Astronomical Data Analysis Software and Systems X
ASP Conference Series, Vol. 238, 2001
F. R. Harnden Jr., F. A. Primini, and H. E. Payne, eds.

The CARMA Data Viewer

Marc W. Pound

University of Maryland

Rick Hobbs, Steve Scott

Owens Valley Radio Observatory

Abstract. The CARMA Data Viewer (CDV) is a CORBA and Java based real-time data viewer for radio interferometric arrays. CDV was designed to solve the problem of a common data viewer for interferometric observatories (the Berkeley-Illinois-Maryland Association array and the Owens Valley Radio Observatory array) with differing visibility data (amplitude and phase) formats and differing hardware and technical specifications. In the coming years, the BIMA and OVRO arrays are to be merged into a common array called CARMA.

We chose CORBA because it is available on many platforms with bindings for many programming languages, enabling remote object support in a heterogeneous computing environment. We chose Java for its cross-platform support and its rich set of lightweight graphical objects, in particular the JTable.

Because of its observatory-independent data model built on top of CORBA, CDV can in principle be used with any interferometer (e.g., SMA, ALMA, VLT).

1. CARMA

CARMA (the Combined Array for Research in Millimeter-wave Astronomy) is a university consortium of Caltech, U. C. Berkeley, the University of Illinois, the University of Maryland, and the University of Chicago. CARMA will combine the existing Owens Valley Millimeter Array, the BIMA Millimeter Array, and new antennas from the University of Chicago at a high altitude site in California. The combined array will consist of six 10.4m, nine 6.1m, and six 3.5m antennas. CARMA's 21 antennas will have unique multi-scale imaging capabilities.

2. Data Format

A visibility is a complex number representing the amplitude and phase correlation between a pair of antennas (a baseline). We defined a single CORBA object, called an ObsRecord, which could encompass simply the different visibility formats, independent of the peculiarities of either telescope array. An ObsRecord contains an array of visibilities records, one for each unique antenna

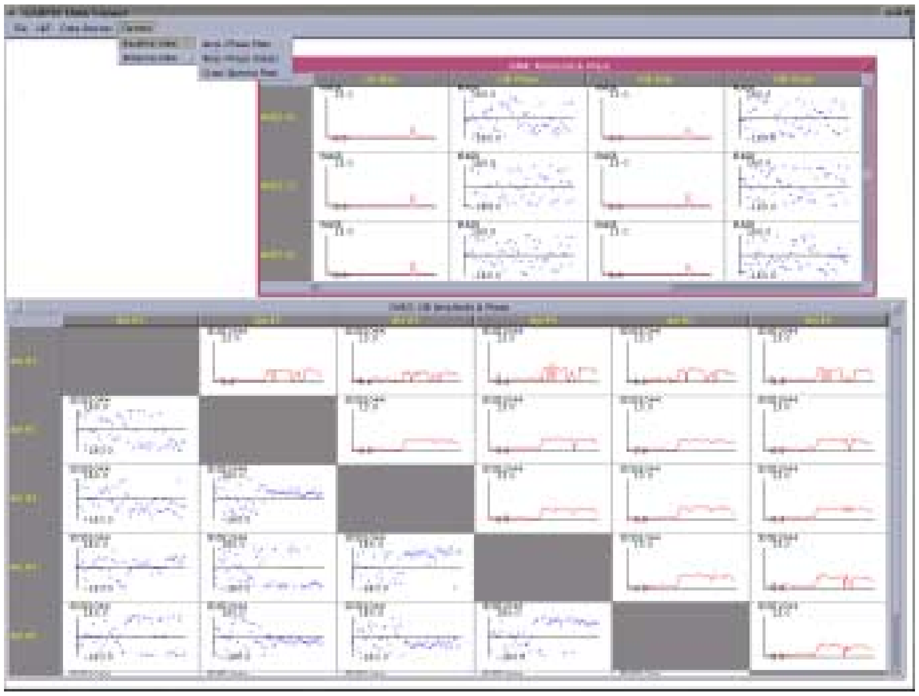

Figure 1. View of the CDV workspace showing different data representations for sideband averages (LSB & USB), which are selected using the Options Menu. Upper window shows baseline-based amplitudes (red trace) and phases (blue dots) vs. time from BIMA. Lower window contains antenna-based plots of amplitude and phase vs. time from OVRO.

pair, measured at a given time from an astronomical source. The visibility data may be multi-channel or single channel and are tagged with frequency, position, velocity, and time information. The channel data are represented in individual spectral windows, which are not necessarily contiguous in frequency. Each observatory produced software to create ObsRecords from its own visibility format and place them on a server using standard CORBA methods. The client-side data viewer can then connect to a server at either observatory (or both at once).

3. The Viewer

CDV is based on the Java 2 JTable class, with a Data Model that matches the ObsRecord specification. Each cell in the JTable contains the data for a particular antenna pair. The phases and amplitudes are displayed in the cells as a function of time (for single channel data; see Figure 1) or frequency (for multi-channel data; not shown). The cell update rate is tied to the observation integration time, typically about 10 seconds.

