REINAS: A Real-time System for Managing Environmental Data*

Eric C. Rosen, Theodore R. Haining, Darrell D. E. Long, Patrick E. Mantey
Baskin Center for Computer Engineering & Computer Science
University of California, Santa Cruz, CA 95064, USA

Craig M. Wittenbrink
Hewlett Packard Laboratories
Palo Alto, CA, USA

Journal of Software Engineering and Knowledge Engineering, vol. 8, no. 1 (1998), pp. 35–53.

Abstract

Managing scientific data is a challenging task, and many of the problems it presents have yet to be adequately solved. The Real-time Environmental Information Network and Analysis System (REINAS) is an operational solution to the problem of collecting and distributing environmental data in a real-time context, as well as supporting data acquisition, retrieval, visualization and long-term data maintenance. The system is built around one or more databases and has been developed to support both real-time and retrospective regional scale environmental science. Continuous real-time data is acquired from dispersed sensors and input to a logically integrated but physically distributed system. The database engine provides a powerful structure to handle data management, but current database technology can have difficulty meeting the performance requirements that a large real-time environmental system demands. The REINAS architecture and current status is described in detail, including the challenges that were addressed in the construction of an operational system which includes a regional wireless instrumentation network comprised of over 200 instrument platforms and 1500 sensor streams producing real-time data of interest to thousands of users.

1 Introduction

The Real-Time Environmental Information Network and Analysis System (REINAS) (described at earlier stages of development in [11, 10, 13]) supports regional-scale environmental science, monitoring, and forecasting through a distributed system connected by the Internet. It is being developed by the University of California at Santa Cruz (UCSC), in conjunction with the Naval Postgraduate School (NPS), and the Monterey Bay Aquarium Research Institute (MBARI). With REINAS, environmental scientists have the ability to observe, monitor, and analyze regional oceanographic and meteorological phenomena from their desk top. It has been designed to provide:

- a flexible set of tools to configure and collect data from instruments in the field in real time,
- an integrated problem solving and visualization system supporting individual and collaborative research using both historical and modeled data, and
- a logically consistent distributed database that stores data independently of file formats, providing a generic instrument independent view of the data streams while maintaining metadata describing where and how data was obtained (i.e. the database tracks data pedigree).

REINAS serves different user groups: *operational forecasters* who monitor current conditions, view standard data products, synthesize new data views, and issue forecasts and warnings; *modelers* who analyze new model

*Supported by the Office of Naval Research under grant N00014-92-J-1807 and by the National Science Foundation under grant IRI-9423881. An earlier version of this paper appears in *Proceedings of 1996 Conference on Software Engineering and Knowledge Engineering*, June 1996, pages 293-300.

products and compare them with other models and with past and present conditions; *experimental scientists* who collaborate with other scientists on-line, observe individual data fields as they are collected, and who may modify data collection methods for an on-going experiment; as well as *instrument engineers* who add new equipment to the system, access metadata describing individual devices and methods of calibration, study maintenance records, and profile sensor quality. In addition, the *general public* (especially the local community of sailors and wind surfers) form an ever-growing group that has come to depend on real-time access to REINAS environmental data via the World Wide Web [17, 18, 26].

Interactive real-time measurement, real-time monitoring, and retrospective data management using REINAS provide forecasters and experimental scientists with a framework to plan their experiments, expeditions, and monitoring activities. The special focus on real-time observation and analysis allows the oceanographic and meteorological communities to identify phenomena as they occur and to react to emerging phenomena and trends by changing the nature and frequency of data collection at sites of interest.

The integrated problem solving and visualization environment provides scientists and forecasters with pictures of environmental features, trends, and relationships to study dynamic environmental behavior in a geographic context. Scientists can fuse data collected in the field with data generated by numerical models and simulations.

System configuration tools simplify instrument engineering tasks, for instance, supporting easy addition of new instruments to the system. Instruments are connected to REINAS by both radio and terrestrial links of various types and data rates.

The database supports access to data, collected by different people and institutions at different times, in a logically consistent manner using a schema that is accessible to (but easily hidden) from the majority of users. The database separates users from many of the mechanical tasks of data management, as it is designed around an architecture integrating real-time data from multiple instrument technologies, classes (numeric, text, images and video), and representations (scalars, vectors, strings, etc). Both scientific data and metadata are stored using a stable schema which maintains data pedigree. An interactive electronic log book allows experimental scientists to tag measurement data with additional arbitrary information.

The remainder of this discussion is organized as follows: A brief review of systems similar in some sense to REINAS is provided, followed by a detailed description of REINAS itself, organized by its major components: the database, instrumentation, and network subsystems. After describing the system architecture, the REINAS real-time concept and associated hardware and software challenges are discussed and subsequently followed by a summary of the structure, evolution, and performance features of the various information models employed by REINAS during its development. Finally, the challenges surrounding the definition and implementation of the system's deliverables, including access methods, interfaces, and ease of use, and other issues, are described before the key lessons and conclusions that the development of REINAS has fostered are presented.

2 Related Work

Several systems exist which are similar in some sense to REINAS and deserve mention in a review of prior efforts in environmental data management. Sequoia 2000 was a large multi-year development project which focused on global environmental science [22, 23]. Sequoia 2000 differs significantly from REINAS in that while REINAS is driven by real-time constraints, Sequoia 2000 was primarily designed for retrospective access with little real-time emphasis. Like REINAS, Sequoia 2000 was built around a relational database [19, 21]. Real-time systems employing less sophisticated database technologies have also been developed [4, 7, 20], as well as others that run at much lower data rates [6]. One such system, StormCast, collects and archives environmental data primarily for current and forecast use, as well as supporting operating system technologies research [8]. Unlike REINAS, StormCast does not address the long-term data management problem, the issues involved in overlaying different types of structured scientific data, or the advantages of treating instruments as active Internet nodes instead of passive data-streams. The NEONS project [16] did employ a relational database engine for environmental data, but unlike REINAS, NEONS did not emphasize real-time access to instrumentation. Some previous Geographic Information Systems employ structures similar to those used by the REINAS database schema [2, 3], while some recent data visualization work [9] is similar to that developed within REINAS. REINAS differs from these previous works in that it stresses very short time-scale real-time access to environmental data through a robust and highly structured database system, supporting regional three-dimensional visualization products as well as custom user-level applications through a generalized query interface.

