REINAS: Real-Time Environmental Information Network and Analysis System: Phase III - SYSTEMS DESIGN*

P.E. Mantey, D.D.E. Long,J.J. Garcia-Luna,
A.T. Pang, H.G. Kolsky (UCSC),
B.R. Gritton (MBARI), W.A. Nuss (NPS)

UCSC-CRL-94-08
March 10, 1994

Baskin Center for
Computer Engineering and Information Sciences
University of California, Santa Cruz
Santa Cruz, CA 95064 U.S.A.

Keywords: Real-time, System Design, Environmental, Sensor, Data Management, Network, Visualization, Monterey Bay, Coastal Meteorology, REINAS

*Supported by a Grant from the Office of Naval Research, No. N-00014-92-J-1807

Contents

Abstract

REINAS is a continuing engineering research and development program with the goal of designing, developing and testing an operational prototype system for data acquisition, data management, and visualization. This system is to support the real-time utilization of advanced instrumentation in environmental science. Advances in continuous time measurements and improved spatial resolution allow the monitoring and understanding environmental phenomena in much greater detail than has previously been possible. The system is also designed to support the retrospective use of integrated environmental data sets.

The project is a multi-year effort of the Baskin Center for Computer Engineering and Information Sciences of the University of California, Santa Cruz, in cooperation with environmental scientists from the Naval Postgraduate School (NPS), Monterey Bay Aquarium Research Institute (MBARI), and the Center for Ocean Analysis and Prediction (NOAA/COAP).

This report documents the Third Phase of REINAS and describes the actual System Design of the project as it is being implemented. During the last half of 1993 the detailed requirements of the proposed users for the system were sharpened, selections of the technologies being used in the prototype system were made, and detailed architecture of REINAS and its subsystems for data collection, data management, processing, and visualization were worked out. The evaluations of the key components of the system are also a continuing task. Several new instruments and potential users not included in Phase II were studied.

The objective of this document is to provide project participants and reviewers with the System Design of REINAS as it enters the prototype implementation phase. It spells out the technologies to be applied, and the plans for implementing the project goals.

The REINAS system has been designed for regional real-time environmental monitoring and analysis. REINAS is a modern system, integrated into the Internet, for conducting interactive real-time coastal air/ocean science. The database design of REINAS is independent of specific database technology and is designed to support operational scientific needs throughput the entire scientific data life-cycle.

Acknowledgements

We wish to acknowledge the dedicated work of all the UCSC faculty, students and staff, and especially our partners from MBARI, NPS and other interested groups. It is difficult to name all those who participated in the REINAS project during the Phase III period, but the following contributed directly to the work or provided data for the report:

- Chapter 1: Instruments Connected to REINAS:
  Daniel M. Fernandez, Eric C. Rosen, Wendell A. Nuss, Patrick E. Mantey.
- Chapter 2: Instrument Interface:
  Michelle D. Abram, Bruce R. Montague, Bruce R. Gritton, Eric C. Rosen, David K. Schreiber.
- Chapter 3: REINAS Network Architecture:
  J.J. Garcia-Luna, Chane L. Fullmer, Stephen Petersen, Shree Murthy, Jochen Behrens.
- Chapter 4: REINAS System Architecture:
  Darrell D.E. Long, Patrick E. Mantey, Theodore R. Haining, Bruce R. Montague, Eric C. Rosen, Bruce R. Gritton.
- Chapter 5: Visualization System Design:
  Alex T. Pang, Naim Alper, Hans-Peter Dommel, Jeffrey J. Furman, Tom H. Goodman, Elijah C. Saxon, Jiahua Wang, Craig M. Wittenbrink.
- Chapter 6: Visualization Modeling:
  Harwood G. Kolsky, Wendell A. Nuss, Cheng Tang, Leslie K. Rosenfeld.
- Chapter 7: Data Compression in REINAS:
  Glen G. Langdon, Craig M. Wittenbrink, William W. Macy, Robert J. Antonucci, Alex T. Pang.
- Chapter 8: Summary and Future Directions:
  Wendell A. Nuss, Patrick E. Mantey, Daniel M. Fernandez.

Executive Summary

The Real-Time Environmental Information Network and Analysis System (REINAS) is designed to support both real-time and retrospective regional-scale environmental science, including monitoring, and forecasting.

Users of REINAS observe, monitor, and analyze regional oceanographic and meteorological phenomena; the initial focus is the local Monterey Bay *sea breeze* air/ocean phenomena [ILS+90]. Unique to REINAS is its emphasis on regional-scale interactive real-time measurement and monitoring. The system and data management architecture are both designed to provide members of the oceanographic and meteorological communities with the ability to identify and visualize phenomena as they occur in real-time and to react to emerging phenomena and trends by reconfiguring instruments at sites of interest. Applying such capability to environmental and coastal science is currently an area of considerable scientific interest.

In the REINAS architecture, continuous real-time data is collected from a variety of dispersed sensors and stored in a logically integrated but physically distributed database. An integrated problem solving environment is being developed to support visualization and modeling by users requiring insight into historical, current, and predicted oceanographic and meteorological conditions. REINAS will support both single-user and collaborative scientific work in a distributed environment.

The visualization environment under construction will provide investigators with pictures of environmental features, trends, relationships, and dynamic behavior in a geographic context. Techniques are being developed to fuse data from sensors, the historical database, and models. Automatic methods of alerting users to interesting changes in the environment are being developed.

Instruments are connected to REINAS by both remote radio and land-line links. The system is designed so that new instruments can easily be added and assimilated by the data management and visualization subsystems. Using a small personal computer, with a Unix operating system and with a standardized interface for attaching instruments, and by attaching this PC to the network using standard networking software, each instrument becomes an intelligent device on the REINAS network. An interactive electronic log book tied to the database will populate and track instrument metadata used for calibration and control.

The data management structure is designed around a data architecture integrating data from multiple instrument technologies, classes (numeric, text, and video), institutions, levels of interpretation, representations, and technology generations. An information modeling process was applied to integrate both primary data and metadata into a stable data architecture.

This report documents Phase III of the project – the System Design. Phase II [MLP+93b] covered the detailed requirements definition and preliminary architecture, Phase I [MLP+93a] the requirements analysis. The purpose of this document is to provide a detailed description of the REINAS architecture, including the instrument support, data management and database design, load paths for data from instruments to storage and users, and the support for visualization. Current status of the REINAS project, including instruments, implementation, and uses, is also included in this document.

1. Instruments Connected to REINAS

1.1 Overview

The REINAS system is being designed to accommodate data from a variety of data sources. The instruments considered in Phase III are surface meteorological stations, ocean buoys, wind profiler radars (including SODAR and RASS), and HF radars (including the CODAR system). In addition to these are data from a number of other sensors that may be included in this phase contingent upon availability of these instruments. These include Doppler lidar, NEXRAD, and rawinsondes.

REINAS must be able to accept data from many diverse sources at greatly varying rates. Sampling periods may be as short as one second (e.g., surface Met stations) or as long as several hours (e.g., vertical wind profiler or CODAR in long-term averaging mode). Instrument interfacing hardware will also vary greatly. Some instruments, such as surface MET stations, will provide relatively primitive interfaces (e.g., a generic datalogger). Others may interface to REINAS using modern microcomputers.

The typical REINAS instrument is a surface weather station or radar such as the CODAR or vertical wind-profiler. Despite differences in instrument specifics and manufacturer, these instruments can be and are usually configured to output data and accept commands through a generic serial interface. Typically, this interface is connected to an automated storage device or dial-up modem, but by connecting the instrument instead to a local REINAS microcomputer which itself is networked in some fashion to the Internet, a generic and flexible connection that enhances the utility of these remote instruments is created.

REINAS has experimented with sample periods as small as one second and generating averaged datasets for archival with temporal resolutions of one minute; however, a wide variety of sample periods will be prevalent among REINAS-accessible Met stations.

Table 1.1 provides estimates of the daily data rates and total archiving requirements expected to be collected by REINAS with all instruments operational. The discussion in Source Characteristics describes the table and presents a scenario that addresses the issues of what kinds of data are collected, the forms in which the data are maintained (raw, averaged uncompressed, averaged compressed), the amount of time it is kept in each of these forms, and based on this scenario, the total amount of data that must be maintained by REINAS.

Instrument	Expected Number	Sample period	Average period	Bytes/ average period	Bytes/ day per instrument	Bytes/ day (all)	Bytes/ day after compression	Estimated compression ratio
MET station	25	1 sec	1 min	90	128 kB	3.20 MB	128 kB	25
Profiler	8	6 min	1 hour	6144	144 kB	1.15 MB	115 kB	10
ADCP	2	15 min	15 min	1280	120 kB	0.24 MB	48 kB	5
CODAR	3		3 hours	8192	64 kB	0.19 MB	19 kB	10
AVHRR	4				8 MB	24.0 MB	1.2 MB	20
GOES	1	1 hour	1 hour	600 kB	14 MB	14.0 MB	720 kB	20
Totals					22 MB	42.7 MB	2.2 MB	20

Table 1.1: Expected Instrument Data Rates

1.2 Directly Connected Instruments

1.2.1 Weather Stations

Surface meteorological (MET) stations represent an extremely important source of real-time weather information. A generic REINAS Met station has been implemented to measure the following five properties: Wind speed, Wind direction, Air temperature, Relative humidity and Barometric pressure.

In addition to the instruments listed above, high quality solar irradiance sensors as well as precipitation gauges may also be present in future REINAS Met stations.

Currently, REINAS receives feeds from four Met stations directly and receives feeds from one through hourly file transfers. This number will increase with the collaboration of other groups (such as the Monterey Bay Pollution Control District, MBUAPCD, which owns and maintains about eight Met station sites). We expect MBUAPCD to gain Internet access and begin sharing data with REINAS during the spring of 1994. Additionally, development of Port-a-Met is currently taking place. This movable MET station will have a radio link to UCSC, thus will provide real-time access to remote, but portable instrumentation.

The following is a list of directly connected conventional Met stations using the REINAS-standard network PC interface, including stations which are on line or currently being prepared:

1. UCSC (on line: 14 June, 1993)
2. Long Marine Laboratory (on line: 30 July, 1993)
3. Lockheed (on line: 02 November, 1993)
4. MBARI-Moss Landing
5. Pt. Lobos (MBARI vessel)
6. UCSC REINAS Port-O-Met Trailer

REINAS currently has access to four directly-connected research-quality MET stations at UC Santa Cruz, Long Marine Laboratory, Lockheed Missile and Space Center in the Santa Cruz Mountains, and at MBARI Moss Landing Marine Laboratory facility. These instruments are interfaced to the Internet via standard REINAS PC systems, which are networked using a variety of physical connections (standard ethernet, analog leased-line, digital microwave).

Work is currently progressing on developing a portable battery-powered MET station and REINAS PC architecture which will be linked to the ethernet via Motorola radio modems. In addition, plans are underway to refurbish a weather station on the MBARI research vessel Pt. Lobos, and link it to the Internet via MBARI's existing microwave ship-to-shore link.

REINAS receives measurements from directly-connected stations in real-time and end-users throughout the UCSC campus routinely monitor the instruments.

1.3 Indirectly Connected Instruments

Feeds from conventional Meteorology and Ocean stations indirectly connected to REINAS are:

1. MBARI Buoys (Meteorology and Ocean data)

2. NPS Met station
3. NOAA buoys

REINAS is indirectly connected to several other Met stations and ocean buoys through anonymous FTP links with other institutions. MBARI provides data from its two buoys in the Monterey Bay on an hourly basis. The Naval Postgraduate School provides data from its Met station at Ft. Ord on an hourly basis as well, although this site is expected to become directly connected in the near future. Data from NOAA buoys within and around the Monterey Bay area are retrieved hourly via an archive at UC San Diego.

Other stations being planned or under investigation are:

1. MBUAPCD Met stations

   Plans are underway to connect the Monterey Bay Unified Air Pollution Control District's (MBUAPCD) central headquarters in Monterey County to the Internet, allowing REINAS indirect access to more than ten Met stations immediately surrounding Monterey Bay. Ideally, these stations would become directly connected REINAS instruments shortly thereafter.
2. National Weather Service (NWS), FAA and other conventional Met stations (via Unidata at NPS)
3. NWS Rawinsondes (via Unidata at NPS)
4. California Dept. of Forestry Met stations (via NPS)
5. Other services that provide local and regional weather and oceanographic data, including NWS, FAA, and California Department of Forestry Met stations in the Monterey Bay area, as well as the large Unidata service.

1.4 Pending Radar Instruments

1.4.1 Radar Wind Profiler

The radar wind profiler owned by the naval Postgraduate School and located at Ft. Ord currently uses the Doppler shift of a 404 MHz radar to measure wind velocity. This radar will be augmented in 1994 by a 915 MHz radar. The system broadcasts three beams from which a vector is calculated to determine the actual speed and direction of the wind. By varying the power level the system can measure winds at variable heights, generally from 550 m to a maximum of 3 to 5 km, with a resolution of 250 m. The 915 MHz profiler utilizes five beams and will provide wind measurements from 60 m to about 3 km at a vertical resolution of about 60 m.

Signal processing is used to estimate wind vectors. Spectra of the In-phase and Quadrature channels are generated. Rather than send the 36 spectra across the phone line, local processing is done to glean the important data from each spectrum.

There are four statistical properties that are kept: Radial velocity, Signal power, Spectral width and Noise level. These need to be saved because some quality control procedures use them to objectively determine if the resultant wind vector can be believed. For example, if the wind is clearly erroneous, then usually one or more of these three values is vastly different from a neighboring wind vector that looks believable. Signal-to-noise ratio is commonly employed to do this. There are about 30 Wind Profiler Radars in the US, including the

404 MHz and 915 MHz radars at Ft. Ord, and the 404 MHz radar at Vandenburg AF base. The rest are scattered throughout the midwest and the east coast.

Arrangements have been made for NOAA to install and provide REINAS access to wind profilers at various locations around Monterey Bay beginning in April, 1994. Some instruments, such as the one to be located at Long Marine Laboratory, will be directly connected to REINAS via existing Internet links. Other more remote instruments (such as the Hollister wind profiler) will be accessed via dial-up modems on an hourly basis.

1.4.2 RASS and SODAR

Related to the wind profiler are RASS and SODAR (SOund Detection and Ranging). Both techniques rely on acoustics. SODAR is analogous to the wind profiler because acoustic signals scatter off small-scale temperature fluctuations in the lower 500 meters and the Doppler shift is measured to determine the component of wind speed along several beams. Monoaxial SODARs transmit one beam and are used primarily to detect the vertical structure of the virtual temperature and multi-beam SODARs provide added information on the virtual temperature structure and the wind speed. One Doppler multi-beam SODAR and several monoaxial SODARs will be deployed in the Monterey Bay vicinity the summer of 1994.

RASS is an instrument that is attached to the wind profiler and it generates acoustic waves that are half the wind profiler radar wavelength. These acoustic waves generate an oscillation in the atmosphere that propagates at the speed of sound. The wind profiler radar can detect this oscillation because it is at the Bragg frequency (or one half the radar wavelength) for the wind profiler (by design). Since the speed of sound is a function of the virtual temperature (which is a quantity related to the actual temperature and humidity), the radar can measure the vertical variation of the virtual temperature. RASS is typically run for about five minutes every hour.

1.4.3 CODAR

Principles

Coastal Ocean Dynamics Applications Radar (CODAR) is a remote sensing system developed by the Wave Propagation Laboratory (WPL) now known as the Environmental Technology Laboratory (ETL), which is part of the National Oceanic and Atmospheric Administration (NOAA). The basic principle of CODAR is that high frequency (HF) radar energy in the frequency band around 10 MHz is resonantly backscattered from the ocean surface by surface waves [Cro55]. Because the resonance is due to reflection from surface waves of a known wavelength (one half the wavelength of the incident radar wave), the phase speed of the reflecting ocean wave is known precisely. Reflected energy received back at the radar site is Doppler-shifted in frequency by an amount equal to the contributions from waves traveling toward and away from the radar plus the contribution from the background current field upon which the waves are traveling. Removal of the known wave contributions provides a remotely sensed measurement of the surface current [BEW77], [BLC85]. The depth extent of the measurement depends on the depth of current influence on the reflecting surface waves. The depth range is estimated to be confined to the upper 1m of the water column.

CODAR Sites

Since March 1992, NOAA has been operating two CODAR sites alongside Monterey Bay through a contract with Codar Ocean Sensors, Ltd., which has built and refined CODAR technology since it was originally developed within NOAA. One site has been located at the Monterey Bay Aquarium Research Institute (MBARI) facility in Moss Landing and the second site has been located at the Hopkins Marine Station in Pacific Grove. The combined data from these sites has provided near-surface current estimates for the southern portion of Monterey Bay. A nearly-continuous three-month period in March to May, 1992 was analyzed and described by Neal [Nea92]. He computed mean CODAR-derived currents as well as their daily variability.

SeaSonde

A third CODAR site was established at the Long Marine Laboratory in Santa Cruz in January, 1993. The instrumentation for this site is also owned by NOAA but it is based on a state-of-the-art CODAR system, which is marketed by Codar Ocean Sensors, Ltd. under the name SeaSonde. The SeaSonde CODAR systems have a greater range than the earlier-generation systems (60 km) and they are capable of providing independent data every half hour.

The processing and data requirements for single-site and multiple-site CODAR data vary depending on the level of data reduction required. Each individual radar is controlled by a Macintosh-II computer, in the case of the SeaSonde or a PDP-11 computer, in the case of the older CODAR units. The raw spectral data returned to the radar can amount to tens of megabytes per day. In the normal operating mode, however, that data is reduced to radial current estimates by the on-site computer. Part of the data reduction involves determination of the range and direction of the reflected signal. This is done by comparing spectral returns from three different antennas co-located at the site. The timing of the returns provides range information and the relative amplitudes of the in-phase and quadrature returns provides angle information [LB83]. The radial files produced on site are less than 10 kilobytes each and are produced, at most, every half hour. This radial data is the basic product of the CODAR sites. The radial data can be combined in different ways, depending on the number of overlapping estimates, to produce vector current maps.

Additional modern CODAR sites based on the SeaSonde technology are being set up around Monterey Bay. Stanford University installed a SeaSonde site 20 km south of Monterey Bay at the Granite Canyon marine station in May 1993. The Naval Postgraduate School has purchased a SeaSonde system that was installed at Pt. Pinos (Pacific Grove) in August, 1993. The Pt. Pinos site will, eventually, replace the older site at Hopkins Marine Station and the equipment from the Hopkins Marine Station site will be used to maintain a single, older-generation CODAR site at Moss Landing.

Network Connections

Currently the Macintosh that controls the CODAR site at Long Marine Laboratory is connected via one of its serial ports to the REINAS PC at Long Marine Lab. Radial data files are delivered to the PC, and thus can be made available on Internet in near real time. Similarly, the sites at Pt. Pinos and Granite Canyon will soon be placed on the Internet, but this will require additional networking support (whether it be leased line or radio modem).

