still work on 1 machine, as well as on more than one.**
- **"List" indices are used to access…**
  - **"All" archives on the serving computer.**
  - **"All - Current" archives which are still on the sampling computers, accessed via read-only NFS.**
- **Current index mechanisms don’t allow to present everything as a single big archive.**
  - ***The "list" index used for "All*" doesn’t handle the same PV in multiple sub-archives.***

V2.9: Engine Rewrite

- Cleanup for Maintainability
  - Code for handling samples was a mess.
    Now data flow a bit cleaner:
    PV → Filter → … → Filter → SampleMechanism.
  - Filters for time stamp checks, repeat counts, … re-used by several sample mechanisms.
  - Still same 3 sample mechanisms
    (monitor, scan-via-get, scan-via-monitor).
  - Cleanup allows thinking about new sample modes like "triggered" or "correlated".
    - Except there's probably no time left to actually do any of that.

V2.9: Dependability

- Engine ran well enough to fill all disks, but
  - Main engine test was running the whole engine under valgrind.
  - Deadlocks fixed upon encounter
    (last: enable/disable deadlock issues).
- Now
  - Unit tests for each utility class, Filter, SampleMechanism, …
  - Instrumented semaphore to help detect violation of lock order before an actual deadlock happens.
  - Eliminated all valgrind and purify warnings in the engine code, even if that means ‘memset(0, …)’ for unused memory regions.
  - Still no guarantee for bug-free software, but better than before (see also: Mantis 260, 258, 257?, 255, 253).

Bundled into V2.9: Index Recovery

- Submitted by Noboru Yamamoto
  - Thanks!
- Python-based tool to build an index file from data files.
  - Searches data files for ‘DATA’ strings.
  - Requires data files from V2-1-2 or later.
  - Creates index via swig calls to archiver’s C++ library.

Archive Viewer (Java Tool)

- Works great in general, but maintenance unclear since Sergei Chevtsov left ORNL.
- Local SNS changes:
  - Application jar from 15MB down to 5MB (Paul Chu @ SNS).
  - Fix for nanosecond overrun that resulted in requests with non-normalized start/end times.
  - Formula turns x/0 into "NaN" instead of exception.
- Wanted:
  - Fix for an initialization issue that causes the viewer to be slow until there is data to plot. Awfully slow when used via remote-X11-access.
    (Found by Paul, but not familiar enough w/ sources to fix).
  - Axis configuration (precision, number of ticks, …).
  - Configurable export
    - How: raw, linear interpolation, …
    - With/without status/severity
    - User-defined precision, exponential notation, …
  - Formula
    - "if" or the more nerdy "… ? … : … " operator.
    - Min(), max() don’t work with constants as in "min(1, x)".
  - Zoom
    - The ‘zoom rubberband’ currently has to stay inside the plot.
      Would be nice to be able to zoom ‘out’ slightly by ‘rubberbanding’ beyond the plot edges.

To Do

- Finish V2.9 tests and release it.
- New type of index
  - "Binary" index (each data block for each channel) is fast for retrieval, but slow to build/update and eventually gets too big.
  - "List" index is small and trivial to build, but slow on retrieval.
  - Need intermediate index:
    Channel names and start/end times of sub-archives.
- Configuration Tools
  - Currently it's expert-friendly;
    setup-once-and-hopefully-don't-change.
  - Is it worth spending (considerable) time on tools that allow "experiments", where users create/start/stop/remove archive configurations?