3 The System

REINAS is a large, distributed, system, made up of hundreds of components scattered across a geographic area of several hundred square miles. The architecture of its various subsystems has evolved over the life of the project into three major pieces: database nodes, instrument nodes, and the connecting Internet Protocol (IP) based network. Figure 1 depicts the flow of measurement data from (potentially remote) instrument nodes on the left, into the system components of a database node, through which client applications may query for real-time or retrospective measurements. Most of these components may be distributed across a collection of networked hosts. In the following sections, the components of the system are described in more detail.

3.1 Database nodes

REINAS includes one or more database engines, into which real-time environmental data flows from dispersed instrument nodes. When data arrives at a database node, it is received by a *stream-server*, which immediately logs the data to disk. A separate *loader* reads from this *stream* log and formally interprets the measurement data and inserts it as appropriate into the database; shared access to the log is coordinated by a robust transaction *log-manager*, which allows data to reliably migrate through the system. Requests for data from client applications are received by the *dispatcher*, which generates Standard Query Language (SQL) queries from a set of "canned" query templates. The dispatcher submits these to the database engine, returning the results in a standard form. In addition, the dispatcher can also act as a transparent mechanism for submitting SQL queries directly to the database.

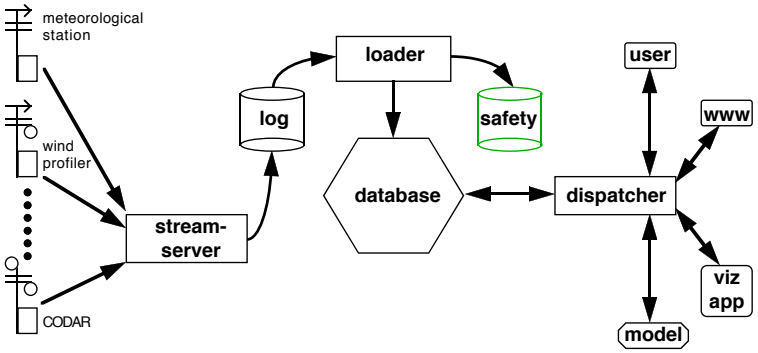

Figure 1: REINAS system data flow, database node view.

The structure in Figure 1 reflects several operational and technological constraints. For example, the input data-rate may exceed the ability of the database to accept new data at times. In such cases, the log provides a reliable buffer to receive data from REINAS instruments, allowing the system to continue operation and eventually migrate all data into the database. The presence of this log also allows the database engine to remain offline for extended periods (for backups or upgrades) while the stream-server continues to receive and store data in its stream log until the database is restored. Corrupt measurements or others that cannot be inserted into the database may be optionally logged to a separate *safety* log which can be interactively processed at a later time. The safety log also allows the system to continue loading real-time measurements even when portions of the database's tablespace may be full or require maintenance. Because the input and output paths of the data in the database are separate, queries to the system can continue to be handled through the dispatcher when logging of measurements to the stream log or loading of new data into the database must be suspended.

3.2 Instrument nodes

Two of the most unique aspects of REINAS are the emphasis it places on real-time data collection, and that the system was conceived and built to be tightly integrated with the Internet and its standard protocols. As a result, instruments within REINAS are accessed as Internet nodes, and measurement data migrates from these remote nodes to a database node of the form presented in Figure 1 using IP and its associated transport layer protocols. In practice, this often means deploying a computer near the remote instrument and networking the computer to the Internet via wireless or land links, as appropriate.

Raw environmental data typically arrives in digital form through a serial or parallel port. The data stream associated with this hardware port is managed by a single process, the *distributor*. The distributor performs all instrument-specific interactions to acquire and parse the data stream, and allows multiple client processes to gain shared or exclusive access (where appropriate) to the data stream. It also implements access control for the configuration and control of the instrument. To promote software portability, the distributor provides a separate standard interface to processes which perform actions on the instrument data itself, such as forwarding the data to a REINAS database node.

Since instrument are typically at the remote end of a slow and unreliable network link, a *logger* logs the stream to disk and a separate *reader* transmits each data record to a stream-server at the database node. To prevent the loss of data, the record from the *instrument log* is deleted only after the stream-server has acknowledged receipt.

Figure 2 illustrates this software architecture applied to the case of a generic meteorological station. A single

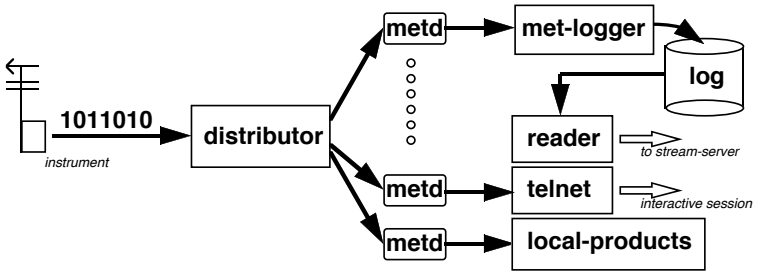

Figure 2: Instrument node data flow.

distributor tuned to a specific flavor of meteorological instrument, such as those provided by a particular vendor, manages the stream. Multiple instantiations of the *metd* daemon interact with the distributor and provide a generic meteorological station interface. The *met-logger* logs the stream, while a *local-products* process maintains a raw local archive to support legacy software. Scientists, system-administrators, and even specialized applications may also interact with the measurement stream or instrument itself in parallel with data collection.