The Moss Landing and Hopkins Marine Laboratory sites cannot be easily networked because of the older computer available at the site (PDP-11), so use of existing dial-up lines, or else direct connection of CODAR's central facility (in Los Altos) is necessary to access the data from these site The data from Moss Landing is of great importance because it provides the orthogonal current component required to obtain the vector current fields all around Monterey Bay.

We are in the process of installing a leased-line Internet link to Ft. Ord. Once this link is in place, a wind profiler and Met station at this site will be incorporated into REINAS as directly connected instruments. Radio is being considered as an alternate link.

1.4.4 Phased Array High-Frequency Radar

Another type of high-frequency radar is currently under development on a joint project that involves researchers from Stanford, the University of Michigan, the Environmental Research Institute of Michigan (ERIM), and UCSC. This high-frequency radar is the next-generation model of the Stanford University four-frequency radar that has been used in a number of previous studies [Ha93], [Tea86], [Fer93]. Unlike CODAR, this radar will use standard phased array techniques to obtain information on signal bearing. This makes signal processing much simpler and more straightforward than it is with CODAR, but requires a much larger coastal area for deployment (on the order of 100 meters). This radar will also have multi-frequency capabilities that will allow it to measure the current shear in the uppermost meter of the ocean.

The current plan is to deploy this radar at the Long Marine Laboratory location sometime during the summer of 1994. This will both serve as a testbed for the new radar and also allow the first comparison ever between the direction-finding technology employed by the CODAR to estimate bearing and the beam-forming technology employed by the phased array radar to estimate bearing.

1.5 Future Instruments

Future instruments depend on negotiations, agreements, etc. as well as the availability of technology. Some under consideration are:

1.5.1 LIDAR

LIDAR (also often referred to as laser radar) has been used extensively for the measurement of various environmental parameters. One example is for the measurement of wind velocity [OIB91]. Unlike wind profilers, that scatter off of turbulence fluctuations, LIDAR scatter relies on the presence of small particles, such as those associated with sea spray or atmospheric aerosols. Doppler measurements of the resulting echo provide estimates of the various components of the wind velocity. By scanning the LIDAR over various angles, the complete wind vector may be determined by best fit of the various components over the region measured. LIDAR also has applications for measurements at the marine boundary layer, for instance, for measuring the near-surface wind field with high resolution (down to 60 meters). Actual scatter of LIDAR energy from particles near and slightly beneath the rough ocean surface can provide valuable information on the characteristics of the components of the ocean surface-wave spectrum [Chu93], [GSS87].

A LIDAR currently in use by Lockheed for air turbulence tests near aircraft may be available this summer for a period of one to three months. This LIDAR operates at a sampling rate of 200 Hz, which would allow complete wind vectors to be determined every minute (thus allowing a much faster estimate of the wind field than would be provided by a wind profiler). This LIDAR could also be configured to scan along the horizon, observing the near-surface winds and the ocean surface motion. Ideal placement would be at a secure site with high-bandwidth Internet access, such as Long Marine Laboratory.

1.5.2 NEXRAD

NEXRAD (Next Generation Radar) is a full-blown meteorological radar that is capable of a large range of environmental measurements. This system operates at a carrier frequency of 2.9 GHz and transmits up to one megawatt of power. NEXRAD relies on the presence of particulate matter (or droplets) for scatter. It is capable of measuring reflectivity up to a maximum range of 460 km and it has an angular resolution of about one degree. These characteristics give NEXRAD the capability to measure storms and precipitation at ranges of up to several hundred kilometers, winds and turbulence at ranges of up to 230 kilometers, and even swarms of bees and flocks of birds at ranges of up to 150 km. NEXRAD is currently being installed in the Santa Cruz mountains by the National Weather Service and we are actively pursuing a real-time connection to this site.

1.5.3 Rawinsondes

Rawinsondes are instrument packages that are sent aloft in balloons. The instruments used are typically the same as those in a conventional MET station, that is measurements are made of wind speed/direction, air temperature, humidity, and barometric pressure. Currently, rawinsondes are launched from several sites in central California twice a day and the data is included in Unidata feeds. This summer NPS plans to launch rawinsondes twice daily. All of this data can be fed to the REINAS system provided Internet links can be established.

1.5.4 Satellite Images

Satellite images (GOES and NOAA AVHRR) are available. Unidata is available from NPS, this includes GOES images at 8km resolution. Another source is University of Hawaii, if higher resolution GOES or NOAA AVHRR data is available.

1.5.5 Digital Video Sources

Real-time monitoring at REINAS instrument sites via remote-controlled television cameras is a proposed addition to REINAS for the years 1995-6. This network of television cameras will provide REINAS users with real-time video pictures from remote sites, and will permit storing in the REINAS database of images and video sequences captured either by user action or according to an established schedule. Video pictures from instrument sites is expected to provide additional information of use to environmental scientists, including cloud pictures, visibility, ocean wave behavior, etc. In addition, video pictures taken from the MBARI ROV may also be included as data sources to REINAS, if appropriate arrangements can be made with MBARI.

Video data from remote cameras will be captured at the instrument sites as digital data, and will be compressed at the camera site. Evaluation will be made of hardware support for video capture on an IBM-PC platform, and compression alternatives for this configuration will be investigated as part of the subsystem design. Drivers to support the video capture hardware may need to be developed for the REINAS PC's operating system (BSD Unix).

Support for remote operation of the cameras, which involves camera operation (panning, zooming and steering) via interaction of a remote user at a workstation, will require selection and testing of hardware controllers, and design and development of software and user interface support for the remote user. Since the user will not immediately see the effects of user actions on the images presented (i.e. there will be some delay due to the dynamics of the controllers and possibly from the network) the user interface will probably need to show the user where the camera is pointing and where the request will point it to avoid "oversteering" by the user.

The cameras will provide real-time video images to users on the REINAS network. Some scheduled operations are envisioned, to capture selected images at regular intervals and to store these in the database. Otherwise, the camera is an available resource over the network, and can be made available to any authorized user. During operation of the camera, others can observe the video coming from the camera. It is expected that some priority scheme will be required for the camera operations, with scheduled operations and selected users having priority for camera control, and casual observers only having camera control when higher priority users do not require control. Management of this prioritized scheduling will require development of appropriate protocols, possibly employing some token passing.

Video sources will produce large quantities of data, and will represent potentially larger data sources than most other REINAS instruments. The REINAS system design will be challenged by these data sources, and the storage and management of large quantities of digital video may also require extensions and modifications to the database management of REINAS. While these future needs have been considered in the present REINAS design, we expect that experience with video and other high data rate sources may lead to extensions and revisions of the REINAS system.

2. Instrument Interface

2.1 REINAS Interface Components

Challenge: The REINAS instrumentation interface must be designed around two challenges associated with environmental data collection: the need for instruments to operate in an independent and isolated fashion, and the ability to support rapid deployment and configuration with minimal maintenance of REINAS software and hardware in the field. In addition, a broad class of instruments must be supported, and the instrumentation interface must be robust and flexible enough to easily allow new or previously unknown instruments to be connected.

Approach: Each instrument interfaces with REINAS through a microcomputer at each node, which autonomously collects data from the instrument and immediately stores it in a local log. In the event of a network failure which isolates the node, the microcomputer continues to operate autonomously, using its considerable local storage to avoid any loss of data. Once connectivity with the merge server has been reestablished, the contents of the local log are used to bring the server up to date.

The microcomputer also provides flexibility in the design and manipulation of the instrument interface. The use of a standard computer configuration to connect *all* instruments to REINAS creates a common environment in which standardized system and device interfaces for every class of device supported by the system can be developed. All software can be developed and updated remotely. When necessary, direct communication with the attached instruments can be established through the REINAS microcomputer, allowing scientists and REINAS engineers to directly manipulate instrument parameters. These abilities combine to reduce and simplify the labor required to deploy and initially connect a new instrument to the system, and allow a more proactive approach to be taken toward maintenance, resulting in a more reliable and useful instrument.

Device Managers: Instrumentation engineers write REINAS Device Managers. A REINAS device manager is a process that provides the REINAS abstraction of the hardware device. This abstraction supports standard control and data functions. Device managers are not written from scratch, rather they follow a framework which provides a very specific proscribed template. Device managers are not REINAS end-user applications, specifically they do not use the REINAS User API. Rather, they are software components which are indirectly part of the implementation of this API.

The instrumentation-oriented components on the REINAS PC consist of the following:

- *DEVICE MANAGER* – Provides REINAS device-oriented functions for a particular hardware device. Each unique device type within REINAS has its own device manager that has been custom built for the specific device. A device manager consists of two sections: a generic section and a custom section. The generic section handles the standard interface to the REINAS system and provides a framework into which the custom routines written by an instrumentation engineer are placed. Instrumentation engineers do not need to be aware of the generic device manager's internals.

  The custom section of a device manager consists of a set of standard functions required of all device managers. These six functions define the *REINAS instrument interface*. These routines are written by the instrumentation engineer. The arguments, return values, and responsibilities of each of these functions is standard and clearly specified.

To actually manage hardware and convert data to a standard REINAS format, instrumentation engineers use a *REINAS hardware support library*. An example of the types of routines in this library would be routines that are useful in reading data from a serial port. This library will grow as additional devices are added to REINAS.

The generic portion of a device manager supports network communication transparently to the instrumentation engineer. This communication is integrated into a multi-buffering package which is also provided by the generic device manager.

All REINAS device managers are themselves managed by one program on the PC called the *collector*. The collector is the primary REINAS PC resource manager. It maintains local configuration and namining information so that a REINAS PC can perform operations in isolation, as well as coordinating activity with the larger REINAS network. Startup information for a specific device manager is managed and provided by the collector. The collector may access a REINAS database to obtain this information, but then keeps a cache of all such information on the local PC.

- *COLLECTOR SECURITY FILTER* – This is a module within the collector which determines if a request to control or access a particular PC instrument or sensor is valid, i.e., it supports access control list (ACL) security operations with respect to any instrument on the PC. The security filter is not visible to instrumentation engineers.

- *REINAS INSTRUMENTATION INTERFACE* – These routines are written by an instrumentation engineer to a specification outlined below. The required instrumentation interface routines are: *Inst_open(), Inst_close(), Inst_get_data(), Inst_set_attrib(), Inst_get_attrib()*.

- *REINAS HARDWARE SUPPORT LIBRARY* – These routines can be used by instrumentation interface engineers from within the functions they write. As previously mentioned, these are functions that perform operations such as open a UNIX serial port, set its characteristics to get 8 bit ASCII, etc. This library also includes the routines required to convert data from an arbitrary architecture to a standard REINAS format.

2.2 Instrument data in PC "Standard" Format

Data received from instruments must be converted to a form recognized by the REINAS system even though data from individual instruments varies widely in format, type, and sampling rate.

In addition, as REINAS operates on both big-endian and little-endian architectures, REINAS must convert data into a standard binary format. This has been accomplishing by providing IEEE floating point routines compatible with the standard Internet *htons()* and *ntohs()* functions. Instrument programmers then define externally visible data structures using a set of REINAS data types that are architecture independent. These types specify the basic type and the number of bits in the representation. Examples are *INT_16*, representing 16 bit integers, *LONG_32* specifying 32 bit integers, and *FLOAT_IEEE* representing a single-precision IEEE floating point value.

There are only two data structures that need be defined by instrumentation engineers using the REINAS data types:

1. *Data Record.* A data structure specifying the format of data records produced by the device. This definition can include variants, i.e., more than one data record format can be generated from a given device, but there must be an indication within the record as to what format has been generated.
2. *Attribute Record.* A data structure specifying instrument configuration and control parameters.

The device engineer at present must explicitly write conversion routines for the above data structures, that is, must write a routine to convert each of the above data structures to both network and host format.

The device engineer also, in conjunction with a data loading or database engineer, defines a database container corresponding to the data record provided by the device, and defines a database system type corresponding to the configuration and control parameters of the instrument. This last definition must incorporate the database 'metadata' associated with the device. These engineers also write a database load function that is device specific and writes the information contained in one device data record structure into the database.

Specific function responsibilities and data format requirements are briefly described in the following section.

2.3 Defined Suite of Standard Functions

2.3.1 REINAS Instrumentation Interface Routines

REINAS Instrumentation Interface routines are routines that need to be written by the instrumentation engineer. Each instrument (class) must have routines provided by the instrumentation engineer that are specific to the individual type of instrument. To make integration into REINAS easier, this set of functions is standardized into a set of standard functions that the engineer must provide. The required functions are: *Inst_open(), Inst_close(), Inst_get_data(), Inst_get_attrib(),* and *Inst_set_attrib().*

These functions work in conjunction with the two data structures described in the previous section: the *data record* and the *attribute record* structures.

- The *Inst_open()* routine accesses a hardware device for further REINAS operations. A handle is returned that is used in the other *Inst* calls. It is the responsibility of this routine to verify that the hardware is present and to access the hardware via UNIX system services (i.e., open a serial port). There are no requirements that this routine perform any network operations. This routine may activate the device, but data that is returned from the device should be discarded, unless an *Inst_get_data()* call is performed.

  Inst_open() is responsible for providing any one-time initialization required to operate the device. A significant part of this initialization is calling the support library function *reinas_init()*, which specifies buffer size, number of buffers, and some timer information.

- The *Inst_close()* routine uses the handle returned by *Inst_open()* and closes the device and performs any necessary cleanup tasks.

- The *Inst_get_data()* routine gathers a data record as defined by the devicngineer and returns. This routine is called by the device manager as specified in the *Inst_open()* call via *reinas_init()*. The *Inst_get_data()* function only needs to obtain and return a single data record from the instrument. The generic section of the device manager then handles packing the data into a buffer structure, logging it into the local log, and sending it to the database.
- The *Inst_get_attrib()* routine obtains hardwaspecific device attribute information. Such information describes the current operational mode or state of an instrument. Some instruments will have functions supporting such enquires (e.g., 'get status') and others will not. In this latter case, the *Inst* routines must track attributes in software (i.e., items such as sampling rate).
- The *Inst_set_info()* routine changes instrument hardware attributes according to the specified instrument attribute structure provided as an argument. Such attributes include items such as sampling rate or sensor direction.

2.3.2 REINAS Hardware Support Library

The REINAS Hardware support library consists of routines that can be called by the instrumentation engineer while implementing REINAS Instrumentation Interface routines. These routines are UNIX and hardware specific and can be expected to grow throughout the lifetime of REINAS. These routines are included in a standard library. With the exception of the previously mentioned *reinas_init()*, these routines are device specific.

2.3.3 REINAS PC to Local Log

After data is collected on a REINAS PC, but before it sent to a merge server, it is temporarily stored in a central log file on the PC's local disk, access to which is arbitrated by the collector. This ensures that data that is waiting to be sent to the merge server will not be lost if the PC crashes. Early versions of REINAS, along with current versions for platforms other than BSDI/386, allow device managers and the reader to read and write buffers of data directly to and from the log file. While this does guarantee recoverability for data not currently in the buffer, it also can lead to excessive disk accesses.

On PCs running BSDI/386, a new method of logging data is currently in use. This method, based upon the Recoverable Virtual Memory software package developed at Carnegie–Mellon University [SMK+93]. Recoverable Virtual Memory (RVM) is a transaction processing system that supports an array of bytes as a single data type. Versions of REINAS that use RVM do not access the disk directly. Instead, a segment of shared memory is created and maintained by the collector and backed by RVM. If a device manager or reader wants to access the log, it gets control from the collector and then accesses the shared memory segment. When it finishes, it releases its lock on the segment, which, in the case of a write, causes the collector to tell RVM to commit the "transaction." The main advantage of this system is that no read from the log will ever cause a disk access [MS94].

3. REINAS Network Architecture

3.1 Networking Support

Communication among REINAS components and users is being accomplished via a mixed-media networking infrastructure that encompasses new and existing telephone lines, Internet connections, radio links, as well as the networking software and hardware needed to control and manage the interconnection of REINAS sites.

Today's REINAS network consists of existing Internet connectivity with a few additional telephone and point-to-point radio links. However, in the future, REINAS needs to include several mobile sites (e.g., boats and trailers with MET stations) and applications that require the transport of large amounts of information, specially for remote visualizations, where one minute animation can require in the order of 160 Mbps. The information exchanged in REINAS will include multiple media (text, voice, images, graphics and animation, and even video), and such information has to be distributed in real time (e.g., during a multimedia conference among multiple sites) over different types of transmission media, including radio links and high-speed lines. Furthermore, the networking infrastructure of REINAS should allow a very large number of sensors to be incorporated into the system. Accordingly, we see six major networking requirements in REINAS: the ability to transport multimedia data in real time, scalability to a large number of geographically-dispersed sensors, mobility of sites, fault tolerance, efficient use of multiple transmission media, and connectivity to the Internet.

The marked differences between REINAS networking needs and traditional networking technology indicated the need for new multimedia networking solutions to REINAS's unique characteristics and to allow REINAS system engineers to better manage and monitor communication resources in support of data management, data visualization, and user communication.

Translating the design of a new multimedia network design into a working prototype requires the development of new algorithms and protocols to control the network, to select routes, to adaptively manage links, to control traffic flows and congestion, and to manage communication resources to meet user and system requirements for data gathering, management, and visualization. The rest of this section provides the status of our architecture and communication protocol design, and outlines future directions.

An early design choice for the REINAS multimedia network is the adoption of the TCP/IP protocol suite. This choice is based on the functionality provided by the protocol suite, the choices of networking equipment available, and the need to develop new network protocols. The main networking research during Phase III has focused on an architecture that integrates multiple transmission media, routing protocols that scale to large very large numbers of nodes and are applicable to wireless environments, and channel-access protocols that support multimedia traffic.

3.2 Network Architecture Integrating Multiple Transmission Media

In the design of the REINAS network architecture, we relax the distinction between links, networks, and internetworks used to provide connectivity, so that the various transmission systems are treated in a uniform manner, leading to a fault-tolerant multimedia system.

The provision for type-of-service routing and load-sharing via multiple types of transmission media is a key component of our design. A communication security architecture could be integrated into such a communication system. This design is based on earlier work by Mathis and Garcia-Luna on survivable multimedia networks [MGLA87].

A major issue in this design is how to organize a heterogeneous mix of communications resources into a fault-tolerant system. The effort is compounded because some of the links are dedicated, others are multiple access channels, and some are actually networks. Assets such as links, networks, and other internetwork systems are treated simply as communication mechanisms to transport data between REINAS routing nodes, which we call *multiband routers*.

To provide for a uniform architectural treatment of the various types of media and networks, we adopt a generic classification of transmission media divided into four groups [MGLA87]:

- Dedicated point-to-point circuits such as radio links.
- Switched point-to-point circuits such as telephone circuits.
- Addressed multidrop or multinode systems such as BARRNET
- Broadcast systems such as multiaccess satellite channels

This classification is based on both the sharing aspects of the media and the switching or addressing aspects. A *link* between multiband routers could be a point-to-point link (e.g., leased lines), a switched link (e.g., dial-up lines), a broadcast system (e.g., various type of multi-point radio), or an addressed system (e.g., a network). This general treatment of the transmission resources provides a basis for real-time communication using any available means.

Many multiband routers will be redundantly connected via multiple parallel links. Accordingly, the concept of a *path* between multiband routers is introduced to symbolize that fact. Thus some control traffic (such as route updates) needs to traverse only a *path* between multiband routers and not every *link*; of course, link status probes would still be carried over each link.