CDV is a workspace application, with individual data viewers running as windows inside the workspace. Multiple Data Sources (i.e., observatories) may

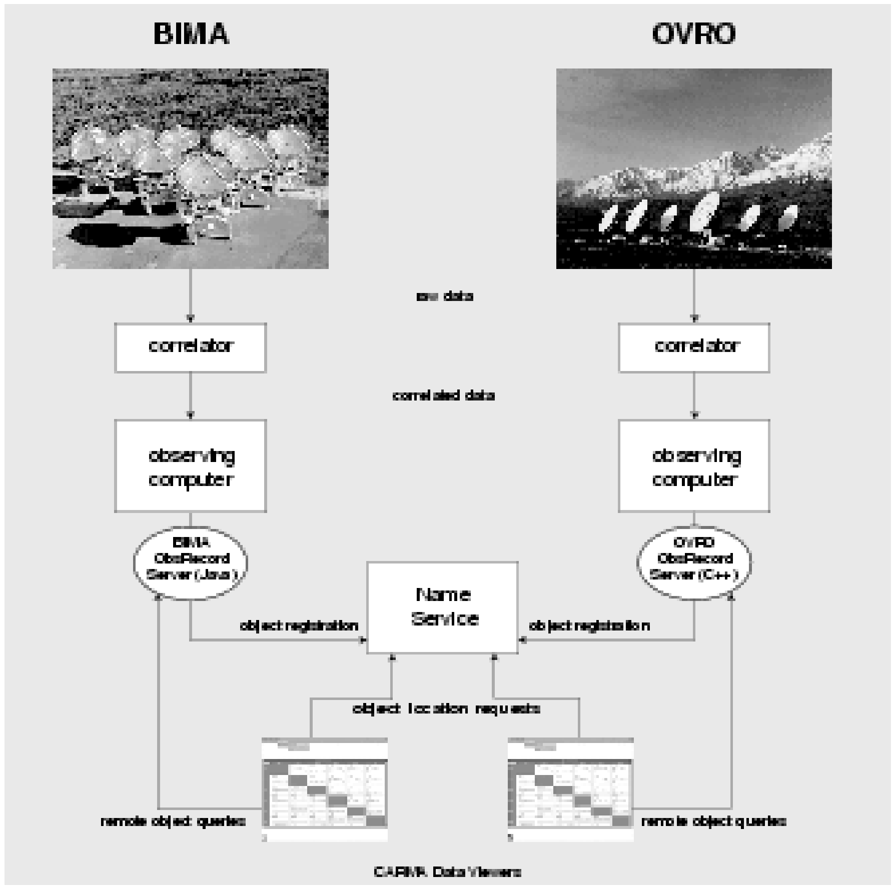

Figure 2. A diagram representing the data flow. ObsRecords are created from the correlated data on each array and registered with a CORBA Name Service. Client viewers can fetch ObsRecords from either observatory by first querying the Name Service for the ObsRecord location, then using the ObsRecord public CORBA interface to obtain its member data.

be monitored simultaneously, and a variety of representations of the data are possible.

Future enhancements will include server and client side caches for low-bandwidth operation and the ability to sort and filter the data (e.g., by time, frequency, baseline separation) using the Java 2 Comparator classes.

4. CORBA Implementation

4.1. Overview

CORBA allows convenient access to remote objects. For the CARMA Data Viewer, the remote objects (ObsRecords) contain a rich hierarchy of header and data that are updated on timescales of a few to tens of seconds. The data alone for a 45 baseline BIMA continuum-only ObsRecord is about 13.5 KB. For 4 spectral windows with 512 channels, the size increases to 200 KB. With 15 baselines, the OVRO data scales to one third the size of the BIMA data. Directly accessing the remote ObsRecord with standard accessor calls to update the Data Viewer would require hundreds of remote method invocations.

4.2. Bottleneck

The latency, overhead, and data transport time can all make significant contributions to performance and must be examined. Latency for calls over the general Internet is typically 100 to 200 msec. There is negligible latency over a 100 Mbps LAN. The overhead in a CORBA call that transfers no information is very small—approximately 1 msec. For transferring data (large CORBA objects) over the Internet, measured transfer speeds were at least 60% of the nominal T1 connecting observatory sites to the Internet. Note that for the largest objects (200 KB), the transfer time is in excess of one second. From measured performance components, the Internet latency (100 to 200 msec) can severely limit CDV if many remote method calls are required per update.

4.3. Solution

The solution adopted was to create a structure that contained all the data items for an ObsRecord. The remote method then returned this structure, effectively moving all of the data for the object in one transaction. The structure can then be used to form a local cloned copy of the remote object. The measured overhead is about 6 msec per transaction for the OVRO continuum data packet (3.5 KB) and should scale with packet size.

4.4. Overall Performance

Using this technique of moving a large structure, it is possible to update a large ObsRecord every few seconds with reasonable latency and CPU loads of a few percent or less on modern CPUS.

4.5. And finally...

Electronic versions of the actual poster in a variety of formats are available at `http://www.astro.umd.edu/~mpound/adass/2000/cdv.html`