3.3 Network

The instrument and database nodes, as well as users running visualization applications, are linked via a logical network connected by the Internet. In some cases, connections are made over a physical network of conventional wire links (e.g. Ethernets, T-1 bridges); elsewhere connections are made using wireless (radio frequency) links of varying bandwidth. The characteristics and performance of these networks are a crucial aspect of the system as a whole, especially in the case of wireless networks.

To achieve satisfactory reliability and performance tuning over wireless links, it has been necessary to implement new wireless-aware protocols. Because wireless links are both lossy and exhibit high latency, congestion based error-correction protocols such as TCP are not appropriate for transmitting real-time data from remote instruments to database nodes [1, 15]. However, since reliable transport must still be guaranteed to insure that *all* data eventually migrates from remote instrument logs into a stream log, some form of retransmission protocol is necessary. REINAS has adopted a simple User Datagram Protocol (UDP) based retransmission scheme utilizing explicit acknowledgements that relies on linear back-off combined with aggressive recovery to deal with lossy network links between instrument and database nodes.

4 The Real-Time Problem

The most challenging aspect of REINAS is its real-time focus. The real-time performance requirements are determined by the time-scale of the environmental phenomena that REINAS monitors. This focus not only provides additional value to environmental researchers who may initially view REINAS simply as a convenient location to store and retrieve data, but it also distinguishes REINAS from other environmental database systems and produces several key performance challenges that must be carefully addressed during design and implementation.

4.1 What does real-time mean?

A real-time system means different things in different contexts. In general, a sense of real-time interaction is often determined by the application interface and the type of feedback provided. A visualization application that allows the user to manipulate static data interactively might be considered real-time by some, even though the data itself may not change. Other users may define a system to be functioning in real-time only when they can perceive that the data is being updated frequently. Still others might define real-time to mean that data is guaranteed to arrive with certain frequency and latency guarantees.

Under REINAS, real-time is defined by the characteristics of the environmental phenomena being observed. For some sensors attached to surface weather stations (such as those directly measuring wind gusts), sub-minute sampling rates are appropriate and real-time in this context means updated values arriving every 10 seconds. For others sensors, such as air-temperature, 60 second sampling periods are appropriate. Radars measuring slower changing phenomena such as ocean currents tend to require longer term averaging, and real-time in this context may infer updates occurring at 30 minute or one hour intervals. However, because REINAS uses a best effort network to deliver these readings, it does not strictly guarantee when data will arrive and is therefore not real-time in a strict systems sense.

4.2 Real-time challenges

Typical real-time or interactive database systems are designed for applications that heavily favor read operations over writes. Examples include airline reservation systems, automatic teller machine interactions, and point-of-sale credit verification systems.

In REINAS, as much as 90 percent of the database activity is devoted to writing new data into the database. Separate inserts from several hundred distinct time-series sensors are independently issued every few seconds. Other more structured data arrive for insertion at slightly less frequent intervals (from every minute to every few hours). User-level regional-visualization applications generate the heaviest demands for reads, but the maximum level of retrievals makes up a relatively small percentage of overall database workload. In a real-time environmental database system such as REINAS, updates of *existing* data are rarely performed. Hence, system performance is constrained primarily by the capacity of the database to incorporate a heavy, steady stream of inserts.

Because current commercial database products cannot always keep pace with the high rate of inserts that REINAS must support, inserts migrate through a disk-based transaction log where they can be buffered and grouped into more efficient bulk loads. As noted earlier, each log is produced by a stream-server and consumed by a loader. If necessary, multiple logs processing parallel insert streams can be maintained. The REINAS database engine, built using high performance hardware such as a Silicon Graphics Challenge L server, is capable of maintaining a measurement data insert rate exceeding 45 inserts per second. This datarate is sufficient for the system's current average datarate, but can be easily overwhelmed during periods of heavy insertions when externally collected measurements or data from temporarily isolated instrument nodes are fed to the system.

The rapid arrival rate of data packets from several instrument nodes to a single database node also presents problems for the stream server and loader. As these components both rely on the log manager to write or read measurement data to or from non-volatile storage, the latency of this storage medium is a performance bottleneck in the path of data into the database. To alleviate this problem, multiple stream-servers, loaders, and logs can be maintained. Measurement data can also be packaged at the instrument node in larger, aggregate packets which help reduce the number of separate log transactions at a given datarate. Ultimately, the log can be moved to lower latency storage such as a RAM disk or other non-volatile memory. At present, REINAS has successfully pursued the latter two strategies and achieved log datarates exceeding 64 kbps during bulk measurement transmissions into the system. Other tests indicate the REINAS log manager is capable of handling approximately 100 separate log transaction per second, a rate which significantly exceeds that currently seen during normal operational or bulk loading.

Commercial databases may also have problems keeping pace with the *demand* for data, especially real-time data. In fact, real-time access to current data represents the most popular type of query among the user community (in terms of the number of separate queries). Although this has not yet proven to be necessary, the REINAS design allows for a cache of recent measurement data to be maintained in a *real-time cache* outside of the database. With such an extension, the dispatcher retrieves real-time data directly from this cache instead of the database, while continuing to provide seamless access to all REINAS data regardless of where it exists (in the database or in the cache).