3.2.1 Routing Protocols for REINAS

A critical element in the provision of fault tolerance and the ability of a network to scale is the choice of the routing protocol used.

For the purposes of routing protocols, an internetwork can be viewed as consisting of a collection of interconnected domains, where each domain is a collection of such resources as networks, routers, and hosts, under the control of a single administration. Current work in interdomain routing has proceeded in two main directions: protocols based on distance-vector algorithms (DVA), which we call distance-vector protocols, characterized by BGP [LR91] and IDRP [ISO91], and protocols based on link-state algorithms (LSA), which we call link-state protocols, characterized by the inter-domain policy routing (IDPR) architecture [Ste92]. The same two basic approaches have been used in the Internet for intradomain routing (e.g., RIP [Hed88] and Cisco's IGRP [Bos92] are based on distance vectors, and ISO IS-IS [ISO89] and OSPF [Col89] are based on link states). We view REINAS as a single domain, and focus on intra-domain routing protocols.

The key advantage of DVAs, and the distance-vector protocols that use them, is that they scale well for a given combination of services. Because route computation is done distributedly, DVAs are ideal to support the aggregation of destinations to reduce communication, processing, or storage overhead [GLA88]. However, although Garcia-Luna and others have proposed DVAs that eliminate the looping problems of old distance-vector protocols like EGP and RIP [GLA93], an inherent limitation of using distance vectors is that routers exchange information regarding path characteristics, not link or node characteristics. Because of this, the storage and communication requirements of *any* DVA grows proportionally to the number of combinations of service types or policies [Jaf84]; therefore, supporting many types of services and policies using any DVA is inherently complex.

Because of the failure in the past to overcome the looping problems of early distance-vector protocols (RIP and EGP in particular), using link states has been considered to be the main practical alternative to Internet routing. However, a key disadvantage of today's link-state protocols is that they require routers to broadcast complete topology information by flooding. As pointed out by Estrin and others [ERH92], this approach does not scale well. The main scaling problems of today's link-state protocols are three: flooding consumes excessive communication resources, requiring each router to compute routes using the same topology database at every router consumes excessive processing resources (e.g., see the results shown in [ZGLA92]), and communicating complete topology information is unnecessary if a subset of links in the network is not used in the routes favored by routers. As a concrete example of the scaling problems of link-state protocols, Garcia-Luna and Zaumen [GLAZ92] have shown that DUAL [GLA93] performs more efficiently than OSPF even in a relatively small network of the size of ESNET, even when OSPF areas and OSPF masks are used in both DUAL and OSPF.

The key concerns regarding the choice of a routing protocol for REINAS are scalability, efficient use of multiple transmission media, and support of real-time multimedia applications. However, as the previous paragraphs describe, the algorithms used for distributed route computation in today's internet routing protocols have severe scaling problems. On the one hand, DVAs need to communicate routing information among routers on a per path basis, which leads to a combinatorial explosion of service types and policies. On the other hand, LSAs require the same topology information to be replicated at all routers, which consumes excessive communication and processing resources in very large internets. An additional problem with today's internet routing protocols is that they are all based on shortest-path algorithms running over a simple graph. In the future REINAS multimedia network, two routers may be connected to each other through more than one link.

Surprisingly, although the inherent limitations of LSAs and DVAs are well known, all of the existing internet routing protocols are based on these two types of algorithms, and the current proposals for interdomain routing in large internets [Chi91], [ERH92], [Ste92] either are based on LSAs and DVAs or leave distributed route computation as open problem.

Because of the multiband nature of the REINAS multimedia network and its expected complex connectivity, choosing to send a packet over a given path on the basis of a simple cost metric (e.g., number of hops that the packet will traverse, or the delay in the path) is not always sufficient. Furthermore, because the routing protocols implemented to date maintain a single shortest path between each source-destination pair of nodes, the throughput and delay over the chosen paths may be suboptimal. In the REINAS multimedia network, multiple paths for each source-destination pair should exist so that routing of packets can be more stable with respect to the traffic load changes.

New approaches are clearly needed to solve the inherent limitations of today's internet routing technology, which dates to the design of the ARPANET routing protocols. Accordingly, during Phase III, we have developed new routing algorithms that address many of the performance issues cited above. The rest of this section summarizes their basic operation and our general direction insofar as applying our results on routing to mobile radio networks.

3.2.2 Loop-Free Path-Finding Algorithm (LPA)

Recently, distributed shortest-path algorithms that utilize information regarding the length and second-to-last hop (or predecessor) of the shortest path to each destination have been proposed to eliminate the counting-to-infinity problem of the distributed Bellman-Ford algorithm (DBF), which is used in a number of today's routing protocols. We call these type of algorithms path-finding algorithms. Although these algorithms provide a marked improvement over DBF, they do not eliminate the possibility of temporary loops. The loop-free algorithms reported to date rely on mechanisms that require routes either to synchronize along multiple hops, or exchange path information that can include all the nodes in the path from source to destination.

During Phase III, we developed the first known path-finding algorithm that is loop-free at every instant. We call this path-finding algorithm the *loop-free path-finding algorithm* (LPA). According to LPA, update messages are sent only to neighboring nodes. Like previous path-finding algorithms, LPA eliminates the counting-to-infinity problem of DBF using the predecessor information. In addition, LPA eliminates all temporary loops by implementing an interneighbor coordination mechanism with which potential temporary loops are blocked before routers can forward data through them. To block a potential temporary loop, a node sends a query to all its neighbors reporting an infinite distance to a destination, before changing its routing table; the node is free to choose a new successor only when it receives the replies from its neighbors. To reduce the communication overhead incurred with interneighbor coordination, nodes use a *feasibility condition* to limit the number of times when they have to send queries to their neighbors. In contrast to many prior loop-free routing algorithms, queries propagate only one hop in LPA. Furthermore, updates and routing-table entries in LPA require a single node identifier as path information, rather than a variable number of node identifiers as in prior algorithms.

To obtain insight into the average performance of LPA, the algorithm was analyzed by simulation using the topologies of typical networks. The performance was compared with DUAL and an ideal link-state algorithm (ILS), which replicates the same topology information at every routing node. The simulation uses link weights of equal cost, and zero link transmission delays. During each simulation step, a node processes input events received during the previous step one at a time, and generates messages as needed for each input event it processes. To obtain the average figures, the simulation makes each link (node) in the network fail, and counts the steps and messages needed for each algorithm to recover. It then makes the same link (node) recover and repeat the process. The average is then taken over all link (node) failures and recoveries. The results of this simulation for Los-Nettos are shown in Table 3.1. The table shows the total number of events (updates and link-status changes processed by nodes), the total number of update messages transmitted, the total number of steps needed for the algorithms to converge, and the total number of operations performed by all the nodes in the network.

Table 3.1: Simulation Results for Los-Nettos

Parameter	LPA		DUAL		ILS	
	mean	sdev	mean	sdev	mean	sdev
Link Failure						
Event Count	88.6	43.3	49.9	18.6	19.0	0.0
Packet Count	33.7	16.6	32.6	11.8	17.0	0.0
Duration	5.4	1.7	6.7	1.3	3.1	0.5
Operation Count	77.6	25.6	69.9	18.6	478.1	27.3
Link Recovery						
Event Count	83.7	7.0	45.7	7.4	36.9	1.9
Packet Count	15.1	3.8	17.0	7.2	34.9	1.9
Duration	2.8	0.6	3.7	0.9	4.1	0.5
Operation Count	97.1	18.8	65.7	7.5	1038.9	45.8
Node Failure						
Event Count	123.8	22.96	67.6	19.7	30.5	9.1
Packet Count	47.3	8.9	45.7	3.3	25.9	7.03
Duration	5.9	1.1	7.0	1.0	4.0	0.4
Operation Count	115.8	27.35	113.6	40.1	683.3	204.2
Node Recovery						
Event Count	175.4	98.3	86.7	34.3	54.3	12.5
Packet Count	26.9	10.4	39.0	11.2	49.7	10.4
Duration	3.6	0.83	4.7	0.5	4.4	0.5
Operation Count	213.1	99.2	132.7	56.13	1626.3	440.3

It is worth noting that, as expected, LPA and DUAL have better overall average performance than ILS after the recovery of a single node or a link. The simulation results also indicate that, insofar as overhead traffic is concerned, the average performance of LPA is comparable to DUAL and ILS. LPA converges faster than DUAL in all cases; in particular, LPA is more responsive in the case of node failures, which is a concern in DUAL's performance. CPU utilization on ILS is two orders of magnitude larger than in LPA and DUAL. On the other hand, LPA converges in almost the same number of steps as ILS after link and node failures. Accordingly, LPA consitutes a more scalable solution for routing in large internets than ILS and even DUAL.

A variant of LPA is what we simply call the *path-finding algorithm* (PFA). This algorithm substantially reduces the number of cases in which routing loops can occur. Its performance is compared with the performance of an ideal topology broadcast (or link-state) algorithm and DUAL. The fact that PFA reduces temporary looping accounts for its superior performance over DUAL and ideal link-state algorithms, which we have shown by simulation.

3.2.3 Link Vector Algorithm (LVA)

During Phase III, we have developed a new type of routing algorithm, which we call *link vector algorithm* (LVA). The basic idea of LVA consists of asking each router to report to

its neighbors the characteristics of each of the links it uses to reach a destination through one or more preferred paths, and to report to its neighbors which links it has erased from its preferred paths. Using this information, each router constructs a *source graph* consisting of all the links it uses in the preferred paths to each destination. LVA ensures that the link-state information maintained at each router corresponds to the link constituency of the preferred paths used by routers in the network or Internet. Each router runs a local algorithm or multiple algorithms on its topology table to compute its source graph with the preferred paths to each destination. Such algorithm can be any type of algorithm (e.g., shortest path, maximum-capacity path, policy path) and the only requirements for correct operation are for all routers to use the same algorithm to compute the same type of preferred paths, and that routers report all the links used in all preferred paths obtained. Aggregation of information can take place adapting the area-based routing techniques proposed for DVAs in the past.

Because LVA propagates link-state information by diffusing link states selectively based on the distributed computation of preferred paths, LVA can reduce the communication overhead incurred in traditional LSAs by flooding all link states. Because LVA exchanges routing information that is related to link (and even node) characteristics, rather than path characteristics, this approach can dramatically reduce the combinatorial explosion incurred with any type of DVA for routing under multiple constraints. The simulation results obtained for LPA tend to indicate that LVA should have better average performance than any link-state or distance-vector algorithm.

An important contribution of this research has been to show that LVA is correct under different types of routing, assuming that a correct mechanism is used for routers to ascertain which updates are recent or outdated. Accordingly, LVAs open up a large number of interesting possibilities for Internet routing protocols; to name a few:

- LVAs can be the basis for the first routing protocols for packet radio networks based on link-state information.
- LVAs can be used to develop intradomain routing protocols that are based on link-state information but require no backbones, which eliminates the difficult network-management problems associated with such protocols as OSPF and ISO IS-IS.

The next steps in our research are:

- Design fail-safe mechanisms to validate routing updates in LVA based on time stamps
- Analyze the average performance of LVAs by simulation
- Implement a routing protocol based on the Routing Information Protocol (RIP) specification that can be deployed in a packet radio network with mobile nodes

3.2.4 Use of GPS Information in Routing

Garcia-Luna and Zaumen [GLAZ93] have recently proposed the first loop-free algorithm that provides multiple paths from any source to any destination in a dynamic topology. The collection of all the loop-free paths from a given source to a destination implied by the routing tables at the network nodes is called the "shortest multipath" from the source to the destination. A node in that multipath can forward packets to a destination through any of its neighbor in the same multipath, without the source having to specify entire paths or establishing connections, and without creating a routing loop (which can occur in duct routing, creating additional congestion).

We are currently investigating augmenting LVA and LPA by using GPS data to aid nodes in the rerouting of data packets aimed at mobile nodes, and to reduce the overhead of the routing algorithm. According to the proposed approach, a routing table entry for a given destination contains the network distance to the destination, a list of feasible neighbors in the multipath to the destination, and the coordinates of the destination. This implies that each node communicates its GPS to its neighbors when it transmits routing table updates. Based on this information, a node forwards packets based on the multipath toward the given coordinates of the destination, that is, a packet specifies the destination and its coordinates. When a destination moves, it sends GPS updates, which are processed by the nodes able to listen the destination's transmission. When a node receives a packet for a given destination that contains old GPS information, the coordinates are update and the packet is rerouted. We will investigate how to define geographical areas where destinations can roam without incurring routing table updates, in order to reduce the frequency of routing-table updates. This problem is very similar to the problem of hierarchical routing; however, the areas have to be defined dynamically, based on the location of mobile nodes.

3.3 Channel Access Protocols

Power usage limitations dictate a minimum number of transmitted packets. The power usage limitations mean that the receiver of a low-power sensor cannot be fully on all of the time. This restriction is imposed by the high current drain (80 to 300 mA) in today's reduced-size receivers. In contrast, current channel-access algorithms require full-time two-way connectivity for transmission and acknowledgment. Even protocols designed for special conditions such as EMCON (emission control) require the receivers to be on throughout the period when communication is expected. For traffic from the sensor to the station that monitors it, power management is not a problem. The sensor itself is aware when a packet is ready for transmission and can wake up the transceiver section; hence, normal demand access algorithms or polling algorithms can be used. The situation is very different for traffic from stations to sensors. Assuming an information bit rate of 256 kbps in the radio channel and that each sensor sends packets of 100 bytes (including headers), a packet transmittal time is 2.4ms and polling 1000 sensors requires about 4.56 seconds, assuming no collisions and about 30% framing overhead. Collisions and retransmissions are likely to occur, and radios with much lower bit rates may have to be used just for economical reasons. Accordingly, it appears that strict polling schemes are not likely to succeed with large numbers of sensors sharing a common radio channel.

There are many ways in which channel access protocols for wireless networks can be classified. For our purposes, it suffices to classify them into contention-based and contention-free protocols. In either type of protocols, dynamic allocation of theannel capacity is necessary to cope with large numbers of users.

Contention-based protocols like the tree algorithms, CSMA and ALOHA [RS90] are perhaps the most popular channel access protocols today. In CSMA/CD, which is used in the Ethernet standard, a station is allowed to transmit when it detects no traffic in the channel. A station listens to the channel while it transmits its own data; therefore, if more than one station attempts to transmit at the same time, the stations detect the collision of their data and they cease to transmit. On the other hand, if the signal of the data packet transmitted by a station is able to propagate throughout the entire channel (a cable in Ethernet) before another station attempts to transmit while perceiving the channel as

idle, the transmission is successful. Therefore, the first portion of the data packets sent by a station can be viewed as the station's reservation for the channel. Rom [Rom86] has proposed a collision detection mechanism for radio channels. The key disadvantage that we see with existing contention-based algorithms is that they require a station to contend for the shared resource every time a user has a data packet to send. Transporting isochronous media requires performance guarantees similar to those achieved with contention-free algorithms.

Contention-free algorithms can be based on either reservations or token passing. Existing reservation algorithms break the channel into a reservation interval and a data interval. Stations trying to send data over the channel attempt to make a reservation or acquire the channel control token during the reservation interval. A station sends data during the data interval if it is successful in making a reservation or receiving the token. As Rom and Sidi point out [RS90], these algorithms entail reaching an agreement on which stations need the channel and apply an arbitration scheme to decide which station should get access to the channel. Such a scheme is a priority structure imposed on th set of users, which are priority classes themselves. The Broadcasting Recognition Access Method (BRAM) [CFL79] and the Mini Slotted Alternating Priority Protocol (MSAP) [KS80] are well-known examples of this type of protocols. A limitation with current contention-free channel access protocols is that they use either a separate control channel, centralized control by a base station (as in Goodman's packet reservation multiple access (PRMA) [Goo90]), or a fixed number of reservation slots, which makes protocols such as MSAP applicable only in small user populations.

During Phase III of this project, we are investigating whether a channel access protocol can be designed that provides performance guarantees like contention-free protocols, without their limitations. A promising approach consists of applying mutual exclusion algorithms based on elections.

4. REINAS System Architecture

4.1 The Scientific Challenge

The primary scientific challenge addressed by REINAS is the conducting of real-time environmental science. In environmental monitoring and analysis, data collection, delivery, and transformation activities feed the intertwined and iterative processes of data analysis, modeling, and visualization. In this information life cycle, shown in Figure 4.1, instruments produce measurements which become observations by the application of calibration algorithms. Scientific databases must store all measurements, observations, and calibration parameters involved in this process. Data assimilation is the process of taking irregular (and often sparse) observations and creating an accurate and meaningful representation of the cent state of the environmental phenomena being observed. Computer graphics 'visualization' is the usual mode of presenting the assimilated data. Models are often used in an iterative fashion to support data assimilation and are also used to generate forecasts.

Meteorological measurements and observations are produced from a variety of instrument sources. These sources include land based weather stations, ships, buoys, aircraft, radars, and satellites. Many of these data sources are not directly accessible to REINAS but can be assimilated using data exchange and format standards adopted by major national and international weather centers.

Oceanographic measurements are much more sparse than meteorological measurements. Ships and buoys provide some data while satellites obtain ocean surface data. REINAS will initially obtain real-time measurements from meteorological stations, wind profiler radars, CODARS (ocean surface current radars), acoustic Doppler current profilers, video cameras, and thermistor chains. Additional devices will be added in the future. Despite the differences in data sources there is broad commonality in the types of information created and used by meteorologists and oceanographers and both groups exhibit interest in air/sea interaction. Activities surrounding data collection, delivery, processing, analysis, modeling, and visualization functions have a common framework, especially from a data management perspective. The challenge for REINAS is to provide a common data storage and manipulation system for these data life cycle activities while supporting the specialized requirements of each discipline.

A problem common to both oceanography and meteorology is that while observations from many sources are provided by operational communication networks managed by military, academic, and private enterprises, this data is often not real-time and is often delivered

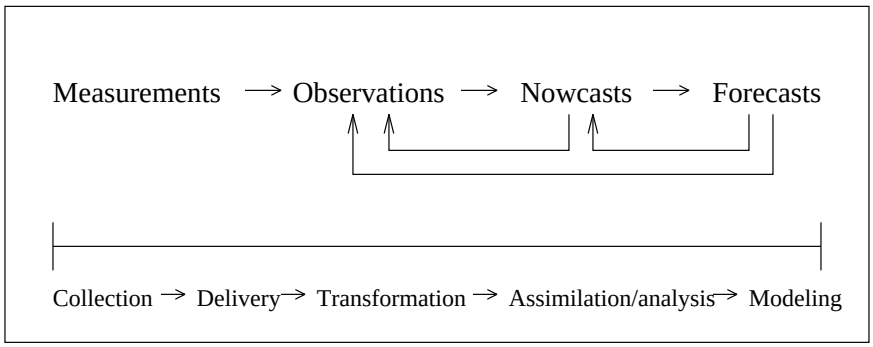

Figure 4.1: Environmental information life cycle – products and processes.

in an un-integrated fashion involving manual operations. As a result, researchers frequently miss opportunities to coordinate data collection activities as interesting phenomena occur.

REINAS seeks to address these challenges by proving a unique real-time environment supporting interactive desktop experimentation by air/ocean researchers. Such a high-quality real-time environment will motivate the adoption of new methods of scientific collaboration and data exchange. Real-time desktop experimentation is potentially promising, and is applied to some degree in related sciences due to necessity [IL92, GB92a].

4.2 REINAS System Architecture

The REINAS system architecture was designed to address the needs of several different groups within the oceanographic and meteorological communities: Operational forecasters monitor current conditions, view standard data products, synthesize new data views, and issue forecasts and warnings. Modeling scientists visually analyze products of new data models and compare these products with the outputs of other models as well as with past and present conditions. Experimental scientists collaborate with other scientists on-line, observe individual data fields from specific sources as they are collected, and may modify data collection methods while an experiment is in progress. Finally, instrument engineers add new equipment to the system, access metadata describing individual devices, observe methods of calibration, study maintenance records, and profile sensor quality.

REINAS provides a system architecture integrating user group activity (both at the office and in the field) over the whole data life cycle. Central to this architecture is tracking the lineage of data elements and system resources and supporting a common information model which makes data accessible to the entire user community.

The REINAS architecture provides the following services:

- seamless access to real-time and retrospective data,
- the performance required to support the most frequently requested products and services,
- access to named resources and devices through a common data model while supporting rapid system configuration,
- dynamic control of system devices for real-time interactive scientific investigation,
- fault tolerant methods of data collection avoiding data loss due to communication link failures resulting from environmental factors, and
- security features restricting access and control privileges of users with respect to data and equipment.

4.2.1 System Organization

As shown in Figure 4.2, REINAS is organized into three subsystems, each corresponding to a specific part of the data life cycle. The instrumentation subsystem collects data and transforms it to a standard portable format. The database subsystem stores data both on-line and on archival media, as well as providing a framework for data manipulation within the system. The user subsystem provides users access to data in the instrument and database subsystems and supports end user visualization and modeling applications. Each subsystem has its own logical network which is defined by ownership and operational

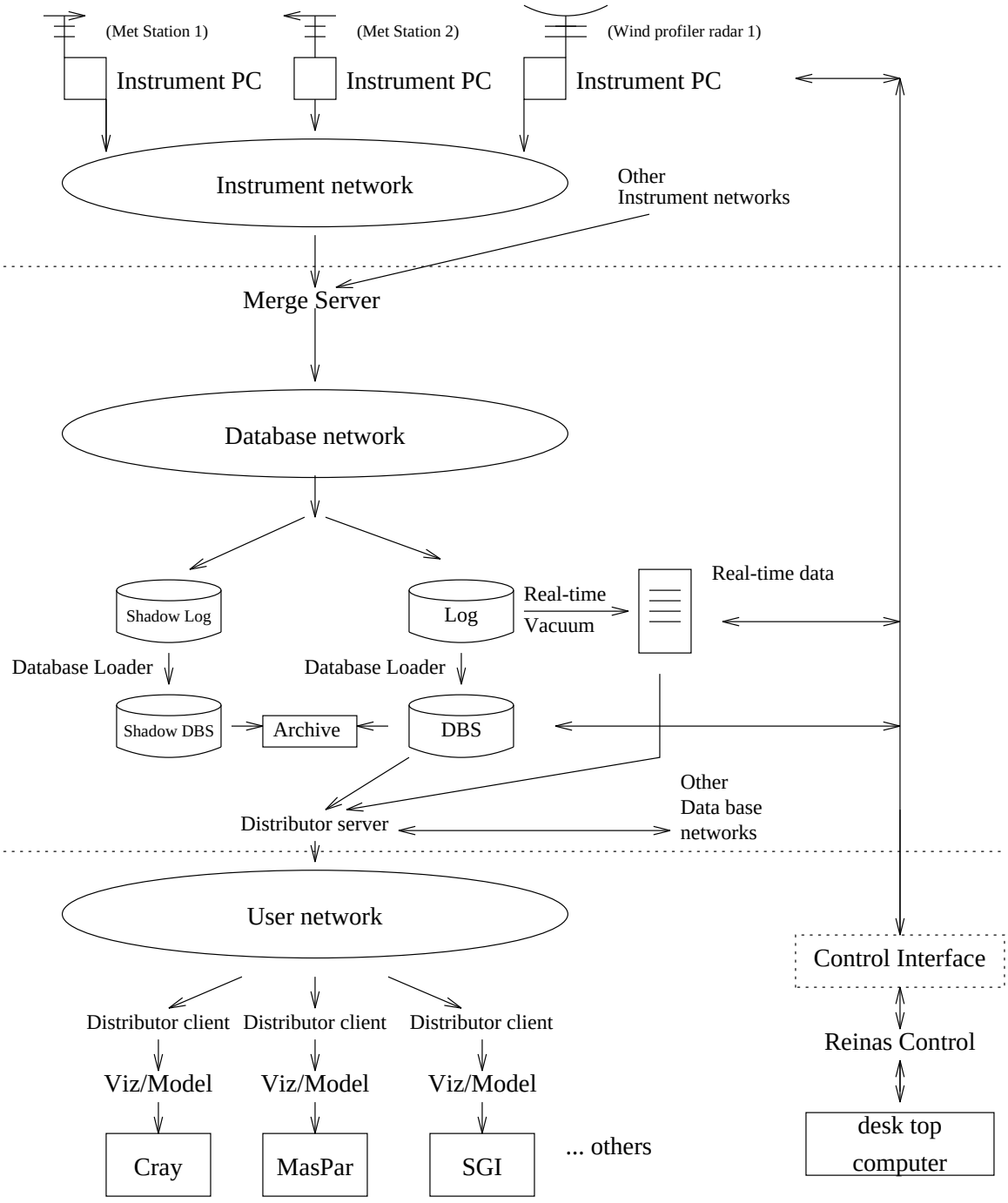

Figure 4.2: REINAS Logical Architecture

characteristics. Functionality is distributed throughput these networks, with any specific computation occurring at the most appropriate location.

The instrumentation subsystem consists of nodes containing measuring devices attached to microcomputers and connected to the Internet. Each microcomputer writes data to a log, converts it to a common format, and transmits the data to the rest of the system at a specified rate. Configuration and control commands for specific instruments and sensors are also executed at this level.

Data obtained by an instrument node is transmitted to the database subsystem. There the data is received by a *merge server* process and written to a database load log. A *database loader* process then reads the data from the log and writes it into one or more historical databases. Data is also read from the log and copied into memory resident real-time data structures which reflect current data values and simple statistics, such as moving averages. Data from both the historical databases and the real-time data structures is seamlessly accessed from end user applications by a process called the *distributor* which can provide applications with real-time state information at a given rate.

Due to the large amounts of data processed by the system, data may eventually be migrated from fast storage in the database to slower archival media such as tape or compact disc. An archiving process in the database subsystem accomplishes this task.

The user subsystem runs REINAS applications. This subsystem consists of general and application specific hardware and software which access the distributor via the Internet using an application software interface. This application interface provides a standard programming model for accessing and controlling both the database and instrument subsystems. User applications retrieve data by using the application interface to issue queries that are executed by the distributor. The user subsystem also contains the REINAS application visualization support routines.

4.2.2 Advantages of the Architecture

Given this layered architectural approach, resources can be duplicated throughout the system and subsystems can communicate with multiple instances of subsystems in higher and lower layers. For instance, instruments can provide input to multiple database subsystems, and a database subsystem can receive input from more than one instrument subsystem. Likewise, a single database subsystem can service multiple user subsystems and a single user application can request services from multiple database subsystems. Database replication can be implemented using multiple database subsystems.

The design of each layer of the system is optimized toward a specific task. Since microcomputers are used as nodes of the instrument subsystem, naming and access control of individual instruments can occur at a low level within the system, providing a framework which is both secure and easy to navigate. Dedication of data acquisition activity to distributed microcomputers permits the database subsystem to be optimized for the manipulation and storage of data through the use of the Andrew file system, large storage devices, and large database servers. Lastly, since user applications for visualization and modeling are computationally expensive, these activities can occur on specialized hardware and do not adversely impact machines involved in other tasks in the system.

Finally, the separation of concerns between layers makes the system extensible through the development of regular interfaces between machines on different layers. Since the different architectural layers all communicate through the Internet, any machine attached

to the Internet can be added to the system through a protocol which conforms to the appropriate interface.

4.3 Data Management from Instrument to Historical Use

4.3.1 Overview

REINAS data management supports user needs throughout the data life-cycle. Scientific data management is not simply a repository problem. Scientific data management must also support a range of tasks starting with data collection and proceeding through interpretation and scientific collaboration. REINAS data management must also address systemic problems challenging current data management practices in ocean and atmospheric sciences: integrated data/metadata handling, data lineage, and quality assessment.

Data management occurs within all system elements. Operational users require immediate access to the instruments and to the state of the network, while retrospective researchers require access to historical data.

Instruments produce different classes of data e.g. numeric, image, video, acoustic) and have different operational profiles. Computational and laboratory processes are also sources of data and are complicated by interpretation activities performed by scientists with differing vocabularies and research backgrounds. Data management functions must provide transport, real-time access, data storage, and retrospective data access. Equally important, each subsystem must support the capture, storage, and access of information documenting the context, content, structure, and representation of the primary data. This class of data is called *metadata*. This integrated end-to-end approach to scientific data management has implications for all subsystems.

The instrument subsystem is more than a transport mechanism. The reliability and integrity of data transport must be assured and a data log synchronized with database loading. Since instrument control protocols can dynamically configure instrumentation suites, instrument configuration history must be maintained.

The data management subsystem must support functional integration, data integration, and data pedigree and quality. Functional integration is provided by the specification of a consistent framework for the tasks of scientific information creation and use. This framework must:

- provide access to real-time, retrospective, and predicted future states through a consistent interface,
- integrate data acquisition, access, analysis, and visualization tasks into a common computing environment, and
- support both environmental scientists and engineers responsible for the development and maintenance of the system.

A stable database architecture is needed which can accommodate the multimedia, multidisciplinary, and multiformat data delivered from the data sources. This architecture must also cope with continuous addition of new metadata and new quality assessments of existing data.

Ultimately, data is analyzed using the visualization subsystem. This subsystem provides a standard user interface by which the user can query the database subsystem and control both the database and instrument subsystems. Also included within the visualization

subsystem are tools for producing standard environmental science information products and 4-D visualization products. Visualization of routine monitoring activity, experiment history, and sampling event history is possible using temporal metadata.

4.3.2 Storage and Access for Long Term Use

Effective end-to-end scientific data management requires support for long-term data use as well as support for the activities of collection through interpretation. Information modeling techniques were applied to the meteorological and oceanographic science domains, to typical operational scenarios, and to high profile requirements. The MBARI Scientific Information Model served as a starting point and was extended to support more diverse representations of observational data in the environmental enterprise [GBD+89, GB92b].

The primary information modeling goals are:

- to identify classes of information created and used by scientists and engineers working in all areas of oceanography and meteorology,
- to analyze objects in each class to identify broad class generalizations and critical specializations,
- to synthesize a data architecture at the appropriate level of abstraction which is unlikely to change in basic structure as technology and science evolve, and
- to identify where the architecture will extend and evolve over time.

The rationale behind these goals is that scientific domain, technology, and system changes can be reflected by changes in database content and not in database structure.

Data paths were followed from source to sink to identify and describe concepts and objects important to each type of data source. End user information needs were analyzed via their data management query forms to identify important objects and concepts which spanned data source types. The result of this analysis was a set of generalized concepts which mapped well to specific objects and concepts while identifying areas where the architecture will evolve.

As an example, this data flow analysis identified the following objects and attributes for an ocean temperature sensor:

- sensor (model, best accuracy, best resolution, range, serial number, operator ID number),
- calibration history (date of calibration, coefficients, date range of applicability, person who performed the calibration),
- data acquisition instrument hosting the sensor (model, configuration options, configuration history),
- processing history (baseline algorithms, software programs implementing algorithms, quality control procedures, person who performed the processing), and
- instrument platform hosting the data acquisition instrument (type, name, responsible person/organization, log of events and malfunctions),
- and data management log (processing events, reprocessing events, archive tracking).

An analysis of user information needs was performed by studying typical user queries which were source independent. For example, the query *"return all collocated temperature and nitrate observations taken at site H3 during the winter season where the temperature is greater than 13 deg-C"* indicates the need for named locality descriptions in addition to a precise space/time tag for each observation to determine collocation. The query *"return all temperature and salinity observations taken from the Pt. Lobos research vessel where the data collection runs are marked as questionable"* indicates the need to track the platform (in this case a ship) which collected the data and the requirement to associate quality assessments with individual and aggregate sets of observations.

4.3.3 Realms:

The architecture consists of major information groups called realms which contain enough substructure to fully capture the semantics of the major types and subtypes of the realm. These realms include: systems, processes, parameters, localities, data generation activities, descriptions, quality assessments, and measurements/observations. Objects in each realm will participate in intra-realm and inter-realm relationships.

The system realm contains generalized and specialized attributes of major classes of systems which occur in the environmental enterprise. Examples include instrument platforms (ships, aircraft, satellites, remotely operated vehicles, buoys, or land meteorological stations), instruments, instrument platform subsystems (winchs or cranes), sensors (temperature sensors or wind speed/direction sensors), and computers.

Process realm objects include those items which document automated or manual procedures intended to accomplish a specific purpose. Examples include calibration algorithms for environmental sensors and laboratory procedures for performing sample analysis.

Objects in the parameter realm are used to define the types of environmental properties which may be represented in the database and the logical and physical form of their representation. This realm supports the requirement to store and reconcile data representing the same concept in different formats.

The locality realm contains objects which represent spatial features of interest in their own right or as spatial identifiers for other database objects. Locality features may be points, two or three dimensional regions, linear networks, or names with no specific boundary definition. Regular sampling/monitoring sites, the spatial extent of a data collection activity, or the spatial extent of an observation data aggregate may be defined.

The data generation realm contains objects defining those things which can be part of the data generation process or document the process. A few important generalizations in this realm include expeditions, projects, experiments, data collection runs, and sampling plans.

The measurement/observation realm contains the primary data of interest to the environmental scientist. Direct sensor outputs, derived observations of environmental properties, and ancillary information which may be tagged with each individual measurement/observation are included. In addition, aggregations of individual observations may be identified and tracked. For example, an image may be seen as an aggregation of the individual pixels comprised of separate, distinct, and accessible environmental observations. Other typical aggregate types include time series, vertical profiles, and spatial/temporal grids.

Quality assessment realm objects document multiple assessments of the quality of individual observations or aggregates of those observations. These assessments may include both quantitative and descriptive assessments by data users.

The descriptive realm contains objects which are used to document the environmental science enterprise and the database system itself. General object types such as person, remark, and calculated summary parameter may be associated with any other object in the database. This is the realm where logical, physical and other special data formats may be described.

Beyond the identification of these generalized realms, another challenge remained: to determine the appropriate logical/physical packaging of measurement/observation data. Two primary alternatives were considered, a state vector representation and a data stream representation.

A state vector representation packages all observations of all types from all collocated sensors into an observation group with a single space/time tag. This form of organization provides the most effective support for queries of the type *"What is the complete state of the environment at point x,y,z?"*. Alternatively, in the data stream model all observations of the same type from all sources in the region are packaged together and physically sorted by time. This model best serves queries which assess the state of a small number of environmental variables over a large viewing area, for instance, *"What is the state of sea surface temperature and wind velocity fields around the Monterey Bay area?"*.

This issue is resolved based on the answer to two questions: how are the data most often requested? and how are the data most effectively managed in terms of data/metadata linkage? The answer to both of these questions indicate a strong preference for the data stream information model.

4.3.4 The Schema

A REINAS schema system is defined as a collection of hardware, software, and procedural components that work in cooperation and perform a specific function or produce a specific product. Systems have configuration, malfunction, and maintenance histories. A system hierarchy exists, as does a REINAS name space reflecting this hierarchy. A system belongs to a parent system and may have child systems. An important system attribute is its type. System types include platform, instrument, sensor, and computer.

Definitions

The schema defines a REINAS process as an automated or manual procedure initiated to accomplish a specific purpose. Every process is an instantiation of a process type. As with systems, all processes exist in a process hierarchy. Processes may invoke, or be invoked by, other processes. An important process attribute is its type. A process type can be either procedure or algorithm. A procedure is an operationally defined activity that produces a known result. A procedure often reflects a human activity performed according to a prescribed sequence of steps.

An algorithm is defined by the schema as a known calculation or mathematical transform. An algorithm often corresponds to a published scientific data processing technique. Algorithms are used in many places in the REINAS scientific database to transform measurements into observations.

A program is defined as a process subtype corresponding to an executable computer program. A program may implement one or more algorithms and any number of programs may use the same algorithm. The database keeps track of the transformations performed on data by keeping track of the algorithms and programs that have transformed the data.

An environmental property is a measurable quantity such as temperature, humidity, or wind speed. An environmental property may be represented by multiple parameter types. For example, temperature might be represented by Kelvin, Fahrenheit, or Centigrade.

The scientific data types used by REINAS are referred to as parameters. Parameters define the primitive formats containing the scientific data in the database. Parameters have types and are associated with the systems and processes that can instantiate elements of that parameter type. Typically, a parameter type is a representation of some environmental property. A value domain is a description of legal data values. Primary units (Kelvin, Fahrenheit, *etc.*), unit modifiers, and legal value ranges may be specified by a value domain. An instantiated parameter is called a data element or (sometimes) a value, field, or item.

Activity Hierarchy

The scientific data stored in a REINAS database is organized into an activity hierarchy as shown in Figure 4.3. At the top of the hierarchy is an expedition. An expedition describes a operational activity with a well defined mission. Ship cruises, aircraft flights, satellite orbits, buoy deployments, met station networks, and remotely operated vehicle dives are all examples of expeditions. An expedition can be associated with one or more projects or experiments.

A data run occurs when collocated systems sample the environment for a continuous period of operation using a fixed configuration of data sources. A data run may be a short activity or may last a considerable period (a one month buoy data run). A data run has an associated locality which represents the extent of the spatial coverage of the data run.

Measurements and observations from the same data run, and of the same type, source, and processing lineage are combined into data streams. Data streams corresponding logically to the parameter types produced by the data sources during the ata run. Individual point observations, profile aggregates, and 2-D or 3-D fields may all be defined as single elements within a stream. A stream element, or group of elements, may also have associated spatial extents. These stream element localities will fall within the locality associated with the data run in which the data stream was produced.

The elements in a data stream are either aggregated or non-aggregated. A non-aggregated element is instantiated as a single parameter value. The parameter can be scalar or complex. An aggregated element is an array of parameters. Aggregated parameter have internal structure (the dimensionality of the array) of which the database is aware. A stream element for an aggregated parameter has the same format as for any other stream element.

A typical aggregated parameter is a wind profile vec. In this case, a profile consists of multiple observations that have all been collected at the same time. Profiles are linear. Complex calculations do not need to be applied to a profile to obtain the locations of each of the elements of the profile. This is different from a satellite image where the relationship between pixels in the image is non-linear and unique to every image.