5 Scientific Data Models

The REINAS approach to data management is dictated by the requirements imposed by all phases of environmental science. The fundamental design decision made for REINAS was to integrate the management of scientific data and metadata, rather than treat these data types separately (or ignore metadata) as is common in other scientific systems. Hence, REINAS developed an information model to structure the activities, interactions, and products relevant to the users and processes performed from experiment planning to data archiving, as well as a structure for the various types of sensor values. The model integrates the specification of *scientific data* such as measurements, sensor values, and other observations, with the specification of *scientific metadata*, including definitions for the content, representation, structure, and context of the scientific data.

Scientific data in general, and data streams in REINAS, can be broadly categorized into one of two distinct classes: environmental or *sensor-produced* data streams (also referred to as *measurement data* or sensor-streams), and meta-data or *data-about-the-data*. REINAS was designed to handle data of each type, as well as many relationships between them.

5.1 Environmental data streams

In its early stages, REINAS focused upon managing data from three specific types of instruments: surface meteorological (MET) stations, Coastal Ocean Dynamics Application Radars (CODARs), and vertical wind-profilers. These core instruments provided a rich set of environmental data-streams; MET stations generate multiple parallel streams of time-series data (scalar and vector valued), profilers produce vertical arrays of vector and related scalar data, while CODARs generate two-dimensional arrays of vectors organized either radially or, when combined from multiple sites, on a regular grid bounded by the local coastline.

The basic type of environmental data is a sensor value, which may comprise of an aggregate of several scalar values. When sampled temporally, the resulting *time-series* data represents the simplest data-type an environmental database system must be able to manipulate (Figure 3). REINAS treats time-series data as a fundamental data-type and handles a variety of different flavors; the most common are *meteorological data* from surface weather stations. Most meteorological sensors, such as those measuring air-temperature, humidity, or solar radiance, present a time-varying scalar quantity. Others, such as a wind-monitor, present a vector, i.e. $\langle$*wind-speed*, *wind-direction*$\rangle$.

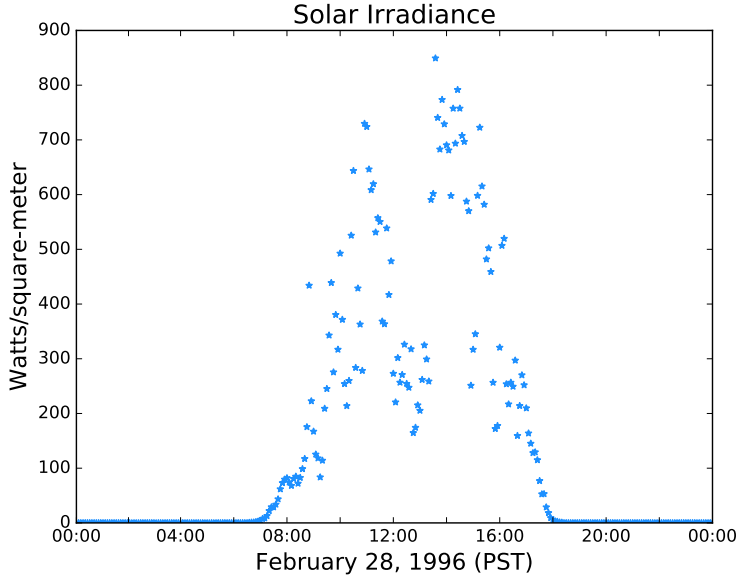

Figure 3: Sample scalar format data.

Doppler radar technology allows *wind-monitor*-type data to be indirectly measured in a vertical column of the atmosphere, at intervals of hundreds of meters to heights of several thousand meters. *Wind-profilers* thus produce a vertical profile of a tall column of air; an acoustic doppler current profiler can perform the analogous

measurement using sound for a water column. Hence, vertical profiles of wind or water currents, possibly augmented with other indirectly measured variables, and updated routinely over time, are also a basic environmental data stream. Such profiles are derived from doppler spectra, which may also be included in the database.

Another type of radar, the CODAR, indirectly measures ocean-currents along a collection of radials from the radar site, at fixed range intervals to distances of a few tens of kilometers. When CODAR *radial data* from two or more appropriately positioned CODAR sites are combined, *a field of ocean surface-current vectors* can be generated (Figure 4). Each radial or vector may also be accompanied with statistical confidence measures. A set of CODAR vector maps collected over time can be used to produce an animation illustrating ocean surface-current trends. Weather surveillance Doppler radars such as the Next Generation Weather Radar (NEXRAD) produce a

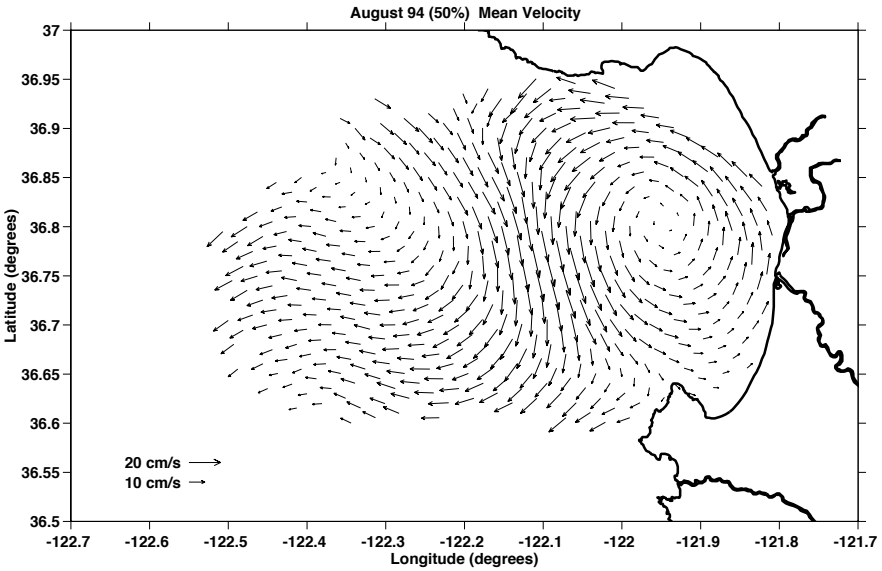

Figure 4: Sample derived CODAR vector data.

series of high resolution three-dimensional *volume scans* of the basic Doppler meteorological radar measurements [24]. Like wind-profilers and CODARs, NEXRAD systems produce data whose underlying structure is critically important to the dataset's effective use.