Data streams are classified into a common type if:

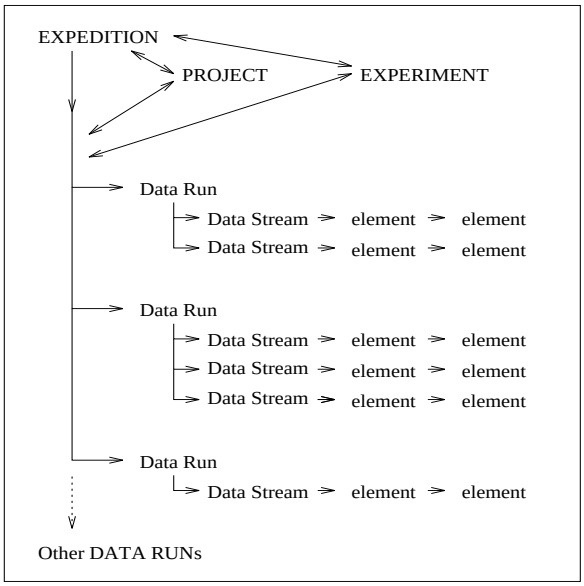

Figure 4.3: Database Organization

- they originate from the same system type and process type,
- they contain the same parameter type, and
- they have identical physical representations.

All primary scientific data is stored in containers. Containers are designed to host time-ordered stream elements from compatible data streams. Such data streams contain elements with logically consistent parameter types and physical representations. Elements from different but compatible data stream types can be stored in the same container.

This schema provides an extensible framework for managing oceanographic and meteorological scientific data. The schema describes the central items that must be tracked to support oceanography and meteorology research. Users need not develop custom data handling solutions as particular data needs can be supported by simply changing database content rather then the schema definition.

To further illustrate this schema, Figure 4.4 shows objects that need to be populated to describe a data stream.

4.4 Montage DBMS Performance

A key feature for any database management system (DBMS) to be used in the REINAS database subsystem is performance. Since large amounts of data are to be merged into the database, any DBMS in the database subsystem must possess an average insertion time which is less than or equal time to the average arrival rate for data in the system. Likewise, archival data queries from the visualization subsystem must be processed in a timely manner. Because the database network may consist of a single database system, the performance of the entire REINAS system may depend on the correct choice of DBMS.

Up to this point in the development of REINAS, the Montage DBMS has been one of the most promising database systems reviewed for potential use as the initial node in the REINAS database subsystem. This selection was made because of the combination

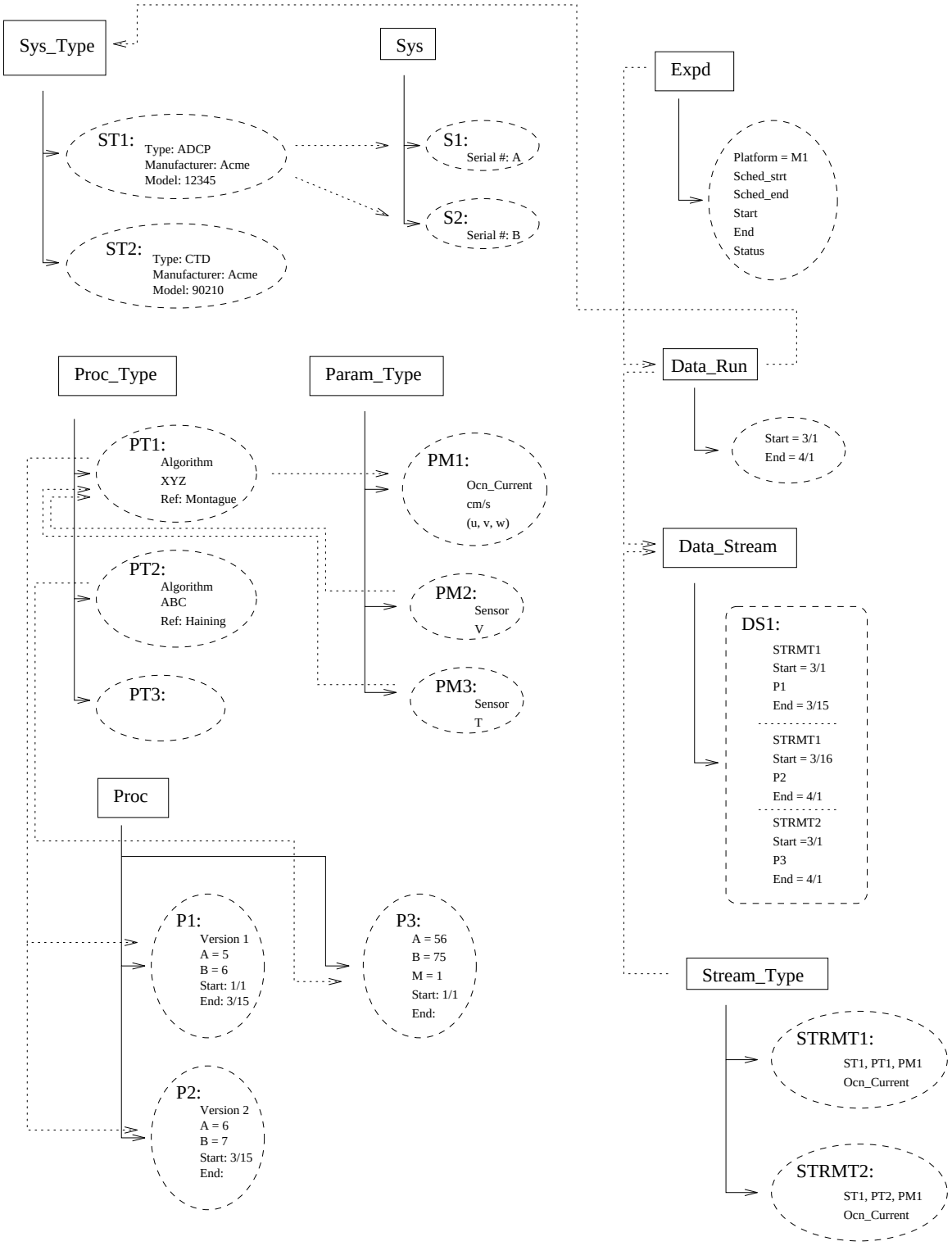

Figure 4.4: An Example Schema

of relational and object-oriented features available for use in Montage. Because DBMS performance is a key factor in the success of REINAS however, the performance of the Montage DBMS must be evaluated.

To this end, a simple stress test has been devised to determine how Montage performs under the loads that REINAS is likely to impose. The following subsections explain the design of test, present the experimental data, and evaluate the results to determine if Montage will be useful in REINAS.

4.4.1 Test Design

One of the potential information bottlenecks in REINAS is the average input service time for the DBMS in the database network. If this time is too long relative to the arrival rate of information, a majority of data within the system will be stored in logs which are read into the database and be unavailable to users on the visualization network.

The initial test on Montage is designed to determine this input service time by saturating the database with input, and measuring the amount of time require to insert into the database. The data used during the tests was taken from four different met station data files each containing approximately twelve thousand sets of readings. Each file was read by a program and the data was input into the database using a committed transaction as fast as the machine running the program could manage.

To determine how the input service time of Montage compares with other database management systems, two other database systems were also tested: Postgres and Sybase. Postgres was chosen because it is the research product on which Montage is based. Sybase was tested because it is an established commercial database product that is currently being used to store data produced by the met stations that will be incorporated into REINAS.

4.4.2 Experimental Data

Two measurements of input service time were taken, CPU time and clock time. The first was taken to determine if the input service time was due to some cost of network transport incurred on the computer running the input application. The second measured the time of input itself. Because the *clock* command available on Sun workstations is only accurate to within 16.66667 microseconds, input times were measured in groups of one hundred to provide higher accuracy while providing some indication of how the insertion time varied during the course of the test.

Figures 4.5 to 4.7 present the clock insertion times for four Sun workstations concurrently running the program which input data into Postgres, Montage, and Sybase database systems. Cedar, fir, cypress are Sparcstation IPCs. Willow is a Sparcstation IPX. Maple is a Sparcstation SLC and was used in place of cypress for two of the test runs because cypress was unavailable.

Figures 4.8 to 4.10 present the corresponding CPU times.

Montage and Postgres were run on a Sparcstation 2 and wrote data to a 2 gigabyte disc connected to the workstation on which they were running. The Sybase server ran on a Sparcstation 10 with 2 processors and wrote to a disk accessed via NFS.

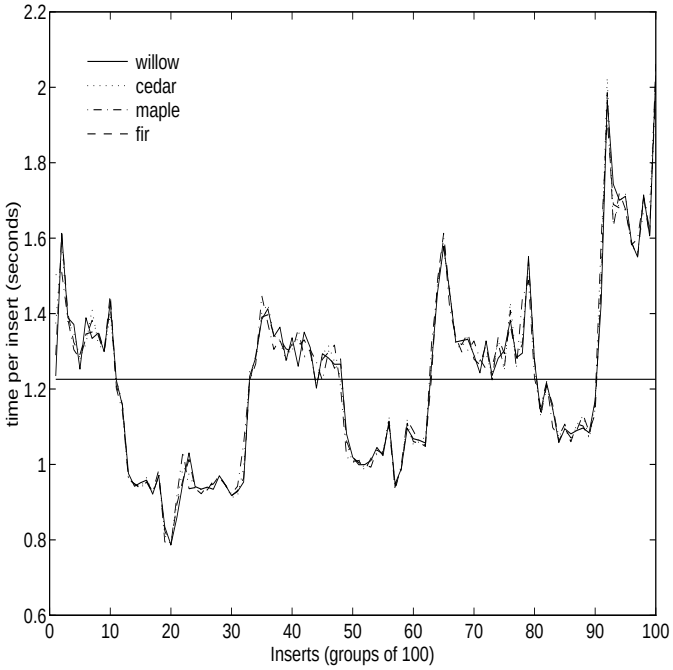

Figure 4.5: Insert clock time under Postgres.

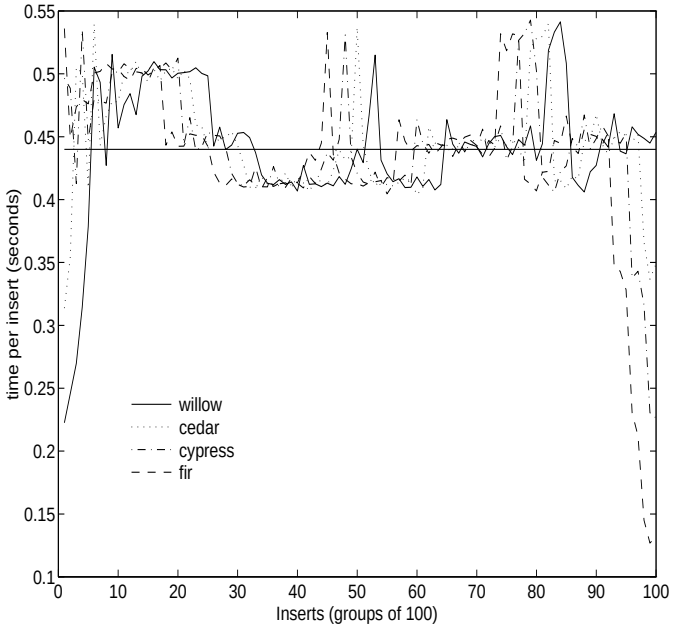

Figure 4.6: Insert clock time under Montage.

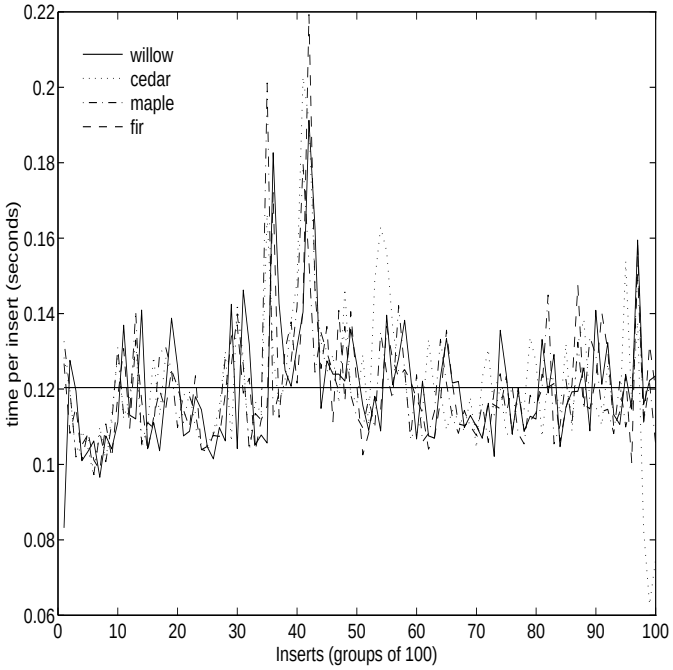

Figure 4.7: Insert clock time under Sybase.

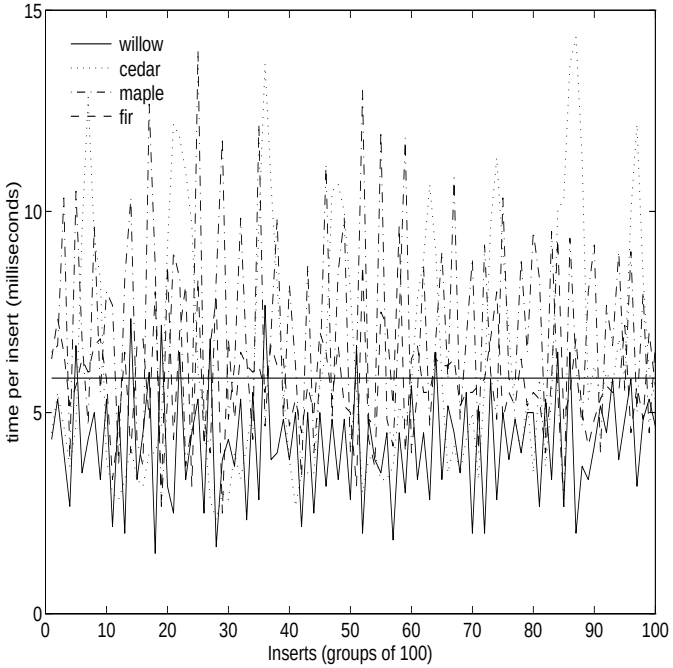

Figure 4.8: Insert CPU time under Postgres.

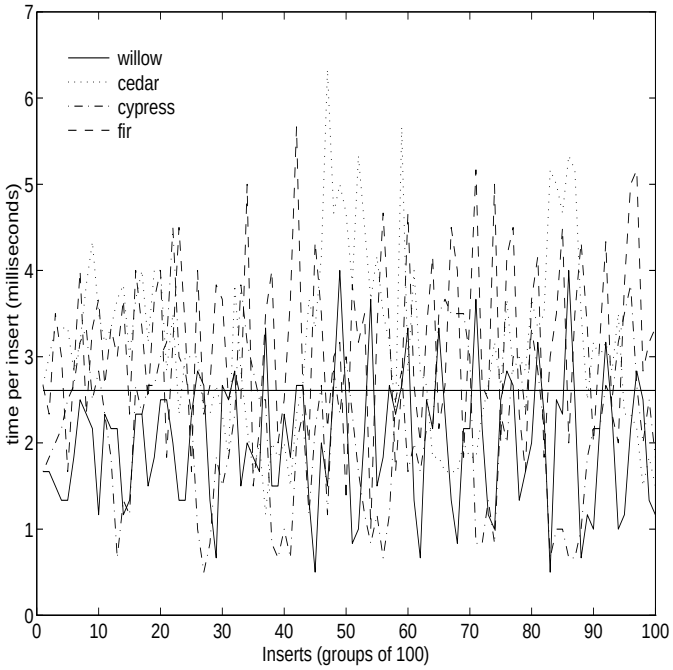

Figure 4.9: Insert CPU time under Montage.

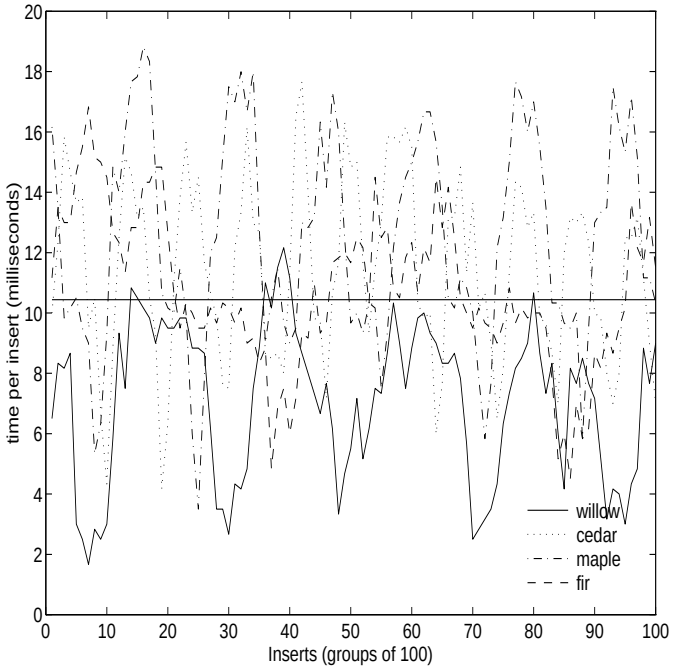

Figure 4.10: Insert CPU time under Sybase.

4.4.3 Results and analysis

These tests yielded a number of interesting observations of consequence to the design and development of REINAS:

- The similar input times for all four machines in both the Montage and Postgres runs may indicate that real time database performance is bound by the load on the single set of disk heads writing the data to physical memory.
- While the real time performance of Sybase was best, Montage came close running on less powerful hardware.
- The performance of Postgres is inadequate for REINAS use with an average input service time over one second, the normal arrival rate for met station data.
- Finally, no clear correlation seems to exist between CPU time and clock time for inserts into the database.

While this test examines one potentially critical performance factor of the Montage DBMS, it does not reflect operational use. If Montage were used as the only node of a database network in REINAS, it would be required to efficiently insert data into the database while active queries exist. Informal tests with an interactive SQL interpreter used to make queries on the data being inserted indicates that Montage performance degrades considerably. Further work tests will be required to determine the extent of this degradation.

4.5 Current Implementation Status

At the present time, the nodes in the instrument network are conventional low-cost 486 PCs running BSDi UNIX and communicating using SLIP (Serial Link IP) or PPP (Point-to-Point Protocol) over leased lines. The instrument node log is implemented using Recoverable Virtual Memory [SMK+93]. The database network layer consists of a Sparcstation/2 and an IBM RS/6000 workstation, both running extended Relational database software. Both Montage and OpenIngres are currently under evaluation [Sto93]. Visualization activities are being performed on Silicon Graphics and Hewlett Packard workstations. Security is to be implemented using Kerberos [SNS88]. The Andrew File System [HKM+88] will be used to provided a single file storage hierarchy for the system.

4.6 Related Work

The REINAS system is a large effort incorporating many research disciplines. Distributed computer technology is being used by projects such as the SEQUOIA 2000 to support large environmental databases [Sto92, SSAA93]. REINAS differs from SEQUOIA 2000 due to its emphasis on real-time desk-top experimentation and its regional focus.

Environmental and GIS systems which focus on coastal or regional air/ocean science and apply geographical information systems (GIS) and visualization techniques are being built. REINAS expands on such approaches by providing a common integrated data model to enhance the value of the scientific data collected.

Developing database schemas which track the metadata associated with a given scientific enterprise or scientific discipline is an ongoing activity [WAW+92]. REINAS will develop a model that supports the oceanographic and meteorological communities and has broad environmental application.

The application of computer science techniques to the conduct of natural science is being widely investigated, especially in areas such as visualization and the application of object-oriented databases [CHMP93]. The REINAS project is developing a unique visualization technology based on smart particles and is investigating object-oriented approaches to the common data model.

REINAS applies and extends many traditional computer science techniques. REINAS applies distributed system techniques in the areas of security, naming, and client-server application design [HKM+88, Sha86]. REINAS uses an object-oriented database design to encapsulate data within the problem domain [HPbC93, SD91]. The design will support a multidimensional structure tailored to analytical as opposed to structural queries [Sta93].

5. VISUALIZATION SYSTEM DESIGN

The visualization component of REINAS is designed to meet the various needs of its users as identified in [MLP+93b]. The highlights of the visualization include: an integrated interface for users to get to their data, either directly from the sensor or through the geographic database; real-time monitoring and retrospective analyses of environmental data; and use of spray rendering for explorative data visualization. These are described below.

5.1 Preliminaries

5.1.1 Region Selection

Originally, the design of the visualization component assumed that the physical scale of study would be comparable to the Monterey Bay. This has since been expanded, at the request of some of our users, to a larger area. Thus, we have added two mechanisms to allow users to navigate through the larger space. The first method allows the user to zoom in/out and pan around using a combination of mouse and button selections. This is desirable for looking at regions close to the current area of study. The second method provides users with a 3D graphical browser to select a region of interest. This method is preferable when the user wants to jump around and look at geographically distant data sets.

5.1.2 Default startup

Since there are several classes of users with different needs and access levels, we have provided a customizable startup resource file. Each user can create a file called ".spray" in their home directory which specifies how the visualization program will appear when they first bring it up. A default system startup file is also provided for those who don't have a ".spray" file. There are several things that can be specified in the startup file including, but not limited to: region of interest (pair of lat/long coordinates), projection method (e.g. lambert conformal), visualization mode (i.e. monitor, forecast, analyze), etc.

5.2 Visualization Modes

5.2.1 Monitor Mode

In this mode, users can watch the most current state of the environment. The underlying routines are calls to the XmetServer. The look and feel is slightly different. See Figure 5.1. Users have a bird's eye view of the region of interest. Environmental sensors are represented by simple icons. Users can select one or more icons to monitor the readings from those sensors. Selection of sites can be done by clicking on the icons or by clicking on the items in the pulldown menu. Users have the option of obtaining a qualitative (see interpolation below) or a quantitative view. For the latter, a popup window is provided for each selected site. Users can then select the field parameter they want plotted. To get a qualitative view of the environment, at least three sites with a common field parameter (e.g. temperature) must be selected. Users then have choices on different interpolation techniques to interpolate the field. Difference images (Figure 5.2 and Figure 5.3) among interpolation methods can also be generated to understand the artifacts produced by some interpolation methods. Data

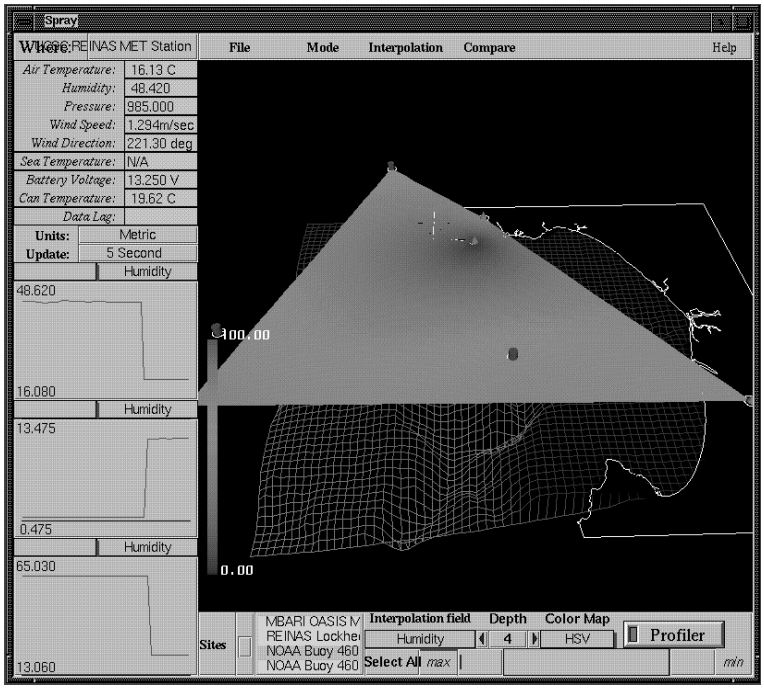

Figure 5.1: View of the Monterey Bay showing sensor locations and an interpolated humidity field from a subset of these sensors.

quality and drop-off rates of sensors are also visually mapped to transparency values (see Figure 5.4).

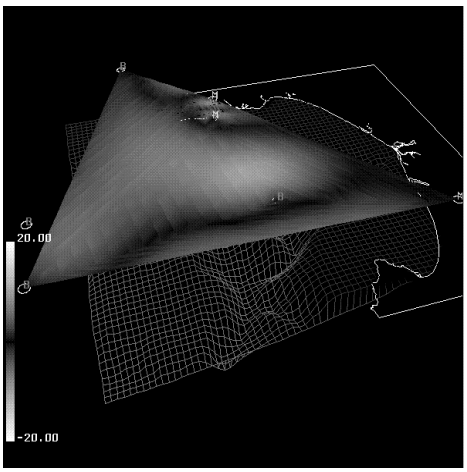

Figure 5.2: Difference image between multiquadric and inverse square distance interpolation

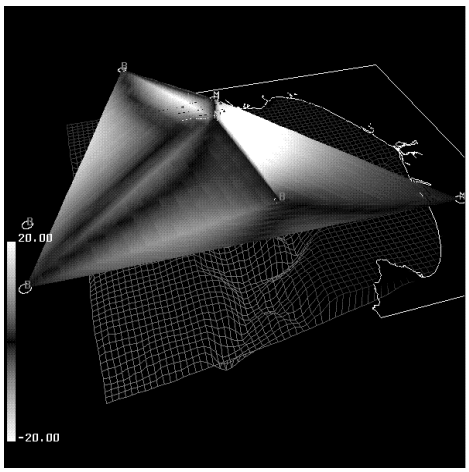

Figure 5.3: Difference image between multiquadric and linear interpolation

Interpolation: When dealing with sparse and scattered data sets, care must be taken when doing interpolation. We have studied several methods such as Shepard's interpolation and Hardy's multiquadrics [PS93, PAFW93], and will be investigating these further for adaptation to handle noisy data and those with varying degrees of confidence.

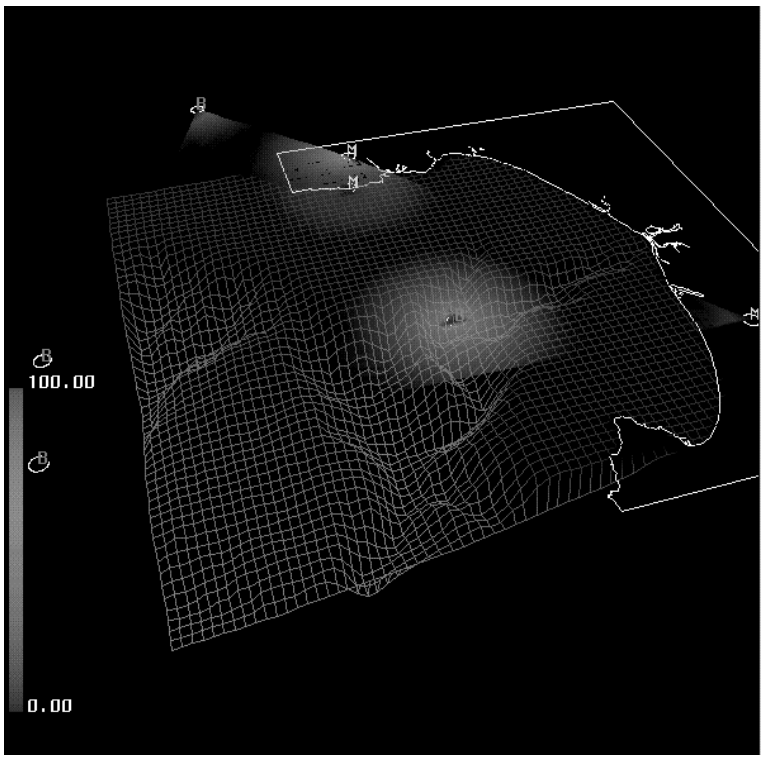

Figure 5.4: Data confidence level is mapped to opacity.

5.2.2 Forecast Mode

Operational forecasters will want to look at standard products from forecast models and satellite observations. Figure 5.5 shows a mock panel of the forecast mode. Buttons are provided for selection of standard forecast products. In addition, users can customize forecast products according to their needs. For example, user-specified contour spacing, user-specified pressure height, etc. These parameters can be specified textually or with sliders.

5.2.3 Analysis Mode

This is where most of the visualization efforts have been concentrated. It allows users to explore large data sets interactively using different visualization techniques. It is also extensible and can easily grow with users' needs. We describe the analysis mode in three sections: spray rendering, mix and match, and database interface.

1. **Spray Rendering:**

   We provide users with the metaphor of spray painting their data sets as a means of visualizing them [PS93, PAFW93]. In its simplest form, data are painted or rendered visible by the color of the paint particles. By using different types of paint particles, data can be visualized in different ways. Figures 5.6, 5.7, 5.8, 5.9 illustrate some possible ways of visualizing different data sets. The key component of spray rendering is how the paint particles are defined. They are essential smart particles (or sparts) which are sent into the data space to seek out features of interest and highlight them.

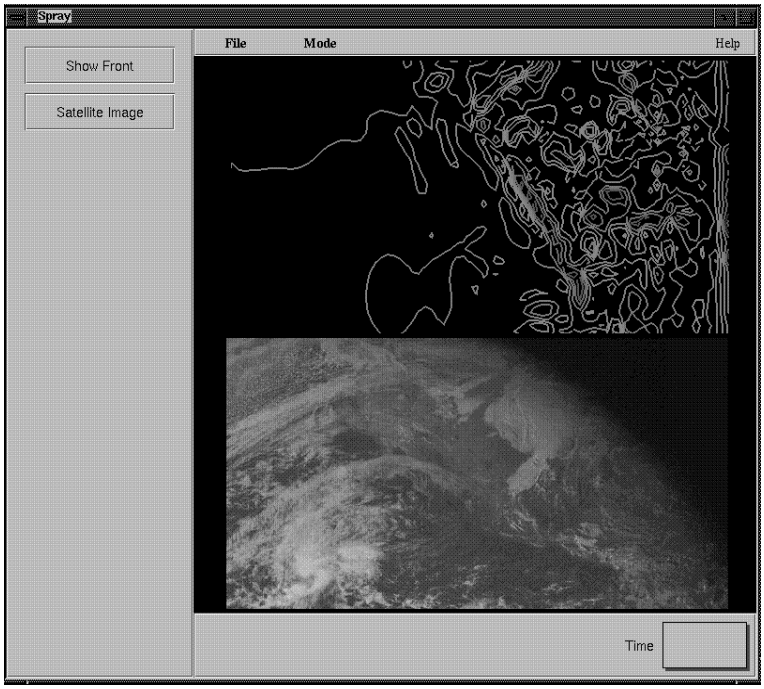

Figure 5.5: Mock panel showing a possible forecast product. Buttons will be added to provide user customizable products.

Among the advantages of this visualization framework are: grid independence (sparts operate in a local subset of the data space and do not care whether data is regularly or irregularly gridded), ability to handle large data sets (sparts can be "large" and provide a lower resolution view of the data set or they can be "small" and provide a detailed view of an area of interest), extensible (it is easy to design new sparts). Sparts can also travel through time-dependent data sets.

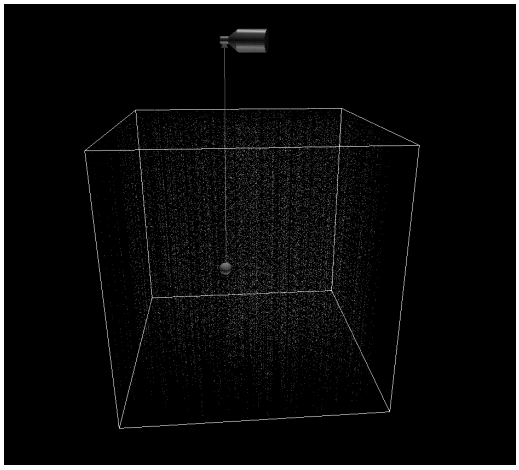

Figure 5.6: Dust clouds showing internal structure of data.

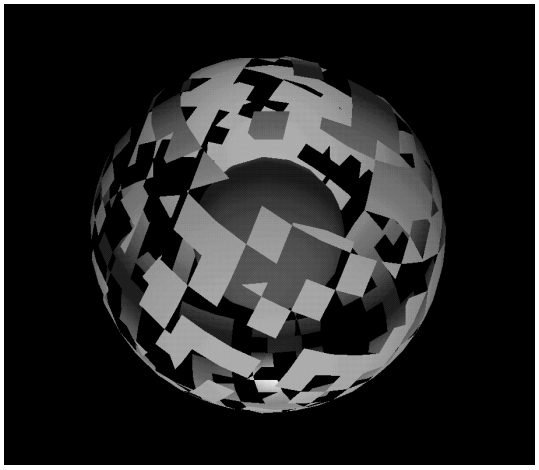

Figure 5.7: Two different iso-surfaces.

2. **Mix and Match**

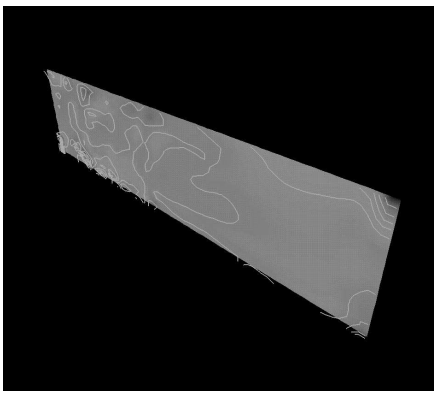

Figure 5.8: Contours and pseudo-color sparts.

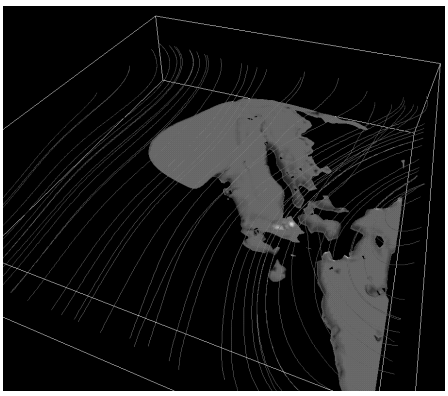

Figure 5.9: Stream-tracking sparts.

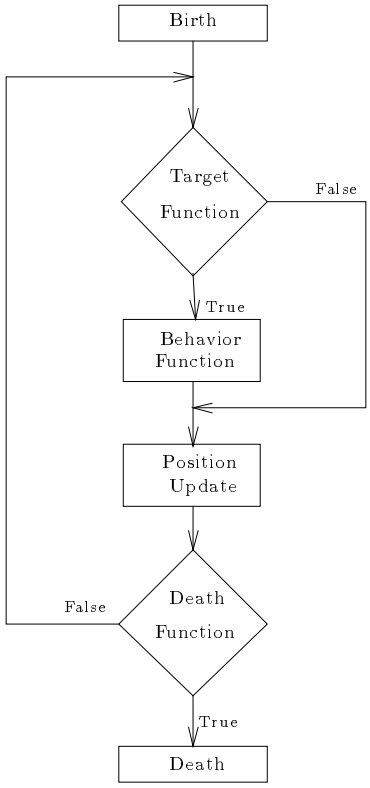

Figure 5.10: Flow chart illustrating the life-time of a typical spart.

This provides users with the ability to graphically create new sparts by mixing and matching different spart components [PA94]. As mentioned earlier, sparts generally seek out and highlight features in the data set. A more careful examination reveals that we can further refine spart actions into four components: targets, behaviors, position update, death/birth functions. Target components are functions that specify what the spart is looking for in the data; behaviors specify what the spart is suppose to do once the target is found, position update function specify where the spart moves

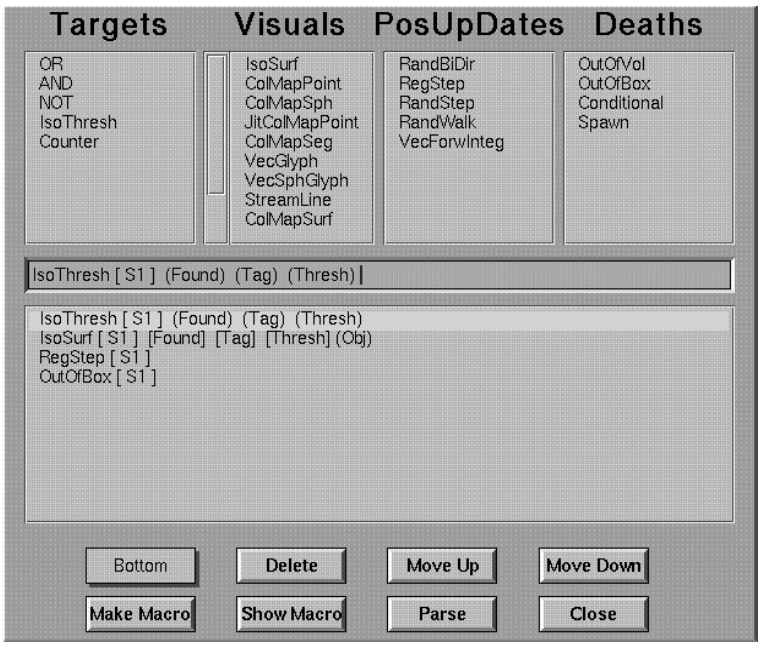

Figure 5.11: Browser for spart components and the construction of an iso-surface spart.

to in the next time frame, and death/birth functions specify when the spart has to terminate itself or when new sparts are born. Sparts can have zero or more target functions. Multiple target functions may be used to fuse different data sets together. Behaviors typically leave graphical objects behind but may also leave invisible markers for communication with other sparts. For efficiency reasons, sparts typically have at least one death function. Figure 5.10 shows the life cycle of a spart. Figure 5.11 shows the construction of a spart and how it might be edited and modified to produce a different visualization effect.