These four data sources require REINAS to manipulate scalar and vector data in point, linear, gridded, and higher-dimensional structures. Many other environmental data streams are similar in construction to one of the core types and can be incorporated fairly easily. REINAS has already been extended to include other interesting environmental data-streams, including multi-spectral satellite imagery, rawinsondes, and three-dimensional regional and global model outputs. In addition, work in progress will extend the system to include video streams from remote camera instruments (the prototype of which is already a popular feature on the World Wide Web [18]). Currently, REINAS accepts measurements from over 200 distinct instrument platforms comprised of over 1500 sensor streams located at about 170 fixed and mobile sites.

5.2 Schema

In order to implement a data model that successfully supports real-time forecasting activities and retrospective research, a schema was designed which attempts to strike an appropriate balance between the richness of the metadata and the ease with which that metadata can be managed on a day to day basis. The operational needs of REINAS require that the REINAS schema employ a data model able to support a growing, and unknown set of instruments, environmental models, and data sets, and that the operations required to add, activate, maintain, and deactivate these sources of data not be excessively complex. Since one of the fundamental goals of REINAS is to provide metadata sufficient to make sensor-produced data useful to scientists over time, the schema must preserve appropriate metadata to allow scientists to determine the history or *pedigree* of sensor-produced data when necessary.

Further, to accommodate the needs of environmental science, research, and forecasting, REINAS must support multiple data classes (numbers, sets of numbers, text, images, video, sound), structured versus unstructured data, multiple aggregation schemes for observations (point data, profile data, two and three-dimensional fields), and multiple sources (*in-situ*, remotely sensed, mobile platforms, historical files and databases).

The initial design to meet these requirements was based on the MBARI Scientific Information Model (MSIM) schema used to store environmental data for retrospective analysis at MBARI [10]. Since REINAS seeks to store metadata about instrument configuration and maintenance as well as store data required for retrospective analysis, MSIM was expanded to include a richer description of some metadata concepts. These design changes resulted in a schema of approximately 66 metadata tables plus containers for scientific data, with a rich and varied set of data abstractions for describing both data and metadata. This initial schema sought to provide real-time query performance by grouping scientific data of similar type within *containers* – special relational tables designed to simplify queries for similar data collected by one or more instruments in a given region. However, because of this schemas overall size and complexity, query performance was poor (especially for site-based queries) and restrictions on concurrent container access introduced unacceptable latencies for queries for real-time measurements. In addition, query development proved a difficult and time-consuming task for system and application programmers. The required level of detail in the metadata also made maintaining the schema very difficult, essentially requiring the attention of a full time database administrator willing and able to deal with metadata issues.

As a result of these difficulties, the first schema design was replaced in favor of one that places greater focus on operational convenience and real-time performance. This schema, originally termed the "REINAS Lite"

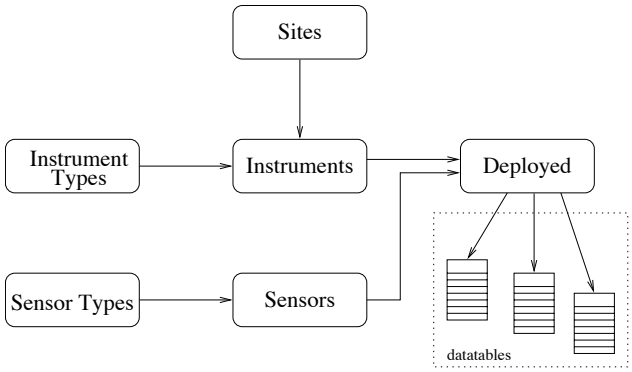

Figure 5: The REINAS LITE schema.

schema (Figure 5), seeks to best encapsulate the operations of creating and activating new instruments, and reduces the number of metadata tables to six, identifying *Sites*, *Instruments*, *Instrument Types*, *Sensors*, *Sensor Types* and *Deployments* as the essential metadata abstractions. To further improve the performance of queries for real-time, site, instrument, and sensor based measurements, the REINAS Lite schema also changes the basic model for organizing scientific data to favor the class of queries most often submitted to the database. This results in a schema which is significantly easier to maintain, and provides much improved performance, but which also sacrifices much of the metadata richness which REINAS originally sought to preserve.

Motivated by these metadata deficiencies in the REINAS Lite schema, a third schema, depicted in Figure 6, which benefits from each of the first two design experiences has been designed. It seeks to remain close to the simplicity of the REINAS Lite schema and borrows many of that schema's core concepts. At the same time, it also adds more of the metadata facilities for recording configuration management and establishing data pedigree that the first schema possessed.

To best meet the data management requirements set out for REINAS, the latest REINAS schema uses the simplifying assumption that any data collected by meteorological or oceanographic activities associated with REINAS can be described by metadata that falls into one of six broad categories:

1. Data collected and stored as a part of REINAS (temperature, humidity, and wind, for example) have a known *Measurement Type* (such as scalar, vector, or grid) and that such measurement types can be re-used to describe new sources of data as they are added to REINAS. Additional *Data Type* information can be associated with a measurement type to provide details about specific environmental parameters.