3. **Database Interface**

   The interface makes three main functions possible. First, the interface shall provide query of the data base, instruments, and other application programs. Second, the interface shall provide control of instruments, data base, and application programs. Third, the interface shall provide recording and archival into the database, and to other application programs. Additionally there are different types of queries and recording: real-time and nonreal-time. The real-time queries and recording are those streams optimized to gather current data from the sensors and data that is sent to collaborating applications. Other queries and recording will be non real-time, but will allow for local caching of data for animation display. The three functional requirements will be briefly reviewed below.

   (a) *Query.* The programatical interface allows specifying a real-time or a retrospective access. Queries must provide strict security for ownership of data, instruments, and experiments. Timing is provided to allow application programs to control the update frequency of real-time data, and also to respond to events.

   (b) *Control.* The instruments, applications, and data base have parameters that are controlled by authorized applications. This control is done through functions.

(c) *Recording.* Saving results of scientific inquiry or visualization is done by recording. Recording allows the archival of the data in the data base, or the transferral of the data to an archival unit like other applications, output, tape drives, or video. These results can then be queried.

A spart's target function may be associated with one or more data sets. These data sets are treated as streams coming off the database server. As the user moves the spray can around, different parts of the data set are being explored. Thus can parameters are translated into database query calls that pulls in the appropriate chunk of data into cache for the sparts to work on. These type of calls are standardized through an API.

5.3 Portability

Currently, the visualization components are programmed and tested using SGI's IrisGL. The graphical user interfaces are built using a public domain package called Forms. Theoretically, this program can run on other platforms that support IrisGL. Ports from third party companies such as Media Vision, DuPont Pixels, etc., allow IrisGL programs to run on Sun, Dec, Amiga and other platforms. These has not been tested to date.

The longer term plan is to port the visualization program to OpenGL. One can consider OpenGL as a window independent version of IrisGL where the graphics functions are left pretty intact and different library functions are provided for dealing with different window and mouse events. Several vendors have already signed on to develop and support OpenGL applications on their platforms. The list include SGI, IBM, DEC, and MicroSoft. Although some vendors have already released OpenGL, we are waiting for it to stabilize before doing our conversion. At the earliest, this would commence in the 3rd quarter of 1994.

5.4 Collaborative Visualization

Most visualization platforms to date run in "single user" mode. This means that users independently create their visualization products in front of a graphics workstation. There is no opportunity to collaborate with colleagues. And communication typically takes the form of static hardcopy reproduction where the transmittal time may take on the order of days. There is also little or no sharing of data or information. Collaborative visualization hopes to break down some of these barriers. In particular, it will connect geographically distributed but networked graphics workstations together to provide users with a means of sharing results and/or data, interactively creating visualization products and generally improving the communication bandwidth among colleagues. We identify two basic modes of operation that need to be supported:

5.4.1 Briefing Mode

In this mode, one of the workstations act as the master. All other connected workstations act as slaves. This is essentially a "show-me" mode where the user at the master workstation have control of the views of all the connected slave stations. Weather briefings can be conducted in this mode. Auxiliary audio and video tools will also be provided to facilitate communication among the participants.

5.4.2 Collaborative Mode

In the collaborative mode, a number of participants can contribute in the creation of a visualization product over the network. There are several components that are needed to make this feasible: session manager, sharing data/cans, floor control, multiple window, audio/video support, different collaboration/compression levels.

The session manager is a piece of software that maintains a list of ongoing sessions and the participants in eachsession. A session consists of a group of participants working on a common theme or problem. Participants may join or leave the session at any time. Thus, the session manager needs to inform the application programs of any changes so that traffic delays are minimized and also late comers may easily catch up with what's going on.

Users can collaborate at different levels. Sharing can occur at the image (visualization product) level, spray can level (abstract visualization objects – AVOs) or data stream level (e.g. files). At the image level, participants can see what the other participants see and may perhaps be able to change view points. At the can level, participants have access to a list of public spray cans put up by other participants. Using these public cans will generate AVOs from the remote hosts and distributed to other participants. Users may also give permission to other participants to have direct access to data streams and replicate those on local machines for faster response times.

In single mode spray rendering, users can create multiple cans but can control only one can at a time (limited by the number of input device – mouse). With multiple users and sharing of spray cans, it is possible that more than one user want to use a particular spray can. Floor control software regulates the use of spray cans.

Just as users can have local and public spray cans, they can also have local and public windows. Users work in their local window and may once in a while look at the public window to see what others are doing. The public window is also where one might do a broadcast as in briefing mode to show other users an item of interest.

Since participants are assumed to be geographically distributed, it may be difficult to get a point across, or try to get the attention of other participants by simply moving the cursor around. It is therefore necessary to include audio/video tools to help facilitate communication.

The different levels of collaboration also implies different requirements for compression. Tradeoffs will have to be made between graphics workstation capabilities, network bandwidth and compression levels. Things that need transmittal can either be images, AVOs (together with can parameters and other transformation matrices), or files.

6. Visualization Modeling

6.1 Princeton Ocean Model

As part of REINAS we had wished to be able to experiment with modeling from the user's point of view and to examine the broader question of interpolation and state-estimation in oceanography. However, developing an ocean model for the Monterey Bay from scratch would be a large effort, requiring more time and expertise than could possibly be justified for the project.

Fortunately at a meeting in October 1993 our colleagues at Fleet Numeric told us of the Princeton Ocean Model. It has the advantages of being small enough to run on a workstation, has good physical approximations, is easy to modify for boundary conditions, and is freely available. We obtained a copy of the code and documentation from NOAA/GFDL by file transfer (ftp gfdl.gov). Since then we have experimented with it in conjunction with the IBM Data Explorer and have inserted the Monterey Bay bathymetry as boundary conditions.

The Princeton Ocean Model, usually called the "Mellor Model", was originally created about 1977 by George L. Mellor and Alan F. Blumberg [BM87]. Since then it has been developed and applied to many oceanographic problems within the Atmospheric and Oceanic Sciences Program of Princeton University, NOAA's Geophysical Fluid Dynamics Laboratory, and Dynalysis of Princeton [EKM92].

The principal attributes of the model are:

1. It contains an imbedded sub-model to provide vertical mixing coefficients. This produces realistic bottom boundary layers which are important in coastal waters and in tidally-driven estuaries.
2. It uses sigma coordinates in which the vertical coordinate is scaled on the water column depth. This is important in dealing with the significant topographical variability such as that encountered in estuaries or over continental shelf breaks and slopes.
3. The horizontal finite difference scheme is staggered, usually called the "Arakawa C" differencing scheme. The model version we received uses curvilinear orthogonal coordinates, which can be a rectilinear or a spherical coordinate system as special cases.
4. The horizontal time differencing is explicit whereas the vertical differencing is implicit. The latter eliminates time constraints for the vertical coordinate and permits the use of fine vertical resolution in the surface and bottom boundary layers.
5. The model uses a free surface and a split time step. The external mode (i.e. surface) is two-dimensional and uses a short time step based on Courant, Friedrichs, Lewy stability conditions and the external wave speed. The internal mode is three-dimensional and uses a longer time step.
6. Complete thermodynamics have been implemented in the model.

6.2 IBM Data Explorer Visualization System

Late in 1993 we obtained a license for the commercial visualization program, the IBM AIX Visualization Data Explorer/6000 Version 1.2 [IBM93a]. It has proved very useful and powerful in working with output from models and other geographic data.

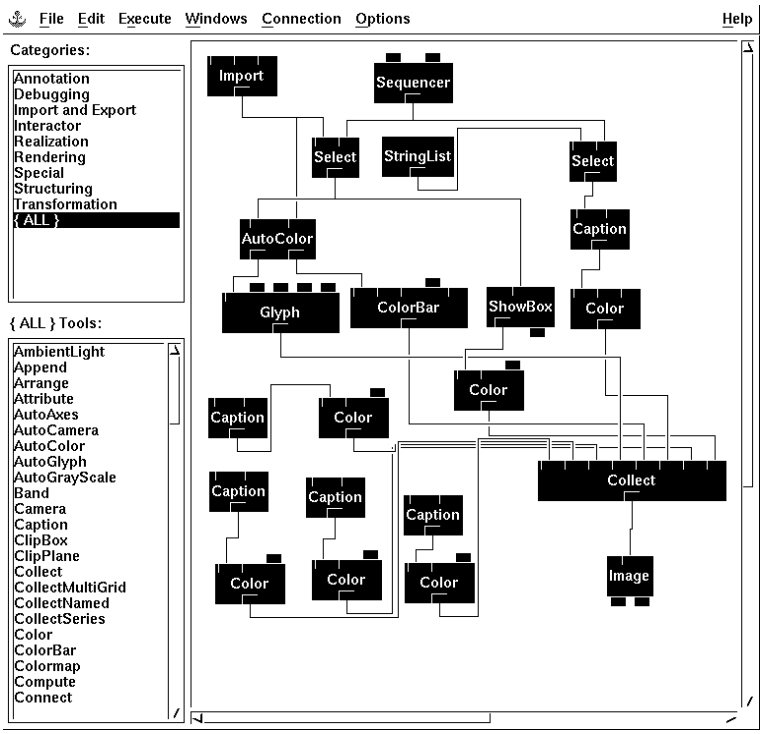

Figure 6.1: An Example of a Data Explorer visual program.

The Data Explorer (DX) enables a user to program using data flow networks. By visual programming a user creates the network on a "canvas" by placing boxes representing different program functions, and connecting them with lines representing the data flow. Data parameters are inherited through the network. The definition of a given box can be expanded to bring its hidden parameters into view for modification if desired. Figure 6.1 shows an example of a visual program.

The Data Explorer works on a client-server model where the user interface is independent of the execution of the visual program. The user interface runs on top of the X Window system and OSF/Motif(TM). It has five major window types [IBM93b]:

- The Visual Programming Editor provides the capability of defining and editing visual programs (including defining macros that may be used in other visual programs or macros).
- Control Panels which enable a user to control parameters within their visualization via Motif dials, sliders, steppers or widgets.
- The Sequencer provides the ability to easily create animations by providing an interface to control frame increments, step, loop and halt.
- The Image window where the visualization is displayed, provides functions to directly manipulate the image.

Data Explorer provides an extensive set of modules that can be used to visualize data. For example, the Isosurface, Streamline, and AutoColor modules perform the standard visualization functions of creating constant-value surfaces, tracing particle paths through velocity fields, and coloring objects based on a data value, respectively.

The Map module is a general purpose module that can map a data field onto an arbitrary object–whether it is a streamline, an isosurface, or even another data field's computational mesh.

The Compute module can perform pointwise arithmetic or trigonometric computation not only on the data, but also on the grid itself. This makes the task of warping a grid a simple matter of entering an expression. Even standard tools, such as Isosurface, operate on multiple types of input grids.

The Data Explorer renderer can handle opaque or translucent surfaces, translucent volumes, and opaque or translucent lines or points–all in the same image.

Because the data itself is self-describing, modules can be flexible in the types of data they accept, and can perform their actions appropriately based on their input.

Figure 6.2 shows an example of Monterey Bay bathymetry displayed on DX. Figure 6.3 is an example of buoy data which can be displayed in real time or from prerecorded tape.

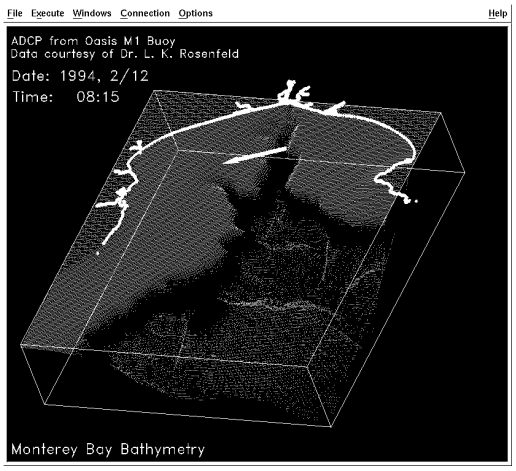

Figure 6.2: Example of Monterey Bay bathymetry displayed on Data Explorer.

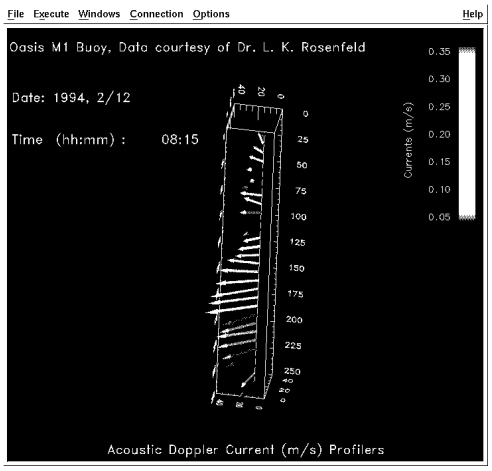

Figure 6.3: Buoy data displayed using direction vectors vs. depth.

The Cornell University Engineering and Theory Center maintains a public repository for IBM Visualization Data Explorer networks, modules, data, macros, etc. It can be reached via anonymous ftp at ftp.tc.cornell.edu. There is also a gopher interface at info.tc.cornell.edu (userid: info).

6.3 Atmospheric Model for Monterey Bay Region

A significant component of REINAS is the use of an atmospheric numerical model to provide a means of dynamically assimilating observations and well as providing numerical forecasts into the future. Real-time forecasters and retrospective researchers both require dynamically consistent depictions of the current state of the atmosphere. Such depictions are routinely made on the large scale motions in the atmosphere by numerous operational forecast centers (National Meteorological Center, Fleet Numerical Meteorological and Oceanographic Center, etc.). However, the production of such depictions on the scale of the Monterey Bay have never been achieved operationally. Real-time forecasters additionally require the ability to examine the predicted future evolution of the atmosphere.

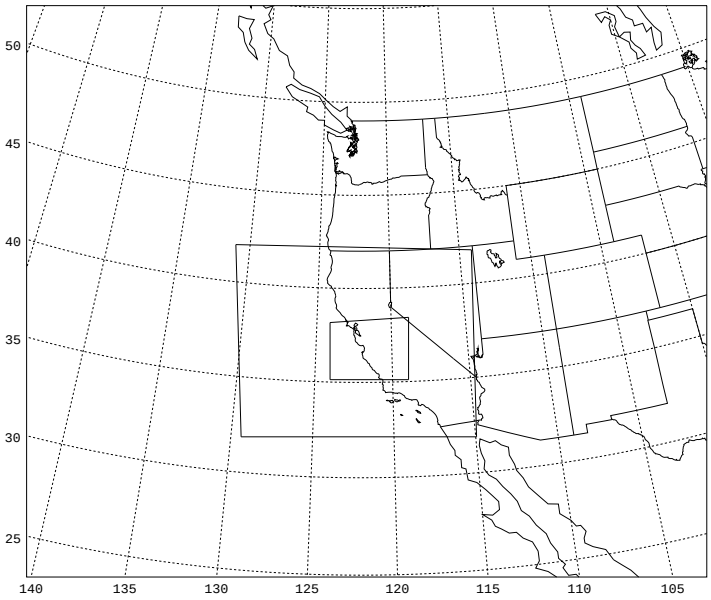

Figure 6.4: The Probable Domain for Numerical Modeling and the Individual Grids within it.

For REINAS, the evolution of the small-scale sea-breeze circulation in the Monterey Bay region is of interest and not available from any other source. Routine real-time simulations of the sea-breeze will be a significant component of the REINAS system.

To achieve the goals of data assimilation and real-time local scale numerical modeling, REINAS will use a version of the Navy Operational Regional Atmospheric Prediction System (NORAPS). This numerical forecast model is presently run by Fleet Numerical Meteorological and Oceanographic Center (FNMOC) and has been described by Hodur [Hod87] in 1987. The NORAPS model uses terrain-following sigma coordinates and can be run on three different map projections. For REINAS, the model will be run on a Lambert Conformal map projection centered on the Monterey Bay. The model domain can contain nested grids and present plans are to use a three nest pattern with the inner-most grid having a grid spacing of 3 km. The number of vertical levels and actual domain size for each nest will be determined based on the constraints of the NPS Cray YMP-EL computer on which the model will run. A sample of the probable domain and the individual grids within it are shown in Figure 6.4.

Topography for this domain is derived from a 30 second resolution topographic database with a vertical resolution of 8 m. The NORAPS model is a hydrostatic model that contains a 1.5 turbulence closure scheme for the boundary layer parameterization and Kuo [Kuo74] cumulus parameterization. The validity of the cumulus parameterization on 3 km grid spacing is highly questionable but the sea-breeze circulation should not require the use of this parameterization. Alternative cloud and precipitation parameterizations will be considered and added as needed once the system becomes operational in REINAS.

In order to do routine real-time numerical forecasts with NORAPS, initial conditions and lateral boundary conditions musbe supplied on a regular basis. For the REINAS system, the lateral boundary conditions on the largest domain will be supplied from forecasts produced by the Navy Operational Global Atmospheric Prediction System (NOGAPS) that are received at NPS in real-time. For the generation of the initial conditions, the data assimilation capabilities of NORAPS will be used. The operational data assimilation for NORAPS consists of a 6 hour update cycle where the obseravtions and a 6 hour forecast are blended together using optimal interpolation techniques. This data assimilation cycle will be modified for REINAS to run in a continuous assimilation mode in order to make use of the more frequent, high-resolution observations being collected by the REINAS system. Several significant problems must be addressed in developing a continuous data assimilation system for REINAS. First, the optimal interpolation technique used operationally in NORAPS is based on statistical functions derived from large-scale atmospheric observations. Either these functions must be replaced with something adequate for smaller-scales or a different interpolation technique must be used.

The present plan is to utilize the multiquadric interpolation technique that has previously been described by Nuss and Titley [NT94]. Research is presently underway to incorporate this interpolation method into the model. The other major problem is that periodic insertion of noisy observations can excite substantial noise in the model. This is particularly a problem when a fixed update cycle is used at intervals less than 6 hours. Plans are to incorporate the multiquadric interpolation into the model integration to achieve an ongoing dynamic balance based on the model equations.

Substantial work remains in coupling the NORAPS model to the REINAS data base using the application program interface (API). Both the extraction of REINAS observations for use by the model as well as the insertion of model analyses and forecasts for subsequent visualization need to be addressed. The temporal frequency of the insertion of observations into the data assimilation system as well as the output of model forecast fields must be defined based upon the capabilities of other parts of the REINAS system.

7. Data Compression in REINAS

7.1 The Need for Data Compression

Data compression is just one resource managed within the REINAS system to achieve higher efficiency. It is most useful when it is under the covers of the system so that the average user does not have to be concerned with its details. It fits between all the major components of REINAS – instruments, computer network, data management, visualization, and simulation.

Data compression is being investigated for use in REINAS to solve several important problems:

1. Data archiving requires keeping data arriving from a large number of sensors, users, and imported processes.
2. Network bandwidths available to a rapidly increasing number of users will be limited.
3. Exporting of REINAS products requires careful resource usage so that the widest number of users will be able to effectively use the data and products generated by REINAS.

Compression helps to fulfill these three goals as efficiently as possible. Because compression is to be used for a variety of reasons there are multiple compression techniques that have been and are being investigated for REINAS.

7.2 Compression for Data Archives

Compression has been investigated to help reduce the amount of storage necessary to archive data. Two weeks of data are small enough, (100 Mbytes to 200 Mbytes), to be archived without compression. See Table 1.1. This data is generated from current instruments such as Met stations, Profilers, ADCP's, CODAR's, AVHRR's, and GEOS. Data older than two weeks may be compressed to secondary storage, and the primary storage reused for incoming data.

This archival storage compression is achieved by using Two-minute data averages from the instruments. This granularity is sufficient by consensus of the interested meteorologists. Once the data are acquired, averaged, and rounded to the precision specified by the meteorologists, lossless waveforms compression is applied. This technique, developed for gray scale coding of satellite images under a NASA Summer Faculty Fellowship [LAML94], can be a common part to any compression techique. Predictive coding produces the prediction error, and a quantization of the previous error is the context. The binary arithmetic coding for encoding context-dependent prediction errors, described in [LM93], is intended for the compression of satellite images. Thus, coding approaches applied for archiving the met data are applicable to most one-dimensional digitized waveforms, as well as to two-dimensional waveforms. Moreover, nothing in the approach precludes its future employment at the met stations themselves.

7.3 Data Compression for Scientific Visualization

REINAS data will be used by many simultaneous users. Because REINAS is using the Internet, and shared NPS, UCSC networks, the number of users will be limited by the bandwidth. Sufficient communication bandwidth is needed. Visualization users get their data over a communication link, and compression at the source offers bandwidth savings. Moreover, if the screen is broadcast to several collaborating users, the bandwidth savings are multiplied. Each user's workstation will have a "decompress and display" capability.

The resource allocation of decompression and compression requires sending the type of data that the user can manage. (See [Cho92, Cho91] for discussion of resource allocation in multimedia). If the viewing station is attached to an overloaded host computer, or has a less powerful processor, then quick decompression is used. One of the simplest and fastest compression algorithms for images, in terms of the decoder, is Vector Quantization (VQ). Macy [Mac93] investigated VQ and modified an algorithm for experimental use in REINAS. Adaptive compression techniques may also be investigated [GYM93].

If the viewing station has a more powerful processor able to perform inverse Discrete Cosine Transform calculations, then JPEG, an emerging ISO/ITU standard, is a possible choice. Available free software and hardware accelerator cards make JPEG compression a viable choice. We have obtained source code and experimented with it. On going research is being done to compare the characteristics of JPEG with VQ to determine advantages and disadvantages.

If the viewing station has the power of a graphics rendering engine, then a third opportunity for possibly even greater compression exists. The visualization objects are compressed and transmitted, and the viewing station renders them locally. Rendering power allows user freedom to change the viewpoint independent of other users, and subsequent visualization objects can be incrementally added to the list of objects to render. The amount of data transmitted over the collaborative session is greatly reduced. For example, a user is sent the higher-level graphics language commands and data to render If the next screen is a slightly rotated view of the same data the VQ or JPEG compression schemes require the compression and decompression of a completely new image. Using the high level commands all that is transmitted is the new rotated view point.

7.4 Data Compression on the Network

We are investigating a method of first compressing the data set and broadcasting to all workstations. Then compressing the higher-level commands necessary to render the data which requires much less bandwidth to transmit. The ability to reduce the size of the data set helps [SZL92, Tur92, FS93]. It may also be possible to render the compressed data sets directly without decompressing them [NH93, Cho92]. By using the workstation capability as the switch by which to choose appropriate methods, many compression approaches can be used within REINAS. The important development we are investigating is expressive graphics language constructs that allow the varying resources to be taken into account on a user by user basis.

8. REINAS Applications in Environmental Science

The period from May 1994 to September 1994 provides a test period for the prototype REINAS system. During this period, numerous meteorological observations will be collected and loaded into the REINAS system in a routine manner. This test period provides a system shakedown period but more importantly provides the opportunity to begin meteorological research using the REINAS system. This research using the REINAS system will provide important refinements to REINAS to give it wide appeal to the environmental science users. Some of these refinements and outstanding issues not addressed in the current system design are described below.

To begin to refine the basic REINAS system from a user perspective, the basic observational data described in other parts of this document must be resident in REINAS. Initially the timeliness of the capture of these data will not be an issue for scientific users as real-time use is not envisioned for science experiments during this test period. (Real time use of REINAS by operations e.g. MBARI's ROV, is expected). More significant to scientists is the ability of users to begin to work with REINAS to examine the variety of data collected during this period.

In addition to the observations described in this report, several concurrent meteorological research efforts will take place during this period. These additional observations will in most cases not be available in real-time but can hopefully be loaded into the REINAS database at a later time. These other research programs are both Office of Naval Research (ONR) sponsored programs. The first program is an effort to study the impact of ships on the marine boundary layer structure and the formation of cloud lines in the ship wake. This program is referred to as the Monterey Area Ship Tracks (MAST) experiment and will primarily employ research aircraft and ships in the Central California Coast region. The second program is an effort to study the dynamics of coastally trapped disturbances along the California coast. This program will employ wind profilers, surface meteorological stations and a small aircraft. These observing platforms will primarily located along the Central California coast.

Refinements to REINAS should occur in three primary areas. The first area of research and refinement should be in developing the REINAS forecast/real-time standard products for future real-time users. Future real-time operational users include the Monterey Bay and San Francisco Bay Air Pollution Control Districts, who have expressed interest the system during demonstrations to them; the National Weather Service (NWS) Monterey Forecast Office; and others that may be added in the future. These users will require easy to use visualization products tailored to their specific application that need to be developed from the basic tools of REINAS.

The next area of active refinement should be to the atmospheric modeling and the data assimilation of the REINAS observations. The high quality mesoscale data set to be taken during the summer of 1994 is required to begin to develop new techniques for assimilating observations on the scale of the Monterey Bay sea-breeze. The third area of refinement should be in the area of station enhancements to fill in holes in the observing network. Research using the observations collected during the summer of 1994 will clearly define critical areas where additional observations are needed to more completely understand the sea breeze interaction with the local topography.

These refinements are not an exhaustive list but are essential to making REINAS a viable retrospective and real-time research tool for a wide range of scientific users and operational forecasters. æ

References

[BEW77] D.E. Barrick, M.W. Evans, and B.L. Weber. Ocean surface currents mapped by radar. *Science*, 198:138–144, 1977.

[BLC85] D.E. Barrick, B.J. Lipa, and R.D. Crissman. Mapping surface currents with codar. *Sea Technology*, October 1985.

[BM87] A. F. Blumberg and G. L. Mellor. A description of a three-dimensional coastal ocean circulation model. In *Three-Dimensional Coastal Ocean Models*, volume 4, page 208. American Geophysical Union, Wash. DC, 1987.

[Bos92] L. Bosack. Method and apparatus for routing communications among computer networks. *U.S. Patent assigned to Cisco Systems, Inc.*, 1992.

[CFL79] I. Chlamtac, W.R. Franta, and K.D. Levin. Bram: The broadcast recognizing access mode. *IEEE Trans. Comm.*, 27(8):1183–1189, 1979.

[Chi91] J.N. Chiappa. A new ip routing and addressing architecture. *Unpublished Draft*, 1991.

[CHMP93] Judith Baynard Cushing, David Hansen, David Maier, and Calton Pu. Connecting scientific programs and data using object databases. *Bulletin of the Technical Committee on Data Engineering*, 16(1):9–13, March 1993.

[Cho91] C. E. Chow. Resourse allocation for multimedia multiparty connections in broadband networks. Report eas-cs-91-6, Univ of Colorado, Boulder, CO, October 1991.

[Cho92] C. E. Chow. Resourse allocation algorithms for multimedia multiparty connections. Report eas-cs-92-1, Univ of Colorado, Boulder, CO, January 1992.

[Chu93] J.H. Churnside. Delta k lidar sensing of surface waves in a wave tank. *Applied Optics*, 32:339–342, 1993.

[Col89] R. Coltun. Ospf: An internet routing protocol. *ConneXions*, 3(8):19–25, 1989.

[Cro55] D.D. Crombie. Doppler spectrum of sea echo at 13.56 mc/s. *Nature*, 175:681–682, 1955.

[EKM92] T. Ezer, D. S. Ko, and G. L. Mellor. Modeling and forecasting the gulf stream. *Marine Tech. Soc. Journal*, 26(2):5–14, 1992.

[ERH92] D. Estrin, Y. Rekhter, and S. Hotz. Scalable inter-doman routing architecture. *Computer Comm. Review*, 22(4), 1992.

[Fer93] D.M. Fernandez. High-frequency radar measurements of coastal ocean surface currents. Phd dissertation, Stanford University, Stanford, CA, August 1993.

[FS93] Thomas A. Funkhouser and Carlo H. Sequin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *SIGRAPH 93*, pages 247–254, 1993.

[GB92a] W. Griethe and J. Burfeindt. CIS communication facilities, data transmission and computer-aided experiment evaluation for users of the MIR space station. *ESA Journal*, 16(4):455–462, 1992.

[GB92b] Bruce R. Gritton and C. Baxter. Video database systems in the marine sciences. *MTS Journal*, 26(4):59–72, 1992.

[GBD+89] Bruce R. Gritton, Dushan Badal, Dan Davis, K. Lashkari, G. Morris, A. Pearce, and H. Wright. Data management at MBARI. In *Oceans 89 Proceedings: The Global Ocean*, pages 1681–1685. IEEE Publications, 1989.

[GLA88] J.J. Garcia-Luna-Aceves. Routing management in very large-scale networks. *Future Generation Computing Systems*, 4(2):81–93, 1988.

[GLA93] J.J. Garcia-Luna-Aceves. Loop-free routing using diffusing computations. *IEEE/ACM Transactions on Networking*, 1(1):130–141, 1993.

[GLAZ92] J.J. Garcia-Luna-Aceves and W.T. Zaumen. Protocol analysis and standard protocol suite selection for disn. SRI SETA Task Briefing to DISA, April 1992.

[GLAZ93] J.J. Garcia-Luna-Aceves and W.T. Zaumen. Shortest multipath routing using diffusing computations. SRI Invention Disclosure, 1993.

[Goo90] D. Goodman. Cellular packet communications. *IEEE Trans. Comm.*, 38(8):1272–1280, 1990.

[GSS87] P.V. Grigor'ev, T.B. Shevchenko, and I.V. Shugan. Investigation of thestatistical properties of a moving sea surface by a laser radar. *Soviet Physics - Lebedev Institute Reports*, 05:42–46, 1987.

[GYM93] Brian K. Guenter, Hee Cheol Yun, and Ressell M. Mersereau. Motion compensated compression of computer animation frames. In *SIGRAPH 93*, pages 297–304, 1993.

[Ha93] E.C. Ha. High-frequency radar measurements of ocean currents and current shears. Phd dissertation, Stanford University, Stanford, CA, June 1993.

[Hed88] C. Hedrick. Routing information protocol. RFC 1058, Stanford Research Institute, SRI International, Menlo Park, CA, June 1988.

[HKM+88] John H. Howard, Michael L. Kazar, Sherri G. Menees, David A. Nichols, M. Satyanarayanan, Robert N. Sidebotham, and Michael J. West. Scale and performance in a distributed file system. *ACM Transactions on Computer Systems*, 6(1):51–81, February 1988.

[Hod87] R.M. Hodur. Evaluation of a regional model with an update cycle. *Monthly Weather Review*, 115:2707–2718, 1987.

[HPbC93] A.R. Hurson, Simin H. Pakzad, and Jia bing Cheng. Object-oriented database management systems: Evolution and performance issues. *IEEE Computer*, February 1993.

[IBM93a] IBM Corp., Armonk, New York. *IBM AIX Visualization Data Explorer/6000 (ver 1.2)*, 1993.

[IBM93b] IBM Corp., Armonk, New York. *IBM POWER Visualization User's Guide*, 1993.

[IL92] Y. Ioannidis and M. Livny. Conceptual schemas: Multi-faceted tools for desktop scientific experiment management. *International Journal of Intelligent & Cooperative Information Systems*, 1(3-4):451–474, December 1992.

[ILS+90] J.M. Intrieri, C.G. Little, W.J. Shaw, R.M. Banta, P.A. Durkee, and R.M. Hardesty. The land/sea breeze experiment (LASBEX). *Bulletin of the American Meteorological Society*, 71(5):656–664, 1990.

[ISO89] ISO. Intra-domain is-is routing protocol. *ISO/IEC*, JCT1/SC6(WG2 N323), September 1989.

[ISO91] ISO. Protocol for exchange of inter-domain roueing information among intermediate systems to support forwarding of iso 8473 pdus. Technical report, International Standards Organization, 1991.

[Jaf84] J.M. Jaffe. Algorithms for finding paths with multiple constraints. *Networks*, 14:95–116, 1984.

[KS80] L. Kleinrock and M. Scholl. Packet switching in radio channels: New conflict-free multiple access schemes. *IEEE Trans. Commun.*, 28(7):1015–1029, 1980.

[Kuo74] H.L. Kuo. Further studies of the parameterization of the influence of cumulus convection on the large-scale flow. *J. Atmos. Sci.*, 31:1232–1240, 1974.

[LAML94] Glen G. Langdon, Robert J. Antonucci, William W. Macy, and Dean Long. On compression of data from a meteorological stations. In *Data Compression Conference, IEEE Computer Society*, Snowbird UT, March 1994. Poster Session.

[LB83] B.J. Lipa and D.E. Barrick. Least-squares method for the extraction of surface currents from codar crossed-loop data: Applications at arsloe. *IEEE Journal of Ocean Engineering*, OE-8:226–253, 1983.

[LM93] Glen G. Langdon and Mareboyana Manohar. Centering of context-dependent components of prediction error distributions. In *Proc SPIE: Applications of Digital Image Processing XVI*, volume 2028, pages 26–31, San Diego, CA, July 1993.

[LR91] K. Lougheed and Y. Rekhter. Border gateway protocol 3 (bgp-3). RFC 1267, Stanford Research Institute, SRI International, Menlo Park, October 1991.

[Mac93] William W. Macy. Variable rate vector quantization for image compression. Report ucsc-crl-xx, Univ of California, Santa Cruz, Santa Cruz, CA 95064, December 1993.

[MGLA87] J. Mathis and J.J. Garcia-Luna-Aceves. Survivable multiband networking. In *IEEE MILCOM '87*, Washington, DC, October 1987.

[MLP+93a] P.E. Mantey, D.D.E. Long, A.T. Pang, H.G. Kolsky, et al. Reinas: Phase one - concept statement. Report ucsc-crl-93-05, Univ of California, Santa Cruz, Santa Cruz, CA 95064, January 1993.

[MLP+93b] P.E. Mantey, D.D.E. Long, A.T. Pang, H.G. Kolsky, et al. Reinas: Phase two - requirements definition. Report ucsc-crl-93-34, Univ of California, Santa Cruz, Santa Cruz, CA 95064, July 1993.

[MS94] Henry H. Mashburn and Mahadev Satyanarayanan. RVM: Recoverable Virtual Memory. Technical report, Carnegie Mellon University, 1994.

[Nea92] T.C. Neal. Analysis of monterey bay codar-derived surface currents march to may 1992. Technical Report M.S. Thesis, Naval Postgraduate School, 1992.

[NH93] P. Ning and L. Hesselink. Fast volume rendering of compressed data. In *Visualization 93*, pages 11–18, San Jose, CA, October 1993.

[NT94] W.A. Nuss and D.W. Titley. Use of multiquadric interpolation for meteorological objective analysis. *Monthly Weather Review*, 122:(to appear), 1994.

[OIB91] L.D. Olivier, J.M. Intrieri, and R.M. Banta. Doppler lidar observations of a land/sea breeze transition on a day with offshore flow. In *Fifth Conference on the Meteorology and Oceanography of the Coastal Zone*, pages 130–142, May 1991.

[PA94] A. Pang and N. Alper. Mix and match: A construction kit for visualization. page (submitted), 1994.

[PAFW93] A. Pang, Naim Alper, Jeff Furman, and Jiahua Wang. Design issues of spray rendering. In *Compugraphics'93*, pages 58–67, 1993.

[PS93] A. Pang and K. Smith. Spray rendering: Visualization with smart particles. In *Visualization'93 Proceedings*, pages 283–290, 1993.

[Rom86] R. Rom. Collision detection in radio channels. *Local Area and Multiple Access Networks (R. Pickholtz, Ed.)*, 1986. Chapter 12.

[RS90] R. Rom and M. Sidi. Multiple access protocols: Performance analysis. *Springer-Verlag New York*, 1990.

[SD91] Michael Stonebraker and Jeff Dozier. Large capacity object servers to support global change research. Technical Report 91-1, SEQUOIA 2000, July 1991.

[Sha86] Marc Shapiro. Structure and encapsulation in distributed systems: the proxy principle. In *Proceedings of the 6th International Conference on Distributed Computing System*, pages 198–204. IEEE Computer Society Press, May 1986.

[SMK+93] M. Satyanarayanan, Henry H. Mashburn, Puneet Kumar, David C. Steere, and James J. Kistler. Lightweight recoverable virtual memory. *Proceedings of the 14th ACM Symposium on Operating Systems Principles*, 5:146–160, December 1993.

[SNS88] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An authentication service for open network systems. In *USENIX Conference Proceedings*, pages 191–202. USENIX, 1988.

[SSAA93] Terence R. Smith, Jianwen Su, Divyakant Agrawal, and Amr El Abbadi. Database and modeling systems for the earth sciences. *Bulletin of the Technical Committee on Data Engineering*, 16(1):33–37, March 1993.

[Sta93] Jeffrey P. Stamen. Structuring databases for analysis. *IEEE Spectrum*, pages 55–58, October 1993.

[Ste92] M. Steenstrup. Inter-domain policy routing protocol specification: Version 1. Internet Draft, May 1992.

[Sto92] Michael Stonebraker. An overview of the SEQUOIA 2000 project. In *Digest of Papers: COMPCON Spring 1992, 37th IEEE Computer Society International Conference*, pages 383–388. IEEE Computer Society Press, February 1992.

[Sto93] Michael Stonebraker. The Miro DBMS. *SIGMOD Record*, 22(2):439, 1993.

[SZL92] William J. Schroeder, Jonathan A. Zarge, and William Lorensen. Decimation of triangle meshes. In *SIGGRAPH 92*, 1992.

[Tea86] C.C. Teague. Multifrequency hf radar observations of currents and current shears. *IEEE Journal of Oceanic Engineering*, OE-11 no.2:250–269, 1986.

[Tur92] Greg Turk. Retiling of polygonal surfaces. In *SIGRAPH 92*, pages 55–64, 1992.

[WAW+92] S.N. Walker, H.St.C. Alleyne, L.J.C. Wooliscroft, D.L. Giaretta, M.A. Hapgood, D.R. Lepine, B.J. Read, M.A. Albrecht, A. Ciarlo, H. Stokke, and P.Torrente. A space-physics query language for the european space information system. *ESA Journal*, 16(4):463–476, 1992.

[ZGLA92] W. Zaumen and J.J. Garcia-Luna-Aceves. Dynamics of link-state and loop-free distance-vector routing algorithms. *Journal of Internetworking*, December 1992.

Index