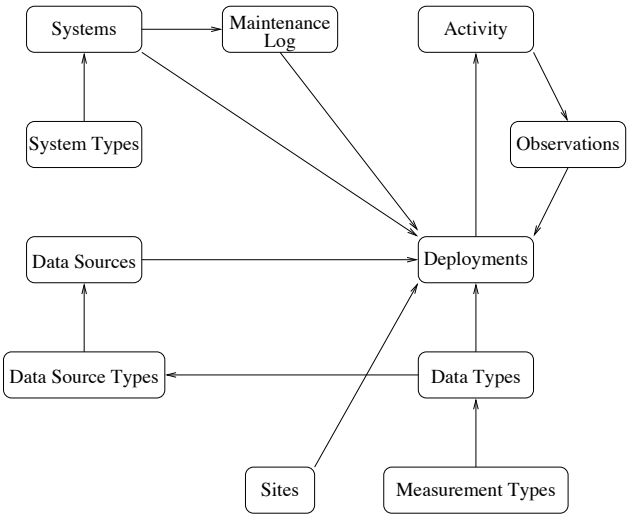

Figure 6: The *revised* REINAS schema.

2. The sources of REINAS data streams have associated with them a geographic location or *Site* which can be described as a small number of subtypes, such as points, lines, circles, rectangles, or time-dependent arbitrary data sets.
3. The sensors, instruments, data sets, calibration processes, and numerical models employed in the earth and marine sciences can be commonly described as *Data Sources* for purposes of identifying owner, manufacturer, and supplier (in the case of hardware) or process id, program name, and author (in the case of software). Classes of such sources have a common *Data Source Type* and a common data type.
4. For maintainance logging and configuration purposes, such data sources are part of, logically connected by, or physically co-located in a *System*, which itself is an instance of a specific *System Type*.
5. The human activities associated with the collection and storage of data in REINAS can be concisely described as an *Activity* with a person or group responsible for the data collection, and a set of free-form *Observations* used to annotate the data collected.
6. A Data Source is operated as a *Deployment* which has a data type, is part of a system, is located at a site, and is part of an activity. The data from each deployment is stored in a specified data table.

By expanding each of these categories using at most three or four database relations, it is possible to create a schema of sufficient complexity to describe environmental and scientific data in general while allowing the meaning of relational links and location of metadata to remain understandable by those operating the system.

The REINAS database defined by these metadata relations is implemented as a single, self describing schema to which load paths from all sources are directed through the *Deployment* table. Standard templates are used for developing and reusing load path software based on a source's associated *Measurement Type*. This template links with the schema, capturing the necessary metadata for each *Source Type* occurs only once, when the first source of that type is added to the system. More importantly, all information required for fusing data from multiple sources is retrievable using geographic location, data type, source type, system, or human activity associated with the data itself as keys. The schema supports the translation of logical concepts such as derived data products directly into concrete hierarchies of *Data Sources*, complex projects as groups of simpler *Activities*, and the logical connection of data streams as *Systems*. The association of *Deployment* with *Data Source* and *Data Type* means that data can be stored in differing formats, and the association of *Deployment* and *System* configuration and maintenance information supports long term data management and data pedigree.

6 Output Products

Data residing within REINAS can be retrieved and used in a variety of ways. Although queries for current-conditions tend to dominate, fairly complex queries are also possible. For some purposes, a *Structured Query*

Language (SQL) [5, 12] interface for submitting queries is sufficient. In general, however, users and user-level applications require a more abstract interface for requesting and receiving data from REINAS.

6.1 Query types

The most popular type of query issued by users is a request for current or real-time sensor values. Often these queries pertain to a specific instrument site and hence they have been labeled *snapshot queries*. Snapshot queries may return one or multiple sensor values, should be relatively easy to submit, and should return results with low latency, particularly when returning real-time data.

It is also possible to query over a broader set of environmental data. For example, one can ask for a plot of the previous hour's mean ocean surface current speeds at every CODAR vector grid-point when the air-temperature at the Monterey Bay Aquarium dropped below a previously recorded minimum air-temperature for the region. Such complex queries may involve manipulating and returning large amounts of scientific data.

REINAS queries may involve querying non-scientific data, and the ability to support such queries has been one of the main goals of the project. Being able to query for the ten year old calibration of a particular temperature probe may be essential for a researcher studying long term trends. Such queries return data about the collection process itself, which helps in the assessment of the validity or uncertainty of the data, and also makes it possible to compare data collected in different times, with different processes, and with different instruments.

6.2 API

Collections of data residing in a well-designed, well-maintained database have little value unless they can be easily retrieved by user-level applications. It is also impossible to exhaustively predetermine the scope of useful applications that may operate on such data collections. As a result, a predefined interface capable of supporting an extensible set of query types was required to insure the utility of data contained within REINAS.

Initially, only one such *application-programmer-interface* existed within REINAS. A low level *REINAS-API* was constructed essentially to provide a generic framework for issuing SQL-level operations against the database. It was quickly realized, however, that only a fairly specialized group of individuals could effectively use this API; its primary drawback was that the application programmer must understand not only SQL but also the REINAS schema. These requirements are unreasonable in a system where the typical application-programmers are environmental scientists with little knowledge of database technology.

To provide better application-development support to users, a higher level API that buffers the client application programmer from the REINAS schema and SQL semantics was designed and implemented within the dispatcher through a set of predefined or "canned" queries. Using a subset of the abstractions defined by the REINAS schema, the dispatcher's canned queries provide the client programmer with enough functionality and flexibility that knowledge of SQL and the underlying schema can be avoided. The set of predefined queries includes requests to retrieve sensor data in real-time or retrospectively, and at a given time or over a time-interval. Sites, instruments, instrument types, sensors, sensor types, and deployments can all be listed, selected, or inverted from a corresponding identifier. The geographical position of a site, instrument, or deployed sensor can also be mapped from any related entity. Other more advanced canned queries are also available. The dispatcher also predefines a specific return format for all possible query results, allowing the application programmer to anticipate the structure of query outputs and to write simple yet general code to parse a variety of different query products. The dispatcher also includes mechanism that allow SQL queries to be "passed through" it to the database, with the results provided to client application using the same standard interface with which canned query results are returned.

Although the dispatcher API has proven to be general, flexible, and yet simple enough to be used by environmental scientists who are not intimately familiar with the REINAS schema or relational database concepts, an additional higher-level API has also been developed within the REINAS effort to facilitate the transport of generic environmental data between a database system like REINAS and powerful multi-dimensional visualization applications which must manage and manipulate large amounts of scientific data of varying structure and size. *RSObjects* is an object oriented API implemented within the C programming language that has been developed to provide such an interface to C and FORTRAN programmers [26]. RSObjects was developed following several other API development efforts with earlier prototype systems. These APIs were difficult to extend, failed to isolate client applications from changes within the dispatcher and database, and were unable to support large objects (among other failures). In order to mitigate these problems, RSObjects provides a dialogue model of interaction where the application asks the database what it may request. In this way the applications maintain backwards compatibility,

while also providing opportunity for querying future instruments and views without requiring recompilation. In this way, RSObjects improves upon traditional interfaces by providing a higher level encapsulation with a reduced and simplified conceptualization in order to hide the underlying schema and other implementations details from the application writer. RSObjects includes both a formal API definition as well as a collection of utility routines that provide the means for safe dynamic memory usage, common query definitions, and convenient organizations of environmental data for visualization access.

6.3 User level applications

To date, REINAS has developed many retrieval applications: generic monitoring tools using conventional glyphs and time-series plots, contour, station plots, and overlaid satellite data, and three-dimensional iso-surface, streamlines, glyphs, and bump map renderings.

The three-dimensional rendered visualizations have provided a new insight into various aspects of the environmental data being archived in REINAS, and have also been the stimulus for several novel visualization methods [14, 25]. These visualizations are not used by the majority of REINAS users, as few REINAS users have direct access to Silicon Graphics or other accelerated rendering platforms necessary for supporting advanced visualization applications. However, the World Wide Web (WWW) provides a mechanism accessible to almost all users and with adequate graphics capabilities to allow for the display of query results from a large number of the scientific data-types present within REINAS. To date, via the WWW, REINAS has developed a following of thousands of users worldwide, many of whom have come to depend on the simple time-series and vector plots that provide real-time regional meteorological and oceanographic conditions [17]. To date, over 2.4 million separate visits have been recorded to the prototype video platform interface [18] which also displays real-time conditions from a nearby REINAS weather station. These web pages support many casual users, but also provide an effective mechanism for data access for many of the collaborators, scientists, and professional meteorologists using REINAS.

7 Conclusions

The initial REINAS proposal written in 1991 described a system architecture based on an assessment of available computer technology and a forecast of developments to be expected in the following years. The proposal included the concept of extending instrument sites into Internet nodes using a local 80x86 class computer or a "REINAS/PC", providing local parsing, processing, and buffering. Now implemented and revised into a mature platform, the REINAS/PC concept has been demonstrated to be a viable and powerful architecture, providing modularity, extensibility, and operational flexibility. Relatively powerful embedded 80486 and Pentium class systems are already available as affordable hardware platforms for extending Unix and the Internet to remote regions where power constraints and environmental factors require an emphasis on hardware and software reliability. The Internet Protocol has already been demonstrated to be a robust and flexible foundation for building a distributed regional network using a variety of technologies (e.g. wireless, serial, Ethernet, ATM), while local buffering and archiving provide insurance against data loss, even in the event of extended network (or remote database) failures. Since this architecture was first developed, microprocessor and related technology costs have declined significantly, and the development of an even cheaper, single-board REINAS/PC with embedded networking options, digital signal processing and input/output capability, and internal static storage is a promising avenue for future research.

Two closely related and key decisions in the development of REINAS were the use of commercially available relational-database systems to support data capture, retrieval and long-term management, as well as the decision to use a *single* database system to support both real-time and retrospective uses of the data. Results from the use and development of REINAS to date indicate that this was indeed a correct decision, provided careful consideration is given to performance constraints during schema design and system implementation.

The remote instrument and stream logs have added reliability and buffering to address the "real-time problem" of data input, supporting an insertion rate driven by continuous data feeds from the instruments. Query support within the dispatcher and the RSObjects API provide structured access to all data in the database in a logical and consistent manner. With the growing use of the REINAS system from remote Internet users via the World Wide Web, most of the queries focus on real-time and recent data. This heavy real-time interest was not anticipated in the original REINAS proposal (which predicted instead a set of primarily scientific users with different, longer-term access requirements) and refinements in the system were necessary to enhance real-time query performance to better support the large number of ad-hoc users of the system. The challenge here is to overcome

the inherent conflict in providing acceptable performance to interactive users with a database schema developed to support scientific users engaged in retrospective analysis. Additional indexing of the data, duplication of certain datatables, and other database tuning efforts continue to be investigated to improve interactive query performance.

Acknowledgements

We would like to acknowledge the efforts of some of the students who have been active developers in the REINAS system, including: Stephen Carter, Ted Dunn, Suzana Djurcilov, Bruce Montague, Carles Pi-Sunyer and Bryan Green. We would also like to thank the many colleagues involved in this collaborative effort, especially our science colleagues Professor Wendell Nuss, Dr. Dan Fernandez, Bruce Gritton, and Professor Jeff Paduan. The REINAS visualization team, led by Professor Alex Pang and including Dr. Naim Alper, Jeff Furman, Zoe Wood, Elijah Saxon, Jonathan Gibbs, and Michael Clifton, have developed a suite of successful high-end visualization applications tightly integrated with REINAS. In addition, John Wiederhold, Catherine Tornabene, Sume Baik and Amy Graf helped develop the network of Monterey Bay regional instrumentation that attracts most of our users.

References

[1] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz. Improving TCP/IP performance over wireless networks. Nov. 1995.

[2] T. Bernath. Distributed GIS visualization system. In *Proceedings of GIS/LIS International Conference*, volume 1, pp. 51–58, San Jose, Ca, Nov. 1992. American Society for Photogrammetry and Remote Sensing.

[3] A. Berrill and G. Moon. An object oriented approach to an integrated GIS system. In *Proceedings of GIS/LIS International Conference*, volume 1, pp. 59–63, San Jose, Ca, Nov. 1992. American Society for Photogrammetry and Remote Sensing.

[4] R. S. Cerveny, S. M. Calderon, M. W. Franjevic, and N. C. Hoffmann. Development of a real-time interactive storm-monitoring program in Phoenix, Arizona. *Bulletin of the American Meteorological Society*, 73(6):773–779, June 1992.

[5] C. J. Date and H. Darwen. *A guide to the SQL Standard: a user's guide to the standard relational language SQL*. Addison-Wesley Longman, Massachusetts, third edition, 1993.

[6] S. Howes. Use of satellite and radar images in operational precipitation nowcasting. *Journal of the British Interplanetary Society, Journal*, 41(10):455–460, Oct. 1988.

[7] 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, May 1990.

[8] D. Johansen. StormCast: Yet another exercise in distributed computing. *Distributed Open Systems in Perspective*, Feb. 1993.

[9] J. P. Lee and G. G. Grinstein. Database Issues for Data Visualization. In *Proceedings of the IEEE Visualization Workshop*. Springer Verlag, 1993.

[10] D. D. E. Long, P. E. Mantey, E. C. Rosen, C. M. Wittenbrink, and B. Gritton. REINAS: A real-time system for managing environmental data. In *Proceedings of the Eighth Software Engineering and Knowledge Engineering Conference (SEKE 1996)*, pp. 293–300, Lake Tohoe, June 1996. SEKE.

[11] D. D. E. Long, P. E. Mantey, C. M. Wittenbrink, T. R. Haining, and B. R. Montague. REINAS: the real-time environmental information network and analysis system. In *Proceedings of the 1995 IEEE Computer Society Compcon Conference*, pp. 482–487, San Francisco, Mar. 1995. IEEE.

[12] R. A. Lorie. *SQL and its applications*. Prentice-Hall, Englewood Cliffs, NJ, 1991.

[13] P. Mantey. REINAS: Real-time environmental information network and analysis system: Phase IV - experimentation. Technical Report UCSC-CRL-94-43, Computer Engineering and Computer Science, University of California, Santa Cruz, 1994.

[14] A. Pang. Spray rendering. *IEEE Computer Graphics and Applications*, 14(5):57–63, 1994.

[15] J. B. Postel. Transmission Control Protocol. Technical Report RFC-793.

[16] C. Ramanathan. Providing object-oriented access to a relational database. In *Proceedings of the 32nd Annual Southeast Conference*, pp. 162–165, New York, NY, Mar. 1994. ACM.

[17] E. C. Rosen. REINAS home page. https://www.soe.ucsc.edu/research/, 1995. Computer Engineering, University of California, Santa Cruz.

[18] E. C. Rosen. SlugVideo home page. http://www.cse.ucsc.edu/research/, 1995. Computer Engineering, University of California, Santa Cruz.

[19] L. A. Rowe and M. R. Stonebraker. The Postgres Data Model. In *Proceedings of the 13th Conference on Very Large Data Bases*, pp. 83–96, Brighton, England, September 1987.

[20] D. J. Schwab and K. W. Bedford. Initial implementation of the great lakes forecasting system: A real-time system for predicting lake circulation and thermal structure. *Water Pollution Research Journal of Canada*, 29(2-3):203–220, 1994.

[21] M. Stonebraker. The design of the Postgres storage system. In *Proceedings of the 13th Conference on Very Large Data Bases*, pp. 289–300, Brighton, England, September 1987.

[22] M. Stonebraker. An overview of the SEQUOIA 2000 project. In *Proceedings of the 37th IEEE Computer Society International Conference Digest of Papers: COMPCON Spring 1992*. IEEE Computer Society Press, Feb. 1992.

[23] M. Stonebraker. Sequoia 2000: A reflection on the first three years. In *Proceedings of the Seventh International Working Conference on Scientific and Statistical Database Management*, pp. 108–116, 1994.

[24] R. L. A. Timothy D. Crum and D. W. Burgess. Recording, archiving, and using wsr-88d data. *Bulletin of the American Meteorological Society*, 74(4):645–653, 1993.

[25] C. M. Wittenbrink, G. G. Langdon Jr., and G. Fernandez. Feature extraction of clouds from GOES satellite data for integrated model measurement visualization. In *Proceedings of the IS and T/SPIE Symposium on Electronic Imagining: Image and Video Processing 1996*, San Jose, CA, Jan. 1996.

[26] C. M. Wittenbrink, E. Rosen, A. Pang, S. K. Lodha, and P. Mantey. Realtime database support for environmental visualization. In *Proceedings of the Second Workshop on Database Issues for Data Visualization*, pp. 111–130, Atlanta, GA, Oct. 1995. IEEE.